Article

A Novel Multirobot System for Distributed Phenotyping

Tianshuang Gao [1], Homagni Saha [2], Hamid Emadi [2], Jiaoping Zhang [3], Alec Lofquist [4], Arti Singh [3], Baskar Ganapathysubramanian [2], Soumik Sarkar [2], Asheesh Singh [3] and Sourabh Bhattacharya [2,*]

[1] Iowa State University, Department of Computer Science; tsgao@iastate.edu
[2] Iowa State University, Department of Mechanical Engineering; hsaha@iastate.edu, emadi@iastate.edu, baskarg@iastate.edu, soumiks@iastate.edu, sbhattac@iastate.edu
[3] Iowa State University, Department of Agronomy; jiaoping@iastate.edu, arti@iastate.edu, singhak@iastate.edu
[4] Iowa State University, Department of Electrical and Computer Engineering; lofquist@iastate.edu
[*] Correspondence: sbhattac@iastate.edu; Tel.: +1-515-294-0569

Version July 31, 2018 submitted to Preprints

Abstract: Phenotypic studies require large datasets for accurate inference and prediction. Collecting plant data in a farm can be very labor intensive and costly. This paper presents the design, architecture (hardware and software) and deployment of a distributed modular agricultural multi-robot system for row crop field data collection. The proposed system has been deployed in a soybean research farm at Iowa State University.

Keywords: Field Robotics, Multi-robot System

---

1. Introduction

Phenomics, an emerging science discipline, can be defined as the amalgamation of different sensors, platforms, and techniques to collect deep phenotypic data to characterize individuals at multiple temporal and spatial scales at different organization scales (ranging from cellular to ecosystem zones). Plant phenomics relies on the integration of plant sciences, engineering and data analytics; and gives insights not available through conventional experimental methodologies.

Recent research has demonstrated the application of phenomics assisted breeding to increase the rate of genetic gain [1]. Plant breeding enabled genetic gain or improvement is fundamental to meet our growing demand for food. Similarly, plant phenomics is essential to accelerate our understanding of plant responses to various biotic and abiotic stimuli and stresses by drawing a connection to their genetic constitution. However, implementation of phenomics in plant breeding and sciences operations requires: (1) collection of large datasets which are difficult to acquire in field settings [2], (2) automated, timely and cost effective data [3]. Although aerial based systems have been deployed to improve the throughput capability of phenotyping [4,5], these platforms are unable to capture information on few architectural and developmental traits, which are some of the most important traits for growth and development. Without inter-disciplinary plant phenomics, traits such as these would require extensive human labor, which limits throughput and adds cost. Additionally, human measurements are prone to error that can be mitigated with plant phenomics techniques. Therefore, a robotic system that can enable seamless data collection of these traits would empower plant scientists to begin unraveling their genetic mechanisms to meet the needs of human population and its lifestyles.

Several agricultural robots have been developed in the recent past. Unlike industrial robots, the challenges for robots in agriculture are diverse. First, agricultural fields do not have structured and controlled environment, in contrast to industrial facilities. Second, agricultural robots operate on farms with infrastructure and operating conditions different from industrial robots. Third, industrial processes can be designed by modules to complete particular tasks by a particular robot, whereas the complex tasks in agriculture sometimes cannot be split into simple actions. Due to the aforestated reasons, agricultural applications require more versatile and robust robots.

One of the earliest robots deployed in agricultural applications was the German BoniRob [6]. This robot was initially developed by AMAZONEN-WERKE together with the Osnabruck University of Applied Sciences, Robert Bosch GmbH and other partners, and is now a part of Deepfield Robotics, a Bosch start-up company. BoniRob was developed to eliminate some of the most tedious tasks in modern farming, plant breeding, and weeding. Many universities have developed their own robots, for example, the multi-purpose Small Robotic Farm Vehicle (SRFV) from Queensland University of Technology, Australia [7]. This modular design allows the SRFV to undertake a range of agricultural tasks and experiments, including harvesting, seeding, fertilizing and weeding management. Another example is the Thorvald II [8], a completely modular platform both on the hardware and the software side. It can be reconfigured to obtain the necessary physical properties to operate in different production systems, for example, tunnels, greenhouses and open fields. Another example of a university-developed agricultural robot is the Phenobot 1.0 [9] from Iowa State University. It is an auto-steered and self-propelled field-based phenotyping platform for tall dense canopy crops.

Companies that work on commercial robots exists as well. For the open field, there is the ANATIS from Carre, the Robotti from AGROINTELLI and Asterix from Adigo[10]. Ecorobotix is developing a solar powered robot for ecological and economical weeding of row crops, meadows and intercropping cultures, while Naio technologies has developed the robots OZ, BOB, TED and DINO. Other robots are designed for greenhouses. One example is the fully-automatic S55 spray robot by Wanjet, Sweden. The robots described above are a representative subset of existing technology.

An important challenge with phenotypic studies is the use of heavy equipment [6,9] in the fields. The large weight of tractors causes soil compaction, which has several negative consequences, including lower yields and more extensive flooding [11,12]. Therefore, farm machines should be lightweight. This is made possible by introducing lightweight mobile robots in farms. As autonomous robots can work autonomously without the need of a human supervisor, several smaller robots can be used to replace heavy tractor, without affecting the throughput.

Another issue for the robots mentioned above is their fairly fixed physical appearance. Although some of the robots have a few reconfigurable features, for example, BoniRob [6] can for instance change its track width, and Thorvald II [8] can be reconfigured to obtain the necessary physical properties to operate in different production systems, there exists no methodology to build custom agricultural data-collection robots from off-the-shelf robots for phenotyping studies.

In this paper, we present the design, construction and deployment of a lightweight distributed multiple robot system for row crop phenotypic data collection. The paper is organized as follows. Section 2 elaborately describes the design of the robots, including software and system architecture. Section 3 details the experimental field setup. Section 4 presents our conclusions.

2. Materials and Methods

Fig 1 shows a schematic of the deployment of the ground robotic system presented in this paper. The autonomous phenotyping system consists of 2 parts:

1. Smaller light weight robots equipped with low resolution sensors that acquire frequent measurements, and infer changes that take place at small-time scales. We refer to them as **rover**.
2. A big robot carrying high resolution sensors for accurate plant disease detection and analysis of spread of disease. We refer to it as Modular Autonomous Rover for Sensing **MARS**.

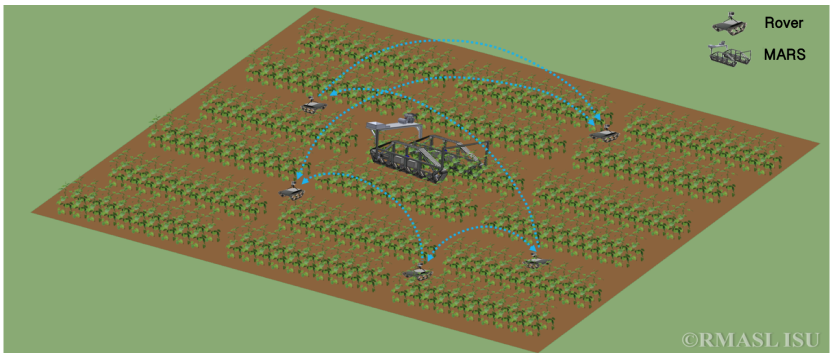


Figure 1. Deployment strategy of robotic phenotyping system

In the remaining part of the section, we provide details regarding the rover and the MARS.

2.1. Robot Configurations

2.1.1. System Requirements for Rover

For phenotyping studies in row crop field, the critical requirement of the robot is that it must fit in between two crop rows and take images without running over the crops. The target soybean plants can grow up to 60cm. The robot must be able to carry a 1 Megapixels RGB camera at 120cm height. Uneven ground and small obstacles should not affect the camera view angle and image quality. The robot has to work in conditions when the ground is damp and muddy. It should maintain high mobility in such condition. The robot also needs to have enough power to be able to navigate the entire field. It must navigate in the field autonomously with minimum human interaction. However, a human operator should be able to override or stop the robot if necessary. The collected images have to be processed on-board with geo-tag and timestamp, and transmitted to a computer.

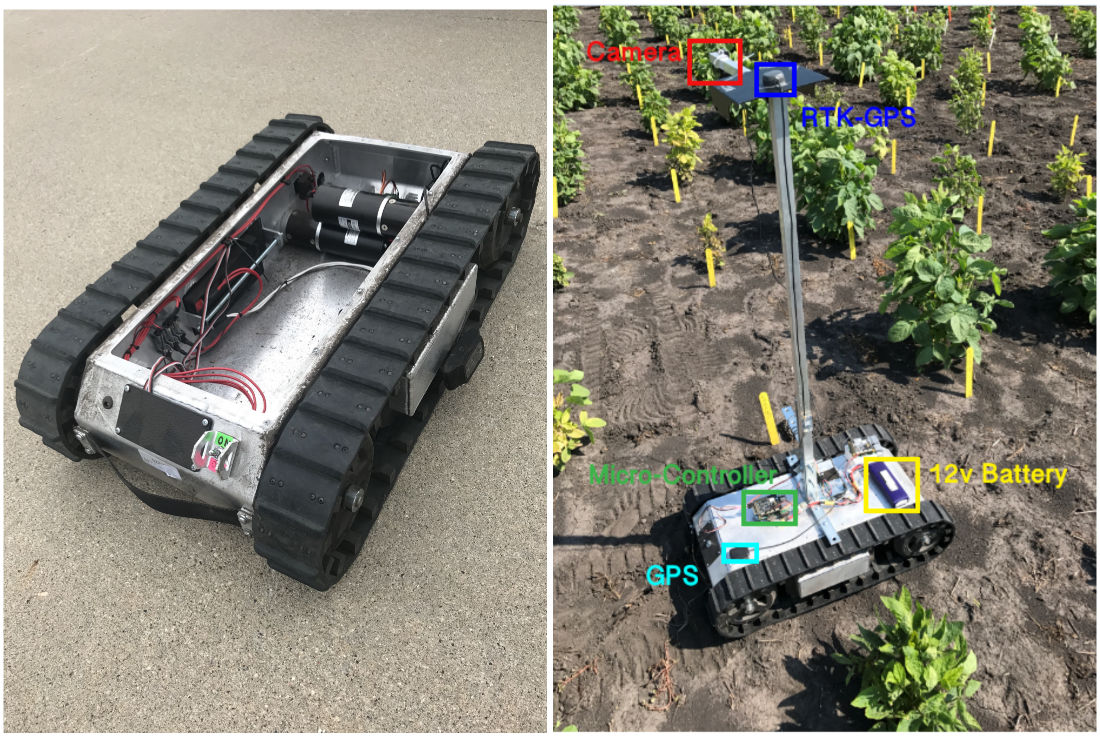


Figure 2. The figure on the left shows the rover. The figure on the right shows the rover in the field taking images of the canopy.

Fig 2 shows the rover. The platform is based on a SuperDroid LT2 sold by SuperDroid Robots. The length of this rover's base is 67cm and the width is 42cm. With a ground clearance of 13cm, the robot can handle most terrains with its aggressive all terrain treads. The weight of this robot is 25kg. With most of it's weight on its base, the robot does not tilt or tip easily in the out-door environment. With the height of 120cm, we are able to mount the imaging sensor anyway in the range of 80 to 120cm. The robot can handle 25kg of additional payload. Pair of tracks driven by the two gear motors can provide a maximum speed of 83m/min. Two 12 Volt 7.2 Ah lead acid batteries in series can provide 2 hours of lifetime in one charging cycle. Fully recharging the batteries takes 0.8 hour.

The imaging system is attached to the metal frame by using a L-shape bracket and an extension aluminum support. The height of the system is adjustable between 80cm and 120cm. To keep the camera steady to obtain stable images, a 3D gimbal is utilized. The camera will always maintain a straight down view of the ground, even when the robot base is tilted caused by uneven ground. Images from on-board camera can be processed efficiently through the strong computing power of Raspberry Pi.

2.2. MARS

The rovers communicate with MARS through the communication layer, presents in section. MARS can acquiring hyperspectral images of selected IDC plants with an onboard Resonon Pika xc camera. It helps to reinforce beliefs about disease propagation, reducing uncertainty of path planning, and provides plant scientists with high quality data to further understanding of the nature of disease. Based on its unique design, MARS creates an artificial imaging environment over each plant, thus ensuring noise-free and consistent images, with an imaging speed of 2 minutes per plant. The hyperspectral data-cube is stored in a dedicated DL(deep-learning) computing platform on MARS which runs deep-convolutional neural networks using dedicated GPUs on the data to infer spatio-temporal progression of the disease. The waypoints are generated by an online importance sampling algorithm developed which minimizes the prediction error at the next step under measurement constraints. In this section we present the design of the MARS.

We formulate the design requirements for such a robot, and give a detailed discussion on realization of such a design. The dynamics of the robot is represented in the state space form and a brief abstraction is provided through a control flow diagram for the auto-steer system with a unique feature that provides the option for parallel user input for increased safety and more flexible decision making. It is then used in an opportunistic sensing scheme for taking high level decisions that makes decision on which region of the crop farm to move next.

2.2.1. Design requirements for MARS

The MARS must either fit within two crop rows or straddle a number of crop rows without causing any damage to the crops themselves. The MARS must be able to tread cultivated land containing previous season crop residue and various other small obstacles not exceeding 5" in height. Small obstacles must not affect the smooth and consistent operation of the MARS with regards to picture quality or tilting/tipping. The MARS must be able to work in conditions where the ground is damp and slippery. This requirement must be fulfilled in two terms. MARS should not lose traction, getting stuck in the mud and becoming immobile, and the components that make the MARS functional should remain undamaged and operable. The MARS must be able to traverse a field where parts of the field may have a grade. The maximum field grade that the MARS must be able to traverse is 30. In terms of power requirement, MARS must have enough power to climb the required grade, maximum grade being set as 30 degrees without tipping over while treading. The MARS must be able to traverse fields and operate for about 6 hours continuously. The MARS must be able to navigate through the field and through the crop rows, taking pictures of soybean plants, by working off of GPS start and end coordinates that will also be used to geotag each picture. The MARS must operate smoothly, take a photograph, and then restart smoothly. The MARSs operation must be completely automated. At the same time a human operator shoul be able to override its operations, if needed. The MARS must perform all of its functions and gather the data independently.

2.2.2. System overview

Keeping in mind the design requirements we propose a conceptual design as shown in fig 3.
1- Steering wheel for manual steering
2- Mounting for computation equipment, laptops, etc.
3- Platform for raising the entire imaging system upwards from the MARS
4- Frame for shading
5- Mounting point for camera system
6- Screw system for vertical movement of imaging system
7- Stepper servo for horizontal movement of imaging system
8- Fuel cylinder
9- Hydraulic pump
10- Tank or reservoir for hydraulic circuit
11- Linear actuators for raising the imaging system out of the MARS
12- Location of single sided hydraulic ram for steering
The length of the MARS is3.78m and the span of is 1.86m so that it could span over two consecutive rows.

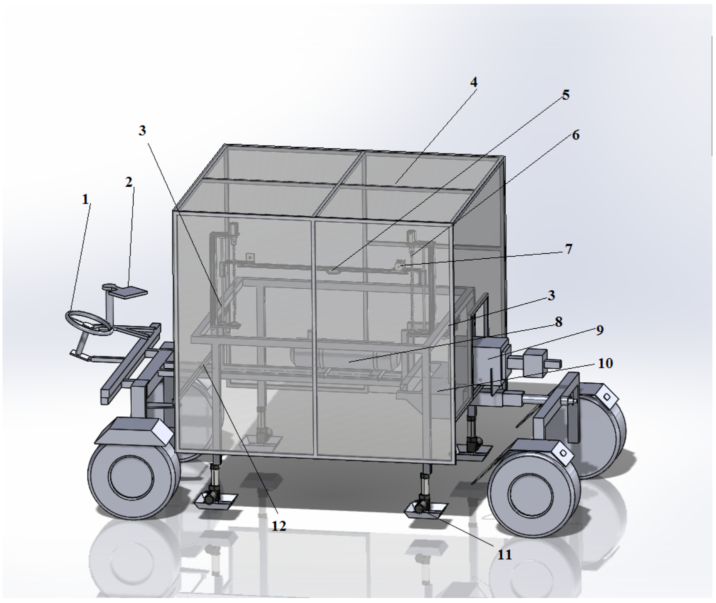


Figure 3. Conceptual design showing interest points

The primary components of the MARS and each of their functions is summarized as in fig 4

2.2.3. Transmission and actuators

Due to the high power density of hydraulic drive systems the vehicle is chosen to be a two wheel hydro static drive. The hydraulic system is controlled by our micro-controller through PWM channels, so that the MARS can be controlled by Cy-Eye system 2.3. In this section a preliminary calculation of torque and traction requirements is done. With an assumed mass of 680.38 kg the force that needs to be applied to overcome gravity can be calculated from the following equation.

$$F_g = \frac{1}{2}Mg \tag{1}$$

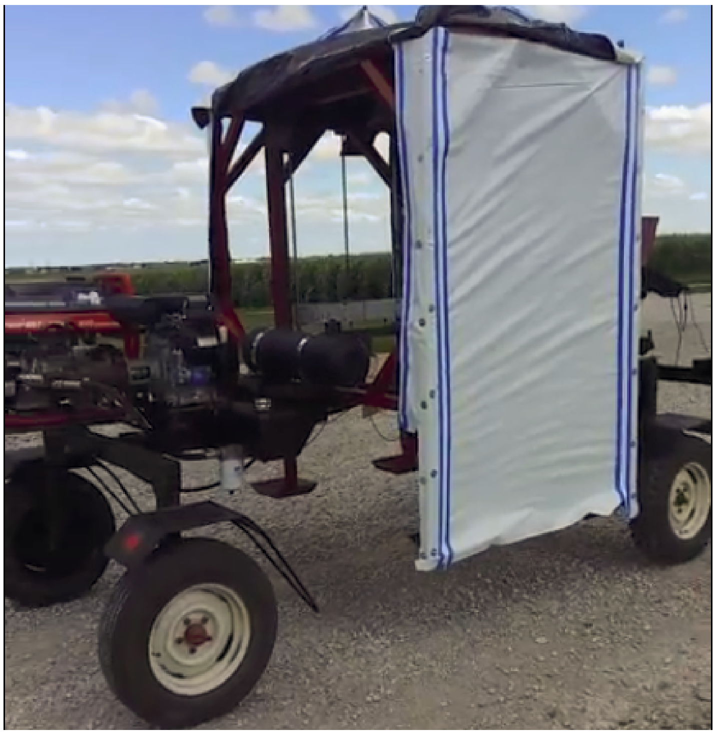


Figure 4. MARS moving out to capture soybean images

From the above equation we obtain the force to be 3337.26 N. The electric motor must also overcome the rolling resistance of the track. The coefficient of rolling resistance is assumed to be .5 (Car tire in sand = .5). The force of rolling resistance can be calculated from the following equation

$$F_r = MgC_{rr} \tag{2}$$

From the above equation we obtain the force to overcome the rolling resistance as 3337.36 N. Thus the total force is equal to 6674.52 N. Assuming an average speed of 5 ft/s and wheel diameter of 2.08 feet, the RPM required is given by the equation

$$RPM = (60fps)/(\pi D) \tag{3}$$

Where fps denotes feet moved per second and D denotes the wheel diameter. From (3) the required RPM is 46. The average mechanical Horse Power (HP) is calculated from the product of force required and velocity -13.63. Thus the torque from the HP and RPM is given by the equation

$$T = 6932.75HP/RPM \tag{4}$$

T is equal to 2110 N-m which is easily realizable by a pump of maximum pressure 2.5Kpsi. The problem of finding the overcoming force for traction is solved in a manner similar to the motor torque problem. The force pushing the rover down the incline of 30 degrees will be same as from (1). Traction force must overcome this force for the rover to move up the hill. The coefficient of friction between a rubber track and loose soil is found to be .8. The Traction force can be calculated from the following equation

$$F_t = \frac{\sqrt{2}}{2}C_tMg \tag{5}$$

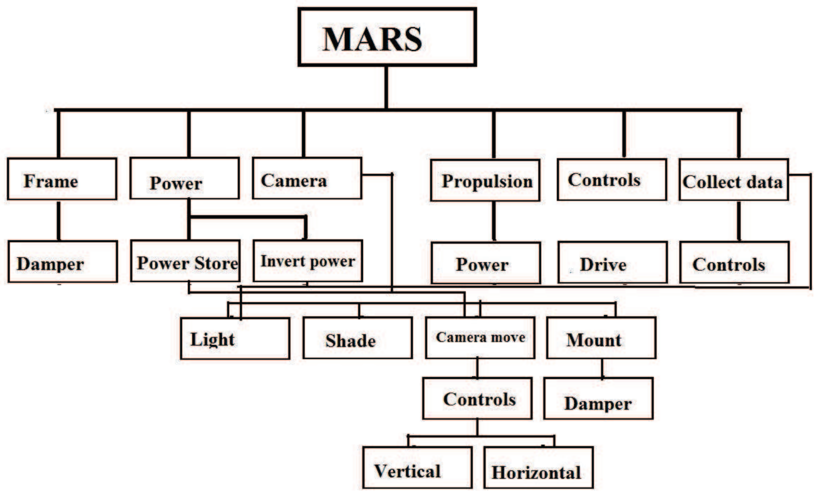


Figure 5. Functional design tree

From (5) the required force is 4624.30 N .The total force would again be obtained by the sum of the above two forces. Since the force is greater than the force pushing the rover down the hill, the rover will be able to climb the hill.

2.2.4. Auto-steer

In this section, the automatic steering system for the phenotyping robot is described. The purpose of the autosteer is to automatically guide the robot along the field tracks using a GPS based navigation system. Use of autosteer technology in precision agriculture can be found in [13], [14], [15]. The primary challenges faced by the autosteer system in this project are:

1. Low power consumption : The steering unit as a whole should be much better off consuming the lowest possible amount of power possible as it is a crucial factor in the utilization of power by the imaging system consisting of the actuators moving the hyper spectral camera, the lighting system and the hyper spectral camera itself. Considering the long hours of operation in the field for data collection, the power consumption of the autosteer should be minimal.
2. Parallel operation : In several situations a driver may be monitoring the functioning of the robotic system, and may need to manually override the system in case of emergencies, thus a two way control is desirable.
3. Compact and robust system : Considering the operating conditions of uneven ground and dirt, the autosteer should be able to last for a long time. Also it should not cause any safety concerns for the driver in case of monitoring operations. Before moving on to the actual implementation of autosteer the implementation of steering on the robot is discussed. A hydraulic power steer is used to aid the driver in case she needs to drive manually. The base of the steering wheel is connected to a spool valve with spring return which opens to either side based on the direction steered by the driver. The opening of the spool valve completes the connection from the high pressure line of the pump to one of the sides of a single acting hydraulic ram. Based on the direction steered by the driver, the ram either pushes or contracts. The acting side of the ram is connected to a common bar using revolute joint. The common bar connects both the wheel hubs using two revolute joints.

Three possible options were evaluated for implementing the auto steer:

1. External electro-mechanical system on top of steering wheel: In this system, a sprocket would be welded to the base of the steering wheel rod, and a chain drive would be connected to an

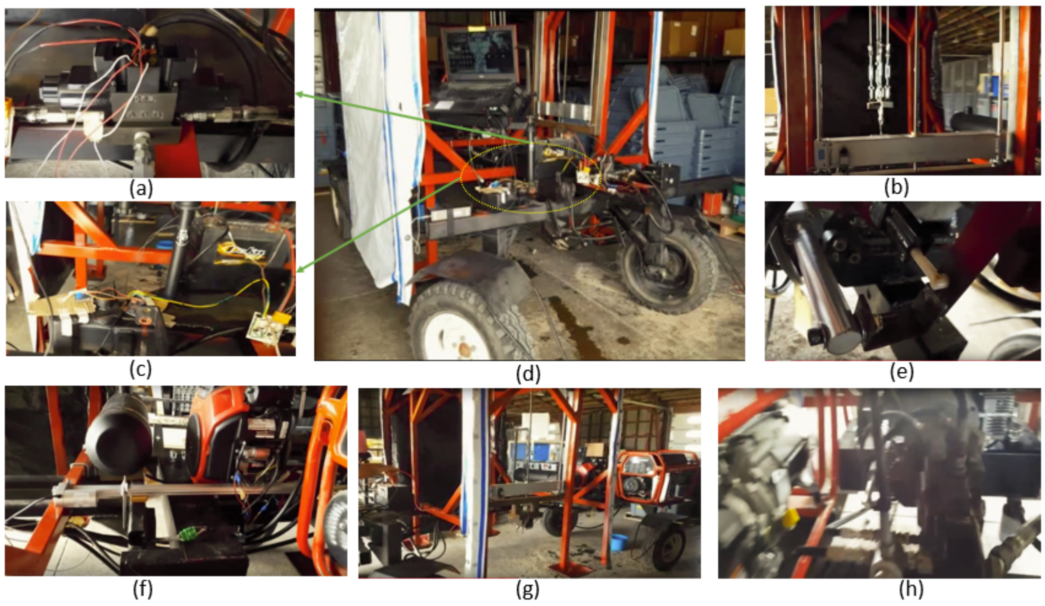


Figure 6. Close up views of various parts of MARS- (a) Electric actuated hydaulic flow control valve, (b) Winch to engage and disengage camera system along with vertical camera motion screw, (c) Switching relays, arduino and autosteer controller (battery at back) (d) Front view of MARS with laptop as main controller (e) Acceleration actuator solenoid pushing the directional valve on pump (f) Linear actuator solenoid, fuel tank, and engine (g) Detachable Imaging system block (h) Generator for artificial lighting and hydraulic pump

external servo motor. The major limitations of the system would be high power consumption and relatively longer assembly time as it would require design of mounting points for the chain drive keeping in mind the safely of the drive. Additionally, the control would be relatively complex in case the driver wants to override the autosteer.

2. Electric steering along with electronic control unit: This would require an electric motor steering instead of hydraulic power steering.The major limitation of this system is in the high cost of the entire system, and relatively longer times in establishing the control algorithms due to involvement of a lot of parameters in the ECU. This system too is expected to consume high power.
3. Electronic hydraulic flow control valve in tandem with hydraulic power steer: In this implementation, the high pressure connections from the pump and the connections to either side of the ram is fed to a three way flow control valve. The flow rate is calibrated using an external flow control valve in series to the electronic double solenoid valve. A linear potentiometer installed on the single acting side of the hydraulic ram provides the necessary feedback to control the amount of time the valve remains open in one of the three directions. The circuit is summarized in Fig. 7. The following circuit is also actually implemented in this paper.

Based on the hydraulic circuit conceived, a control model is formulated for the autosteer and is presented in Fig 8. It takes in the difference of the desired value of steering angle and the sensed value of the current steering angle based on linear potentiometer attached to the hydraulic ram(hence giving us x_h) and applies it to the controller. A PI controller would be sufficient for the desired purpose. The justification for the estimated plant model can be found in the section on navigation control.

2.2.5. Imaging System

The imaging system of the MARS was designed with the motive of capturing RGB as well as hyperspectral images of plants of interest in the shortest amount of time and with minimum loss of

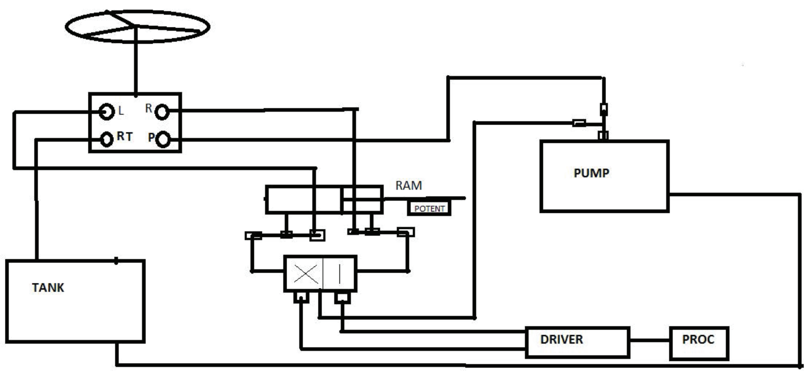


Figure 7. Hydraulic autosteer schematic

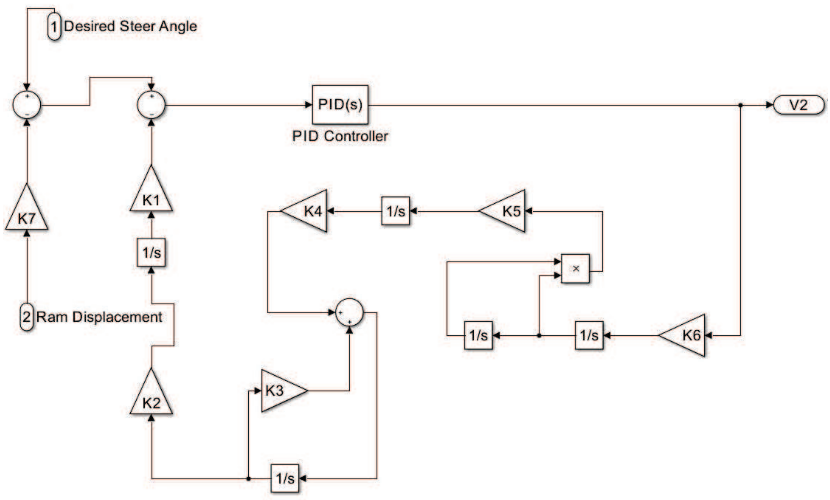


Figure 8. Control block diagram

image quality. It consists of five main subsystems: The camera system, he camera platform system, the shading system, the camera movement system and, the camera mount and lighting system.

2.2.6. Hyperspectral Camera System

The hyperspectral camera used in research is a Pika XC line scanning imager from Resonon. The hyperspectral camera captures data in the form of layers, at a wider range than the visible light spectrum. Line scanning means that the stage moves while the camera is stationary and the image is collected one line at a time. This camera captures a range from 400-1000 nm divided into 240 wavebands. So there are 240 layers with each layer being a black and white reflectance map showing levels of reflectance at each specific waveband. The lighting provides even illumination over 350-2500nm which is necessary for hyperspectral data collection. And the camera is calibrated to a white reference bar prior to each day of imaging so that lighting conditions are standardized. Hyperspectral camera offers extended wavelengths as well as advantage for early disease detection because we're able to observe internal symptom development before we can visually see the development of disease symptoms. Fig 9 illustrates a 1000x1000x125 hyperspectral image of soybean plant canopy captured using a snapshot hyperspectral camera. The hyperspectral image consists of 125 wavelengths from a range of 450-950 nm at a spectral sampling interval of 4nm.

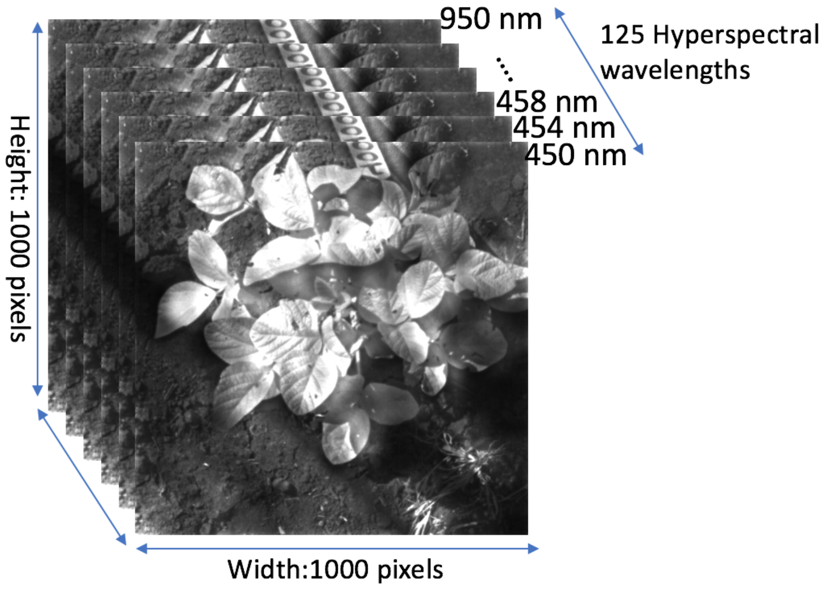


Figure 9. Hyperspectral image of a soybean plant canopy

2.2.7. Camera Platform System

The camera platform system consists of two parts. The first part is four 80/20 aluminum bars. Those bars are connected in a rectangular shape of length 68.2 inches and width 60.5 inches. An 18 inch by 3 inches by 1/4-inch steel plate is bolted to each transverse bar's center. Each steel plate has two, 2-inch diameter, cylindrical rods welded 13 inches apart. Each rod is 2.5 inches long and has chamfered ends of distance 1.5 inch. These rods fit in a counterpart on the second part of the camera platform system that is fixed to the MARS. This is the point of engagement and disengagement between the camera platform system and the MARS. Once the platform is disconnected from its counterparts, it rests on four 35-inch-long legs. Two legs are connected at the rear end of the longitudinal bars, and the other two at 58.2 inches from the rear end. Next, four electrically powered linear actuators are mounted at the bottom of each leg to disengage the platform from the counterparts on the MARS. The actuators are connected to four 12 inches by 12 inches by 1/8-inch steel sheets that act as feet to reduce pressure and prevent the actuator from sinking into the ground. The second part of the camera platform system are the counterparts that the platform is fixed on. These are flat steel plates are like the other two but with holes instead of rods. Each of these is fixed on 14-inch 80/20 aluminum that are perpendicular to MARS main axis. All 80/20 connections are done using anchors. Figure 10 shows the design in detail.

2.2.8. Shading System

The shading system was designed in order to block direct light from the sun from reaching the hyperspectral camera and interfering with image capturing of the Pika XC hyperspectral camera. Two risks were identified through the Design Failure Mode Effect Analysis (DFMEA) that have a Risk Priority Number (RPN) higher than 120 which was considered an RPN high enough to lead to additional consideration. Those two risks were the shading material tearing or the shading system tipping. Initially the design was to place the shading on the camera platform system. After calculations were completed it was identified that this option would not be feasible with the average wind speeds in Iowa. The calculations for tipping are shown in table 1 Variations in wind speed would also lead to vibration transfer to the hyperspectral camera. For these reasons the shading frame was mounted to the rover and not the camera movement system. Separating the shading from the camera frame ensured that the hyperspectral camera would be under the most ideal conditions possible in an outdoor

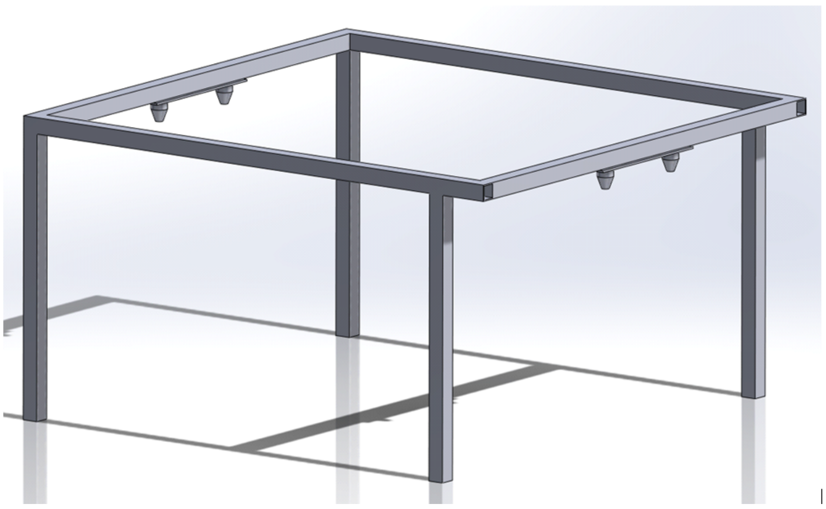


Figure 10. The frame design for camera platform

Table 1. Risk assessment for tipping of shade to wind speeds

Vehicle speed(ft/s)	Tipping Force(F)	Moment(lb.ft)	Wind speed (mph)	Force per unit area(F/A)
5	1.213	2.274	3.409	0.000
10	4.851	9.095	6.818	0.001
15	10.914	20.463	10.227	0.002
20	19.402	36.379	13.636	0.003
25	30.316	56.842	17.045	0.005
30	43.655	81.853	20.455	0.008
35	59.419	111.411	23.864	0.011
40	77.609	145.516	27.273	0.014

setting. Mounting the shading frame to the heavy rover also mitigated the risk that the shading system would tip over. Next, the shading material was chosen to minimize the risk of tearing, degrading due to exposure to the sun, the ability to block out light, and ability to minimize heat build-up from exposure to radiation from the sun. The BOLD, blackout light deprivation tarp, made by Americover was found to be the ideal solution. The BOLD tarp is specially designed for greenhouses. The tarp is made from a mix of polyethylene resins and is scrim reinforced. This ensures that the tarp is tear resistant and is strong enough to withstand the expected conditions, and polyethylene also does not degrade in sunlight. The tarp consists of two layers. First, the inner black layer contains carbon black which ensures total light deprivation. Next, the outer layer is white and contains UV inhibitors and thermal stabilizers to reduce heat and condensation build-up. The Bold tarp was specially designed to minimize every risk that had been identified with the shading material, and that is why it was chosen. The geometry of the shading system is shown in Figure 11 . It was chosen in order to minimize size, limit light introduced to the hyperspectral camera, and fit around other parts of the rover.

2.2.9. Camera Mount and Lighting System

The camera and lighting system was constructed in two main assembles. One of the assemblies was the fabrication of the lighting frame. This frame was constructed using 1 inch outside dimension square 6063 aluminum tubing with 1/8-inch wall thickness. This material was selected because it provided flat surfaces for simple fastening of the sections using aluminum corner brackets and for mounting the lights to while also being a lightweight material that kept the weight of the lighting frame to a minimum Four purchased corner brackets and 16 1/4-20 bolts are used to construct the frame into a square which measures 9 in on the inside and 11 in on the outside. This frame will be

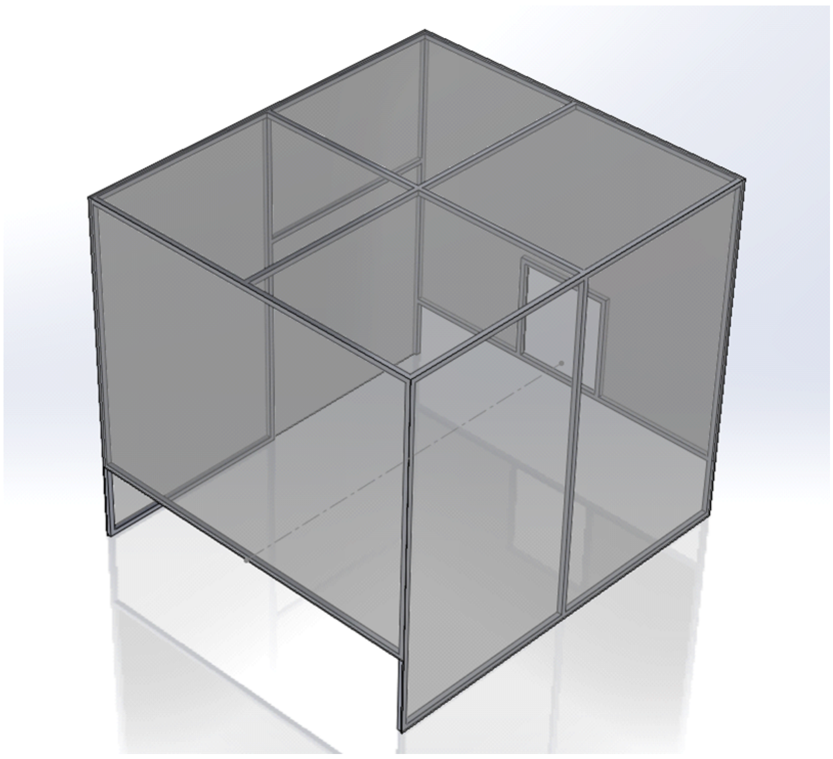


Figure 11. The shade design for camera platform

assembled to the camera bracket in a way that the light frame will completely encompass the bracket and the hyperspectral camera as shown in figure 12.

The second major assembly of the camera and lighting system is the camera bracket which is a mount that secures the hyperspectral and GoPro cameras to the camera movement system. This bracket is comprised of two 1/4-inch-thick A36 low carbon structural steel that are cut and drilled to specific dimensions and two equal sized support bars of 1018 steel cut to the proper dimensions. One plate is fabricated to hold both the Pika XC hyperspectral and GoPro cameras while the other plate is fabricated to fasten the camera mounting plate to the camera movement system and to secure the belt for the horizontal pulley system. A 1 foot by 2-foot plate of the A36 steel was purchased and cut to the size using a combination of the band saw and horizontal band saw. To secure the lighting frame to the camera bracket, a steel brace was designed. This brace was fabricated from a 1/8-inch-thick steel plate and is cut to be 9-inch-long and 1 inch wide. Two tabs of the same material were fabricated to be 2-inch-long and 1 inch wide. The tabs were fixed to the 9-inch brace component with the use of MIG welding only to the inside of the seams of the tabs and back. Finally, once both subassemblies are built, and assembled to one another the cameras and lights were secured in the proper locations using bolts which have been purchased. The Pika XC camera had a 3D printed housing created from a SolidWorks CAD model. The inside of this housing will possess a layer of the vibration isolation pad to additionally limit vibration to the camera. Another layer of the isolation pad will also be placed between the metal of the camera bracket face plate and the camera housing for the same purpose. The vibration isolation pad will be cut using shears and a hole drilled using a drill press. Gorilla glue will be used to adhere the pads to the surfaces.

2.2.10. Camera Motion Control

Here we provide a brief overview of the factors involved in camera motion control system through figure 14. Through proper experimentation each of the relevant control parameters were found out. Additional option is presented to users of the system to change the parameters as they wish.

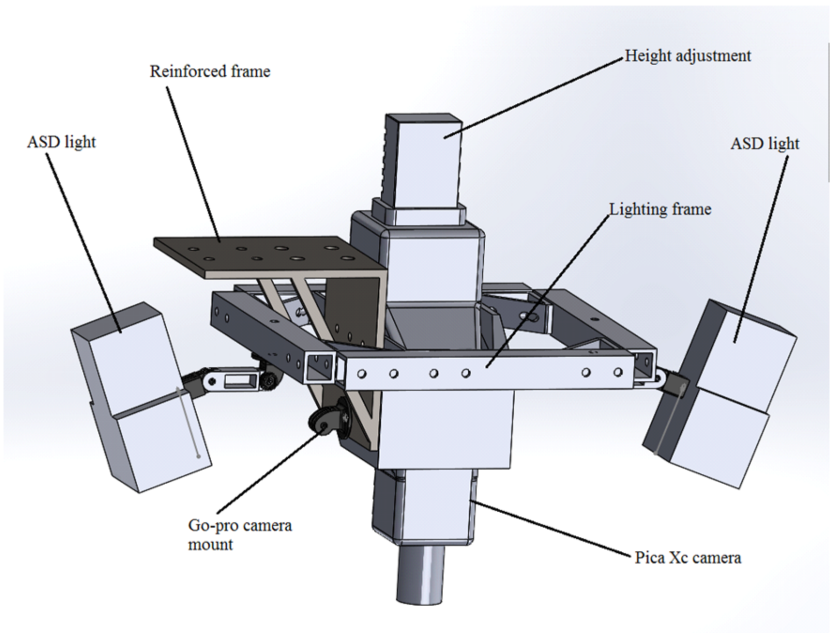


Figure 12. The camera mount design

2.2.11. Controller Formulation

In this system, we will develop a very basic control for navigation of the MARS based on the coordinates mapped by the RTK-GPS navigation system. The first step is to model the non-holonomic constraints and develop a kinematic equation. The following assumptions are taken for motion of the MARS:

1. No slippage.
2. Low angles of steering which reduces the problem formulation to a problem very similar to bicycle motion, in which considering the motion of only front and rear wheels of one side of the vehicle is enough.

Let u and v be the coordinates with respect to the centre of gravity of the vehicle and (x,y) be the coordinates in the frame of RTK-GPS. Figure 9 shows the frames and the associated directions. From [16], the kinematic equations combined with the dynamic equations can be obtained and written in the state-space form as the set of following equations

$$\dot{x} = \left[cos(\theta) - \frac{btan(\phi)sin(\theta)}{l} \right] u \tag{6a}$$

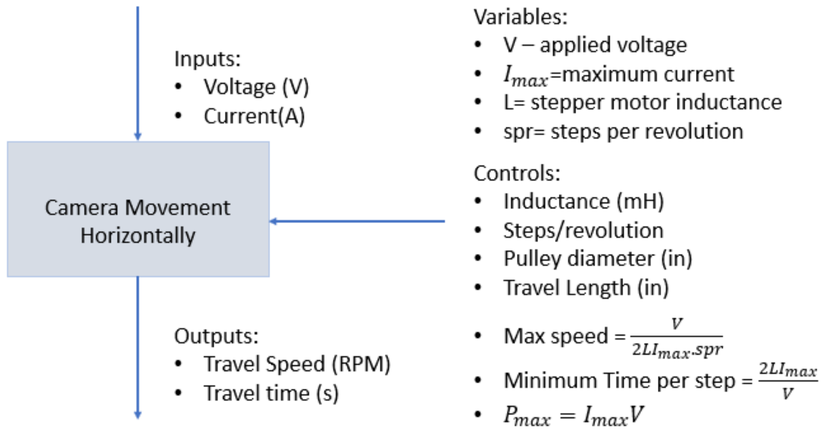
$$\dot{y} = \left[sin(\theta) + \frac{btan(\phi)cos(\theta)}{l} \right] v \tag{6b}$$

$$\dot{\theta} = \frac{tan(\phi)}{l} u \tag{6c}$$

$$\dot{u} = \frac{u(b^2m + J)\dot{\phi}tan(\phi)}{\gamma} + \frac{l^2cos(\phi)^2F_d}{\gamma} \tag{6d}$$

$$F_d = g(u, v_1) \tag{6e}$$

$$\dot{\phi} = h(\phi, v_2) \tag{6f}$$



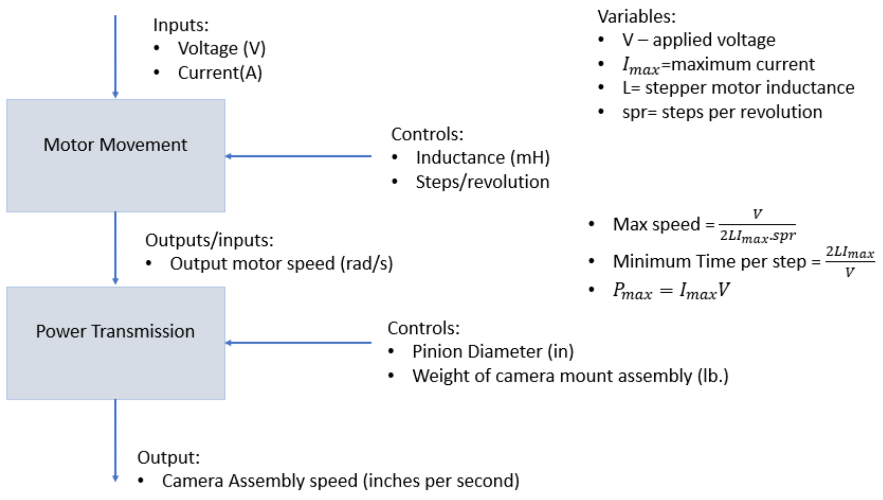

Inputs:
• Voltage (V)
• Current(A)
Motor Movement
Variables:
• V – applied voltage
• I_{max}=maximum current
• L= stepper motor inductance
• spr= steps per revolution
Controls:
• Inductance (mH)
• Steps/revolution
Outputs/inputs:
• Output motor speed (rad/s)
• Max speed = $\frac{V}{2LI_{max}.spr}$
• Minimum Time per step = $\frac{2LI_{max}}{V}$
• $P_{max} = I_{max}V$
Power Transmission
Controls:
• Pinion Diameter (in)
• Weight of camera mount assembly (lb.)
Output:
• Camera Assembly speed (inches per second)

Figure 14. Horizontal (top) and Vertical (bottom) camera movement

314 The equation for gamma is given by $\gamma = cos(\phi)^2 \left[l^2 m + (b^2 m + J)(tan(\phi)^2) \right]$

Thus v_1 and v_2 are the two control variables. g(t) and h(t) are arbitrary functions of the variables u, v_1 ,v_2 and ϕ .More precisely, v_1 is the input voltage to the linear actuator that controls the throttle to the pump which in turn controls the flow rate to the hydraulic motor and hence the driving force of the MARS. v_2 is the input voltage to the three way solenoid controlled on-off valve which actuates the hydraulic ram in either direction and hence varies the steering angle ϕ of the MARS. The load torque on the hydraulic motor when the flow is varied using a valve is given by the valve control of motor motion (VCMM) equation as follows.

$$P_s = \frac{D_m^2 N_m^2}{3600\eta_{vm}^2 K_{vt}^2} + \frac{2\pi T_l}{\eta_{mm} D_m} \tag{7}$$

In the control law formulation, P_S will be assumed to be constant throughout operation over time. Also the motor displacement is constant over time. In the simplest case it is assumed that the valve coefficient is linearly proportional to the displacement x of the linear actuator and the force exerted by the linear actuator is directly proportional to the applied control voltage v_1. Also it may be assumed

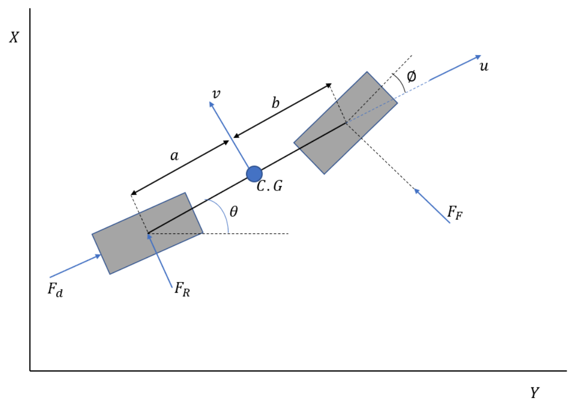


Figure 15. General coordinates for simplified car like robot

that load torque at the output of motor is directly proportional to the driving force and rpm from the equation above is directly proportional to the velocity along the axis of the vehicle u. This gives rise to three states as shown in the following equations.

$$\dot{x_s} = x \tag{8a}$$

$$\dot{x} = K_s v_1 \tag{8b}$$

$$F_d = P_s - \frac{k_1 u^2}{x_s^2} \tag{8c}$$

As mentioned in the section of autosteer, the three way hydraulic valve can be assumed to contain a bidirectional needle valve for adjusting the flow rate in both the directions which is used for actuating the hydraulic ram turning the wheels left and right. Thus the steering angle ϕ is linearly proportional to the displacement $\mathrm{x_h}$ of the hydraulic ram. let $x_h = K_\phi \phi$

For controlling the 3 way valve a bidirectional needle valve is used and the number of turns on the needle screw is linearly proportional to the flow rate across the valve. The flow rate across the valve and to the piston affects the pressure difference across the piston which varies according to the square of flow rate. If the needle valve is controlled by a linear actuator, the force on the screw is proportional to the control voltage v_2. To summarize, we have the following set of equations for the states.

$$\dot{\phi} = \frac{x_{h1}}{K_\phi} \tag{9a}$$

$$\dot{x_{h1}} = \frac{K_p P - C x_{h1}}{M} \tag{9b}$$

$$\dot{P} = K_t t t_1 \tag{9c}$$

$$\dot{t_1} = K_v v_2 \tag{9d}$$

$$\dot{t} = t_1 \tag{9e}$$

In the above equations C is the damping constant for the hydraulic circuit, and K_p is the constant associated with pressure difference. Thus the control flow diagram for the autosteer of the MARS is obtained as shown in the earlier section on autosteer in figure 8. For simplicity the constants starting from $1/(K_\phi)$,(K_p/M),$(-C/M)$,K_t and K_v is replaced by simple constants K_1 to K_7.

2.3. System Architecture

In order to send waypoints and monitor the position and attitude of rovers and MARS, we link the ground robots to our ground control station , which is a part of a larger system called Cy-Eye. Both rover and MARS communicate with each other through Cy-Eye. The system, shown in Fig 16, is a fully autonomous waypoint based system that consists of a centralized multi-robot ground station that can assign target points for every robot, and generate waypoints on the given field map. Since Cy-Eye system is a fully autonomous GPS waypoint based system, navigation is entirely calculated on-board based on the given waypoints. Unlike most current systems that require a human to remotely control them, the centralized ground station sends each robot a list of generated waypoints. After each robot receives the waypoints, the onboard controller with on-board sensors can navigate each robot to follow the given path, and eventually reach to the target. We also use this station to initiate commands, and monitor rovers' status while in operation. It takes in a pre-generated GPS waypoints array for each rover before a human initiates the experiment. Then, it transmits the waypoints to each rover through the Wi-Fi network. Once each rover received the waypoint, an operator can initiate the imaging process. During the process, the operator can view live image and collected data through the graphic user interface. The robots send back their GPS locations every second. The operator interacts with the robots by viewing and operating the ground station only. The collected data will be saved both on-board and in the ground control station. If emergency occurs, the operator is able to interrupt the process by switching into manual mode. Meanwhile, the ground station is responsible for coordinating the path for multiple robots based on UDP protocol for collision avoidance. After two pre-generated waypoints overlap, it will calculate two alternate non-overlapping waypoints and update the original waypoints.

The system uses Wi-Fi network to communicate over the local network or internet between the ground station and the robots. The protocal is based on Mavlink and UDP.

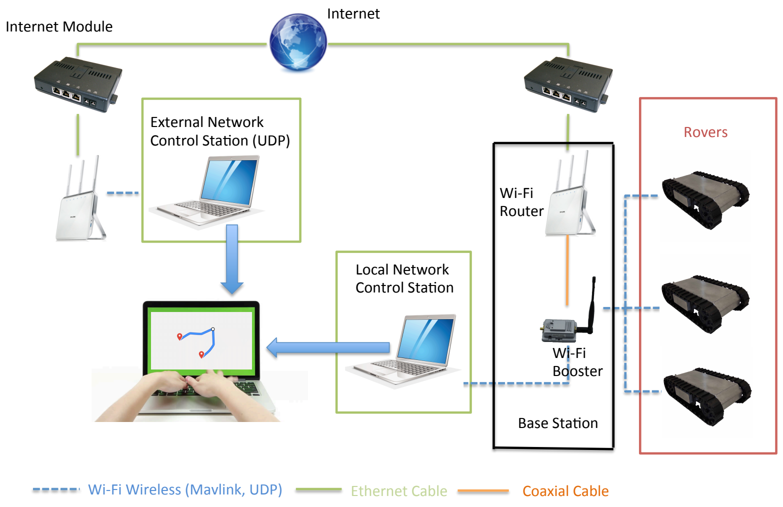

Internet Module
Internet
External Network Control Station (UDP)
Rovers
Wi-Fi Router
Local Network Control Station
Wi-Fi Booster
Base Station
Wi-Fi Wireless (Mavlink, UDP)
Ethernet Cable
Coaxial Cable

Figure 16. Architecture of Cy-Eye.

2.4. Micro-Controller and Sensors

The micro-controller on-board is a common Linux-based computer, Raspberry Pi 3 [17], with a Hardware Attached on Top , Navio2. Navio2 shows in Fig 17 has already integrated many sensors, which include GPS receiver, barometer, accelerometers, gyroscopes and magnetometers. It extends Raspberry Pi's GPIO pins into PWM, UART ADC, and I2C interfaces for add-on sensors and communication devices. It accepts 14 channels of PWM output for motors and servos. Some of sensors we have integrated in this system includes RGB camera, telemetry radio, RTK-GPS, and laser range finder. Since Raspberry Pi is commonly available and highly scalable both in terms of the type of sensors and the number of sensor nodes [18], it makes the addition of new hardware and software module even more straightforward. The Raspberry Pi's GPU allows some basic image processing on-board possible. The total weight of this robot controller is only 75g. It can be easily mounted on most of the off-the-shelf robot platforms so that an off-the-shelf robot platform can be easily turned into an autonomous robot inside of the Cy-Eye system.

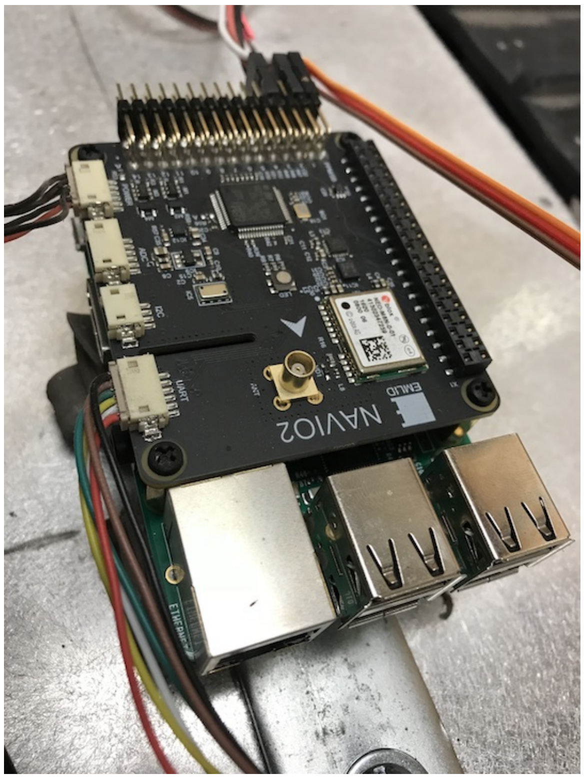


Figure 17. Micro-controller

2.5. Software Integration

As an experimental tool, the focus of this testbed is to enable researchers to test a wide variety of multi-vehicle related algorithms and implementations in a real environment. Our goal was to develop a system that allows users to easily integrate and control different robots. It should provide navigation function to researches allowing them concentrate on the core algorithm being experimented.

To allow any robot to navigate autonomously, we programmed the controller to make it to follow the way-points either given by human or generated by a computer through customizing an open source autopilot software, called ArduPilot [19]. It is an autopilot software capable of controlling almost any common vehicle system, from conventional airplanes, multi-rotors and helicopters, to cars and boats and even submarines. Almost any mobile machine can be easily transformed into a autonomous robot, by simply integrating the micro-controller mentioned in 2.4. Robot Operating System (ROS), a widely used software framework for robotics community, is also supported. It allows users to easily implement the existing algorithms or programs on our platform Cy-Eye. We customized the autopilot software to support peer-to-peer communication, which made decentralized experiments possible.

2.6. Middleware and Architecture

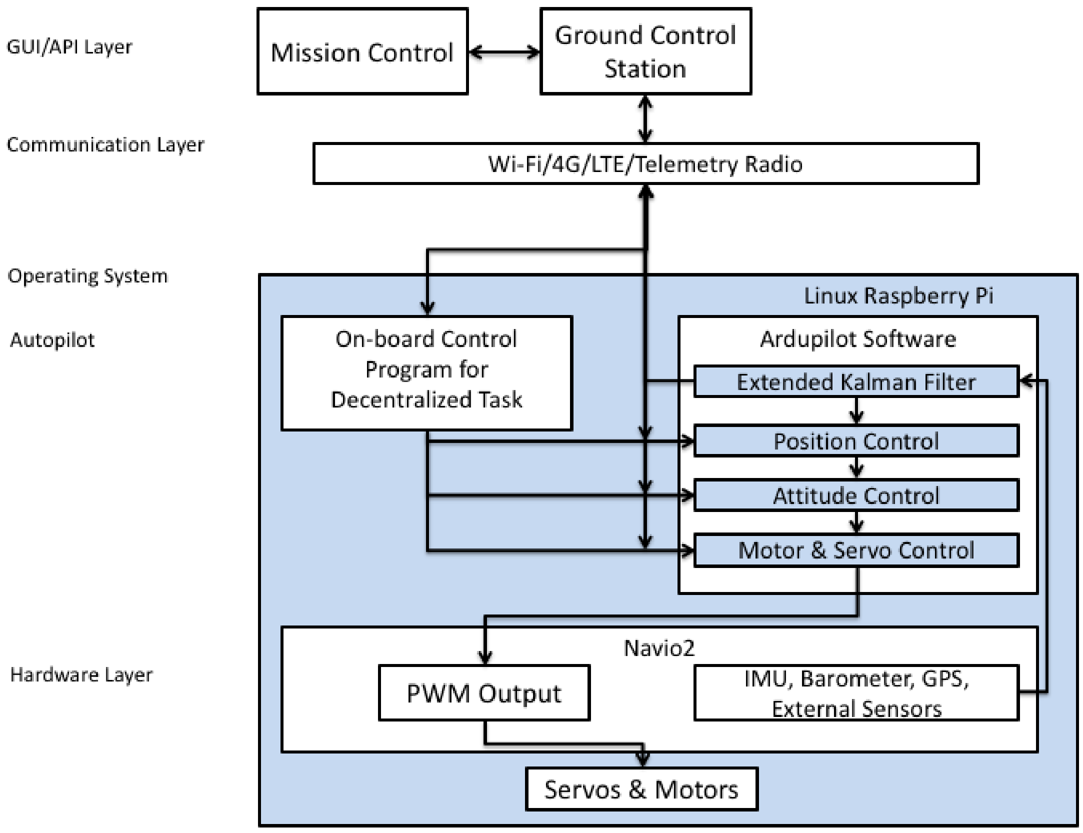
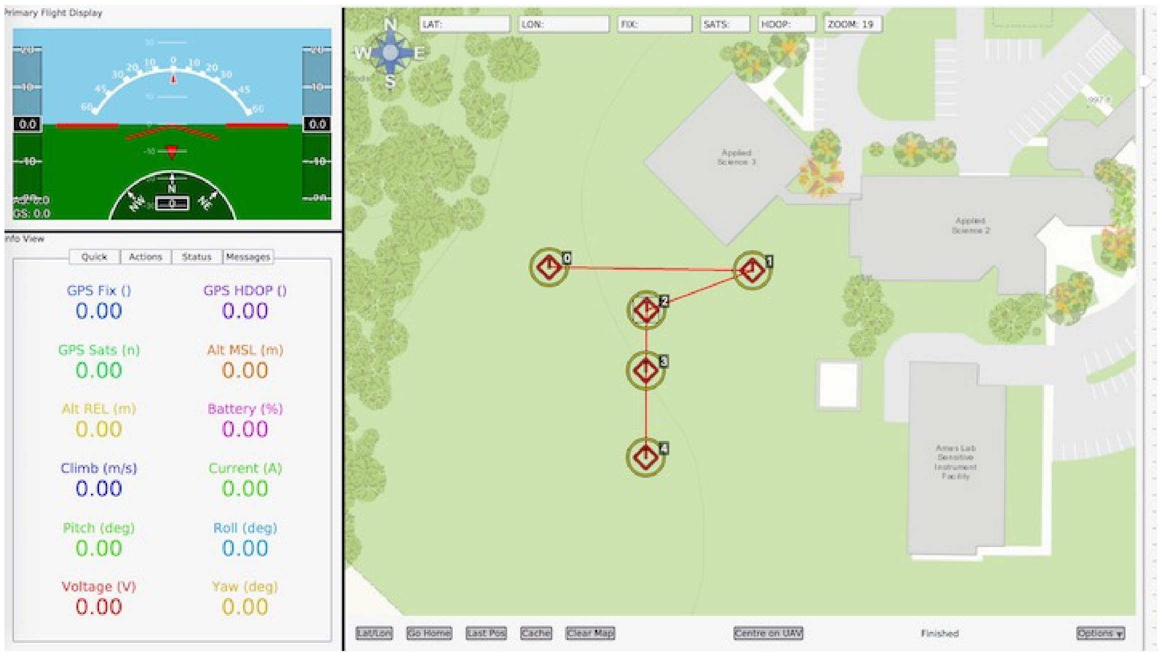


Figure 18. Robot architecture

Figure 16 shows the diagram of the components and setup of the communication system between robots and ground station. Figure 18 is a high-level view of the Cy-eye architecture. In the GUI layer, the ground control station is a graphic user interface which integrated with mission control can manage waypoints and data collection points.



GPS Fix () 0.00
GPS HDOP () 0.00
GPS Sats (n) 0.00
Alt MSL (m) 0.00
Alt REL (m) 0.00
Battery (%) 0.00
Climb (m/s) 0.00
Current (A) 0.00
Pitch (deg) 0.00
Roll (deg) 0.00
Voltage (V) 0.00
Yaw (deg) 0.00

Figure 19. Screen shot of APM Planner 2.0

The ground control station is compatible with multiple operating systems. The one we integrated in our testbed is an open-source software application, called APM Planner 2.0 [20] shown in Fig. 19, that can be run on Windows, Mac OS, and Linux. More importantly, it displays and logs real-time information on position, attitude, and sensors data for multiple vehicles. It can also be used to control robots in experiment. We customized the ground control station for uploading mission commands.

In addition to receiving flight control commands from computers, this architecture is also flexible. In the communication layer, telemetry radios made the decentralized experiments possible. Robots are able to communicate directly with others without a ground control station. Each vehicle is identified by a unique ID number. Decentralized algorithms can be implemented on autopilot by modifying the on-board control program.

The Micro-controller described in section 2.4 is capable of controlling almost any common vehicle system, from conventional airplanes, multi-rotors and helicopters, to cars and boats and even submarines. Almost any mobile machine can be easily transformed into a autonomous robot. Since the autopilot software is running on a Linux system, this made virtually infinite control range using either Wi-Fi or 4G/LTE through Internet. And 256 robots can be simultaneously deployed for experiments, which is more than enough for most of researchers. The UART, ADC, I2C, and four standard USB and 9 unused PWM interfaces even made the add-on sensors and communication devices possible.

2.7. Current Application

This section presents the current application that is implemented on the robot system. We present the informative sampling technique for robots in the field in order to minimize the total distance traveled by the robot for energy saving. Issues involving this waypoint generation process include as to how to pick the next visiting points move in the field, and how to model the environment based on data collected. For tackling such issues, the Gaussian Process (GP) and information-theoretic metric, i.e., Mutual Information (MI) are adopted correspondingly. We first introduce the GP for learning the environment model. Then we present an assignment and scheduling algorithm for the robots to reach their goal positions. The assignment algorithm minimizes the total distance traveled by the robot, and the scheduling algorithm minimizes the total time taken by the robots to reach their goal based on minimized paths.

2.7.1. Gaussian Process

A GP is a stochastic process (where random variables are collected from some set) such that for any finite subset obtained from the collection has a multivariate Gaussian distribution [21]. GP's prediction performance depends on the covariance matrix and the input training data set. The covariance matrix is obtained by defining and choosing the prior covariance function which identifies the relationship between two independent data points. The selection of prior covariance function is essentially important for constructing the covariance matrix. In other words, the covariance function defines the nearness or similarity of data points.

We formally describe how to use GP for prediction. Let $S = \{(x^i, y^i)\}_{i=1}^n, x^i \in \mathbb{R}^d, y^i \in \mathbb{R}$ be a training set, where y^i is an observation of Gaussian Process of $f(x) \sim \mathcal{GP}(0, k(x, x'))$ at location x^i. $k(x, x')$ is defined as the covariance function of a real GP of f. We assume a noisy version of measurement of y^i as follows

$$y^i = f(x^i) + \epsilon^i, i = 1, \dots, n \tag{10}$$

where ϵ^is are i.i.d White noise variables with variance σ_n^2.

Now, let $M = \{(x_*^i, y_*^i)\}_{i=1}^m$ be a set of i.i.d. test data set obtained from the same unknown distribution as S. For compactness, let

1. $X = [(x^1)^T; (x^2)^T; \dots; (x^n)^T] \in \mathbb{R}^{n\times d}$ be training input;
2. $X_* = [(x_*^1)^T; (x_*^2)^T; \dots; (x_*^m)^T] \in \mathbb{R}^{m\times d}$ be test input;
3. $\mathbf{f}_* = [f(x_*^1); f(x_*^2); \dots; f(x_*^m)] \in \mathbb{R}^m$ be function values corresponding to test input;
4. $\mathbf{y} = [y^1; y^2; \dots; y^n] \in \mathbb{R}^n$ be training output;

In a GP, the marginal distribution over any subset of inputs must have a joint multivariate Gaussian distribution. Therefore, in this context, for training and test data sets, we have[22]

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right) \tag{11}$$

where $K(\cdot,\cdot)$ represent covariance matrices, $K(X,X) \in \mathbb{R}^{n\times n}, K(X, X_*) \in \mathbb{R}^{n\times m}, K(X_*, X) \in \mathbb{R}^{m\times n}, K(X_*, X_*) \in \mathbb{R}^{m\times m}$, I is the $n \times n$ identity matrix.

The last relation follows from that the sums of independent Gaussian random variables is also Gaussian.

Applying the rules for conditioning Gaussians, the predictive equations can be obtained that

$$\mathbf{f}_* | \mathbf{y}, X, X_* \sim \mathcal{N}(\mu_*, \Sigma_*) \tag{12}$$

where, $\mu_* \triangleq K(X_*, X)(K(X,X) + \sigma_n^2 I)^{-1}\mathbf{y}, \Sigma_* \triangleq K(X_*, X_*) - K(X_*, X)(K(X,X) + \sigma_n^2 I)^{-1} K(X, X_*)$.

Based on the discussion above, typically the covariance functions have some free parameters. For this work we select squared-exponential covariance function as

$$k(p, q) = \sigma_f^2 exp(-\frac{1}{2}(p-q)^T A (p-q)) + \sigma_n^2 \delta_{pq} \tag{13}$$

where $A = diag(\gamma)^{-2}$. One can determine the level of correlation by the parameters γ for each dimension of p and q. σ_f^2 and σ_n^2 represent the variances of the prediction and noise, respectively. δ_{pq} is the Kronecker delta function that is 1 if $p = q$ while 0 otherwise.

2.7.2. Estimation of Hyperparameters

Based on the above discussion in the last section, we have hyperparameters $\sigma_f^2, \sigma_n^2, \gamma$ to be estimated. Let $\mathbf{w} \triangleq \{\sigma_f^2, \sigma_n^2, \gamma\}$. High accuracy of estimation of hyperparameters in the covariance functions can improve the model describing the underlying environment. In this context, we adopt the k-fold cross-validation (CV) [22] via maximum likelihood estimation to solve the problem. As the training data set is only available to be used for estimation, we discuss an extreme case of the k-fold cross-validation (CV) where $k = n$, the number of training points such that leave-one-out cross-validation (LOO-CV) is correspondingly used. The log-likelihood of LOO is as follows

$$L(X, \mathbf{y}, \mathbf{w}) \triangleq \sum_{i=1}^{n} \log p(y^i | X, \mathbf{y}_{-i}, \mathbf{w}) \tag{14}$$

where $\mathbf{y}_{-i}$ denotes all outputs in the training data set except the one with index i. The explicit form of $\log p(y^i | X, \mathbf{y}_{-i}, \mathbf{w})$ can be seen in [23]. For obtaining the optimal values of hyperparameters $\mathbf{w}$, we use the gradient descent method as follows

$$w_j(t+1) = w_j(t) - \alpha(t) g_j(w_j(t)) \tag{15}$$

where $\alpha(t)$ is the step size, $g_j(w_j(t))$ is the partial derivative of L at $w_j(t)$ for j-th hyperparameter set.

2.7.3. Informative Motions

Robot motions are determined by informative sampling locations, which directly affects the path planning for a robot. For tackling the challenge of future informative sampling locations, a field can be regarded as a discrete grid map in which each grid identifies a possible sampling location. According to the GP, the mean and variance information associated with the measurement at each grid can be predicted accordingly. However, the selection of sampling location is another issue such that an information-theoretic metric for quantifying the information between the sampled locations and unsampled locations is Mutual Information (MI). We formally summarize the MI between two data sets, M and N as follows:

$$I(M; N) = I(N; M) = H(M) - H(M|N) \tag{16}$$

where $H(M, N), H(M|N)$ are entropy and conditional entropy, respectively which can be calculated by

$$H(M) = \frac{1}{2} \log((2\pi e)^{\mathcal{B}} |\Sigma_{MM}|) \tag{17a}$$

$$H(M|N) = \frac{1}{2} \log((2\pi e)^{\mathcal{B}} |\Sigma_{M|N}|) \tag{17b}$$

where $\mathcal{B}$ is the size of M. One can calculate the covariance matrix Σ_{MM} and $\Sigma_{M|N}$ via the posterior GP in Eq. 12.

In light of the MI metric, to at most level obtain the information of true model, it is equivalent to finding new sampling points in the unsampled part that can maximize the MI between sampled locations and unsampled part of the map. We denote by X the entire space (including all grids) and by D a subset of X with a certain size $|D| = n$. Therefore, the set of D includes the most mutual information required for generating the best predictive model. Thus, the following optimization problem is obtained

$$\max I(D; X \backslash D) \tag{18a}$$

$$\text{s.t. } D \in \mathcal{D} \tag{18b}$$

where $\mathcal{D}$ represents all possible combinatorial sets, each of which is of size n.

To solve the optimization problem described above, one can adopt dynamic programming scheme [23] to get the optimal subset D^*. Algorithm 1 gives the set of points that need to be measured for the next step of the distribution estimation. It should be noted that D^* only contains the sampling points that can give the most information for the purpose of best model prediction performance, but does not give us any information about the final path planning. The algorithm for path planning in a farm field is described in the next section.

Algorithm 1 Informative Motions and Nonparametric Learning

```
 1: **Input**: $X$, $\mathbf{y}$, $k$ covariance function, $X_*$, $m = |X_*|$
 2: **Output**: Optimal hyperparameters $\mathbf{w}^*$
 3: Use $X$,$\mathbf{y}$,$k$ to estimate the hyperparameters in Eq. 14
 4: **for** $z = 1 : m$ **do**
 5:     Divide $X_*$ into two parts and customize one part of $X_*^z$ and $X$ into $N$ while another part of $X_*$ into $M$
 6:     Use posterior GP Eq. 12 to calculate the covariance matrices in Eq. 17
 7: **end for**
 8: Solve the Eq. 18
 9: Include the new data point into $X$ to update GP
10: Repeat Step 3
```

2.7.4. Path Planning

At each iteration, the assignment of the robots to the goal position needs to be computed by the ground station, and communicated to the robots through the Wi-Fi network. In this section, we present an assignment and scheduling algorithm for the robots to reach their goal positions. The assignment algorithm minimizes the total distance traveled by the robot, and the scheduling algorithm minimizes the total time taken by the robots to reach their goal based on minimized paths.

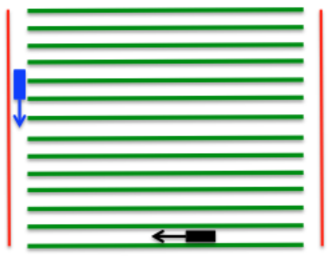


Figure 20. Two robots moving in the field. The Black robot moving toward left in between two rows. The blue robot is moving downward on the left headland.

Let us consider a field containing m rows and n columns. The field can be represented as a graph. Each node in the graph is a location of a canopy. We allow only one robot on a node at the same time to avoid collision. Since the distance between any two adjacent columns in the field is not enough for a robot, the robots can only travel in between the rows, which means that they can only switch their rows on the two headland, illustrated in Fig. 20, at the end of each row. The nodes in the graph are not adjacent to the nodes above and below them, except the ones on the left and right most columns. The two headlands are called left path and right path. We have k robots and k goal points in the field, $k \leq m \times n$. We use a set $\mathbf{A}$ to represent the robots and a set $\mathbf{B}$ to represent the goal points.

2.7.5. Robot-Target Assignment

For each robot $r_{ij} \in \mathbf{A}$ such that $i \in m, j \in n$, in general, Breadth-first-search algorithm can be used to calculate the distance between the robot r_{ij} and any target $t_{uv} \in \mathbf{B}$ such that $u \in m, v \in n$. However, in the row-crop field, a robot can either run towards the left or right, so it has two possible routes to reach to each target. We select the shorter of the two paths. We use the distances to create a cost matrix for each robots and goal points by using the following algorithm.

Algorithm 2 Compute cost matrix given robots and goal points position

```
Input: Two matrices. A and B are robots and goal points location sets
Output: Cost matrix C
1: function COSTMATRIX(A, B)
2:     for each $r_{ij}$ do
3:         for each $t_{uv}$ do
4:             $\mathbf{C}_{ij} = |i - u| + \min(j + v, 2n - (j + v))$
5:         end for
6:     end for
           return C
7: end function
```

With the cost matrix **C**, we can compute this assignment problem with a well known combinatorial optimization algorithm named Hungarian algorithm [24] in polynomial running time. The total sum of the paths between each robot and its target is minimized.

2.7.6. Tasks Scheduling

The planning algorithm has to ensure that two robots do not collide while they travel on their respective paths. Before we propose our scheduling algorithm, we present the following properties associated with the output of the Hungarian algorithm.

Lemma 1. *In the output of the Hungarian algorithm, two robots do not share any segment of their paths in which they are moving in opposite directions.*

Proof. If two robots, r_1 on the left of r_2, can achieve their goal points by moving in opposite directions and intersect at any moment, r_1's path must overlap with r_2's path. The total length of these two paths can be reduced by switching their goal points thereby reducing the total path length by twice the length of the overalapping section. However, this is a contradiction since we assumed that the initial paths minimized the sum of lengths of each path. □

Lemma 2. *Consider a row with m robots and n goals such that m robots are only allowed to use the left path.*

1. If $m > n$, then there is a unique reassignment that allows $m - n$ robots to leave the row, and the remaining n robots to reach their goals without any collision.
2. If $n > m$, then there is a unique reassignment that allows $n - m$ robots to enter the row, and the remaining m robots in the row to reach the m goal points without any collision.

The reassignments do not change the sum of path lengths, and do not increase the total time required to complete the task.

Proof. 1. Let us consider a labeling of robots $\{r_1, \ldots, r_m\}$ from left to right with the first robot on the left numbered as 1. Goals are labeled from left to right $\{g_1, \ldots, g_n\}$ in a similar manner. Goal g_n is assigned to robot r_m, g_{n-1} is assigned to r_{m-1}, g_{n-2} is assigned to r_{m-2}, and so on, until g_1 is assigned to r_{m-n+1}. The leftover $m - n$ robots can leave the row without any collision with other robots. This reassignment does not change the sum of the path lengths. Between any two robots being reassigned, the original travel distance for robot r_i and r_j are d_i and d_j. The distance between r_i and r_j is d. After reassignment, the traveling distance for r_i is $d_j - d$, and the traveling distance of R_j is $d + d_i$. The total traveling distance is still $d_i + d_j$. Since the total travelling distance remains unchanged, the total time required to complete the task will not increase either.

2. Consider the same labeling as in the previous case. Goal g_n is assigned to robot r_m, g_{n-1} is assigned to r_{m-1}, g_{n-2} is assigned to r_{m-2}, and so on, until g_{n-m+1} is assigned to r_1. The leftover $n-m$ goals can be taken care off by $n-m$ robots entering from left path without collision. We sort these robots with increasing order by traveling distance to this row. Then we assign the leftover $n-m$ goals from right to left with the sorted robots. Similarly, this reassignment does not change the sum of path lengths, and does not increase the total time required to complete the task.

□

After reassignment, collision will never happen on rows. For the collision on left and right paths, we propose a strategy that allows the robot that has higher priority to move on to the node that is going to cause collision. The robot that is going to travel more distance has higher priority. Before each robot starts moving towards its next waypoint, if the next waypoint of robot a is same as the next waypoint of robot b, we let the robot which has lower priority to wait until its next waypoint is not conflict with other's next waypoint. This strategy provides us a collision free scheduling on left and right paths.

By using these strategies, the total traveling distance for all the robots should still be minimized. The total time taken to complete the entire task is the maximum of each individual robot's task plus the total waiting time caused by collision avoidance.

2.7.7. Time Complexity

The path planning algorithm for minimizing the sum of total traveling distance over all robots is $O(n^3)$, where n is the number of robots. The COSTMATRIX in Algorithm 2 will take $O(n^2)$ time for constructing the cost matrix. Given the cost matrix, we compute the optimal assignment by Hungarian algorithm in $O(n^3)$ [24] steps. Reassignment process in Lemma 1 takes $O(n \log n)$ since we need to sort all the robots and goal points. Therefore, the entire path planning algorithm is $O(n^3)$.

3. Result

3.1. Experimental setup

The experimental plot is 76cm wide between two connected rows and it has 32 by 39 soybean plants. We use a TP-Link AC1750 Wi-Fi router to serve as the base station to provide the communication between robots and control station. It provides network coverage with a 12v power supply. A RTK-GPS base station, Reach RS, is placed next to the Wi-Fi router to set up our RTK-GPS base station and provide centimeter accuracy. It streams GPS corrections data to individual robot over the network via TCP.

The initial positions of the robots are randomly selected. The set of points that need to be visited by the robots are generated by Algorithm 1. Paths and waypoints for each robot are output of the path planning algorithm, described in 2.7.4. Since the waypoints for each robot are generated off line from Mission control 18, a human operator with a laptop only needs to start or stop data collection by toggling a button on graphic user interface. Each robot follows the given waypoints and collects data. Fig 21 shows two rovers collecting images in the experimental field at ISU and sample image collected by the rovers.

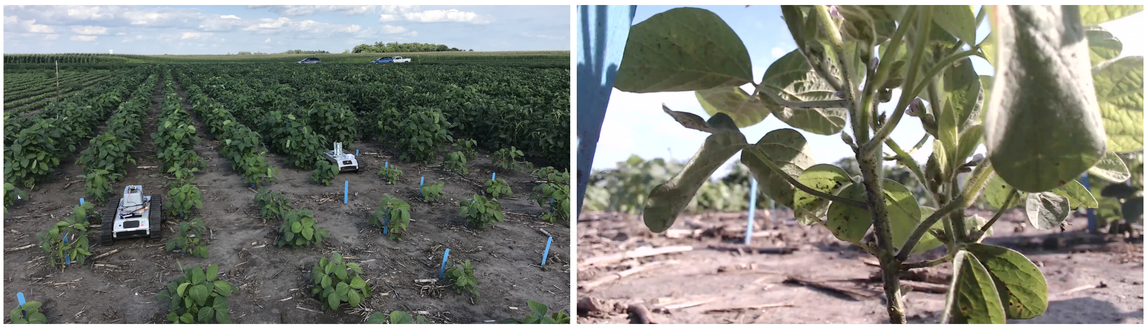


Figure 21. Rovers deployed in the soybean research field at ISU and sideways image of soybean plant.

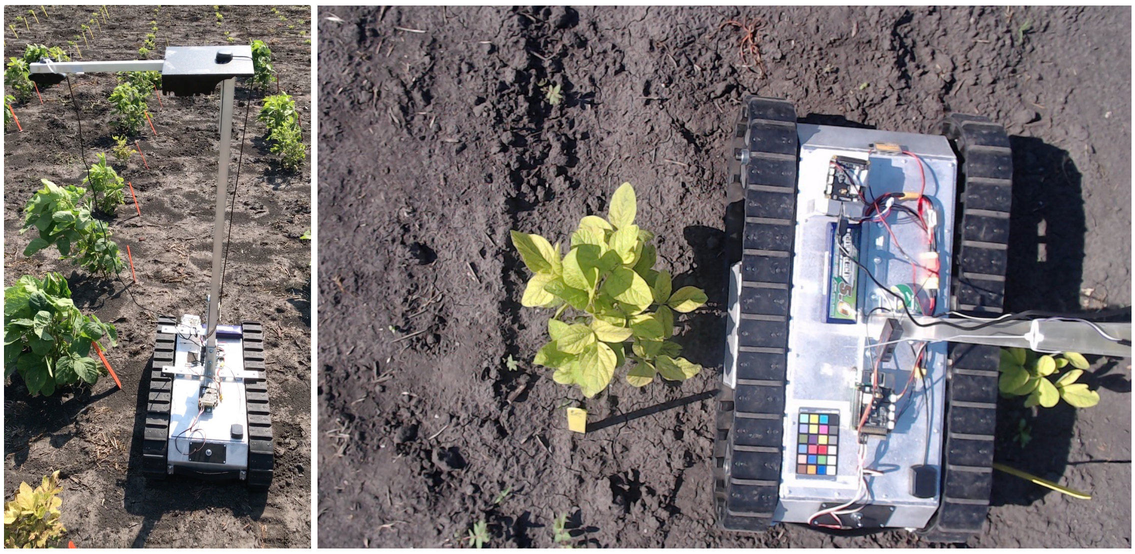


Figure 22. Image of soybean canopy from the overhead camera mounted on the rover.

3.2. Simulation

In this section, we presents the result for path planning algorithm. In our simulations, we consider a planar environment as our field, and our goal is to estimate the distribution of a scalar field based on adaptive sampling with a group of mobile robots equipped with sensors. The environment and the scalar function can be a agricultural plant area and any phenomena which is distributed spatially, respectively. We consider the scalar function as the distribution of an IDC value in a farm field.

In order to show the effectiveness of the proposed algorithm, we use synthesized data, generated based on 2D Gaussian distribution. We assume that the intensity of the scalar function is in proportion to the probability density function of a bimodal Gaussian distribution with the following mean and covariance values.

$$\mu_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \sigma_1 = \begin{bmatrix} 0.2 & .3 \\ 0.3 & 1 \end{bmatrix} \mu_2 = \begin{bmatrix} 1 \\ 0.2 \end{bmatrix} \sigma_2 = \begin{bmatrix} 0.4 & .2 \\ 0.2 & .1 \end{bmatrix}.$$

We discretize the field into a grid map, and we associate each grid with a target value which describes the intensity of disease in the plant. Fig 23 shows the distribution of infected plants in the field. We pick 10% of grid points randomly as the training set, and we collect the new data points based on the aforementioned algorithm. We consider 3 robots(i.e. $n = 3$), initially deployed in 3 different positions in the field. At each step, three sample points are picked, based on Algorithm 1. Each robot is associated to a sample point, based on total distance traveled by the robots, to reach the sample points from the current positions. The evolution of estimation during sampling steps are shown in Fig 23. We

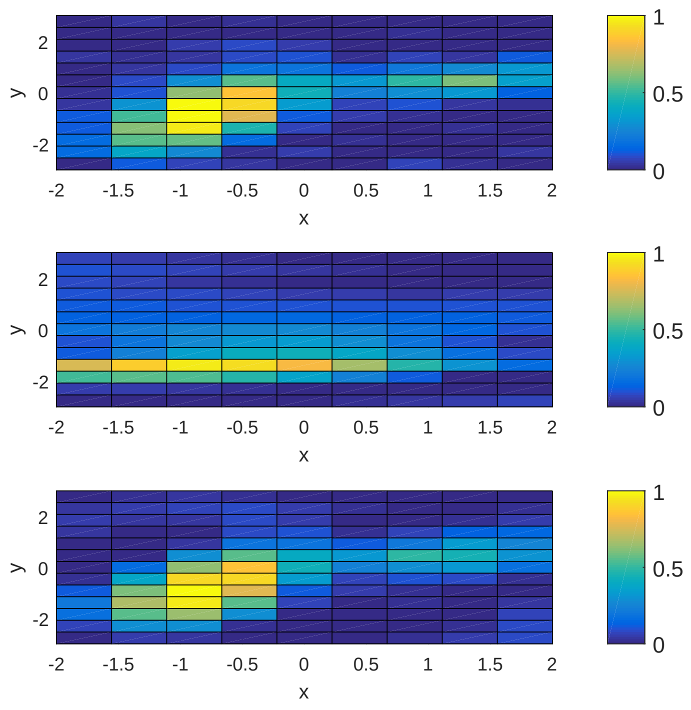


Figure 23. The plot at the top shows the heat-map of a scalar function between 0 and 1, in a planar field. The middle plot shows the initial distribution of the estimation. The bottom plot is the final distribution estimation after informative sampling.

compared the error computed from the difference of mean value of the estimated distribution with the truth data. Total number of sampling operation is 30, and Fig 24 shows the histogram of error for this 30 cases along collecting data in different steps. It reveals an average decrease in the estimation error with increase in the number of iterations.

In the second simulation, we examine the real data, collected from a soybean field. Fig 22 shows the sample data we collected from the real soybean field by using a rover. Fig 25 shows the real distribution of IDC value in the field. The intensity of yellowness in each plant is quantified between 0 and 5 as IDC value. We apply our algorithm on a field, which is a 8 by 39 grid points. We pick 30 points randomly as a training set. In order to get a new sample point, we follow the Algorithm 1, and it leads to the informative path.

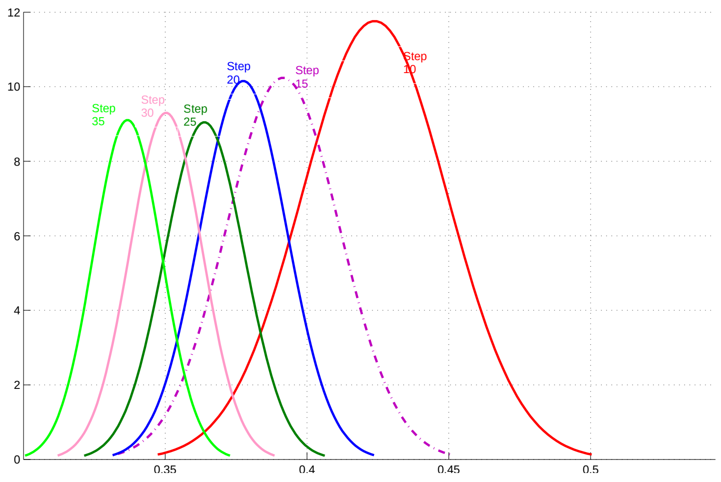


Figure 24. Histogram of error.

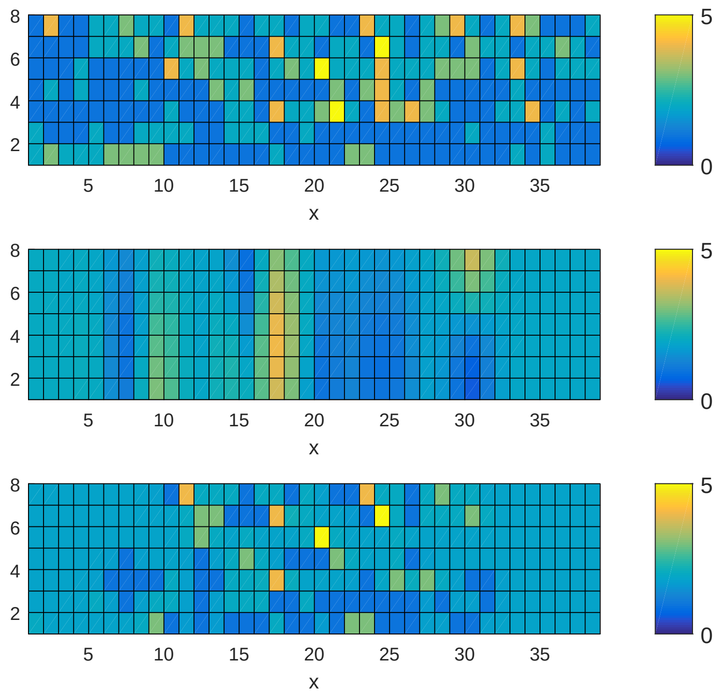


Figure 25. The plot on the top shows the heat-map of IDC values, scaled between 0 and 5, in a soybean field. The plot in the middle shows the initial distribution estimation. The bottom figure is the final distribution estimation after informative sampling.

Table 2. Output performance summary

Time per picture (seconds)	6.00
Time to raise and lower system per 2 plants (s/drop)	4.29
Plants per drop (plants/drop)	2.00
Diameter of a plant (feet)	2.50
plants per row (plants)	100.00
Max camera speed (ft/s)	0.42
Maximum Vehicle speed (ft/s)	4.40
time to scan 100 Plants (minutes)	14.52
Plants scanned in an hour	413.269
Plants in 8 hour day	3306.15

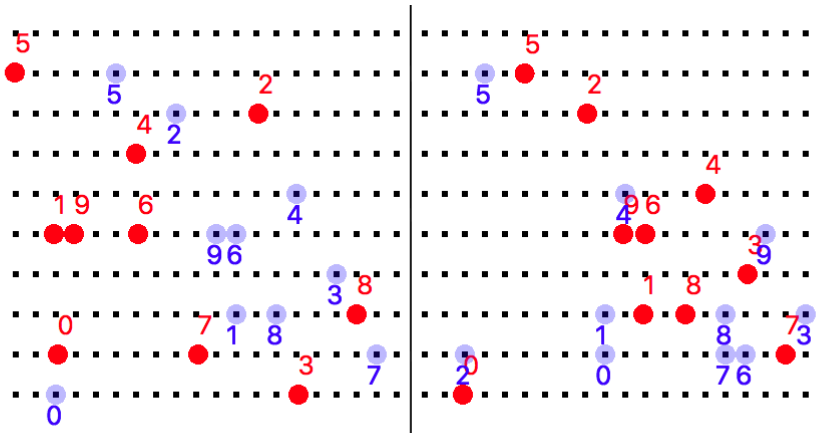


Figure 26. Two consecutive path planning algorithm simulation. The image on the left is when $t = i$, and the image on the right is when $t = i + 1$. Red dots indicate the position of the robots. Blue dots represent the location of the sample points.

Figure 26 shows the path planning and assignment for two consecutive iterations. On the left part of Figure 26 is the first round of simulation, and the right part is the second round. It shows the assignment for robots and goals. Blue dots indicate goal points and red dots represent robots. Index of robots is on top of red dots, and index of goal points is below blue dots. The robot will visit the goal point that has same index number. A video accompanying the submission shows the paths of the robots and the IDC estimates as the iterations progress for several simulation scenarios.

3.3. MARS performance

In this section we present a detailed summary of MARS as a vehicle and its performance and resource usage in the worst case scenario. Table 2 while describing all the subtasks demanded from the robot, also sets benchmark metrics for doing such tasks. Table 3 sets benchmarks for power usage across various components of the system.

4. Conclusions

In this paper, we presented the design, construction and deployment of a lightweight distributed multi-robot system for row crop phenotypic data collection. Next, we presented an entropy-based informative path planning technique for the robots to navigate in the field. This involves a Gaussian process model for the robots to sample the field to obtain the goal positions of each robot. Next, we proposed a collision-free path planning algorithms for the multiple robots to reach their goal positions in a row crop field. Finally, we presented a deployment scenario for the robots in the field.

Table 3. Resource usage summary

	Quantity	Watts	Total Watts	Hours	Watt hours	Volts	Amps
lap top computer	1	60	60	12	720	19	3.157895
lights	2	70	140	12	1680	15	9.333333
stepper motors	3	6.666667	20	12	240	n/a	3.15
arduino	1	0.35	0.35	12	4.2	7	0.05
Pika XC	1	2.5	2.5	12	30	8	0.3125
kinect Camera	1	3.75	3.75	12	45	5	0.75
total power required					3339		
total amps							16.00373
Inverter efficiency	0.8						

As a part of our future work, we plan to modify the rover design to make it suitable for a wider range of phenotyping activities. Another direction of future work is to add aerial vehicles into the multi-robot system to acquire multi-resolution phenotyping capabilities.

Author Contributions: Data curation, T.G. and H.S.; Formal analysis, T.G., H.S. and H.E.; Methodology, T.G., H.S. and J.Z.; Software, T.G. and A.L.; Supervision, A.S., B.G., A.S., S.S. and S.B.; Writing—Original Draft Preparation, T.G.; Writing—Review & Editing, S.B.

Acknowledgments: The authors wish to thank Johnathon M Shook for help with the field experiments and Koushik Nagasubramanian for hyperspectral images. The authors would also like to thank the undergraduate students involved in ME Capstone projects for conceptual design of MARS.

References

1. Xavier, A.; Hall, B.; Hearst, A.A.; Cherkauer, K.A.; Rainey, K.M. Genetic Architecture of Phenomic-Enabled Canopy Coverage in Glycine max. *Genetics* **2017**, *206*, 1081–1089, [http://www.genetics.org/content/206/2/1081.full.pdf]. doi:10.1534/genetics.116.198713.
2. Naik, H.S.; Zhang, J.; Lofquist, A.; Assefa, T.; Sarkar, S.; Ackerman, D.; Singh, A.; Singh, A.K.; Ganapathysubramanian, B. A real-time phenotyping framework using machine learning for plant stress severity rating in soybean. *Plant methods* **2017**, *13*, 23.
3. Bai, G.; Ge, Y.; Hussain, W.; Baenziger, P.S.; Graef, G. A multi-sensor system for high throughput field phenotyping in soybean and wheat breeding. *Computers and Electronics in Agriculture* **2016**, *128*, 181 – 192. doi:https://doi.org/10.1016/j.compag.2016.08.021.
4. Shi, Y.; Thomasson, J.A.; Murray, S.C.; Pugh, N.A.; Rooney, W.L.; Shafian, S.; Rajan, N.; Rouze, G.; Morgan, C.L.; Neely, H.L.; others. Unmanned aerial vehicles for high-throughput phenotyping and agronomic research. *PloS one* **2016**, *11*, e0159781.
5. Tattaris, M.; Reynolds, M.P.; Chapman, S.C. A Direct Comparison of Remote Sensing Approaches for High-Throughput Phenotyping in Plant Breeding. *Frontiers in Plant Science* **2016**, *7*, 1131. doi:10.3389/fpls.2016.01131.
6. Ruckelshausen, A.; Biber, P.; Dorna, M.; Gremmes, H.; Klose, R.; Linz, A.; Rahe, F.; Resch, R.; Thiel, M.; Trautz, D.; others. BoniRob: an autonomous field robot platform for individual plant phenotyping. *Precision agriculture* **2009**, *9*, 1.
7. Bawden, O.; Ball, D.; Kulk, J.; Perez, T.; Russell, R. A lightweight, modular robotic vehicle for the sustainable intensification of agriculture **2014**.
8. Grimstad, L.; From, P.J. The Thorvald II agricultural robotic system. *Robotics* **2017**, *6*, 24.
9. Fernandez, M.G.S.; Bao, Y.; Tang, L.; Schnable, P.S. A high-throughput, field-based phenotyping technology for tall biomass crops. *Plant physiology* **2017**, pp. pp–00707.
10. Arbo, M.H.; Utstumo, T.; Brekke, E.F.; Gravdahl, J.T. Unscented Multi-Point Smoother for Fusion of Delayed Displacement Measurements: Application to Agricultural Robots **2017**.
11. Batey, T. Soil compaction and soil management–a review. *Soil use and management* **2009**, *25*, 335–345.
12. Nawaz, M.F.; Bourrie, G.; Trolard, F. Soil compaction impact and modelling. A review. *Agronomy for sustainable development* **2013**, *33*, 291–309.

13. Vellidis, G.; Ortiz, B.; Beasley, J.; Hill, R.; Henry, H.; Brannen, H. Reducing digging losses by using automated steering to plant and invert peanuts. *Agronomy* **2014**, *4*, 337–348.
14. Peterson, D. Development of a harvest aid for narrow-inclined-trellised tree-fruit canopies. *Applied engineering in agriculture* **2005**, *21*, 803–806.
15. Bao, Y.; Nakami, A.D.; Tang, L. Development of a field robotic phenotyping system for sorghum biomass yield component traits characterization. 2014 Montreal, Quebec Canada July 13–July 16, 2014. American Society of Agricultural and Biological Engineers, 2014, p. 1.
16. Moret, E.N. Dynamic modeling and control of a car-like robot. PhD thesis, Virginia Tech, 2003.
17. Pi, R. Raspberry Pi Model B, 2015.
18. Ferdoush, S.; Li, X. Wireless sensor network system design using Raspberry Pi and Arduino for environmental monitoring applications. *Procedia Computer Science* **2014**, *34*, 103–110.
19. Anderson, C. Ardupilot project, 2016.
20. Team, A.D. APM Planner 2. Open Source Project, 2016.
21. Williams, C.K.; Rasmussen, C.E. Gaussian processes for machine learning. *the MIT Press* **2006**, *2*, 4.
22. Rasmussen, C.E. Gaussian processes for machine learning **2006**.
23. Ma, K.C.; Liu, L.; Sukhatme, G.S. Informative planning and online learning with sparse gaussian processes. Robotics and Automation (ICRA), 2017 IEEE International Conference on. IEEE, 2017, pp. 4292–4298.
24. Kuhn, H.W. The Hungarian method for the assignment problem. *Naval research logistics quarterly* **1955**, *2*, 83–97.