12-2019

Flex-Ro: A Robotic High Throughput Field Phenotyping System

Joshua N. Murman
University of Nebraska-Lincoln, josh.murman@huskers.unl.edu

FLEX-RO: A ROBOTIC HIGH THROUGHPUT FIELD PHENOTYPING SYSTEM

by

Joshua Nathanael Murman

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska-Lincoln

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Agricultural and Biological Systems Engineering

Under of the Supervision of Professor Santosh K. Pitla

Lincoln, Nebraska

December 2019

FLEX-RO: A ROBOTIC HIGH THROUGHPUT FIELD PHENOTYPING SYSTEM

Joshua Nathanael Murman, M.S.

University of Nebraska, 2019

Advisor: Santosh K. Pitla

Research in agriculture is critical to developing techniques to meet the world's demand for food, fuel, fiber, and feed. Optimization of crop production per unit of land requires scientists across disciplines to collaborate and investigate new areas of science and tools for data collection. The use of robotics has been adopted in several industries to supplement labor, and accurately perform repetitious tasks. However, the use of autonomous robots in commercial agricultural production is still limited. The Flex-Ro (Flexible structured Robotic platform) was developed for use in large area fields as a multipurpose tool to perform monotonous agricultural tasks.

This work presents the design and implementation of the control system for the Flex-Ro machine. The machine control architecture was developed for safe operation with redundant emergency stops and checks. Operators use the remote-control device to maneuver the machine in uncontrolled environments. Autonomous field coverage was developed using global positioning system (GPS) guidance. The guidance system tracked within 4 cm of the guidance line 95% of the time at a travel speed of 4 kph. Waypoint guidance was implemented and demonstrated such that Flex-Ro could be programmed to follow complex paths and curves.

High-throughput plant phenotyping is a continuously developing and evolving field of plant science. The methods used to collect phenotyping data include drones, satellites, manual measurement, and ground rovers. A suite of phenotyping sensors was installed onto the Flex-Ro to cover large field areas. The system was verified in soybean research plots at the University of Nebraska-Lincoln (UNL) Spidercam phenotyping facility. Positive correlations between the Spidercam and Flex-Ro phenotyping data were established. The Flex-Ro was able to statistically distinguish between soybean variety emergence and maturity differences. The late season phenotyping data showed statistical differences between the fully irrigated versus deficit plots. Basic economic calculations estimated the cost to operate the Flex-Ro machine for field phenotyping use at approximately $5.50/ha.

Acknowledgments

I would first like to thank my advisor, Dr. Santosh K. Pitla, for encouraging me to take this next educational step. Thank-you for your guidance and mentorship through this process.

Thank-you to my graduate committee members Dr. Yufeng Ge and Dr. Joe D. Luck for their support.

Thank-you to the University of Nebraska Agricultural Research Division for support and use of Spidercam field space to test Flex-Ro. Thank-you to David Scoby and Geng 'Frank' Bai for collecting and processing the Spidercam phenotyping data.

Thank-you to the in-kind donors to the Flex-Ro machine; Kubota Engine America supported by Anderson Industrial Engines, ifm efector inc., and Danfoss Power Solutions.

Finally, thank-you to my parents, Craig and Deb, for always supporting me and for teaching and continuously exemplifying to me the values of faith, work ethic, and character.

Table of Contents

Table of Figures

Table of Tables

Table of Equations

Chapter 1 Introduction

The agricultural industry supports the world's needs for food, feed, fiber and fuel. Global economic and population growths are projected to increase by over 40 percent by 2050 (Bruinsma, 2009; USDA, 2019). This translates to a corresponding increase in global agricultural production by 70 percent. There are two ways to increase crop production, more yield per area or expansion of farmable land (Bruinsma, 2009). During this period of growth, the planted acres within the United States is projected to remain steady (USDA, 2019). Plateaued commodity prices with increasing input costs are driving thin margins, and limited land requires increased productivity per area. As a result, continuous research on optimization of resources is paramount to the success of modern farming operations.

1.1 Research in Agriculture

Agricultural research supports the development of efficient crop production systems. Research institutions receive grants to support work investigating cause and effect relationships across all aspects of the agriculture industry. The findings are presented to the public via extension outreach of the universities, allowing the producers to implement discoveries.

Farmers must balance costs to benefits to maximize production while maintaining profitable operation. Costs incurred during the growing season include tillage, nutrient application, seed, pesticides, herbicides, irrigation, and harvesting operations. Each one

of these items affects the output or yield of the crop. Research is conducted across all aspects of the farming operation, which seeks to draw correlations between variable inputs to outputs. Scientists from many disciplines find applicable research questions in the agricultural industry. An abridged list includes soil scientists, agronomists, economists, engineers, entomologists, geneticists, plant breeders, statisticians, traders, financial analysts and computer scientists. Each of these stakeholders hold a position within the agricultural value ecosystem and can benefit from effective research.

There are two desired outcomes for a successful research program related to agricultural production. First is increased yield (revenue) and the second is reduced inputs (cost). Management practices must balance revenue with costs to remain profitable. For example, excessive nutrient application would increase crop output; however, the increased costs may not be recouped with proportional yield gain (Cassman, 1999). There are non-financial implications to farming management practices also. Environmental concerns from chemical misuse is one example. Successful agricultural research ideally benefits all stakeholders of the agricultural value chain. One particularly important subset of agricultural research is the development of desirable plant characteristics.

1.1.1 Plant Breeding

Plant breeding is the method of developing crops to achieve desirable characteristics (Atefi, 2019). The current rate of increased plant productivity must continue to rise to meet the demands of the world (Araus and Cairns, 2014). Plant breeding targets increasing yield and key traits for harvestability and marketability (Fehr, 1991). For

example, a corn plant should have a high yield with strong stalks, deep roots and disease resistance. Plants resistant to insects and disease require fewer pesticides and plant traits for drought tolerance are desirable in many locations.

Plant breeding success relies on both qualitative and quantitative data. Traditionally, the breeders developed plants by visually selecting the best from each generation (Fehr, 1991). This process would be repeated several times until the desired output was achieved. New methods in genetics provide ways to accelerate breeding progress, and better target desired traits (Fahlgren et al., 2015; Fehr, 1991).

Accelerating the progress of plant genetic development is critical to meeting the world demand for increased production of food, feed, fiber, and fuel. Even with the science of molecular breeding, rapid characterization of a plant's physical response given its genotype to an environment is still limited (Atefi, 2019). Objective quantification and qualification of phenotypic data is crucial to developing plants with the most desirable traits (Fahlgren et al., 2015).

1.1.2 Phenotyping

Phenotyping is the characterization of a plant's physical and performance related traits (Dhondt et al., 2013). Plant breeders collect this data for genotypes in specific environments. Leaf area index, leaf number, canopy temperature, water content, vegetative indices, canopy coverage, and stem diameter are examples of physical traits measured or calculated. The environments may be controlled, uncontrolled or measured (Dhondt et al., 2013).

Plants respond uniquely to different environmental conditions (Atefi, 2019). Plant breeders and production farmers can leverage early prediction of crop output. Plant breeders use early season phenotype data to draw preliminary conclusions on a genotype to begin developing the next generation (White et al., 2012). Yield relationships to early season phenotyping data can be statistically established. With this method, farmers marketing crop futures contracts would have a better estimate of yields and total production.

Development of reliable correlations of a plant phenotype to environmental conditions and genotypes requires extensive datasets. Long term studies are often conducted in semi-controlled research fields or controlled greenhouses with installed phenotyping systems (Foix et al., 2015). Destructive methods of measuring plant characteristics have previously been used, but limit the temporal data collection (Furbank and Tester, 2011). Large sample sizes are needed to achieve representative growth curves of a genotype. Non-destructive phenotyping uses sensors and imaging techniques to directly measure or capture data which can be used to calculate plant characteristics. These sensors can be mounted to devices for high-throughput data collection. Advancements in computational processing capacity have enabled rapid phenotyping of large populations. Current high-throughput techniques maintain high correlation to ground truth measurements (Bai et al., 2016).

Incoporating high throughput phenotyping into the plant breeding cycle will facilitate the increase in crop productivity needed to match global demands (Furbank and Tester, 2011). Phenotyping research is being developed on resolutions from plant to field level.

Each scale has benefit to plant breeders and crop consultants who make management decsisons based on the current state of the crop. Shakoor et al., (2017) illustrated the plant breeding cycle and scales Figure 1.1.

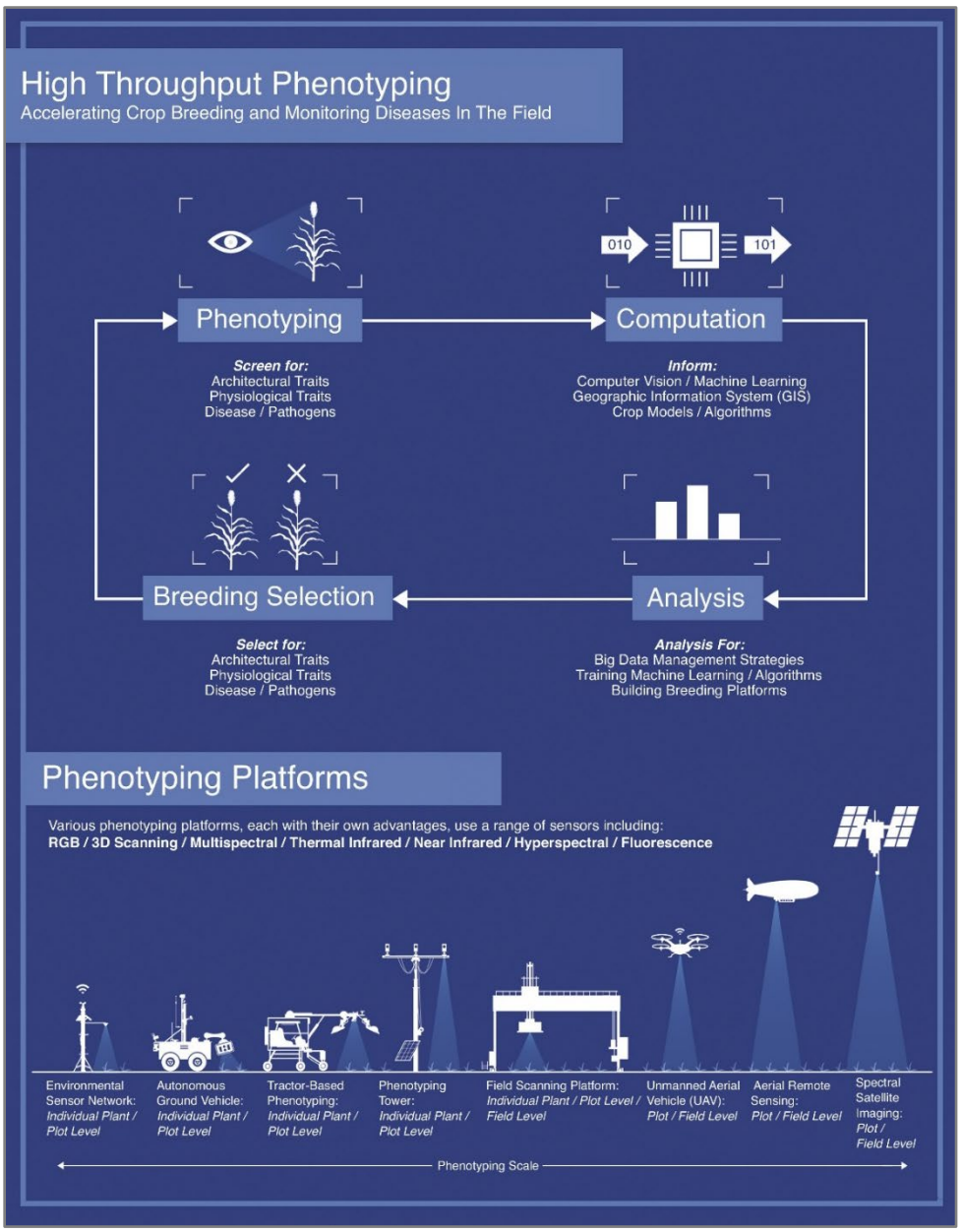


Figure 1.1: Phenotyping and crop-breeding cycle. Scale and resolution of developed phenotyping platforms (Source: Shakoor et al., 2017).

1.2 Use of Technology in Agriculture

Technology has been continuously incorporated into agriculture to improve the production system. Mechanization from horse to tractor revolutionized the farming industry. Machine features became more complex as technology continued to develop, and operating stations were increasingly designed for ergonomics and comfort. Electronic incorporation into agricultural machinery began with the release of a planting population monitor by DICKEY-john (Stone et al., 2008). Serial communication was first used to simplify connections to implements. Progress towards standardizing communication on a machine controller area network (CAN) bus began in the 1980's (Stone et al., 2008). A standardized high-level CAN protocol (message format) allowed for the continued development of agricultural technologies and paved the way for brand agnostic devices.

1.2.1 The Rise of CAN bus

CAN bus technology for off-highway machinery led to the development of complex machinery systems. Multiple electronic control units (ECUs) were used to control the subsystems of machine. Electronic displays and switches in the cab required communication with the ECUs. The Society of Automotive Engineers (SAE) and American Society of Agricultural Engineers (ASAE) jointly developed SAE J1939 as a response to the need to standardize communication protocol on off-highway machinery (Marx, 2015). J1939 defined this high-level message structure (application layer) for communication on the two wire twisted pair CAN bus (physical layer). Control, interface, and diagnostic messages were defined within the standard's application layer. Processor

advancement preceded development of the virtual terminal (VT) which integrated machine and implement controls onto a user interface display (Stone et al., 2008). As global position system (GPS) accuracy continued to improve, the use of the VT expanded to include automatic steering applications (Buick, 2006).

1.2.2 Navigation Systems

GPS technology became prevalent on agricultural machinery first with the implementation of precision mapping, and later automatic steering control. The turn of the century led to rapid advances in GPS hardware development and accuracy (O'Connor, 1997). Research by O'Connor (1997) and Bell (1999) developed steering control systems based on GPS location. As time progressed, GPS hardware became more accessible and overall decreased cost of systems led to a shortened return on investment time (Buick, 2006). Automatic guidance improved field coverage efficiency by reducing overlap. Modern navigation systems have repeated accuracy of +/- 2.5 cm by using real-time-kinematic (RTK) corrections for the GPS signal (Baillie et al., 2018).

The development of automatic navigation control systems led to the delivery of other related operations. Automatic swath guidance has been extended to provide headland turn coverage (Baillie et al., 2018). Total machine automation controls the tractor and implement through the turn, disengaging and restarting the operation on the next swath. Machine cooperation (e.g. leader and follower) technologies have been developed as a progression of automatic navigation (Thomasson et al., 2018). The current state-of-the-art technologies are operator assisted automation, or level 3 (out of 5) automation as defined

by Case IH (CNH Industrial America LLC, Burr Ridge, IL). The operator must remain in the cab ready to resume control in case of unexpected events or encounters (Case IH, 2018). Future development towards full autonomy will need to include advanced path planning and obstacle detection and avoidance (Baillie et al., 2018; Bell, 1999).

1.3 Robotics in Agriculture

The use of robotics in agriculture, while still commercially limited, is seeing rapid development (McAllister et al., 2019). Robots are designed to relieve operators of long working days and reduce overall manual labor (Werner, 2016). The use of robotics in precision agriculture increases management resolution by working unattended for long hours. Further, a smaller size compared to traditional machinery reduces soil compaction (Godoy et al., 2012).

Agricultural robots have been developed in several configurations. The use of battery power is common for smaller scale platforms (Bak and Jakobsen, 2004; Bangert et al., 2015; Griepentrog et al., 2012; Slaughter et al., 2008). However, sole electric power has runtime limitations due to the required time to charge (Werner, 2016). Internal combustion robotic platforms have also been developed for agricultural use (Godoy et al., 2012; Werner, 2016). Petroleum powered robots have the advantage of long run times paired with short refueling periods. However, a combustion engine requires increased maintenance compared to an electric drivetrain. In either case, digital systems must be able to control all aspects of the vehicle.

1.3.1 Agricultural Robotic Control Systems

The development of autonomous agricultural robots includes research on control system methodologies. Robotic control systems are developed similar to subsystems implemented on machinery (Troyer, 2017). Autonomous operation consists of four main stages. A machine must start with route planning. This may be present as algorithms on the machine or be pre-defined and uploaded. Coverage strategies are optimized for maximum field efficiency (J. Jin and L. Tang, 2010). The route is then augmented with environment data during operation, most commonly to avoid obstacles. After the current route is accepted, the trajectory and speed of the machine is determined. Finally, local feedback control manages the actuators of the robot to the desired operating state (Paden et al., 2016).

Robotic steering controllers are designed from a kinematic or dynamic model of the machine. Kinematic models, while less computationally expensive, are limited to slower speed operation (Bell, 1999). Advanced dynamic control methodologies can improve performance on machines which encounter a lot of variability (Uzunsoy, 2018). Different implements, payloads, and operating speeds contribute to steering controller performance (Lakkad, 2004). Simulations are used to verify controller functionality and test different scenarios without the need for the physical machine (Lakkad, 2004; Tu, 2013). The control system must compensate for variations in terrain to track the navigation line (Cariou et al., 2009).

Robots operating in uncontrolled environments may encounter obstacles at any time. Obstacle detecting sensors are installed in order to reduce the likelihood of collisions. (Emmi et al., 2014). Several methods of obstacle detection have been researched and evaluated. Methods include LiDAR's (Biber et al., n.d.) infrared (IR) sensor (Pitla et al., 2010a) and lasers (Oftadeh et al., 2013). Accurate detection and classification of obstacles within the field environment will be important for large scale deployment.

Robotics in production agriculture are likely to manifest as several small robots operating in cooperation (Emmi et al., 2014; McAllister et al., 2019; Pitla et al., 2010b). Modular robots would be added depending on the need of the operation (Emmi et al., 2014). Swarm control architecture depends on the task. Equal distribution of work is well suited to seeding type applications, and leader-follower architecture is more suited to harvest operations (Pitla et al., 2010b).

Substantial amounts of data must be transferred between the subsystems of the robot and between the units in the swarm. CAN bus communication provides a method for handling messages within the on machine network (Baek et al., 2008). Communication between robots within the field will facilitate job coordination (Pitla et al., 2010b). Robust network systems and relaying information to the master controller will allow the robots to be adaptable wide variety of applications.

1.3.2 Ag. Robotic Applications

There are many applications in agriculture which are well suited to robotics. The first adaptation will replace labor intensive repetitive tasks (Emmi et al., 2014). Robots

currently developed are low power and designed for non-ground engaging activities. These include targeted mechanical weeding (Åstrand and Baerveldt, 2002), precision spraying (Bangert et al., 2015), and crop scouting (Bangert et al., 2015; Shafiekhani et al., 2017)

Slaughter (2008) authored a state-of-the-art review of robotic weeding technologies. Several different methods were described as ways autonomous rovers managed weeds. Since then commercialized technologies have been developed. EcoRobtix (ecoRobotix ltd, Yverdon-les-Bains, Switzerland) Naïo Technologies (naïo Technologies Escalquens, France) and FarmWise (FarmWise Labs, Inc. San Francisco, CA) are all examples of autonomous weeding prototypes which appear available in the commercial sector.

Crop scouting traditionally is completed by a trained agronomist. The agronomist must balance productivity with resolution of field coverage. Agronomists data supplements producers in decision making about crop inputs and applications. Plant phenotyping uses scouting data to draw correlations to a genotype given the measured or controlled environment.

1.3.3 Ag Robotic Phenotyping

Phenotypic data collection is laborious. Several concepts and prototypes have been developed and implemented to facilitate high-throughput phenotyping. Phenotyping platforms vary in scale and resolution as seen previously in Figure 1.1. Data captured from unmanned aerial vehicles and satellites is used to measure broad areas on plot and

field level resolution. Ground vehicles and devices collect higher resolution data, from plot to individual plant (Shakoor et al., 2017).

High throughput ground based phenotyping platforms were first developed on manual push carts. White and Conley (2013) and Bai et al. (2016) instrumented carts to measure plot level phenotypic traits. The use of carts enabled multiple sensor mounting configurations and cover more area as a result. A stop-measure-go technique was used for plot coverage and is well suited to manual operation of the cart (Bai et al., 2016). Strong correlations were established to ground truth measurements to prove the viability of the cart phenotyping system.

The development of phenotyping carts enabled faster coverage compared to handheld devices and a higher resolution than UAVs. However, pushing the cart and manually triggering data collection required a full-time technician. Several self-propelled phenotyping devices have been developed. Andrade-Sanchez et al., (2014) developed a manually driven high clearance phenotyping platform. The machine was easily adaptable to a variety of sensors and was not limited by payload capacity. Shafiekhani et al. (2017) developed Vinobot which was a smaller scale autonomous platform to collect phenotypic data on research plots. Bangert et al. (2015) implemented a phenotyping application onto the BoniRob autonomous robot.

Space in the phenotyping field exists for a high-resolution, high-throughput platform for high-acreage applications. Suites of sensors have been proven to show correlations to ground truth phenotyping measurements. Manual and self-propelled platforms are tied to

operators which inherently limit coverage and productivity. A truly autonomous platform would collect or stream data over large coverage areas and allow researchers and crop consultants to make informed decisions.

1.4 Conclusions

Considerable progress has been made on robotic systems for use in agriculture. Researchers have developed robotic control systems and many applications, specifically in the plant phenotyping community. However, there is a lack of synthesized machines which are field ready, especially for large acreage applications. Researches will be able to use this high coverage data to facilitate new science on the productivity of commercial agriculture.

1.5 Thesis Objectives

The aim of this thesis is to continue the development of the Flex-Ro platform developed by Werner (2016). A field capable research platform for phenotyping is proposed as the first use case for the Flex-Ro platform. Five objectives have been outlined as follows:

1. Develop and verify a redundant safety stop system to stop potential unintended machine motion.
2. Autonomously navigate between 30in crop rows and complete headland turns resulting in supervised autonomous field coverage.
3. Follow a preset waypoint path to facilitate go-to-start and return-to-home applications.

4. Implement obstacle detection to react to small sized objects during field operation.
5. Compare phenotyping data collected with the Flex-Ro PhenoBar to the ground truth data collected using the UNL Spidercam facility.

1.5.1 Thesis Hypothesis

1. The Flex-Ro phenotyping data collected will directly correlate with the measurements taken by the Spidercam phenotyping utility.
2. The Flex-Ro phenotyping data will reveal with statistical significance the difference between two treatments within the research field.

Chapter 2 Flex-Ro Control System Development and Implementation

2.1 Introduction

The Flex-Ro machine was designed for large area field operations. The design of a control system for a machine depends on the environment in which it must operate. Constraints are established based on the parameters of the task. Agricultural fields are semi-controlled environments with limited access. Knowledge about the field before operation could include information related to boundaries, crop-row placement, and internal obstacles or hazards. Traditionally, machine operators react to unforeseen circumstances such as obstacles and adverse field conditions. Robotic machines must be able to programmatically manage unexpected circumstances while finishing the task assigned.

Coverage of a row crop field requires three basic operations. The machine must first be positioned at the starting swath. Then the robot needs to navigate between the swath rows, without damaging the crop. The headland area is either made up of crop rows perpendicular to the swath or open space and the robot must use this headland space to continue into the next swath. This process is repeated until the field has been completely traversed. Finally, the robot must continue to a staging area where it can be picked up.

Objectives for the Flex-Ro control system were extracted from the requirements for basic field coverage. This also includes reaction to obstacles during field operation. Obstacles

in scope include pedestrian sized objects, but not holes and washouts. The Flex-Ro platform must also facilitate manual operation. Teleoperation via remote control was required for maneuvering the machine to its storage location, loading of the machine onto a trailer, and initializing the machine coverage at the field.

2.1.1 Chapter Objectives

1. Develop and verify a redundant safety stop system to stop potential unintended machine motion.
2. Autonomously navigate between 30 in. crop rows and complete headland turns resulting in supervised autonomous field coverage.
3. Follow preset waypoint path to facilitate go-to-start and return-to-home applications.
4. Implement obstacle detection to react to pedestrian sized objects during field operation.

2.2 Materials and Methods

2.2.1 Control System Hardware

Several different components make up the Flex-Ro control system. The four main machine subsystems were the engine, hydrostatic drive, steering, and human machine interface. Digital electronic controls were required for the machine to be operated programmatically. The electronic control units (ECUs) were linked via controller area network CAN bus. Each of the subsystem controllers required compatibility with the

CAN communication. The two operation modes, manual and automatic, published commands via the machine CAN bus to the subsystems.

Machine control requires processing an input signal, performing calculations and logic operations, and outputting a control signal. Typical inputs to a controller can be analog voltage, digital signals, or communication protocol (digital waveforms interpreted as bits). Controller outputs are typically voltages or currents to drive electric actuators or relays. Outputs may also send digital messages which can be received by other controllers. Selection of a controller for an application requires knowledge of the system, and what the required inputs and outputs (I/O) will be. One controller may not be able to process the required number of I/O channels for a machine. Further, even if the I/O channels were available, significant processing power would be required which may introduce lag and processing error and result in unexpected machine behavior. In this case, several controllers which can communicate together form a distributed control network.

Eight controllers were used on the Flex-Ro distributed control network. The electronic control units or ECUs were manufactured by Danfoss (Danfoss Power Solutions, Ames IA). The Danfoss controllers were selected for available pin configurations as well as their ability to communicate using the CAN bus on the Flex-Ro machine. There are two models of ECUs, three MC024-110s (24 pin) and five MC012-110s (12 pin). These Danfoss controllers are programmed using PLUS+1 GUIDE software. The graphical programming method is intuitive and robust programs can be created quickly without needing extensive knowledge in embedded controls.

Remote operation was developed using a Danfoss DP600TM display, a Danfoss JS1000 joystick, and Magnetek WIC-2402 Wireless CAN Bridge. These components were mounted in an enclosure with a strap which held the device for comfortable operation (Figure 2.1).

Figure 2.1: Remote control box developed for teleoperation of the Flex-Ro platform.

The laptop used a universal serial bus (USB) to CAN bridge for reading and writing messages on Flex-Ro's CAN bus. The FlexRoRun application programmed using MATLAB app designer, was developed to facilitate high level navigation control. The CAN bridge used was a Kvaser Memorator Pro x2 HS (Kvaser AB, Mölndal, Sweden). A Vector CANcase XL Log (Vector North America Inc. Novi Michigan) was also used during testing.

Shown in Table 2.1 is the list of primary components which were implemented as part of the control system for the Flex-Ro machine.

Table 2.1: List of control system hardware used on the Flex-Ro machine.

Model	Manufacturer	Purpose
MC024-110	Danfoss	Electronic control unit (24 pin)
MC012-110	Danfoss	Electronic control unit (12 pin)
WIC-2402	Magnetek	Wireless radio CAN bridge
Victor SPX	Vex Robotics	Steering motor controller
JS1000	Danfoss	Joystick for machine maneuvering
DP600TM	Danfoss	Display for remote operation
Memorator Pro 2xHS v2	Kvaser	USB to CAN bridge
AG-372	Trimble	GPS receiver
O3M151	ifm	3D Smart Sensor, obstacle detector
O3M950	ifm	IR illumination unit

2.2.2 CAN Bus J1939 Distributed Control Network

A distributed control network has many advantages. The controllers within the system split the processing of inputs and outputs for each subsystem. A controller of the system, usually with direct operator inputs, sends out machine control messages. For example, the operator commands the machine to slow down and steer to the right using a joystick. The hydrostatic drive controller will receive the message to slow down and adjust the hydraulic fluid flow. Simultaneously, the steering controllers received the message to

turn right and actuate the wheel angle to the desired operating position. Another benefit of the distributed control network is the ability to add or remove nodes without affecting the rest of the system. The nodes of the Flex-Ro bus can be seen in Figure 2.2.

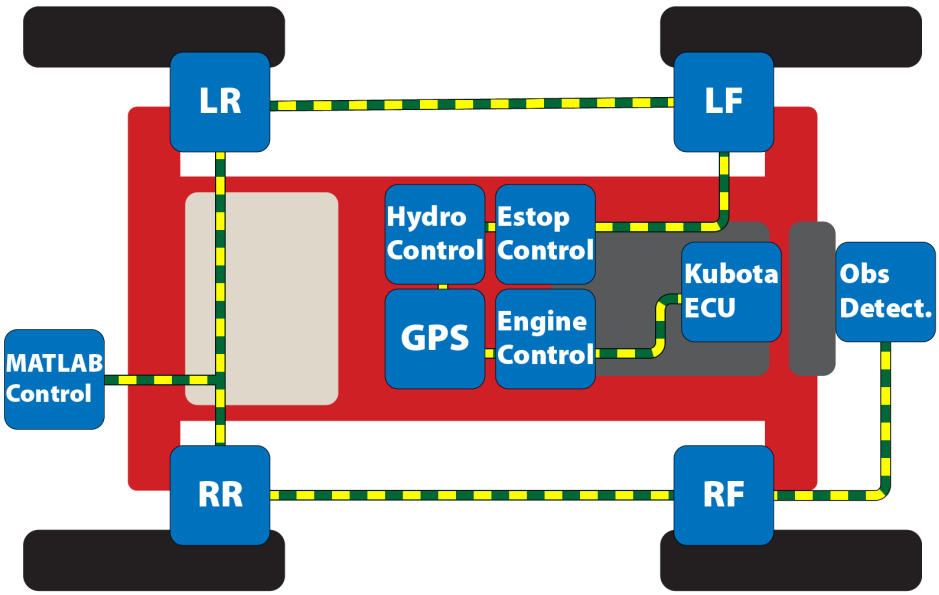


Figure 2.2: CAN node layout on the Flex-Ro platform. Dashed wire shows connection via J1939 CAN bus.

A CAN bus network connects compatible ECU's with a twisted pair of wires (Bell, 2002). These wires are used to send data bits across the bus. This data is received by other controllers and processed as commands or machine data. One standard high-level protocol for the formatting of these bits is SAE standard J1939 (Bell, 2002; Marx, 2015). The standard specifies how these data bytes are grouped and sent, called messages. Each message contains identifier bytes and a data payload. The identifier provides information to the ECUs on where the message came from and what it contains. The other ECUs on the bus can choose to process the message if programmed to receive it or ignore it.

The J1939 standardized communication provides a means to easily record machine data (Marx, 2015; Rohrer, 2017). Many devices have been developed to enable receiving and publishing messages on the CAN bus. A USB to CAN bus bridge allows the computer to send and receive messages in real time. There are many software applications for CAN logging and real-time decoding (Rohrer, 2017). However, programmatically sending CAN messages using a laptop in response to inputs is more limited. MATLAB, LabVIEW, and Visual Studio are a few examples. MATLAB was chosen to design a graphical user interface (GUI) given its ability to utilize existing hardware and the accessibility to useful toolboxes. The MATLAB developed Vehicle Network Toolbox is a suite of functions for sending and receiving messages on the CAN bus. The user can reference a database which MATLAB uses to automatically decode and encode message data. Further, the MATLAB app can be deployed to an executable file so others could install the program and run the Flex-Ro machine.

The messages created for the Flex-Ro platform used J1939 standard and proprietary formats. The source addresses selected for each ECU corresponded to global source addresses defined in the standard. Data to be transmitted used existing SLOT (scaling, limit, offset and transfer) definitions when applicable. Each of the messages were sent as broadcast without specific destination addresses. Priority was assigned based on urgency of the message. For example, the e-stop message received the highest priority to ensure the quickest response time to emergency.

2.2.3 Human Machine Interface

2.2.3.1 Remote Control Operation

A machine remote-control interface is necessary for basic operation. Navigating from a storage location to field or loading onto trailer requires safe and reliable human control. Indoor automatic guidance for a field machine is impractical due to unpredictable building enviroments and loss of GPS signal for location information. The Flex-Ro wireless remote control must be able to drive the machine, monitor operating variables, change system parameters, and perform an alignment of the steered wheels. The remote uses a display and joystick for intuitive ergonomic control. The right hand is dedicated to controlling the machine travel and steering via the joystick. The left hand is free to adjust parameters on the screen including the brake release, speed range, steering mode, and most importantly the e-stop.

The remote application was programmed using Danfoss PLUS+1 GUIDE software. The main screen of the remote-control interface includes information and controls for normal operation. This main display is shown in Figure 2.3.

Figure 2.3: Main operating screen of the Flex-Ro remote. Danfoss DP600TM display.

The numbered softkeys were mapped to controls for engine run and start, cruise activation, e-stop, speed range, steer mode, brake release, and menu access. The machine is programmed to stop when an obstacle is encountered. The operator can override the obstacle detection by pushing and holding the ‘OK’ button. Obstacle detection resumes once the button is released. Engine rotations per minute (rpm) and cruise set speed are adjusted using the arrow key pairs. The current softkey assignment is displayed on the screen as an icon. This includes changing controls, for example, brake release or apply which depends on the current machine state. There are graphics for the current speed range and steering mode. Values displayed on the main screen include remote and machine battery voltage, coolant and engine oil temperatures, cruise set speed, and GPS indicated vehicle speed.

Figure 2.4: Left - Remote diagnostics screen. Right - Remote steering calibrate screen.

The second page is the steering calibrate page which is shown in Figure 2.4 (Right). Optical quadrature encoders provide feedback for the steered wheels. The position of the wheel must be calibrated as the encoder only provides a relative pulse count. The count is zeroed at the wheel center position during calibration. The gearing of the steering motor and resolution of the encoder translates to +/- 42,186 counts to +/- 90-degree steering angles. Current absolute wheel position is saved at 2 Hz to non-volatile memory in case the machine is shut-off while the wheels are not at 0 degrees. The steering calibrate page includes controls for activating steering calibrate mode, changing which wheel to calibrate, and saving new center position. The graphics display which wheel is currently being calibrated. Outside of calibration mode graphically shows the current feedback angle of each wheel. Finally, included on the steering calibration screen is the same e-stop softkey in case of unintended steering or machine motion.

The last screen currently implemented is a diagnostic display, Figure 2.4 (Left). The e-stop button remains and is assigned to the same softkey. The diagnostics screen shows which ECU triggered an active e-stop. The user can then quickly diagnose the root cause of the e-stop flag. Also shown are engine hours, fuel rate, engine oil pressure, compass

bearing, and estimated fuel level. Two of the softkeys are for resetting the fuel level to either a full or half tank. More modules could be easily added to the diagnostic screen depending on the need of the operator.

The remote-control interface was developed such that individuals with no experience could operate the machine with minimal instruction. Components and information displayed on the screen can be easily changed, added, or deleted depending on the application installed on the machine. The remote is not, however, intended to become a high-level controller. An operator cannot drive to the field and select a navigation path using solely the remote at the time of publishing. Remote operating instructions are included in Appendix B.1.

2.2.3.2 Laptop and MATLAB Control Interface

Autonomous operation of the machine requires processing beyond the capability of a typical microcontroller. The high-level machine controller processes the current machine pose and position and calculates a steering angle. A high-level controller programmed using MATLAB App Designer was developed for Flex-Ro. While not proposed as a long-term solution for machine control, the use of the laptop provided many benefits including the ability to quickly develop and debug programs and implement a graphical user interface. The MATLAB Vehicle Network Toolbox provided the framework needed to communicate with the machine's J1939 CAN bus controllers. The MATLAB app, called FlexRoRun, was deployed as an executable application. Other users could then install the program without a MATLAB license.

Operation of the program begins with the initialization of the USB to CAN bridge. Currently, the tool supports a Vector CANcase XL Log and Kvaser Memorator Pro 2xHS v2. A CAN message database file was created which contained both J1939 standard and custom Flex-Ro messages. Use of the database streamlines the decoding and encoding process of message transmission. The main program execution loop is time triggered at 5Hz. This rate was chosen to match the incoming signal from the GPS and was sufficient for the dynamics of the machine.

There are several parts to the FlexRoRun main run page as shown in Figure 2.5. Each of the main control systems were divided into modules. Essential engine, hydrostatic drive, steering, and e-stop controls are accessed on the main run page. Keyboard shortcuts were mapped to buttons and sliders, which reduced error prone mouse clicking. The NebrasTrack (Flex-Ro's automatic navigation system) controls are also located on the main run page. The module includes a map of the current track and machine's location, tracking activation, and steering control output for debugging. Lateral shift buttons provide fine and coarse adjustment of tracking location relative to the defined path.

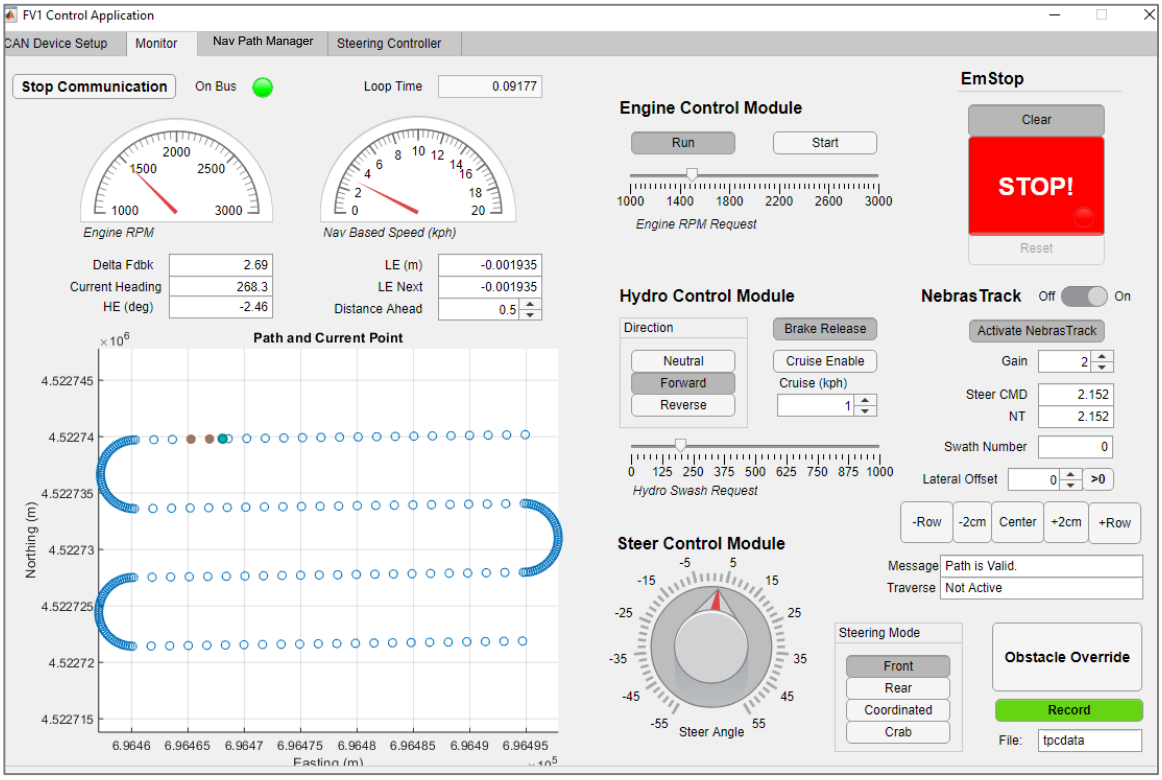


Figure 2.5: Main application tab for the FlexRoRun application. Developed using MATLAB App Designer.

The Navigation Manager page (Figure 2.6) contains track creation controls. There are several ways to import a navigation track. Recording an AB line requires either manual entry of the latitude and longitude coordinates or driving to the A and B points. The user can then generate a field coverage map, assuming a rectangular field. Parameters include track width, and number of swaths. Points are populated based on the recorded AB points, and a simple constant radius headland turn is calculated. The other tracking modes can also be activated on this page. Modes include waypoint following, AB Parallel passes, and AB Traverse headland turns.

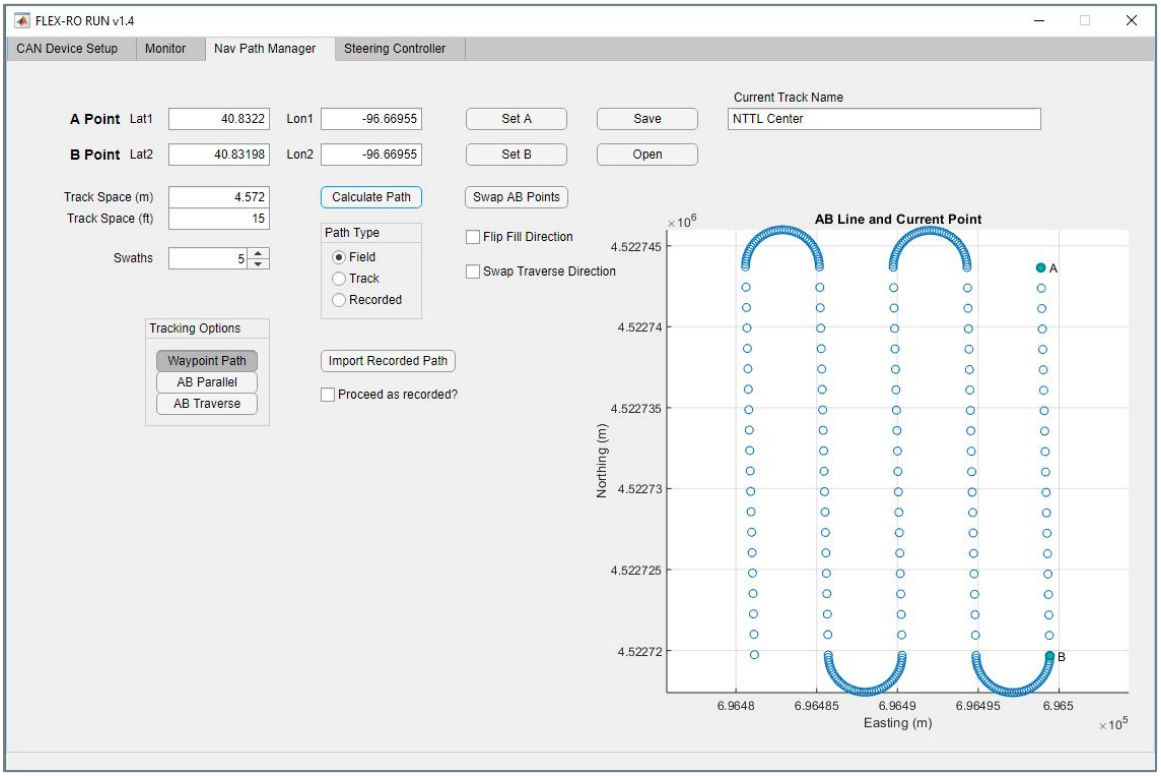


Figure 2.6: Navigation tab within the FlexRoRun application. Developed using the MATLAB App Designer.

The navigation tab includes controls for importing and saving a prerecorded path. Saving a track saves the point array and track metadata, such as track-width, navigation type, and track name. The current track is saved during application shutdown. When the application is relaunched the last used settings and track are loaded. Basic FlexRoRun operating instructions are included in Appendix B.1.

2.2.4 Low Level Machine Control

2.2.4.1 Engine Control

The engine control ECU receives start and stop commands via a CAN message from the main machine controller (either remote control or FlexRoRun). Those signals are

processed into digital outputs of the run and start pins of the Kubota engine ECU. The engine control ECU translates requested engine rpm into a TSC1 J1939 standard message. The engine control CAN message contents are defined in Table 2.2.

Table 2.2: Engine control message. Start and Length shown as Byte.bit

Message Description: Engine control data from main machine controller.

ID: 0x10FF4427

Start:	Length:	Description:
0.0	0.2	Engine run enumeration
0.2	0.2	Engine start enumeration
1.0	2.0	Engine RPM request

2.2.4.2 Hydrostatic Drive Control

The hydrostatic drive (hydro) control ECU processes drive command messages into pump control signals. The pump controls machine speed by varying the swash plate angle of the tandem piston pump. Each pump supplies two drive wheels so the swash plate commands must be the same. These commands are either received via remote joystick or from the high-level controller. Before the machine can be moved, a brake release signal must be received. Once the command is processed, the control-cut-off valve is activated which supplies pressure to the wheel brakes. The brakes are in normally ON position with spring pressure. Direction and magnitude commands are included in the hydro control CAN message described in Table 2.3. Ramp up and down parameters are programmed into the controller for smooth acceleration and to avoid damage to the

pump. The deceleration was tuned so the machine slows rapidly and without tire skid. A cruise control function is also supported which controls the swash plate angle using a closed loop between requested and actual machine speed.

The hydro-controller also processes the machine's response to detected obstacles. The hydro controller manages vehicle speeds, so if an obstacle is detected, the controller can react quickly. The function of the obstacle detection algorithm is covered more in depth in Section 2.2.8.

Table 2.3: Hydrostatic drive pump control message. Start and Length shown as Byte.bit

Message Description: Hydro. control data from main machine controller.

ID: 0x4FF4127

Start:	Length:	Description:
0.0	0.2	Direction enumeration
0.2	0.2	Brake release enumeration
0.4	0.2	Cruise control enumeration
0.6	0.3	Obstacle override enumeration
1.0	2.0	Drive magnitude
3.0	1.0	Cruise speed request

2.2.4.3 Steering Control

There is a steering control ECU located at each wheel. A CAN message sends a commanded center steering angle as well as the current steering mode enumeration. Each ECU processes this message using a PLUS+1 GUIDE Ackermann steering block (Appendix D.2). This block calculates the wheel angle based on the mode and centerline command. There are four programmed steering modes; front, rear, coordinated, and crab. The centerline commanded angle and steering mode are included in one of the steering messages described in

Table 2.4. A closed loop PI controller commands a motor driver via PWM (pulse width modulation) signal to turn the wheel. Digital encoder pulses are counted to determine the wheel's current angle. There are +/- 42,186 pulses to turn +/- 90°. Direction of turn is determined by the sign of the phase offset between A and B encoder channels. The controller periodically saves the absolute wheel position in case of machine power down.

Table 2.4: Steering control messages. Start and Length shown as Byte.bit

Message Description: Steering command information.

ID: 0x8FF4327

Start:	Length:	Description:
0.0	2.0	Centerline commanded steer angle
2.0	0.2	Steer mode enumeration

Message Description: Steering calibration commands.

ID: 0x1CFF4527

Start:	Length:	Description:
0.0	0.2	Calibration mode enable
0.2	0.2	Active calibration wheel
0.4	0.2	Save wheel calibration enumeration

Each of the four wheels steers independently of the others. Ackermann's steering angles require the inner and outer wheels to steer to different angles based on a centerline commanded angle. In this system, the wheels may arrive at their individual commanded angle at separate times. This is especially apparent at very sharp steering angles when the inner wheel angle turns to 90-degrees and the corresponding outer wheel angle is near 60-degrees. Synchronization was implemented to slow the speed of the wheel that has a smaller delta to the next commanded steer angle. Tuning adjusted the proportion gain on the delta until the wheels arrived at extreme steering angles simultaneously.

Each steering ECU also monitors a physical e-stop switch. When the switch continuity is broken, an e-stop flag is immediately sent across the CAN bus with the highest priority. The flag remains until the e-stop switch is reset, and control is resumed when the user resets the machine with the remote or FlexRoRun software.

2.2.4.4 Emergency Stop Network. Start and Length shown as Byte.bit

Each of the controllers sends a status message at 10 Hz. The main machine controller (remote control or FlexRoRun) processes the status messages from all the machine controllers to check for e-stop flags. If the main controller does not receive a message from an ECU after 2 seconds, an emergency flag is set. After a flag is set, all controllers must send a reset signal to ensure the machine is ready to return to service. A heartbeat signal within each message ensures that the controller is properly functioning and is on the bus each time a new message is sent. The contents of the status CAN message are outlined in Table 2.5.

Table 2.5: Example status message. Start and Length shown as Byte.bit

Message Description: Status message from hydrostatic drive controller.

ID: 0x0FF572E

Start:	Length:	Description:
0.0	1.0	Heartbeat (0 – 255)
1.0	0.2	Emergency stop enumeration

2.2.5 Navigation Error Calculation

The implementation of a straight-line tracking algorithm is the first step for automatic navigation. The operating environment for the Flex-Ro machine is row crop fields. Successful navigation down the rows without crop damage is of highest importance for the machine. There has been a significant amount of research in the implementation of various controller algorithms for automatic steering (Bell, 1999; Boyali et al., 2018; Godoy et al., 2012; O'Connor, 1997; Troyer, 2017). Kinetic based control while simple and effective, lacks robustness during higher speeds. Dynamic control algorithms require more development time, as well as a higher number of sensor inputs and computational power. The navigation controller for Flex-Ro was first developed using a kinetic model.

The first step in determining an output steering angle was calculating the lateral (perpendicular) error from the desired tracking line. This desired tracking line was defined by two points (A and B) recorded in latitude and longitude coordinates. Conversion from latitude and longitude degrees to Universal Transverse Mercator (UTM) coordinates in meters enables a direct calculation of lateral error. Conventionally, the easting direction is along the UTM x axis while the northing direction is along the UTM y axis. The calculation of the lateral error uses three points, A, B and current position, C.

The cross product by the vector $\overrightarrow{BA}$, and the vector $\overrightarrow{BC}$, gives the area of the parallelogram (shaded in Figure 2.7) formed by the two vectors. A parallelogram must have the same area as a rectangle (orthogonal sides) with the same perpendicular distance between parallel lines. Thus, the area divided by length from B to A ($\left\|\overrightarrow{BA}\right\|$) results in

the perpendicular distance L from the line $\overline{BA}$ to point C. This calculation is valid at any point along the line $\overline{BA}$ (navigation line).

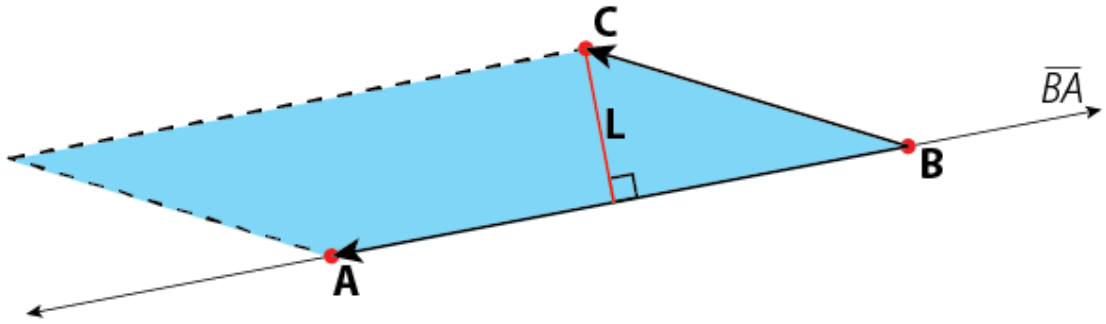


Figure 2.7: Lateral error (perpendicular distance) at point C from line defined by points A and B.

The sign of the lateral error has not been applied at this point. The sign convention is to remain the same regardless of machine orientation. The navigation control system will manage the controller response based on if the machine is traveling in forward or reverse. The calculation of the sign depends on the difference in slopes between the navigation line machine's current point. The convention was developed based on point A shown in Figure 2.7. It should be noted that the UTM coordinates will always be positive values, alleviating potential complications. Only the sign of the slope difference is applied to the lateral error. The MATLAB script for calculating lateral error is attached in Appendix D.1 for reference.

Equation 2.1: Perpendicular offset from navigation line calculated from three points, where A and B define the navigation line

$$laterror = \frac{\|\overrightarrow{BA} \times \overrightarrow{BC}\|}{\|\overrightarrow{BA}\|}$$

Equation 2.2: Applying the sign of lateral error based on the line (AB) and current points.

$$laterror = laterror * sign\left(\left(\frac{C_y - A_y}{C_x - A_x}\right) - \left(\frac{B_y - A_y}{B_x - A_x}\right)\right)$$

2.2.5.1 Heading Error Calculation

It is important to know the current machine heading for navigation control. The Trimble Ag-372 GPS publishes heading information along with the GPS indicated speed and location on the CAN bus. The compass heading convention defines North to be 0° with positive clockwise rotation (i.e. driving straight east is a 90° heading). A limitation of the GPS unit was that the heading couldn't accurately be determined until the machine was in motion. However, this would only be a factor for a very short period as the vehicle initialized motion. More accurate methods of determining heading, including the use of two GPS units, will be considered for future development.

The Flex-Ro machine was to navigate down a path in either direction. The heading error was defined as the current machine heading minus the track heading. The track heading was defined as the angle from due north clockwise to the line formed by A and B. Also, the A and B points can be swapped with no effect. In other words, if the machine heading is 180° off the track heading, there should be no heading error. The result of such processing can be viewed in Figure 2.8. Note that the track angle assumed to be

following is 0° (straight north) and 20° clockwise from north as examples.

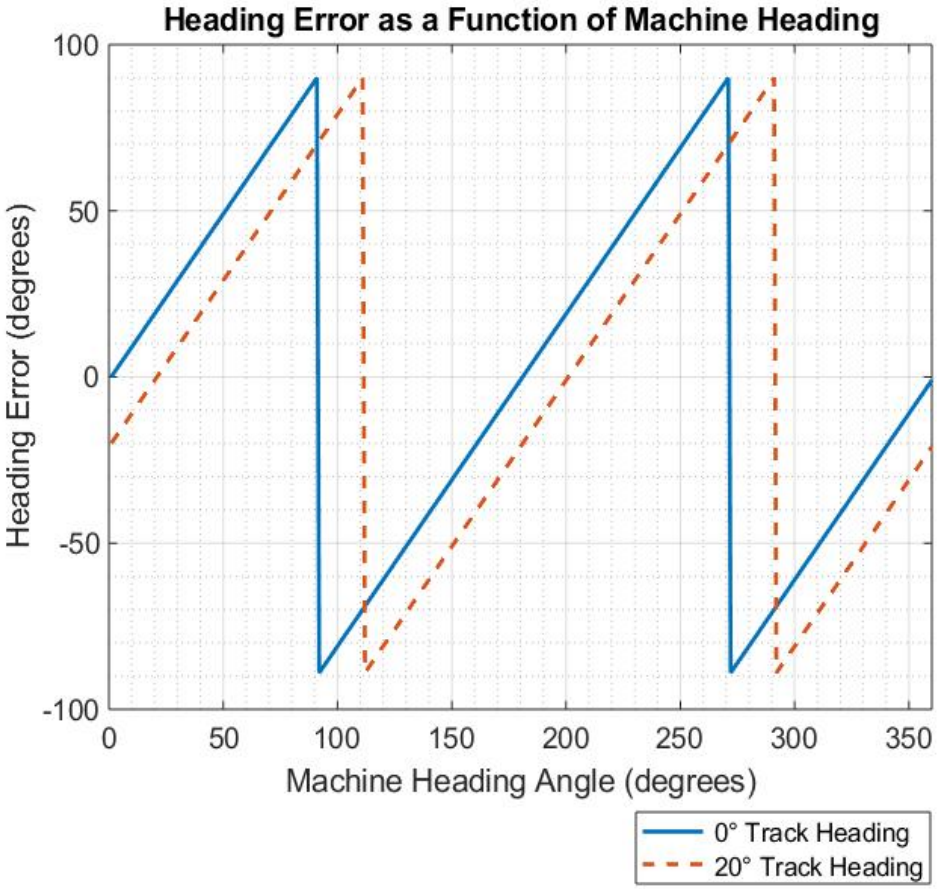


Figure 2.8: Heading error as a function of machine heading. Discontinuities at vertical lines.

The track heading was calculated using the ‘legs’ MATLAB mapping toolbox function. The two navigation points in latitude and longitude are aruguments and provide an output in degrees in the standard compass coordinate system, directly comparable to the GPS heading output.

2.2.6 Waypoint Navigation

Waypoint navigation is defined as following a path which is defined by a matrix of points. The machine navigates from point to point in the sequence defined by the path matrix. The path can be any shape, but in its most basic form a straight line is drawn between points. The curvature of the path can be calculated by using the next three points ahead. Waypoint following is used for navigation of paths other than straight A and B point parallel tracking.

MATLAB provides a kinematic based function for calculating a steering angle based on pose, heading error, and velocity. This equation is based on research completed at Stanford University, by Hoffmann et. al (2007) and Paden et. al (2016). The control equation was originally used on a car named Stanley which competed in the DARPA Grand Challenge 2005. The base function uses a pure pursuit type strategy without accounting for curvature of the path.

Flex-Ro uses the factors of the Stanley Lateral Controller slightly different than it was designed. Arguments into the equation are reference pose and current pose. This method reflects the pure pursuit nature of the controller. Flex-Ro required a stable line following algorithm with GPS coordinates. As the lateral and heading errors were already calculated, the current pose (machine origin) was set to zero. As a result, the distance to point in the x direction (longitudinal) became a tunable factor, and the y distance (lateral) was set to be the lateral error as shown in Figure 2.9. The heading error was set as the reference heading.

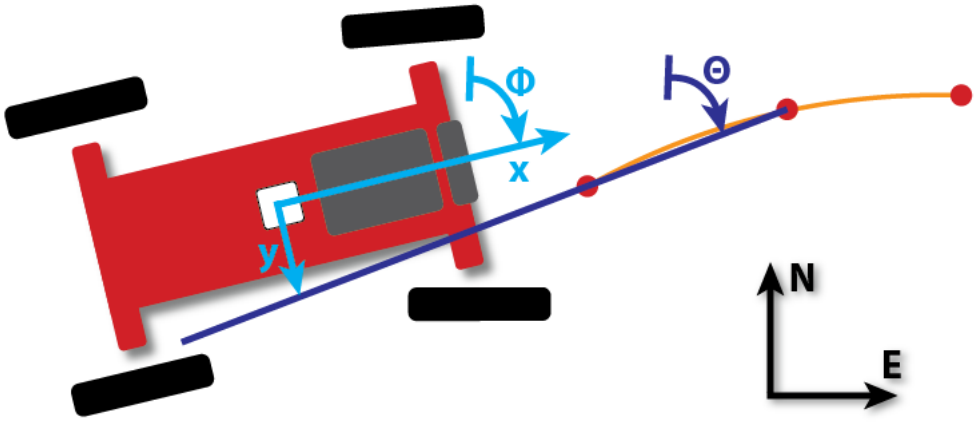


Figure 2.9: Vehicle coordinate system convention with respect to geographical north.

The other tunable factor was the position gain. The position gain controlled how aggressively the machine responded to lateral error. Increasing the gain would result in a system with a quicker response and less steady state error, but also caused instability as speed increased.

Waypoint navigation included more than the straight AB lines mentioned above. First, it was desired that the machine could start a path at any point and traverse in either direction. The cycle time also had to be fast enough for smooth navigation without lag. The machine position and heading would then have to determine what points in the path would be used for navigation. A navigation path selection algorithm selected the next three points that were in front of the machine, and within a maximum lateral error.

The waypoint paths were stored in a matrix of UTM x and y coordinate locations. When navigating, during each program cycle, the matrix of points is transformed into the machine coordinate system using the current heading of the machine. (The transformed

points are indexed to their original UTM coordinates to retain accuracy due to error in the heading data.) Then the points can be sorted and filtered. The negative x points (behind the machine) are eliminated. Then programmatically, the points in front of the machine are checked to ensure they are within the current swath. The first two points (in UTM coordinates) are then used for the lateral and heading error calculation. The first three points (in machine coordinates) are used to calculate the curvature of the upcoming path.

The curvature is used as an added factor to the pure pursuit type algorithm. The current path curvature is calculated by using a custom MATLAB function (Mjaavatten, 2018). The curvature of the upcoming path is used to bias the steering angle based on the geometry of the machine. This function outputs the vector which points to the center of the circle defined by the three points. The sign of the y coordinate tells whether the path is curving left or right, and the direction of the required steering angle as a result.

The bias of the steering angle helps correct for upcoming curvature in the path but does not provide information on whether the machine is inside or outside the curve. Short linear segments are used to calculate the lateral and heading error information. The steering angle from the Stanley controller is then added to the steer bias and then sent via centerline commanded angle message to the steering ECUs.

2.2.6.1 Field Path Generation

The basic operating environments for the Flex-Ro machine are research plots with straight rows. There are open headlands with plenty of clearance for turning at the end of the rows. A basic tool for generating these types of field paths was developed for quick

processing once the machine was on-site. One advantage would be to navigate fields without access to as planted navigation data. The A and B points were set by positioning the machine in a set of rows at each end of the field. Parameters such as the track width, fill direction, and number of swaths were entered and then a path is calculated. If the as planted navigation track data was available, the latitude and longitude coordinates of the A and B points could be entered directly to calculate the field coverage swaths.

2.2.6.2 Recorded Path Import

Paths which are driven in semi-controlled environments are well suited for autonomous navigation using the Flex-Ro's waypoint navigation. The FlexRoRun application processes data which contains navigation performance and machine position information. A record button on the FlexRoRun main page toggles the saving of this data to a log file accessible by the program. If the user wants to record a path, a log is taken while the user manually drives the machine down a path. Then, within the NavManager tab, this log is loaded and processed into a waypoint path. The path can then be followed as recorded (following the same direction) or in either direction. Lateral offsets can also be added to compensate for multiple swaths of the same recorded path.

2.2.7 Headland Turn Strategies

A headland turn is the maneuvering a machine from one working pass to the next. There are multiple ways to complete a headland turn. The independent four-wheel steer capability of the Flex-Ro enables four different steering strategies. Changing the instantaneous center of rotation (ICR) is possible to maximize turn efficiency. Crab

steering allows for sideways travel. Thus, a headland ‘turn’ can be completed without changing machine direction. There are advantages and disadvantages to both types of turn strategies. The scope of this thesis was to compare the traditional front wheel steering method with the traverse type (Figure 2.10).

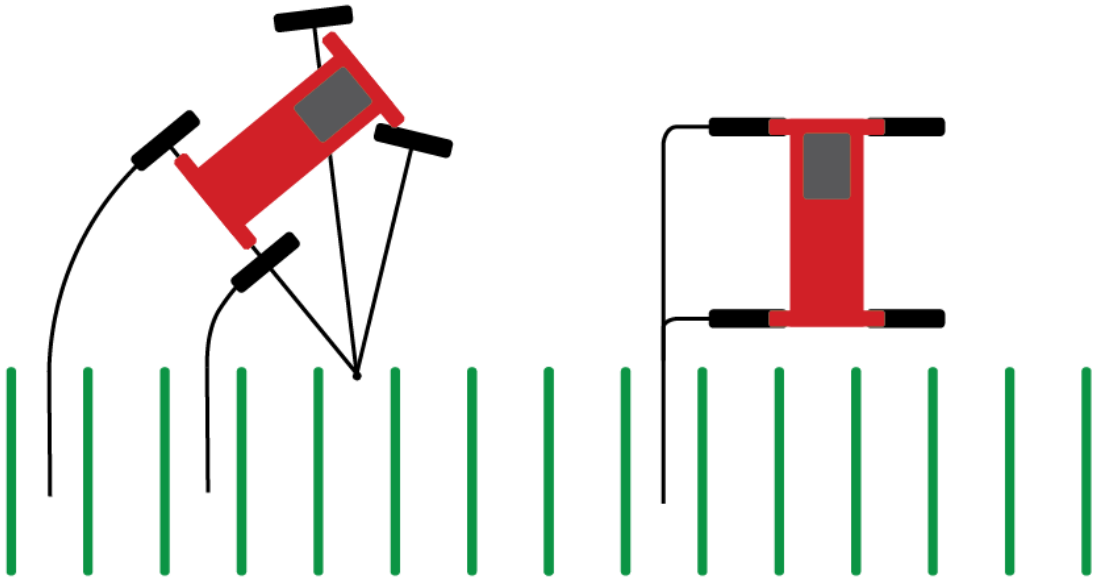


Figure 2.10: Left: Conventional front wheel steer headland turn. Right: Traverse headland navigation method.

2.2.7.1 Conventional Radius Turn

The conventional method of steering consists of a fixed rear axle and Ackermann’s angles for the front wheels. The turn begins once the vehicle has fully entered the headland. There are multiple paths the machine can follow when completing a front wheel turn. The simplest is a semicircle tangent to the crop swath passes or U turn. This path can be easily generated using the AB points of the path, and the desired track width. The density of points on the arc directly affects the accuracy of the machine path following using the Flex-Ro waypoint algorithm.

2.2.7.2 Headland Traverse

The headland traverse turn is activated once the machine fully enters the headland. This area is currently defined as the zone outside the crop rows and perpendicular to navigation segment AB. The machine stops and waits until the wheels turn to 90°. The machine then slowly travels until the lateral error to the next swath is less than 10 cm. The machine then stops, waits until the wheels have returned to their normal orientation, and proceeds down the next path.

2.2.8 Obstacle Detection

Traditional machinery operators function as obstacle detectors. Autonomous operation requires either a completely controlled area of operation (no possibility of obstacles) or sensors which can warn of upcoming collisions. The motivation for installing the obstacle detection system is primarily for operator and bystander safety. Damage to the machine and its operating environment were also concerns which could be mitigated with the implementation of an obstacle detector. The objectives for the obstacle detection system are as follows:

1. Implement an obstacle detection sensor into the CAN based vehicle control architecture of Flex-Ro.
2. Stop Flex-Ro automatically within 1-2m of a pedestrian sized object at typical field operating speeds.

It was important to implement obstacle detector solutions for Flex-Ro which would utilize the existing J1939 CAN bus network. The O3M 151 3D Smart Sensor produced by ifm (ifm efector, inc. Malvern, PA) used J1939 messages to transmit processed object information and is seen in Figure 2.11.

Figure 2.11: ifm O3M 151 3D Smart Sensor installed on the front of the Flex-Ro platform.

This sensor uses an IR (infrared) pixel array to calculate the time of flight and resulting distance to object. The pixel matrix consists of 64 (horizontal) x 16 (vertical) IR dots, over an aperture size of 70° (horizontal) x 23° (vertical). The range is advertised to be 35 meters. However, at that distance, one IR pixel covers 77 x 91 cm. As a result, the object would have to be very large (e.g. dump truck) for reliable detection. The relatively slow

operating speeds of the Flex-Ro platform require only a short detection range. Three update rates are programmable, 25, 33, and 50 Hz. Flex-Ro used an update rate of 25 Hz.

One of the main benefits of the O3M151 sensor is the direct interface with the J1939 CAN bus network. Ifm provides three firmware packages for the sensor; obstacle detection, distance measurement, and windrow navigation. Vision Assistant, ifm's custom application programing interface (API), makes configuring the sensor simple. The program shows real time data and constants such as orientation, and sensitivity can be adjusted.

The obstacle detection firmware was uploaded for the application on the Flex-Ro machine. The sensor can track and report up to 20 objects over the CAN bus, however, the Flex-Ro was programmed to track 3 to limit busload. A large amount of data is available for each object over two CAN messages. A decoded CAN trace for one object is given in Figure 2.12.

18.824774	CAN 2	4FF10EFx		MoCa_Obj_0_A		CAN Frame		Rx		8		8		3C	BC	CA	CA	75	10	87	D8
Obj_0_vx		0.0000	m/s		3C	relative velocity of object, x direction															
Obj_0_Type		0	n/a		0	Type identifier of the object...															
Obj_0_vy		0.0000	m/s		3C	relative velocity of object, y direction															
Obj_0_Measured		1	n/a		1	Flag indicating that this object has been measured in actual frame															
Obj_0_ay		0	m/s^2		A	relative acceleration of object, y direction															
Obj_0_ep		6	enum		6	existence probability, provided as an enum ...															
Obj_0_ax		0	m/s^2		A	relative acceleration of object, x direction															
Obj_0_qvx		6	enum		6	quality of vx signal provided as an enum ...															
Obj_0_az		0	m/s^2		5	relative acceleration of object, z direction															
Obj_0_TrackAge		3	enum		3	Age of track, provided as an enum...															
Obj_0_Id		65	n/a		41	id of object															
Obj_0_zMin		0.8000	m		21C	minimum z coordinate of object															
Obj_0_vz		0.0000	m/s		C	relative velocity of object, z direction															
Obj_0_A_cnt		3			3	Obj 0 part A message counter															
18.825914	CAN 2	4FF11EFx		MoCa_Obj_0_B		CAN Frame		Rx		8		8		20	ED	B7	7C	FE	8F	F7	D9
Obj_0_dz		0.6400	m		20	maximum z coordinate of object (zMax = zMin + dz, dz = zMax - zMin)															
Obj_0_dy		0.5800	m		7ED	delta value of y coordinate of object's second point (y2 = y1 + dy, dy = y2 - y1)															
Obj_0_dx		-0.1000	m		7CB	delta value of x coordinate of object's second point (x2 = x1 + dx, dx = x2 - x1)															
Obj_0_x1		1.8800	m		FFE	x coordinate of object's first point															
Obj_0_y1		0.5600	m		FBC	y coordinate of object's first point															
Obj_0_qvy		6	enum		6	quality of vy signal, provided as an enum ...															
Obj_0_History		0	n/a		0	Flag indicating that this object has been seen on bus															
Obj_0_B_cnt		3			3	Obj 0 part B message counter															

Figure 2.12: CAN message output from one object detected by ifm O3M 151 3D Smart Detector. Viewed and decoded on Vector CANalyzer software.

The Flex-Ro machine used this obstacle data to prevent forward collisions. The object location and speed combined with the machine speed, can be used to predict how far the machine will stop from the object. This information commands the machine to either continue, slow to stop, or brake to stop. The “slow to stop” mode commands the swash plate of the hydraulic pump to ramp down to zero, smoothly bringing the machine to a halt. The brake to stop uses the wheel spring brakes. Using these brakes abruptly stops the vehicle and is potentially damaging to the machine. The use of the brakes is a last resort safety trigger for objects less than 50 cm from the front of the machine.

The equations of motion are used to calculate the stopping distance between the object and machine at the current state. A distance of two meters was set as the desired stopping distance from the front of the machine to an obstacle. The first calculation is to predict

the object's location based on its CAN message data. The time used in the calculation is the loop period of the controller.

Equation 2.3: Predicting the object's position with respect to the machine's x coordinate.

$$x_{obj,pred} = x_{0,o} + v_{x,o}t + a_{x,o}t^2$$

Equation 2.4: Predicting the object's position with respect to the machine's y coordinate.

$$y_{obj,pred} = y_{0,o} + v_{y,o}t + a_{y,o}t^2$$

Variables x and y are distances relative to the machine's coordinate system and are sent by the obstacle detector. The variable $x_{0,o}$ is the initial position of the object, $v_{x,o}$ and a_{xo} are the object's velocity and acceleration in the x direction. The y coordinate of the object is used to determine if it will be in front of the machine. The stopping distance of the machine can be predicted with the equation using deceleration and current velocity.

Equation 2.5: Calculating the machine stopping distance given the current velocity.

$$x_{stop,machine} = \frac{{v_{x,m}}^2}{2a_{x,m}}$$

Variables $v_{x,m}$ and $a_{x,m}$ are the velocity and acceleration (or deceleration) in the x direction of the machine. If the object is calculated to be within the 2-meter safe stop distance in front of the machine, the ramp to neutral mode is activated. Normal operation is resumed once the object is out of range.

The obstacle reaction was evaluated using a static 40 cm x 100 cm board. The Flex-Ro machine approached at different speeds within the field operation range. These speeds

were 2, 4, 6, 8, and 10 kph. Then the stopping distance from the front of the machine to the obstacle was measured and recorded for three repetitions.

2.3 Results and Discussion

2.3.1 Safe Stop System

A fail-safe emergency stop system is essential to all moving machinery. The primary hazard for machine operation and testing of new algorithms is loss of control and unpredicted motion. Redundant safety features were incorporated to mitigate this hazard. Four e-stop switches were positioned around Flex-Ro for quick access. An example of one can be seen in Figure 2.13. A dedicated softkey serves as the e-stop on the remote interface. Each e-stop was verified to work and trigger a machine shutdown in less than two seconds.

Figure 2.13: One of four red e-Stop buttons located on exterior of machine for quick and safe access.

The use of an untethered remote control introduced potential connectivity issues. Quickly stopping in the event of connection failure was critical to safe machine operation. Redundant checks ensured the machine would trigger a shut-down in the same way pressing the e-stop button would. The remote continuously monitored the bus for the presence of all required ECUs. The author verified e-stop operation by intentionally unplugging ECUs during remote operation. The wireless CAN-bridge (Model: WIC-2402, Magnetek, Menomonee Fall WI) included a high-side driver which switched off when connection was lost. This would trigger an e-stop flag on the Flex-Ro power ECU and result in a machine shut-down. Remote connectivity was assessed for both out of range (connection dropped) and remote power off events. The machine stopped safely

within two seconds of the fault occurring for both tests. The tested line of sight range of the remote was near 300 m.

Tethered laptop operation required similar e-stop failsafe requirements. An e-stop must be triggered when the laptop becomes disconnected. Other e-stop events include the laptop freezing or malfunctioning, an operator pressing a button or key, and an on-machine e-stop switch activation. Each of these e-stop modes were verified. Laptop e-stop response times matched that of the remote-controlled systems, and repeatedly shut-down the machine within two seconds.

Both remote and tethered operation e-stop systems were designed to be redundant and trigger in common failure modes. These redundancies were tested and verified and provided a basis for developing more advanced controls. Researchers were confident if a failure or undesired motion occurred when testing control algorithms, the machine could safely be stopped.

2.3.2 Navigation

The scope of the navigation control system for this thesis was to navigate a field of 30-inch (76cm) crop rows with open (no crop) headlands. Successful navigation would be proven simply by inspecting for crop damage and maintaining a 95% lateral error of less than 10cm. John Deere (Deere and Co. Moline, IL) uses a 95% lateral error in advertising literature of their AutoTrac system. ("Accuracy equation explains how AutoTrac™ accuracy is derived," 2013) The 95% error is calculated by sorting the absolute value lateral errors from smallest to largest. The value at the 95% index of the total length of

array was recorded. As a result, 95% of the lateral error values were smaller than the reported value. The threshold of 10 cm was chosen as a round number nearing the limit which would cause crop damage. Successful headland turns were defined as the machine re-entering the next swath without crop damage. Two methods of headland turns were to be compared for performance and efficiency. Waypoint navigation would be quantified using the 50 cm 95% lateral error. The machine would need to be able to track down a narrow farm road during these operations, allowing wider margin of error compared to crop rows. Navigation tracking parameters were tested in normal field conditions.

2.3.2.1 AB Line Navigation

The primary purpose of the AB line navigation was to drive down existing crop rows. The first application for the Flex-Ro machine was phenotyping, which required a slow travel speed, less than 5 kph. Thus, the initial navigation controller algorithm was tested at these speed ranges. AB line passes were conducted in several locations, however, corn stubble (Figure 2.14) proved to be beneficial for visually verifying successful navigation. Tuning continued until performance met the desired requirements. The Tracking Performance Calculator (TPCalc) app programmed using MATLAB app designer completed post processing and calculation of tracking parameters. The TPCalc interface (screenshot in Appendix D.3) quickly processed recorded tracking data. Outputs were 95% lateral and heading errors, as well as average speed. Plots included the path in relative UTM coordinates and lateral and heading errors over time. This tool provided rapid feedback while making tuning changes.

Figure 2.14: Testing AB line navigation in corn stubble.

A 100 m stretch of level field ground was used for the straight AB line verification test runs. The recorded 95% error at an average speed of 3.7 kph was 3.23 cm. This was well within the acceptable bounds required for navigation within the 76 cm crop rows. A plot of recorded lateral error in AB line tracking data can be found in Figure 2.15.

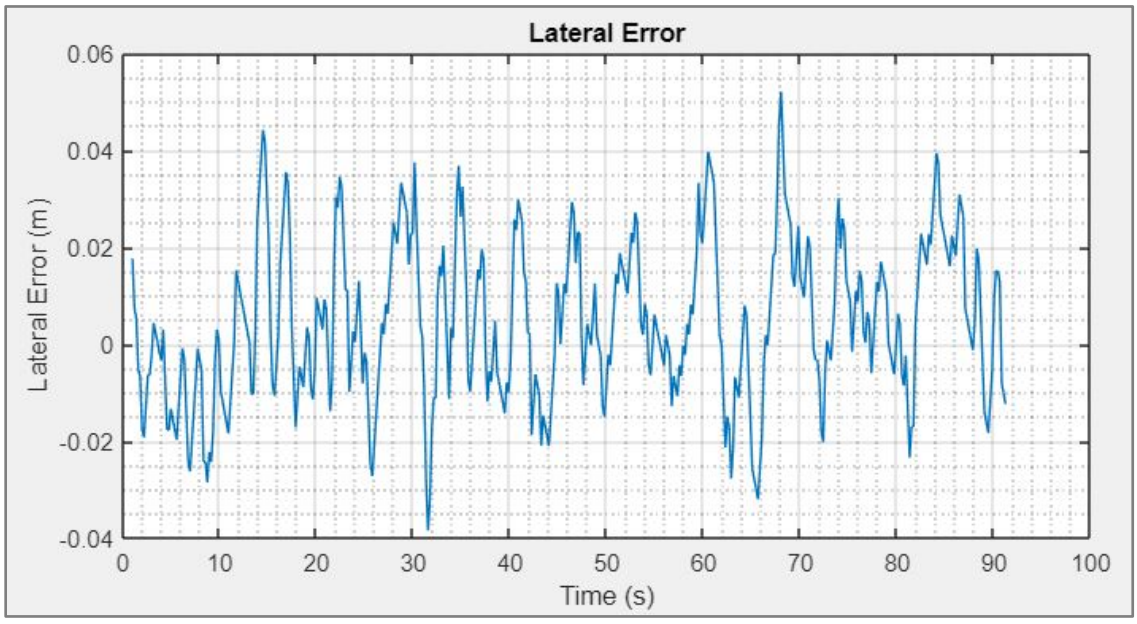


Figure 2.15: Lateral error over time, automatically tracking on level field ground at 3.7 kph.

AB line navigation was also tested in the reverse (front boom) direction. The recorded 95% lateral error in reverse was 3.9 cm at an average speed of 3.9 kph which met the automatic navigation objective.

Variation in tracking performance can be attributed to the precision of the GPS. The Flex-Ro machine used a Trimble Ag-372 without correction service. Further, it was noticed that during recording, some GPS points were missing. This indicated that the position wasn't always updated during each program loop. This often-caused spikes in the lateral error on the following loop when the position was updated.

2.3.2.2 Waypoint Navigation

Testing of the waypoint navigation began with recording a path to follow. The GPS points were recorded at 5 Hz. The data recorded was thinned so the remaining points

were one meter apart. This density of points provided smooth navigation around complex curves. A simple path cleaning script was developed in MATLAB to perform this procedure. This script can be found in Appendix D.4.

A remotely driven path was recorded using the Flex-RoRun interface. Then this path was cleaned and set as the current navigation track. The path was to be followed at 3 kph and consisted of curved and straight sections. These were features which could be present in a return-to-home track. Machine tracking performance would be evaluted in comparison to this recorded path. The machine navigated the path at a 19.2 cm 95% error. This met the desired requirement to track within 50 cm 95% error when following a prerecorded path. The path coordinates and lateral error while tracking is plotted in Figure 2.16.

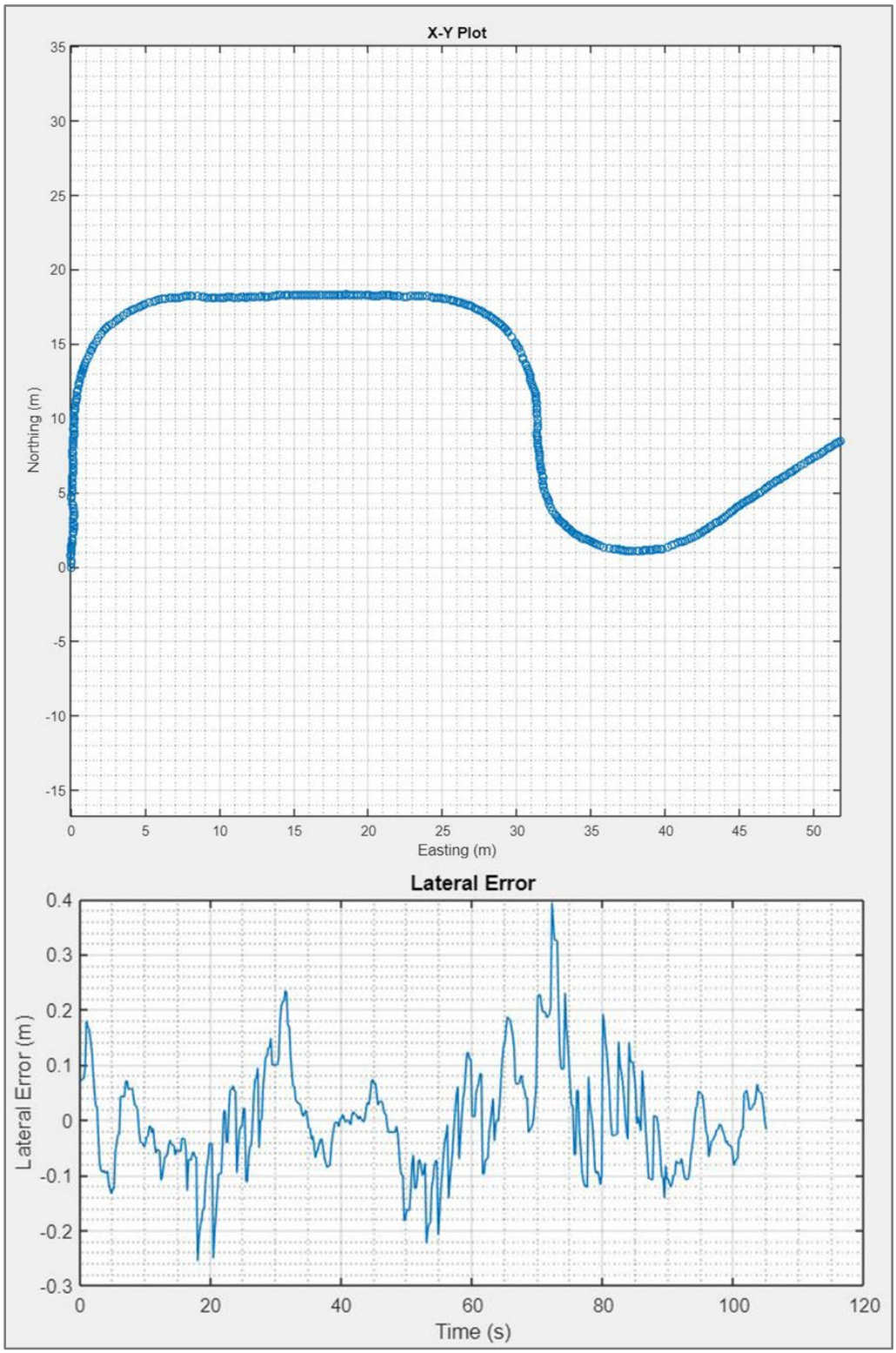


Figure 2.16: Automatic waypoint path following and lateral error over time. Coordinates translated to where data recording was initialized.

2.3.2.3 Headland Turn

Results of the headland turn controller development compares the performance between a ‘U’-type front wheel steer and traverse (crab) method. There are advantages and disadvantages to both methods, depending on the application. The phenotyping application developed for Flex-Ro was sensitive to changing light conditions. A traverse turn kept the sensing boom facing the same direction to reduce shadow variation during field passes. Alternatively, a spraying application may require the boom to remain at the rear of the machine.

Headland turn time directly correlates to field efficiency which means the less time spent turning the more efficient the operation. Turning methods were compared at the same set speed, 3 kph. While driving the machine faster through the headland would result in a higher efficiency, tests showed decreased accuracy when entering the next swath resulting in a longer period to reacquire the AB track. The swath width was set to 4.572 m (15 ft. or 6-30 in. rows). The front-wheel only headland turn (Figure 2.17, left) was completed in 17.8 seconds. The traverse turn in the right frame of Figure 2.17 took 16.6 seconds. The traverse method, while traveling less physical distance, must stop and wait until the wheels turn to 90 degrees before traversing to the next pass. Both turn methods resulted in a fast reacquisition of the crop AB line at the slow vehicle speed.

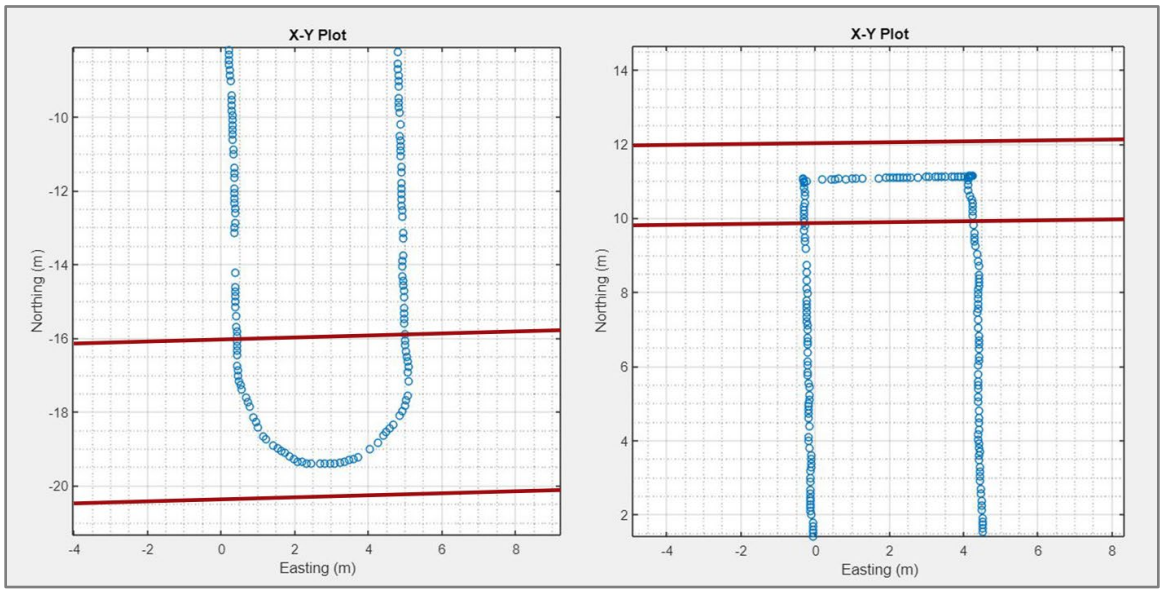


Figure 2.17: Recorded GPS data for front wheel headland turn (left) and four-wheel crab headland traverse (right). Coordinates translated to where data recording was initialized.

Certain fields also have restricted headland areas. An advantage to traverse turning is the significantly reduced amount of distance needed when compared to the front-wheel-steer method. Even with the tight turning radius of Flex-Ro, the front-wheel steer method required 4.5 m of headland to navigate back into the crop rows. Comparatively, the traverse method required headland width equal to the length of the machine wheelbase, 2 m.

2.3.3 Obstacle Reaction

The implemented obstacle detection system was evaluated as a proof of concept and not a comprehensive obstacle prevention system. This system was installed to aid the operator in the event of unforeseen obstacles, only in the forward direction. Initial tests were conducted to verify the detectors ability to repeatedly identify obstacles at a set x and y coordinates. The sensor was mounted to a small stationary platform for evaluation for

detecting an obstacle with a height and width of 100 cm and 40 cm, respectively. The results of this stationary test can be visualized in Figure 2.18. The x mean absolute error was 24 cm and the y mean absolute error was 14 cm.

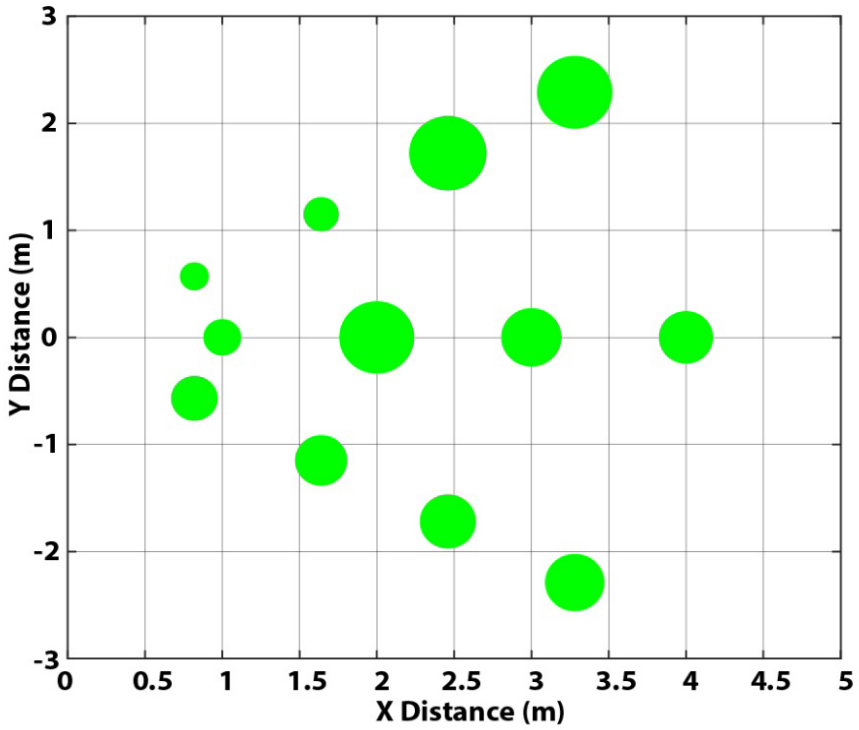


Figure 2.18: Mean error magnitude at obstacle set position recorded by ifm O3M 151 3D Smart Sensor.

The dynamic response of the obstacle detector and machine stopping was verified by approaching an obstacle at a set speed in automatic mode and measuring the resting distance of the machine to the obstacle (Figure 2.19).

Figure 2.19: Dynamic obstacle approach test. Stopping distance was measured from detector on front to board.

The size of the obstacle was 40 cm wide by 100 cm tall. The results of the dynamic obstacle approach test can be seen in Figure 2.20.

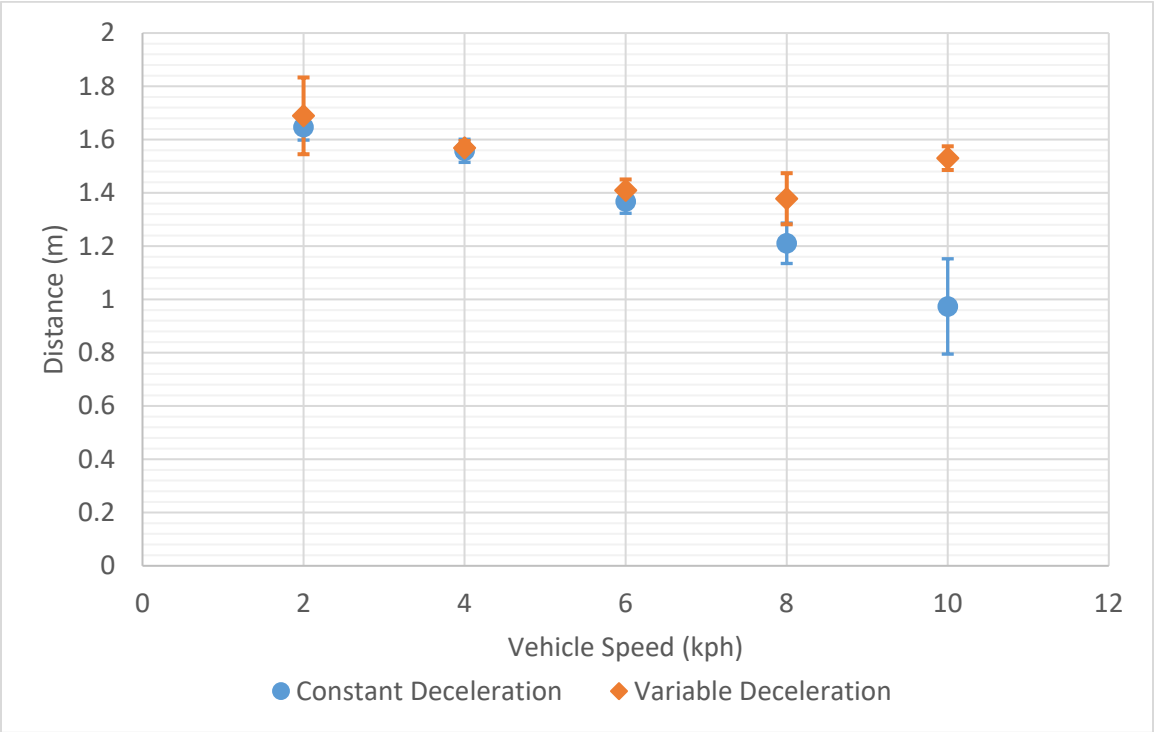


Figure 2.20: Measured distance from obstacle after machine automatically stops due to detected potential collision.

The constant deceleration points are the machine's response assuming initial conditions do not affect stopping distance. It was found, as swash plate commands and the engine RPM changed the machine deceleration factor varied. The distance stopped from the obstacle became uncomfortably close at higher approach speeds when using a constant deceleration factor. A deceleration rate which compensated for initial conditions resulted in a more consistent ability to stop at the desired distance away from the object. The implementation of the variable deceleration factor can be found in Appendix D.9. The improved stopping distances after implementing variable deceleration factors are shown in Figure 2.20.

2.4 Conclusions

The control system developed for the Flex-Ro machine met the defined objectives. Safety systems were designed and verified to ensure fail-safe operation, whether using the remote or autonomous control. Four e-stops around the machine and each control interface allows the user to manually stop any undesirable operation. Further, ECU intercommunication ensures all required controllers are on-line. Loss of communication with either the remote or one of the ECUs triggered an e-stop event. The six e-stop methods were tested with 100% success.

The MATLAB developed Flex-Ro Run application provided a means to programmatically control the machine. Successful navigation controlled the machine to track between 74.6 cm (30 in) crop rows. Further, waypoint navigation was implemented so the machine could complete fields autonomously. Field coverage was tested and verified in university research plots. The machine was programmed to follow pre-defined paths, enabling special operations such as driving the machine to a staging area once the field coverage was complete. Straight line and waypoint paths were navigated with 95% errors of 3.23 cm and 19.2 cm respectively while traveling at 4 kph.

Collision detection and reaction was implemented and evaluated in automatic mode in the primary direction of travel. An ifm O3M 151 3D Smart Sensor was installed on the front of the machine. The obstacle detector was added as a node on the existing machine J1939 CAN bus. The hydrostatic drive control ECU received the obstacle messages and controlled the machine speed accordingly. The system repeatedly identified and stopped

for pedestrian sized objects in a variety of environments, while approaching at normal field operating speeds.

Chapter 3 Flex Ro Phenotyping Application Evaluation

3.1 Introduction

Flex-Ro was developed to be a multi-purpose platform with many field applications. The first application to be implemented on the machine was for high-throughput plant phenotyping. Plant phenotyping is well established at the plot level (square meters), but high-resolution coverage at the field level (hectares) remains limited. A manual push-cart phenotyping sensor system was developed by researchers at the University of Nebraska-Lincoln (Bai et al., 2016). Significant labor requirements and a stop-measure-go data collection technique limited the productivity of the device. Migration of the sensor suite onto the Flex-Ro machine would capitalize on the machine's autonomous capability to cover large areas while unattended.

The University of Nebraska-Lincoln is uniquely positioned in advanced field phenotyping with the recent installation of the Spidercam phenotyping system (Bai et al., 2019). This system is a tested and calibrated suite of sensors which can be positioned anywhere over the designated one-acre plot. The sensing height ranges from 0 to 10 meters. A subsection of the one-acre plot coverage area was designated for use by the Flex-Ro machine. Researchers selected soybeans as it would enable Flex-Ro to operate through the growing season without clearance issues. At the end of the season, statistical comparisons between the Spidercam and Flex-Ro phenotyping data were to be completed. This data would validate that the phenotyping system installed on Flex-Ro would be suitable for large field use.

3.1.1 Chapter Objectives

Before the Flex-Ro is operated in large fields, it will be important to determine the validity of the data collection system.

1. Implement phenotyping sensor package on the Flex-Ro machine.
2. Compare phenotyping data collected with the Flex-Ro PhenoBar to the data collected using the UNL Spidercam.

3.1.2 Chapter Hypotheses

Hypotheses were formulated to statistically compare the Flex-Ro and Spidercam phenotyping systems.

1. The Flex-Ro phenotyping data collected will directly correlate with the measurements taken by the UNL Spidercam.
2. The Flex-Ro phenotyping data will show a statistically significant difference between genotypes and or treatments within the soybean field.

3.2 Materials and Methods

3.2.1 PhenoBar

There are three main components which make up the PhenoBar system. The sensor units, the toolbar which holds the sensor units, and the height adjustment mechanism of the toolbar. The toolbar and adjustment mechanism were designed to accommodate a variety of applications beyond phenotyping. The toolbar itself is constructed of t-slot extruded aluminum. The four-bar adjustment mechanism keeps the sensor units parallel to the

ground and enables a wide range of height settings. The PhenoBar is shown mounted to the Flex-Ro in Figure 3.1.

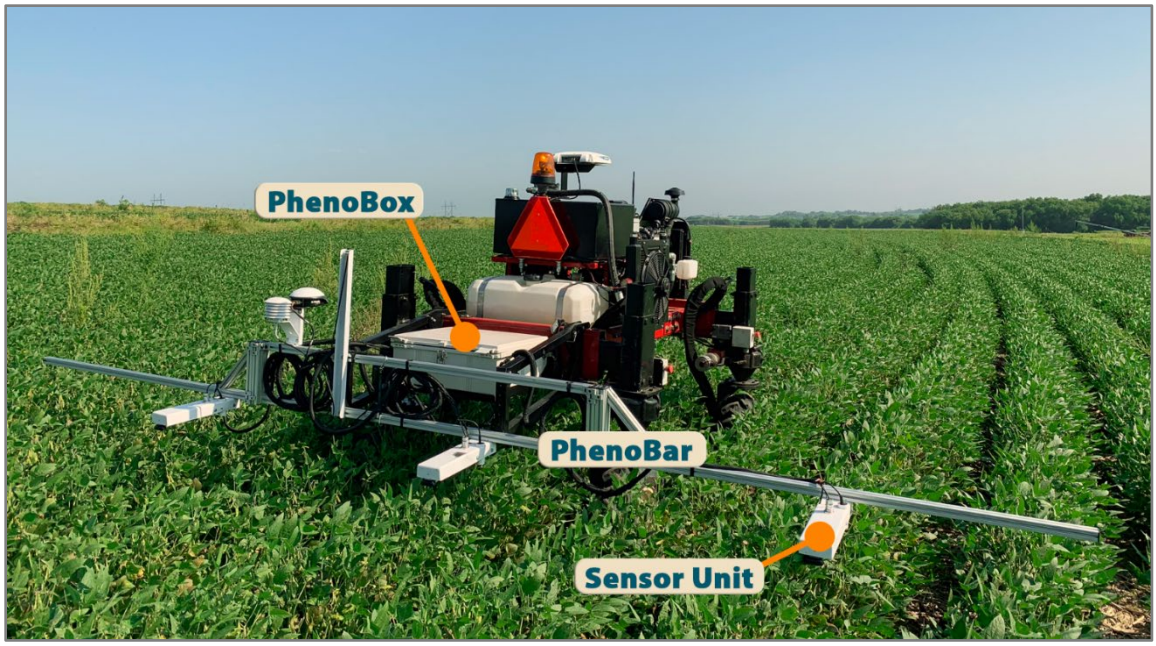


Figure 3.1: PhenoBar mounted to the Flex-Ro machine. Three sensor units cover a 4.5m swath.

The suite of sensors installed onto the Flex-Ro machine were selected from those used by Bai et. al. (2016). Three sensor units were downward facing to record data from a crop row. Each of these units included a passive fiber optic, a red green blue (RGB) camera, an ultrasonic distance sensor and an infrared radiometer. Also installed was a global

positioning system (GPS) for georeferencing data points. The sensors and related specifications are listed in Table 3.1.

Table 3.1: List of sensors installed in each unit on the Flex-Ro PhenoBar.

Sensor	Model and Manufacturer	Output	Phenotype Measured
Ultrasonic	ToughSonic30, Senix Corporation, Hinesburg, Vermont	Analog Voltage	Height
Spectrometer	CCS175, Thorlabs Inc, Newton, New Jersey	Digital USB	Reflectance Spectra
RGB Camera	C270, Logitech, Newark, California	USB	RGB Images
Infrared Radiometer	SI-131 Apogee Instruments, Logan Utah	Analog	Canopy Temperature

3.2.2 PhenoBox

The PhenoBox houses the data acquisition hardware for the Flex-Ro phenotyping system. The box distributed power to the required components; the LabJack U6, Startech 10-Port USB Hub, Thorlabs Spectrometers, power inverter and laptop. Connectors and passthroughs facilitate connection to the three crop sensor units and GPS. The machine CAN bus was also routed to the PhenoBox to connect with the laptop. This would allow future development of a CAN bus phenotype sensor network or other CAN enabled applications. The PhenoBox (Figure 3.2) was designed to be modular to allow for additional and or different sensors. The pinout of the PhenoBox connectors is given in Appendix C.3.

Figure 3.2: PhenoBox installed with laptop tray which contains data acquisition hardware for the Flex-Ro phenotyping system.

3.2.3 Data Processing Methods

3.2.3.1 LabVIEW

A LabVIEW Virtual Instrument (VI) was developed as an extension of the work completed by Bai et al. (2016). The collection method used by Bai et al. (2016) was a stop-measure-go method, in which the manual pushcart was moved, and a new data point was captured. This process was labor intensive and would not capitalize on the benefits of an autonomous machine for high throughput phenotyping. Automatic cyclic data collection would be better suited for the Flex-Ro. The machine was to travel slowly, and data points would be periodically recorded. Primary concerns of recording while in

motion were to eliminate blur in the pictures and ensure that the geotagged data points were synchronized.

Three main subsections of data collection were implemented in the LabVIEW VI. These included image capture, spectrometer data recording, and logging geotagged analog sensor data. The data was processed and displayed on the VI's front panel. A button on the front panel enabled writing of the data to file. Each data point was assigned an ID, such that images, spectra, and analog sensor data could be merged during post processing. The recording rate was limited by the image capture time. The basic snapshot NI-IMAQ sub-VI was used sequentially for each camera.

The spectral reflectance data was saved as a text file, with each column relating to a specific wavelength. The spectrometer has a range of 500-1100 nm with a step size of 0.16 nm. The wavelength data array is saved for each spectrometer during sensor initialization. Although each of the spectrometers capture the same range of reflectance, the specific recorded wavelength values vary slightly. This data is combined for viewing on the VI front panel and used during post processing.

The analog sensors included on the PhenoBar include the ultrasonic height sensors and infrared radiometers used for measuring the canopy temperature. This data was split up and assigned an ID and position on the boom. The GPS coordinates were transformed to each sensor unit based on the current heading of the machine and its position on the toolbar. The GPS points were georeferenced to the image and spectral data during post processing.

The LabVIEW front panel, as shown in Figure 3.3, was designed to facilitate data monitoring and recording. A file directory is specified upon program initialization, and all files generated are saved within that directory.

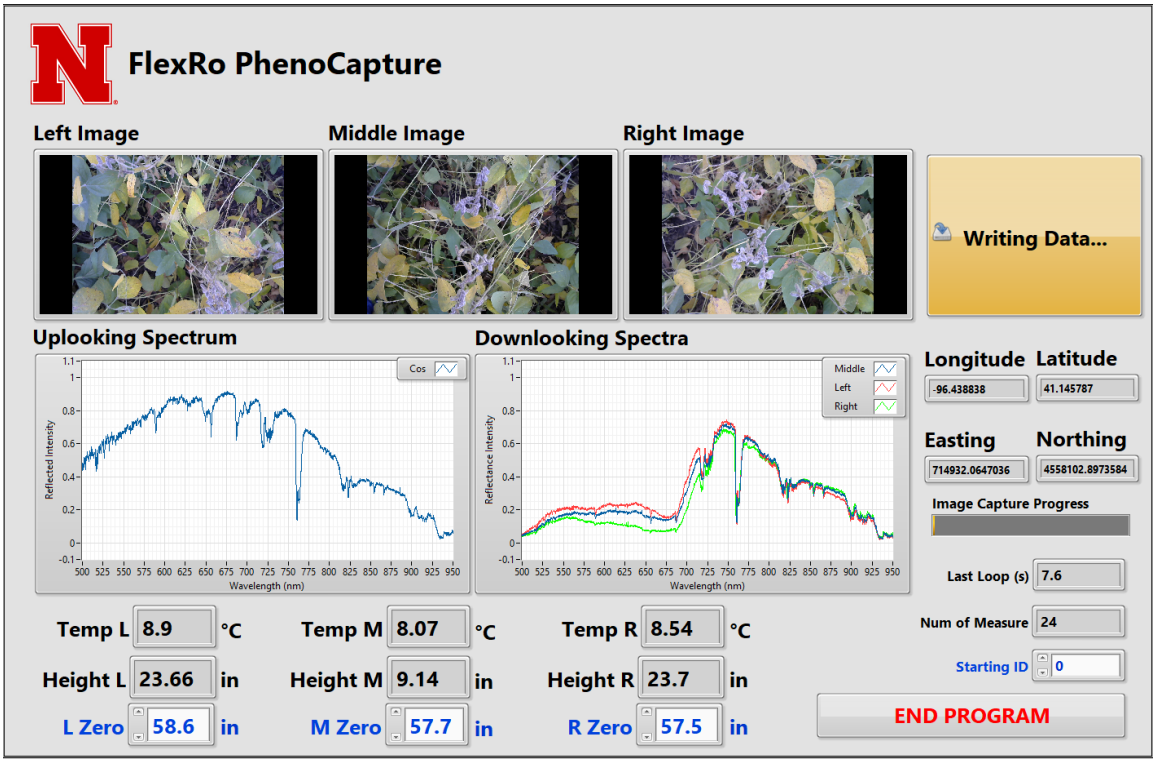


Figure 3.3: LabVIEW front panel used as the phenotyping data acquisition system for the Flex-Ro.

3.2.3.2 MATLAB Post Processing Tool

The three data files associated with a phenotyping data collection event need to be combined. A program developed using MATLAB App Designer processes and synchronizes the data. The PhenoCalc app's Field Summary tab is shown in Figure 3.4 and Plot Summary Tab is shown in Figure 3.5.

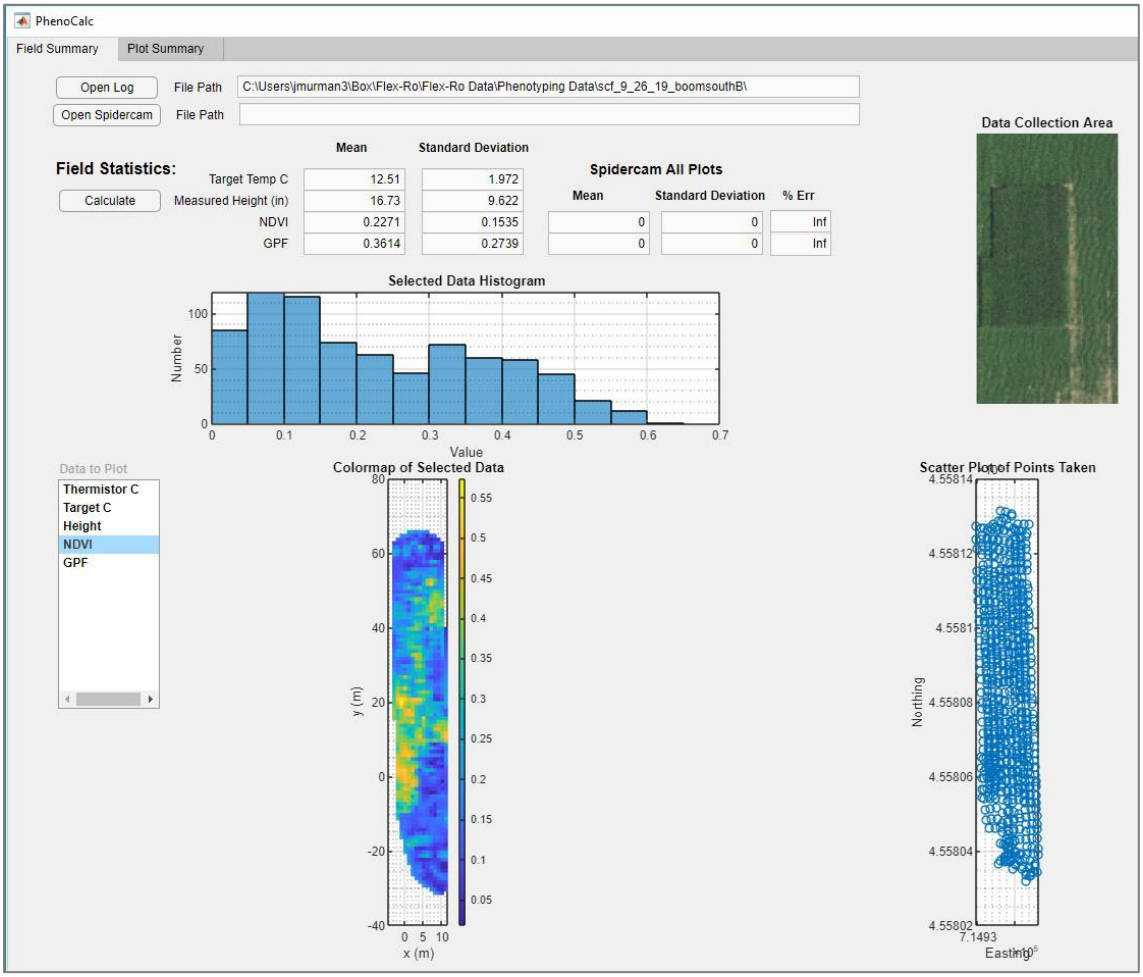


Figure 3.4: Custom PhenoCalc application Field Summary tab to process raw Flex-Ro captured data.

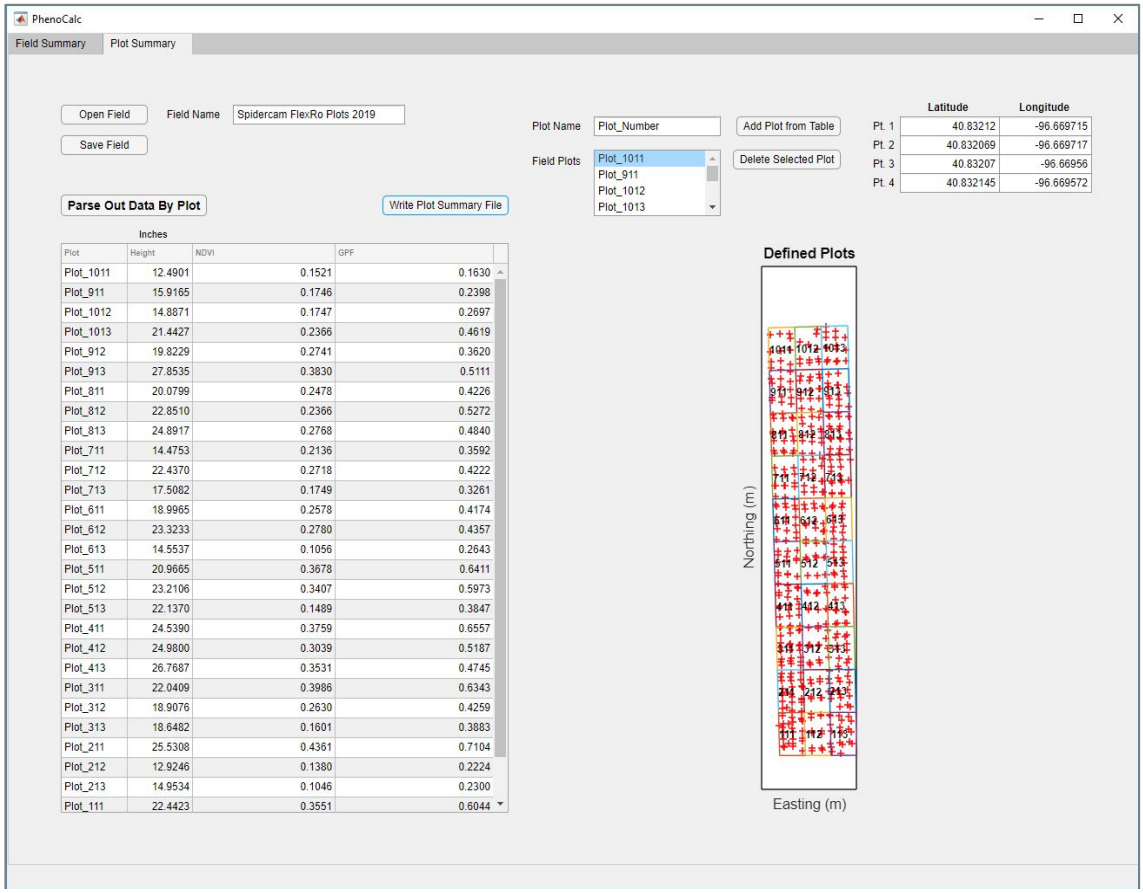


Figure 3.5: Custom PhenoCalc application Plot Summary tab used to parse per plot data from large matrix.

The spectral data is used for calculating vegetation indices. Previously studied indices include NDVI (Normalized Difference Vegetation Index), NDRE (Normalized Difference Red Edge Index), PRI (Photochemical Reflectance Index), and SIF (Solar Induced Fluorescence) (Bai et al., 2019). The indices are a ratio based on the magnitude of spectral reflectance at different wavelengths. NDVI was used for the initial tests of the Flex-Ro phenotyping system and could be compared directly to the Spidercam. The MATLAB program calculates the NDVI using the following equation.

Equation 3.1: Calculating the NDVI using the magnitude of reflectance at 705 and 750 nm.

$$NDVI = \frac{\rho_{750} - \rho_{705}}{\rho_{750} + \rho_{705}}$$

Variables ρ_{750} & ρ_{705} are the magnitudes of reflectance at wavelengths of 750nm and 705nm respectively.

The RGB images are used to estimate canopy coverage by calculating the green pixel fraction (GPF). A segmentation algorithm calculates the GPF based on the method presented by Bai et al., (2016). The image is converted to the L*a*b color space, then Otsu's method is used to threshold the image using the 'a' channel. The pixels remaining after thresholding divided by the total number of pixels results in the GPF.

The calculated NDVI and GPF are combined with the analog data points and GPS coordinates to form a master list. This master list can be used to calculate averages and generate an interpolated color map for the field of the different traits. The portion of the Spidercam field used for this research contained 30 plots, with 5 genotypes and 2 different treatments. The data recorded by the Flex-Ro platform was parsed for individual plot analysis. This was completed using the master list of data, and plot polygons generated from field recorded GPS points. MATLAB provides a function (inpolygon) which selects only points contained within a polygon. After parsing, the points in each plot were averaged and saved to file for further comparison. These plot averages could be directly compared to the Spidercam generated data.

3.2.4 Field Data Collection Strategy

As mentioned, to achieve maximum throughput using the Flex-Ro platform, continuous data was collected. Due to the small size of the research plots and long sample times, a speed of 1 kph was used. The three sensor units were positioned directly over a row, each 60 inches apart as shown in Figure 3.6. The plot was covered in two swaths, offset by one row to achieve high resolution total plot coverage. This ensured a sensor unit would pass over each row, and a representative average was recorded. In contrast, the Spidercam takes just one data point over the entire plot.

Figure 3.6: The Flex-Ro collecting data 67 days after planting (DAP). Each of the sensor units are positioned directly over a row.

The spectrometers, which rely on the intensity of reflected light, are sensitive to changing conditions. While shadows can't be eliminated using this system, methods can be used to

reduce error. The Flex-Ro machine recorded phenotyping data with the boom always on the South side (the research plot rows ran north and south). The angle of the sun's rays through the growing season reduced the amount of shadows present. Further, the data collected had the same steady state condition and could be compared directly.

3.3 Results and Discussion

The Flex-Ro collected field phenotyping data seven times during 2019 growing season. The research objective was to validate the Flex-Ro phenotyping system by comparing it to the developed and tested Spidercam (SPC) phenotyping system. Thirty soybean plots over 0.21 acres were selected for the study. Within these thirty plots, there were five genotypes and two irrigation treatments. The five soybean varieties represented a range of maturities. The two irrigation treatments were full and deficit.

The Spidercam phenotyping system was previously tested and verified with ground truth measurements (Bai et al., 2019). The Flex-Ro data resulted in plot average canopy height, canopy coverage, and NDVI. The canopy temperature sensors on the Flex-Ro were found to be out of calibration and were not used for comparison.

3.3.1.1 Crop Height

Canopy height measurements over time show crop growth curves. The Flex-Ro PhenoBar used an ultrasonic sensor positioned directly over the crop row to measure crop height. The height of the crop was calculated by subtracting the sensor value from the measured distance to bare ground.

The Spidercam calculated plot height using a LiDAR sensor. First, the distance to the soil was measured at the start of the growing season. During the growing season, the 10th percentile distance within the plot as measured by the LiDAR was recorded. Subtracting these measurements resulted in the average canopy height (Bai et al., 2019). The correlation between plot height averages between the Flex-Ro and Spidercam phenotyping systems can be seen in Figure 3.7.

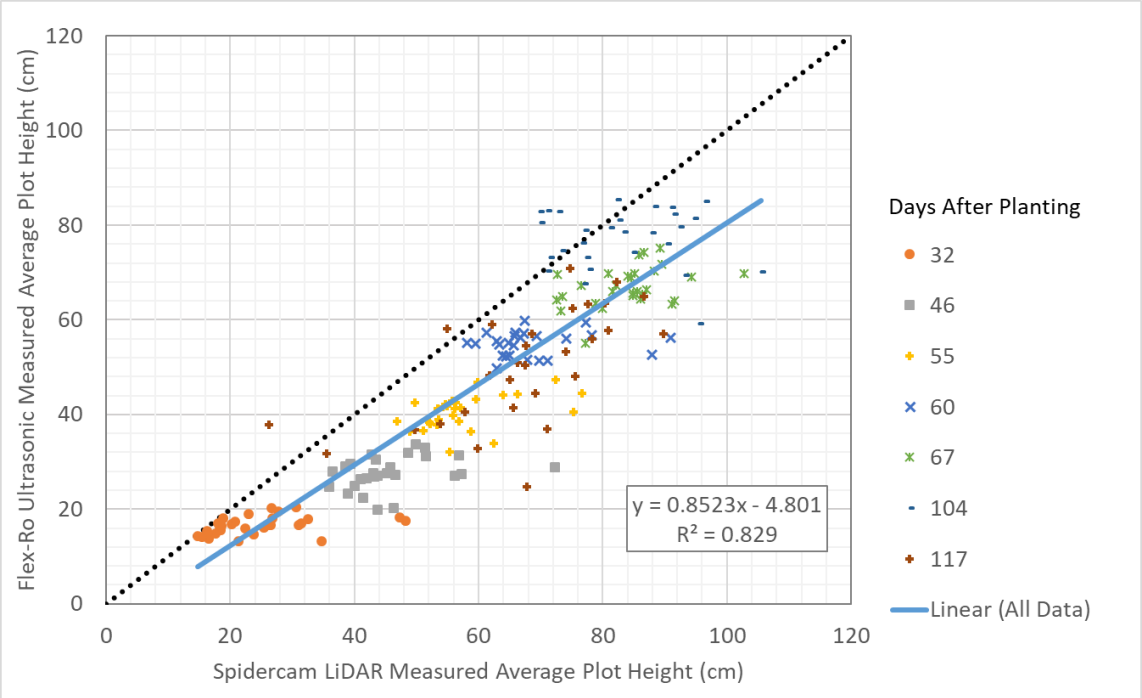


Figure 3.7: Correlation between Flex-Ro and Spidercam measured crop plot average height.

The ultrasonic sensor is highly susceptible to variations in canopy density. This explains the offset in correlation to the Spidercam measured height. While this difference is present, the slope of the correlation is near one at 0.8523 ($R^2 = 0.829$), indicating that both systems are measuring corresponding changes of height over time. Height measurement accuracy was reduced as the crop neared maturity and began to defoliate.

Ground truth plot height data was recorded four times throughout the growing season using a yardstick. Four measurements were taken in each plot and averaged. Figure 3.8 shows the comparison between the Spidercam LiDAR, ground truth (GND), and Flex-Ro ultrasonic height measurements. The data presented is the crop plot averages separated by irrigation treatments. The Spidercam data (SPC) tends to over-estimate height compared to the ground truth while the Flex-Ro measurement is near the same range until divergence around 67 days after planting (DAP).

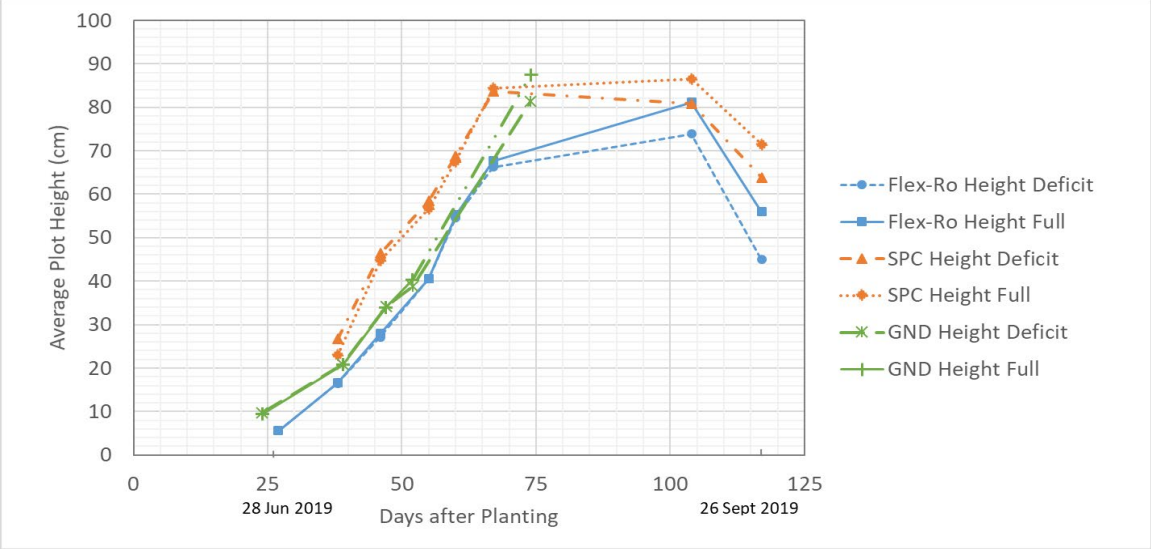


Figure 3.8: Temporal comparison of crop plot height averages per treatment with different measurement techniques.

Figure 3.9 shows the temporal canopy height measurements by the Flex-Ro between genotypes. Only the replications in the full irrigation treatment zones are plotted.

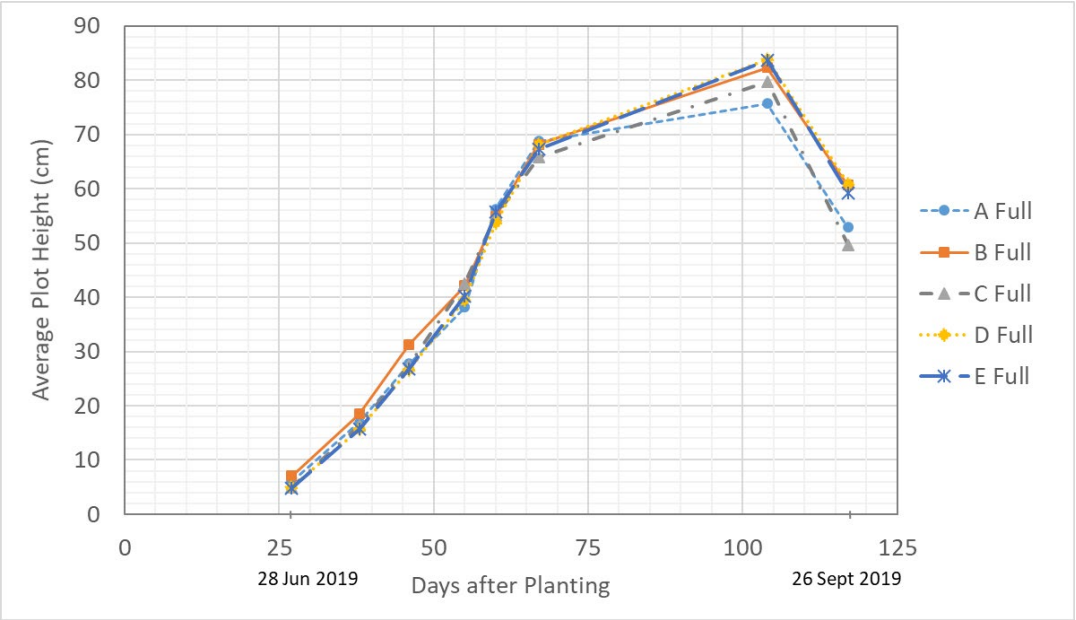


Figure 3.9: Average plot height over time per genotype with full irrigation treatment as recorded by the Flex-Ro platform.

3.3.1.2 NDVI

The Flex-Ro and Spidercam both used spectrometers connected to a downward pointing fiber-optic cable to measure reflectance. The spectrometer measures the spectral reflectance across a range of wavelengths (500 – 1100 nm). The PhenoCalc post processing tool used the recorded wavelength and corresponding reflectance to calculate the NDVI. The wavelengths used to calculate the Flex-Ro and Spidercam NDVI were 750nm and 705nm. The Spidercam used the same wavelengths for the NDVI calculation. Plot averages of NDVI values were compared and correlated (Figure 3.10). The linear fit resulted in a slope of 0.7472 and R^2 value of 0.7814.

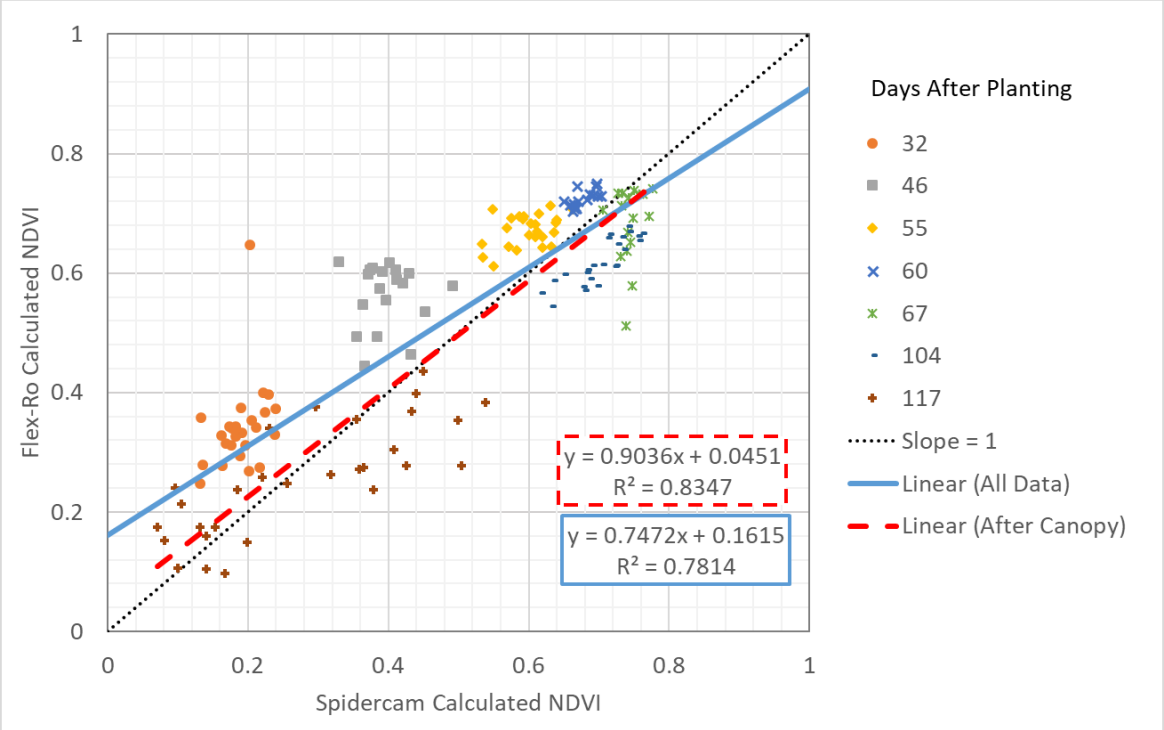


Figure 3.10: Correlation between Flex-Ro and Spidercam measured NDVI and linear correlation.

One reason for error between the two systems is the proximity of the fiber optic cable end to the plant. The Flex-Ro positions the sensor ~1m directly above the row. The sensor is designed to capture data for just that one row. As a result, the reflectance data is more influenced from the crop itself and less from the surrounding soil. Comparatively, the Spidercam is positioned 5m above the plot, and is designed to capture the whole plot in one data point. Thus, especially during the early season, a significant amount of soil will be within the reflectance measurement, and the intensity from the crop will be less. Removing the early season (32 and 46 DAP) NDVI data and recalculating the linear relationship resulted in a slope of 0.9036 and R^2 value of 0.8347. This new linear fit indicated the SPC and Flex-Ro systems recorded very similar NDVI measurements after the crop canopy was more established.

The Flex-Ro may be more capable of detecting early season phenotypic differences with the proximity of the sensor bar. However, care must be taken to ensure the sensor unit is directly over the row, or the reflectance intensity could vary significantly. The NDVI could also be used to identify growth trends between genotypes and treatments. This data could show early season vigor or response to drought. The NDVI of the two treatments are plotted over time as recorded by the Flex-Ro and Spidercam phenotyping systems (Figure 3.11).

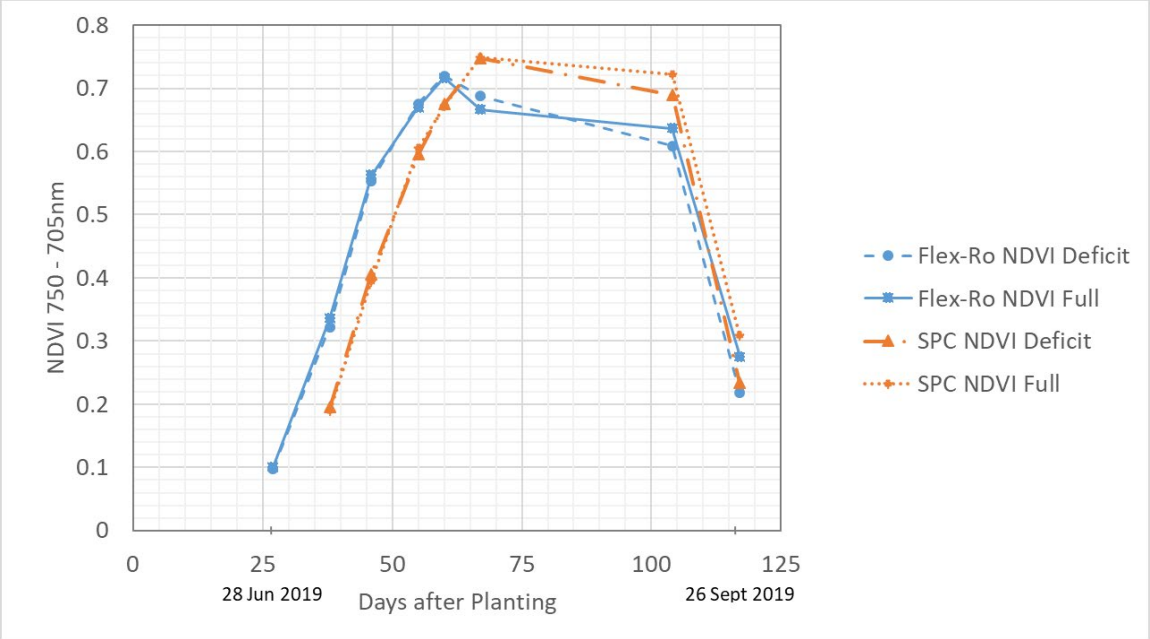


Figure 3.11: Temporal comparison of NDVI split into the two field treatments comparing phenotyping systems.

Figure 3.12 shows the comparison between fully irrigated genotypes over time as recorded by Flex-Ro. The dip in NDVI at 67 DAP was likely in response to changing the height of the Flex-Ro PhenoBar. The height was changed to maintain a consistent toolbar

distance over the crop through the growing season. This would result in less intense reflectance and corresponding NDVI. Lighting conditions may have also been a factor.

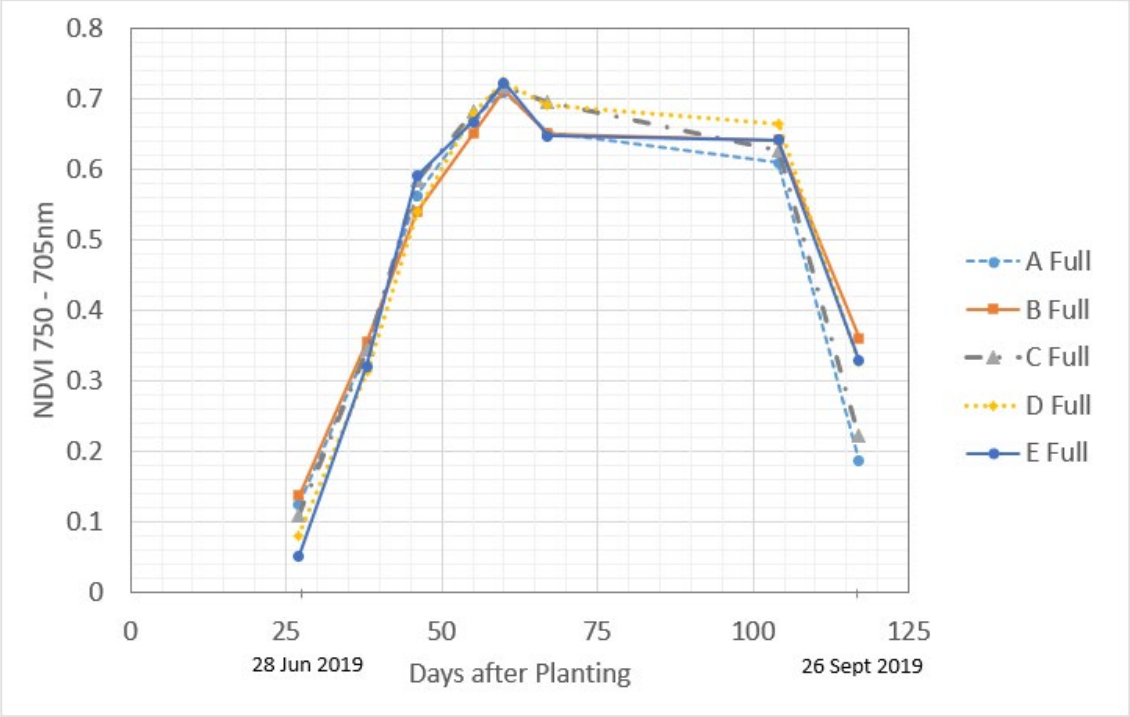


Figure 3.12: NDVI (750-705 nm) as calculated from Flex-Ro data per genotype with full irrigation treatment.

3.3.1.3 Canopy Coverage

The Flex-Ro and Spidercam differed in the calculation of crop canopy coverage. The Flex-Ro used an RGB camera and processed the images to determine the green pixel fraction (GPF). The number of green pixels divided by the total number of pixels in the image resulted in an approximation of the percentage of plant in the image. The image segmentation worked well in the early season, when there was minimal shading within the canopy and a clear contrast between foliage and soil. However, as the season

progressed, the segmentation algorithm had difficulty distinguishing between shaded area within the plant and soil. Two examples of segmentation are shown in Figure 3.13.

Figure 3.13: Examples showing result of image segmentation to calculate GPF. Segmented 'non-green' pixels shown in orange for clarity.

The distance of the camera above the crop also affected the calculated canopy coverage. The data set taken at 67 days after planting (DAP) had a higher boom height than the data set taken on 60 DAP. Raising the boom widens the view frame, and in this case, the camera was able to capture more soil. This resulted in a similar calculated canopy coverage, when the Spidercam showed a linear progression.

The Spidercam calculated the crop coverage using a multi-spectral camera. An NDVI image was created using the NIR and red images. As a result, the soil could be easily segmented. This resulted in accurate segmentation, as the multispectral images covered most of the plot.

Otsu's segmentation method had significantly reduced performance for near full canopy when compared to corresponding Spidercam canopy coverage data. The fit between the Spidercam and Flex-Ro had as slope of 0.5862 with an R^2 value of 0.643 (Figure 3.14).

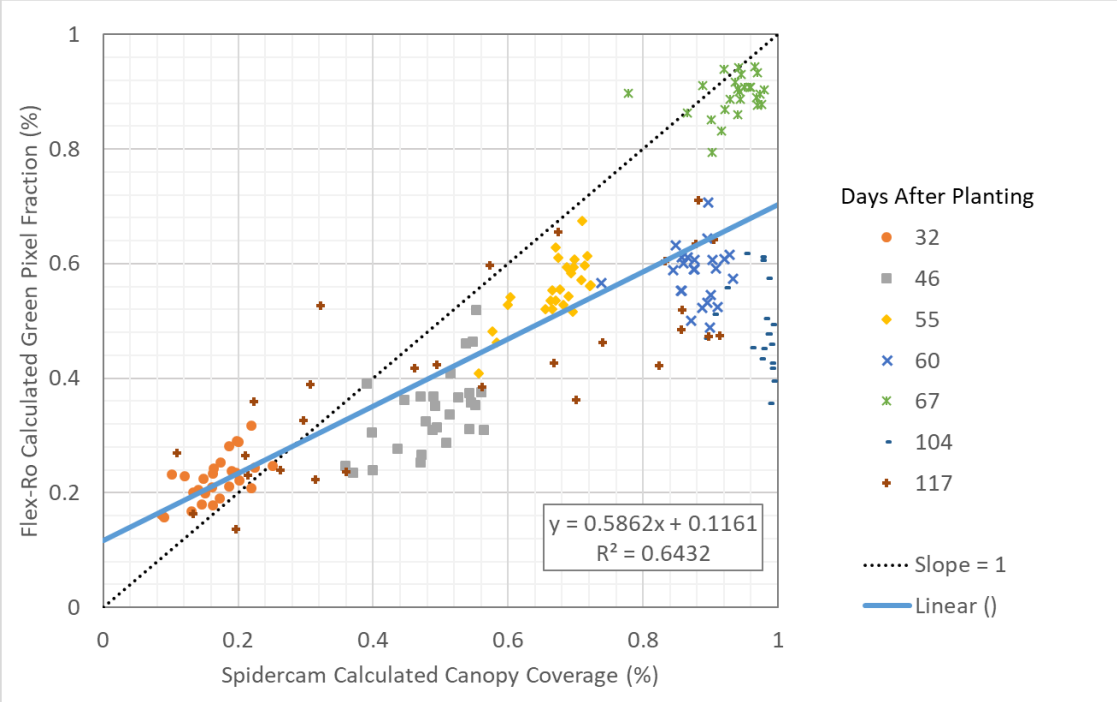


Figure 3.14: Correlation between Flex-Ro and Spidercam calculated crop canopy coverage.

Figure 3.15 shows the temporal comparison between the canopy coverage from the Flex-Ro and Spidercam phenotyping systems. Plot averages are separated into the different

irrigation treatments. Erroneous points due to segmentation difficulty can clearly be seen at 60 and 104 days after planting.

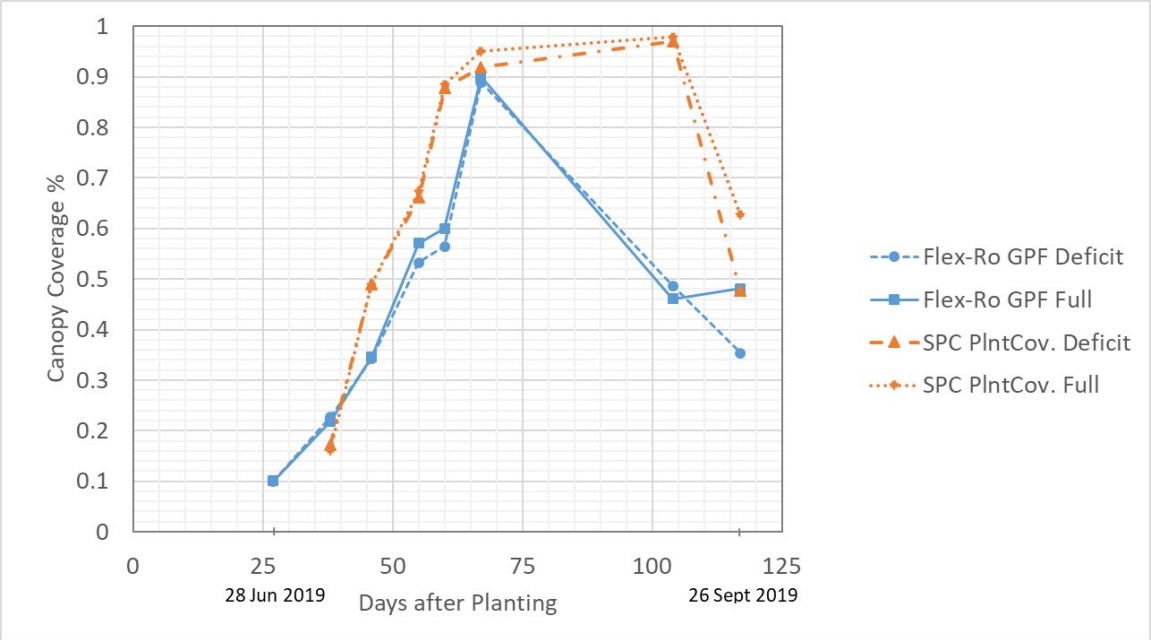


Figure 3.15: Temporal comparison of canopy coverages split into field treatments comparing phenotyping systems.

Figure 3.16 compares the canopy coverage of the different genotypes with full irrigation treatment. Data during the early season can be correlated to emergence characteristics. As the plots neared full canopy (after 60 DAP) the segmentation algorithm resulted in erroneous data.

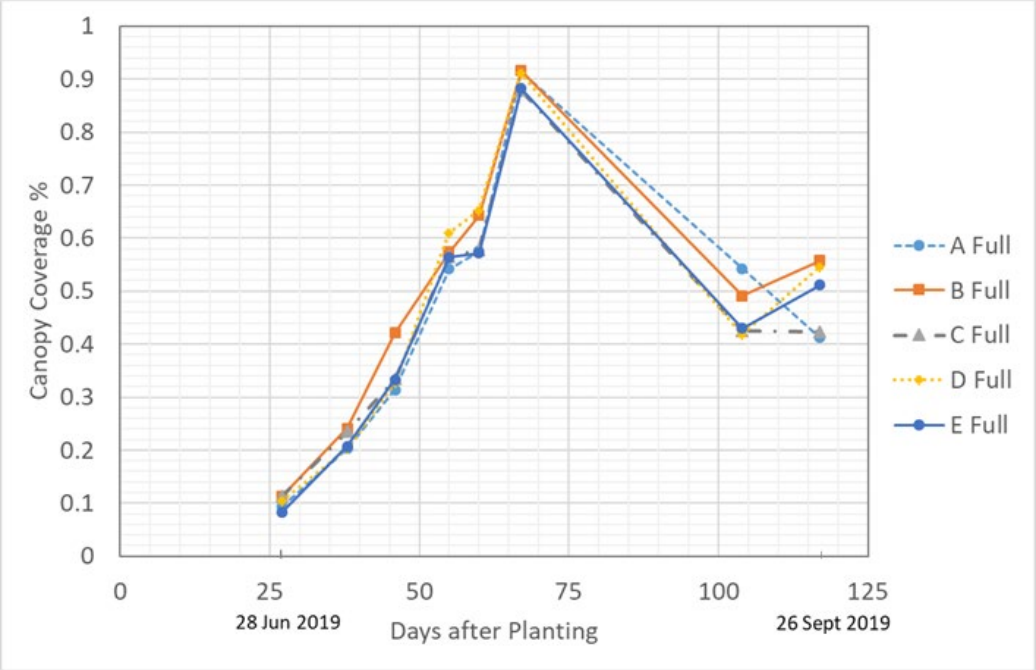


Figure 3.16: Canopy coverage calculated from Flex-Ro data over time per genotype with full irrigation treatment.

3.3.2 Identification of Treatments and Genotypes using Flex-Ro

The field experiment utilized a randomized complete block design. The two treatments within the soybean plots were full and deficit irrigation. The five genotypes had six repetitions which were split into the two irrigation treatments. The start of 2019 was abnormally wet, receiving consistent rainfall well into July.

Analysis of variance (ANOVA) is used to test for the existence of differences between treatments. An ANOVA table was calculated for the Flex-Ro and Spidercam data recorded at each of the dates. The ANOVA table was set-up to indicate whether the phenotype data indicated a statistical difference between the genotypes or irrigation

treatments. If a difference was detected within the genotypes, a pairwise comparison was completed to show which genotypes were statistically different. Only the early and late season measurements indicated differences between either genotypes or treatments. Table 3.2 shows the results of the ANOVA tests and significant differences measured. (DAP measurements without significance are not shown)

Table 3.2: Results of testing for statistical difference in recorded phenotyping data means of the SPC and Flex-Ro phenotyping systems.

	DAP	27		104		117	
	Groupings	Genotype	Treatment	Genotype	Treatment	Genotype	Treatment
Height	Flex-Ro	*	-	*	***	-	**
	Statistic Difference	E < B		A < E	L < F		L < F
	Spidercam	-	-	-	-	-	-
	Statistic Difference						
NDVI	Flex-Ro	*	-	*	*	**	*
	Statistic Difference	E < B		A < D,E	L < F	A < D,E	L < F
	Spidercam	-	-	**	**	***	*
	Statistic Difference			A < D,E	L < F	A < D,E & C < E	L < F
Canopy Cov.	Flex-Ro	***	-	-	-	-	**
	Statistic Difference	E < B					L < F
	Spidercam	-	-	-	-	**	*
	Statistic Difference					A < D,E & C < E	L < F

Significance Level: * 0.05, ** 0.01, *** 0.001

If significance was detected the difference is listed between genotypes or treatments. Only the periods with statistically significant differences are shown. The test of interaction between treatments and genotypes showed no statistical significance.

Table 3.3: Genotypes and irrigation treatments. ' - soy seed brand 1, " - soy seed brand 2.

Genotype	Maturity	Emergence	Irrigation Treatment	
A'	2.6	7	L	Deficit
B'	2.7	6	F	Full
C"	2.9	1		
D'	3.3	8		
E"	3.6	2		

Table 3.3 lists the genotypes and growth characteristics as provided in brand literature. The Flex-Ro Phenotyping system was able to statistically decipher different emergence characteristics in the early season. Maturity differences became apparent at the end of the season and were statistically identified using both the Flex-Ro and Spidercam. Genotypes with a higher maturity number keep their leaves longer and was verified with the ANOVA tests, the variation in maturity can be seen in Figure 3.17. The irrigation treatment differences were also statistically identified towards the end of the growing season by the Flex-Ro and Spidercam.

Figure 3.17: Flex-Ro in the Spidercam research field collecting data at 117 DAP. Differences in maturity can clearly be seen between plots.

3.3.2.1 Yield Correlation

The final plot yields adjusted to 13% standard moisture were tabulated and correlated to the phenotypic measurements recorded throughout the growing season. Data for both Flex-Ro and Spidercam were compared. Each of the 30 plots phenotyping data averages

were separately correlated to its yield. The results and statistical significance are shown in Table 3.4. A visual representation of canopy coverage and its correlation to yield over time is given in Figure 3.18.

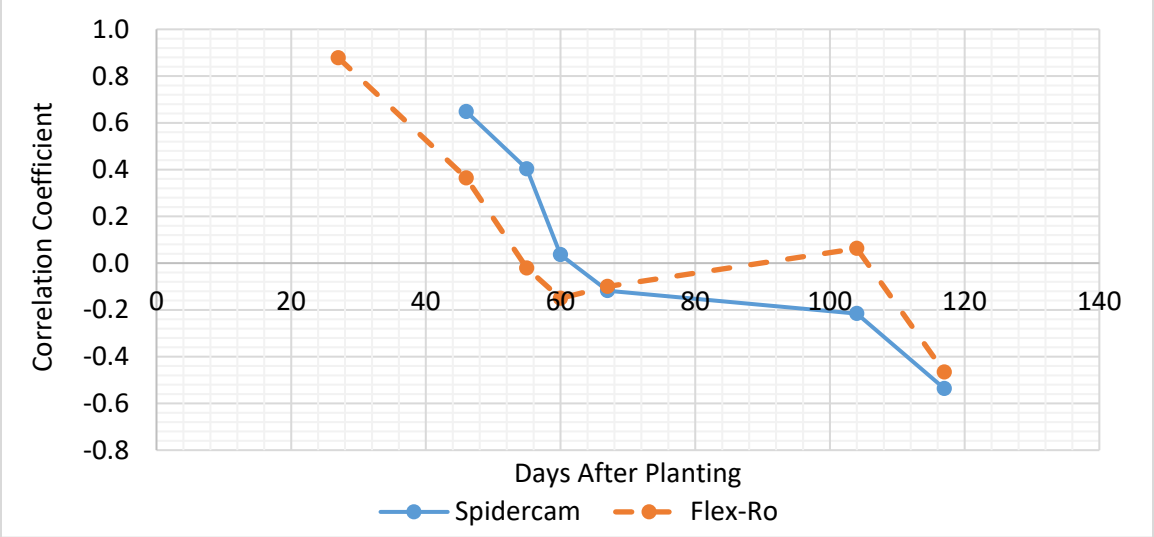


Figure 3.18: Correlation coefficient of plot canopy coverage to yield over time of the Flex-Ro and Spidercam.

Table 3.4: Correlations between recorded data and final plot yield with significance level.

		Canopy Coverage		NDVI		Height	
Date	DAP	SPC	Flex-Ro	SPC	Flex-Ro	SPC	Flex-Ro
28-Jun-19	27	-	0.878 ***	-	0.517 **	-	0.718 ***
17-Jul-19	46	0.648 ***	0.364	0.470 *	-0.007	0.192	0.258
25-Jul-19	55	0.403 *	-0.022	0.430 *	0.237	0.133	0.429 *
31-Jul-19	60	0.036	-0.151	0.285	0.018	0.132	0.418
7-Aug-19	67	-0.118	-0.100	0.046	-0.074	0.201	0.235
13-Sep-19	104	-0.216	0.063	-0.572 ***	-0.433 *	0.012	-0.238
26-Sep-19	117	-0.537 **	-0.466 **	-0.637 ***	-0.589 ***	-0.268	-0.426 *
Significance Level:		0.001 ***	0.01 **	0.05 *			

The Flex-Ro shows high correlation for all measurements during the early season (27 DAP) which may indicate that early season vigor is strongly correlated to yield (Figure 3.19). The Flex-Ro canopy coverage (during the early season when the segmentation worked well) correlated especially strongly with final yield. The Spidercam data at 27 DAP was not available.

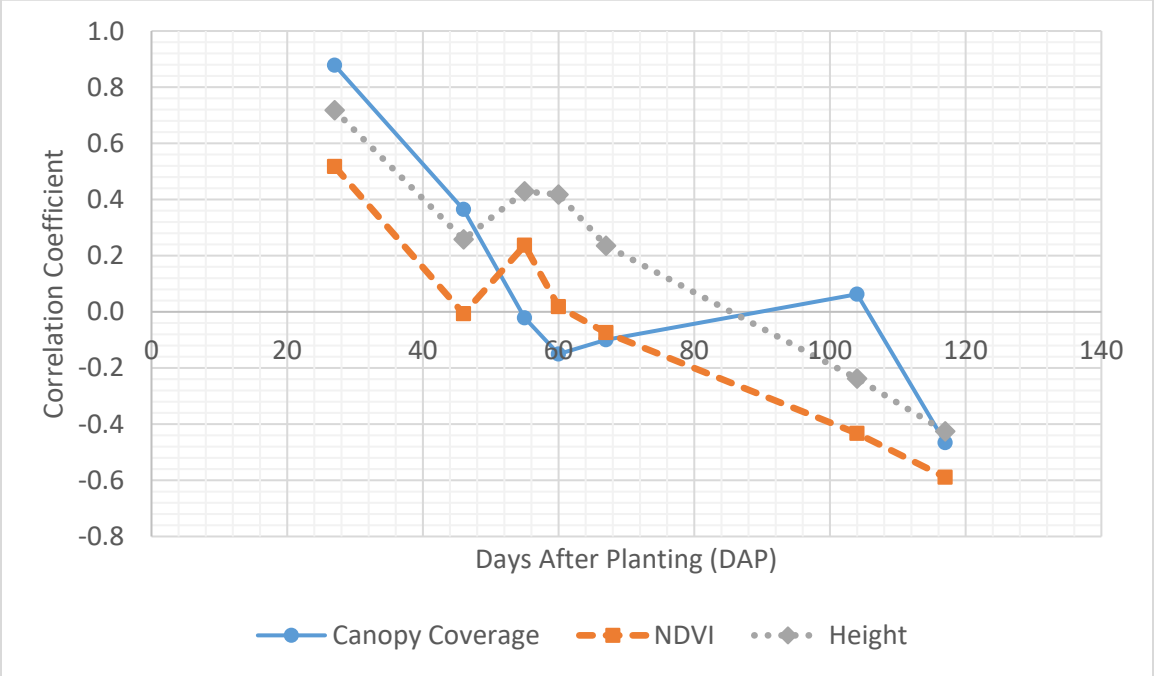


Figure 3.19: Correlation coefficient of phenotype data measured by the Flex-Ro to plot yield.

At the end of the growing season, significant correlations are negative with respect to yield. The plots with a shorter maturity (lower canopy coverage, and NDVI) had a higher yield in this experiment. The Flex-Ro and Spidercam phenotyping systems both significantly showed these correlations.

3.4 Conclusions

The Flex-Ro phenotyping system was evaluated during the 2019 growing season in soybean research plots. Phenotyping traits measured included NDVI, canopy coverage and crop height. While correlations could be drawn between the Spidercam and Flex-Ro data, a slope of one was not established. The error within the measurements resulted from using different methods to extract the phenotyping data. For example, the crop canopy coverage was calculated using RGB images from Flex-Ro and multi-spectral images from the Spidercam. The other main operational difference is that the Flex-Ro continuously moves during data collection in comparison to the Spidercam which covers each plot in one image.

The Flex-Ro phenotyping system was able to statistically differentiate between five genotypes and two treatments within the research field. The variation in emergence and maturity of the soybean varieties correlated to the differences measured by the Flex-Ro and Spidercam phenotyping systems. The final yield of each plot could be positively correlated to phenotyping data at 27 DAP and negatively correlated at the end of the growing season. The soybeans with the shorter maturity rating yielded better in this experiment.

The Flex-Ro phenotyping system was developed to continuously record data over a large area. This data was proven statistically significant, but not directly matched to the Spidercam system. The Flex-Ro requires further agronomic experimentation to verify its usefulness over large field areas.

Chapter 4 Flex-Ro Operational Power Requirements and Cost Estimation

4.1 Introduction

The economics of farming operations depends on several factors. Capital machinery costs are driven by the size of the farming operation and the ability to meet critical operating time windows. Determining the operating costs of machinery on a per area basis helps inform purchasing decisions. Modern technologies are adopted if they are proven to add value to an operation. The cost of the Flex-Ro machine for phenotyping or field scouting will determine the value which must be extracted from the high-resolution data.

The popularization of CAN bus led to the development of a variety of methods to collect, view, and store machine and agronomic data. Post processing of the CAN data leads to insights in the machine operating parameters. Direct input costs, such as fuel use, can be calculated, as well as other metrics, including field efficiency to determine machine utilization. Real time CAN bus monitoring gives users the ability to track the machine and its vitals, as well as monitor the current operation.

4.1.1 Chapter Objectives

1. Evaluate power required for low draft operations, such as phenotyping.
2. Determine an estimated cost of operation for the Flex-Ro platform.

4.2 Materials and Methods

The CAN bus is the primary on-vehicle transmission pipeline of machine information. Several tools have been developed to record this data, with varying degrees of functionality. The levels of CAN loggers are as follows. The basic logger will process the incoming data into raw hexadecimal format. The data is not human readable but can be parsed out if a description of the CAN message contents is available. Another type of CAN logger will process the raw CAN frames into engineering units, as specified by an accompanying database file. This data is usually stored locally for further post processing. Services exist to provide processed CAN data in real-time to operators and managers. Machine signals are monitored via tablet or laptop and are also recorded for report generation.

4.2.1 Machine Data Collection

4.2.1.1 Flex-Ro Run Data Collection

The FlexRoRun application described in Chapter 2 included a built-in data recording method. The data included local navigation information (e.g. lateral error), as well as machine information (e.g. engine speed). The recording rate was 5 Hz. The processed data and timestamps were saved to a local file for analysis.

4.2.1.2 CAN Bus Data Collection

The Flex-Ro's raw CAN bus data was collected with a Kvaser Memorator Pro 2xHS v2 device (Figure 4.1). The device connected directly to the machine's J1939 standard

diagnostics port. All CAN data transmitted was logged locally using a memory card. Recording rates vary and are set by the user. Filters can also be implemented to restrict the messages which are saved to the device. The data was post processed using a CAN database file.

4.2.1.3 Farmobile Data

A Farmobile PUC (Passive Uplink Connection) (Farmobile, Leawood KS) (Figure 4.1) displays and records CAN data processed into engineering units. The PUC connects directly to the machine J1939 diagnostic port. Uploaded information can be viewed and analyzed using the Farmobile DataEngine. The data is streamed to the Farmobile cloud using included cellular network connectivity. The device also utilizes a GPS antenna for data point georeferencing.

Figure 4.1: Left: Farmobile PUC data streaming device. Right: Kvaser Memorator Pro 2xHS v2 CAN logger.

4.2.2 Phenotyping Power Use

Phenotyping itself is a completely passive activity, meaning the only power required is for running the electronics. The most significant power use, then, is the tractive effort to move the machine through the field and the steady state parasitic losses. Activities with elevated power requirements would be loading onto a trailer and higher travel speeds.

The power used during a phenotyping operation was to be analyzed. This would be compared to steady-state operation on level concrete to negate machine losses. The Nebraska Tractor Test Laboratory (NTTL) test track was used for recording data. The machine would be operated at 1500 rpm and swash plate magnitude of 2.5% both in the

field and on the track. This resulted in a machine speed of approximately one kph, the speed used while phenotyping the research plots. Data were automatically recorded using the Farmobile PUC at 1 Hz.

4.2.3 Flex-Ro Cost of Operation Estimation

The Flex-Ro machine operating costs were estimated based on recorded data and operating assumptions. The per area cost includes fuel and fixed capital recovery. The labor cost was left out of this analysis as only a small amount of time would be needed for delivery, set-up, and retrieval. The gasoline cost is set at $0.92/L ($3.50/gal). The capital recovery assumes a $100,000 machine, with a 6-year life, covering 4,000 ha per year. The user of the machine is proposed to be a corporate agronomy service. The productivity estimate assumes a working swath width of 18.3 m. This width was selected as a common swath for a variable rate applicator. The Flex-Ro has three sensor units mounted on the PhenoBar. These sensor values were averaged into one data point for the density calculation of this analysis. Further research must be completed to determine what the optimum swath width and density of the Flex-Ro phenotyping application would be.

4.3 Results and Discussions

The Farmobile PUC was installed on the Flex-Ro before the first field phenotyping operation. Machine data were streamed and recorded throughout the season. The research field was covered at 1 kph to maximize resolution for the small plots. There are two factors which affect machine travel speed. First, the engine speed directly drives the

hydraulic pumps and is variable from 1000 to 3400 rpm. The flow through the hydraulic drive motors is regulated by the swash plate angle in the piston pump. Either of these factors can be adjusted to change machine travel speed.

Fuel rate increased directly with engine speed. Thus, it was desirable to operate the machine at the lowest possible engine rpm for maximum fuel economy. The lower limit to the engine speed is determined by the pressure drop required to drive the wheel motors.

4.3.1 Phenotyping Power Requirements

The power used during the field plot trials was determined from the Farmobile recorded data. Researchers desired to compare the additional power losses in the field setting compared to a level concrete track. The Farmobile 1 Hz data was processed so only the steady state passes were considered while operating in the field and on the track.

The mean of engine percent load and percent torque were the same in the field and on the concrete track. The mean percent engine torque for the field and concrete operation were 28.4% and 28.8%, respectively. The mean fuel rates were also statistically the same between operations. The only way to distinguish between field and concrete operations was the histogram of travel speed. The distributions of recorded data are shown in Figure 4.2.

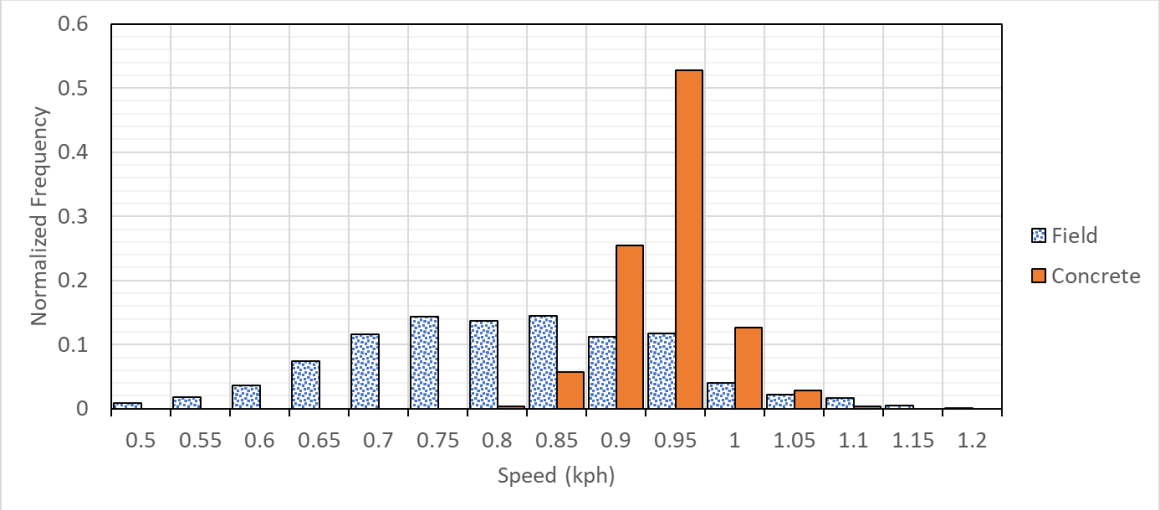


Figure 4.2: Recorded distribution of machine speed between Spidercam research field and concrete track.

The mean travel speeds in the field and on the concrete track were 0.79 kph (0.49 mph) and 0.92 kph (0.57 mph), respectively. A 16% slip percentage was calculated (Equation 4.1) for the infield operation. Wet soil conditions contributed to the excess slippage shown by this study.

Equation 4.1: Calculation of the slip percentage as a ratio of velocity on concrete (v_c) and soil (v_s).

$$slip\ \% = \left(\frac{v_c}{v_s} - 1\right) \cdot 100$$

4.3.2 Flex-Ro Cost of Operation Estimation

The economic analysis shown in Figure 4.3, (US Standard unit version in Appendix A.4) was completed to provide approximate operating costs of the Flex-Ro machine in a practical production setting. The reflectance sensors used in the Flex-Ro phenotyping system are passive, meaning they rely on an external light source (the sun during outdoor

activities). This may be a limitation for working hours during the growing season as the machine could only operate during conditions with optimal daylight. An active reflectance sensor would optimize the robot's ability to operate autonomously, potentially through the night improving the economic viability of this field scouting operation.

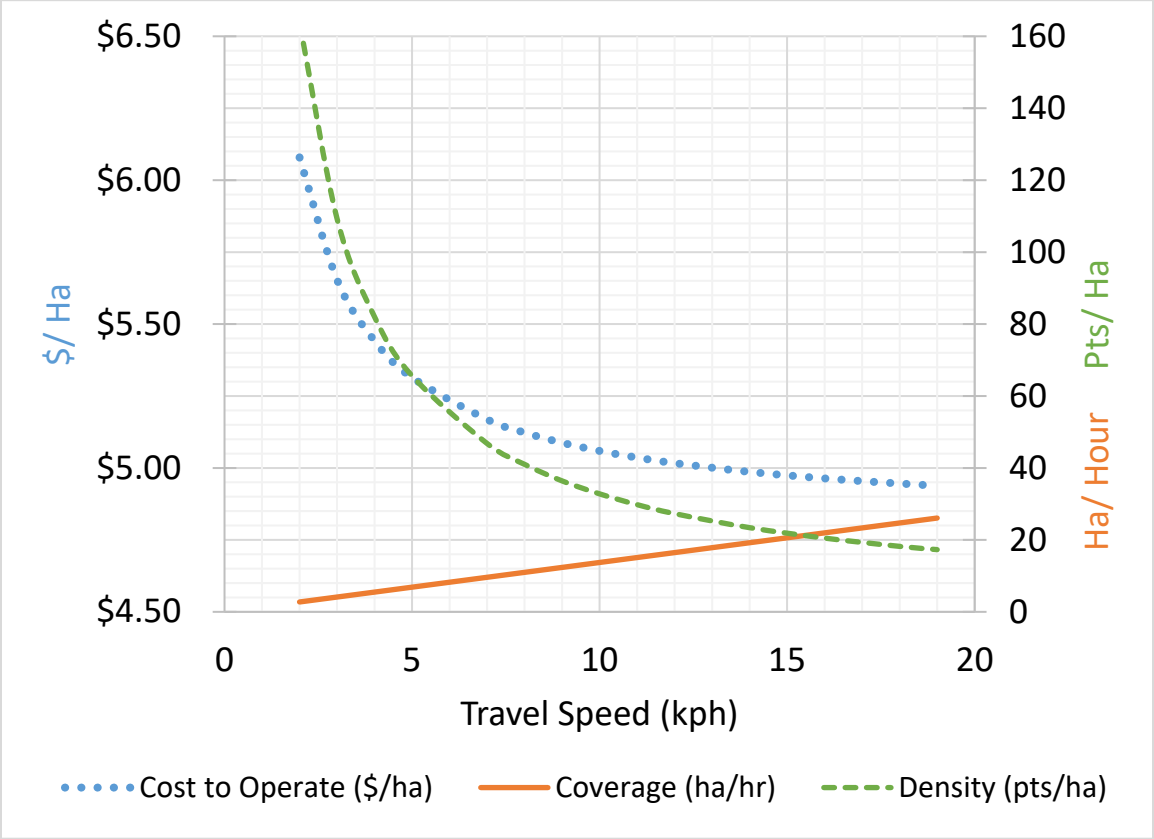


Figure 4.3: Economics of Phenotyping operation using the Flex-Ro machine with a swath width of 18.3 m and data collection cycle time of 8 sec.

4.4 Conclusions

Investigation of the power requirements of the Flex-Ro machine during phenotyping revealed surprising results. No statistically significant differences were attributed to operating in the field in comparison to a concrete track. Further research needs to be completed to determine the power requirements at higher speeds that would be used in production field settings.

Estimates of machine operating costs were calculated based on fixed parameters and productivity assumptions. The approximate cost to operate the Flex-Ro as a phenotyping field scout is $5.50/ha More research could be completed to quantify some of the assumptions made in these calculations. Field efficiency is highly variable and is dependent on field size and shape. This would be a factor to the end user in determining the machine productivity and scheduling.

Chapter 5 Conclusions and Future Work

Robotic applications in agriculture are currently under development, and it's only a matter of time until commercial autonomous solutions are widely available. The Flex-Ro machine was developed as an autonomous field robot which accommodates a variety of applications. The first application installed was for high-throughput field phenotyping or scouting.

Phenotyping at some level has been in practice since people began selectively saving seeds from the most productive plants for the next year's planting. This science has developed into an advanced field of crop breeding and genetics. Tools have been developed to quantify the physical characteristics of plants using sensor-based data. However, high-resolution field phenotyping solutions are still limited. The Flex-Ro platform's first application was to take an existing and proven suite of phenotyping sensors and collect data while autonomously covering the field.

While the physical Flex-Ro machine was nearly complete, significant work remained on the machine's control system. A robust safety stop system needed implementation, as well as high-level navigation controls. Further, the robot had no means to react to obstacles. The safety system implemented allowed operators to stop the machine six different ways and included automatic stops for loss of communication between controllers.

A navigation algorithm was implemented using a GPS and a MATLAB app which communicated on the machine's CAN bus. Field path maps are planned to include swath coverage and headland turns. Waypoint navigation allowed for the Flex-Ro machine to follow predefined paths for special applications. Teleoperation via remote control provided a means for manual maneuvering.

An infrared time-of-flight sensor provided by ifm was implemented to detect obstacles in the machine's primary direction of travel. The sensor was installed as a proof of concept to verify the machine would appropriately react to obstacles in a variety of field environments. Field tests in automatic navigation mode verified that the sensor successfully detected pedestrian sized objects and triggered the machine to stop at a safe distance from the obstacle.

The Flex-Ro phenotyping suite of sensors was based off a push-cart system presented by Bai et. al. (2016). Sensors were installed on the PhenoBar and data was recorded locally using a LabVIEW program. The Flex-Ro phenotyping system was operated in the Spidercam phenotyping system research field so data could be compared throughout the growing season. Correlations were skewed due to the different data collection methods and processing procedures between the Flex-Ro and Spidercam. The Flex-Ro phenotyping system was able to statistically differentiate between soybean genotypes with different emergence and maturity characteristics. The late season phenotyping data showed statistical differences between the fully irrigated versus deficit plots. The phenotyping data collected by the Flex-Ro indicated a strong correlation coefficient to plot yield at the start of the growing season.

Economics must be considered for the viable use of any technology. The operating cost of the Flex-Ro machine was estimated at $5.50/ha based on power used during basic operations and productivity assumptions.

5.1 Future Work

Several limitations remain with the Flex-Ro platform and application. The control system requires further development for intuitive human interface. Methods for remote monitoring and field path uploading have yet to be implemented. The use of a MATLAB app for long-term field use is impractical and should be transferred to a microprocessor designed for off-highway machinery. The GPS data should be augmented with an inertial measurement unit (IMU) and dynamic machine model for high accuracy navigation.

The Flex-Ro machine can move in any direction. The obstacle detection package must be able to monitor the machine's intended direction of motion.

Additional sensors may need to be installed before unsupervised autonomy can be achieved. Hydraulic pressure and temperature sensors as well as wheel speed sensors would be needed to monitor the hydraulic condition and ensure the machine is not stuck.

Future developments of the Flex-Ro platform must remain mindful of the developing standards pertaining to autonomous machines. Compliance with regulatory and safety standards will remain crucial to the success of the platform.

The Flex-Ro phenotyping system requires development of the LabVIEW program to ensure properly synchronized data points. Delay between recording of the GPS

coordinates and sensor data introduces error, as the machine is continuously moving while recording. Further, data collection currently requires a laptop with LabVIEW and an operator to initiate the program and data recording. A completely autonomous platform should be able to initiate recording and provide a means to monitor the phenotyping data in real time. Advanced onboard processing and internet connectivity would greatly enhance the usefulness of the machine.

The passive reflectance sensors limit the operating time of field scouting operations during daylight hours. Installing active sensors would fully use the autonomous capability of the machine. The Flex-Ro would be able to operate through the night with the use of active reflectance sensors to greatly expand the field scouting capability.

Costs for the operation of the Flex-Ro could be further refined with the continued use of the machine in large field settings. Practical limits for speed and data resolution could be optimized to the desired application. For example, at what level does the number of points per acre for phenotyping impact the value of the operation. Other applications for the machine may also be developed and evaluated.

The field of agricultural robotics is continuously evolving. Academia and start-up companies are developing promising prototypes, but practical commercial field application remains limited. The transition to autonomy appears inevitable but has yet to be economically proven. Diligent work must continue on the Flex-Ro to remain relevant as an autonomous field robot.

References

Accuracy equation explains how AutoTrac™ accuracy is derived [WWW Document], 2013. URL http://salesmanual.deere.com/sales/salesmanual/en_NA/ams/2010/feature/autotrac/accuracy_equation.html?sbu=ag&link=prodcat (accessed 10.30.19).

Andrade-Sanchez, P., Gore, M.A., Heun, J.T., Thorp, K.R., Carmo-Silva, A.E., French, A.N., Salvucci, M.E., White, J.W., 2014. Development and evaluation of a field-based high-throughput phenotyping platform. Funct. Plant Biol. 41, 68–79. https://doi.org/10.1071/FP13126

Araus, J.L., Cairns, J.E., 2014. Field high-throughput phenotyping: the new crop breeding frontier. Trends Plant Sci. 19, 52–61. https://doi.org/10.1016/j.tplants.2013.09.008

Åstrand, B., Baerveldt, A.-J., 2002. An Agricultural Mobile Robot with Vision-Based Perception for Mechanical Weed Control. Auton. Robots 13, 21–35. https://doi.org/10.1023/A:1015674004201

Atefi, A., 2019. In vivo Human-Like Robotic Phenotyping of Leaf and Stem Traits in Maize and Sorghum in Greenhouse (Ph.D.). The University of Nebraska - Lincoln, United States -- Nebraska.

Baek, W., Jang, S., Song, H., Kim, S., Song, B., Chwa, D., 2008. A CAN-based Distributed Control System for Autonomous All-Terrain Vehicle (ATV). IFAC Proc. Vol., 17th IFAC World Congress 41, 9505–9510. https://doi.org/10.3182/20080706-5-KR-1001.01607

Bai, G., Ge, Y., Hussain, W., Baenziger, P.S., Graef, G., 2016. A Multi-Sensor System for High Throughput Field Phenotyping in Soybean and Wheat Breeding. Comput. Electron. Agric. 128, 181–192. https://doi.org/10.1016/j.compag.2016.08.021

Bai, G., Ge, Y., Scoby, D., Leavitt, B., Stoerger, V., Kirchgessner, N., Irmak, S., Graef, G., Schnable, J., Awada, T., 2019. NU-Spidercam: A Large-Scale, Cable-Driven, Integrated Sensing and Robotic System for Advanced Phenotyping, Remote Sensing, and Agronomic Research. Comput. Electron. Agric. 160, 71–81. https://doi.org/10.1016/j.compag.2019.03.009

Baillie, C.P., Lobsey, C.R., Antille, D.L., McCarthy, C.L., Thomasson, J.A., 2018. A review of the state of the art in agricultural automation. Part III: Agricultural machinery navigation systems, in: ASABE Annual International Meeting Detroit. Presented at the ASABE Annual International Meeting, American Society of Agricultural and Biological Engineers. https://doi.org/10.13031/aim.201801591

Bak, T., Jakobsen, H., 2004. Agricultural Robotic Platform with Four Wheel Steering for Weed Detection. Biosyst. Eng. 87, 125–136. https://doi.org/10.1016/j.biosystemseng.2003.10.009

Bangert, W., Kielhorn, A., Rahe, F., GmbH, A.-W.H.D., Kg, C., Albert, A., Biber, P., Grzonka, S., Haug, S., Michaels, A., GmbH, R.B., Mentrup, D., Hänsel, M., GbR, H., Kinski, D., Möller, K., Ruckelshausen, A., Scholz, C., Sellmann, F., Strothmann, W., Trautz, D., 2015. Field-Robot-Based Agriculture: "RemoteFarming.1" and "BoniRob-Apps" 8.

Bell, J., 2002. Network Protocols Used in the Automotive Industry.

Bell, T., 1999. Precision Robotic Control of Agricultural Vehicles on Realistic Farm Trajectories.

Biber, P., Weiss, U., Dorna, M., Albert, A., n.d. Navigation System of the Autonomous Agricultural Robot "BoniRob" 7.

Boyali, A., Mita, S., John, V., 2018. A Tutorial On Autonomous Vehicle Steering Controller Design, Simulation and Implementation. ArXiv180303758 Cs.

Bruinsma, J., 2009. THE RESOURCE OUTLOOK TO 2050: 33.

Buick, R., 2006. GPS Guidance and Automated Steering Renew Interest In Precision Farming Techniques 10.

Cariou, C., Lenain, R., Thuilot, B., Berducat, M., 2009. Automatic Guidance of a Four-Wheel-Steering Mobile Robot for Accurate Field Operations. J. Field Robot. 26, 504–518. https://doi.org/10.1002/rob.20282

Case IH, 2018. Case IH Defines Categories of Autonomy and Announces Pilot Program.

Cassman, K.G., 1999. Ecological intensification of cereal production systems: Yield potential, soil quality, and precision agriculture. Proc. Natl. Acad. Sci. U. S. A. 96, 5952–5959.

Dhondt, S., Wuyts, N., Inzé, D., 2013. Cell to whole-plant phenotyping: the best is yet to come. Trends Plant Sci. 18, 428–439. https://doi.org/10.1016/j.tplants.2013.04.008

Emmi, L., Gonzalez-de-Soto, M., Pajares, G., Gonzalez-de-Santos, P., 2014. New Trends in Robotics for Agriculture: Integration and Assessment of a Real Fleet of Robots. Sci. World J. 2014, 1–21. https://doi.org/10.1155/2014/404059

Fahlgren, N., Gehan, M.A., Baxter, I., 2015. Lights, camera, action: high-throughput plant phenotyping is ready for a close-up. Curr. Opin. Plant Biol. 24, 93–99. https://doi.org/10.1016/j.pbi.2015.02.006

Fehr, W.R., 1991. Principles of Cultivar Development. W.R. Fehr, Ames, Iowa.

Foix, S., Alenyà, G., Torras, C., 2015. 3D Sensor planning framework for leaf probing, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Presented at the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6501–6506. https://doi.org/10.1109/IROS.2015.7354306

Furbank, R.T., Tester, M., 2011. Phenomics – technologies to relieve the phenotyping bottleneck. Trends Plant Sci. 16, 635–644. https://doi.org/10.1016/j.tplants.2011.09.005

Godoy, E.P., Tangerino, G.T., Tabile, R.A., Inamasu, R.Y., Porto, A.J.V., 2012. Networked Control System for the Guidance of a Four-Wheel Steering Agricultural Robotic Platform. J. Control Sci. Eng. 2012, 1–10. https://doi.org/10.1155/2012/368503

Griepentrog, H.W., Jaeger-Hansen, C.L., Dühring, K., 2012. Electric Agricultural Robot with Multi-Layer-Control 7.

J. Jin, L. Tang, 2010. Optimal Coverage Path Planning for Arable Farming on 2D Surfaces. Trans. ASABE 53, 283–295. https://doi.org/10.13031/2013.29488

Lakkad, S., 2004. Modeling and Simulation of Steering Systems for Autonomous Vehicles. The Florida State University.

Marx, S.E., 2015. Controller Area Network (CAN) Bus J1939 Data Acquisition Methods and Parameter Accuracy Assessment Using Nebraska Tractor Test Laboratory Data. University of Nebraska-Lincoln.

McAllister, W., Osipychev, D., Davis, A., Chowdhary, G., 2019. Agbots: Weeding a field with a team of autonomous robots. Comput. Electron. Agric. 163, 104827. https://doi.org/10.1016/j.compag.2019.05.036

Mjaavatten, A., 2018. Curvature of a 2D or 3D curve.

O'Connor, M.L., 1997. Carrier-Phase Differential GPS for Automatic Control of Land Vehicles.

Oftadeh, R., Aref, M.M., Ghabcheloo, R., Mattila, J., 2013. Mechatronic Design of a Four Wheel Steering Mobile Robot with Fault-Tolerant Odometry Feedback. IFAC Proc. Vol. 46, 663–669. https://doi.org/10.3182/20130410-3-CN-2034.00092

Paden, B., Čáp, M., Yong, S.Z., Yershov, D., Frazzoli, E., 2016. A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. IEEE Trans. Intell. Veh. 1, 33–55. https://doi.org/10.1109/TIV.2016.2578706

Pitla, S.K., Luck, J.D., Shearer, S.A., 2010a. Low-Cost Obstacle Detection Sensor Array for Unmanned Agricultural Vehicles, in: 2010 Pittsburgh, Pennsylvania, June 20 - June 23, 2010. Presented at the Annual International Meeting, American Society of Agricultural and Biological Engineers. https://doi.org/10.13031/2013.36203

Pitla, S.K., Luck, J.D., Shearer, S.A., 2010b. Multi-Robot System Control Architecture (MRSCA) for Agricultural Production, in: 2010 Pittsburgh, Pennsylvania, June 20 - June 23, 2010. Presented at the Annual International Meeting, American Society of Agricultural and Biological Engineers. https://doi.org/10.13031/2013.36212

Rohrer, R., 2017. Investigation of Petroleum Use in Off-road Agricultural Machinery and Analysis of J1939 Controller Area Network (CAN) Data for Advanced Machinery Testing. University of Nebraska-Lincoln.

Shafiekhani, A., Kadam, S., Fritschi, F., DeSouza, G., 2017. Vinobot and Vinoculer: Two Robotic Platforms for High-Throughput Field Phenotyping. Sensors 17, 214. https://doi.org/10.3390/s17010214

Shakoor, N., Lee, S., Mockler, T.C., 2017. High throughput phenotyping to accelerate crop breeding and monitoring of diseases in the field. Curr. Opin. Plant Biol., 38 Biotic interactions 2017 38, 184–192. https://doi.org/10.1016/j.pbi.2017.05.006

Slaughter, D.C., Giles, D.K., Downey, D., 2008. Autonomous robotic weed control systems: A review. Comput. Electron. Agric., Emerging Technologies For Real-time and Integrated Agriculture Decisions 61, 63–78. https://doi.org/10.1016/j.compag.2007.05.008

Stone, M.L., Benneweis, R.K., Bergeijk, J.V., 2008. Evolution of Electronics for Mobile Agricultural Equipment. Trans. ASABE 51, 6.

Thomasson, J.A., Baillie, C.P., Antille, D.L., McCarthy, C.L., Lobsey, C.R., 2018. A review of the state of the art in agricultural automation. Part II: On-farm agricultural communications and connectivity, in: ASABE Annual International Meeting Detroit. Presented at the ASABE Annual International Meeting, American Society of Agricultural and Biological Engineers. https://doi.org/10.13031/aim.201801590

Troyer, T., 2017. Event and Time-Triggered Control Module Layers for Individual Robot Control Architectures of Unmanned Agricultural Ground Vehicles. University of Nebraska-Lincoln.

Tu, X., 2013. Robust Navigation Control and Headland Turning Optimization of Agricultural Vehicles (Dissertation). Iowa State University, Ames, Iowa.

USDA, 2019. USDA Agricultural Projections to 2028. USDA.

Uzunsoy, E., 2018. Yol ve Yol-Dışı Yüzeylerde Taşıt Dinamiği Benzetimleri – Karşılaştırmalı Bir Analiz. SAÜ Fen Bilim. Enstitüsü Derg. 1–1. https://doi.org/10.16984/saufenbilder.333872

Werner, J.P., 2016. Flex-Ro: Design, Implementation, and Control of Subassemblies for an Agricultural Robotic Platform. University of Nebraska-Lincoln.

White, J.W., Andrade-Sanchez, P., Gore, M.A., Bronson, K.F., Coffelt, T.A., Conley, M.M., Feldmann, K.A., French, A.N., Heun, J.T., Hunsaker, D.J., Jenks, M.A., Kimball, B.A., Roth, R.L., Strand, R.J., Thorp, K.R., Wall, G.W., Wang, G., 2012. Field-based phenomics for plant genetics research. Field Crops Res. 133, 101–112. https://doi.org/10.1016/j.fcr.2012.04.003

White, J.W., Conley, M.M., 2013. A Flexible, Low-Cost Cart for Proximal Sensing. Crop Sci. 53, 1646–1649. https://doi.org/10.2135/cropsci2013.01.0054

Appendix A Supplemental Information

Appendix A.1: 2019 Spidercam and Flex-Ro Phenotyping Data Master List:

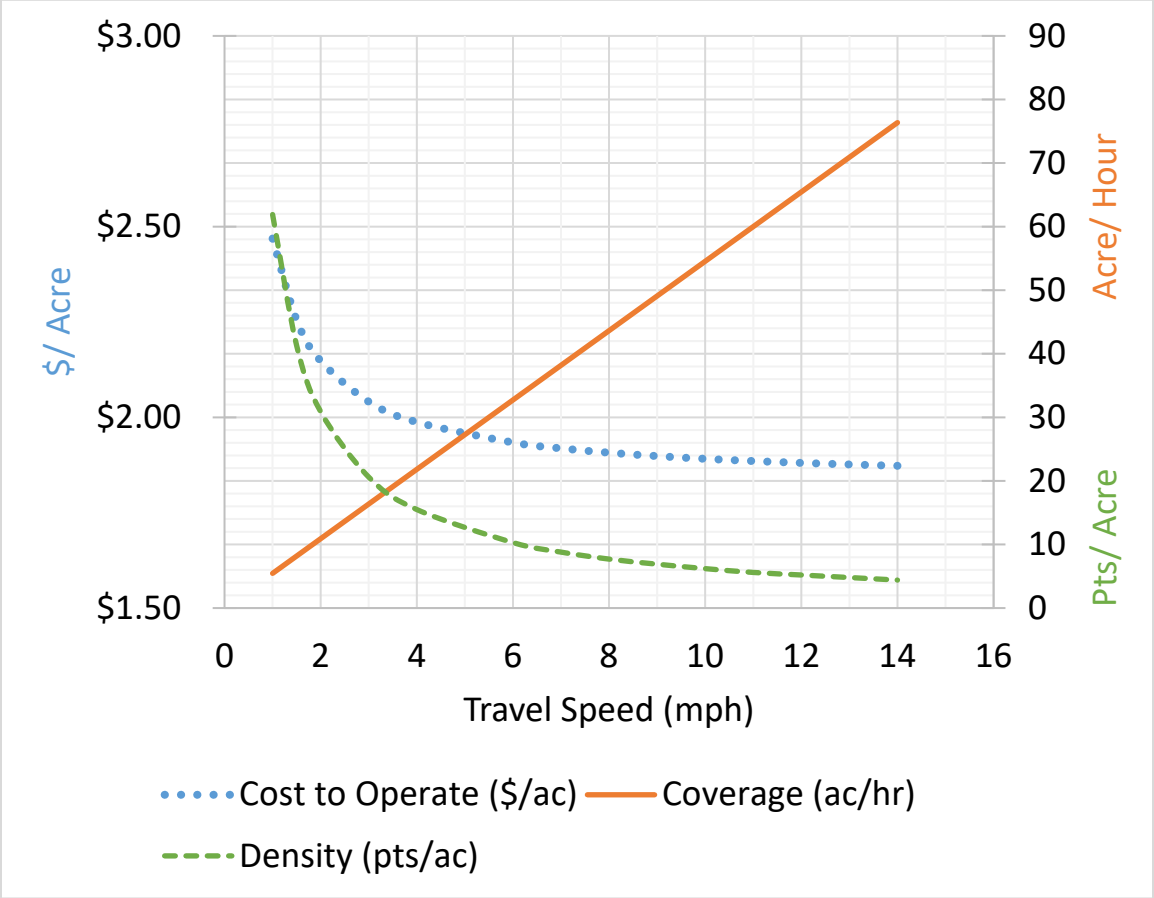
https://unl.box.com/s/1xlzq59q8vtcrzdgl2xs4ngvjmkb45qo

Appendix A.2: 2019 Flex-Ro Phenotyping Raw Phenotyping Data:

https://unl.box.com/s/dcrm00dx7pdpa4a3zhapbqjw90jk2wg3

Appendix A.3: Additional Media:

https://unl.box.com/s/iyjbtqk6kp8dqsk950hcgp66bu9cw5sd



Appendix A.4: Economics of Phenotyping operation using the Flex-Ro machine. Swath width set at 60 ft. Averages the three sensor units into one point for the points per acre calculation. Assumes new data point collected every 8 seconds.

Appendix B Flex-Ro Guides

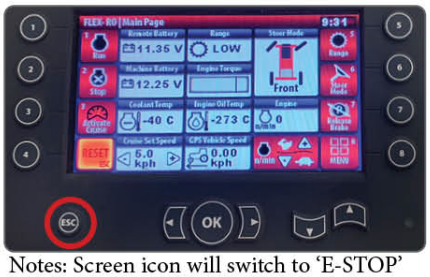
The Flex-Ro Operating Guides are presented as an introduction and will not cover every operating scenario.

FLEX-Ro Operator's Guide: Remote Control

1. Turn on machine & remote power.

Notes: Red LED on box will glow faintly.
- If not, check battery disconnect.
- Green power switch on remote.

2. Press 'ESC' to reset E-Stop system.



Notes: Screen icon will switch to 'E-STOP'
- Ensure machine E-Stop buttons are reset!
- Rotary beacon will illuminate.

3. Start engine

Notes:
1. Press 'Run' soft-key. Box LEDs Illuminate.
2. Wait a few seconds for fuel priming.
3. Press &hold 'Start' soft-key until started.

3. Enable hydro

Notes:
1. Press 'Release Brake' soft-key.
2. Use joystick Y-Axis.
3. Change speed range with 'Range' soft-key.

Recommended Operating Points

1. Engine RPM 1300 - 2600 RPM.
2. Hydro in 'MID' range.
3. Remote battery minimum of 10.3 V.

Obstacle Override

Note. Front obstacle detected will also stop the machine in reverse!
1. Press and hold 'OK' soft-key and operate remote until clear of obstacle.
2. Releasing 'OK' will immediately reactivate detection.
3. **It is the operator's responsibility to prevent collisions. Do NOT rely on obstacle detector.**

Cruise

Note. Cruise uses GPS based speed. Must be outside and LED on GPS glowing green.
1. Move machine in the desired direction of travel.
2. Press and release 'Activate Cruise' soft-key. Icon will change.
3. Release joystick to Neutral position. Move straight left and right to steer.
4. Any forward or back joystick movement will deactivate cruise.
Note: Start with cruise set-point within 5kph of current machine speed.

Updated: 03 JULY 2019 | J. Murman

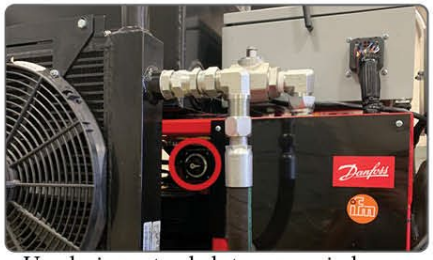
Appendix B.1: Flex-Ro Remote Operation Guide page 1.

FLEX-Ro Operator's Guide: Remote Control

E-STOP Won't Reset

- Check to be sure all buttons pulled out.
- Machine power is turned on.
- All ECU's connected.

Battery disconnect location



- Use during extended storage periods or during maintenance.

Remote Soft-Keys

Up Arrow - Increment engine speed
Down Arrow - Decrement engine speed
Right Arrow - Increment cruise set-point
Left Arrow - Decrement cruise set-point
OK - Hold for obstacle override
ESC - E-STOP Reset
1 : 8 - As described on screen
Power switch - Round green rocker on left end panel.

Shutdown Procedure

1. Return engine to idle for ~1min.
2. Press and release 'Engine Stop' soft-key.
3. Press and release 'E-STOP' soft-key.
4. Turn off red machine power switch.
5. Turn off remote.
6. Switch battery disconnect if leaving for longer period of time.

Updated: 03 JULY 2019 | J. Murman

Appendix B.2: Flex-Ro Remote Operation Guide page 2.

FLEX-Ro Operator's Guide: FLEX-RO RUN

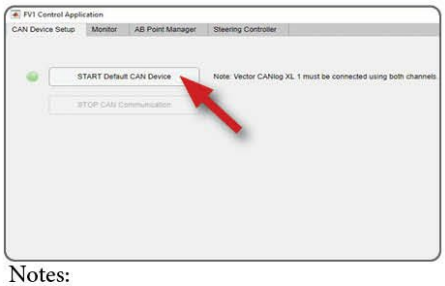
1. Turn on machine power.

Notes: Red LED on box will glow faintly.
- If not, check battery disconnect.

2. Connect laptop to machine.

Notes: - Connect USB cable
- Connect laptop charge cord.

3. Launch FlexRoRun Application

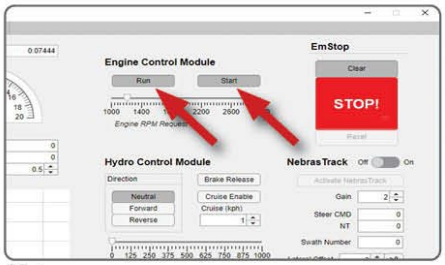


Notes:
1. Click 'START Default CAN Device'
2. Ensure green 'LED' illuminates.

4. Reset E-Stop System

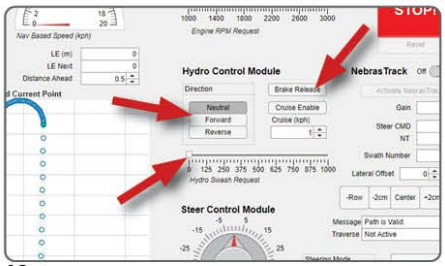
Notes:
1. Click 'Start Communication'
- Loop time should begin changing.
2. Click 'Reset' in the EmStop Module.

5. Start Engine



Notes:
1. Click 'Run'
2. Wait a few seconds for fuel priming.
3. Click 'Start'

6. Drive Machine



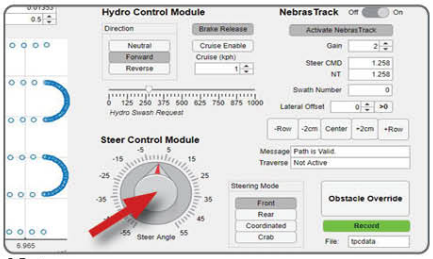
Notes:
1. Click 'Brake Release' or 'b' on keyboard.
2. Select direction and Hydro Swash request
- Or 'd' on keyboard, 'Up'/'Down' arrows on keyboard to change swash request.

Updated: 03 JULY 2019 | J. Murman

Appendix B.3: Flex-Ro FlexRoRun Operation Guide page 1.

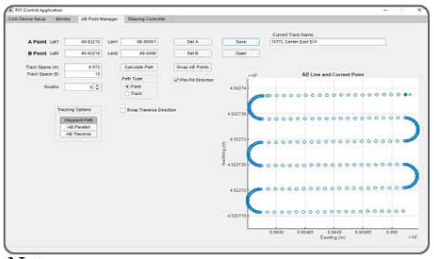
FLEX-Ro Operator's Guide: FLEX-RO RUN

7. Steer Machine



Notes:
1. Click desired angle on Steer Knob
2. Use 'Left' / 'Right' arrow keys to step steering angle.

8. NebrasTrack



Notes:
1. Click 'Set A' drive to next point.
2. Click 'Set B'
3. Input desired track parameters and click 'Calculate.' Save if desired.

Keyboard Shortcuts

Note: MUST CLICK ON BLANK SPACE OF WINDOW BEFORE USING SHORTCUTS!

Up Arrow	Increment hydro swash
Down Arrow	Decrement hydro swash
Right Arrow	Increment steer position
Left Arrow	Decrement steer position
Space	Neutral
Enter/Return	Activate NebraskTrack
Escape	E-Stop Button
b	Brake release toggle
c	Activate cruise toggle
d	Neutral > Fwd, Fwd <-> Rev
o	Obstacle override toggle
r	Engine run toggle
s	Engine start toggle
t	NebrasTrack enable toggle
1	Low Idle (1000 rpm)
2	Mid Idle (1500 rpm)
3	Mid-High Idle (2300 rpm)

Shutdown Procedure

1. Return engine to idle for ~1min.
2. Click 'Run'
3. Click 'Stop Communication'
4. Navigate to 'Can Device Setup' tab
5. Click 'STOP CAN Communication'
6. Close application.
7. Shut of machine red power switch.

Recommended Operating Points

1. Engine RPM Mid idle (1500 rpm).
2. Use keyboard shortcuts for maneuvering.

Obstacle Override

Note. Front obstacle detected will also stop the machine in reverse!
1. Toggle Override using button or keyboard shortcut.
2. Operator must manually reactivate obstacle detection!
3. It is the operator's responsibility to prevent collisions. Do NOT rely on obstacle detector.

Updated: 03 JULY 2019 | J. Murman

Appendix B.4: Flex-Ro FlexRoRun Operation Guide page 2.

Appendix C Wiring Tables

Appendix C.1: Flex-Ro main box connector pinout table.

	Pin Number	Description	Inside Box	Outside Box (if different)
DTM SIZE PINS	1	Brake Control Gnd	Black	
	2	Rf Gnd	Black	
	3	Lf Gnd	Black	
	4	Lr Gnd	Black	
	5	Rr Gnd	Black	
	6	GPS CAN Low	Green	
	7	GPS CAN High	Yellow	
	8	GPS Power	Red	
	9	GPS Ground	Black	
	10	SteerBus Power	Red	
	11	SteerBus Gnd	Black	
	12			
DT PINS	13	Hydro Brake Control	Lt. Blue	
	14	Hydro Right Front - c2p9	Purple	
	15	Hydro Left Front - c2p10	Orange	
DTP	16			
	17			
DT PINS	18	Hydro Left Rear - c2p11	Brown	
	19	Hydro Right Rear - c2p12	Blue	
	20			
DTP PINS	21			
	22			
	23			
	24			
	25	Toolbar Box Ground	Black	
	26	Toolbar Box Power	Red	
	27			
	28			
DT PINS	29	Steering Bus 12V +	Red	
	30	Steering Bus Gnd	Black	
	31			
DTP	32	MagHex Gnd	Black	
	33	MagHex 30A +	Red	
DT PINS	34	MagHex 2A +	Red	
	35	Steering Contactor Gnd	Black	
	36	Steering Contactor +	Blue	
DTM SIZE PINS	37	Steer CAN Hi	Yellow	
	38	Steer CAN Low	Green	
	39	Toolbar Box CAN Hi	Yellow	
	40	Toolbar Box CAN Low	Green	
	41			
	42			
	43	Start	Blue	Red
	44	Ignition	Purple	Brown
	45	Charge Indicator	Orange	White
	46	MIL	White	Blue
	47	Engine CAN Hi	Yellow	
	48	Engine CAN Low	Green	

	Max
DTM	7.5 A
DT	13 A
DTP	25 A

Appendix C.2: Flex-Ro fuse panel description table.

Description	Fuse Size (A)	Fuse Number		Description	Fuse Size (A)
MagHex Large	30	1	2	External Bus (Steering ECUs)	15
MagHex Small	5	3	4	Toolbar Box	25
ECU 0x2E (Hydro)	10	5	6	Machine GPS	5
ECU 0xD0 (Engine)	10	7	8	Open	-
ECU 0x1E (Power)	10	9	10	CAN Bridge WIC 2402	15
Indicator Lights	7.5	11	12	J1939 Diag. Port	10

Appendix C.3: Flex-Ro PhenoBox pinouts table.

	Pin No.	Description	Inside Box	LabJack Port	Outside Box (if different)
LEFT CONNECTOR	1	ground-ultra sonic	black		
	2	12V -ultra sonic	red		
	3	voltage input - ultrasonic	white	9	
	4	Differential High-IRT	red	0	
	5	Differential High-IRT	black	1	
	6	SingleEnd - IRT	green	2	
	7	analog ground 1-IRT	clear	GND	
	8	analog ground 2-IRT	blue	GND	
	9	excitation 2.5V - Irt	white	2.5V EXCIT	
	10				
	11				
	12				
	13				
	14				
	15				
	16				
CENTER CONNECTOR	1	ground-ultra sonic	black		
	2	12V -ultra sonic	red		
	3	voltage input - ultrasonic	white	10	
	4	Differential High-IRT	red	4	
	5	Differential High-IRT	black	5	
	6	SingleEnd - IRT	green	3	
	7	analog ground 1-IRT	clear	GND	
	8	analog ground 2-IRT	blue	GND	
	9	excitation 2.5V - Irt	white	2.5V EXCIT	
	10				
	11				
	12				
	13				
	14				
	15				
	16				
RIGHT CONNECTOR	1	ground-ultra sonic	black		
	2	12V -ultra sonic	red		
	3	voltage input - ultrasonic	white	11	
	4	Differential High-IRT	red	6	
	5	Differential High-IRT	black	7	
	6	SingleEnd - IRT	green	8	
	7	analog ground 1-IRT	clear	GND	
	8	analog ground 2-IRT	blue	GND	
	9	excitation 2.5V - Irt	white	2.5V EXCIT	
	10				
	11				
	12				
	13				
	14				
	15				
	16				
UP CONNECTOR	1	GPS Gnd	Black		
	2	GPS Power	Red		
	3	GPS CAN Hi	Yellow		
	4	GPS CAN Low	Green		
	5	CS215 Power 12V	Red		
	6	CS215 Signal	Green	FI07	
	7	CS215 Gnd	White		
	8	CS215 Gnd	Black		
	9	CS215 Gnd	Clear		

	Port No.	Decription
USB HUB	1	Labjack U6
	2	GPS Serial
	3	Left Spectrometer
	4	Center Spectrometer
	5	Right Spectrometer
	6	Upward Spectrometer
	7	Left Camera
	8	Center Camer
	9	Right Camera
	10	

FLEX_RO CAN Message Contents

ECU	SA
Engine	0x00
Engine Control	0xD0
Hydro	0x2E
Power	0x1E
Remote	0x27
Joystick	0x33
Steering LF	0x80
Steering LR	0x81
Steering RF	0x82
Steering RR	0x83

Priority	Hex
1	0x00
2	0x04
3	0x08
4	0x10
5	0x14
6	0x18
7	0x1C

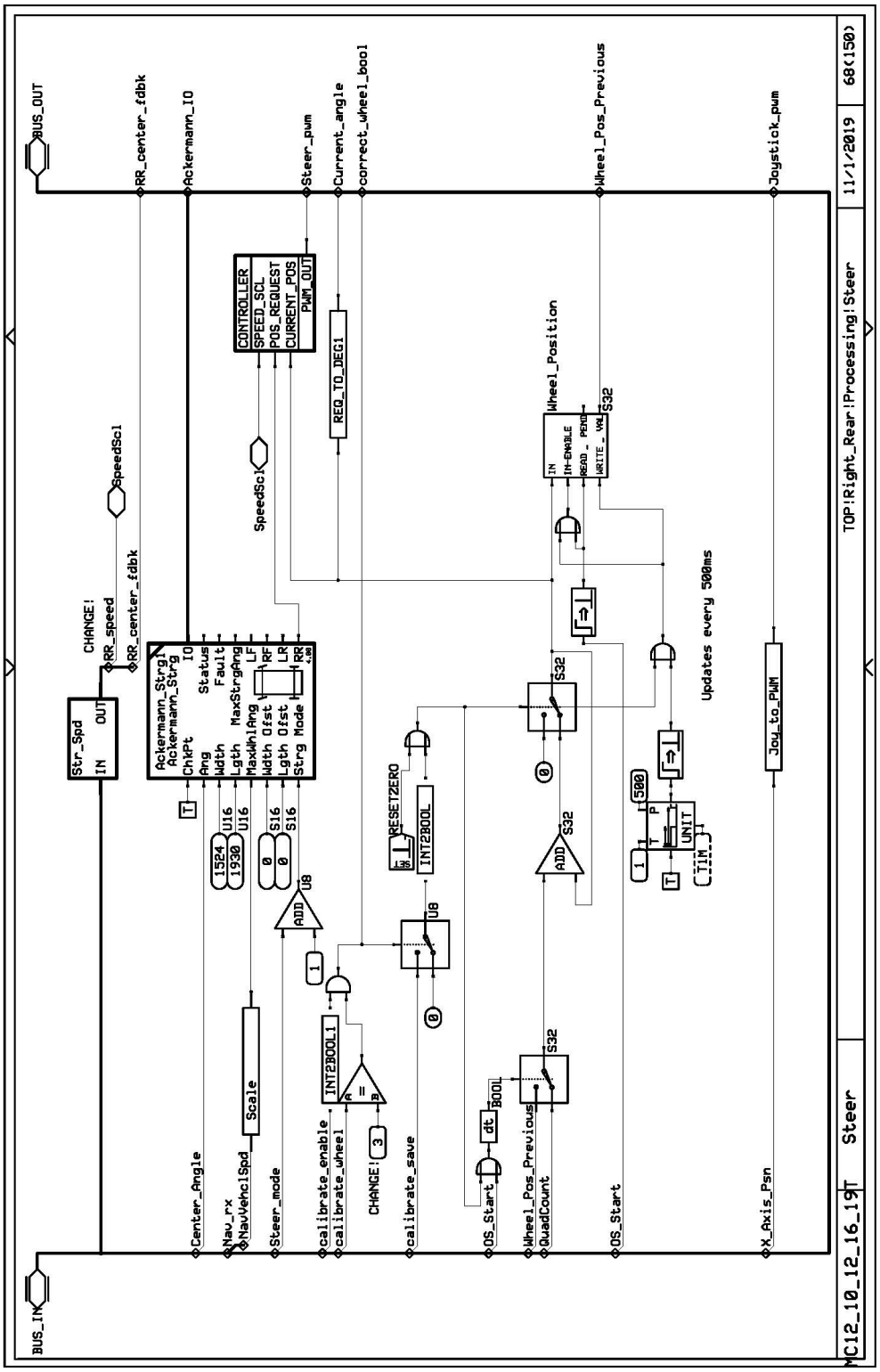
Source	Message ID	Description	Start Byte.bit	Length	Offset	Scale	Range	Unit
Power	18FF461E	Heartbeat/Voltage						
		Heatbeat	0.0	8	0	1	0-255	Count
		Machine Voltage	1.0	16	0	1	0-65535	mV
Remote	04FF5127	Hydro Control Message A						
		Direction	0.0	2	0	1	0-3	Enumerated B
		Brake Release	0.2	2	0	1	0-3	Enumerated A
		Cruise Enable	0.4	2	0	1	0-3	Enumerated A
		Magnitude	1.0	16	0	1	0-10000	%
		Cruise Speed Request	3.0	8	0	0.1	0-25.5	kph
Remote	00FF5127	Emergency Stop						
		Heatbeat	0.0	8	0	1	0-255	Count
	00FF56D0	Emergency Stop	1.0	8	0	1	0-3	Enumerated C
Remote	08FF4327	Steering Angle Control			27			
		Centerline Angle	0.0	16	-9000	1	0-18000	Deg*100
		Mode	2.0	2	0	1	0-3	Enumerated D
Remote	1CFF4527	Steering Calibrate			27			
		Calibration Mode	0.0	2	0	1	0-3	Enumerated A
		Calibration Wheel	0.2	2	0	1	0-3	Enumerated D
		Calibration Save	0.4	2	0	1	0-3	Enumerated A
Remote	10FF4427	Engine Control			27			
		Run	0.0	2	0	1	0-3	Enumerated A
		Start	0.2	r	0	1	0-3	Enumerated A
		Engine RPM Request	1.0	16	0	1	1000-3400	RPM
Engine Control	00FF56D0	Emergency Stop						
		Heatbeat	0.0	8	0	1	0-255	Count
		Emergency Stop	1.0	8	0	1	0-3	Enumerated C

Description	Transmitted	Value
Enumerated A	Disabled	0
	Enabled	1
	Error	2
	Not Available	3
Enumerated B	Reverse	0
	Neutral	1
	Forward	2
	Error	3
Enumerated C	Not Active	0
	Estop Active	1
	Reset	2
	Error	3
Enumerated D	Left Front	0
	Right Front	1
	Left Rear	2
	Right Rear	3
Enumerated D	Front	0
	Rear	1
	Coordinated	2
	Crab	3

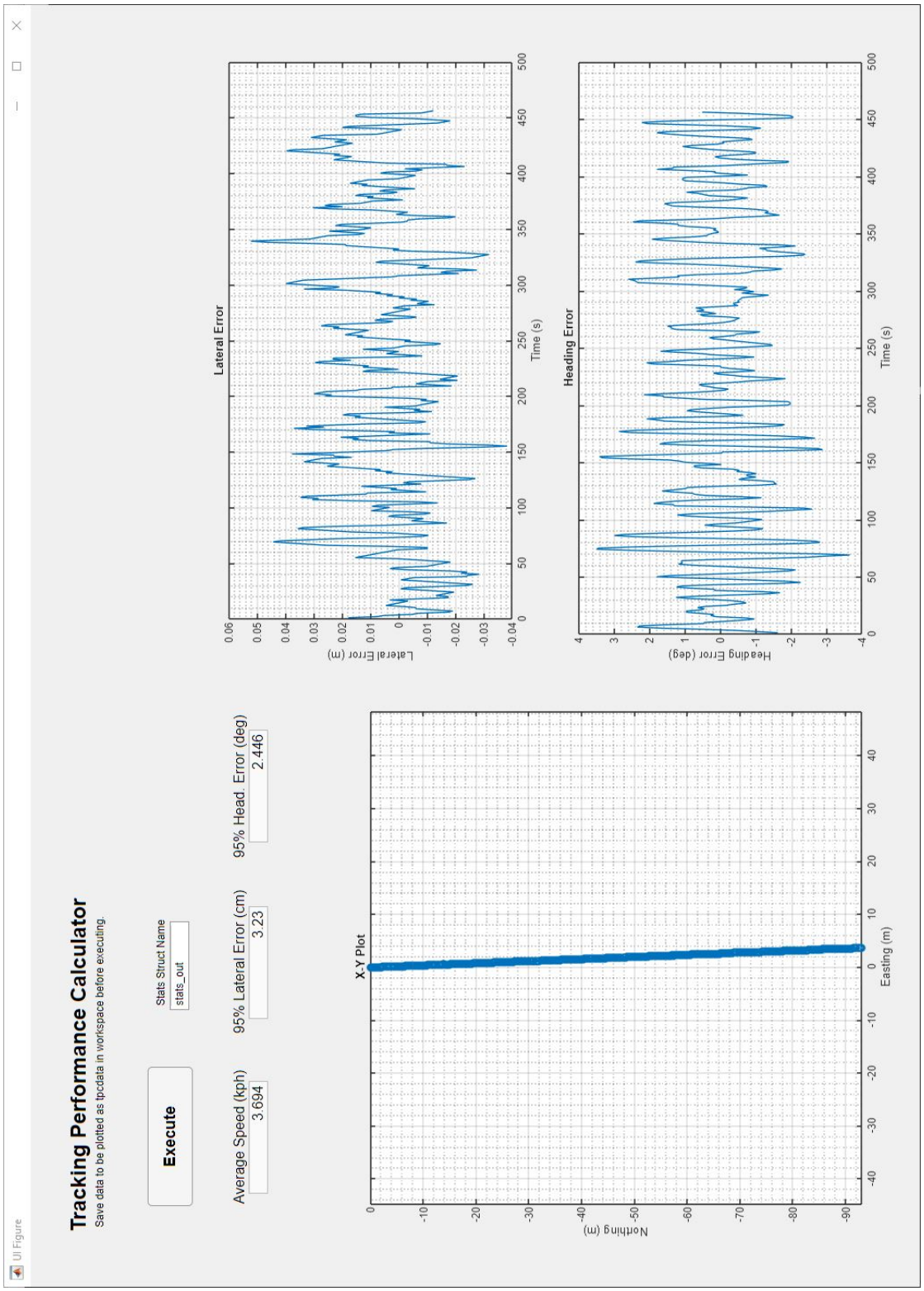
Appendix D Selected Code and Screen Captures

Appendix D.1: Lateral error calculation code block.

```
function latout= laterror(app,trdata,xc,yc)
    %Calculate lateral error from three points. A,B and current.
    %Perpendicular offset from line.
    %Extract track A and B points for current track.
    v1 = trdata.apt;
    v2 = trdata.bpt;

    pt = [xc,yc,0];
    a = v1 - v2;
    b = pt - v2;
    latout = norm(cross(a,b)) / norm(a);
    sgn = sign((v2(1)-v1(1))*(pt(2)-v1(2))-(v2(2)-v1(2))*(pt(1)-v1(1)))
    latout = latout*sgn;

    if isnan(latout)
       latout = 0;
    end
```



Appendix D.2: PLUS+1 GUIDE page for the Flex-Ro steering ECUs.



Appendix D.3: Tracking performance calculator screen-capture.

Appendix D.4: Strip excess points function for converted recorded datapoints to waypoint matrix code block.

```
function track = stripExcessPts(trackin)
% Strips excess points from recorded GPS logs for use as waypoint
% navigation.

% University of Nebraska-Lincoln
% Josh Murman 2019

%% Inputs
% trackin = [easting,northing,0] matrix. Trailing zero is optional.

%% Outputs
% track = [easting,northing,0] matrix. Zero only included if included in
% input.

%% Initialize Variables

h = 1;
j = 1;
k = 2;
pt_spacing = 1; %meters

%% Main
for i=1:length(trackin)

    %Reset magnitude
    mag = 0;

    %Start with first point, iterate through next points until distance is
    %greater than the desired point spacing.
    while mag < pt_spacing

        %Stop execution if at the end of the recorded matrix.
        if k == length(trackin) || h == length(trackin)
            break
        end
        %Calculate magnitude of distance between the two points.
        mag = sqrt((trackin(k,1)-trackin(h,1))^2 + ...
            (trackin(k,2)-trackin(h,2))^2);

        k = k+1
    end %while loo
    if k == length(trackin)
        break
    end

    %Assign track point, update counters.
    track(j,:) = trackin(h,:);
    j = j+1;
    h = k;
    k = 1+k;

end %for loop
end %function
```

Appendix D.5: The script used for importing raw Flex-Ro collected phenotyping data.

```
% Button pushed function: OpenLogButton
        function OpenLogButtonPushed(app, event)
 [app.basename, app.folder] = uigetfile('*.txt','Pick Log Text File');

            app.FilePathEditField.Value = app.folder;

            app.listing = dir(app.folder);

            processedfname = [app.folder,'processed.mat'];

            selection = 'Reprocess';

            if isfile(processedfname)

                f = app.PhenoCalcUIFigure;

                message = sprintf('Found preprocessed data!');

                selection =
uiconfirm(f,message,'Success!','Icon','success','Options',{'Continue','Reproce
ss','NDVI'});

            end

            if ~strcmp(selection,'Reprocess')

                load(processedfname,'wd');

                app.wd = wd;

            elseif ~strcmp(selection,'NDVI')


                %Define file names
                logfname = [app.folder,'log.txt'];
                specleftfname = [app.folder,'Spec-Left.csv'];
                specmidfname = [app.folder,'Spec-Middle.csv'];
                specrightfname = [app.folder,'Spec-Right.csv'];
                specupfname = [app.folder,'Spec-UP.csv'];
                waveleftfname = [app.folder,'Wave-Left.csv'];
                wavemidfname = [app.folder,'Wave-Middle.csv'];
                waverightfname = [app.folder,'Wave-Right.csv'];
                waveupfname = [app.folder,'Wave-UP.csv'];

                csvnames =
{specleftfname,specmidfname,specrightfname,specupfname,waveleftfname,wavemidfn
ame,waverightfname,waveupfname};
```

```
                rdnames =
{'specleft','specmid','specright','specup','waveleft','wavemid','waveright','w
aveup'};

                f = app.PhenoCalcUIFigure;

                d = uiprogressdlg(f,'Title','Please
wait...','Message','Importing your data...');

                app.rd.log = csvread(logfname ,0,2);

                imindex = unique(app.rd.log(:,1));
                r = length(imindex);

                for i = 1:length(csvnames)

                    app.rd.(rdnames{i}) = csvread(csvnames{i});

                    d.Value = i/length(csvnames);
                    d.Message = 'Reading files...';

                end



                d.Value = 0;
                d.Message = 'Calculating NDVI...';


                specnames = {'specleft','specmid','specright'};
                wavenames = {'waveleft','wavemid','waveright'};

                for i = 1:length(specnames)

                    for j = 1:r

                        app.rd.ndvi.(specnames{i})(j,1) =
calcNDVI(app.rd.(wavenames{i}),app.rd.(specnames{i})(j,:),app.rd.waveup,app.rd
.specup(j,:));

                    end

                    d.Value = i/length(specnames);

                end

                d.Value = 0;
                d.Message = 'Processing images...';


                camnames = {'CamL-','CamM-','CamR-'};
                camsave = {'CamL','CamM','CamR'};
                imtype = '.png';
```

```
                mkdir(app.folder,'Processed_Images')

                for i = 1:length(camnames)

                    for j = 1:r

                        imnum = imindex(j);

                        currim = [camnames{i},num2str(imnum),imtype];
                        impath = [app.folder, currim];

                        I = imread(impath);

                        [gpf,BW] = GPFcalc(I);

                        saveim = [app.folder,'Processed_Images\',currim];

                        imwrite(BW,saveim);

                        clear I

                        app.rd.gpf.(camsave{i})(j,1) = gpf;

                        d.Value = ((i-1)*r+j)/(length(camnames)*r);
                    end

                end

                datanames = {'Left','Mid','Right'};

                p = 1;

                for i = 1:r

                    for k = 1:3

                        app.rd.LJdata.(datanames{k})(app.rd.log(p),:) =
app.rd.log(p,4:8);

                        p = p+1;
                    end


                    d.Value = i/r;
                    d.Message = 'Organizing data...';

                end

                app.wd.headings = {'UTM Easing','UTM Northing','Thermistor
C','Target C','Height','NDVI','GPF'};

                for i = 1:3
```

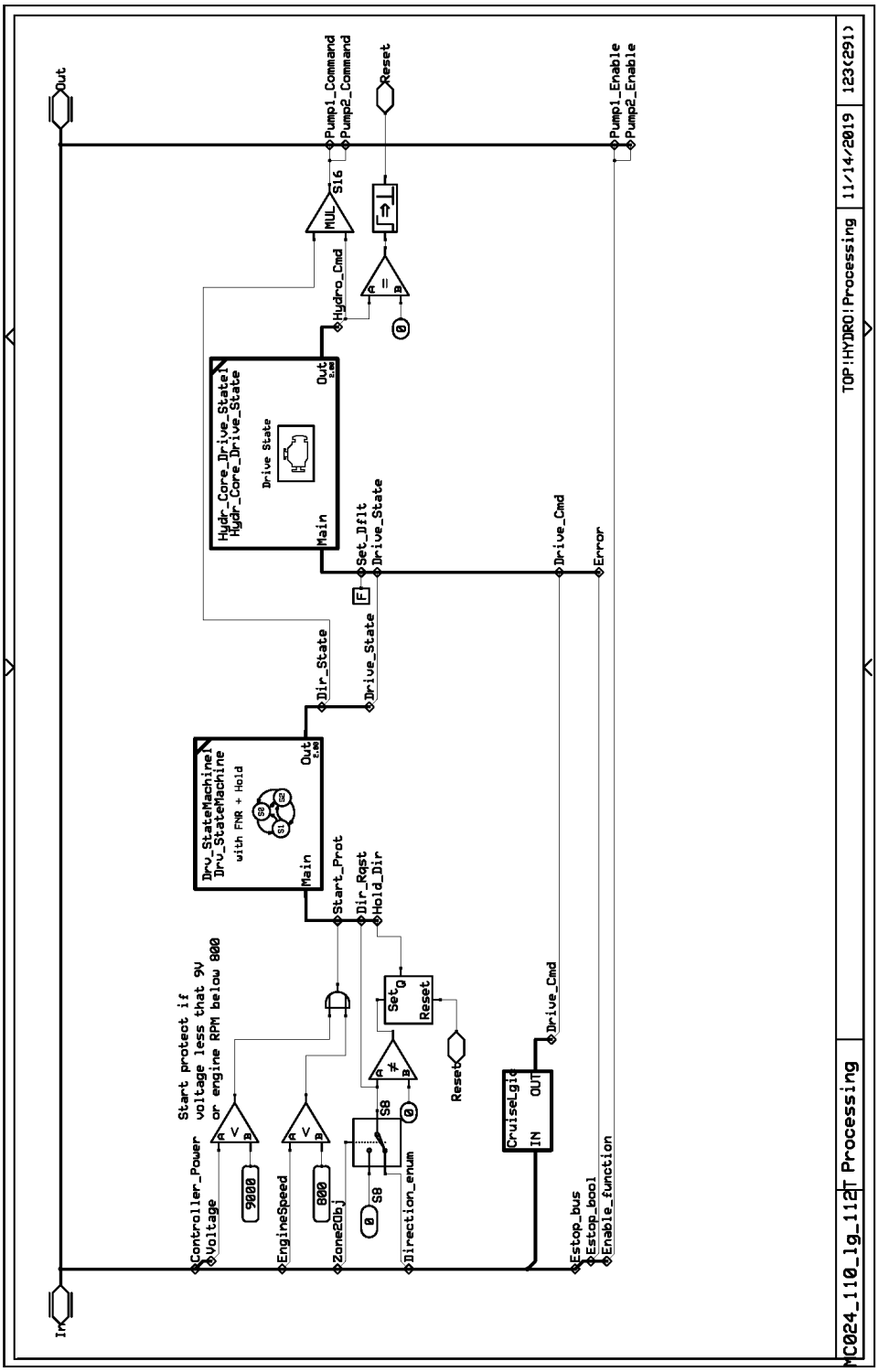
```
                    for j = 1:r

                        LJdata = app.rd.LJdata.(datanames{i})(j,:);
                        ndvi = app.rd.ndvi.(specnames{i})(j,1);
                        gpf = app.rd.gpf.(camsave{i})(j,1);

                        app.wd.master((i-1)*r+j,:) = [LJdata, ndvi, gpf];

                    end

                    d.Value = i/3;
                    d.Message = 'Structuring data...';

                end


                savefname = [app.folder,'processed.mat'];

                wd = app.wd;

                save(savefname,'wd')

                close(d)

            end %End of preprocessed if statment

            if strcmp(selection,'NDVI')

                %Define file names
                logfname = [app.folder,'log.txt'];
                specleftfname = [app.folder,'Spec-Left.csv'];
                specmidfname = [app.folder,'Spec-Middle.csv'];
                specrightfname = [app.folder,'Spec-Right.csv'];
                specupfname = [app.folder,'Spec-UP.csv'];
                waveleftfname = [app.folder,'Wave-Left.csv'];
                wavemidfname = [app.folder,'Wave-Middle.csv'];
                waverightfname = [app.folder,'Wave-Right.csv'];
                waveupfname = [app.folder,'Wave-UP.csv'];

                csvnames =
{specleftfname,specmidfname,specrightfname,specupfname,waveleftfname,wavemidfn
ame,waverightfname,waveupfname};
                rdnames =
{'specleft','specmid','specright','specup','waveleft','wavemid','waveright','w
aveup'};

                f = app.PhenoCalcUIFigure;

                d = uiprogressdlg(f,'Title','Please
wait...','Message','Importing your data...');

                app.rd.log = csvread(logfname ,0,2);
```

```
                imindex = unique(app.rd.log(:,1));
                r = length(imindex);

                for i = 1:length(csvnames)

                    app.rd.(rdnames{i}) = csvread(csvnames{i});

                    d.Value = i/length(csvnames);
                    d.Message = 'Reading files...';

                end



                d.Value = 0;
                d.Message = 'Calculating NDVI...';


                specnames = {'specleft','specmid','specright'};
                wavenames = {'waveleft','wavemid','waveright'};

                for i = 1:length(specnames)

                    for j = 1:r

                        app.rd.ndvi.(specnames{i})(j,1) =
calcNDVI(app.rd.(wavenames{i}),app.rd.(specnames{i})(j,:),app.rd.waveup,app.rd
.specup(j,:));

                    end

                    d.Value = i/length(specnames);


                end

                for i = 1:3

                    for j = 1:r


                        ndvi = app.rd.ndvi.(specnames{i})(j,1);


                        app.wd.master((i-1)*r+j,6) = ndvi;

                    end

                    d.Value = i/3;
                    d.Message = 'Structuring data...';
```

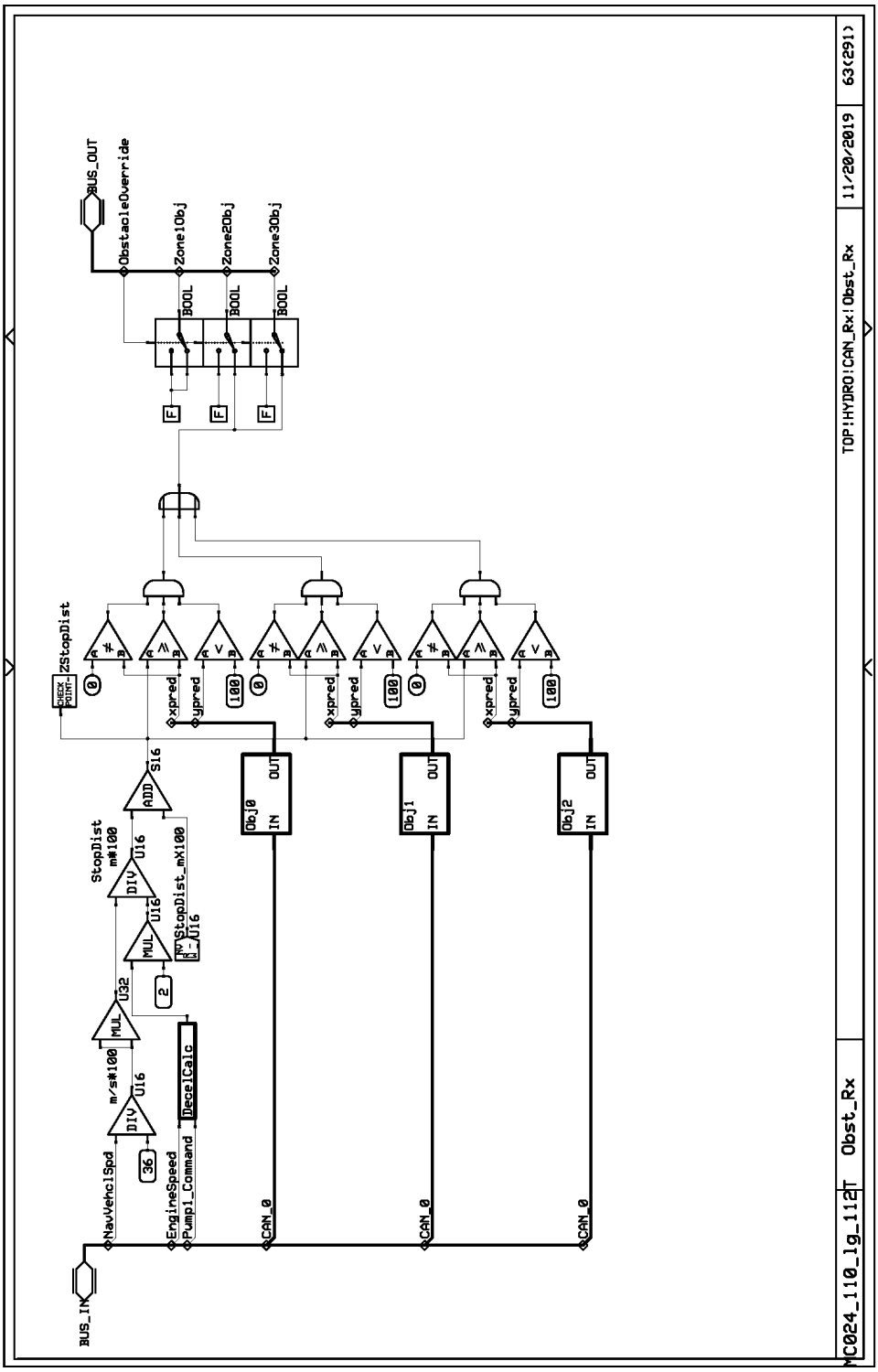
```
    end

    savefname = [app.folder,'processed.mat'];

    wd = app.wd;

    save(savefname,'wd')

    close(d)

end

savefname = [app.folder,'processed.mat'];

save(savefname,'wd')

scatter(app.ScatterAxes,app.wd.master(:,1),app.wd.master(:,2))

assignin('base','wd',app.wd);

app.CalculateStatistics.Enable = 'On';
app.MapItButton.Enable = 'On';
app.PlotData.Enable = 'On';
```

Appendix D.6: Script for calculating the green pixel fraction (GPF).

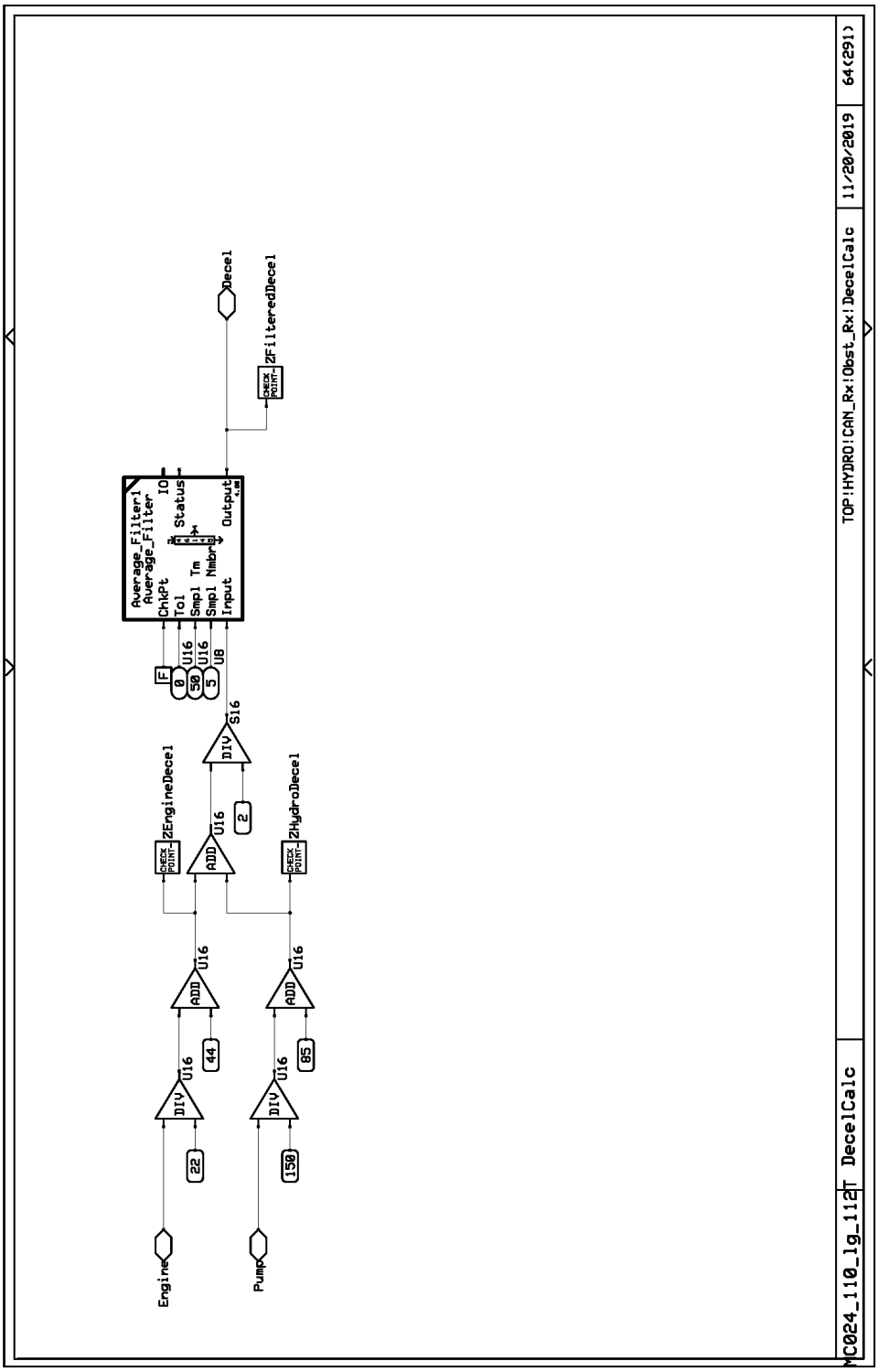
```
function [gpf,maskedRGB] = GPFcalc(rgbim)
%% Calculate green pixel fraction using the L*a*b color model and
%thresholding using the 'a' channel.


%Convert RGB image to L*a*b
lab = rgb2lab(rgbim);
%
% %Pull out the 'a' channel
lab_a = lab(:,:,2);
%
% %Acquire histogram for thresholding
[counts,~] = imhist(lab_a,96);
%
% %Apply the Otsu method of thresholding
T = otsuthresh(counts);

Imin = min(min(lab_a));
Imax = max(max(lab_a));

Irange = Imax-Imin;

T = Imin+Irange*T;

%% Calculate GPF from processed binary image.

[BW,maskedRGB] = greenMask(rgbim,Imin,T);

numPixels = numel(BW);

numGreen = sum(BW(:));

gpf = numGreen/numPixels;


end
```



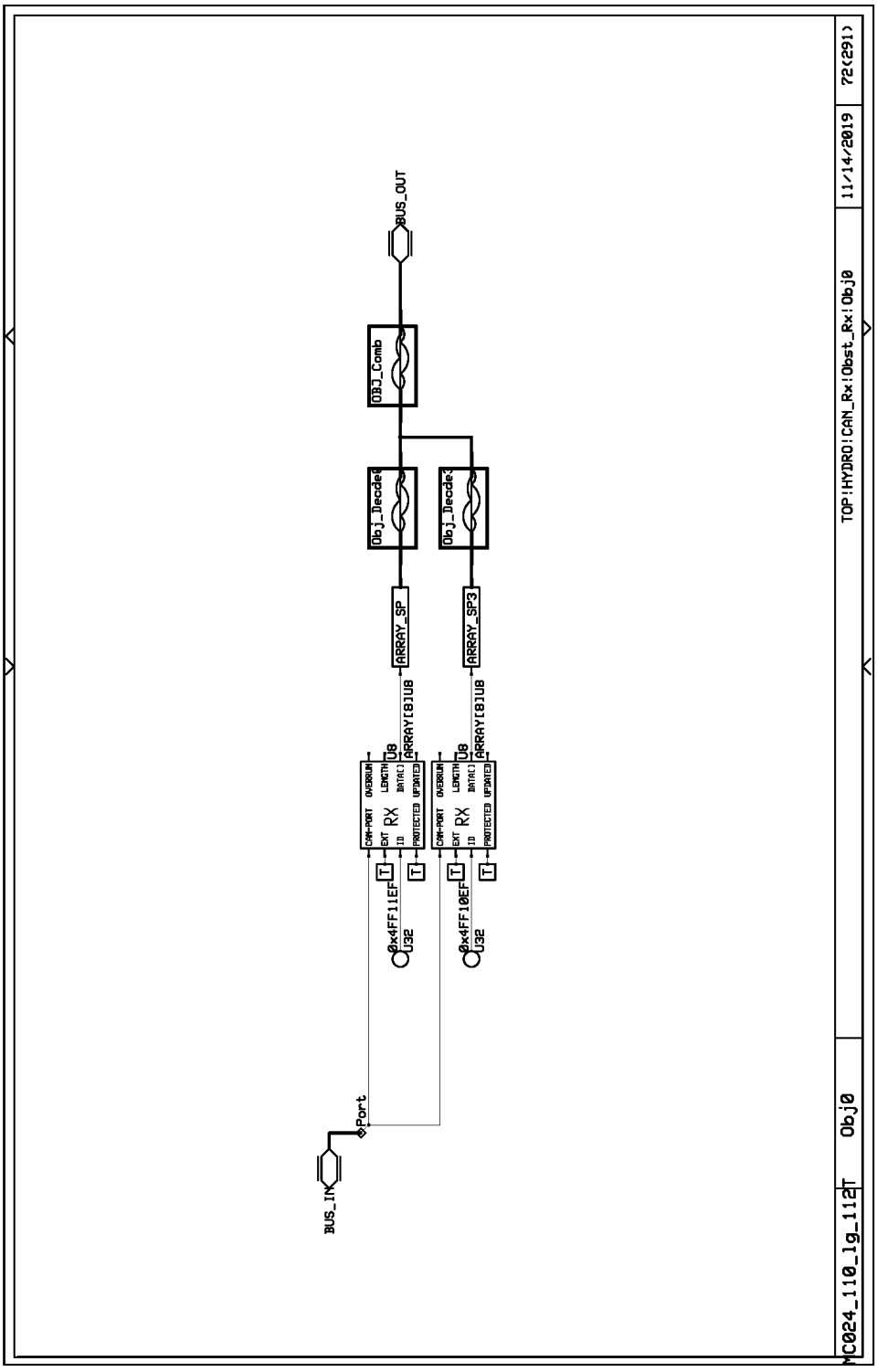
Appendix D.7: Flex-Ro Hydrostatic control main PLUS+1 page.



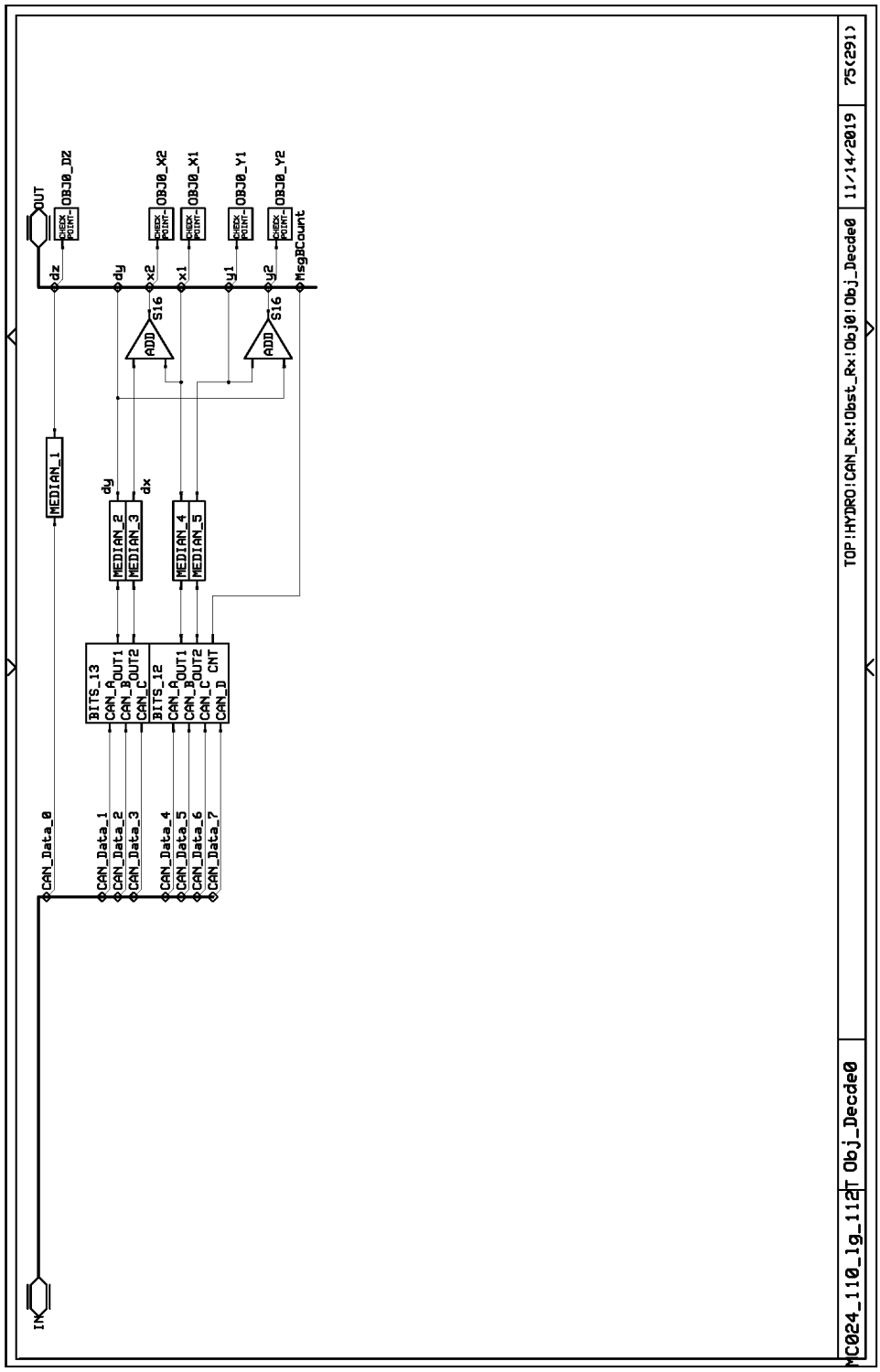
Appendix D.8: Flex-Ro PLUS+1 GUIDE Page Obstacle Reaction Algorithm



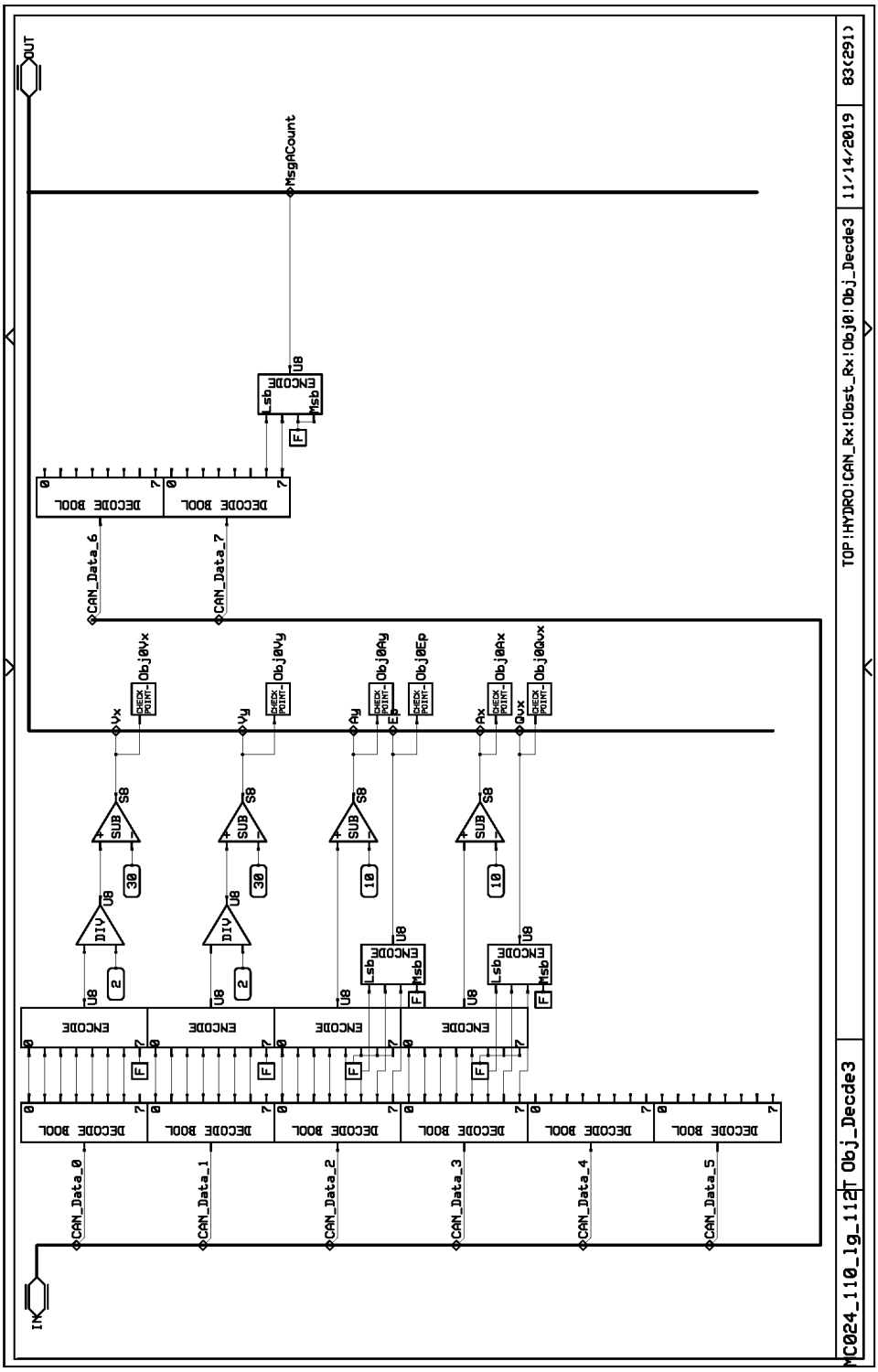
Appendix D.9: Flex-Ro PLUS+1 GUIDE Page implementing the variable deceleration factors.



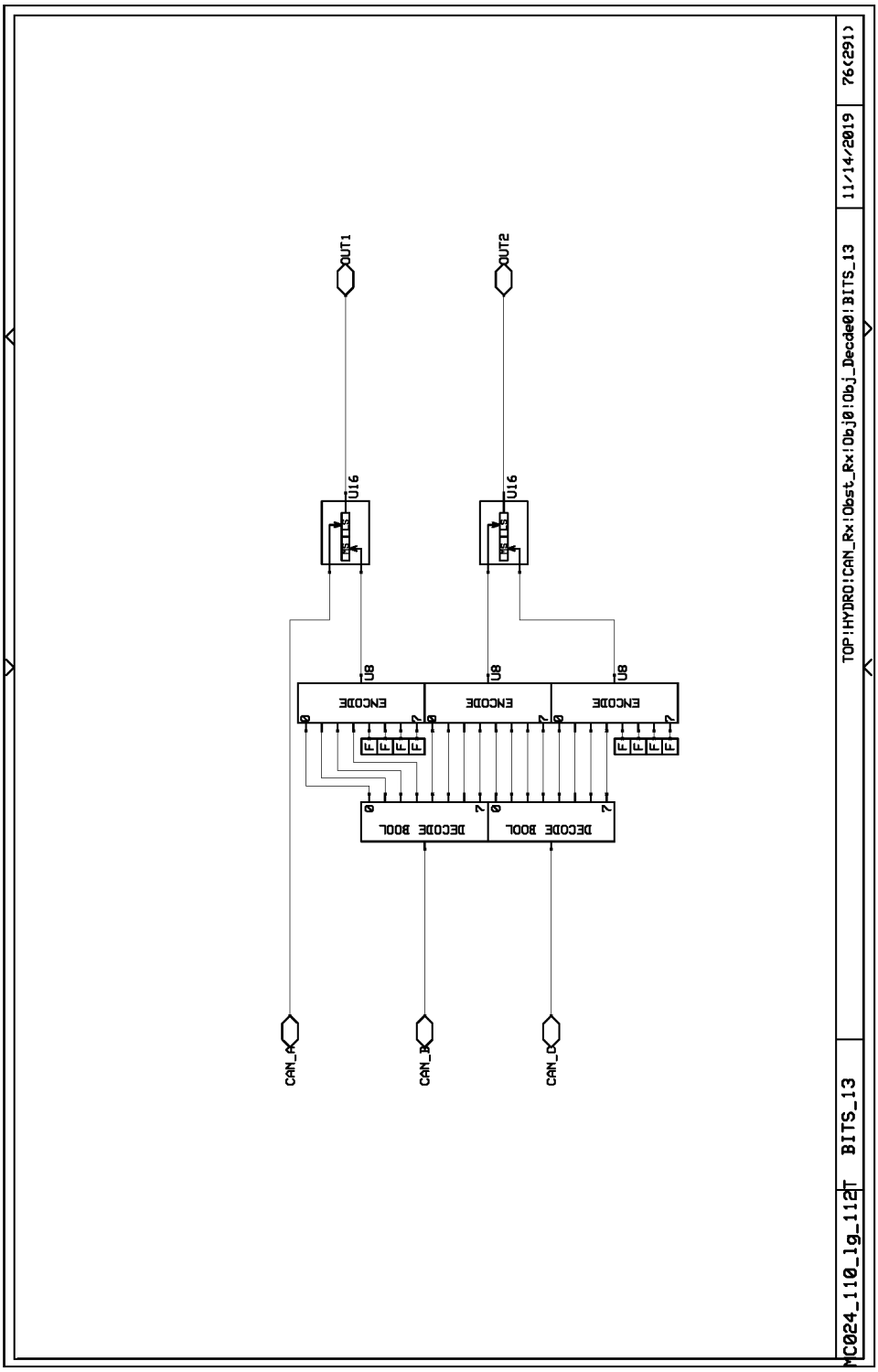
Appendix D.10: Flex-Ro obstacle detection CAN-receive PLUS+1 page.



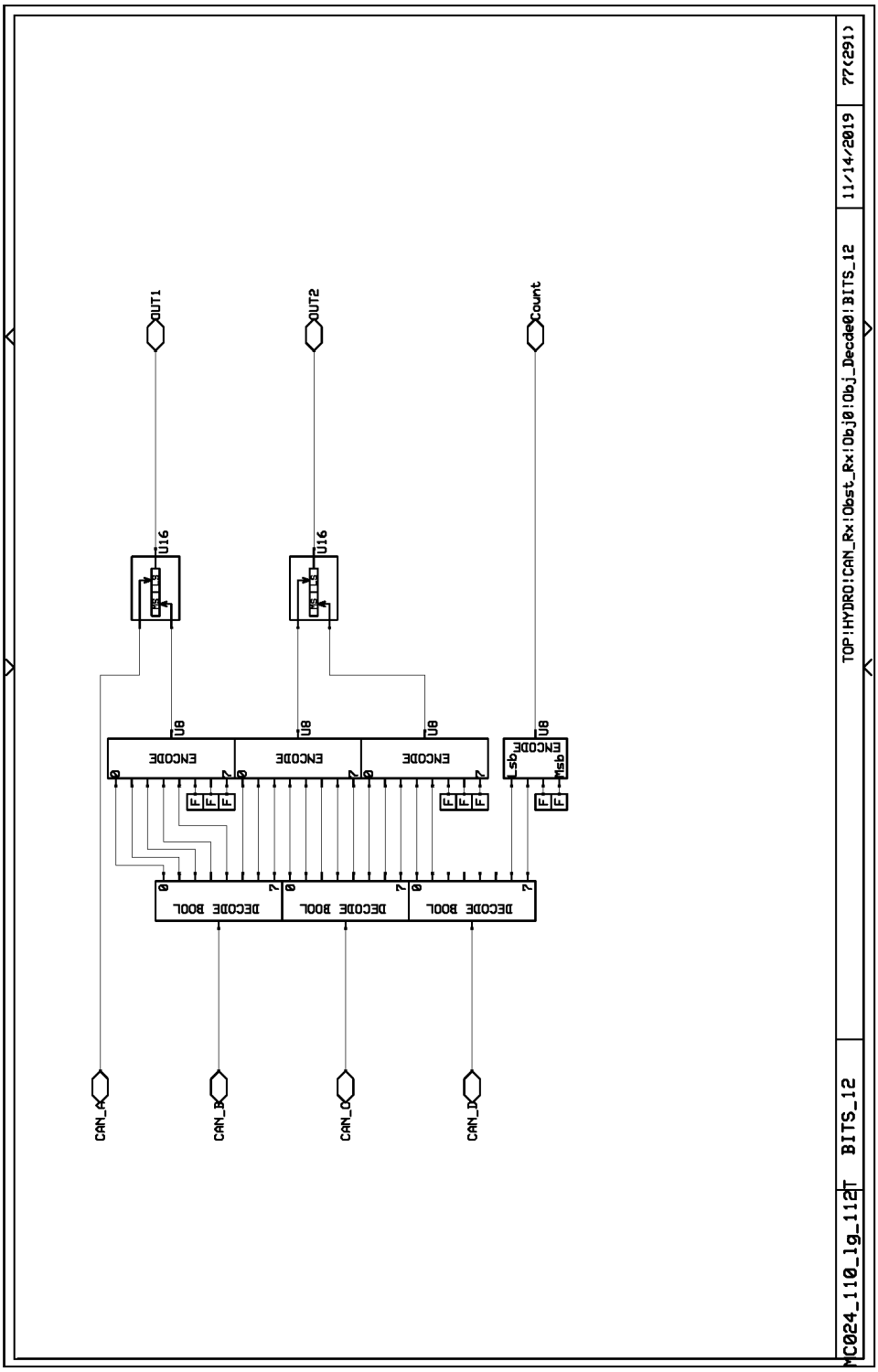
Appendix D.11: Flex-Ro obstacle detection CAN-decode PLUS+1 page.



Appendix D.12: Flex-Ro obstacle detection CAN message decode PLUS+1 page.



Appendix D.13: Flex-Ro obstacle detection CAN message decode PLUS+1 page.



Appendix D.14: Flex-Ro obstacle detection CAN message decode PLUS+1 page.

END OF DOCUMENT