HANDHELD AND ROBOTIC PHENOTYPER DESIGNS

by

YONG WEI

B.Eng., Shanxi Agricultural University, P.R. China, 1996
M. Eng., Shanxi Agricultural University, P.R. China, 2001

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Biological and Agricultural Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2017

Abstract

One of precision agriculture researches currently focuses on the relationship between plant phenotype, genotype, and ambient environment, including critical investigations of a multi-sensor-integrated phenotyping platform and data mining technology for big data. This study examined the designs of two phenotyping platforms and developed machine vision (MV) technology to estimate wheat growth status and count wheat head.

The GreenSeeker, an infrared thermometer (IRT), a web camera, and a global positioning system (GPS) receiver were integrated into one handheld phenotyping platform, named as Phenocorn. The Phenocorn allowed simultaneous collection of the normalized difference vegetative index (NDVI) and canopy temperature (CT) with precise assignment of all measurements to plot location by GPS data points. The Phenocorn was tested using a field trial of 10 historical and current elite wheat (*Triticum aestivium*) breeding lines at the International Maize and Wheat Improvement Center (CIMMYT) in Ciudad Obregon, Mexico, during the 2013 and 2014 growing seasons. Results showed that the NDVI data, PVC (percent vegetation coverage) data, and temperature data obtained by the handheld phenocorn could availably reflect the wheat growing status in the field, and the handheld phenocorn could be used as an instrument to do plant phenotyping information collection.

This study also used the modular design method to design the mechanical structures of a robot-based phenotyping platform, named as Phenorobot. Its control system was based on a Controller Area Network (CAN bus). The basic function performances such as steering function, lifter load, and movement features were tested in the laboratory. Proposed design indicators were achieved, demonstrating its potential utilization for field experiments.

Image acquisition is one of the main data collection methods for plant phenotyping research. The method for extracting plant phenotyping traits based on MV was explored in this research. Experiments for detecting the wheat development based on the images taken in the field were designed and carried out from March to June 2015, and a method based on color analysis to estimate percent vegetation coverage (PVC) of wheat was developed. A wheat growth model based on the PVC was used for the wheat growth status analysis. In addition, a wheat head counting method was developed and divided into three steps: wheat head image segmentation, leaf debris elimination, and wheat head counting. This paper proposes the first wheat head counting model (WCM) based on the pixels group measurement of wheat heads. Compared to the Joint Points Counting (JPC) method (Liu et al., 2014) and the Wheatear Shape Index (WSI) method (Frédéric et al., 2012), the WCM more accurately counted wheat heads from images taken in the experiments.

HANDHELD AND ROBOTIC PHENOTYPER DESIGNS

by

YONG WEI

B.Eng., Shanxi Agricultural University, P.R. China, 1996
M. Eng., Shanxi Agricultural University, P.R. China, 2001

A DISSERTATION

submitted in partial fulfillment of the requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Biological and Agricultural Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2017

Approved by:

Major Professor
Naiqian Zhang

Copyright

YONG WEI

2017

Abstract

One of precision agriculture researches currently focuses on the relationship between plant phenotype, genotype, and ambient environment, including critical investigations of a multi-sensor-integrated phenotyping platform and data mining technology for big data. This study examined the designs of two phenotyping platforms and developed machine vision (MV) technology to estimate wheat growth status and count wheat heads.

The GreenSeeker, an infrared thermometer (IRT), and a global positioning system (GPS) receiver were integrated into one handheld phenotyping platform, named as Phenocorn. The Phenocorn allowed simultaneous collection of the normalized difference vegetative index (NDVI) and canopy temperature (CT) with precise assignment of all measurements to plot location by GPS data points. The Phenocorn was tested using a field trial of 10 historical and current elite wheat (*Triticum aestivium*) breeding lines at the International Maize and Wheat Improvement Center (CIMMYT) in Ciudad Obregon, Mexico, during the 2013 and 2014 growing cycles. Results showed that the NDVI data, PVC (percent vegetation coverage) data, and temperature data obtained by the handheld phenocorn could availably reflect the wheat growing status in the field, and the handheld phenocorn could be used as an instrument to do plant phenotyping information collection.

This study also used the modular design method to design the mechanical structures of a robot-based phenotyping platform, named as robotic phenotyper. Its control system was based on a Controller Area Network (CAN bus). The basic function performances such as steering function, lifter load, and movement features were tested in the laboratory. Proposed design indicators were achieved, demonstrating its potential utilization for field experiments.

Image acquisition is one of the main data collection methods for plant phenotyping research. The method for extracting plant phenotyping traits based on MV was explored in this research. Experiments for detecting the wheat development based on the images taken in the field were designed and carried out from March to June 2015, and a method based on color analysis to estimate percent vegetation coverage (PVC) of wheat was developed. A wheat growth model based on the PVC was used for the wheat growth status analysis. In addition, a wheat head counting method was developed and divided into three steps: wheat head image segmentation, leaf debris elimination, and wheat head counting. This paper proposes the first wheat head counting model (WCM) based on the pixels group measurement of wheat heads. Compared to the Joint Points Counting (JPC) method (Liu et al., 2014) and the Wheatear Shape Index (WSI) method (Frédéric et al., 2012), the WCM more accurately counted wheat heads from images taken in the experiments.

Table of Contents

List of Figures

List of Tables

Acknowledgements

First of all, I want to express my heartfelt thanks to Dr. Naiqian Zhang, my supervisor, for all the time and energy he has put into training me as a student of agricultural engineering at Kansas State University. I benefited much from his critical thinking, extensive knowledge, scholarly expertise, and kind encouragement before I entered the doctoral program. However, being one of his many students in its real sense has provided me with a precious opportunity to be exposed to his unforgettable tutelage of various kinds, academic and non-academic.

My particular thanks should go to my supervisory committee: Dr. John Hatcliff, Dr. Paul Armstrong, Dr. Ajay Sharda, and Dr. Ignacio Ciampitti for their scholarly advice, meticulous style, and generous help while researching my project at Kansas State University. Thank Dr. Paul Armstrong again for his financial support to the project of designing the robotic phenotyper. I am also indebted to Dr. Jesse Poland for his financial assistance to the project of handheld phenotyper. I thank Dr. Joseph Harner, Head of the Biological and Agricultural Engineering Department, for his support and encouragement.

Moreover, I want to thank Mr. Jonathan M. Zeller for his very patient and time-consuming assistance while manufacturing the robotic phenotyper, to Dr. Jared Crain for his cooperation in the research and development of the handheld phenotyper and his hard work for testing the handheld phenotyper at the International Maize and Wheat Improvement Center (CIMMYT) at Ciudad Obregon, Mexicoand. I thank Dr. Xu Wang, and Mr. Jared Barker Ⅲ for their assistances and suggestions while studying at Kansas State University.I am heartily thankful to the department staff, Ms. Barbara Moore, Mr. Randy Erickson, and Ms. Cindy Casper for helping me on all aspects of my study and research in the BAE department. I also offer my regards and blessings to all of those who supported me in any respect during my life in Kansas State University.

Finally yet importantly, I want to express my sincere thanks to my family, and especially, to my parents and my wife Ms. Xiaoying Guo for the support, kindness, encouragement, and understanding they have given to me for many years.

Chapter 1 - INTRODUCTION AND OBJECTIVES

Global demand for crop production is rapidly increasing with the growth of population, diet shifts, and biofuel consumption. It is estimated that global crop production will have to achieve a 100-110% increase to meet the projected needs by 2050 (Tilman et al., 2011). That is, global crop production should keep growing at a rate of 2.4% per year, which is a tremendous challenge. According to statistical data, the fields of top four crops - maize, rice, wheat, and soybean - are increasing at a rate of 1.6%, 1.0%, 0.9%, and 1.3% per year, respectively, all of which are lower than the required production growth rate of 2.4%. Moreover, the increasing rate of total global crop production was only 28% between 1985 and 2005, among which 20% increase was due to the rise of crop yields per hectare (Ray et al., 2012). To meet the global food demands by 2050 while reducing the environmental impacts of agricultural expansion, the preferred solution is to boost crop yields rather than using more land (Ray et al., 2012; Tilman et al., 2011). The most effective solution to boost crop yields is to improve the breeding efficiency (Araus et al., 2014).

Phenomics is an area of biology that measures the physical and biochemical traits of organisms. It is a discipline to study the genotype-phenotype mapping, identify the genetic basis of complex traits, and explain the causation at the phenotypic level (Houle et al., 2010; Furbank and Tester, 2011). Phenotype and genotype are two fundamental concepts related to phenomics. The definition of an organism's genotype is the set of genes that the organism carries, and the organism's phenotype is all of its observable traits, which are influenced by its genotype and the environment. Phenotyping is represented by a set of tools and methodologies of studying phenomics (Fiorani and Schurr, 2013).

Accurately assessing and recording phenotypic data are essential for plant breeders, agronomists, plant physiologists, and geneticists. With the rapid advances in high-throughput genotyping technology, scientists have a tremendous opportunity to generate high-density genomic data for crop improvement (Morrell et al., 2011). Successfully understanding genotypic data requires large amounts of phenotypic data to explore the genotype-phenotypic relationship (Campos et al., 2004; Cobb et al., 2013). However, phenotyping has been lagging behind genomic capabilities due to the lack of high-throughput technologies to access integrated phenotypes (Houle et al., 2010; Araus and Cairns, 2014). Therefore, phenotyping is often described as the current bottleneck in research (Houle et al., 2010; White et al., 2012; Cobb et al., 2013, Myles et al., 2009, White et al., 2012). Development of accurate, high-throughput phenotyping technology would provide tremendous potentials for discovering the complex interactions among genotype, phenotype, and environment (Breccia and Nestares, 2014).

There are two ways of characterizing phenotypes - extensive phenotyping and intensive phenotyping. Extensive phenotyping is defined as sampling a wide variety of phenotypes, and intensive phenotyping is defined as repeatedly sampling one phenotype through time. Both methods are important for phenomics (Houle et al., 2010). Multi-sensor, integrated phenotyping platforms and big data mining technologies are crucial research issues related to both extensive and intensive phenotyping.

1.1 Handheld phenotypers

Phenotyping platforms can be ground-based or aerial-based. Ground-based phenotypers can be stationary, vehicle-based, or handheld. One advantage of any phenotyping platform, field or greenhouse, is the reliability of automated data collection. Recording data directly from sensors removes human errors from the system by reducing the number of times that data is

manually entered or transcribed from data collection to completed analysis. In addition, each step may be performed by different individuals, further increasing the possibilities that errors or "blunders" are introduced into the analysis and bias the results (Taylor, 1987).

In recent years, many high-throughput phenotyping (HTP) platforms have been developed. These included tractor-mounted platforms, platforms operated on cranes, towers similar to sports stadium cameras, and aerial vehicles. While all these platforms have a potential for use, there are also limitations associated with each of them. Tractor-mounted vehicles require an experienced operator and could be limited by maneuverability in the field. Cranes and cable robots are limited by the amount of area that they can cover. Aerial vehicles are usually weight limited. In addition to these physical limitations, the cost for these platforms is often significant. Costs estimated by White et al. (2012) range from $100,000 USD for a tractor mounted phenotyping platform to over $1,000 USD for a one-hour drone flight.

While each phenotyping platform has its advantages, there is a current need for a highly mobile, field-based phenotyping platform that could be deployed in locations throughout the world at an affordable cost. The affordable cost will enable those working in developing countries or remote field locations to capitalize on the development in field-based HTP (Tester and Langridge, 2010). To address this type of phenotyping, a mobile platform that integrates georeferenced, simple and basic sensors to measure canopy characteristics becomes necessary. To reduce the cost and improve the maneuverability for flexible field scouting, the platform may be manually driven - carried on shouder or pushed on wheels.

1.2 Robotic phenotypers

Breeding is a number game in essence: the more crosses and environments used for selection, the greater the probability of identifying superior variation (Araus and Cairns, 2013).

However, characterizing large mapping populations and diversity panels of thousands of recombinant inbred lines multiple times throughout their whole growth cycle by HTPP (High Throughput Phenotyping Platform) is a hard, repetitive, and time-consuming task, especially for the operators of the HTPP. In the same time, hundreds of times of testing for a tractor-mounted phenotyping platform on the filed results in the environment impact of soil compaction. Therefore, developing autonomous, light-weighted, driverless phenotyping robot is necessary for plant phenotyping research. Robotic phenotypers have the advantages of lighter weight, lower labor costs, fewer energy consumption, less environmental impacts, and higher measurement efficiency (Luis et al., 2014). With the help of a GPS-based auto-navigation system, a robotic phenotyper can automatically work at the field-scale, reducing unintended human error by the operator (white et al., 2012). Another advantage of robotoic phenotypers is their ability to work during night, which would double the productivity and allow plant traits that become more significant when there is no light to be detected. Although no commercial solution has become available for robotic phenotyping (Deery et al., 2014), many prototypes have been developed and will be operative in the future (Rubens et al., 2011).

1.3 Objectives

From 2012 to 2013, in the Instrumentation and Control Laboratory of the Biological and Agricultural Engineering Department at Kansas State University, we developed a tractor mounted, field-based, high-throughput phenotyping platform for rapid, simultaneous measurement of plant characteristics in three crop rows (Jared et al., 2016). The phenotyper has been successfully running in field for the past four seasons. Based on this experience and the research needs for handheld and robotic phenotypers, we chose the following specific objectives for this study:

1. Developing a low-cost, handheld phenotyping platform to measure basic crop traits - leaf greenness and canopy temperature, and record crop images at various growth stages. The platform will be equipped with a computer-based data acquisition system that is capable of handling multiple sensors and a GPS device that allows georeferencing of sensor data, which would provide much higher throughput than manual phenotyping. The affortable cost would allow the phenotyper to be used in less developed countries for urgently needed phenotyping to enhance their crop breeding and production.
2. Developing a robotic phenotyping platform to allow multiple sensors for fast, autonomous, high-throughput phenotyping. The robotic platform will allow autonomous or semi-atinomous phenotyping in crop fields. The adjustable width, height, and shielding of the platform will allow the platform to be used for various types of crops during their entire growth season. The light weight of the platform will reduce soil compaction, thus allowing multiple phenotyping passes. The robotic nature of the platform will allow night operations to observe certain crop traits that can be observed only during night and it would also double the productivity in phenotyping. These features cannot be found in vehicle-driven phenotypers. Very few research on robotic phenotypers can be found in existing literature and the ones found in literature do not possess all these features. Experiences and lessens learned during the design and testing of the proposed robotic phenotyper would greatly enhance the literature in this area. The developed robotic phenotyper can be directly used to assist research in phenotyping and precision agriculture for various crops in Kansas, including wheat, corn, soybean, sorghum, and forage crops.

3. Studying methods to extract useful phenotypic information from the sensor data. Specifically for this study, two traits that relate to wheat growth will be studied as examples ------ the wheat growth trend and the total number of heads formed at maturity. A digital camera will be used as the sensor and image processing techniques will used to extract the information from the images.

Handheld, vechicle-driven, and robotic phenotypers are the basic platform types for ground-based phenotyping of field crops. A vehicle-based phenotyper has been previously developed in the BAE Instrumentation and Control Laboratory. Completions of the hand-held and robotic phenotypers will alow comparison among the three types of phenotypers. Like the vehicle-based phenotyper, the developed hand-held and robotic phenotypers will become useful tools for researchers at KSU and other institutions on field phenotyping for various field crops and under various conditions. The research on wheat traits will provide an example to other researchers on the use of the phenotypers with a typical sensor – a digital camera. Once developed, the hand-held and robotic phenotypers will allow many types of sensors, including the digital camera, to conduct various phenotyping works.

Chapter 2 - LITERATURE REVIEW

2.1 Phenotypic prediction model

Phenotypic prediction according to the genetic composition of lines is a fundamental step forward for plant science and crop improvement in the 21st century (Lorenz et al., 2011; White et al., 2012; Araus and Cairns, 2014). The ability of accurately predicting a "phenotype" could lead to better understanding of the genetic basis of complex traits such as yield (Cabrera‐Bosquet et al., 2012). Progress in breeding high-yielding crop plants for meeting the future food and fuel needs by 2050 would significantly accelerate if the prediction of phenotypic consequences of genetic makeup of an organism becomes available (Hammer et al., 2006; Furbank and Tester, 2011). Phenotypic data are considered 'high-dimension, small sample size' (HDSN) data, which can be addressed with many potential models, such as partial least-squares regression, random decision forest, and support vector machines, although most genotype-phenotype mapping is inherently nonlinear (Houle et al., 2010). Rajasingh et al. (2008) proposed the concept of 'causally cohesive genotype-phenotype model', which has the quality of forcing components in a genotype-phenotype relation to cohere in a logically consistent and ordered way. However, in the standard population genetic models phenotypic data are assigned directly to genotype without involving intermediate process (Rajasingh et al., 2008; Houle et al., 2010).

2.2 Field-based phenotyping (FBP) platform

Field-based phenotyping (FBP) is an approach to deliver the requisite throughput for numbers of plants or populations, as well as an accurate description of trait expression in real-world cropping systems (White et al., 2012). The FBP platform is composed of multiple sets of proximal sensors, data acquisition, and a storage system. It provides the ability to assess plants in

real-world conditions and with population sizes consistent with those needed for breeding programs and quantitative genetic studies (Yu et al., 2008; White et al., 2012).

Several high-throughput phenotyping (HTP) platforms, capable of generating large quantities of data quickly, have been reported at both the controlled (greenhouse and growth chamber) and field-based levels (Nagel et al., 2012; Tisné et al., 2013; Montes et al., 2011; Busemeyer et al., 2013; Andrade-Sanchez et al., 2014; Haberland et al., 2010; White and Bostelman, 2010;Albus et al., 1993; White and Bostelman, 2010; Zarco-Tejada et al., 2009; Merz and Chapman, 2011).

A tractor mounted platform developed by Monte et al. (2011) used light curtains and spectral reflectance sensors for nondestructive, high throughput phenotyping. Prashar et al. (2013) developed a mobile, in-field phenotyping platform to investigate the relationship between canopy temperature of potato and its final tuber yield. The platform included a ThermaCAM P25 infrared camera (FLIR systems with the spectral range of 7.5–13 mm, USA) and a microbolometer detector with a spatial resolution of 320X240 pixels. Sankaran et al. (2015) developed an unmanned aerial phenotyping platform to evaluate the emergence rate and spring stand (an estimate of winter survival) of three winter wheat market classes. Allah et al. (2015) used an UAP (Unmanned Aerial Platform) equipped with a multispectral camera to assess the spatial field variability and detect low-nitrogen (low-N) stress tolerance of maize.

White et al. (2012) reviewed many of the field based systems, including tractor-mounted platforms, platforms operated on cranes, towers similar to sports stadium cameras, and aerial vehicles. Based on the application environment and features of the phenotyping platforms, Deery et al. (2014) classified the platforms into three types: 1) fixed systems, 2) mobile in-field systems, 3) airborne systems.

2.3 Robotic vehicles for agricultural applications

Automation of agricultural machines and development of agricultural robots (smart machines) are essential for lowering production costs, increasing production efficiency, enhancing the quality of fresh produce, and reducing the drudgery of manual labour in current and future field operations (Bakker et al., 2010; Choi et al., 2015).

Robots are of great complexity because they are comprised of several different sub-systems which are connected and correctly synchronized to execute tasks as a whole (Bechar and Vigneault, 2016). Most robots for industrial applications perform relatively simple, repetitive, well-defined, and pre-determined tasks in stable environment (Bechar and Vigneault, 2016). However, robots for agricultural applications need to have the abilities to deal with complex and highly variable environment and produce (Hiremath et al., 2014). The robotics systems used in agriculture are more sophisticated than those used in industry because they usually work with unstructured objects under unstructured environments (Bechar and Vigneault, 2016). Under such environment, autonomous robots can easily fail because of unexpected events. Thus, robotic systems used in agriculture need to be more sophisticated, making development of agricultural robots more difficult and expensive (Steinfeld, 2004).

Significant amounts of agricultural robot research worldwide have emerged in the past thirty years. Specialized sensors (Global Positioning Systems, laser-based sensors, and machine vision), actuators (hydraulic cylinder, linear or rotation electronic motor), and electronic equipment (PLC, industrial PC, and embedded computers) have been integrated in varous agricultural robots with the aim of configuring autonomous systems to shift operator activities in agricultural tasks (Emmi et al., 2014). In the past three decades, there have been 255 published articles about transplanting and seeding robots research, 326 articles about plant protecting

robots, and 302 articles about (selective) harvesting robots (Bechar and Vigneault, 2016). Most of the agricultural robots have not yet been commercialized for field operation because of their production inefficiencies (Zhao et al., 2016). There are also some agricultural robots being put in practice with good performance for dedicated tasks. Examples include milking robots, transplanting robots used in controlled environment, grafting robots, and autonomous combines and tractors (Yasushi et al., 2001; Thuilot et al., 2002; Kolbach et al., 2013).

Robotic agriculture, or utilization of robots equipped with proper agricultural tools to accomplish specific field tasks, is the developing trend of future farming (Blackmore et al., 2005; Luis et al., 2014). The development of agricultural robots has experienced an increasing interest, and many experts have been exploring the possibilities of developing more rational and adaptable vehicles based on a behavioural approach (Pedersen et al.,2008). According to the crop production cycle, agricultural robots can be divided into three types - seedling robot, plant protecting robot, and (selective) harvesting robot (Blackmore et al., 2005). For orchards and horticultural crops, there have been several prototypes developed, such as oranges harvesting robot, strawberries harvesting robot, tomatoes selective harvesting robot, apple harvesting robot, asparagus selective harvesting robot, weeding robot, and grafting robot (Hannan and Burks,2004; Chi & Ling, 2004; Cembali et al., 2005; Kondo et al., 2005; Zhao et al., 2011).

Crop scouting robots had the same function of collecting crop data as phenotyping robots, which have mainly assisted in weed and disease controll. For instance, Bak and Jacobsen (2004) developed an API (Accurate Positioning Interface) platform for patch spraying (Bak and Jacobsen, 2004). The third API platform, further developed by Aalborg University in Denmark, has four-wheel drive, four-wheel steering with two motors per wheel, and its height clearance and track width are 60cm and 1m, respectively (Bisgaard et al., 2004). By now, only a limited

number of research has investigated robotic phenotyping platforms (Montes et al., 2011; Nagel et al., 2012; Tisné et al., 2013; Sankaran et al., 2015; Allah et al., 2015). Rubens et al. (2011) developed a phenotyping robot, and its height clearance could be manually adjusted.

Biber et al. (2012) developed an autonomous robot to perform repeated phenotyping tasks for plant breeders. Its chassis height and width are adjustable. However, because the adjustment range of the chassis height is from 0.4 to 0.8 meters, this phenotyper can only perform data collection at the early stage of crop growth. Thus, designing a flexible chassis to allow phenotyping of crops within their entire growth season is necessary.

2.4 Potential technologies for measuring the plant leaf traits

Digital-based system and sensor technologies have been used in various plant phenotypers to allow a broad range of evaluation of complex traits, such as yield, growing period, resistance to diseases, plant architecture, and other fundamental quantitative parameters (Kumar et al., 2015). Digital RGB imaging, spectroscopy, thermography, fluorescence imaging, 3D stereo imaging and LIDAR (Light Detection and Ranging), and tomography have been tested for plant phenotyping (Großkinskyet al., 2015).

2.4.1 3D reconstructions by lasers, stereo cameras, and time of flight cameras

3D imaging systems are used to estimate biomass, canopy structures, and plant height in field experiments. In controlled environment, 3D imaging systems have been implemented to conduct detailed studies of plant morphology, such as leaf angles and surface, and overall growth rates (Großkinsky et al., 2015). Biskup et al. (2007) presented an area-based, binocular stereo system composed of commercially available components that allowed three-dimensional reconstruction of small- to medium-sized canopies on the level of single leaves under field conditions. Spatial orientations of individual leaves were calculated with the 3D matching and

segmentation techniques (Biskup et al., 2007). Pengelly et al. (2010) used a 3D nondestructive imaging technique to measure the relative growth rate (RGR) of plants. Chaivivatrakul et al. (2014) proposed and demonstrated the efficacy of an automatic corn plant phenotyping system. They acquired 3D image data from a 3D time-of-flight camera integrated with a plant rotating table to form a screening station. By an experimental study with five corn plants at their early growth stage (V3), they obtained promising results with accurate 3D holographic reconstruction (Chaivivatrakul et al., 2014). Kempthorne et al. (2015) reconstructed the wheat leaf surface from three-dimensional scanned data using a parameter isolation technique.

2.4.2 Light curtain arrays

Light curtain (LC) is a recently introduced phenotyping technology. The setup consists of a pair of parallel bars, one radiating and the other receiving the emitted light. The receiving bar detects whether an object interruptes the light beams. By scanning the crop, LC arrays produce a binary data set of the plants' profile (Fanourakiset al., 2014). Light curtain arrays have been successfully used to assess canopy height in the field (Montes et al., 2011; Busemeyer et al. 2013). Fanourakis et al. (2014) explored the method of measuring the leaf area by LC scanning and evaluated the accuracy and applicability of LC in estimating the maximum height (from the base to the highest leaf tip) and leaf area on a phenotyping platform under controlled environment.

2.4.3 RGB imaging

RGB imaging is a useful tool for morphological studies of plants. Usually, RGB imaging is used to detect projected structures, surfaces, and colors of diverse plant organs such as shoot, root, seeds, and leaf spots. With these images, plant traits such as growth dynamics, disease, pigmentation, and senescence are evaluated by image processing technology (Hartmann et al.,

2011; Großkinsky et al., 2015; Kloth et al., 2015). Bojacá et al. (2011) used a DX4530 digital camera to take field potato images to estimate canopy coverage by quantifying the percentage of pixels representing the plants within the total number of pixels in the image (Bojacá et al., 2011). Based on the RGB images, Ramfos et al. (2012) proposed a straightforward and inexpensive digital image analysis method to measure leaf area and the length of strapped-shaped seagrass leaves.

Image analysis algorithms are digital image processing techniques used to extract meaningful information from images. One website dedicated to plant biology describes 139 image analysis software tools that are manually operated, automated, or semi-automated (Plant Image Analysis, 2016). Image segmentation classifies or clusters an image into several disjointed parts by grouping the pixels to form regions of homogeneity based on pixel characteristics such as gray level, color, texture, intensity, and other features. Knowledge-based approaches such as intensity-based methods, discontinuity-based methods, similarity-based methods, clustering methods, graph-based methods, pixon-based methods, and hybrid methods can be utilized to segment an image (Khan and Ravi, 2013).

Intensity-based segmentation, or the threshold-based approach, is one of the simplest methods for segmenting an image. Images are classified based on the postulate that pixels belonging to a certain range of intensity values represent one class, and the rest of the pixels in the picture represent another class. Thresholding can be globally or locally implemented (Khan and Ravi, 2013). Wang et al. (2013) used OTSU (named after Nobuyuki Otsu) and CANNY (developed by John F. Canny in 1986) operators to segment an area of a target leaf by choosing thresholds with the mapping function, the shape identification algorithm, and pattern recognition. The optimization process of the algorithm, which included mapping function, shape

identification algorithm, morphological methods, and logical operations, was designed to obtain the entire leaf edge precisely. Experiments showed that this algorithm feasibly and effectively segmented jujube leaf images from real-time video systems. Wang et al. (2013) proposed a thresholding method when they explored the relationship between rice image feature parameters and three plant indices (i.e., aboveground biomass, N content, and leaf area index). The threshold was set based on the magnitude and distribution of the green channel minus red channel values.

A geometric-optical (GO) model classified objects within an image scene into four categories: sunlit foliage, sunlit background, shaded foliage and shaded background (Fan et al., 2013). Zeng et al. (2015) proposed a new leaf area index (LAI) extraction method using sunlit foliage from downward-looking digital photography under clear-sky conditions. Using this method, an automated image classification algorithm called LAB2 extracted the sunlit foliage. A path length distribution-based method estimated the clumping index. Leveled digital images were used to quantify the leaf angle distribution (LAD) and G (leaf projection) function, and the LAI was eventually obtained by introducing a GO model that quantified the sunlit foliage proportion (Zeng et al., 2015). Macfarlane and Ogden (2012) proposed the LAB2 method, which utilized a* and b* values of each pixel to classify green vegetation by a minimum-distance-to-means classifier, where a* and b* are parameters used in the CIE L*a*b* color space. (L* is the luminance component, a* represents color on the green-magenta axis, and b* represents color on the blue-yellow axis.).

Chapter 3 - DESIGN OF A HANDHELD PHENOTYPER

3.1 Introduction

In this chapter, we describe a handheld phenotyping platform developed in the Instrumenatation and Control Laboratory of the BAE Department at KSU. The phenotyper is aptly named "Phenocorn" because of its physical resemblance to the mythical unicorn. The Phenocorn integrates spectral reflectance (Normalized Difference Vegetative Index, NDVI), canopy temperature (CT), plant video, and geo-referenced data collection. The Phenocorn was evaluated across three years of replicated trails consisting of 10 elite breeding lines of wheat (Triticum aestivum) in two countries – Mexico and Bagladesh, and demonstrated the utility of this platform for rapid assessment of accurate plant phenotypes (Crain et al., 2016).

3.2 Material and Methods

3.2.1 Phenocorn design

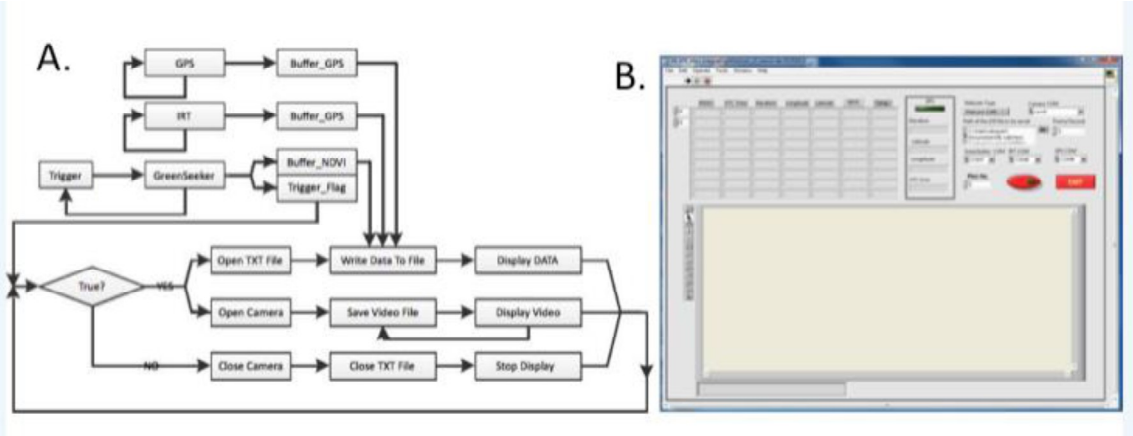
The Phenocorn integrates several pieces of hardware into one functional unit (Figure 3.1). At a basic level, the Phenocorn is comprised of an infrared thermometer (IRT) sensor to measure canopy temperature (CT), an NDVI sensor for spectral reflectance measurements, and a high-precision GPS unit to geo-reference all data. NDVI and CT were chosen as parameters to measure because of their documented relationship to yield (Amani, 1996; Babar and Reynolds, 2006; Gutierrez et al., 2010). Geo-referencing gives each sensor measurement a precise location and prevents assigning data to the wrong plot and entry. In 2014, we made an additional modification by adding a web camera to record color images that were likewise geo-referenced. The sensors are all connected to a laptop computer serving as the main control unit for the

system. The model, technical specifications, and approximate cost for all components are listed in Appendix A.

Figure 3.1 Integrated hardware components of Phenocorn.

The system was operated through a custom-designed National Instruments LabView program. Within the program, data collection is initiated by pressing the GreenSeeker trigger and data is logged as long as the trigger is held. The software written in LabView 2012 (National Instruments, Austin, TX) had two main functions. The first function was to establish a connection with the sensors and the GPS unit and to collect the data from these devices. A separate module was built in LabView for each sensor because the connection parameters and output data for each device differ. Second, the GPS and sensors readings were processed and recorded into a useable file format. Each new set of data was written to a new line in a text file at a rate of 10 Hz, with each line having a unique index number along with sensor observation data. A block diagram of the Phenocorn software system is provided in Figure 3.2 (Crain et al., 2016).

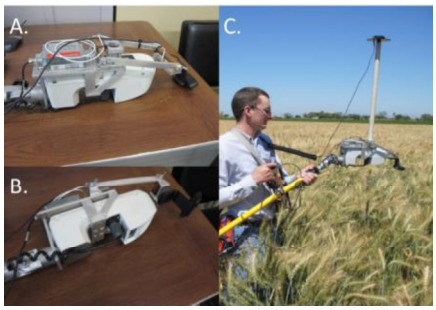
Figure 3.2 Block diagram of the Phenocorn software system (A) and its Labview control platform (B) (Crain et al., 2016).



3.2.2 Phenocorn data acquisition

In the field each instrument (GPS, IRT, and camera) was mounted to the handheld GreenSeeker, using a custom fabricated mounting bracket. The bracket enclosed the head of the GreenSeeker sensor without blocking either the light source or the collector (Figure 3.3). All sensors were mounted so that they had a nadir field of view, and the GreenSeeker (base of the Phenocorn) was held approximately 80 cm above the crop canopy. Data were collected at a walking speed of approximately 1-2 m s^{-1} and were recorded from the entire experiment area, including plot borders and alleys. The Phenocorn records NDVI, CT, and GPS location at 10 Hz, and color photos were saved at 3 Hz (Crain et al., 2016).

Figure 3.3 A and B. Custom mounting bracket to hold individual sensors. C. Using the Phenocorn as a handheld platform in the field, Ciudad Obregon, Mexico (Crain et al., 2016).



3.2.3 Summary of raw data and derived parameters

As mentioned above, the handheld phenocorn was used to collect the indexes and information related to wheat in the field, which included NDVI, CT, and wheat images. The raw data and their applications are shown in Table 3.1. Crain et al. (2016) studied the relationships of NDVI vs. wheat yield and CT vs. wheat yield respectively in 2014. The results will be demonstrated in section 3.3.3.

Table 3.1 Raw data and their applications (White et al., 2012).

Index	Instrument	Target trait	Applications or relevant traits	Yield
NDVI	GreenSeeker	Nitrogen Leaf area index Plant biomass	Plant nitrogen status especially under stress;overall growth	related
CT	Thermometer	Transpiration	Instantaneous transpiration and hence crop water status	related
Image	Camera	Maturity Multiple stages	Plant development; Leaf orientation and size; Tracking leaf senescence;Seedling emergence,onset of grain-filling,senescence;Flower number; wheatheads number	related

3.3 Results and Discussion

3.3.1 Phenotyping platform test

To test the Phenocorn, an existing field trial was utilized at the International Maize and Wheat Improvement Center (CIMMYT), Ciudad Obregon, Mexico. The trial consisted of 10 genotypes selected from historical and current elite breeding lines. The plots were 10 meters long by 2.4 meters wide. Wheat was planted on raised beds with two rows per bed. The trial was planted on February 27, 2013 and November 22, 2013 for each year (2013 and 2014, respectively) and irrigation and nutrient levels were maintained at optimal levels with pesticides applied as needed. The experiment of the phenotyping platform test was done on 21 Feb. 2014

by Dr. Jared Crain. Nineteen plots were measured using the handheld phenocorn, and the data files, including one text format file for storing the measured data and 19 corresponding videos, were saved in the specified folder. All values of the parameters were saved in one text file, including number of the plot, number of the set of data points, UTC-Time, longitude, latitude, NDVI, canopy temperature, and the path of video.

The data and video of the Plot1 were not used because they were only collected for testing the handheld phenocorn at the beginning of the experiment. The data and videos of the other 18 plots were used to analyze the phenocorn's performances. Based on the data, the mean sampling frequency is almost 9.8Hz, and the frame ratio of the videos is 1.6 frames per second, as shown in Table 3.2. For the plot2, the length of the plot was 44.7m calculated by the GPS coordinates. The total frames number was 59 (Table 3.2). If the condition was that the video could cover the whole plot by frames without overlap between the two neighboring frames, the mean width of each frame should be 0.75m during measuring with constant speed. In fact, the width of each frame was between 0.35m and 0.45m. So, the videos taken in the experiments didn't cover the corresponding whole plots because of its lower frame ratio. The conclusion through experimentation was that the laptop performance, especially the graphics performance, was limiting the frame rates.

Table 3.2 Data for calculating sampling frequency and frame ratio.

Plots	Starting Time*	End Time*	Time (s)	Data Points	Frames	Sampling Frequency	F/s
plot2	172522.4	172558.8	36.4	323	59	8.9	1.6
plot3	172602.4	172639	36.6	346	58	9.5	1.6
plot4	172642	172720.2	38.2	370	61	9.7	1.6
plot5	172724	172801	37	367	59	9.9	1.6
plot6	172803.8	172842.8	39	391	64	10.0	1.6
plot7	172846.2	172925	38.8	388	64	10.0	1.6

plot8	172927.4	173007.8	40.4	405	66	10.0	1.6
plot9	173011.8	173051.2	39.4	394	63	10.0	1.6
plot10	173054.2	173133.8	39.6	398	65	10.1	1.6
plot11	173137.6	173216	39.4	386	62	9.8	1.6
plot12	173229	173307	38	374	63	9.8	1.7
plot13	173311	173349.6	38.6	371	61	9.6	1.6
plot14	173353.2	173432.2	39	387	63	9.9	1.6
plot15	173435.8	173514.6	38.8	390	63	10.1	1.6
plot16	173517.4	173556.8	39.4	392	65	9.9	1.6
plot17	173600.8	173637.8	37	371	60	10.0	1.6
plot18	173640.6	173717.8	37.2	372	61	10.0	1.6
plot19	173721.6	173756.8	35.2	353	58	10.0	1.6
Mean						9.8	1.6

* UTC-Time, format: HHMMSS.S.

One developed Matlab program was used to process these data. The program functions included: doing image processing and estimating the percent vegetation coverage (PVC) at the corresponding GPS coordinates for 18 videos, normalizing the NDVI data, PVC data, and temperature data separately, and drawing the NDVI , PVC, and temperature distribution plots in the geographic space, as shown in Figures 3.4, 3.5, and 3.6. From Figures 3.4 and 3.5, we found that the surfaces of the 3D plots represented the wheat growing status, different status at different position, and especially the boundaries between two blocks were clearly showed in Figure 3.4. Similarly, the peaks in Figure 3.5 also represented the positions where boundaries between two blocks existed. In one word, the NDVI, PVC, and temperature data obtained by the handheld phenocorn could have reflected wheat growing status in the field, and the handheld phenocorn could be used as an instrument to do plant phenotyping information collection.

Figure 3.4 Normalized NDVI distribution in the geographical space.

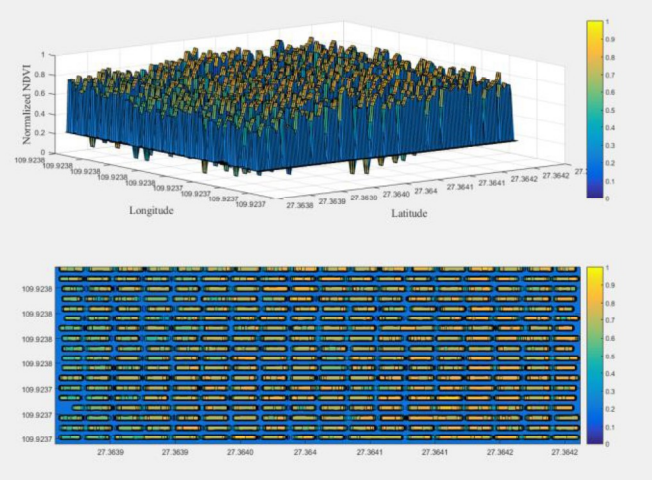


Figure 3.5 Normalized PVC distribution in the geographical space.

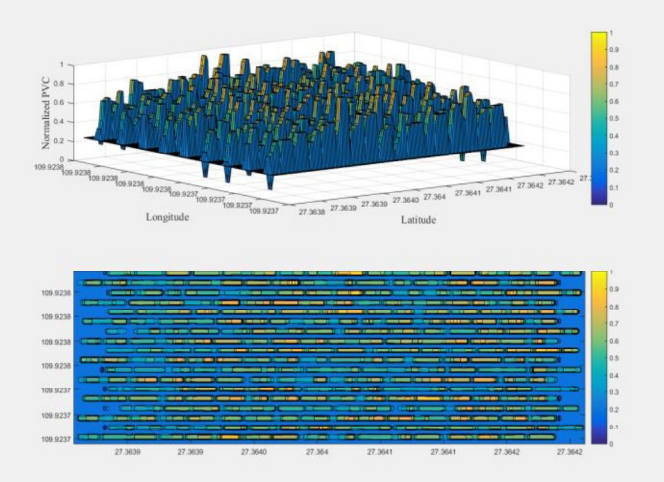
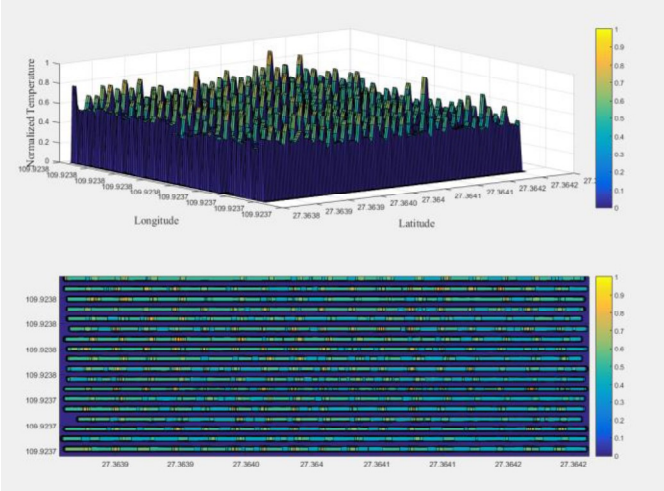


Figure 3.6 Normalized temperature distribution in the geographical space.



3.3.2 Phenocorn CT (Canopy Temperature) test

Crain et al. (2016) finished a series of experments to validate the readings of the Phenocorn CT to current instruments in 2013. They collected concurrent canopy temperatures with both the Phenocorn and a handgun IRT across five time points throughout the vegetative and grain fill growth period. As shown in Table 3.3, there was a high correlation ($p<0.01$) on four of the five days between instruments. The results indicated that the Phenocorn could provide reliable temperature data that was similar to current methods (Crain et al., 2016).

Table 3.3 Correlation coefficient between Phenocorn CT vs. handgun IRT (Crain et al., 2016).

Date	Growth Stage	Phenocorn CT vs. Handgun CT
3/29/13	Vegetative	-0.08
4/1/13	Vegetative	0.84***
4/4/13	Vegetative	0.52**
5/1/13	Grain Fill	0.67***
5/14/13	Grain Fill	0.74***

1)* Model significant at the 0.05 level of probability.
2)** Model significant at the 0.01 level of probability.
3)*** Model significant at the <0.001 level of probability.

The correlation between canopy temperature and grain yield was evaluated by Crain et al. (2016) in 2013 as shown in Table 3.4. The results showed that there were significant negative correlations between canopy temperature and grain yield during grain fill, and non-significant negative correlations during the vegetative growth stages. By comparing to the results of the previous studies (Keener and Kircher, 1983; Balota et al., 2007), Crain et al. (2016) concluded that the Phenocorn IRT was performing as well as current canopy temperature measurement techniques.

Table 3.4 Correlation coefficient between Phenocorn CT and grain yield for vegetative and grain filling growth stages in wheat (Crain et al., 2016).

Date	Growth Stage	Phenocorn CT vs. Yield
3/29/13	Vegetative	-0.02
4/1/13	Vegetative	-0.34
4/4/13	Vegetative	-0.22
5/1/13	Grain Fill	-0.55**
5/14/13	Grain Fill	-0.40*

1)* Model significant at the 0.05 level of probability.
2)** Model significant at the 0.01 level of probability.
3)*** Model significant at the <0.001 level of probability.

3.3.3 NDVI vs. grain yield

Crain et al. (2016) also studied the relationship between NDVI and grain yield. They collected NDVI simultaneously with canopy temperature for each date of data observations. The results were shown in Table 3.5. NDVI was significantly correlated to grain yield ($p<0.001$) during the grain fill stages. They came to the same conclusion as other research (Babar and Reynolds, 2006) that the correlation of NDVI and grain yield increased throughout the growing season. That is, the ability of the Phenocorn to collect NDVI, and NDVI's relationship to grain yield will allow researchers to make more informed selection decisions (Crain et al., 2016).

Table 3.5 Correlation coefficient between Phenocorn NDVI and grain yield for vegetative and grain filling growth stages in wheat (Crain et al., 2016).

Date	Growth Stage	NDVI vs. Yield
3/29/13	Vegetative	0.09
4/1/13	Vegetative	0.24
4/4/13	Vegetative	0.46*
5/10/13	Grain Fill	0.65***
5/14/13	Grain Fill	0.65***

1)* Model significant at the 0.05 level of probability.
2)** Model significant at the 0.01 level of probability.
3)*** Model significant at the <0.001 level of probability.

3.3.4 Advantages and disadvantages of phenocorn

Phenocorn was designed to provide dense phenotypic data for genetic analysis and plant breeding. Some of the unique advantages in the Phenocorn are that it simultaneously collects multiple measurements including NDVI, CT, and images, but also data is geo-referenced. Geo-referencing will lead to less error in data transcription, and thus higher data integrity. The addition of image collection to the Phenocorn opens an entirely new avenue for high-throughput data collection that could add value to breeding programs. This also highlights the flexibility of the underlying software platform of the Phenocorn, which allows for modification and addition of new sensors (Crain et al., 2016). In addition, phenocorn still have the other advantages such as good for monitoring and resolution, flexible deployment, and low cost. However, because it is a kind of handheld instrument, it also has some disadvantages such as time consuming, limited with payload, and hard operation for large experiments.

3.3.5 Discussion about the total cost of phenocorn

The phenocorn was considerably less expensive than other phenotyping systems (Crain et al., 2016). The components integrated in the Phenocorn are listed in Table 3.6. The total cost of one Phenocorn is US$13,680 while high-clearance vehicle phenotyping platforms have been estimated at more than US$100,000 (White et al., 2012). A large propotion of the expense is from the SXBlue III-L (US$8400), which is a high-precision differential correction GNSS with centimeter-level resolution. If the user does need high level of precision, less accurate and less expensive GNSS will be used and the total cost of the phenocorn will be reduced. In the same manner, the total cost can be reduced by selecting other more affordable sensors (Crain et al., 2016).

Table 3.6 Phenocorn components and their prices.

Instrument	Company	Specifications	Price
Aspire 4830	Acer Inc., San Jose, CA	Windows 7, 2.4 Ghz process, 8 GB RAM, 64 bit	$750
GreenSeeker	Trimble, CA	RS-232 to USB converter used	$4,000
SXBlue III-L	Geneq, Montreal, Canada	Omnistar HP for 95% accuracy <= 10cm	$8,400
CT Serial Thermometer	MicroEpsilon, NC	Analog to Digital (AD) converter, Temperature resolution 0.1 °C, System accuracy: ±1 °C	$430
C920 Camera	Logitech	HD 1080p	$100
Total cost			$13,680

3.4 Conclusions

The Phenocorn integrates two proximal sensors currently used in plant phenotyping as well as geo-referencing, which allows for a faster data collection in the field and streamlined data processing. The test results about the handheld phenocorn showed that the handheld phenocorn could be used as an instrument to do plant phenotyping information collection.

To further evaluate the Phenocorn, Crain et al. (2016) did a serial of field trials at the International Maize and Wheat Improvement Center (CIMMYT), Ciudad Obregon, Mexico. Validation of the Phenocorn shows that it performs as well as current methods for canopy temperature; additionally, NDVI and CT data from the Phenocorn were significantly correlated to grain yield. The Phenocorn is a robust, affordable platform that can be modified to fit user' needs and should help close the gap between genomics and phenomics.

3.5 Future work

As a first-generation system, the phenocorn has been able to simultaneously collect NDVI (Normoalized Differnece Vegetative Index), CT (Canopy Temperature), and plant images

with precise assignment of all measurements to plot location by georeferenced data points (Crain et al., 2016). However, the improved design to reduce the total cost and instrument weight according to the requirements of practice research will be the work in the future, which will promote rapid dissemination and utilization of the phenocorn in the plant breeding field.

Chapter 4 - ROBOTIC PHENOTYPER DESIGN

4.1 Introduction

The robotic phenotyper is designed for varous types of field crops, including wheat and other small grains, soybean, corn, grain sorghum, cotton, and forage crops. The average height of corn is about 2.4m. The average height of grain sorghum, soybean, wheat, cotton, and other small grain or forage crops is 1.0 – 1.4m. The robotic phenotyper is designed for most of the field crops. Thus, a proper chassis height needs to be determined. The height of the chassis should be the crop height plus a space for the sensors to perform properly. An example is the focal length of a digital camera. Excessive chassis height would require a large base of the phenotyper to maintain the stability, which may not be desirable. Therefore, the selected chassis height should allow field phenotyping for most interested crops except corn, for which the phenotyper can only be used during the vegetative growth stage. The minimum chassis height is limited by the drive mechanism of the robot.

In Kansas, the row spacing for most field crops is either 25.4 or 76.2 cm. To achieve high throughtput phenotyping, multiple crop rows need to be covered simultaneously. Design of the chassis width should allow simultaneous phenotyping of five rows for crops planted at 25.4 cm spacing, and two rows for those planted at 76.2 cm. Exessive chassis width not only will increase the weight of the phenotyper, thus causing soil compaction, it will also reduce the maneuverability of the robot.

The chassis height and width need to be adjustable to allow the robotic phenotyper to work on different crops and under different conditions.

An advantage of a ground-based phenotyper over an aerial-based phenotyper is the much larger payload capability. Many sensors prohibited for aerial applications can be used on ground-based phenotypers. However, there is still a limit on the payload on a ground-based phenotyper. Excessive weight mounted on the chassis of the robotic phenotyper may slow down, even stall the robot. Many sensors have been used in phenotyping and many more will be attempted in the future. These sensors have different weights. Many sensors are light in weight (below 10 kg), but some heavier sensors such as a light detection and ranging (LiDAR) sensor or a electromagnetic induction (EMI) sensor may weigh up to 20 kg. In addition, some sensors may need special mounting brackets which would further increase the weight. To provide the flexibility for sensor selection, a reasonably high payload capability that allows up to 10 sensors to be mounted is desirable.

The ground speed of the phenotyper direcly determines its throughput. Ideally, a fast ground speed is preferred. However, because each sensor has its response time, the robot should move at a speed that would allow all the sensors mounted on the robot to yield reliable measurements at a proper sampling frequency. From the experience we obtained in developing the vehicle-based phenotyper, we learned that most sensors used on that phenotyper were able to collect stable measurement data at a sampling frequency of 10 Hz, while the GPS unit used to georefernce the measurement data could reliably update the location data at a sampling frequency of 5 Hz (Barker et al., 2016). At a ground speed of 3.2 km/h and a sampling frequency of 10 Hz, the robot would move 0.89m each second and the spatial distance between adjacent sampling points would be 8.9cm, For a 0.9x1.8m block in a breeding field, about 20 data points can be taken when the robot travels through a block in the longitudinal direction, which is sufficient for most measurements. For image sensors, with a frame rate of 25 frames per second,

the spatial distance between adjacent image frames would be 3.56cm, which is also sufficnet for most image processing schemes, such as leave size measurement. A higher ground speed of the robot may result in insufficient measurement resolution and accuracy.

Optical sensors, including most image sensors, are sensitive to ambient light. Thus, it is desirable for the phenotyper to have a flexible shading mechanism. Depending on the location of the sun and the time of the day, the shading area should be adjustable, from partial shading to complete sahding so that consistent measurements can be made for most sensors.

To allow the robot to follow complicated routing in field, including headland turns, the robot must have a four-wheel drive, with each wheel controlled individually. Because the robot is completely powered by batteries, power consumption for each electric and electronic component needs to be minimized, and the period for continuous operation of the robot needs to be sufficiently long. Finally, the robotic phenotyper must have a manual shutoff function so that when danger is present, the robot can be immediately stopped.

Based on these considerations, the following design specifications were determined:

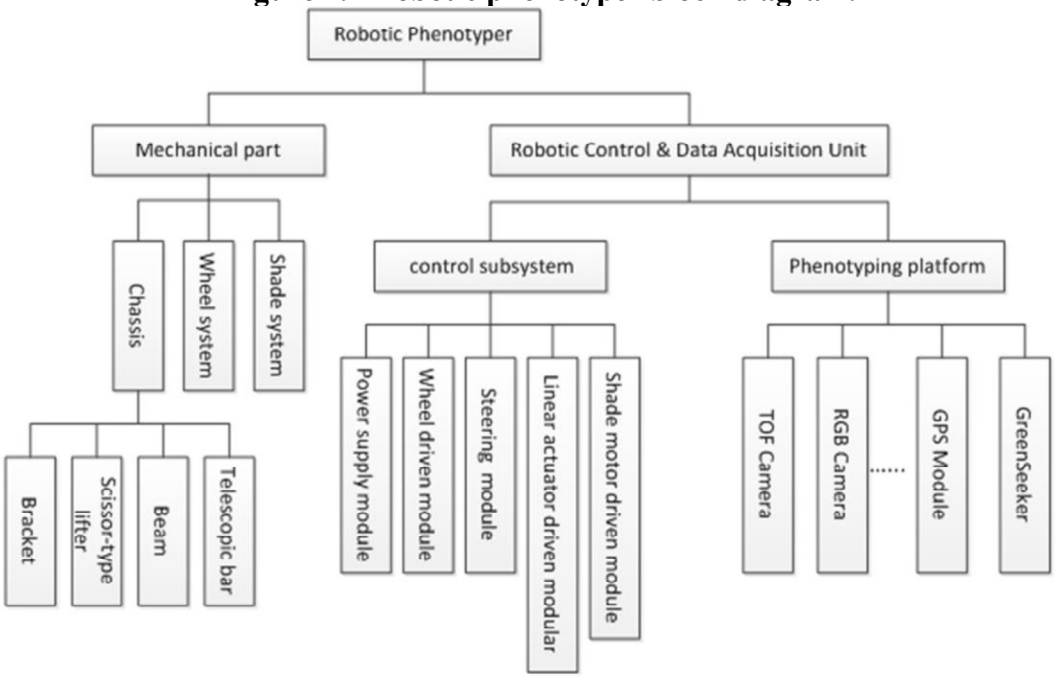
- Adjustable chassis height: from 0.9 to 1.7m;
- Manually adjustable chassis width: from 0.5 to 1.5m.
- Load capacity:80kg;
- Traveling speed in field: 3.2 km/h;
- Adjustable shades;
- Four-wheel drive and four-wheel steering with two motors per wheel system;
- Continuous operating period for the four hubmotors, four steering motors, other DC motors, the control system, data acquisition system, and all sensors: 6 hours;
- Emergency stop function: equipped.

The main design tasks include designing the mechanical structures using a modular design approach, developing a communication and control system among multiple microcontrollers based on CAN bus, and testing basic function performances - steering, load lifting, and movement. These functions are evaluated through tests.

4.2 Material and Methods

The robotic phenotyper used a modular design approach. The modular design method typically decomposes a large system into smaller subsystems, or modules, with specific functions (You and Smith, 2016). In addition to cost reduction and design flexibility, modular design offers benefits such as flexible augmentation and exclusion. The subdivision of modules is based on the minimum interaction between modules but the maximum interaction between components

Figure 4.1 Robotic phenotyper block diagram.



within the modules (You and Smith, 2016). Modules in the robotic phenotyper are shown in Figure 4.1.

The robotic phenotyper is designed to possess crop-sensing capacities and autonomous operations in order to achieve high-throughput data collection of crop phenotypes with continuous operation in the field. The system consisted of five principal components: a light chassis with adjustable clearance and width, a four-wheel-drive system, a shade system, a control system, and a phenotyping sensor station.

4.2.1 Chassis

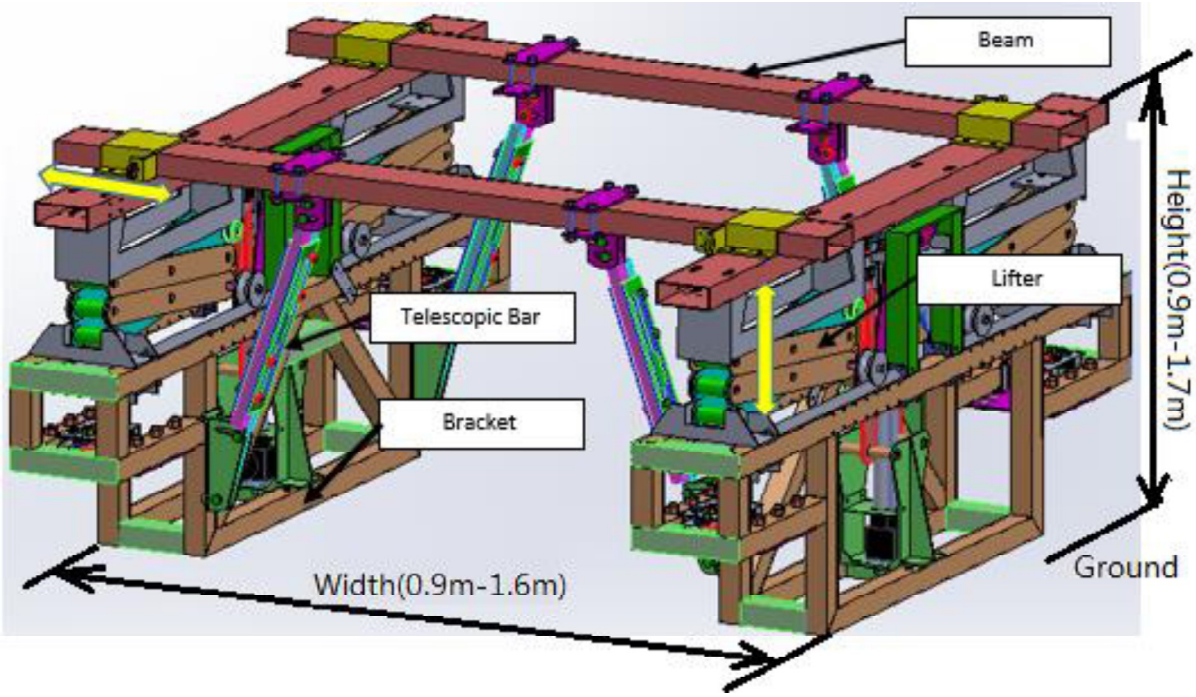
Usually, there are two basic robot chassises used in agriculture as shown in Figure 4.2. One kind of robot chassis is designed for robot easily running between two crop rows as the Rowbot system in Figure 4.2 (a), and another one for robot easily moving over the crop in the field (Figure 4.2 (b)). The designed chassis for phenotyping robot is similar to the BoniRob2, but its mechanical structure should be more flexible and its clearance height and width should be adjustable in order to meet the measuring need of covering the crop whole growth cycle.

Figure 4.2 Two kinds of robot chassis (M. Berducat, 2015).

(a) Rowbot system (b) BoniRob2

As shown in Figure 4.3, the designed chassis consisted of brackets, telescopic bars, lifters, and beams. The brackets on both sides supported the robotic phenotyper, and the wheel systems, lifters, control systems, and batteries were attached to the brackets. The beams connected and secured the two sides of the chassis. The width and height of the chassis were designed to be adjustable, and the telescope bars attached to the bracket and beam stabilized the robotic phenotyper after the lifter adjusted the beam height. As the yellow arrows show in Figure 4.2, the left side of the chassis was able to move along the beams after the beam-fastening screws were removed, and the lifter systems adjusted the height of the robotic phenotyper. The height of the robotic phenotyper was adjustable from 0.9 m to 1.7 m. Two scissors units were built into each side of the lifter; the adjustable height of each unit was 0.4 m. The number of scissors units in the lifters could be changed based on the final crop heights. The width of the chassis ranged from 0.9 m to 1.6 m.

Figure 4.3 Chassis of robotic phenotyper.



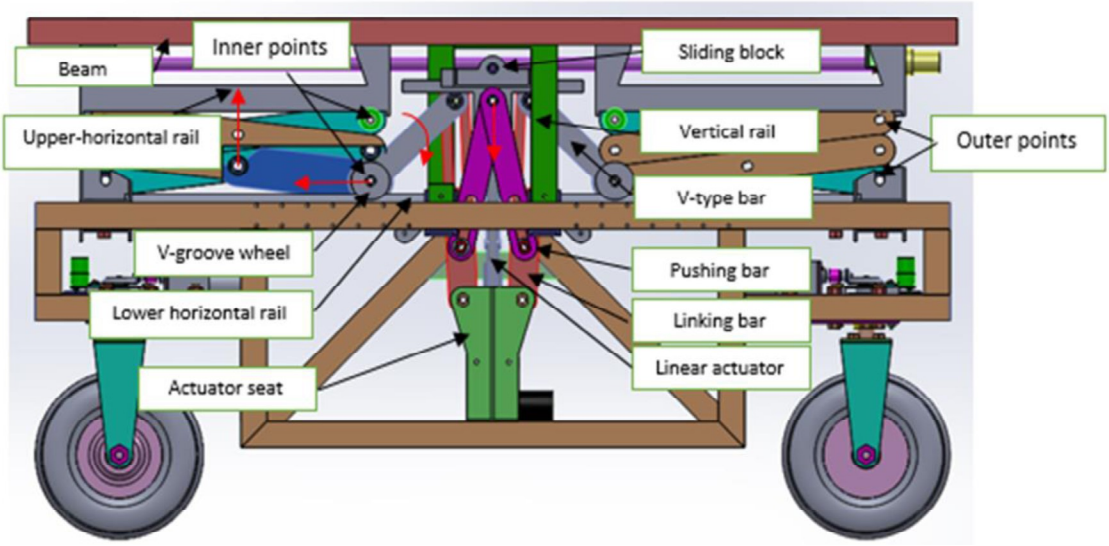
4.2.1.1 Lifter design

There are two kinds of adjustable robot chassis reported in the literatures (Rubens et al., 2011; Biber et al., 2012). One clearance height of robot chassis is manually adjustable (Figure 4.4 (a)), and that of another one is automatically adjustable (Figure 4.4 (b)). The designed lifter with scissor-type mechanical structure in the chassis is used to adjust the clearance height with electrical linear actuator. There isn't any one literature reported about this design idea by now.

Figure 4.4 Two types of adjustable robot chassis for phenotyping (Rubens et al., 2011; Biber et al., 2012).

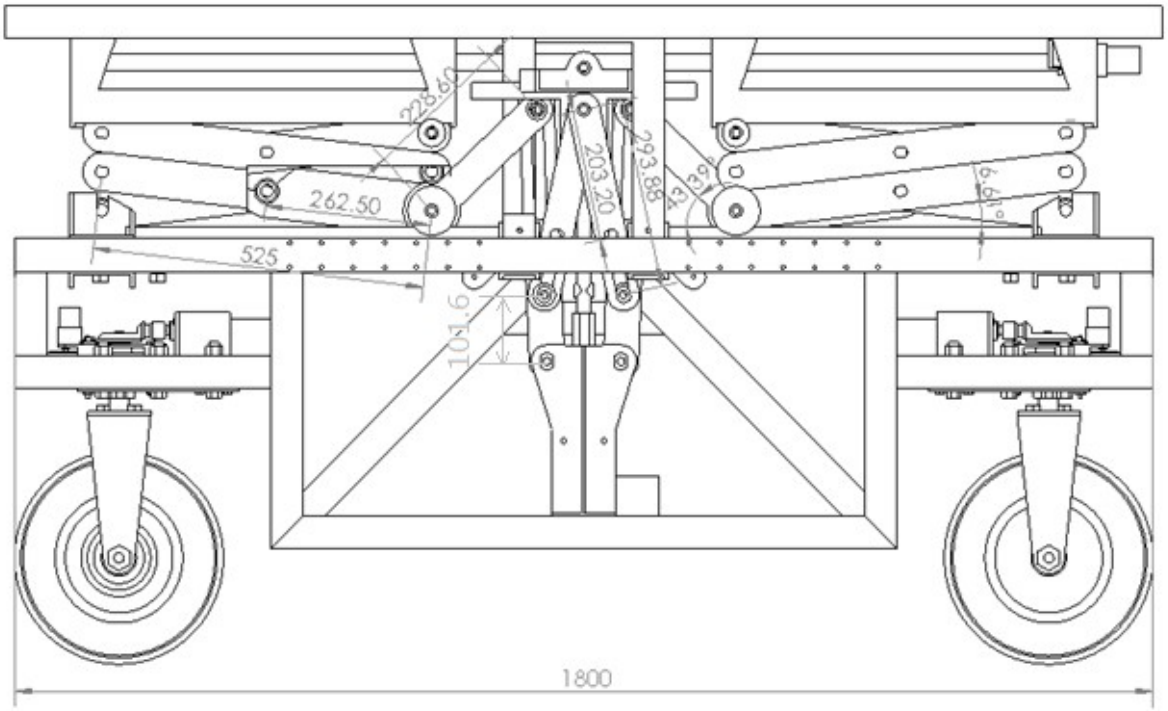
(a) (b)

Figure 4.5 Scissor-type lifter in chassis.



The technical details were shown in Figure 4.5. The chassis contained a symmetrical lifter on each side of the robot. Each lifter included a linear actuator, actuator seat, linking bars, pushing bars, v-type bars, lower horizontal bars, v-groove wheels, vertical rails, sliding block, upper-horizontal rails, and beam (Figure 4.5). The two outer points at the upper and lower corners of the scissors-type parts were fixed on outer corners of the upper horizontal rail and bracket, respectively. The linear actuator drove the two inner points of the scissors-type parts moving along the upper horizontal rails and lower horizontal rails. The beam connected the two sides of the lifter in order to maintain simultaneous movement of the two sides. The red arrows in Figure 4.5 indicate movement directions of corresponding components during lifting.

Figure 4.6 Key component dimensions in the lifter.



4.2.1.2 T&W-type bar design

A T&W mechanism was used for the lifter design. The T&W-type michanism connected a T-type pressing part connected to the linear actuator and screwed a W-type pushing bars to the T-type part. As shown in Figure 4.5, the T&W-type driving structure is comprised of a T-type sliding block, pushing bar, and linking bar. Based on bracket dimensions and design objectives, dimensions of key components are determined and labeled in Figure 4.6. The design of this driving structure attempted to ensure that the lifting range exceeded 0.35 m per scissors unit and that the maximum lift load exceeded 40 kg per side of the robotic phenotyper. The bracket length was 1.8 m, as shown in Figure 4.6. The stroke of the 24V linear actuator was 250 millimeters, of which 200 mm was used; its maximum thrust was 2500N.

Figure 4.7 (a) shows a design sketch of the lifter, and Figure 4.7 (b) and (c) show a two-phase lifting process in which the T-type bar causes the pressing phase and the pushing bar causes the pushing phase, respectively. Because a smooth transition from the pressing phase to the pushing phase during lifting is essential, the design aimed to determine the moving length Dx (Figure 4.8 (a)) of the end of the V-type bar that was connected to the T-type bar so that the pushing bar was in the working position before the end of the v-type bar arrived at the end of the T-type bar, as shown in Figure 4.7 (b).

Figure 4.7 Lifting process: (a) initial position, (b) change of critical driving position from pressing to pushing, and (c) final position.

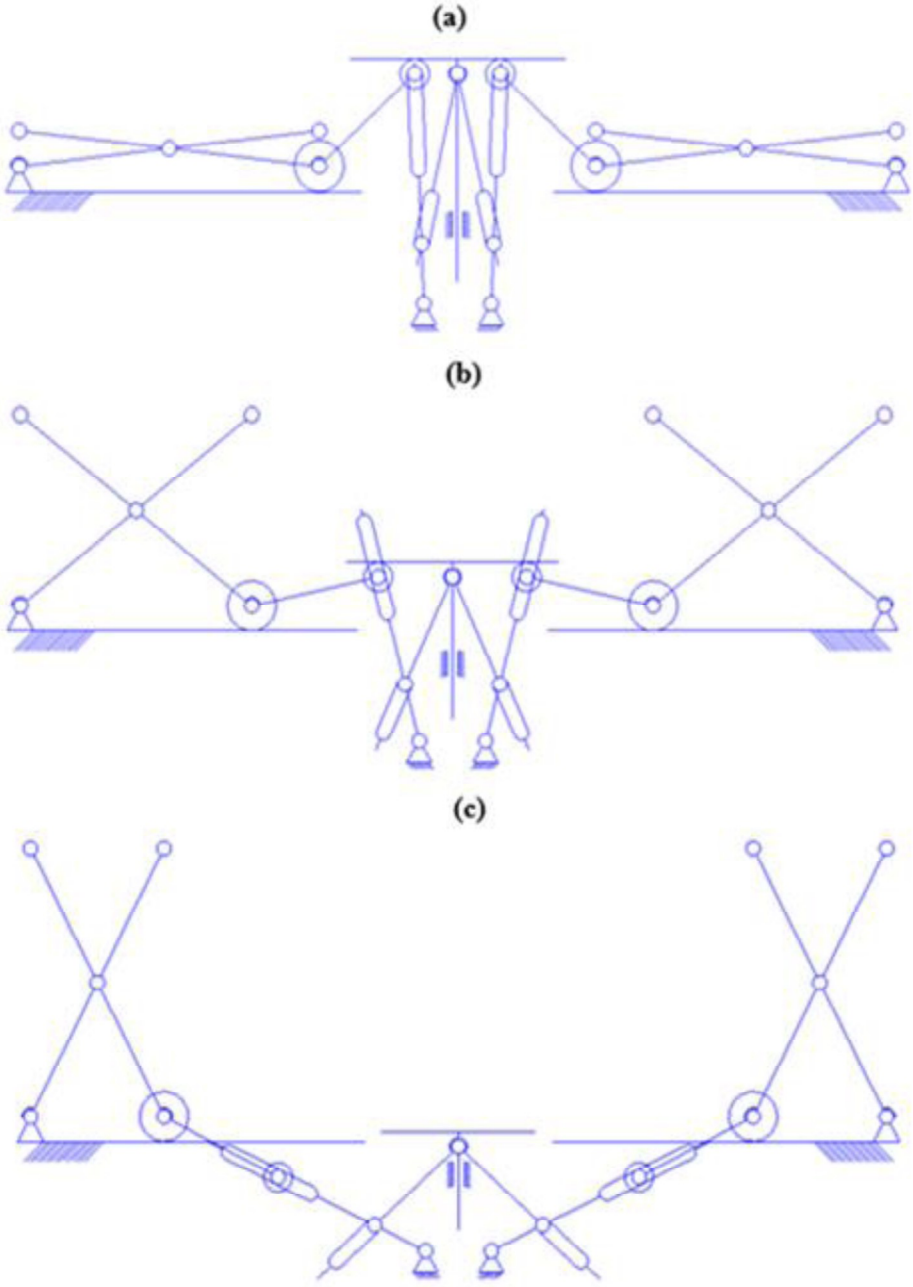
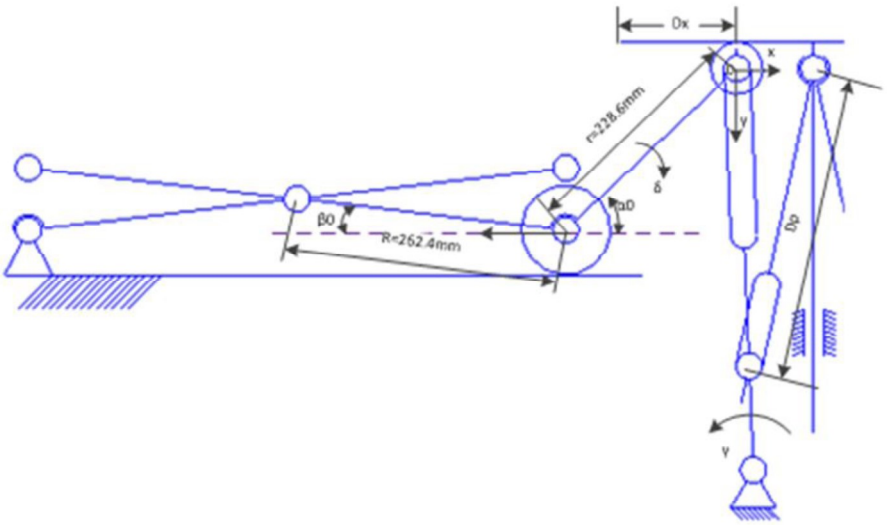
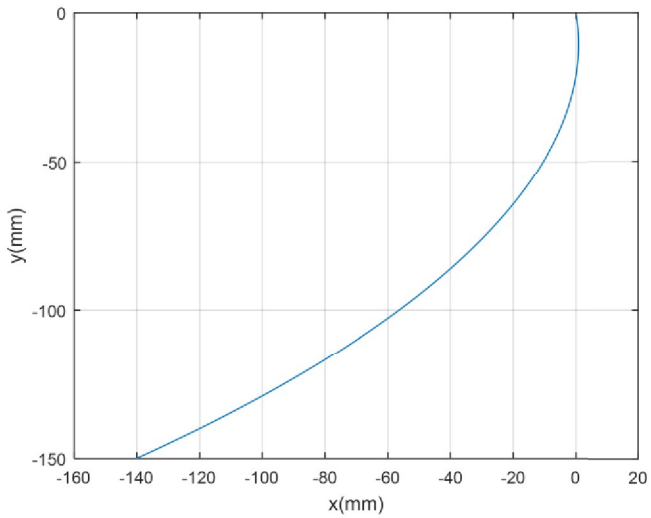


Figure 4.8 (a) Partial sketch, (b) movement locus of the V-type end during lifting.

(a)



(b)



The movement locus of the end of the V-type bar was initially calculated. As shown in Figure 4.8 (a), the initial slope angle of the right scissors bar was 6.6º (represented by β0), the initial angle of the v-type bar was 43.4 º (represented by α0), and the turning angle of the V-type bar was δ.

Two component motions, x and y, were present on the end of the V-type bar that was connected to the T-type bar. The x values were calculated using Equation 4.1 and 4.2， where -150 ⩽ y ⩽ 0 mm, in which the minus sign represents downward motion. The y was a vector with a numerical value equal to the vertical downward displacement of the T-type bar. The motion locus of the end of the V-type bar was based on calculated results and plotted in Figure 4.6 (b).

$$\delta = \alpha_0 - \sin^{-1}(sin\alpha_0 + \frac{y}{r}) \quad (4.1)$$

$$x = 2R(cos\beta_0 - \cos(\beta_0 + \delta)) + r(cos\alpha_0 - \cos(\alpha_0 - \delta)) \quad (4.2)$$

The slope angle of the linking bar was then calculated based on obtained x and y results. A partial sketch for calculating the critical point is shown in Figure 4.9. Rotation angle γ of the linking bar and the distance between point P1 and point P3 were calculated using Equations 4.3 and 4.4, respectively.

$$\gamma = 180 - \frac{180}{\pi}\tan^{-1}(-\frac{P3Y+y}{P2X+x}) \quad (4.3)$$

$$D_p = \sqrt{(P3X - r_0 cos\gamma)^2 + (P3Y + y - r_0 sin\gamma)^2} \quad (4.4)$$

According to Figure 4.6, r_0 was 101.6 mm. Coordinate values of point P3 are listed in Table 4.1; calculation was done by Matlab software. The calculation objective was to find a *Dp* result nearly equal to 203.2 mm; seven sets of results are listed in Table 4.2. Based on data in the brown-shaded column, the critical point occurred when the T-type bar moved downward 129 mm, resulting in a minimum Dx value of 100 mm. Main dimensions of the T-type bar are shown in Figure 4.10. In order to meet design requirements, the design lifting height per scissors unit was 405 mm.

Figure 4.9 Partial sketch for calculating the critical point.

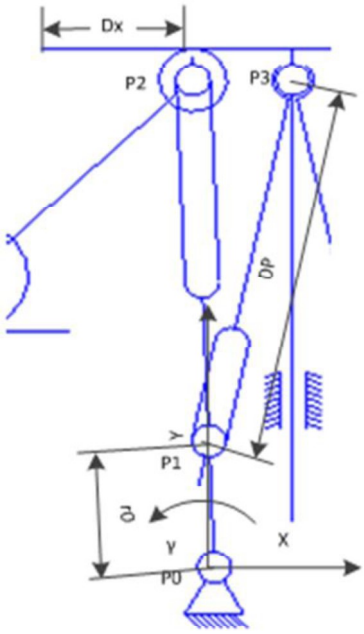


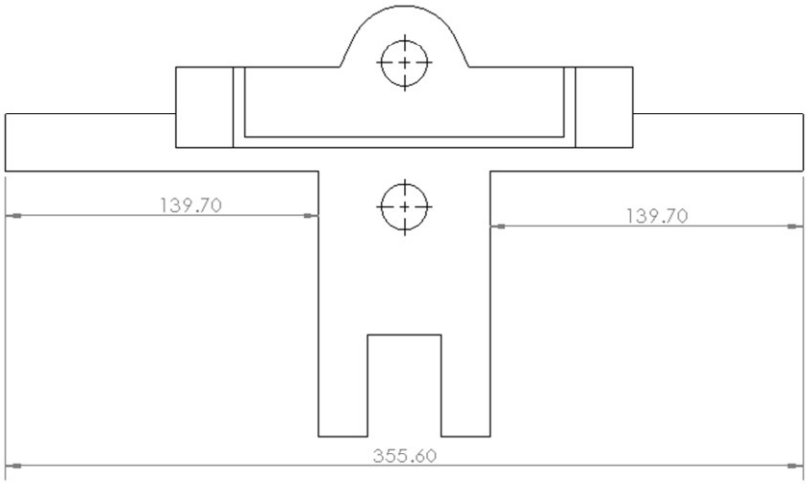
Table 4.1 Initial center position of the four hinges in the coordinate system.

points	X(mm)	Y(mm)
P0	0	0
P1	-4	103.9
P2	-16	390.9
P3	58.4	390.9

Table 4.2 Movement parameters value related to the key components during lifting.

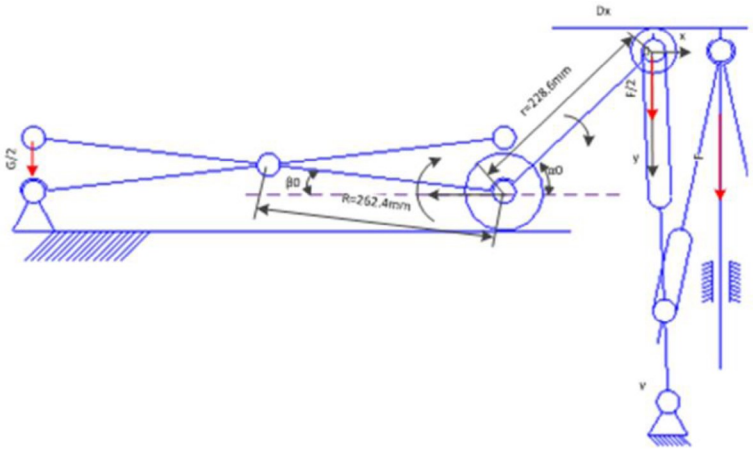
y(mm)	-124	-125	-126	-127	-128	-129	-130
x(mm)	-91.6	-93.2	-94.9	-96.6	-98.3	-100	-101.7
δ(Degree)	31.5	31.8	32	32.3	32.5	32.8	33
γ (Degree)	112	112.3	112.7	113.1	113.5	113.9	114.3
Dp(mm)	209.1	208	206.7	205.6	204.4	203.2	202

Figure 4.10 Main dimensions of the T-type bar.



A lifter's lifting characteristics represent lifting load capability. As mentioned, two lifting phases utilize either the pressing method or the pushing method. Therefore, this study investigated lifting characteristics related to the two phases. As shown in Figure 4.11, only one half of the lifter was analyzed due to its symmetrical structure. The load was assumed to be *G*,

Figure 4.11 Partial sketch for calculating lifting characteristics in the pressing phase.



where the unit was *N* (Newton) and the pulling force applied by the linear actuator was *F*. The

force diagram of the V-type bar shown in Figure 4.11 was derived from the symmetrical structure of the T-type block and Newton's third law of motion. In addition, the lifting velocity was assumed to be constant, resulting in the torque equilibrium Equation (4.5):

$$G R\cos(\beta_0 + \delta) = \frac{F}{2} r\cos(\alpha_0 - \delta) \qquad (4.5)$$

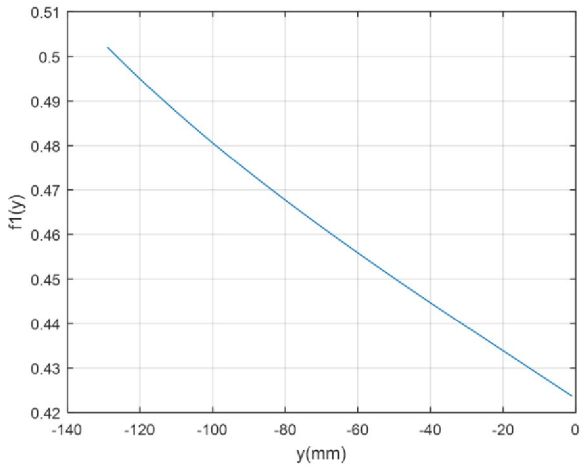
δ in the equation is replaced by the equation (4.1), then we can get the following lifting characteristics equation, where -129$\leqslant$ y $\leqslant$ 0.

$$f_1(y) = \frac{G}{F} = \frac{r\cos(\sin^{-1}(\sin\alpha_0 + \frac{y}{r}))}{2R\cos(\beta_0 + \alpha_0 - \sin^{-1}(sin\alpha_0 + \frac{y}{r}))} \qquad (4.6)$$

The curve of the equation (4.6) is shown in Figure 4.12.

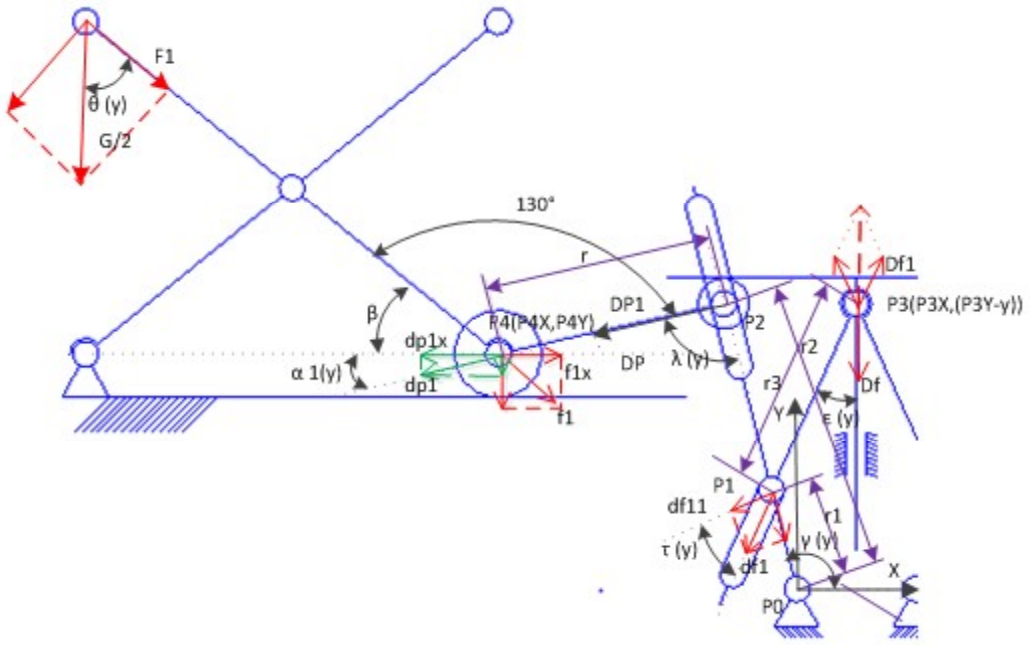
Lifting characteristics in the pushing phase were then calculated; forces on the bars and corresponding component forces are shown in Figure 4.13, in which rotation angles of all related bars are represented by Latin letters. In the pushing phase, component force *dp1x* (green in Figure 4.13) had to be at least equal to component force *f1x* (red in Figure 4.13) in order for the lifter to work:

Figure 4.12 Lifting characteristics curve of the function f_1(y).



$$f_{1x} = \frac{G}{2}\sin(2\,\theta(y)) \qquad (4.7)$$

Figure 4.13 Partial sketch of the lifter for calculating the lifting characteristics in pushing phase.



and,

$$f_{dp1x} = \frac{DF}{2} \times \frac{r_1 \cos(\tau(y)) cos(\alpha_1(y))}{r_2 sin(\lambda(y)) \cos(\varepsilon(y))} \qquad (4.8)$$

Using equation (4.7) and (4.8), the lifting characteristics function was obtained:

$$f_2(y) = \frac{G}{DF} = \frac{r_1 \cos(\tau(y)) cos(\alpha_1(y))}{r_2 \cos(\varepsilon(y) sin(\lambda(y)) \sin(2\,\theta(y))} \qquad (4.9)$$

Where, r_1=101.6mm; r_2=292.6mm. The functions of y,$\tau(y)$, $\alpha_1(y)$, $\varepsilon(y)$, $\lambda(y)$, and $\theta(y)$, were resolved using the law of cosines.

$$\varepsilon(y) = \tan^{-1}\left(\frac{P3X}{P3X+y}\right) + \cos^{-1}\left(\frac{r_3^2 + P3X^2 + (P3Y+y)^2 - r_1^2}{2r_3\sqrt{P3X^2+(P3Y+y)^2}}\right) \qquad (4.10)$$

$$\tau(y) = \cos^{-1}\left(\frac{r_3^2 + r_1^2 - P3X^2 - (P3Y+y)^2}{2r_3 r_1}\right) - \frac{\pi}{2} \qquad (4.11)$$

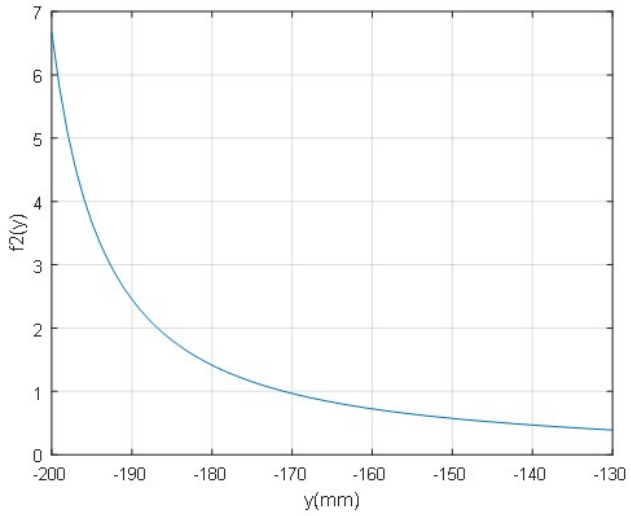
$$\gamma(y) = \tan^{-1}\frac{P3Y+y}{P3X} + \cos^{-1}\left(\frac{r_1^2 + P3X^2 + (P3Y+y)^2 - r_3^2}{2r_1\sqrt{P3X^2+(P3Y+y)^2}}\right) \qquad (4.12)$$

$$\lambda(y) = \cos^{-1}\left(\frac{r_2^2 + r^2 - P4Y^2 - \left(r_2\cos(\gamma(y)) - \sqrt{r^2 - \left(r_2\sin(\gamma(y)) - P4Y\right)^2}\right)^2}{2r_2 r}\right) \quad (4.13)$$

$$\alpha_1(y) = \tan^{-1}\left(\frac{r_2\sin(\gamma(y)) - P4Y}{r_2\cos(\gamma(y)) - r_2\cos(\gamma(y)) + \sqrt{r^2 - \left(r_2\sin(\gamma(y)) - P4Y\right)^2}}\right) \quad (4.14)$$

$$\theta(y) = \alpha_1(y) + \frac{2}{9}\pi \quad (4.15)$$

Figure 4.14 Lifting characteristics curve of the function f_2(y).



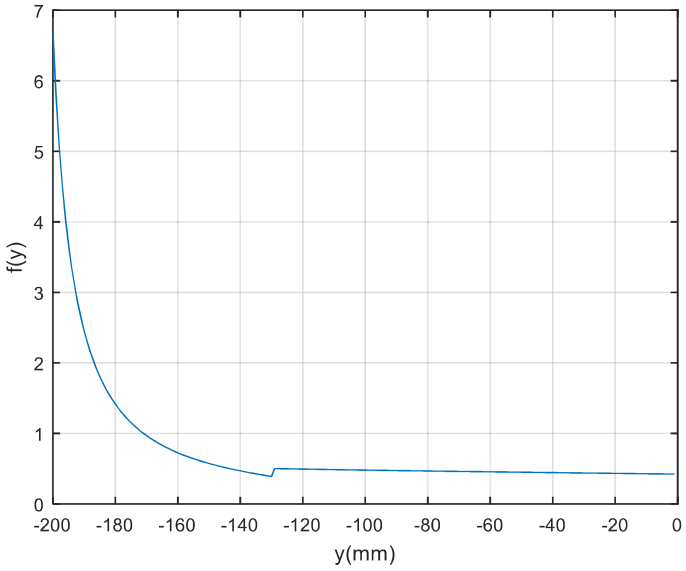
As shown in Figure 4.13, *P3X* and *P4Y* were constant parameters with values 58.4 and -294.6, respectively. Similarly, the values of r_1, r_2, and r_3 were 101.6, 292.6, and 203.3, respectively. The range of *y* was greater than -200 and less than -129. The lifting characteristics curve in the pushing phase (Figure 4.14) was obtained using Matlab software to calculate the equations (4.9) to (4.15). Therefore, the lifting characteristics function was written as

$$f(y) = \begin{cases} \frac{r\cos(\sin^{-1}(\sin\alpha_0 + \frac{y}{r}))}{2R\cos(\beta_0 + \alpha_0 - \sin^{-1}(sin\alpha_0 + \frac{y}{r}))} & (-129 \le y \le 0) \\ \frac{r_1\cos(\tau(y))cos(\alpha_1(y))}{r_2\cos(\varepsilon(y)sin(\lambda(y))\sin(2\,\theta(y))} & (-200 \le y < -129) \end{cases} \quad (4.16)$$

The entire lifting characteristics curve is shown in Figure 4.15, where the minimum value of G/DF is equal to 0.39 where y is equal to -130. So, the Maximum Load (ML) was obtained by the Equation (4.17) where the maximum driving force of the linear actuator is 2500N. The result shows that the design meets the design requirement that the maximum load more than 80kg.

$$ML = \frac{2500\times(G/DF)_{min}}{9.8} = \frac{2500\times0.39}{9.8} = 99.5(Kg) \qquad (4.17)$$

Figure 4.15 Lifting characteristics curve of the function *f(y)*.



4.2.2 Wheel system

As shown in Figure 4.16, the robotic platform in this study included four-wheel drive and steering (i.e., four sets of wheel systems and traction and steering are its main functions). As shown in Figure 4.17, each set of the wheel system included a wheel with hub motor, a wheel fork, a thrust ball bearing, a base board, a radial ball bearing, a steering gear, a driving pinion, an encoder system, and a steering motor. The hub motor of 36V and 500W in the wheel was used to move the robotic platform. The wheel fork was mounted on the base board via a radial ball bearing, and a thrust ball bearing was located on the shaft of the wheel fork under the base board to limit the upward vertical movement of the wheel fork and ensure that the wheel fork rotated

smoothly around the axis of the shaft. The steering motor of 12V and 60W steered the wheel using the gear pair with a reduction of 2:1. The photoelectric encoder system with 1.5º resolution was fixed on the shaft of the steering motor to passively or actively detect the rotating angle of the wheel fork.

Figure 4.16 Robotic platform with four wheel drive and steering.

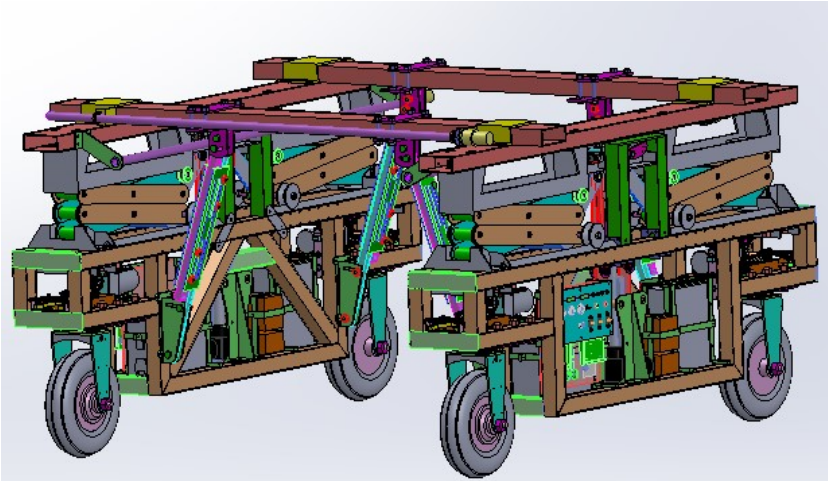
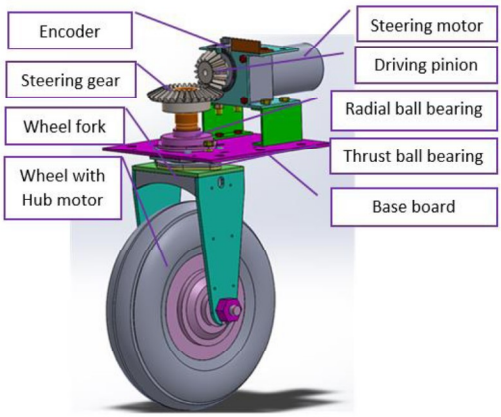


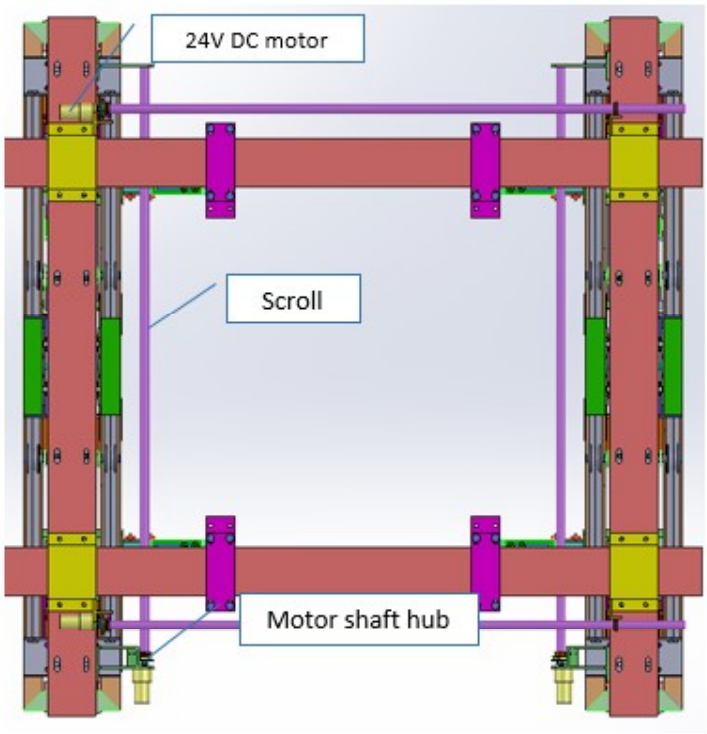
Figure 4.17 Wheel system.



4.2.3 Shade system

A shade system was used to reduce sunlight's impact on sensors that detect crop traits. This study used four sets of shade system, each of which contained a 24V DC motor, hub, and scroll, as shown in Figure 4.18. Black cloth affixed to the scroll could be moved up or down by the DC motor.

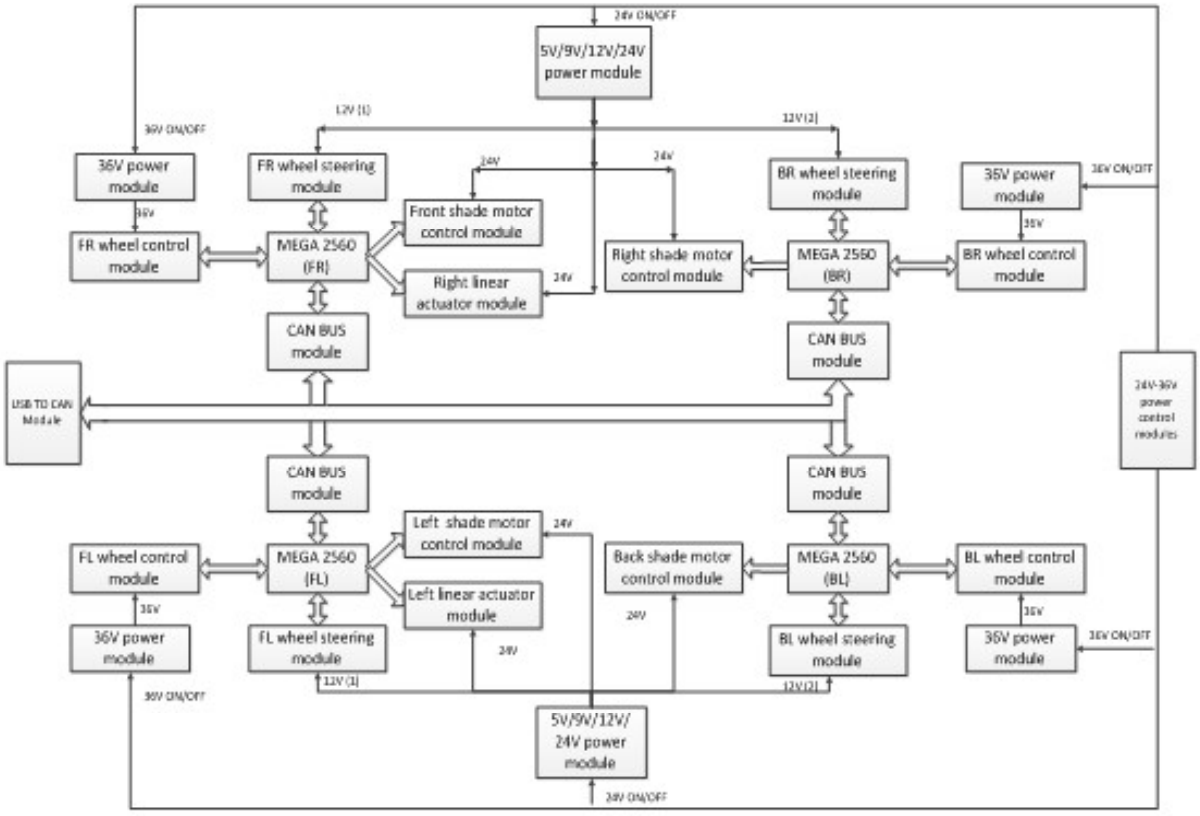
Figure 4.18 Shade system on the Robotic platform.



4.2.4 Robotic control system

A block diagram of a robotic control system is shown in Figure 4.19. Four control subsystems for the four wheel systems, respectively, connected via CAN bus. The power-supplying system, which was powered by batteries, included 5V/9V/12V/24V power modules, 36V power modules, and 24V-36V power control modules, as shown in Figure 4.19. Each control subsystem was composed of a microcontroller module--- Arduino MEGA2560, a CAN bus module, a wheel control module, a wheel steering module, a shade motor control module, and a linear actuator module. The robotic control system was connected to a control system and data acquisition unit installed at a laptop through a USB-to-CAN converter. A schematic of the control system and its wiring diagram are provided in Appendix C and D, and the control system panel is presented in Appendix E.

Figure 4.19 Robotic control system.

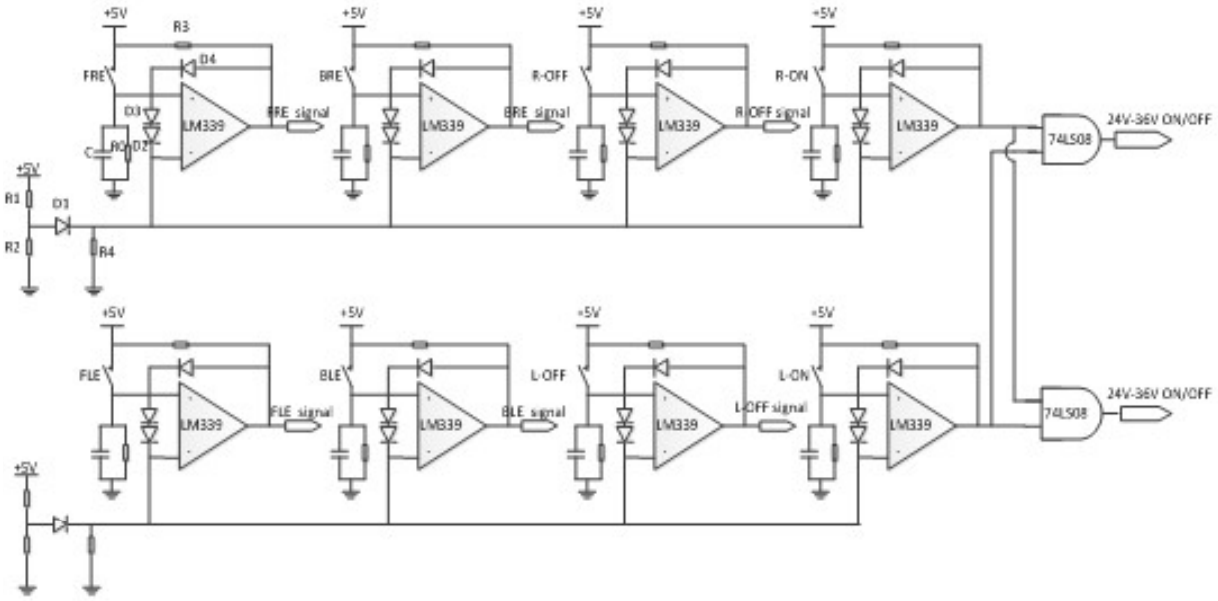


4.2.4.1 Emergency control for power supply module

The power supply module provided voltages to individual components, 36V for hub motors, 24V for linear actuators and DC motors in the shade system, 12V for the steering motor and cooling fans in the driver modules of the hub motors, and 5V for MEGA2560 and other circuits. Two functions were available for the 24V-36V power control module: ON/OFF function and emergency stop function. Four buttons were used in both the right side and left sides of the robotic phenotyper: ON button, OFF button, front emergency button, and back emergency button. The emergency stop function allowed the robotic phenotyper to stop whenever the front emergency button or back emergency button was pressed in order to avoid colliding with

something or people on the robot forward path. The emergency stop function was applied using the interlock circuits shown in Figure 4.20.

Figure 4.20 Interlock circuits for 24V and 36V power control.

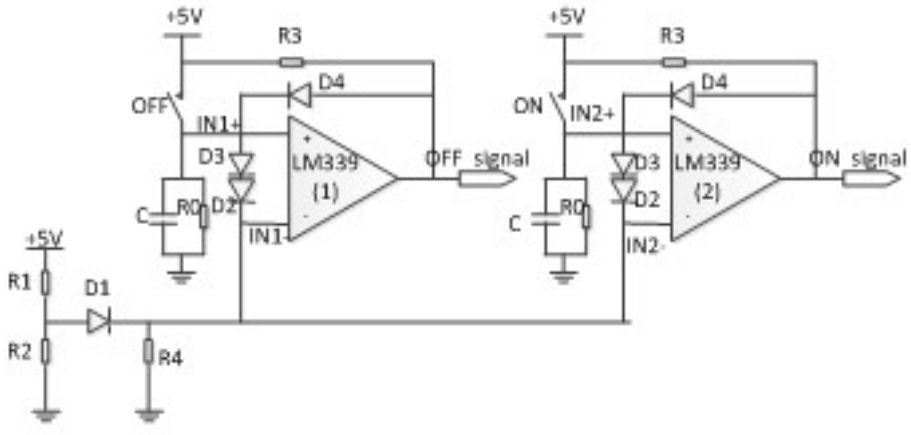


Each side of the robotic phenotyper contained one interlock circuit, and the main component used in the circuit was Quad Comparator—LM339. Every comparison circuit had a self-lock function that relied on the resistor and capacitor (RC) circuit. Four comparison circuits were incorporated into one interlock circuit, and all comparison circuits were connected by reverse input pins. As long as one of the four buttons was pressed, the corresponding output of the comparison circuit was high and the others were low. The buttons are titled according to their functions and fixed positions. For example, the FRE (front right emergency) button was fixed on the end of the front right side of a bracket, and it allowed the emergency stop function. Similarly, the BRE button represented the back right emergency button, and the R-OFF and R-ON buttons represented the OFF and ON buttons, respectively, on the right side of the robotic phenotyper. Outputs of the comparison circuit with ON buttons in both interlock circuits were inputs of the 2-input AND gates, as shown in Figure 4.20. However, if both ON buttons were pushed, then

outputs of the 2-input AND gates were HIGH and the 24V and 36V power sources were on. If anyone of the other six buttons is pushed, the outputs of the 2-input AND gates changed to LOW, and the 24V and 36V power source were off.

A portion of interlock circuit that included two comparison circuits is shown in Figure 4.21. When any button is pushed in an interlock circuit, the output of the comparison circuit with the pushed button changes to HIGH and maintains this state, while the other outputs change to LOW or maintain a low state. In the 2-way interlock circuit in this study, for example, when the ON button was pushed, the input voltage at IN2+ was 5V and the R_0C circuit was charged to 5V. The ON signal was High because $V_{IN2+} > V_{IN2-}$. In order to maintain the HIGH state after releasing the ON button, V_{IN2+} had to exceed V_{IN2-} in the feedback circuit with diode D4. Similarly, when the OFF button was pushed, the OFF signal changed to HIGH. In order to ensure that the ON signal could change to LOW, V_{IN2-} had to exceed V_{IN2+} in the interlock circuit.

Figure 4.21 2-way interlock circuit.



Therefore, in order to meet the above requirements, the values of resistors R0 and R4 and capacitor C in the interlock circuit had to be determined while values of R1, R2, and R3 were known. The voltage dividing circuit comprised of resistors R1 and R2 provided initial input voltage to LM339 and ensured that the comparison circuit had LOW output at any time. Diode

D1 guaranteed unidirectional output of the voltage dividing circuit. The average drop voltage of diodes used in the circuit was V_d = 0.5V, which was obtained using a testing circuit. In the initial state when the power was ON, the input voltage V_0 on the pin IN1- generated by the voltage dividing circuit was higher than zero, and the output of the comparison circuit was LOW. V_0 was obtained via Equation (4.18):

$$V_0 = \frac{(V_{CC} - V_d)/R_3 + 2V_d/R_4}{1/R_0 + 1/R_3 + 1/R_4} - V_d \qquad (4.18)$$

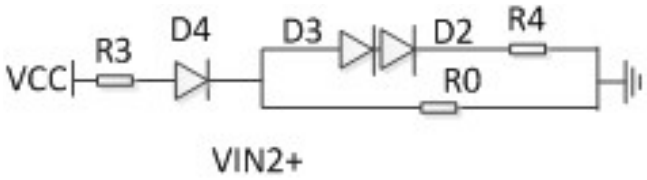
In order to maintain a HIGH status after releasing the ON button, V_{IN2+} had to exceed V_{IN2-} in the feedback circuit with diode D4 (i.e., diodes D_2 and D were inoperable) and at least

$$\frac{V_{cc} - V_d}{R_3 + R_0} \times R_0 > V_0 \qquad (4.19)$$

Where, V_{CC}=5V, R_3=5.1kΩ, and V_d=0.5V. Then

$$R_0 > \frac{8.67R_4 - 117.3}{28R_4 + 275} \quad (R_4 > 13.6k\Omega) \qquad (4.20)$$

Figure 4.22 Simplified circuit for calculating Vin2+.



Similarly, in order to ensure that the ON signal change to LOW while the OFF button was pushed, V_{IN2-} had to exceed V_{IN2+} in the interlock circuit. Immediately before the OFF button was pushed, the V_{IN2+} was obtained using Equation (4.21)**:**

$$V_{IN2+} = \frac{(V_{cc} - V_d)/R_3 + 2V_d/R_4}{1/R_0 + 1/R_3 + 1/R_4} \qquad (4.21)$$

At the moment while the OFF button was pushed, V_{IN2-} was,

$$V_{IN2-} = V_{IN1-} = V_{CC} - 2V_d \qquad (4.22)$$

Therefore, based on the inequality $V_{IN2-} > V_{IN2+}$,

$$R_0 < \frac{R_3R_4(V_{CC}-2V_d)}{V_dR_4-(V_{CC}-4V_d)R_3} = \frac{20R_4}{R_4-30} \quad (R_4 > 30) \qquad (4.23)$$

Based on the inequalities (4.21) and (4.23), the relationship between R0 and R4 was represented with the inequality (4.24), and the value interval is shown in Figure 4.23.

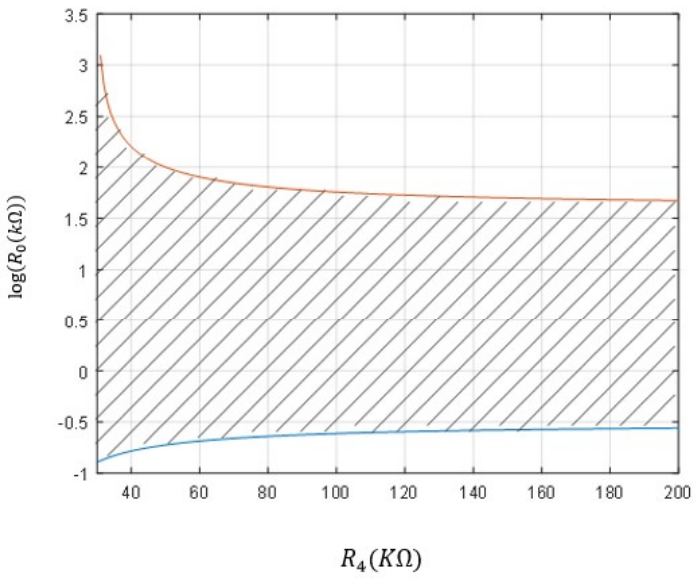
$$\frac{8.67R_4-117.3}{28R_4+275} < R_0 < \frac{20R_4}{R_4-30} \quad (R_4 > 30) \qquad (4.24)$$

The value of resistor R4 was 56kΩ. So, the range of R0 was determined by the inequality (4.24).

$$0.233(\text{k}\Omega) < R_0 < 61.8(k\Omega) \qquad (4.25)$$

The R_0C circuit was essential to the comparison circuit, and its time constant τ determined the anti-interference ability and response speed of the system. Based on inequality (4.25) and experiments, C = 4.7μF, R_0 = 60kΩ, and time constant τ equals 0.282s.

Figure 4.23 Value interval of R0 and R4.



4.2.4.2 Wheel control module

A functional block diagram of the wheel control module is shown in Figure 4.24. The module in the figure includes a hub motor with tire, a hub motor controller, and a 12-bit digital-to-analog converter. The hub motor of 36V and 500W in the wheel caused the robotic platform to move. Eight wires, including three thick wires and five thin wires, were connected to the hub

motor to power the motor and Hall sensors inside the motor and to transfer Hall sensor signals to the hub motor controller. Wire specifications are listed in Table 4.3.

Figure 4.24 Wheel control module.

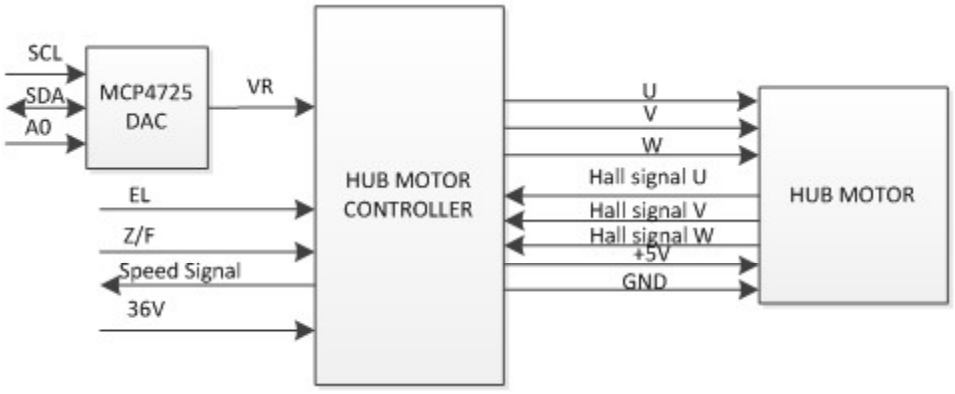


Table 4.3 Wires specification in the hub motor.

Thick wires	Functions	Thin wires	Functions
Yellow wire	36V phase U	Yellow wire	Hall signal U
Green wire	36V phase W	Green wire	Hall signal W
Blue wire	36V phase V	Blue wire	Hall signal V
		Red wire	5V for Hall sensors
		Black wire	GND

Figure 4.25 Wheel with hub motor.

The hub motor controller provided 36V power to the three phases of U, W, and V according to the order determined by the feedback signals from corresponding Hall sensors. As shown in Figure 4.26, the hub motor controller was a Hall-balanced car driver board with a maximum drive capacity of 500W and working voltage ranging from 12V to 36V. The EL pin,

known as the enable control terminal, was connected to a digital pin of the MEGA2560. The motor ran when the digital pin was HIGH, otherwise the motor would stop. The Z/F pin was the reversing control terminal, and VR was the analog input pin used to control wheel speed ranging from 0 to 5V. The signal pin was a pulse output for detecting wheel rotation speed, which was used to achieve constant speed control. Because the resolution of the 8-bit PWM (Pulse Width Modulation) of the MEGA2560 was too low to smoothly adjust the wheel speed, a 12-bit digital-to-analog converter (MCP4725) controlled by an I2C interface was used, as shown in Figure 4.27.

Figure 4.26 Hall balanced car driver board.

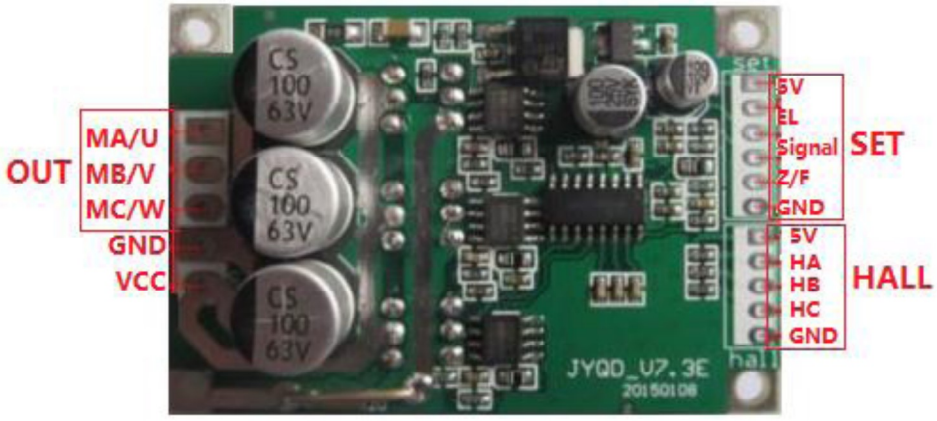


Figure 4.27 MCP4725.

4.2.4.3 Steering control module

As shown in Figure 4.28, the steering control module included a steering motor of 12V and 60W, DC motor drive board, and photoelectric encoder. Module VNH5019, which operated from 5.5 to 24V and delivered a continuous 12A, was used to drive the steering motor. Pin PWM

was the speed control terminal, and pins INA and INB were the direction control ends. A photoelectric encoder was designed to achieve precise control of the steering. As shown in Figure 4.29, three sensors, one encoder disk, and one senor bracket were included in the photoelectric encoder. Sixty narrow gaps were equably distributed along the disk circumference, and two infrared transmission photo sensors (A and B) were fixed on the sensor bracket with 90º phase difference. A window comparator circuit was designed to amplify the output of sensor A or B and reshape its output signal to square wave (Figure 4.30). A reflectance sensor C was used to detect the initial point of the encoder disk. Therefore, the steering control module and MEGA 2560 formed a closed-loop control system, as shown in Figure 4.31.

Figure 4.28 Steering control module.

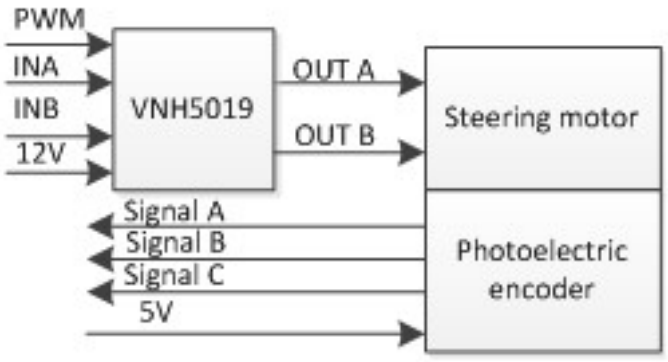


Figure 4.29 Photoelectric encoder fixed on the DC motor.

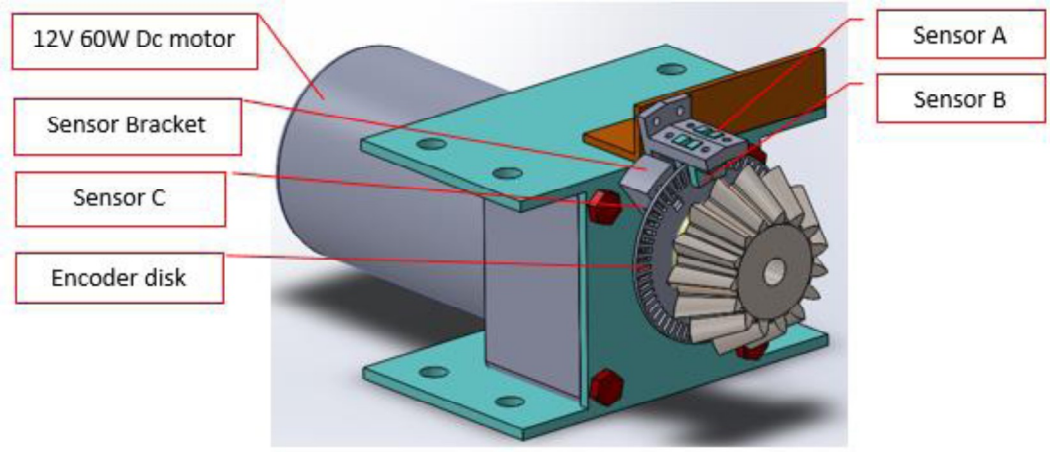


Figure 4.30 Window comparator circuit for sensor A/B.

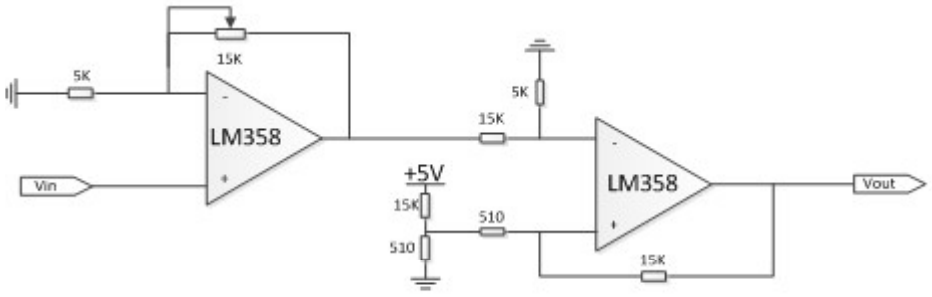
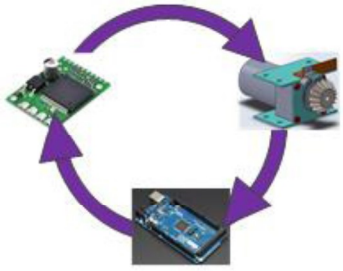


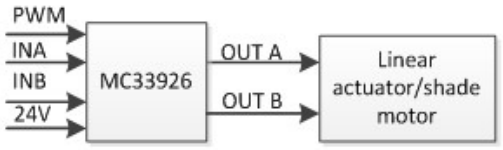
Figure 4.31 Close-loop control of steering system.



4.2.4.4 Linear actuator module and shade motor module

As shown in Figure 4.32, linear actuator or shade motor module is a simple module, which includes a DC motor drive board----MC33926 and a 24V linear actuotor /a 24V shade motor. MC33926 has an operating range of 5-28V and can deliver almost 3A continuously to the linear actuator or shade motor.

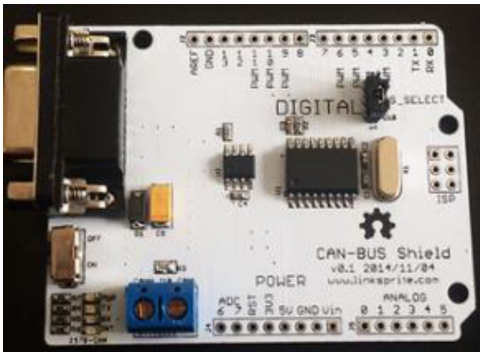
Figure 4.32 Linear actuator/ shade motor module.



4.2.4.5 CAN bus module

As shown in Figure 4.33, the CAN bus module adopted a CAN bus shield for Arduino, using the CAN bus controller of MCP2515, SPI (Serial Peripheral Interface) interface, and MCP2551 transciever. The CAN transfer rate was as high as 1Mb/s.

Figure 4.33 CAN-BUS shield for Arduino.



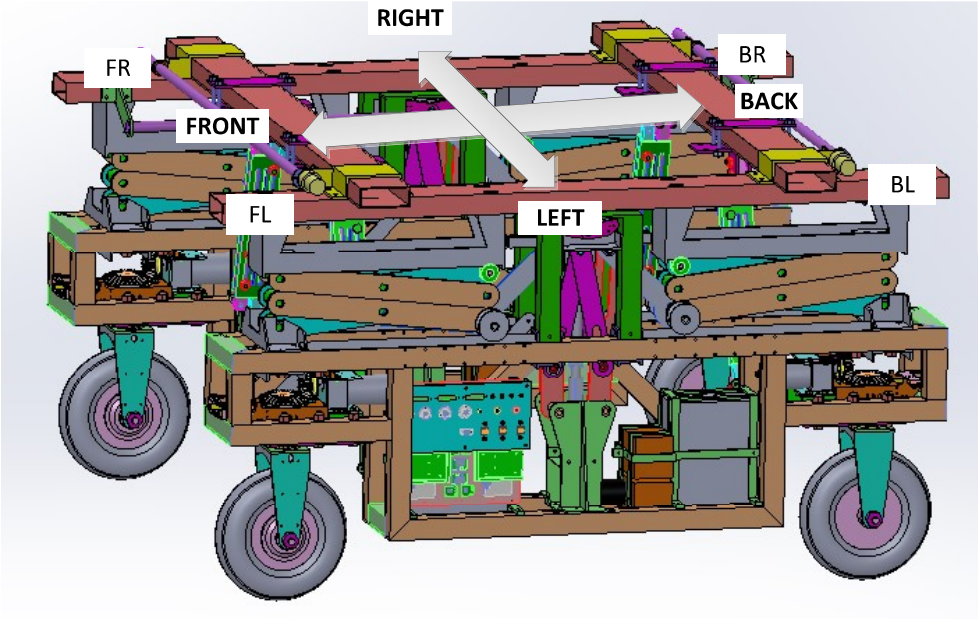
4.2.5 Control system software design

As shown in Figure 4.19, the four control subsystems had identical structures, so they used the same software. The only difference among the systems was their addresses, which were given based on the physical position of the corresponding wheel. For example, address FR indicates that the subsystem operated the wheel and corresponding steering system at the front right corner of the robotic chassis. Subsystem addresses are illustrated in Figure 4.34. The microcontroller board used in the subsystem was Arduino MEGA2560.

According to the block diagram of the robotic control system in Figure 4.19, the software had the following functionos:

- Control the wheel to run at a specified speed;
- Control the wheel fork to turn according to the previously defined angle;
- Detect the deflection angle of the wheel fork while the robot moves to a specific path;
- Manully operate the linear actuator to move the upper part of the chassis upward or downward;
- Manually operate the shade cloth on the scroll to move upward or downward;
- Work according to commands received from CAN bus.

Figure 4.34 Address definitions according to the wheel position in the chassis.



The control program was developed using the platform of Arduino 1.6.5 and then downloaded to a MEGA2560 board to work based on the flowchart shown in Figure 4.35. All actuators in the robot were automatically controlled by CAN bus commands or manually operated by push buttons in the control system. Communication protocols used in the control system are listed in Appendix D. The previously defined registers corresponded to each actuator in the control program. When the CAN bus command was received, the control parameter was initially stored in the corresponding register after address verification. The parameter was then read and executed through the read register and setting parameter modules. Figure 4.36 shows the functions of the parameter setting module: it established parameters of the HUB motors in the wheels and the steering system and then sent the speed and deviation angles of the steering system to the control unit. One changing-function switch was available for the entire control system, and four rounded buttons and three double-position unlocked switches were used for each side of the control subsystem, as shown in Appendix C. When the changing-function switch was off, one of the three double-position unlocked switches determined whether the robot moved

forward or backward; the other two switches operated two of the four shade motors. When the changing-function switch was on, one of the three double-position unlocked switches adjusted the lifter height, and the other two operated the steering system. The green round push button was used to turn the 24V and 36V power sources on, and the other three red buttons were used to turn off the motor's power in an emergency, as in the case when the robot needed to be stopped immediately.

Figure 4.35 Flowchart of the control program.

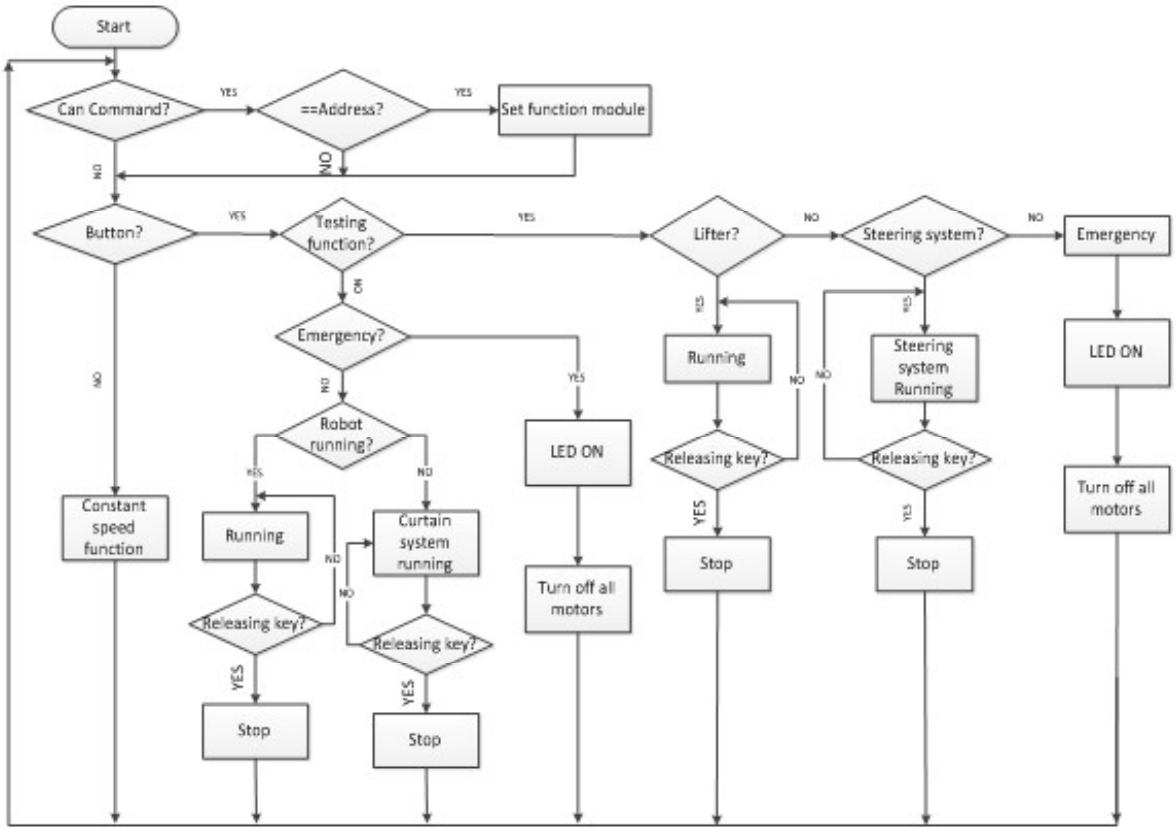
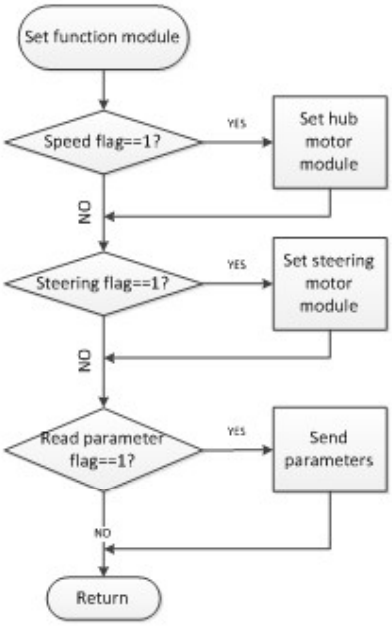
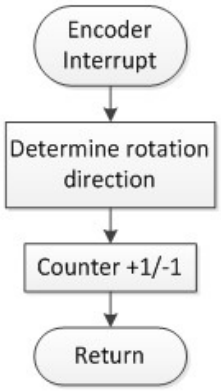


Figure 4.36 Flowchart of set parameter.



As mentioned, a photoelectric encoder (Figure 4.29) was designed to achieve precise control of the steering system. The outputs A and B from the encoder were modulated into the rectangle waves using a window comparator circuit. One of the two rectangle waves was connected to PIN 3 of MEGA 2560, and interrupt 1 was used to count the pulse number in order to accurately and timely obtain the turning angle of the steering system. Figure 4.37 shows a flowchart for the encoder counter.

Figure 4.37 Flowchart for the encoder counter.



4.3 Results

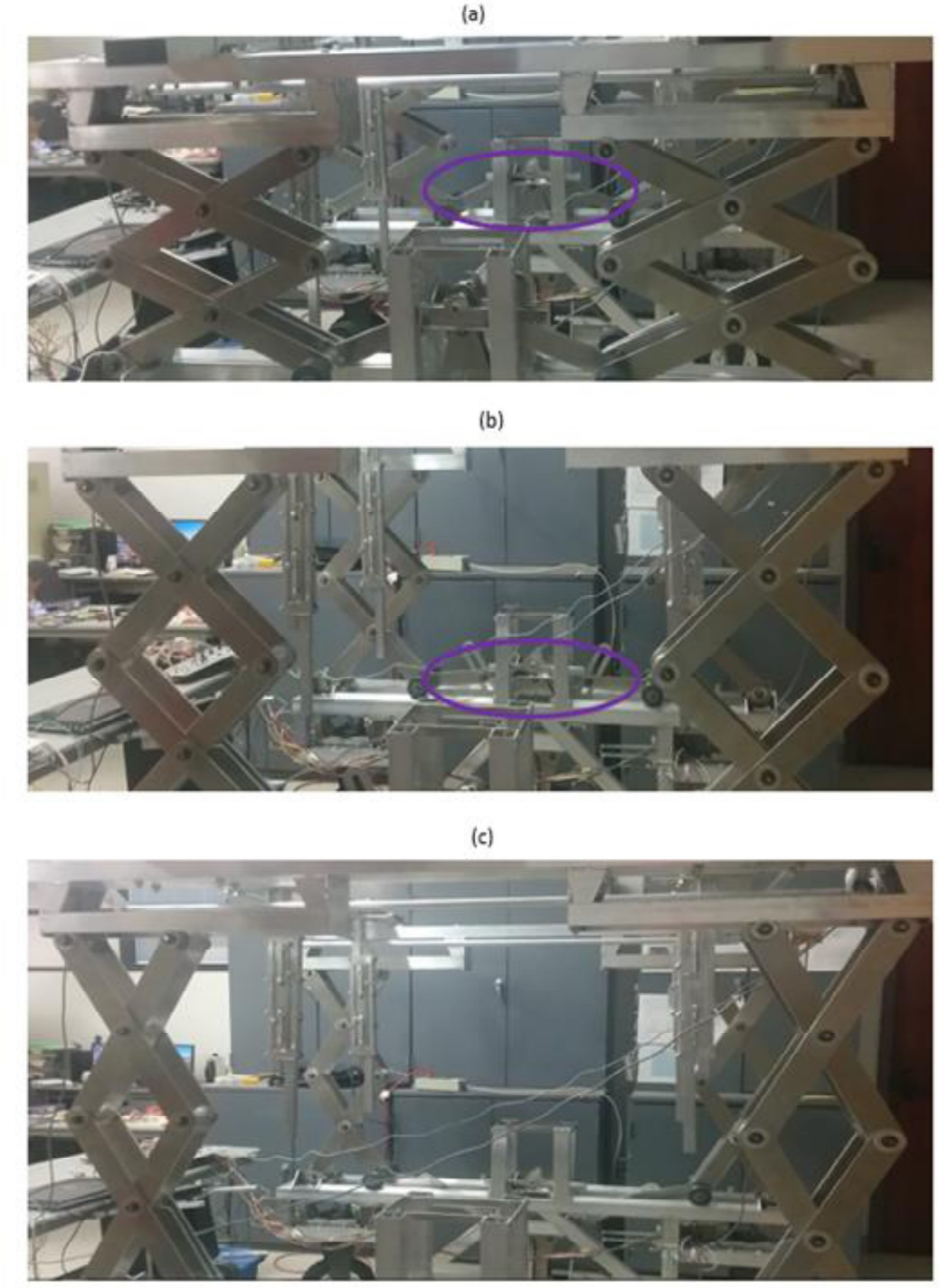
Based on the design, the platform of the robotic phenotyper was constructed as shown in Figure 4.38. Basic functions test showed that the structure and control system met the design requirements.

Figure 4.38 Platform of robotic phenotyper during testing stage.

4.3.1 Scissors-type lifter test

As mentioned, the lifter design requirement demanded that the height of the robotic phenotyper be adjustable betweent 0.9 m and 1.7 m, with two scissors units built into one side of the lifter. In order to elevate the scissors-type lifter to the maximum height, a T&W mechanical structure was designed, manufactured, and assembled with the linear actuator. As described in Section 3.2, a pressing stage (the area within the purple circle in Figure 4.39 (a)) and a pushing stage (the area within the purple circle in Figure 4.39 (b)) were evident during lifting. Because *Dx* exceeded 100 mm according to Equation (4.4) for the dimensions of the T-type bar, a smooth transition from pressing stage to pushing stage occurred during testing; the final height of the chassis was 1.73 m. The final position of the lifter is shown in Figure 4.39 (c).

Figure 4.39 Scissor-type lifter in the testing stage: (a) pressing stage, (b) pushing stage, (c) final position.



4.3.2 Steering system test

The encoder with two grooved infrared transmission photo sensors and one QTR sensor was crucial in the closed-loop control steering system. The QTR reflectance sensor was used to return the wheel fork to zero position and reset the angle register; the other two infrared photo sensors mounted on the bracket with 90º phase difference were used to determine the turning direction and turning angle of the wheel fork. Based on the feedback signal of the encoder, the steering system accurately obtained the specified angle, as shown in Figure 4.40.

Figure 4.40 Steering system in the testing stage.

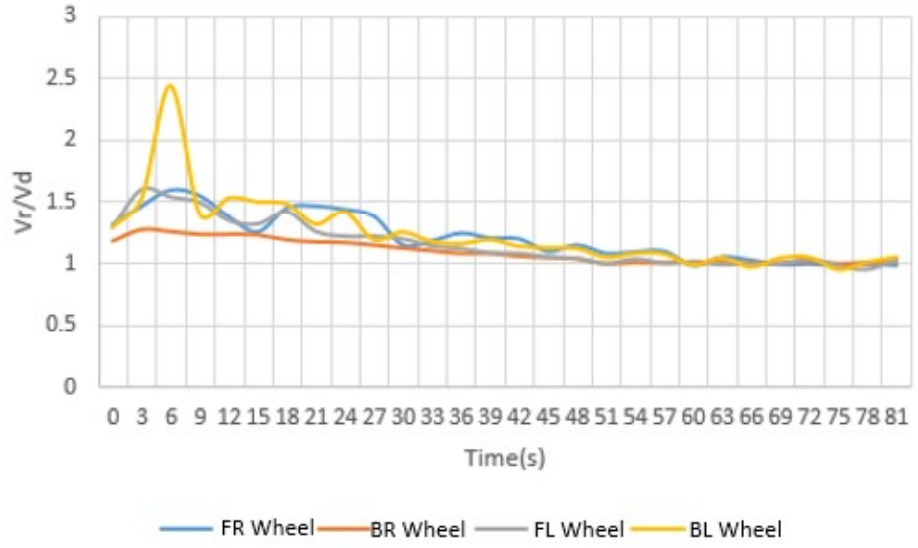
4.3.3 Maintaining constant speed

Constant speed must be maintained during robotic phenotyper operation. As mentioned, wheel rotation speed was controlled by an analog input signal that ranged from 0 to 5 volts. In order to smoothly increase or decrease speed and then maintain a constant speed, a 12-bit digital-to-analog convertor MCP4725 was used in the control system. An experiment for testing the sustaining a constant speed was conducted under a no-load condition. The initial number input to MCP4725 was determined by a proportional equation based on desired speed V_d. If present speed V_r exceeded desired speed V_d, the input value of MCP4725 subtracted eight within a time interval of 3 seconds. The graded number 8 was obtained by test, which was the minimum graded value to get the speed response of the hub motor. If present speed V_r was less than desired

speed V_d,, the input number of MCP4725 continuously added 8 with a time interval of 3 seconds. Constant speed characteristics ($^{V_r}/_{V_d}$) of the four wheels were calculated based on the test data.

As shown in Figure 4.41, the speed became nearly stable after 63 seconds. The maximum deviation error, which was 5.5%, occurred at the wheel at the back left corner of the robot; the other wheels showed deviations of less than 3.3%.

Figure 4.41 Constant speed trait without load.



4.4 Discussion

Up to the time this dissertation is written, the mechanical structure of the robotic phenotyper was completed, but the networking and control parts are not yet completed. Thus, only limited tests were conducted to exam the performance of the phenotyper against the design specifications. This included 1) chassis adjustment, 2) individual and simultaneous control of the drive wheels, 3) control of the shade, and 4) emergemncy stop. Tests on the chassis showed that the mechanical structure of the chassis is flexible and the height and width are adjustable within the specified ranges of 0.9-1.7m and 0.5-1.5m, respectively. However, the process of adjusting

the chassis height was semi-automatic, which required an experienced technical staff to accomplish.

The four drive wheels have been tested individually and simultaneously. When four drive wheels were powered simultaneously, the robot moved as expected. However, no extensive tests have been done to observe the robot motions. A potential shortcoming of the design is that the diameter of the drive wheels may be too small, which may cause stall of the robot in field, especially on uneven or wet surfaces.

The total cost of the mechanical parts, metal materials, and electronical components is about $11,000 (Table 4.4).

Table 4.4 Total components cost.

Components	Cost
Wheel systems	$4,214
Brackets	$1,800
Lifters	$2,600
Shading sytems	$1,300
Control systems	$1,100
Total	$11,014

4.5 Conclusion

In general, the platform of the robotic phenotyper was designed and manufactured using a modular design approach, and the results of functional tests showed that the structure and control system met originally proposed specifications. The chassis with a scissors-type lifter was the innovative feature of the design, and the state-of-the-art T&W-type lifter design effectively lifted the robot height from 0.9m to 1.7m. The design procedure for the lifting mechanism established in this study can be used for systems with different dimensions and loads. The

telescope-type bars used in the chassis stabilized the chassis while the robot was in operation. Design of an automatic telescope-type system would ideally allow the chassis height of the robot to be adjusted automatically.

4.6 Future work

The design and manufacture of the robotic phenotyper and its control systems were complted, and some basic function performances such as steering, load lifting, and movement features were tested in the paper. For the applictions in practice, there still are many research tasks to do in the future, which are:

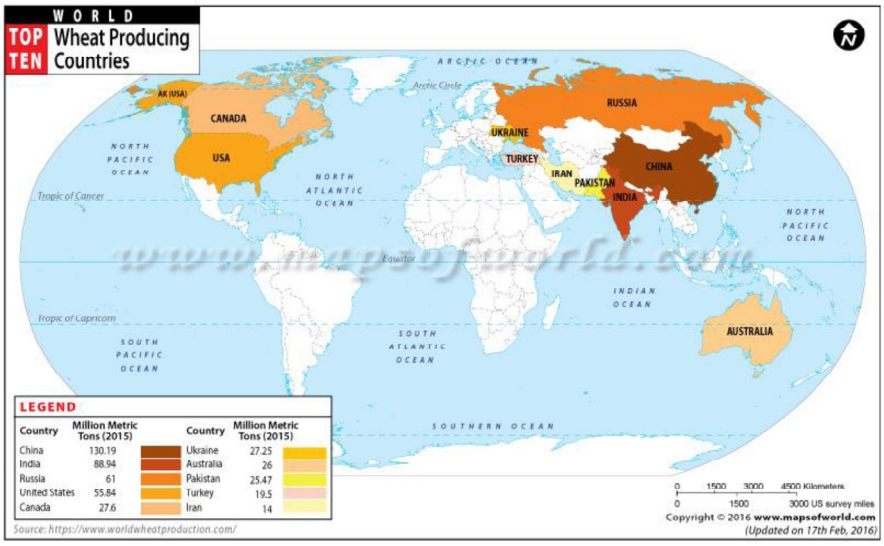
1) Navigation system design based on GPS and machine vision technology;
2) Remote control station design (remote distance >1 Mile);
3) Multi-sensor system integration and data acquisition system design;
4) Soil box design for testing robot movement;
5) Design of the algorithm of robot moving control and test;
6) Comprehensive test including navigation system, data acquisition system, and remote control station;
7) Robotic phenotyper test in the field.

Chapter 5 - IMAGE PROCESSING TO DETECT WHEAT GROWTH STATUS AND COUNT WHEAT HEADS

5.1 Introduction

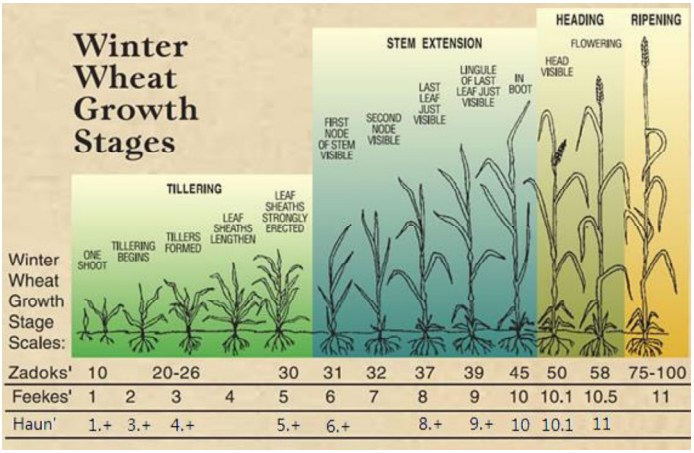
Wheat (*Triticum aestivium*) originated in the Levant region in the eastern Mediterranean but is currently cultivated worldwide, as shown in Figure 5.1. World production of wheat was 734.24 million tons in 2015, making it the second-most produced cereal after corn (966.37 million tons) (USDA, 2016). Because it has a higher protein content than other major cereals, corn, or rice, wheat has become the dominating source of vegetable protein in human food. Two types of wheat are grown throughout the world: winter and spring wheat. The primary difference between the wheat types is that winter wheat requires a period of cold temperature to begin reproduction, but spring wheat does not. Development from germination to maturity takes 280-359 days for winter wheat and 120-145 days for spring wheat. The seeding rate ranges from 20 plants/ft^2 to 30 plants/ft^2 (AGRI-FACTS, 2007).

Figure 5.1 Top ten wheat-producing countries (from http://www.mapsofworld.com).



The wheat life cycle consists of five stages: the seedling and tillering stage, the jointing stage, the booting stage, the heading and flowering stage, and the maturity stage. The seedling stage spans the germination of the seed in the soil to the appearance of the first leaf; tillering occurs before the winter beginning. The jointing stage begins when the stalk grows the second node, and the wheat shows maximum green. The booting stage starts with the appearance of the wheat head and ends with the emergence of awns. The heading and flowering stage includes the start of flowering, followed by pollination and fertilization. The green color of the wheat begins to decrease in the maturity stage, demonstrating two substages: the milk stage and the ripening stage. A very moist kernel with 30% water forms during the milk stage, and then the kernel loses all moisture and changes to a yellow color during the ripening stage (Kakran et al., 2012).

Figure 5.2 Winter wheat growth stages by Zadoks', Feekes', and Haun' scales (Jackson and Williams, 2006).



Numerical scales such as Zadoks, Feekes, and Haun scales have been developed to describe the growth stages of wheat, as shown in Figure 5.2 (Jackson and Williams, 2006). Based on wheat growth stages, farmers can optimize yield using fertilizer, irrigation, herbicide, insecticide, or fungicide, and breeders can compare wheat strains and estimate wheat performances to select the optimal strain. Wheat growth stages traditionally have been measured manually and determined by people experiences. Muhammad et al. (2005) used emergence rate

as one of the four top parameters to assess the growth of eight wheat cultivars under saline conditions. With the help of a meter quadrate, they randomly threw the meter quadrate into the net plot area and then manually counted the number of plants enclosed by the quadrate (Muhammad et al., 2005).

5.1.1 Wheat traits detection using image processing

The development of sensor technology and computer science has revealed the potential of digital image processing for automatic detection of wheat development. Wheat parameters such as green coverage, leaves, stems, flowers, and wheat heads are all research objects in image processing (Lukina et al., 1999; Cointault et al., 2008; Kakran and Mahajan, 2012; Jiang et al., 2012). The green pigment of a wheat crop is closely related to its age: Maximum green pigment is evident during early development stages and minimum green pigment occurs in a mature wheat crop. Kakran and Mahajan proposed that the wheat crop age could be determined by measuring the green color in the wheat images (Kakran and Mahajan, 2012). Cointault et al. (2008) proposed a color-texture analysis method based on RGB images taken in the field in order to count wheat heads and predict wheat yield. A total of 20 images were tested, and wheat heads were counted using image segmentation, classification, and morphological analysis, resulting in recognition rates between 73% and 85% (Cointault et al., 2008). Both the method for determining wheat age and the algorithm for counting the wheat heads have certain limitations. Kakran and Mahajan (2012) developed a method to estimate wheat age based on image processing on three sets of images taken during the early and late stages without considering the entire growth season. The algorithm of counting wheat heads developed by Cointault et al. (2008) was built on wheat imges of only one variety taken under specified light conditions. The influence of weather condition was not considered.

5.1.2 Image analysis algorithms for plant phenotyping

A color space is a method to specify, create, and visualize colors, and a color is usually expressed using three coordinates, or parameters, that denote the position of the color in the color space (Ford and Roberts, 1998). Commonly used color spaces include RGB (red, green, blue), CMY(K) (cyan, magenta, yellow, black), HSL (hue, saturation, lightness), HSB (hue, saturation, brightness), the Ycc color model, the LAB color model, and the YUV color model. Certain color spaces are usually more beneficial for certain applications, and the use of a specific color space depends on the application object (Ford and Roberts, 1998). For example, Zeng et al. (2015) proposed a method to extract the Leaf Area Index (LAI) in LAB color space from images taken under sunny weather conditions. Many literatures have also reported development of a variety of image processing algorithms to extract plant traits such as LAI, PVC, stems, flowers, and wheat heads in RGB color space (Lukina et al., 1999; Cointault et al., 2008; Kakran and Mahajan, 2012; Jiang et al., 2012). Other literature has reported use of average lightness obtained in LAB color space to increase uniformity of image lightness and then extract the objects in RGB color space (Zhu and Zhou, 2015).

Vegetation coverage, expressed in percentage, refers to the ratio of a vertical projection area of regional plants to its geographic area. Lukina et al. (1999) used the software of Micrografx Picture Publisher® version 7.0 to estimate the Percent Vegetation Coverage (PVC) of winter wheat. Zhao et al. (2009) presented a method based on particle swarm optimization and the K-means clustering methods to estimate the PVC of various types of agricultural images. The results showed that the method could efficiently achieve a high classification accuracy of approximately 90%. However, there was no literature reporting practical methods to estimate the PVC in wheat fields throughout the entire growth season.

Researches on wheat head counting based on image processing has focused on the counting accuracy. The main difficulty has been counting densely distributed heads in the images. Several methods, including the splitting and merging method, the wheat head shape index (WSI) method, and the joint point counting (JPC) method (Germain et al., 1995; Frédéric et al., 2012; Liu et al., 2014), have been reported. The splitting and merging method could obtain the highest counting accuracy, but was time-consuming beacause the method involved spliting connected lobes of wheat heads and then restructuring individual wheat heads again(Germain et al., 1995). The counting accuracy of the WSI method was quite low because only the relationship between the areas of the head surface and the surface of its convex hull was considered in this method. For the JPC method, because the number of wheat heads was estimated based on the joint points in the image following a skeleton operation, high-quality, high-resolution images with smooth outlines were required. This type of image was difficult to obtain without human assistance.

5.1.3 Objectives

Based on past studies found in literature, the objectives of this study were:

- Developing image processing algorithms for estimating percent vegetation coverage (PVC) of wheat;
- Evaluating wheat growth status based on PVC data;
- Counting wheat heads based on a texture-color analysis method;
- Studying the effect of weather conditions on the accuracy in counting wheat head counting.

5.2 Material and Methods

5.2.1 Material

The study for detecting wheat growth status and counting wheat heads was conducted from March to June, 2015, on a sowed wheat field in Manhattan, Kansas (latitude: 39.23247°, longitude:96.58081°), as shown in Figure 5.3. Seven areas were randomly selected for study and labeled with wooden bars to ensure that images were always taken at the same positions.

Figure 5.3 Experimental location

As shown in Figure 5.4, a digital camera was placed in the nadir direction on the horizontal bar of a frame, at a height of 1.2 m from the ground. The field of view was 0.6 m^2, covering three rows of wheat. The digital camera is a Canon brand, model number EOS Rebel T5i, with images of 5184X3456 pixels, giving an image resolution of about 50 pixels/mm^2. Pictures were taken once every three days, except in rainy weather, and three pictures were taken at every position. All pictures were downloaded and renamed in the previously defined format. For example, a picture entitled FWH20150311-1 indicated that it was the first picture taken on March 11, 2015, at the selected area labeled FWH. The period of taking picture was from 11 Mar to 10 June 2015. Thus, totally 122 pictures were taken for each selected area. All images were

processed using MATLAB R2015a on an Intel(R) Core(TM) i7-2600 computer with a frequency of 3.4 GHz and 4 G RAM.

Figure 5.4 Camera and mounting frame

5.2.2 Methodology

5.2.2.1 Extraction of vegetation coverage

In 1999 Lukina et al. first predicted Percent Vegetation Coverage (PVC) using digital image processing. Their study results showed that the estimated PVC was highly correlated with NDVI measurement for wheat canopy, and correlation was higher than 0.8. Kakran and Mahajan (2012) sought to determine wheat crop age by measuring the green color in wheat images obtained from digital image processing, thereby demonstrating that PVC not only represents wheat canopy coverage status in the corresponding area, but it also indicates wheat change with time, or status of wheat development. Consequently, this section primarily addresses the application of image processing to estimate PVC and analyze wheat development.Image segmentation, a mid-level processing technique, was the primary research focus in this study. Following image segmentation, the PVC was attained using its definition-----the sum of green pixels divided by total pixels in the image.

5.2.2.1.1 Method for determining the applicable green threshold (Selecting the images taken from the area marked Winter to demonstrate the calculation process)

This study utilized the OTSU method named after Nobuyuki Otsu, the k-means clustering method, and the maximum entropy (ME) method combined with three kinds of color indices, the RGB-based color indices, the LAB-based color indices, and RGB&LAB-based color indices, to estimate the applicable thresholds for extracting the green pixels in the images. The OTSU method, proposed by Nobuyuki Otsu in 1979, maximizes the weighted sum of between-class variances of foreground and background pixels to estimate an optimum threshold. This method has been widely used for image segmentation (Huang et al., 2012; Khan and Ravi, 2013). The k-means algorithm divides image pixels into *K* classes with *K*-1 thresholds by minimizing the total within-segment variance. The computing process searches for each cluster centroid, or the shortest total distance of data points to the corresponding centroid (Khan and Ravi, 2013). Because entropy represents uniformity of gray level pixel distribution (i.e., the more uniform the pixel gray level distribution, the greater the entropy), the ME method solves for the optimal threshold at which the sums of the two classes entropies have the maximum values respectively (Qi, 2014). The RGB-based color indices were calculated in the RGB color space by Equations 5.1, 5.2, and 5.3:

$$EG = 2G - R - B \qquad (5.1)$$

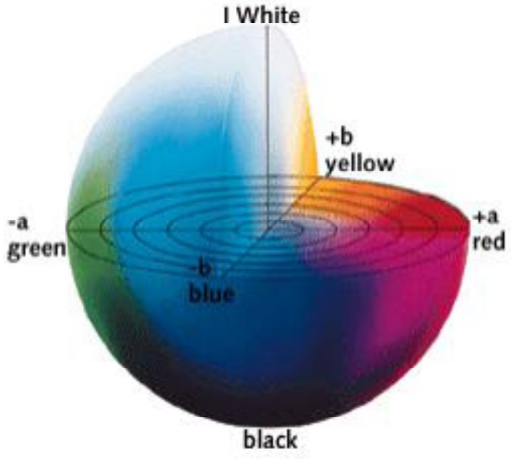
$$NDI_{gr} = \frac{G-R}{G+R} \qquad (5.2)$$

$$NDI_{gb} = \frac{G-B}{G+B} \qquad (5.3)$$

where *R*, *G*, and *B* are color data of red, green, and blue in the image, respectively; NDI_{gr} and NDI_{gb} are normalized difference indexes using green and red and green and blue, respectively (Stajnko et al., 2004; Zhao et al., 2009). Commission Internationale Ed I'eclairage (CIE) defines

LAB color space (Figure 5.5) as L for luminance, A for color on the green-magenta axis, and B for color on the blue-yellow axis. Therefore, the LAB-based color indices were the A component and then extracts the green components in the LAB color space.

Figure 5.5 CIELAB color space (from www.linocolor.com).



The definition of Percent Vegetation Coverage (PVC) is shown in Equation 5.4:

$$PVC = 100 \times \frac{Num_{GP}}{Num_{TP}}(\%) \qquad (5.4)$$

where, Num_{GP} =The number of green pixels in an image;

Num_{TP} =The total pixel number in the image.

Thus, the most important process for estimating the PVC is to extract the green pixels from the image using an image segmentation method, as shown in Figure 5.6.

Figure 5.6 Green pixels in an image

The critical step was to select a proper threshold for green pixels. To do so, the two color spaces (RGB and LAB) and three segmentation methods were combined to form six combinations, as shown in Table 5.1. Results of these tests were then compared.

Table 5.1 Acronyms of the six methods.

Color indices\Methods	K-means method	ME method	OTSU method
RGB	KRGB	MRGB	ORGB
LAB	KLAB	MLAB	OLAB

Totally 36 of the 122 images, taken from 11 Mar to 10 June 2015 in the "Winter" area, were randomly selected for the study. Figure 5.7 (a) and (b), obtained by ORGB method, showed that the thresholds changed with time. And the three images extracted from image WHF-20150317-1, image WHF-20150410-1, and image WHF-20150608-1 using ORGB method were shown in Figure 5.8 (a), (b), and (c), respectively. Figure 5.8(b) gave the best result because almost no pixles representing soil background were extracted.

Figure 5.7 (a) Thresholds vs Time in LAB color space, (b) Thresholds vs Time in RGB color space

(a) (b)

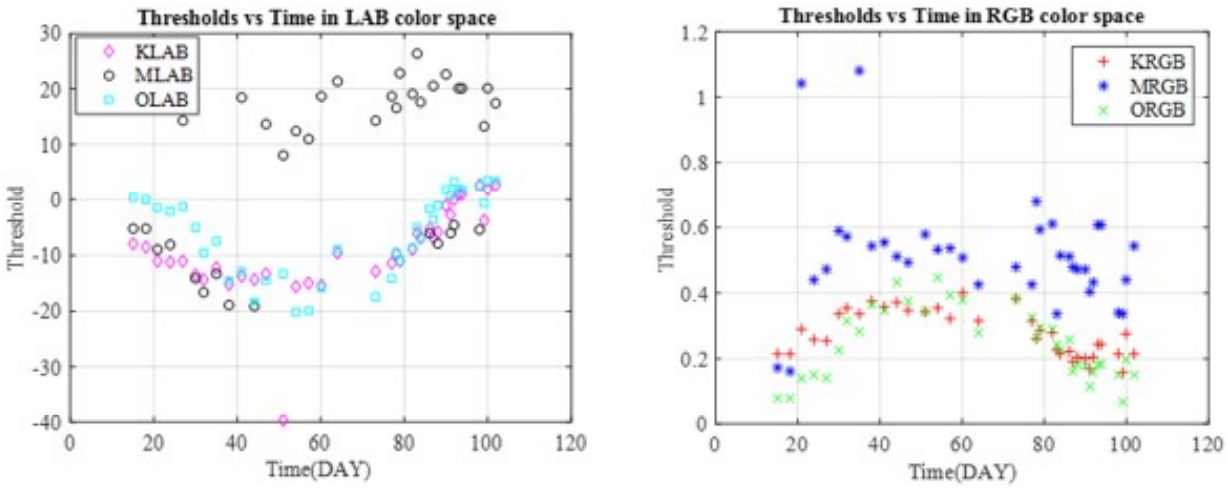


Figure 5.8 (a) Image extracted from image WHF-20150317-1, (b) Image extracted from image WHF-20150410-1, (C) Image extracted from image WHF-20150608-1.

(a) (b) (c)

The applicable threshold was determined based on the quality of image segmentation. A new criterion R_{noise} was defined by Equation (5.5):

$$R_{noise}(N \leq n) = 100 \times \frac{Num_n}{Num_T}(\%) \qquad (5.5)$$

where, Num_n =The number of pixels representing soil background;

Num_T =The total number of pixels in the image.

The first 7 images were selected to be processed using the six methods, and the R_{noise} values were calculated by Equation (5.5). The results are listed in Table 5.2.

Table 5.2 $R_{noise}(N \leq 20)$ values obtained using 6 methods

Images	KRGB R_{noise}	MRGB R_{noise}	ORGB R_{noise}	KLAB R_{noise}	MLAB R_{noise}	OLAB R_{noise}
WHF-20150314-1	2.3%	3.2%	16.4%	1.2%	3.5%	13.8%
WHF-20150317-1	2.4%	3.5%	17.6%	1.1%	3.1%	15.5%
WHF-20150320-1	4.9%	100%	12%	3.8%	5.6%	10.2%
WHF-20150323-1	0.9%	4.0%	2.7%	0.2%	0.3%	6.8%
WHF-20150326-1	0.6%	5.6%	1.3%	0.4%	0	5.6%
WHF-20150329-1	1.2%	8.9%	1.7%	1.2%	1.1%	5.3%
WHF-20150301-1	0.5%	5%	0.4%	0.4%	0.5%	0.8%

In Table 5.2, the 100% (red color) value was obtained using the method MRGB from image WHF-20150320-1. Three sets of images, including the original images and their extracted images, were shown in Figure 5.9. The main difference between the three original images was that image WHF-20150320-1 was brighter than the other two. That was why only some small green groups (Green pixel number ≤ 20) was extracted from the image WHF-20150320-1. The result showed that the Maximum Entropy method was more sensitive to sunlight than the K-means and OTSU method. From Table 5.2, we also found that the R_{noise} values of the first three images estimated using either the ORGB method or the OLAB method were greater than the others and these values were regularly decreasing when the wheat matured. In the same way, the image processing results of three sets of images were shown in Figure 5.10 (a), (b), and (c). The result indicated that the OTSU method is more sensitive to wheat growth.

Figure 5.9 Image processing results with the MRGB method for (a) images WHF-20150317-1, (b) images WHF-20150320, and (C) image.

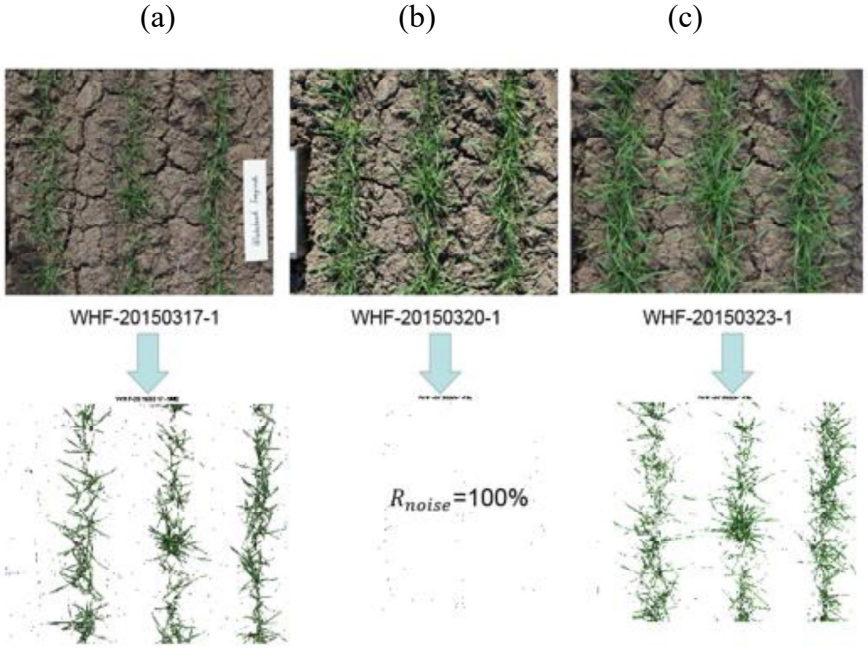
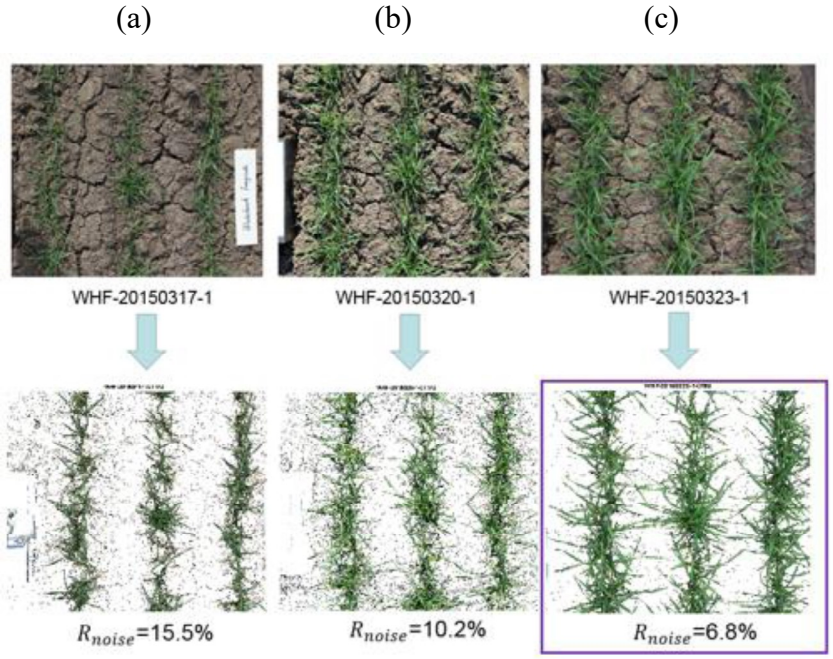


Figure 5.10 Image processing results with the OLAB method for (a) images WHF-20150317-1, (b) images WHF-20150320, and (C) image about WHF-20150323-1.

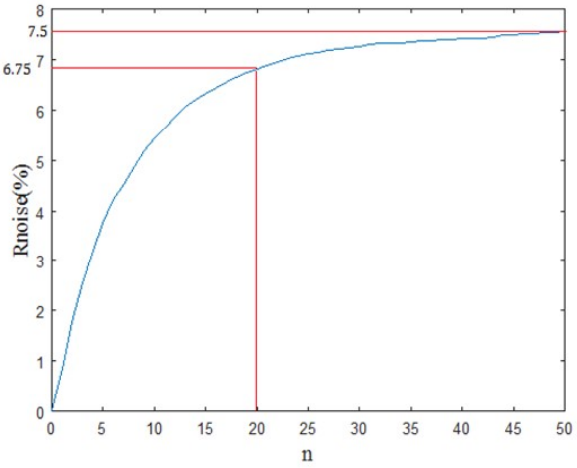


The extracted groups in the image marked with a purple rectangle in Figure 5.10 (c) were analyzed. The R_{noise} curve ($0 \leq n \leq 50$) is shown in Figure 5.11. The value of R_{noise} ($n \leq 20$) was equal to 6.75%, which represented 90% of the value of R_{noise} ($n \leq 50$). The result showed that most pixles in the groups, with pixel number less than or equal to 20, were from the soil background. Therefore, a criterion for selecting the applicable threshold was found to be

$$R_{noise}\ (n \leq 20) < 1\% \qquad (5.6)$$

The applicable threshold was obtained when the $R_{noise}(n \leq 20)$ of the extracted image was less than 1% during image segmenting.

Figure 5.11 R_{noise} curve ($0 \leq n \leq 50$) for the extracted image from WHF-20150323-1



A test on criterion (5.6) was conducted. For images taken from the "Winter" area, the applicable threshold was found to be -9.56, which was obtained by the OLAB method using image WHF-20150401-1 based on criterion (5.6). This threshold with the OLAB method was used to process the first seven images taken in the "Winter" area, and the results are listed in Table 5.3. The results showed that the background noise were greatly reduced. Therefore, the criterion for selecting the optimal threshold was validated. This criterion was then applied to select the applicable thresholds for all six methods, and their results are listed in Table 5.4 and Table 5.5 respectively.

Table 5.3 Results for testing criterion (5.6)

Images	R_{noise} with OLAB method	R_{noise} with *THoptimal*
WHF-20150314-1	20.2%	2.2%
WHF-20150317-1	22.5%	2.6%
WHF-20150320-1	15.2%	5.3%
WHF-20150323-1	7.6%	0.3%
WHF-20150326-1	6.2%	0.8%
WHF-20150329-1	6.1%	2.3%
WHF-20150401-1	0.9%	0.8%

* $n \leq 50$

Table 5.4 Applicable thresholds from the three methods in LAB color space.

Images	KLAB		MLAB		OLAB	
	R_{noise}	Threshold	R_{noise}	Threshold	R_{noise}	Threshold
WHF-20150314-1	1.2%	-7.9%	3.5%	-5.1%	13.8%	0.4%
WHF-20150317-1	1.1%	-8.5%	3.1%	-5.2%	15.5%	-0.1%
WHF-20150320-1	3.8%	-11.1%	5.6%	-8.9%	10.2%	-1.5%
WHF-20150323-1	0.2%	-11.3%	0.3%	-8.2%	6.8%	-2%
WHF-20150326-1	0.4%	-11.1%	0	14.2%	5.6%	-1.2%
WHF-20150329-1	1.2%	-13.4%	1.1%	-14.2%	5.3%	-5%
WHF-20150401-1	0.4%	-14.4%	0.5%	-16.7%	0.8%	-9.6%

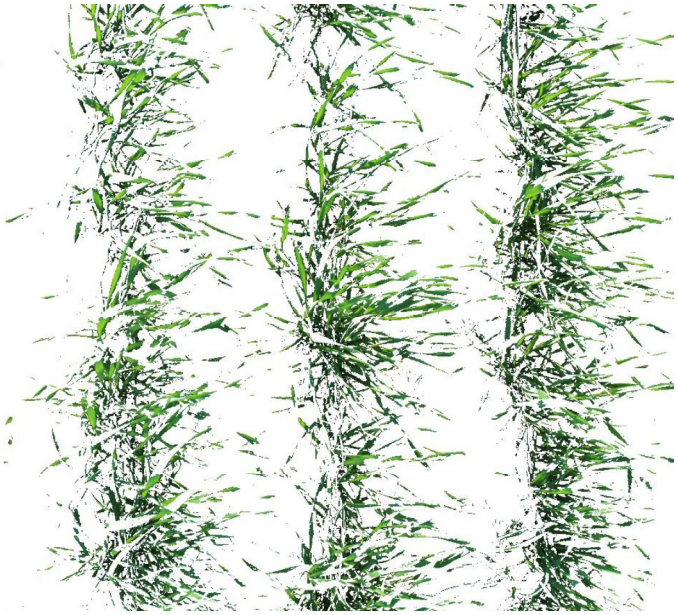
* $n \leq 20$

Table 5.5 Applicable thresholds from the three methods in RGB color space.

Images	KRGB		MRGB		ORGB	
	R_{noise} (%)	Threshold	R_{noise} (%)	Threshold	R_{noise} (%)	Threshold
WHF-20150314-1	2.31	0.22	3.24	0.17	16.37	0.08
WHF-20150317-1	2.43	0.22	3.55	0.16	17.60	0.08
WHF-20150320-1	4.90	0.29	100.00	1.04	11.97	0.14
WHF-20150323-1	0.85	0.26	3.95	0.44	2.65	0.15
WHF-20150326-1	0.60	0.25	5.61	0.47	1.29	0.14
WHF-20150329-1	1.23	0.34	8.85	0.59	1.69	0.23
WHF-20150401-1	0.55	0.35	5.05	0.57	0.43	0.31

* $n \leq 20$

Figure 5.12 Green leaves extracted from the image WHF-20150401-1 using the MRGB method.



As shown in Figure 5.12, the green leaves extracted from image WHF-20150401-1 using the MRGB method gave a R_{noise} value of 5.05%, the red number in Table 5.5. The large R_{noise} value was mainly caused by the groups of green pixels (pixel number ≤ 20) , not the noise from the image background. The six applicable thresholds were then obtained from only one image, WHF-20150401-1(Figure 5.13). The values with purple color were listed in Table 5.4 and 5.5.

Figure 5.13 Image WHF-20150401-1

In order to select the best threshold to accurately extract the green pixels, a semiautomated algorithm was developed to count the actual green pixels in the image. The methods to be compared included the six methods listed in Table 5.1, plus methods KR&L, MR&L, and OR&L. Green pixeles were counted by lines. The semiautomated algorithm included three steps. The first step was to select most of the green pixels in one line by an automated method. The "a" component and "b" component in the LAB color space were used to select. Any pixel in the red dash line area ($-100 \leq a \leq -40; 120^{\circ} \leq \gamma \leq 190^{\circ}$) shown in Figure 5.14 was counted and marked with red color as shown in Figure 5.15 (d).

Figure 5.14 Area of green pixels in the LAB color space.

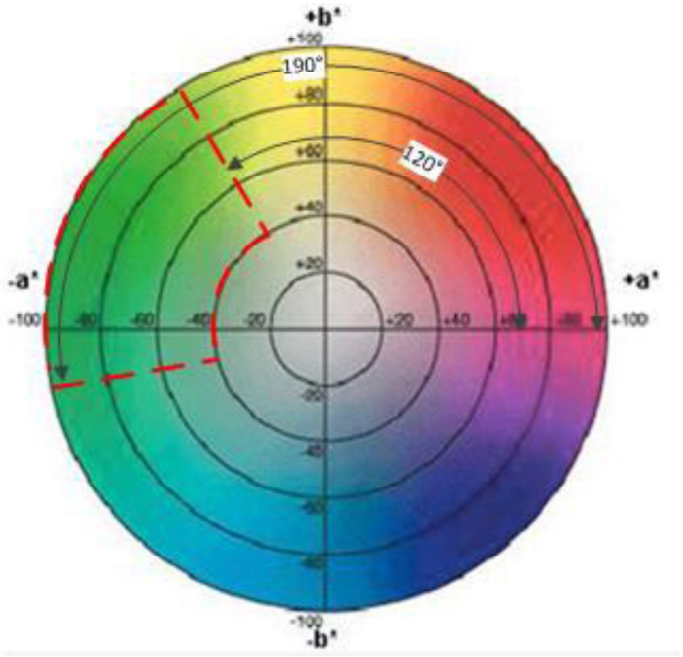
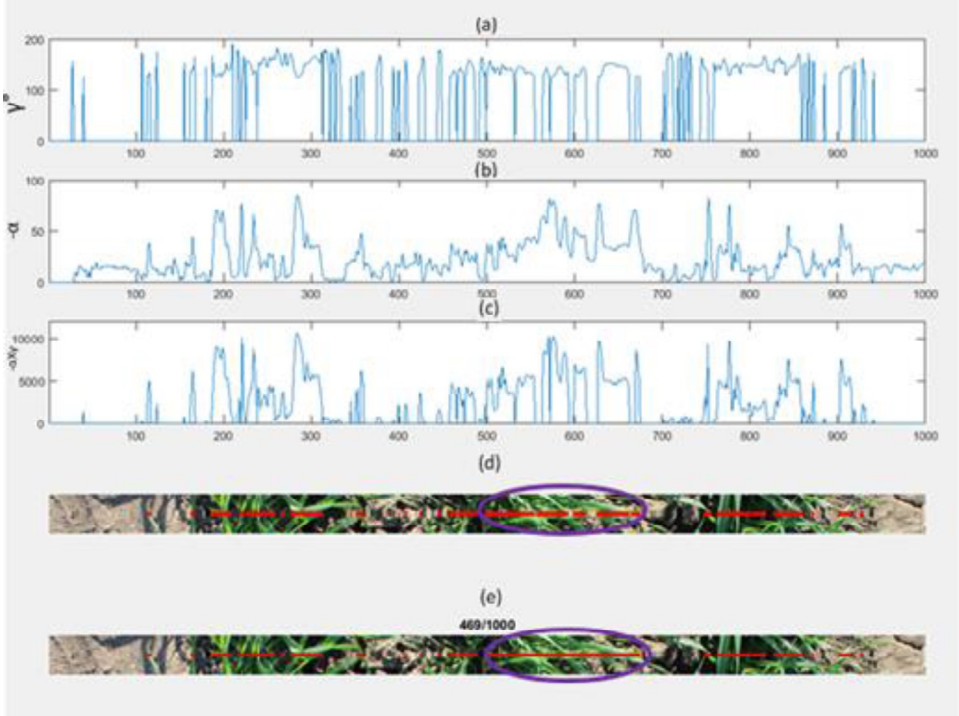


Figure 5.15 Process of counting the green pixels, (a) γ curve of pixels in the LAB color space, (b) a component of pixels in the LAB color space, (c) results of $-a \times \gamma$, (d) results by the automated method, (e) results by manually counting.



In the purple ellipse of Figure 5.15 (d), some green pixels were missed. The missed pixels were manually counted in the second step and are shown in Figure 5.15 (e). After green pixel counting, the PVC was obtained and the extracted image was shown in Figure 5.16. Because background noise stll existed, a third step was used to filter the noise. The resulting image was shown in Figure 5.17, and the actual PVC was equal to 48.7%, which was obtained from image WHF-20150401-1.

Figure 5.16 Image after counting green pixels.

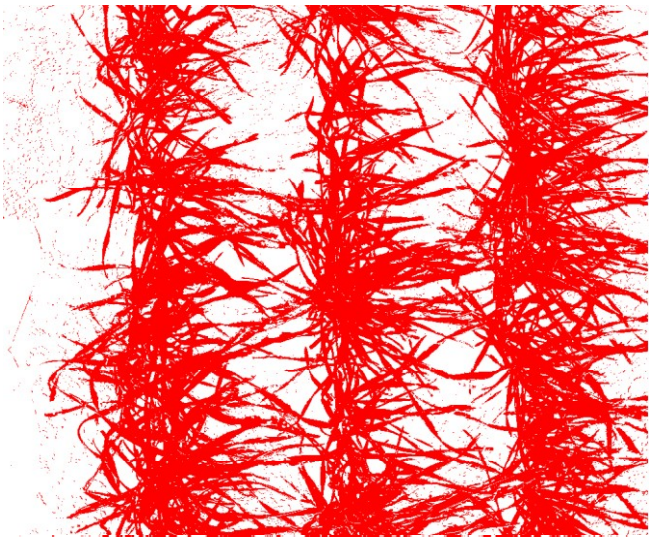
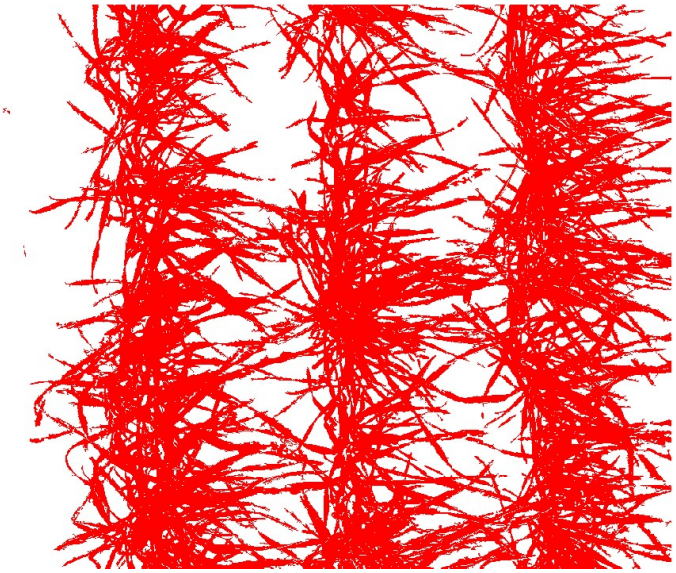


Figure 5.17 Image after filtering the background noise.



Nine thresholds were used to segment the image WHF-20150401-1, and the PVCs obtained are listed in Table 5.6. The selected applicable threshold for all images taken in the "Winter" area was -9.6 which was the A component value in the LAB color space because the PVC value (48.4%) obtained using this threshold was closest to the actual PVC value (48.7%).

Table 5.6 PVCs of the extracted image by the nine thresholds in the bracket pairs.

Methods	LAB	RGB	LAB&RGB
K-Means	46.6% (-14.4)	43.5% (0.35)	38.2% (-14.4&0.35)
ME	39.7% (-16.7)	20.3% (0.57)	18.1% (-16.7&0.57)
OTSU	48.4% (-9.6)	46.5% (0.31)	44.2% (-9.6&0.31)

5.2.2.1.2 Determination of the other four applicable thresholds

The same method was used to estimate the other four selected applicable thresholds to extract the green pixels from the images taken the areas named Fungici, TRAY1, TRAY9, and U6837. The total selected applicable thresholds and the methods used to calculate them were listed in Table 5.7.

Table 5.7 Applicable thresholds and methods used to calculate them.

Wheat pictures	Applicable methods	Applicable thresholds
Pictures of Winter area	OLAB	-9.6
Pictures of Fungici area	OLAB	-9.1
Pictures of TRAY1 area	ORGB	0.24
Pictures of TRAY9 area	ORGB	0.26
Pictures of U6837 area	OR&L	-1.1&0.15

5.2.2.1.3 Validation of the image processing algorithm

Three images from each weat group, including the images used to determine applicable threshold, were selected. The semiautomatical green pixel counting method was employed to count the actual PVCs for comparison. The actual PVCs, calculated PVCs, and Relative errors are listed in Table 5.8. The maximum and minimum erros were 9.1% and 0.3%, respectively.

Generally, the maximum errors occurred during the early wheat growth stage due to the impact of soil background.

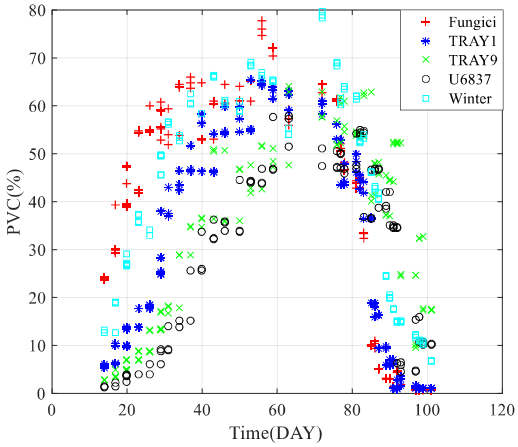
5.2.2.1.4 Extraction of PVCs of all images

By using these thresholds, all images taken in the five areas, Fungici, TRAY1, TRAY9, U6837, and Winter, were segmented, and the results are compared in Figure 5.18.

Table 5.8 Validation of methods of extracting green pixels.

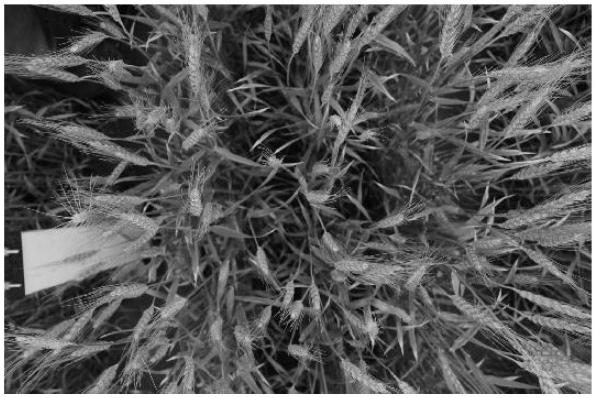
Areas	Pictures	Actual PVCs (%)	Calculated PVCs (%)	Relative errors (%)
Winter area	WHF-20150326-1	33	30	-9.1
	WHF-20150401-1	49.5	48.4	-2.2
	WHF-20150426-1	63.8	59.7	-6.4
Fungici area	FWH20150314-1	27.1	24.7	-8.9
	FWH20150323-2	50.2	48	-4.4
	FWH20150326-1	48	44.2	-7.9
TRAY1 area	TRAY1-20150401-1	25.7	25.9	0.8
	TRAY1-20150410-1	44.9	44.6	-0.7
	TRAY1-20150517-1	65.6	65.8	0.3
TRAY9 area	TRAY9-20150329-1	12.9	12.4	-3.9
	TRAY9-20150410-1	35	34.1	-2.6
	TRAY9-20150416-1	52.8	49.8	-5.7
U6837 area	U6837 -20150407-1	16	17.5	9.4
	U6837 -20150416-1	37.5	36.6	-2.4
	U6837 -20150426-1	51.1	51.8	1.4

Figure 5.18 PVCs of all images.



5.2.2.2 Texture-color analysis for counting wheat heads

Figure 5.19 Gray-tone wheat picture.



The gray-tone wheat picture in Figure 5.19 shows that texture is the main difference between the wheat head and the leaves, stems, soil ground, etc. Although some literatures have counted wheat heads based on color features, texture features, or both, the recognition rate has been affected by weather, lighting, planting method (drilling or broadcasting), or training method (Cointault et al., 2008; Liu et al., 2014). In this study, we proposed a combined texture-color analysis method based on the hypothesis that a wheat image can be divided into wheat head and background by texture features. The computing process of this method is shown in Figure 5.20.

Figure 5.20 Computing process of the texture-color analysis method.

Tuceryan and Jain (1993) defined image texture as a function of spatial variation in pixel intensities (gray values). Using image texture features to recognize image regions is an essential method in machine vision (Tuceryan and Jain, 1993). In the 1970s Haralick (1970) proposed the gray level co-occurrence matrix (GLCM), a well-known and widely used method for image analysis. Co-occurrence probabilities are conditional joint probabilities of all pair combinations of gray levels in the spatial window of interest given two parameters: interpixel distance (δ) and

orientation (θ) (Clausi, 2002). The probability measurement is given by Equations (5.7) and (5.8) (Clausi, 2002):

$$P_r(x) = \{p(i,j)|(\delta,\theta)\} \qquad (5.7)$$

$$p(i,j) = \frac{P_{ij}}{\sum_{i,j=1}^{N_g} P_{ij}} \qquad (5.8)$$

Where $p(i,j)$ =the co-occurrence probability between grey level i and j in a normalized GLCM;

P_{ij} =the number of occurrences of grey levels i and j within the given window (δ,θ);

N_g=the quantized number of grey levels;

The mean, standard deviations, and other parameters for the matrix rows and columns are calculated using Equations (5.9)-(5.21) (Haralick et al., 1973; Soh and Tsatsoulis, 1999; Clausi, 2002).

$$\mu_x = \sum_i \sum_j i \cdot p(i,j) \qquad (5.9)$$

$$\mu_y = \sum_i \sum_j j \cdot p(i,j) \qquad (5.10)$$

$$\mu = \sum_i \sum_j (ij) p(i,j) \qquad (5.11)$$

$$\sigma_x = \sum_i \sum_j (i-\mu_x)^2 \cdot p(i,j) \qquad (5.12)$$

$$\sigma_y = \sum_i \sum_j \left(i-\mu_y\right)^2 \cdot p(i,j) \qquad (5.13)$$

$$p_y(j) = \sum_{i=1}^{N_g} p(i,j) \qquad (5.14)$$

$$p_{x+y}(k) = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j) \,\big|\, i+j=k \qquad k = 2,3,\cdots,2N_g \qquad (5.15)$$

$$p_{x-y}(k) = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j) \,\big|\, |i-j| = k \qquad k = 0,1,\cdots,N_g-1 \quad (5.16)$$

$$HX = -\sum_{i} p_x(i) log\{p_x(i)\} \qquad (5.17)$$

$$HY = -\sum_{j} p_y(j) log\{p_y(j)\} \quad (5.18)$$

$$HXY = -\sum_{i}\sum_{j} p(i,j) log\{p(i,j)\} \quad (5.19)$$

$$HXY1 = -\sum_{i}\sum_{j} p(i,j) log\{p_x(i)p_y(j)\} \quad (5.20)$$

$$HXY1 = -\sum_{i}\sum_{j} p_x(i)p_y(j) log\{p_x(i)p_y(j)\} \quad (5.21)$$

In this study, the optimal texture features were initially selected from the 23 texture features proposed by Haralick. Equations related to the 23 texture features are listed in Table 5.9 (Haralick et al., 1973; Soh and Tsatsoulis, 1999; Clausi, 2002).

Table 5.9 Equations related to 23 texture features.

Autocorrelation(autoc)	$f_1 = \sum_{i}\sum_{j} (ij)\, p(i,j) \qquad (5.22)$
Contrast(contr)	$f_2 = \sum_{n=0}^{N_g-1} n^2 \left\{ \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j) \,\big|\, |i-j| = n \right\} \quad (5.23)$
Correlation(Matlab)(corrm)	$f_3 = \sum_{i}\sum_{j} \frac{(i-\mu_x)(j-\mu_y)p(i,j)}{\sigma_x\sigma_y} \quad (5.24)$

Correlation(corrp)	$f_4 = \sum_i \sum_j \frac{(ij)p(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y}$ (5.25)
Cluster Prominence(cprom)	$f_5 = \sum_i \sum_j \left(i + j - \mu_x - \mu_y\right)^4 p(i,j)$ (5.26)
Cluster Shade(cshad)	$f_6 = \sum_i \sum_j \left(i + j - \mu_x - \mu_y\right)^3 p(i,j)$ (5.27)
Dissimilarity(dissi)	$f_7 = \sum_i \sum_j \lvert i - j \rvert \, p(i,j)$ (5.28)
Energy(energy)	$f_8 = \sum_i \sum_j p(i,j)^2$ (5.29)
Entropy(entro)	$f_9 = -\sum_i \sum_j p(i,j) \log\left(p(i,j)\right)$ (5.30)
Homogeneity(Matlab)(homom)	$f_{10} = \sum_i \sum_j \frac{p(i,j)}{1 + \lvert i - j \rvert}$ (5.31)
Homogeneity(homop)	$f_{11} = \sum_i \sum_j \frac{p(i,j)}{1 + (i - j)^2}$ (5.32)
Maximum Probability(maxpr)	$f_{12} = \max_{i,j} p(i,j)$ (5.33)
Sum of squares: Variance(sosvh)	$f_{13} = \sum_i \sum_j (i - \mu)^2 \, p(i,j)$ (5.34)
Sum Variance(savrh)	$f_{14} = \sum_{i=2}^{2N_g} (1 - f_{16})^2 p_{x+y}(i)$ (5.35)
Sum Average(savgh)	$f_{15} = \sum_{i=2}^{2N_g} i p_{x+y}(i)$ (5.36)

Sum Entropy(senth)	$f_{16} = -\sum_{i=2}^{2N_g} p_{x+y}(i) \log\{p_{x+y}(i)\}$ (5.37)
Difference Variance(dvarh)	$f_{17} = \sum_{i=0}^{N_g-1} i^2 p_{x-y}(i)$ (5.38)
Difference Entropy(denth)	$f_{18} = -\sum_{i=0}^{N_g-1} p_{x-y}(i) log\{p_{x-y}(i)\}$ (5.39)
Information measure of correlatin1(inf1h)	$f_{19} = \frac{HXY - HXY1}{max\{HX, HY\}}$ (5.40)
Information measure of correlation2(inf2h)	$f_{20} = (1 - \exp[-2(HXY2 - HXY)])^{1/2}$ (5.41)
Inverse Difference (INV)(homom)	$f_{21} = \sum_i \sum_j \frac{p(i,j)}{1 + \lvert i-j \rvert}$ (5.42)
Inverse Difference Normalized(INN)(indnc)	$f_{22} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \frac{p(i,j)}{1 + \lvert i-j \rvert / N_g}$ (5.43)
Inverse Difference Moment Normalized(IDN)(idmnc)	$f_{23} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \frac{p(i,j)}{1 + (i-j)^2 / N_g^2}$ (5.44)

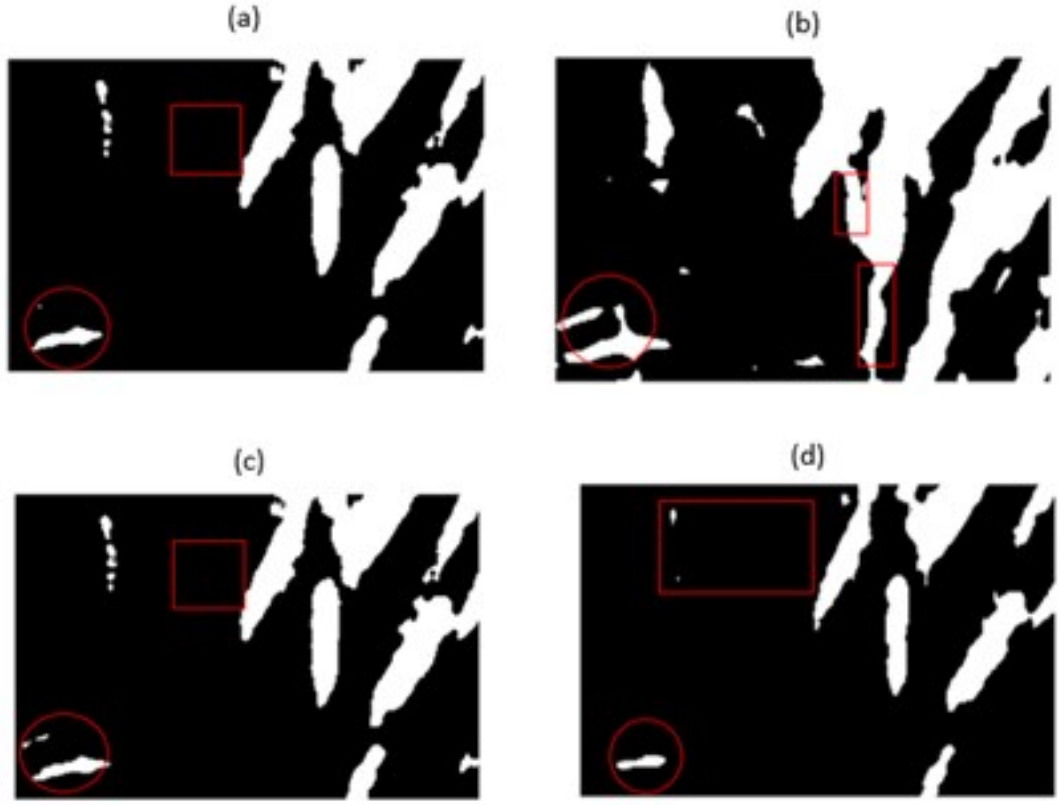
A portion of one wheat image, shown in Figure 5.21, was used to select the parameters that best describe texture characteristics of wheat. A MATLAB code, written by Avinash Uppuluri (2010) to calculate the 23 texture features, was used in the image processing in this study. Gray-level figures based on texture features were created, as listed in Appendix M. Comparison of these gray-level figures to the original image shown in Figure 5.22 revealed that

the four texture parameters (autocorrelation, variance, sum variance, and sum average) are suitable for characterizing wheat texture. The binary images of the four texture parameters are illustrated in Figure 5.22. The white regions in the binary images represent wheat heads or leaf debris with green-yellow texture. Comparison of the four binary images to the original image revealed the following facts:

- Although the parameter Savgh is most sensitive to the wheat texture, it causes noise (e.g., leaf debris with green-yellow texture [area within red circle in Figure 5.22 (b)] and dark green stems [areas within red rectangles in Figure 5.22 (b)]).
- Figure 5.22 (a) and (c) show that the two parameters Autoc and Sosvh almost identically discriminated wheat heads and their background.
- As shown in Figure 5.22 (d), parameter Svarh had most efficiently removed noise: Leaf debris in the red circle was smaller than others in Figure 5.22 (a), (b), and (c), but Svarh simultaneously removed the two wheat heads in the rectangular area.

Figure 5.21 A portion of one wheat image.

Figure 5.22 Binary images of the corresponding grey level image.



Based on these facts, parameters Savsgh, Autoc, and Sosvh were selected to extract wheat heads from the original image. Figure 5.23 (a) is the original image, image (b) is the binary figure obtained by texture analysis, and image (c) shows the extracted wheat heads. The subsequent objective was to eliminate leaf debris, as indicated by red circles, and small green spots (pixels less than 50), as shown in Figure 5.23 (c). Individual objects, as shown in Figure 5.23 (b) and (c), were then extracted and saved in smaller image files, as shown in Figure 5.24.

Figure 5.23 Extracted wheatheads image

(a)

(b)

(c)

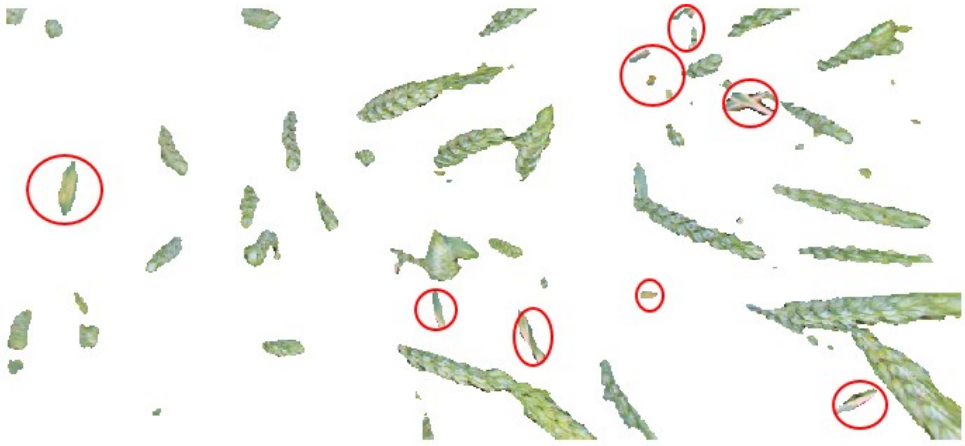


Figure 5.24 Extracted RGB and binary images

Color indices shown in Equations (5.1), (5.2), and (5.3) were then used to eliminate non-green pixels in the images of individual objects. The percent non-green coverage (PNGC) was calculated using Equation (5.45):

$$PNGC = \frac{P_t - P_g}{P_t} \times 100\% \qquad (5.45)$$

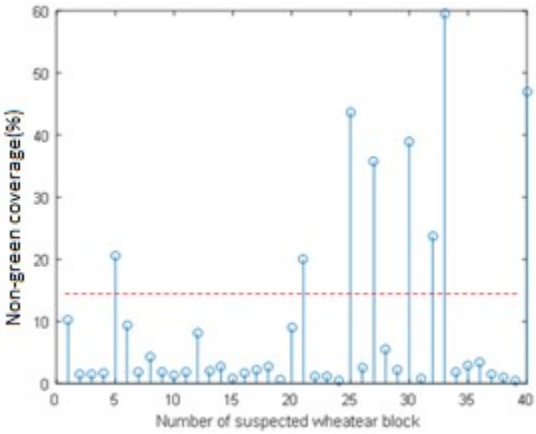
Where

P_t =total number of pixels number before color analysis

P_g=total number of green pixels number after color analysis.

The PNGC were calculated after color analysis, and a stem plot was made, as shown in Figure 5.25. The threshold value was determined to be 15%, so all leaf debris with green-yellow texture were eliminated.

Figure 5.25 Non-green coverage versus block number



The final step of the algorithm was to count wheat heads. The wheat heads were directly counted if the heads were separated from each other; if the wheat heads were connected as shown in Figure 5.23, however, their connecting pattern must be judged before counting (Germain et al., 1995). The literatures have reported several methods for counting wheat heads, such as the splitting and merging method, the wheat head shape index method, and the joint point counting (JPC) method (Germain et al., 1995; Frédéric et al., 2012; Liu et al., 2014). The splitting and merging method splits the connected lobes of wheat heads based on skeleton analysis and restructures every wheat head using corresponding neighboring lobes based on the maximum length limitation for one wheat head and orientation similarity analysis (Germain et al., 1995). Frédéric et al. (2012) proposed the wheat shape index and Equation (5.46) to calculate wheat shape index:

$$Q = \frac{S_c - S}{S} - 1 \quad (5.46)$$

where *S* is the surface and *Sc* is the convex hull surface of the pixel group, as shown in Figure 5.26 (a) and (b).

Figure 5.26 Pixel groups with concave shapes (a) and convex hull of a concave shaped pixel group (b).

(a) (b)

They also hypothesized that *X* number of wheat heads could be estimated using inequality (5.47).

$$\frac{X-1}{10} < Q < \frac{X}{10} \qquad (5.47)$$

Although the value of wheatear shape index *Q* must be greater than zero, *Q* was less than zero because $0 < \frac{S_c - S}{S} < 1$, indicating an input error in Equation (5.46). Instead the equation to calculate the wheat shape index should be

$$Q = \frac{S_c}{S} - 1 \qquad (5.48)$$

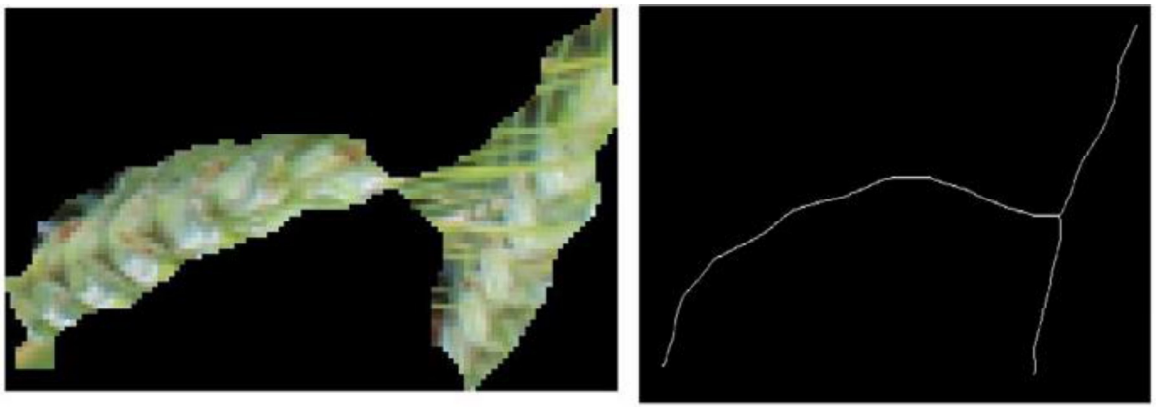
The JPC method requires joint points in the image following a skeleton operation for an image of multiple connected wheat heads; *X* number of wheat heads could be calculated by

$$X = N + 1 \qquad (5.49)$$

Where N is the number of the joint points in the image (Figure 5.27 (a) and (b)).

Figure 5.27 Image with two wheatheads (a) and its skeleton image (b) for counting wheatheads.

(a) (b)



The wheat head shape index method and the JPC method were used to count wheat heads, and the counting results are shown in Figure 5.28(a) and (b). The number of red addition signs represented the number of wheat heads.

Figure 5.28 Wheatear counting results, (a) using the WSI method and (b) using the JPC method.

(a)

(b)

A new method based on the wheat head counting model (WCM) was proposed in this study. As mentioned, every pixel group in the image was extracted and saved as a new binary image following texture analysis and segmentation (Figure 26 (a)). The function of *regionprops* in the Matlab image tool box was used to measure the geometrical parameters of every independent pixel group in an image, including Area(S), MajorAxisLength(L_{max}), MinorAxisLength(L_{min}), ConvexArea(S_c), ConvexArea(S_c)and Perimeter(P). A new concept, area ratio (AR), (S_r) was given by

$$S_r = \frac{S}{L_{max} \times L_{min}} \tag{5.50}$$

Another new concept, perimeter ratio (PR), (P_r) was represented by

$$P_r = \frac{P}{2(L_{max}+L_{min})} \tag{5.51}$$

The number of wheat heads in every image was determined by the four parameters AR S_r, wheat head shape index (WSI) Q, solidity p_s, and PR p_r. Then a WCM based on the image measurement was given by

$$y_i = \beta_0 + \beta_1 S_{ri} + \beta_2 Q_i + \beta_3 p_{si} + \beta_4 p_{ri} + \beta_5 S_{ri} Q_i + \beta_6 S_{ri} p_{si} + \beta_7 S_{ri} p_{ri} + \beta_8 Q_i p_{si} + \beta_9 Q_i p_{ri} + \beta_{10} p_{si} p_{ri} + \beta_{11} S_{ri} Q_i p_{si} + \beta_{12} S_{ri} Q_i p_{ri} + \beta_{13} Q_i p_{si} p_{ri} + \beta_{14} p_{si} p_{ri} S_{ri} + \beta_{15} S_{ri} Q_i p_{si} p_{ri} + e_i \quad (5.52)$$

Where, y_i---------the observed wheatear number in the i^{th} image block;

$\beta_0, \beta_1, \cdots \beta_{15}$------the regression coefficients;

e_i------the left-over noise corresponding to the i^{th} image block

$$e_i \sim NIID(0, \delta^2)$$

In order to calculate wheat head counting model, 51 blocks of pixel group were extracted by experiments and measured using the function *regionprops()*. Their measurement data were recorded. The multiple linear regression model expressed by Equation (5.53) was obtained by the function *regress()* in MatlabR 2015.

$$\widehat{y_i} = -176.56Q_i - 19.33p_{si} + 15.88p_{ri} + 198.67S_{ri}Q_i + 27.99S_{ri}p_{si} - 23.34S_{ri}p_{ri} + 390.61Q_i p_{si} + 145.7Q_i p_{ri} - 492.89S_{ri}Q_i p_{si} - 182.37S_{ri}Q_i p_{ri} - 323.83Q_i p_{si} p_{ri} + 436.33S_{ri}Q_i p_{si} p_{ri} \quad (5.53)$$

and, $R^2 = 0.95$ (5.54)

$F = 65.65\ with\ \alpha = 0.05$ (5.55)

Because $F_{0.05}(15,35) = 1.96 \ll F$ （$F_\alpha(K-1, N-K)$, F-distribution at level α with K-1, N-K degrees of freedom under the null hypothesis, here K=16, N=51） and based on a Type I error rate of 5%, the null hypothesis was rejected and at least one of the coefficients β_j $(j = 0,1,2 \cdots 15)$ was considered to differ from zero, resulting in significant association between the

wheat head number and at least one of the image measurement values. Counting results using WCM are shown in Figure 5.29. Disadvantages and advantages of the three methods and needed improvements are discussed below.

Figure 5.29 Wheatear counting results using the WCM (Wheatear Counting Model) method.

5.3 Results

5.3.1 Percent vegetation coverage

5.3.1.1 Wheat growth model (WGM) based on PVC

All PVC data of each selected area are presented. In order to intuitively illustrate wheat growth status and compare PVC differences among the five selected areas, the wheat growth model (WGM) related to PVC versus time was defined by

$$g_i = \beta_0 + \beta_1 t_i + \beta_2 t_i^2 + \beta_3 t_i^3 + \beta_4 t_i^4 + \beta_5 t_i^5 + \beta_6 t_i^6 + e_i \qquad (5.56)$$

Where, g_i---------the observed PVC on the i^{th} day from 1 Mar 2015;

$\beta_0, \beta_1, \cdots \beta_6$------the regression coefficients;

e_i------the left-over noise corresponding on the i^{th} day

$$e_i \sim NIID(0, \delta^2)$$

All PVC data were used in the regressional analysis, and the coefficients of every model corresponding to the selected area were obtained, as listed in Table 5.10.

$$\widehat{g_i} = \widehat{\beta_0} + \widehat{\beta_1} t_i + \widehat{\beta_2} t_i^2 + \widehat{\beta_3} t_i^3 + \widehat{\beta_4} t_i^4 + \widehat{\beta_5} t_i^5 + \widehat{\beta_6} t_i^6 \qquad (5.55)$$

Table 5.10 Models' coefficients.

Coeffi.	Fungici*	TRAY1*	TRAY9*	U6837*	Winter *
$\widehat{\beta_0}$	0	0	0	0	0
$\widehat{\beta_1}$	0	0	0	0	0
$\widehat{\beta_2}$	0.001	0	0	0	0.001
$\widehat{\beta_3}$	-0.034	-0.015	0.002	-0.002	-0.069
$\widehat{\beta_4}$	0.771	0.386	0.003	0.117	2.194
$\widehat{\beta_5}$	-4.306	-3.21	-0.282	-3.017	-30.871
$\widehat{\beta_6}$	0	0	0	25.633	168.722

* Fungici, TRAY1, TRAY9, U6837, and Winter are the wheat labels at the selected area.

Statistics features are listed in Table 5.11.

Table 5.11 Models' statistics features

Name	R square	F -Value	Estimate of error variance
Fungici	0.957	526.579	27.281
TRAY1	0.961	587.131	19.938
TRAY9	0.915	257.732	34.964
U6837	0.946	344.314	21.440
Winter	0.953	401.759	20.757

Because $F_{0.05}(6,109) = 2.18 \ll$ anyone of F-values in Table 5.5 （$F_\alpha(K-1, N-K)$, F-distribution at level α with K-1, N-K degrees of freedom under the null hypothesis, here K=7, N=116）and based on a Type I error rate of 5%, the null hypothesis was rejected and at least one of the coefficients β_j $(j = 0,1,2 \cdots 6)$ was considered to differ from zero, demonstrating a significant association between the PVC and time.

5.3.1.2 Analysis of wheat growth status based on WGM

Based on the models of WGM, wheat growth curves of the five selected wheat areas are shown in Figure 5.30. For real-world application, wheat growth status can be divided into two

phases: growth period and mature period. The date range of the turning point from the growth period to the mature period was May 15-18, 2015, after which time the wheat entered the Feekes 11.0 stage. The wheat growth curve (WGC) showed the differences of wheat development in various areas.

Figure 5.30 Wheat growth curves

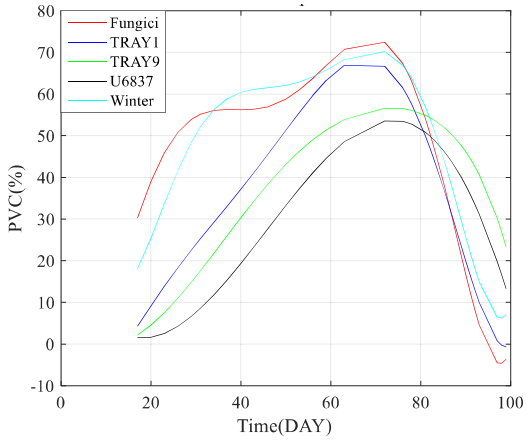


Figure 5.30 also shows that the PVC decreasing rate were different compared to each other throughout the mature period, and developing speed curves of the five selected wheat reinforced this result, as shown in Figure 5.31. In fact, all wheat in the five areas were affected by leaf rust in the mature period, as shown in Figure 5.32, and the conclusion was made that TRAY9 wheat was less affected than the others, and the Fungici wheat was most severely affected of all the wheat, as described in Appendix L.

Figure 5.31 Wheat developing speed curves

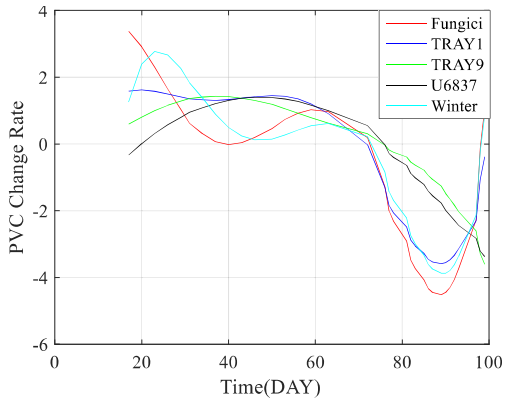


Figure 5.32 Wheats affected by leaf rust, (a) Fungici wheat, (b) U6837 wheat, (c) TRAY1 wheat, (d) TRAY9 wheat, (e) Winter wheat.

5.3.2 Wheat head counting algorithm

As mentioned, the wheat head counting process based on machine vision was divided into three steps: wheat head image segmentation, leaf debris elimination, and wheat head counting. This study investigated the effects of light intensity on image segmentation, the efficiency of leaf debris elimination, and wheat head counting methods. The studied images were selected and cut from the corresponding original pictures; and the dimensions of cut images were

usually 400X400 pixels, and the window dimension to calculate texture features was 11X11 pixels.

Figure 5.33 (a) Original image, (b) autocorrelation grey-level image; (c) sum variance grey-level image under sunny weather

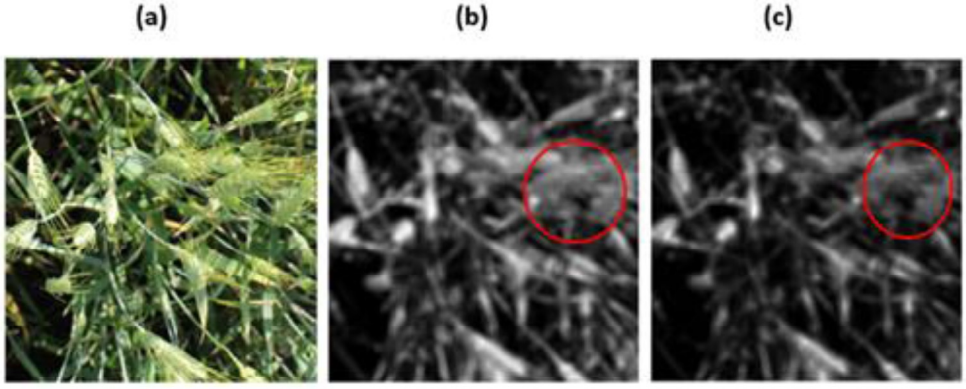
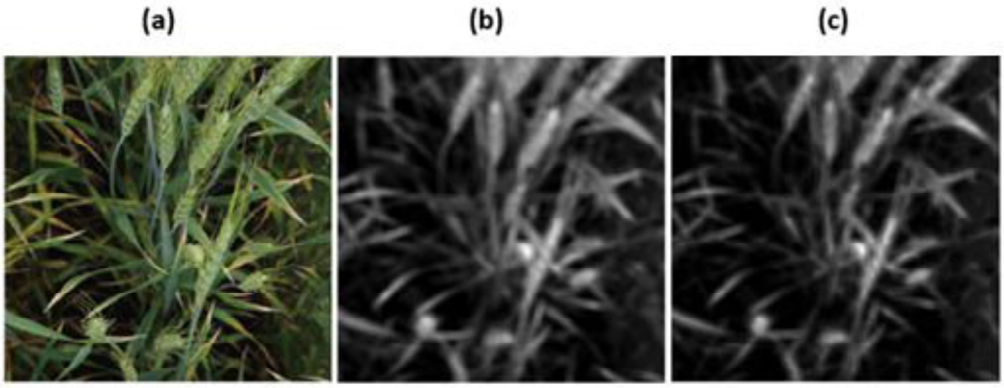


Figure 5.34 (a) Original image, (b) autocorrelation gray-level image, and (c) sum variance gray-level image under cloudy weather



5.3.2.1 Wheat head image segmentation based on texture analysis

Shade in the wheat image in sunny weather was a challenge for image segmentation based on texture analysis. Eighteen selected images were taken under sunny weather or cloudy conditions to be used in this study. Figures 5.33 and 5.34 showed two examples of the original images and corresponding texture parameter gray-level images. The texture parameter gray-level images show that the wheat heads in the gray-level images in Figure 5.33 had more gray levels than the wheat heads in the gray-level images in Figure 5.34. The gray level of wheat heads in

Figure 5.34 was almost uniform, but the gray level of wheat heads in the red circle was darker than other wheat heads in Figure 5.33. Significant gray-level difference between wheat heads in same image can easily lead missing wheat heads after converting the image into binary image. Therefore, two algorithms were used for various lighting conditions. For the image taken in sunny weather like the RGB image in Figure 33, the algorithm obtained the gray sum of two gray-level images (Figure 5.33) and then converted the summed gray-level image into a binary image. For the image taken in cloudy weather, however, the algorithm converted the two gray-level images into two binary images and then combined the two binary images into one using OR Boolean operation. Finally, the extracted wheat head image was obtained by image segmentation based on the created binary image. A new concept of wheat head extracting rate R_{wer} was given by Equation (5.58):

$$R_{wer} = \frac{N_o}{N_e} \times 100\% \qquad (5.58)$$

Where

N_o=number of wheat heads in the original image

N_e=number of extracted wheat heads in the segmented image.

The wheat head extracting rates are listed in Table 5.12 for images taken under various weather conditions and time. Data with light brown backgrounds were obtained from images taken under cloudy weather conditions. The average of R_{wer} for images taken under sunny and cloudy weather conditions were 90.1% and 82.8% respectively.

Table 5.12 Wheat head extracting rates for various images

Exp. No	N_o	N_e	R_{wer}(%)
705-7	21	20	95.2
705-5	21	21	100
705-6	17	16	94.1

705-11	15	11	73.3
705-13	16	13	81.3
705-14	20	20	100.0
705-15	17	16	94.1
705-2	23	19	82.6
705-9	18	16	88.9
705-10	16	16	100
705-12	27	23	85.2
705-19	16	10	62.5
705-20	26	24	92.3
705-21	21	18	85.7
705-22	20	18	90
705-23	28	19	67.9
705-24	20	18	90
705-25	26	17	65.4
Average of R_{wer}			85.6

5.3.2.2 Leaf debris elimination based on color analysis

Extracted wheat head images were obtained following the texture analysis, as shown in Figure 5.35. Except the wheat head blocks in the image, there were some small leaf debris. The primary objective of the color analysis was to eliminate the leaf debris in order to obtain accurate head counting. In order to assess the capability of the color analysis method in eliminating leaf debris, two evaluation criteria (leaf-eliminating rate R_{ler} and leaf-eliminating error rate R_{leer}) were defined by Equations (5.59) and (5.60):

$$R_{ler} = \frac{N_{lde}}{N_{ld}} \times 100\% \qquad (5.59)$$

Where, N_{lde}= the number of eliminated leaf debris in the images;

N_{ld}=-the total number of leaf debris in the images;

Figure 5.35 Extracted wheat head images, (a) image taken under sunny weather, (b) image taken under cloudy weather.

(a) (b)

$$R_{leer} = \frac{N_{we}}{N_{lde}+N_{we}} \times 100\% \qquad (5.60)$$

Where, N_{we}=the eliminated wheatear number wrongly taken as the leaf debris by the method;

Table 5.13 Leaf-eliminating rate and eliminating error rate

Exp. No.	N_{ld}	N_{lde}	N_{we}	R_{ler}	R_{leer}
705-7	8	0	0	0	
705-5	8	0	0	0	
705-6	8	2	0	25	0
705-11	11	0	0	0	
705-13	1	1	0	100	0
705-14	4	0	0	0	
705-15	7	1	1	14.3	50
705-2	3	0	0	0	
705-9	4	1	0	25	0
705-10	2	1	0	50	0
705-12	5	2	0	40	0
705-19	2	1	0	50	0
705-20	4	3	1	75	25
705-21	5	4	0	80	0
705-22	8	6	1	75	14.3
705-23	5	2	0	40	0
705-24	5	4	0	80	0
705-25	2	2	1	100	33.3

Similar to Table 5.12, data with the light brown background in Table 5.13 were obtained from images taken under cloudy weather conditions. The average number of leaf debris block for images taken under sunny and cloudy weather conditions were 6.25 and 4.2 respectively. The average R_{ler} were 17.4%, and 61.5% respectively. Thus, the color analysis method for eliminating leaf debris was more effective under cloudy conditions. Loss of wheat heads probably occurred during thresholding in the color analysis.

5.3.2.3 Wheatear counting methods

As mentioned, the WCM method, JPC method, and WSI method were used in this study. Because the three methods were used to count wheat heads in the blocks after color analysis, the study was done under the assumption that all leaf debris had been completely eliminated. All tested data are listed in Table 5.14; the following relationship (Equation 5.61) is true for all methods.

$$N_{wib} = N_{wc} - N_{uc} - N_{oc} \qquad (5.61)$$

Where, N_{wib}=the total number of actual wheatear in the blocks of one experiment;

N_{wc}=the number of wheatear counted by the method;

N_{oc}=the number of wheateat over-counted;

N_{uc}=the number of wheatear under-counted;

Table 5.14 Wheatear number using three methods.

Exp. No	N_{wib}	WCM Method			JPC Method			WSI Method		
		N_{wc}	N_{oc}	N_{uc}	N_{wc}	N_{oc}	N_{uc}	N_{wc}	N_{oc}	N_{uc}
705-7	20	19	1	-2	31	12	-1	48	28	0
705-5	21	22	2	-1	38	18	-1	70	50	-1
705-6	16	15	0	-1	27	11	0	49	33	0
705-11	11	17	6	0	23	12	0	57	46	0
705-13	13	11	2	-4	15	2	0	21	9	-1
705-14	20	18	1	-3	22	3	-1	38	18	0
705-15	15	20	6	-1	29	14	0	38	23	0

705-2	19	18	0	-1	26	7	0	47	28	0
705-9	16	16	0	0	15	0	-1	39	23	0
705-10	16	15	0	-1	20	5	-1	30	14	0
705-12	23	17	1	-7	38	15	0	36	15	-2
705-19	10	13	3	0	12	2	0	38	28	0
705-20	23	26	3	0	24	1	0	56	33	0
705-21	18	19	2	-1	24	6	0	43	25	0
705-22	17	18	1	0	22	11	-6	32	15	0
705-23	19	19	1	-1	24	6	-1	51	32	0
705-24	18	19	2	-1	24	6	0	43	25	0
705-25	16	15	0	-1	18	4	-2	33	17	0

In order to compare the three methods, three evaluation criteria were defined using Equations (5.62), (5.63), and (5.64):

- Over-counting rate R_{oc}

$$R_{oc} = \frac{N_{oc}}{N_{wib}} \times 100\% \qquad (5.62)$$

- Under-counting rate R_{uc}

$$R_{uc} = \frac{|N_{uc}|}{N_{wib}} \times 100\% \qquad (5.63)$$

- Total counting error R_{ce}

$$R_{ce} = R_{oc} + R_{uc} \qquad (5.64)$$

Table 5.15 Wheat head counting accuracy for the three methods

Exp. No.	WCM Method			JPC Method			WSI Method		
	R_{oc}	R_{uc}	R_{ce}	R_{oc}	R_{uc}	R_{ce}	R_{oc}	R_{uc}	R_{ce}
705-7	5.0%	10%	15%	60%	5%	65%	140%	0	140%
705-5	9.5%	4.8%	14.3%	85.7%	4.8%	90.5%	238.1%	4.8%	242.9%
705-6	0	6.3%	6.3%	68.8%	0	68.8%	206.3%	0	206.3%
705-11	54.5%	0	54.5%	109.1%	0	109.1%	418.2%	0	418.2%
705-13	15.4%	30.8%	46.2%	15.4%	0	15.4%	69.2%	7.7%	76.9%
705-14	5.0%	15%	20%	15%	5%	20%	90%	0	90%
705-15	40.0%	6.7%	46.7%	93.3%	0	93.3%	153.3%	0	153.3%
705-2	0	5.3%	5.3%	36.8%	0	36.8%	147.4%	0	147.4%
705-9	0	0	0	0	6.3%	6.3%	143.8%	0	143.8%
705-10	0	6.3%	6.3%	31.3%	6.3%	37.5%	87.5%	0	87.5%
705-12	4.3%	30.4%	34.8%	65.2%	0	65.2%	65.2%	8.7%	73.9%

705-19	30%	0	30%	20%	0	20%	280%	0	280%
705-20	13.0	0	13%	4.3%	0	4.3%	143.5%	0	143.5%
705-21	11.1%	5.6%	16.7%	33.3%	0	33.3%	138.9%	0	138.9%
705-22	5.9%	0	5.9%	64.7%	35.3%	100%	88.2%	0	88.2%
705-23	5.3%	5.3%	10.5%	31.6%	5.3%	36.8%	168.4%	0	168.4%
705-24	11.1%	5.6%	16.7%	33.3%	0	33.3%	138.9%	0	138.9%
705-25	0	6.3%	6.3%	25.0%	12.5%	37.5%	106.3%	0	106.3%
Average	11.7%	7.7%	19.4%	44%	4.5%	48.5%	156.8%	1.2%	158%

Table 5.15 showed that, the WCM method performed better than the JPC and WSI methods. And, for the WCM method, the average counting error rate for images taken under cloudy and sunny weather conditions were 14%, and 26% respectively.

5.3.2.4 WCM-based algorithm test

Although the wheat head detection rate (90.1%) for images taken in sunny weather was greater than that under cloudy weather (82.8%), images taken without sunlight interference were recommended in this study due to their improved leaf debris elimination rate in color analysis and counting error rate using WCM method. The final image processing algorithm modified based on the above results was tested, where the non-green coverage threshold was 25%, and the image used in the test was taken without sunlight interference (under cloudy weather conditions). To further study these limitations, eight images taken from the Fungici and Winter areas were selected, and the WCM-based algorithm was employed to process these images. The results are shown in Table 5.16. The evaluation parameters as defined in equations (5.58), (5.59), (5.60), (5.62), (5.63), and (5.65) were calculated and are listed in Table 5.17. The average wheat head extraction rate (R_{wer}) was 78.6% after the first processing step. The average leaf debris eliminating rate (R_{ler}) was 77.1%, and the average leaf debris eliminating error rate (R_{leer}) was 4.5% after the second processing step. The average overcounting rate R_{oc} was higher than 24%, indicating the need for further research. The recognition rates (RR) were calculated using equation (5.65), and the average recognition rate was 79.4% by using the WCM-based algorithm.

$$RR = \frac{N_{wc} - N_{oc}}{N_{wc}} \qquad (5.65)$$

Table 5.16 Image processing results of the images taken from Fungici and Winter areas.

Images	N_o	N_e	N_{ld}	N_{lde}	N_{ew}	N_{wib}	N_{wc}	N_{uc}	N_{oc}
FWH20150522-1	66	51	9	5	1	51	56	6	11
FWH20150522-2	69	48	13	10	2	48	50	2	4
FWH20150523-1	53	44	18	12	2	44	44	7	7
FWH20150523-2	61	46	7	5	1	46	48	4	6
WHF20150522-10	47	45	22	21	2	45	51	3	9
WHF20150522-11	52	42	40	35	5	42	52	3	13
WHF20150523-1	51	47	31	24	0	47	62	7	22
WHF20150523-2	48	41	49	42	2	41	56	2	17

Table 5.17 Evaluation parameters about the WCM-based algorithm.

Images	R_{wer}(%)	R_{ler}(%)	R_{leer}(%)	R_{oc}(%)	R_{uc}(%)	RR(%)
FWH20150522-1	77.3	55.6	16.7	21.6	11.8	80.4
FWH20150522-2	69.6	76.9	16.7	8.3	4.2	92
FWH20150523-1	83.0	66.7	14.3	15.9	15.9	84.1
FWH20150523-2	75.4	71.4	16.7	13	8.7	87.5
WHF20150522-10	95.7	95.5	8.7	20	6.7	82.4
WHF20150522-11	80.8	87.5	12.5	31	7.1	75
WHF20150523-1	92.2	77.4	0	46.8	14.9	64.5
WHF20150523-2	85.4	85.7	4.5	41.5	4.9	69.6
Mean	82.4	77.1	11.3	24.8	9.3	79.4

5.4 Discussion

5.4.1 Application of PVC

As mentioned, PVC not only represents the status of wheat canopy coverage in the corresponding area, changes in PVC with time can reflect the status of wheat development. Therefore, the PVC map pertaining to wheat in the field can be used to instruct the field management in situations such as calculating spectral indices for topdress N application. The PVC map can also be used for plant breeding research, such as determining differences among various trains according to the wheat growth curves and selecting optimal wheat varieties for given field conditions.

5.4.2 Wheat head counting method

Experiments showed that many factors affected the wheat head counting accuracies, including weather conditions, the date pictures were taken, and wheat varieties. As shown in Figure 5.36 (a), the wheat heads were located with the wheat leaves in the image, increasing the difficulty to extract the heads from the leaves, as shown in Figure 5.36 (b). Similarly, as shown in Figure 5.37, counting the total number of wheat heads was also difficult because some heads were occluded by others. Because non-green coverage was used as the threshold to eliminate leaf debris, this algorithm could only be used to count wheat heads before the wheat senescent stage.

Figure 5.36 Wheatheads in the leaves.

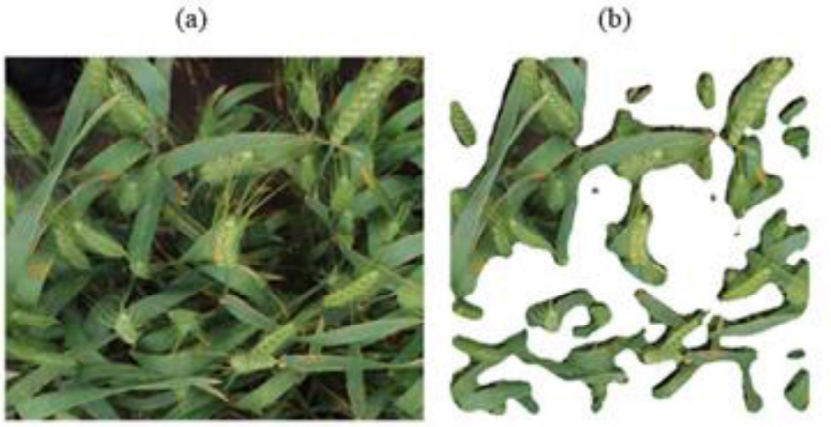


Figure 5.37 Crowed wheatheads.

5.5 Future work

The study on wheat growth trend and head conting using image processing technology was only preliminary. Further research may be focused on the following aspects:

- Exploring more accurate method to extract green pixels in the image;
- Improving wheat head extraction rate (R_{wer});
- Reducing the over-counting rate R_{oc} based on morphological analysis.

Chapter 6 - CONCLUSION

6.1 Conclusions

This research presented the designs of two phenotyping platforms and described the development of a machine vision technology for estimating wheat growth status and counting wheat heads. The following conclusions were obtained from this study:

1. Test results on the handheld phenotyper, Phenocorn, showed that the phenotyper can be reliably used as an instrument to collect plant phenotyping information.To further evaluate the Phenocorn, Crain et al. (2016) conducted a serial of field trials at the International Maize and Wheat Improvement Center (CIMMYT), Ciudad Obregon, Mexico. Validation of the Phenocorn showed that it performed as well as other methods for canopy temperature measurement; additionally, NDVI and CT data from the Phenocorn were significantly correlated to grain yield (Crain et al., 2016).

2. Design and analysis of the robotic phenotyper, including the mechanical structure and CAN-based communication system, were completed. Performances of the basic functions, including steering, load lifting, and moving, were tested in the laboratory. Results showed that the proposed design specifications were basically met. Integration of a laptop, proximal sensors, with a data acquisition system and complete design of the control system are the primary tasks for the next step of development.

3. Machine vision-based detection of phenotyping traits was studied. An algorithm for PVC estimation based on color analysis was estabilished. This algorithm can be used to estimate the wheat growth status and diagnose plant health conditions. Compared to the JPC method and the WSI method, the WCM method proposed in this study was more effective in counting wheat heads. The rexperimantal results of 8 different images using the WCM-based algorithm indicated

that the average over-counting error rate was 24.8% and the average recognition rate RR was 79.4% for images taken under cloudy weather conditions.

6.2 Future work

6.2.1 Future work on handheld phenocorn

The handheld phenocorn developed in this study was based on a first-generation design. In order to promote rapid dissemination and utilization of the device in practical plant breeding field, reducing the total cost and weight should be the focus of future work. As discussed in Section 3.3.5, the cost may be reduced by using a less accurate and less expensive GPS selecting more affordable sensors, and the weight may be reduced by using Nimh batteries instead of Lead-acid batteries (Crain et al., 2016).

6.2.2 Future work on robotic phenotyper

For the robotic phenotyper, the design, fabrication, and assembly of the mechanical structures have been completed. Further testing on the reliability of the mechanical parts and design of the navigation system, electronic control, integration of multiple sensors with data acquisition, algorithms for complete robot control, and laboratory and field tests remain to be completed.

6.2.3 Future work on image processing algorithms

In order to improve the accuracy of the algorithms for crop growth trend study and wheat head countings, further research will be needed. This will include improving the algorithm for green pixel extraction and reducing the over-counting rate.

Chapter 7 - ACKNOWLEDGMENTS

This work was supported through the National Science Foundation–Plant Genome Research Program (IOS-1238187) and the US Agency for International Development Feed the Future Innovation Lab for Applied Wheat Genomics (Cooperative Agreement No. AID-OAA-A-13-00051). This work was also supported by USDA Center for Grain and Animal Health Research, through project NACA 58-3020-5-005.

References

AGRI-FACTS.2007. Using 1000 kernel weight for calculating seeding rates and harvest losses. Available at: http://www1.agric.gov.ab.ca/$department/deptdocs.nsf /all/agdex81/$file/ 100_22-1.pdf? OpenElement. Accessed 11 Jan. 2016.

Albus, J., R. Bostelman, and N. Dagalakis. 1993. The NIST Robocrane. J. Robot. Syst. 10(2): 709–724.

Allah, M Zaman, O Vergara, J L Araus, A Tarekegne, C Magorokosho, P J Zarco‐Tejada, A Hornero,A Hernández Albà, B Das, P Craufurd, M Olsen, B M Prasanna, and J Cairns. 2015. Unmanned aerial platform‐based multi‐spectral imaging for field phenotyping of maize. Plant Methods (2015) 11:35

Amani, I. 1996. Canopy temperature depression association with yield of irrigated spring wheat cultivars in a hot climate. J. Agron. … 129.

Andrade-Sanchez, P., M.A. Gore, J.T. Heun, K.R. Thorp, A.E. Carmo-Silva, A.N. French, M.E. Salvucci, J.W. White, and E.A. Carmo-Silva. 2014. Development and evaluation of a field-based high-throughput phenotyping platform. Funct. Plant Biol. 41(2007): 68–79.

Araus, J.L., and J.E. Cairns. 2014. Field high-throughput phenotyping: the new crop breeding frontier. Trends Plant Sci. 19(1): 52–61.

Babar, M., and M. Reynolds. 2006. Spectral reflectance to estimate genetic variation for in-season biomass, leaf chlorophyll, and canopy temperature in wheat. Crop breeding & genetics, 46(3): 1046-1057.

Bakker, Tijmen, Kees Asselt van, Jan Bontsema, Joachim Müller, and Gerrit Straten van.2010. Systematic design of an autonomous platform for robotic weeding. Journal of Terramechanics, 47 (2010): 63–73.

Balota, M., W. Payne, S. Evett, and M. Lazar. 2007. Canopy temperature depression sampling to assess grain yield and genotypic differentiation in winter wheat. Crop Sci. 47(4): 1518.

Bechar, Avita, and Clèment Vigneault.2016. Agricultural robots for field operations: Concepts and components. Biosystems engineering,149(2016):94-111.

Berni, J. A. J., Zarco-Tejada, P. J., Suarez, L., Gonzalez-Dugo, V., & Fereres, E. 2009. Remote sensing of vegetation from UAV platforms using lightweight multispectral and thermal imaging sensors. Available at: http://www.ipi.uni-hannover.de/fileadmin/institut/pdf/isprs-Hannover2009/Jimenez_Berni-155.pdf. Accessed 11 Jan. 2016.

Biber,Peter, Ulrich Weiss, Michael Dorna, and Amos Albert. 2012. Navigation system of the autonomous agricultural robot “BoniRob”. Available at: http://www.cs.cmu.edu/~mberge. Accessed 15 August 2016.

Biskup, Bernhard, Hanno Scharr, Ulrich Schurr, and Uwe Rascher.2007. A stereo imaging system for measuring structural parameters of plant canopies. Plant, Cell and Environment (2007), 30:1299–1308.

Blackmore, Simon, Bill Stout, Maohua Wang, and Boris Runov.2005. Robotic agriculture—the future of agricultural mechanization?. 5th European Conference on Precision Agriculture, ed. J. Stafford, V. The Netherlands, Wageningen Academic Publishers. 2005, 621-628.

Bojacá, Carlos Ricardo, Sady Javier García, Eddie Schrevens.2011. Analysis of Potato Canopy Coverage as Assessed Through Digital Imagery by Nonlinear Mixed Effects Models. Potato Research (2011), 54:237–252.

Breccia G., and Nestares G..2014. Next-generation phenotyping in plants: old problems, new promises. Journal of Basic & Applied Genetics, 25(1):5-8.

Busemeyer, L., A. Ruckelshausen, K. Möller, A.E. Melchinger, K. V Alheit, H.P. Maurer, V. Hahn, E. a Weissmann, J.C. Reif, and T. Würschum. 2013. Precision phenotyping of biomass accumulation in triticale reveals temporal genetic patterns of regulation. Sci. Rep. 3: 2442.

Campos, H, M Cooper, JE Habben, GO Edmeades, and JR Schussler. 2004. Improving drought tolerance in maize: a view from industry. Field Crops Research, 90 (1):19-34.

Cembali, T., J.R. Folwell, T. Ball, and D.D. Clary. 2005. Economic comparison of selective and non-selective mechanical harvesting of asparagus. ASAE Paper No. 053003. St. Joseph Mich:ASAE.

Chaivivatrakul, Supawadee, Lie Tang, Matthew N. Dailey, Akash D. Nakarmi.2014. Automatic morphological trait characterization for corn plants via 3D holographic reconstruction. Computers and Electronics in Agriculture, 109 (2014):109–123.

Chapman, Scott C. , Torsten Merz , Amy Chan , Paul Jackway , Stefan Hrabar ,M. Fernanda Dreccer , Edward Holland , Bangyou Zheng , T. Jun Ling, and Jose Jimenez-Berni .2014. Pheno-Copter: A Low-Altitude, Autonomous Remote-Sensing Robotic Helicopter for High-Throughput Field-Based Phenotyping. Agronomy 2014, 4:279-301.

Chi, Y.T., and P.P. Ling.2004. Fast fruit identification for robotic tomato picker. ASAE Paper No. 043083. St. Joseph Mich:ASAE.

Choi, K.H., S.K. Han, S.H. Han, K.H. Park, K.S. Kim, and S. Kim. 2015. Morphology-based guidance line extraction for an autonomous weeding robot in paddy fields. Computers and electronics in agriculture, 113,266-274.Thuilot, B., C. Cariou, P. Martinet, and M. Berducat. 2002. Automatic guidance of a farm tractor relying on a single CP-DGPS. Autonomous robots, 13(1):53-71.

Clausi, David A.2002. An analysis of co-occurrence texture statistics as a function of grey level quantization. Can.J Remote Sensing, 28(1):45-62.

Cobb, J.N., G. Declerck, A. Greenberg, R. Clark, and S. McCouch. 2013. Next-generation phenotyping: requirements and strategies for enhancing our understanding of genotype-phenotype relationships and its relevance to crop improvement. Theor. Appl. Genet. 126(4): 867–87.

Cointault, F., D. Guren, J-P. Guillemin, and B. Chopinet. 2008. In-field Triticum aestivum head counting using colour-texture image analysis. New Zealand Journal of Crop and Horticultural Science, 2008(36): 117-130.

Cointault, Frédéric, Ludovic Journaux, Johel Miteran, Marie-France Destain, and Xavier Tizon.2008. Improvements of image processing for wheat head counting. Available at: http://orbi.ulg.ac.be/bitstream/2268/30556/1/Cointault.pdf. Accessed 27 January 2015.

Crain, Jared L., Yong Wei, Jared Barker III, Sean M. Thompson, Phillip D. Alderman, Matthew Reynolds, Naiqian Zhang, and Jesse Poland.2016. Development and Deployment of a Portable Field Phenotyping Platform. Crop science, 56:1-11.

Deery, David, Jose Jimenez-Berni, Hamlyn Jones, Xavier Sirault, and Robert Furbank. 2014. Proximal Remote Sensing Buggies and Potential Applications for Field-Based Phenotyping. Agronomy 2014, 5:349-379.

Emmi, Luis, Mariano Gonzalez-de-Soto, Gonzalo Pajares, and Pablo Gonzalez-de-Santos. 2014. New Trends in Robotics for A griculture: Integration and Assessment of a Real Fleet of Robots. The Scientific Woreld Journal, 2014.

Fan, W., Y.Gai, X.Xu, B.Yan. 2013. The spatial scaling effect of the discrete-canopy effective leaf area index retrieved by remote sensing. Sci. China headth Sci., 2013(56):1548–1554.

Fanourakis, Dimitrios , Christoph Briese, Johannes FJ Max, SilkeKleinen, Alexander Putz, Fabio Fiorani, Andreas Ulbrich, and Ulrich Schurr.2014. Rapid determination of leaf area and plant height by using light curtain arrays in four species with contrasting shoot architecture. Plant Methods 2014, 10:9.

Feng ,Rui , Yushu Zhang, Wenying Yu , Wei Hu, Jinwen Wu, Ruipeng Ji , Hongbo Wang, Xianli Zhao.2013. Analysis of the relationship between the spectral characteristics of maize canopy and leaf area index under drought stress. ActaEcologicaSinica, 33 (2013): 301–307.

Feng, W., X. Yao, Y. Zhu, Y.C. Tian, W.X. Cao.2008. Monitoring leaf nitrogen status with hyperspectral reflectance in wheat. Europ. J. Agronomy, 28 (2008): 394–404.

Fiorani, Fabio, and Ulrich Schurr. 2013. Future Scenarios for Plant Phenotyping. Annual Review of Plant Biology, 64 (1):267-291.

Fitzgerald, Glenn, Daniel Rodriguez, Garry O'Leary. 2010. Measuring and predicting canopy nitrogen nutrition in wheat using a spectral index—the canopy chlorophyll content index (CCCI). Field Crops Research, 116 (2010): 318–324.

Ford, Adrian, and Alan Roberts. 1998. Colour space conversion. Available at: http:// www.poynton. com/ PDFs/coloureq.pdf. Accessed on June 27 2016.

Frédéric, C., J. Ludovic, R. Gilles, G. Christian, O. David, D. Marie-France, G. Nathalie, G. Gilbert, L. Olivier, and M. Ambroise. 2012. Texture, color and frequential proxy-detection image processing for crop characterization in a context of precision agriculture. . Available at: http://cdn.intechopen.com/pdfs-wm/36493.pdf. Accessed on June 27 2016.

Furbank, Robert T., and Mark Tester. 2011. Phenomics – technologies to relieve the phenotyping bottleneck. Trends in Plant Science December, 16(12):635-644.

Germain, C., Rousseaud, R., & Grenier, G. 1995. Non Destructive Counting of Wheatear with Picture Analysis.Proceeding of 5th IEE International conference on Image Processing and its Applications, Edimburg, UK, 1995.

Green JM, Appel H, MacNealRehrig E, Harnsomburana J, Chang J-F, Balint-Kurti P, et al. 2012.PhenoPhyte: a flexible affordable method to quantify 2D phenotypes from imagery. Plant Methods, 2012:8:45.

Grift, Toney, Qin Zhang, Naoshi Kondo, and K.C. Ting. 2008. A review of automation and robotics for bio-industry. Journal of Biomechatronics Engineering, 1(1): 37-54.

Großkinsky, Dominik K., Jesper Svensgaard, Svend Christensen, and Thomas Roitsch. 2015. Plant phenomics and the need for physiological phenotyping across scales to narrow the genotype-to-phenotype knowledge gap. Journal of Experimental Botany, 66(18): 5429–5440.

Gutierrez, M., M.P. Reynolds, and A.R. Klatt. 2010. Association of water spectral indices with plant and soil water relations in contrasting wheat genotypes. J. Exp. Bot. 61(12): 3291–303.

Haberland, J.A., P.D. Colaizzi, M.A. Kostrzewski, P.M. Waller, C.Y. Choi, F.E. Eaton, E.M. Barnes, and T.R. Clarke. 2010. AgIIS, Agricultural Irrigation Imaging System. Appl. Eng. Agric. 26(2): 247–253.

Hammer, Graeme, Mark Cooper, Franc¸ois Tardieu, Stephen Welch, Bruce Walsh, Fred van Eeuwijk, Scott Chapman, and Dean Podlich. 2006. Models for navigating biological complexity in breeding improved crop plants. TRENDS in Plant Science, 11(12): 1360-1385.

Hannan, W., and F.T. Burks. 2004. Current developments in automated citrus harvesting. ASAE Paper No. 043087. St. Joseph Mich:ASAE.

Haralick, Robert M., K. Shanmugam, and I. Dinstein. 1973. Textural features for image classification. IEEE Transactions on Systems, Man and Cybernetics, 3(6):610–621.

Hartmann, Anja, Tobias Czauderna, Roberto Hoffmann, Nils Stein, and Falk Schreiber. 2011. HTPheno: An image analysis pipeline for high-throughput plant phenotyping. BMC Bioinformatics 2011, 12:148.

Hiremath, Van der Heijden, G. W. A. M., Van Evert, F. K., Stein, A., and Ter Braak, C.J.F. 2014. Laser range finder model for autonomous navigation of a robot in a maize field using a particle filter. Computers and Electronics in Agriculture,100,41-50.

Houle, David, Diddahally R Govindaraju, and Stig Omholt. 2010. Phenomics: the next challenge. Nature Reviews Genetics, 11 (12):855-866.

Huang, M., W. Yu, and D. Zhu. 2012. An improved image segmentation algorithm based on the OTSU method. 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 135-139. Washington, D.C.: IEEE Computer society.

Jackson, Lee, and Jack Williams. 2006. Growth and development of small grains. Available at: http://anrcatalog.ucanr.edu/pdf/8165.pdf. Accessed 23 June 2016.

Jared Barker III, Naiqian Zhang, Joshua Sharon, Ryan Steeves, Xu Wang, Yong Wei, and Jesse Poland. 2016. Development of a field-based high-throughput mobile phenotyping platform. Computers and electronics in agriculture, 122:74-85.

Jiang, Ni, Wanneng Yang, Lingfeng Duan, Xiaochun Xu, Chenglong Huang, Qian Liu.2012. Acceleration of CT reconstruction for wheat tiller inspection based on adaptive minimum enclosing rectangle. Computers and Electronics in Agriculture, 85 (2012): 123–133.

Kakran, Anil, Rita Mahajan.2012. Monitoring Growth of Wheat Crop using Digital Image Processing. International Journal of Computer Applications. 50(10):18-22.

Keener, M.E., and P.L. Kircher. 1983. The use of canopy temperature as an indicator of drought stress in humid regions. Agric. Meteorol. 28: 339–349.

Kempthorne D, Turner IW, Belward JA, McCue SW, Barry M, Young JA, Dorr GJ, Hanan J, Zabkiewicz JA. 2015. Surface reconstruction of wheat leaf morphology from three-dimensional scanned data. Functional Plant Biology, 42:444–451.

Khan, A. M., Ravi. S.. 2013. Image Segmentation Methods: A Comparative Study. International Journal of Soft Computing and Engineering (IJSCE), 3(4):84-92.

Kloth, Karen J, Cindy JM ten Broeke, Manus PM Thoen, Marianne Hanhart-van den Brink. 2015. High-throughput phenotyping of plant resistance to aphids by automated video tracking. Plant Methods (2015) 11:4.

Kolbach, R., K.L. Kerrisk, S.C. Carcia, and N.K. Dhand. 2013. Effects of bail activation sequence and feed availability on cow traffic and milk harvesting capacity in a robotic rotary dairy. Journal of dairy science, 96(4):2137-2146.

Kondo, N., K. Ninomiya, S. Hayashi, T. Ohta, and K. Kubota. 2005. A new challenge of robot for harvesting strawberry grown on table top culture. ASAE Paper No. 043083. St. Joseph Mich:ASAE.

Kumar, Jitendra, Aditya Pratap, and Shiv Kumar.2015. Phenomics in Crop Plants: Trends, Options and Limitations- Traits for Phenotyping. New York,USA:Springer.

Lang, A. R. G. and R. E. McMurtrie. 1992. "Total leaf areas of single trees of Eucalyptus grandis estimated from transmittances of the sun's beam." Agriculture and Forest Meteorology 58: 79-92.

Li, Lei, Qin Zhang, and Danfeng Huang. 2014. A Review of Imaging Techniques for Plant Phenotyping. Sensors 2014, 14, 20078-20111.

Liu, T., C. Sun, L. Wang, X. Zhong, X. Zhu, and W. Guo. 2014. In-field wheatear counting based on image processing technology. Transactions of Chinese Society for Agricultural Machinery, 45(2):282-290.

Lorenz, Aaron J., Shiaoman Chao, Franco G. Asoro, Elliot L. Heffner, Takeshi Hayashi, Hiroyoshi Iwata, Kevin P. Smith, Mark E. Sorrells, and Jean-Luc Jannink. 2011. Genomic selection in plant breeding: knowledge and prospects. Adv. Agron. 110, 77–123.

Luis Emmi, Mariano Gonzalez-de-Soto, Gonzalo Pajares, and Pablo Gonzalez-de-Santos.2014. New Trends in Robotics for Agriculture: Integration and Assessment of a Real Fleet of Robots. The Scientific World Journal, 2014:1-21.

Lukina, E.V., M.L. Stone, and W. R. Raun. 1999. Estimating vegetation coverage in wheat using digital images. JOURNAL OF PLANT NUTRITION, 22(2):341-350.

Macfarlane, C., G.N. Ogden. 2012. Automated estimation of foliage cover in forest understory from digital NADIR images. Methods Ecol. Evol., 2012(3):405–415.

Merz, T., and S. Chapman. 2011. Autonomous unmanned helicopter system for remote sensing missions in unknown environments. Int. Arch. … XXXVIII(September): 14–16.

Montes, J.M., F. Technow, B.S. Dhillon, F. Mauch, A.E. Melchinger. 2011. High-throughput non-destructive biomass determination during early plant development in maize under field conditions. Field Crops Res 2011, 121:268–273.

Morrell, P.L., E.S. Buckler, and J. Ross-Ibarra. 2011. Crop genomics: advances and applications. Nat. Rev. Genet. 13(2): 85–96.

Muhammad, Aslam Khan, Muhammad Abid, Nazim Hussain and Tahir Imran.2005.Growth analysis of wheat (Triticum aestivum L.) cultivars under saline conditions. Int. J. Agri. Biol., 7(3):508-510.

Myles, S., J. Peiffer, P.J. Brown, E.S. Ersoz, Z. Zhang, D.E. Costich, and E.S. Buckler. 2009. Association mapping: critical considerations shift from genotyping to experimental design. Plant Cell 21(8): 2194–202.

Nabeel Kadim Abid AL-SAHIB, and Salih Rashid MAJEED. 2012. Environmental mobile robot based on artificial intelligence and visual perception for weed elimination. INCAS BULLETIN, 4(4):11-25.

Nagel, Kerstin A., Alexander Putz, Frank Gilmer, Kathrin Heinz, Andreas Fischbach, Johannes Pfeifer, Marc Faget, Stephan Blossfeld, Michaela Ernst, Chryssa Dimaki, Bernd Kastenholz, Ann-Katrin Kleinert, Anna Galinski, Hanno Scharr, Fabio Fiorani, and Ulrich Schurr. 2012. GROWSCREEN-Rhizo is a novel phenotyping robot enabling simultaneous measurements of root and shoot growth for plants grown in soil-filled rhizotrons. Functional Plant Biology, 39 (11):891-904.

Pedersen S. M, Fountas S., and Blackmore S.. 2008. Agricultural robots—applications and economic perspectives, Service Robot Applications, Yoshihiko Takahashi (Ed), ISBN: 978-953-7619-00-8, In Tech, Available at: http://www.intechopen.com/books/service_robot_applications/agricultural_robots_applications_and _economic_perspectives. Accessed 11 Jan. 2016.

Pengelly JJ, Sirault XR, Tazoe Y, Evans JR, Furbank RT, von Caemmerer S. 2010. Growth of the C4 dicot Flaveriabidentis: photosynthetic acclimation to low light through shifts in leaf anatomy and biochemistry. Journal of Experimental Botany, 61: 4109–4122.

Plant Image Analysis. 2016. Available at: http://www.plant-image-analysis.org/, accessed on17Jan.2016.

Prashar, A.Yildiz, J., McNicol, J.W., Bryan, G.J., Jones, H.G.2013. Infra-red thermography for high throughput field phenotyping in Solanum tuberosum. PLoS One 2013. 8(6):e5816.

Qi, Chengming.2014. Maximum entropy for image segmentation based on an adaptive particle swarm optimization. Appl. Math. Inf. Sci. 8(6):3129-3135.

Rajasingh, Hannah, Arne B. Gjuvsland, Dag Inge Vage, and Stig W. Omholt. 2008. When Parameters in Dynamic Models Become Phenotypes: A Case Study on Flesh Pigmentation in the Chinook Salmon (Oncorhynchus tshawytscha). Genetics 179: 1113–1118.

Ramfos, Alexis, Andreas Gazis, and George Katselis.2012. Development and evaluation of an automated digital image analysis software for obtaining seagrass leaf metrics. Botanica Marina, 55(6): 601–610.

Ray, Deepak K., Nathaniel D. Mueller, Paul C. West, Jonathan A. Foley. 2013. Yield trends are insufficient to double global crop production by 2050. PLoS ONE 8, e66428.

Ray, Deepak K., Navin Ramankutty, Nathaniel D. Mueller, Paul C. West, and Jonathan A. Foley. 2012. Recent patterns of crop yield growth and stagnation. Nat. Commun. 3, 1293.

Rubens A.Tabile, Eduardo P. Godoy, Robson R. D. Pereira, Giovana T. Tangerino, Arthur J. V. Porto, and Ricardo Y. Inamasu.2011. Design and development of the architecture of an agricultural mobile robot. Eng. Agríc., Jaboticabal, 31(1):130-142.

Sankaran, Sindhuja, Lav R. Khot, and Arron H. Carter. 2015. Field-based crop phenotyping: Multispectral aerial imaging for evaluation of winter wheat emergence and spring stand. Computers and Electronics in Agriculture, 118 (2015):372–379.

Soh, Leen-Kiat, and Costas Tsatsoulis.1999. Texture analysis 0f SAR sea ice imagery using gray level co-occurrence matrices. IEEE Transaction on geoscience and remote sensing, 37(2):780-795.

Stajnko, D., M. Lakota, and M. Hočevar. 2004. Estimation of number and diameter of apple fruits in an orchard during the growing season by thermal imaging. Computers and Electronics in Agriculture 42 (2004): 31–42.

Steinfeld, A. 2004. Interface lessons for fully and semi-autonomous mobile robots. In paper presented at the IEEE international conference on robotics and automation.

Tester, Mark, and Peter Langridge. 2010. Breeding technologies to increase crop production in a changing world. Science, 327 (5967):818-822.

Thorp, K.R., M.A. Gore , P. Andrade-Sanchez , A.E. Carmo-Silva, S.M. Welch, J.W. White , A.N. French.2015. Proximal hyperspectral sensing and data analysis approaches for field-based phenomics. Computers and Electronics in Agriculture. 118(2015):225-236.

Tisné, Sébastien, Yann Serrand, Liên Bach, Elodie Gilbault, Rachid Ben Ameur, Hervé Balasse, Roger Voisin, David Bouchez, Mylène Durand-Tardif, Philippe Guerche, Gaël Chareyron, Jérôme Da Rugna, Christine Camilleri, and Olivier Loudet. 2013. Phenoscope: an automated large-scale phenotyping platform offering high spatial homogeneity. The Plant Journal, 74 (3):534-544.

Tuceryan, Mihran, and Anil K. Jain.1993. Handbook of pattern recognition & computer vision. New Jersey, River Edge: World Scientific Publishing Co., Inc.

USDA. 2016. World agricultural production. Available at: http:// apps. fas.usda.gov/ psdonline /circulars/ production.pdf. Accessed 23 June 2016.

Wang, Jianlun, Jianlei He, Yu Han, Changqi Ouyang, Daoliang Li. 2013. An Adaptive Thresholding algorithm of field leaf image. Computers and Electronics in Agriculture, 96 (2013): 23–39.

Wang, Yuan, Dejian Wang, Gang Zhang, Jun Wang. 2013. Estimating nitrogen status of rice using the image segmentation ofG-R thresholding method. Field Crops Research, 149 (2013):33–39.

White, J., P. Andrade-Sanchez, M.A. Gore, K.F. Bronson, T.A. Coffelt, M.M. Conley, K.A. Feldmann, A.N. French, J.T. Heun, D.J. Hunsaker, M.A. Jenks, B.A. Kimball, R.L. Roth,

R.J. Strand, K.R. Thorp, G.W. Wall, and G. Wang. 2012. Field-based phenomics for plant genetics research. F. Crop. Res. 133: 101–112.

White, J.W., and R. V Bostelman. 2010. Large-area Overhead Manipulator for Access of Fields.

White, Jeffrey W., Pedro Andrade-Sanchez, Michael A. Gore, Kevin F. Bronson, Terry A. Coffelt, Matthew M. Conley, Kenneth A. Feldmann, Andrew N. French, John T. Heun, Douglas J. Hunsaker, Matthew A. Jenks, Bruce A. Kimball, Robert L. Roth, Robert J. Strand, Kelly R. Thorp, Gerard W. Wall, and Guangyao Wang. 2012. Field-based phenomics for plant genetics research. Field Crops Research, 133 (0):101-112.

WinterGreen.2016. Agricultural Robots:--Markets Reach $16.3 Billion By 2020. Available at: http://wintergreenresearch.com. Accessed 11 August 2016.

Yasushi Hashimoto, Haruhiko Murase, Tetsuo Morimoto, and Toru Torii. 2001. Intelligent systems for agriculture in Japan. IEEE control systems magazine, 2001:71-85.

You, Zih-Hao, and Shana Smith.2016. A multi-objective modular design method for creating highly distinct independent modules. Res Eng Design (2016) 27:179–191.

Yu, J., J.B. Holland, M.D. McMullen, and E.S. Buckler. 2008. Genetic design and statistical power of nested association mapping in maize. Genetics 178(1): 539–51.

Zarco-Tejada, P.J., J. a. J. Berni, L. Suárez, G. Sepulcre-Cantó, F. Morales, and J.R. Miller. 2009. Imaging chlorophyll fluorescence with an airborne narrow-band multispectral camera for vegetation stress detection. Remote Sens. Environ. 113(6): 1262–1275.

Zeng, Yelu , Jing Li , Qinhuo Liu , Ronghai Hu, Xihan Mu, Weiliang Fan, Baodong Xu, Gaofei Yin , and Shengbiao Wu.2015. Extracting Leaf Area Index by Sunlit Foliage Component from Downward-Looking Digital Photography under Clear-Sky Conditions. Remote Sens., 2015(7):13410-13435.

Zhao, B., Z. Song, W. Mao. 2009. Agriculture extra-green image segmentation based on particle swarm optimization and K-Means clustering. Transactions of the Chinese Society for agricultural machinery, 40(8):166-169.

Zhao, De-An, Jidong Lu, Wei Ji, Ying Zhang, and Yu Chen.2011. Design and control of an apple harvesting robot. Biosystems engineering, 110(2011):112-122.

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

Zhu, C., and P. Zhou. 2015. Target extraction of green crops under the influence of light. Available at: www.paper.edu.cn/. Accessed on June 27 2016.

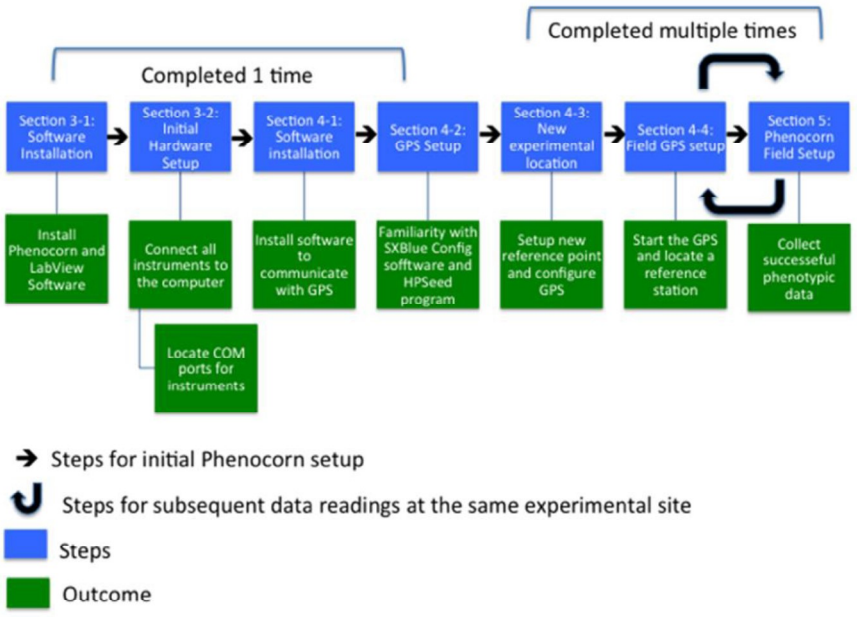
Appendix A-Phenocorn Components and Technical Specifications

Instrument	Company	Field of View	Output	Output Type	Other Specifications	Sensor Cost†
Aspire 4830	Acer Inc., San Jose, CA				Windows 7, 2.4 Ghz process, 8 GB RAM, 64 bit	$750
GreenSeeker	Trimble, CA	1cmX60cm	10 Hz/ 38400 baud	RS-232	RS-232 to USB converter used	$4000
SXBlue III-L	Geneq, Montreal, Canada		10 Hz / 19200 baud	USB	Omnistar G2 for 95% accuracy <= 10cm	$6695 $1400 Omnistar G2 Subscription
CT Series Thermometer	MicorEpsilon, Raliegh, NC	1:2	10 Hz/ 9600 baud	analog	Analog to Digital (AD) converter, Temperature resolution 0.1 °C, System accuracy: ±1 °C	$355 $75 AD converter
C920 Logitech			~3 Hz	USB	HD 1080p	$100

†Represents approximate cost, may be different

Appendix B-User's Manual

1. User Guide Documentation



2. Introduction

This guide was made with Phenocorn Software Version 100314. The guide walks through starting the SXBlue GPS and the Phenocorn software program, checking data integrity, and common issues associated with the platform. Many of the steps describe and then demonstrate the steps using screen captures of the actions. Screen shots often include either the entire Windows desktop, or only of the software in use. This should help the reader, and provide an opportunity to find any differences that may occur. While we have taken every effort to insure its accuracy exact steps may be different based on different computers and operating systems. Our goal is that this document should prepare the reader to collect high quality phenotypic data in the field, and detail many of the problems and solutions that have occurred while operating the equipment.

CAUTION: Due to the nature of the LabView executable file, any hardware substitutions may not be compatible. At a minimum substituted hardware would have to conform to the baud rate and data output as listed hardware.

The manual has been put together in chronological fashion from first assembling components to phenotyping. Thus, more detailed descriptions are provided headlier in the manual, and subsequent steps that repeat a process are only referenced, with the reference by section and subsection of the initial step(s). If a particular step does not seem familiar, information in previous sections can add detail. Even though the manual is highly documented, and will be crucial the first several uses, it should become no more than a reference guide for the experienced user.

3. Abbreviations

SBAS, satellite based augmentation system; GPS, global positioning system; BER, bit error rate; NMEA, National Marine Electronics Association; GUI, graphical user interface: IRT, infrared thermometer; NDVI, normalized difference vegetative index

4. Initial Hardware and Software Setup

Summary: This section shows the installation and start up for the first time that the Phenocorn program is used. Once the initial setup has been completed, field data collection can begin by proceeding to Section 6 Phenocorn Field Setup.

4.1 Software Installation

These steps were completed using Microsoft Windows 7, and show how to install the Phenocorn software.

4.1.1 Download the Phenocorn Program software from

http://people.beocat.cis.ksu.edu/~jcrain/

4.1.2 Unzip the downloaded software into a folder (C:drive used in example), the unzipped file will create a folder named “PHENOCORN100314”.

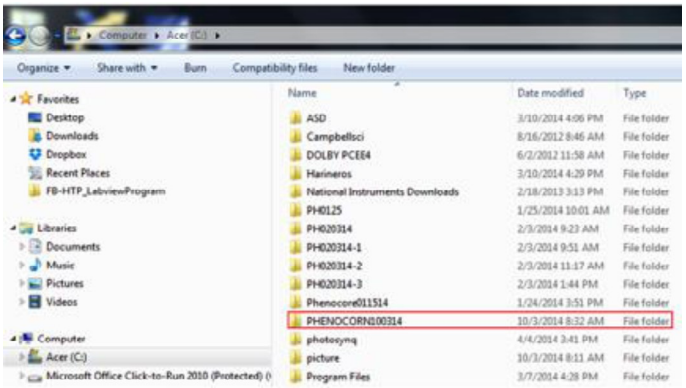


Figure B.1 Unzipped "PHENOCORN100314" file.

4.1.3 Open the “PHENOCORN100314” folder and install the program by:

Clicking My Installer → Volume → setup

Depending on the security software on the computer, it may be necessary to allow the software to install.

4.1.4 A dialog box will appear prompting the user to click “Next”

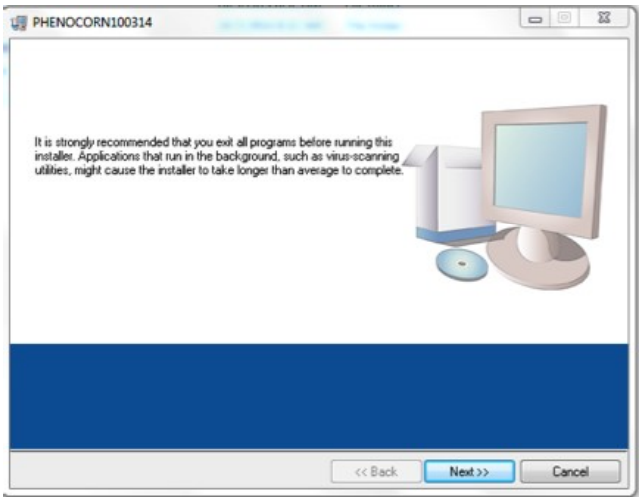


Figure B.2 Phenocorn software installation dialog box, prompting user to click next to continue the installation.

4.1.5 A dialog box asking for destination directory for both the Phenocorn program and the National Instruments software opens. If the default destinations are not desired make changes as necessary. The National Instruments products should be installed in their own unique directory. Once the destination is set click “Next”.

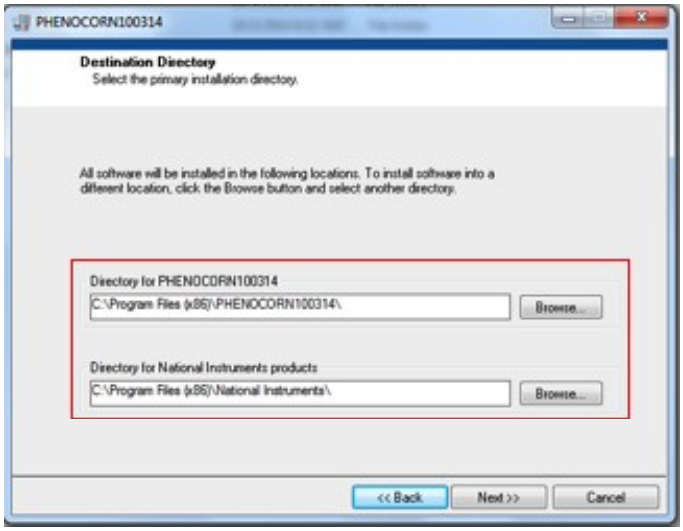


Figure B.3 Dialog box prompting the user for destination directories. Default can be used, or destination can be modified by selecting “Browse”. A destination for both the Phenocorn software and National Instruments Software is required.

4.1.6 A dialog box prompting the user to accept the software license appears. This is the National Instruments license, which must be accepted in order to run the Phenocorn program. The Phenocorn program is made using LabView a National Instruments software.

Figure B.4 Dialog box asking user acceptance of terms for National Instruments Software.

4.1.7 Click “Next” in the Start Installation dialog box to install the software. The first dialog box shows the programs to be installed, and the subsequent dialog box shows installation progress. This may take several minutes to complete.

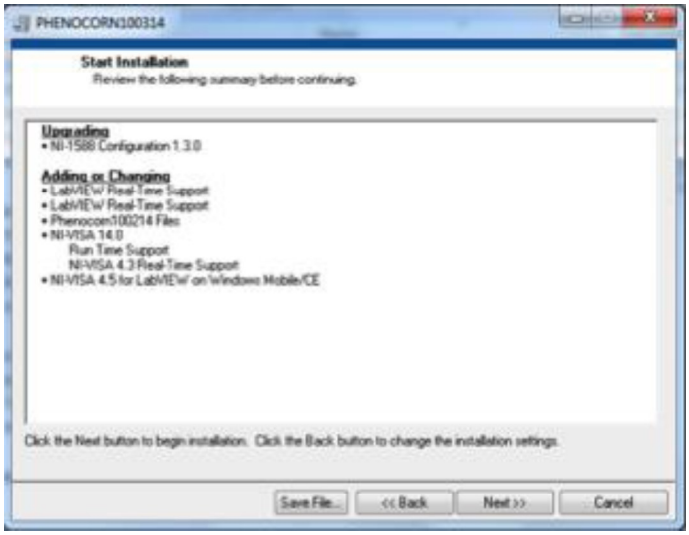


Figure B.5 Installation Dialog box showing programs that will be installed or modified. Clicking “Next” begins installation.

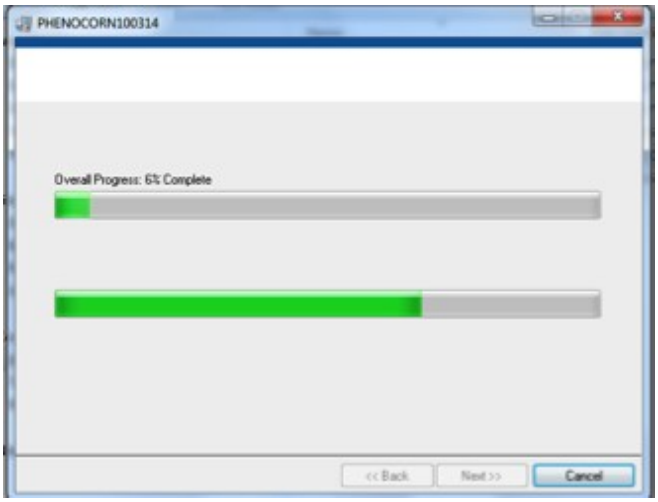


Figure B.6 Progress of the installation of the Phenocorn and LabView Software.

4.1.8 Click “Finish” once the installer is complete.

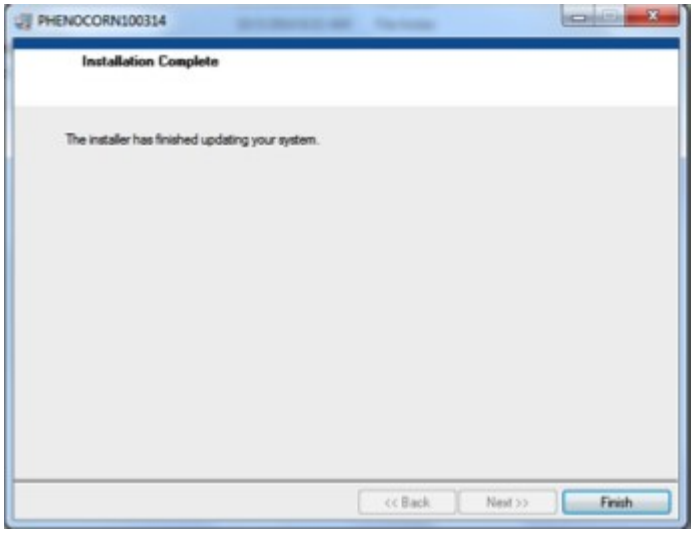


Figure B.7 Completed installation of the Phenocorn Program.

4.1.9 At the end of the install the computer must be restarted.

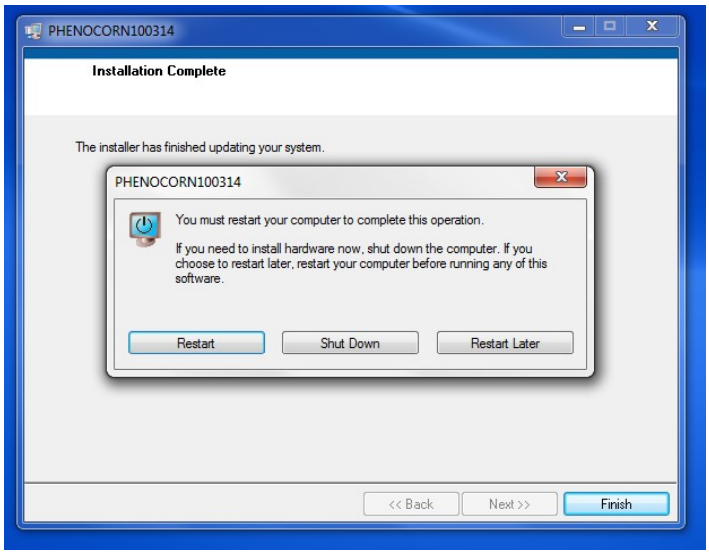


Figure B.8 Restarting the computer after installation

4.1.10 The software has been installed. Clicking on the Windows start menu should now display a new folder named “PHENOCORN100314”.

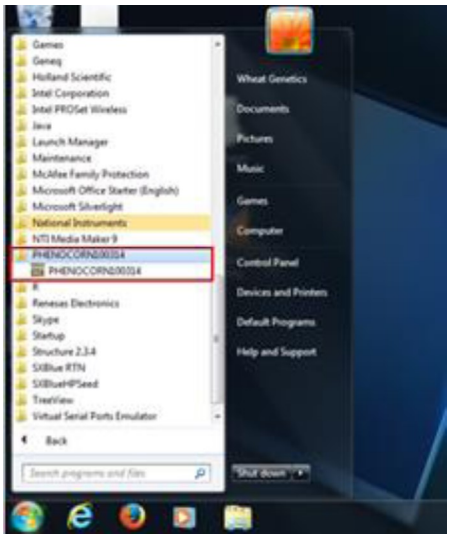


Figure B.9 New software folder “PHENOCORN100314” after installation.

4.2 Initial Hardware Setup

These steps work though how to initially plug in different instruments, via USB, find the instrument COM port numbers, and set the LabView interface to correctly read data from the instruments.

4.2.1 Connect all instruments to the computer using USB ports. Instruments connected include GreenSeeker, SXBlue IIIL GPS, CT Series Thermometer, and C920 Logitech camera.

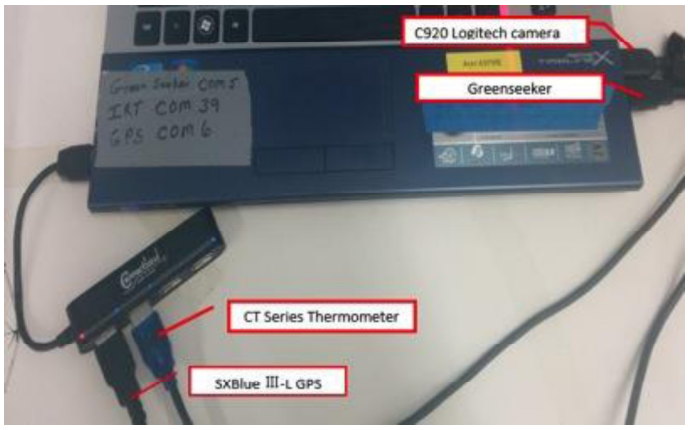


Figure B.10 Connecting all instruments to a lap top computer.

4.2.2 Find the COM port number for each instrument.

1. Click on the **Start** button and then choose **Control Panel**.
2. Click on the **System and Security** link.
3. In the *System and Security* window, click on the **Device Manager** link located under the *System* heading tab.
4. Expand the "Ports (COM & LPT) by clicking on it. The COM number of the instruments will be visible.

Note: The COM port settings may differ based on computer. It is essential to use the correct COM port for each computer so that the data from the Phenocorn can be recorded properly. The following steps illustrate correctly setting the COM port.

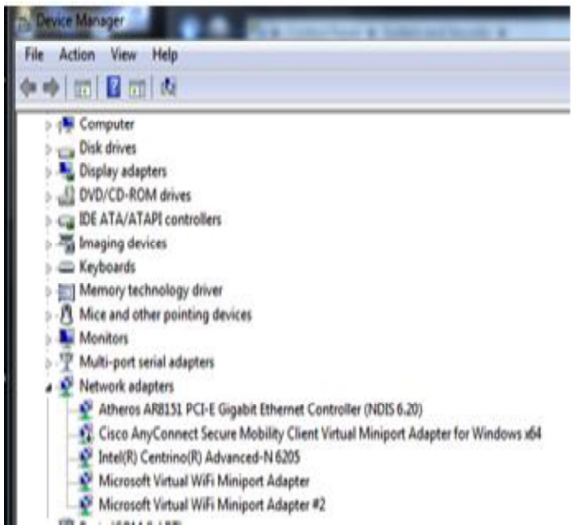


Figure B.11 COM port number for Phenocorn Instruments.

Troubleshooting: The Ports (COM & LPT) may not show all of the USBs. If this happens look for a tab Other Devices. With the computer connected to the internet right click and update driver software Figure B.12.

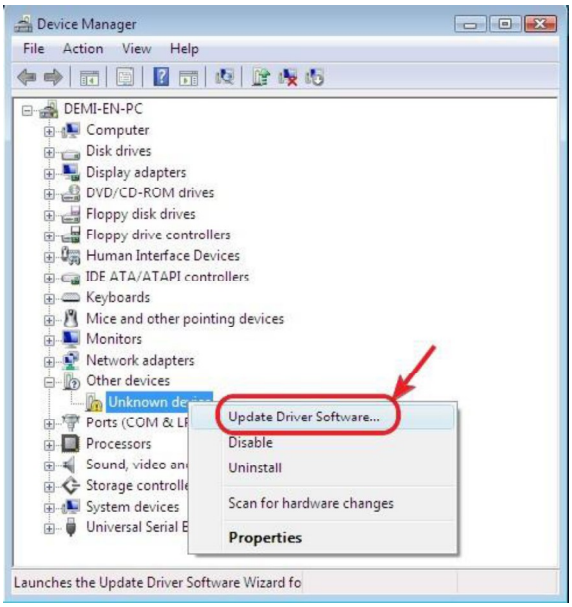


Figure B.12 Updating USB device driver software.

The following table can be used to record COM port information:

Table B.1 Instrument baud rate and COM port specifications. COM port must be found for each individual computer.

Instrument	Company	Output	Output type	COM port
GreenSeeker	Trimble, CA	10HZ/38400 baud	RS-232to USB	
SXBlue III-L	Geneq, Montreal, Canada	10HZ/19200 baud	USB	
CT Serial Thermometer	MicroEpsilon, NC	10HZ/9600 baud	USB	
C920 Camera	Logitech	<3Frames/Second	USB	

4.2.3 To find the COM port of the web camera, open the NI MAX program. NI Max and an Icon was installed with the Phenocorn software installation (Figure B.13). The Measurement&Automation window will open. Click and expand the Devices and Interfaces, then expand the NI IMAQdx Device to find the camera COM number Figure B.14 . The camera port may not be like other ports named COMXX, instead it may be camX for the port number.

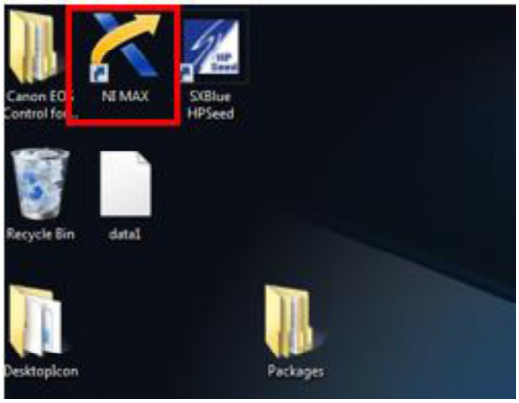


Figure B.13 NI MAX icon on desktop.

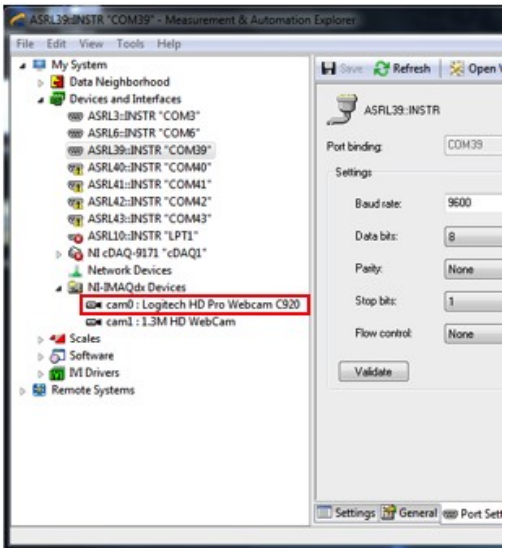


Figure B.14 Measurement & Automation Explorer of NI MAX. NI-IMAQdx Devices shows COM port for the web camera. For this example the port is cam0, highlighted in red box.

4.2.4 Within NI MAX set the COM ports. This allows for communication between sensors, the computer, and software. The NI MAX COM ports must match the computer COM ports for each instrument, and the baud rate within NI MAX must match the sensor baud rate. Set the COM ports and baud rates by

1. Click on the Devices and Interfaces tab. This will display all of the ports and instruments.

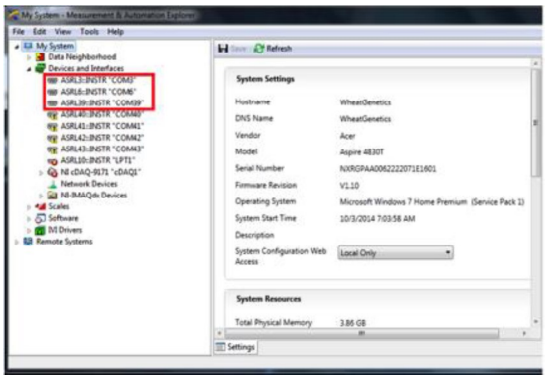


Figure B.15 Devices and Interfaces tab.

2. From the expanded menu in the Devices and Interfaces tab, click on the COM that has the GreenSeeker, this is COM found in Section 4.2.24.2.24.2.24.2.24.2.24.2.2, (example COM3 for GreenSeeker)

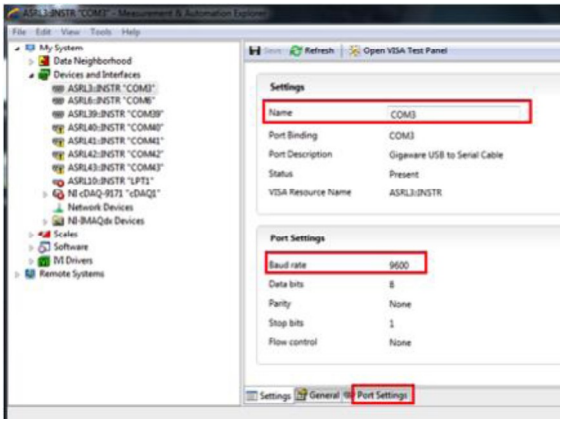


Figure B.16 NI MAX screen after clicking on GreenSeeker COM.

3. Click on the “Port Settings” tab that is in the lower part of the right window in NI MAX. Set the baud rate to 38,400 for the GreenSeeker, and click the “Save” button.

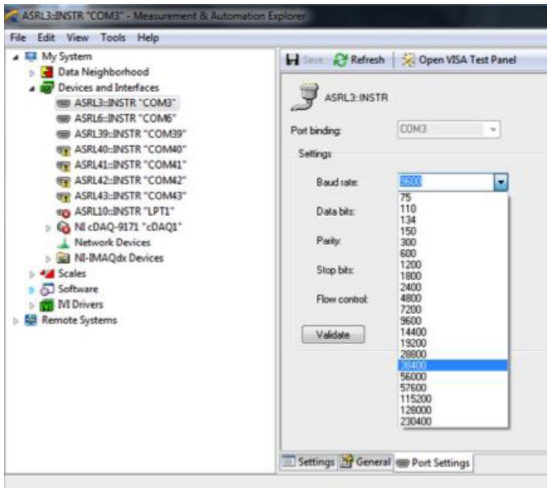


Figure B.17 Setting the GreenSeeker baud rate.

4. Repeat this process for each instrument, selecting the correct COM port, changing the baud rate, and saving the settings. Correct settings are found in Table B.1

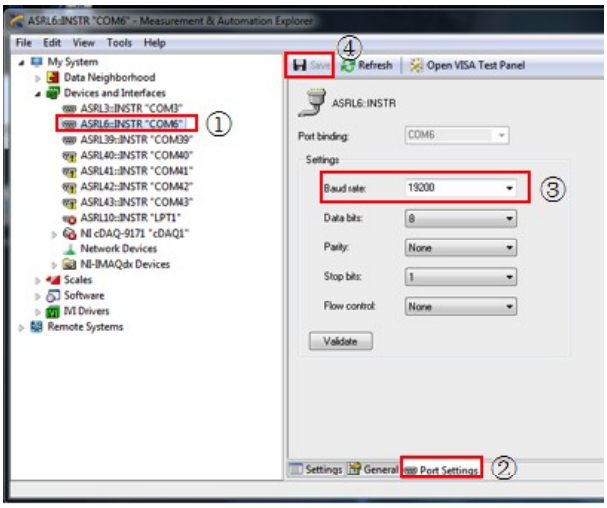


Figure B.18 Setting COM and baud rate settings for each instrument. Using information from Table B.1each instrument is set by: 1. Clicking on the instrument. 2. Changing to the Port Settings tab. 3. Selecting the correct baud rate. 4. Saving the settings by clicking the Save button.

5. GPS Setup:

5.1 Initial Software Setup

5.1.1 Download SXBlue Config Setup (Current version v3.5.5.0) from http://sxbluegps.com/download/

5.1.2 Unzip the download and use the installation wizard to install SXBlue Config, a quick start icon should appear on the desktop.

5.1.3 Download SxBlueHPSeed PC Setup (Current version v.1.2.4852) from http://sxbluegps.com/download/

5.1.4 Unzip, and run the SXBlue HPSeed program. Follow the setup wizard and the installation should also include a desktop icon SXBlue HPSeed.

5.2 Initial GPS Setup

5.2.1 Connect the GPS to the computer using the USB port of the SXBlue, and have the antennae in view of the sky to receive GPS signal.

5.2.2 Click on the SXBlue Config Desktop Icon and find the correct Port and Baud Rate setting. Use the port for your specific operating system, which was found in Section 4.2.2 or Table B.1.

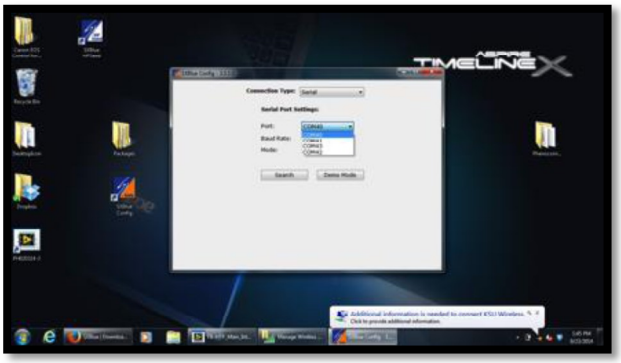


Figure B.19 Selecting correct port and baud rate.

5.2.3 After filling in correct port and baud rate click Search.

Figure B.20 Software querying receiver settings.

5.2.4 Click Advance.

The screen that opens up has multiple tabs that allow monitoring of the GPS quality and signal. The tabs provide many options to adjust the settings within the SXBLue, the main tabs that the Phenocorn uses include:

Position: Parameters including date, time, Latitude, Longitude. Note the Diff Used and Diff Status parameter and value. Diff Used provides an indication that correction is being used, using Omnistar it should be OMNIHP or OMNIG2, if different from theses settings the correction may not be sufficiently accurate. Diff Status provides information of how the signal is being tracked.

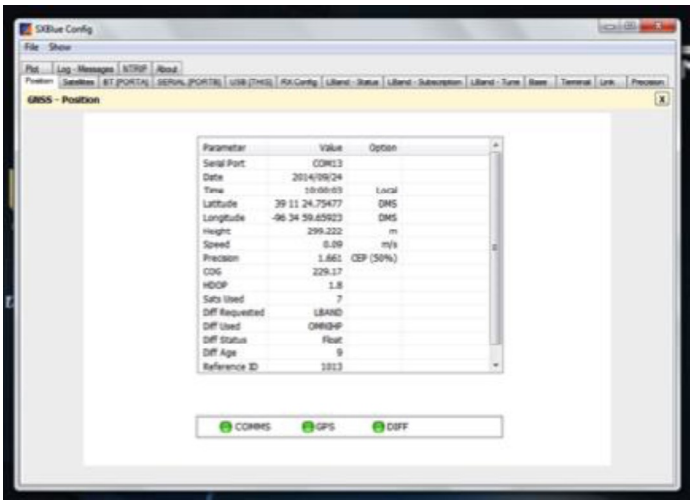


Figure B.21 SXBlue Config home screen, multiple tabs can be used to evaluate system functions.

Satellites: Displays satellites that are being tracked. If no satellites are displayed, or there are very few satellites, the GPS receiver may be obstructed from receiving GPS signal.

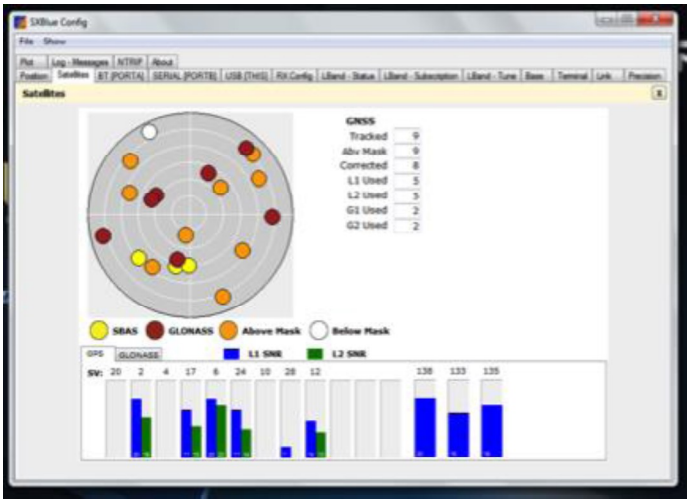


Figure B.22 Satellite information in the SXBlue Config program.

USB(This): Displays what type of NMEA GPS strings are being output by the receiver, and the speed. These can be changed by selecting string of interest and changing from the drop down menu (SectionB.3.). The Phenocorn uses on the $GPGGA string at 10 Hz. If this is changed then follow Section B.2.6B to save the configuration file.

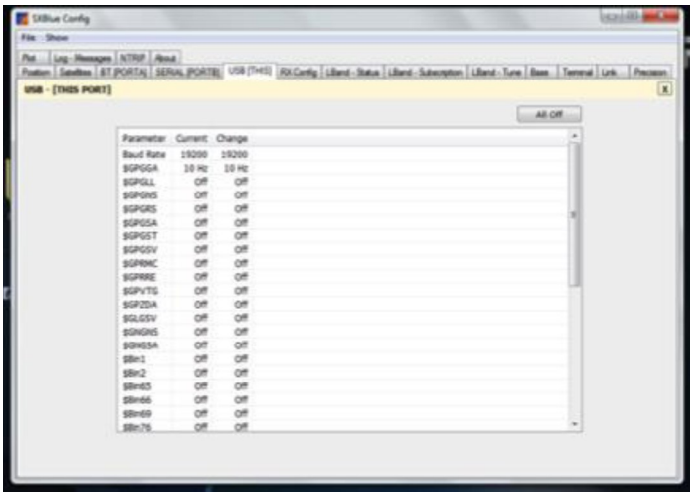


Figure B.23 USB(THIS) port displaying the GPS information that the receiver is sending to the computer.

Precision: Displays the current error (precision status). Use the right hand column that has σ(1σ) to determine precision, there are two measurements one for latitude and one for longitude. At <0.05m (<5cm) this is a standard deviation so the 2σ or 95% accuracy will be under 10 cm.

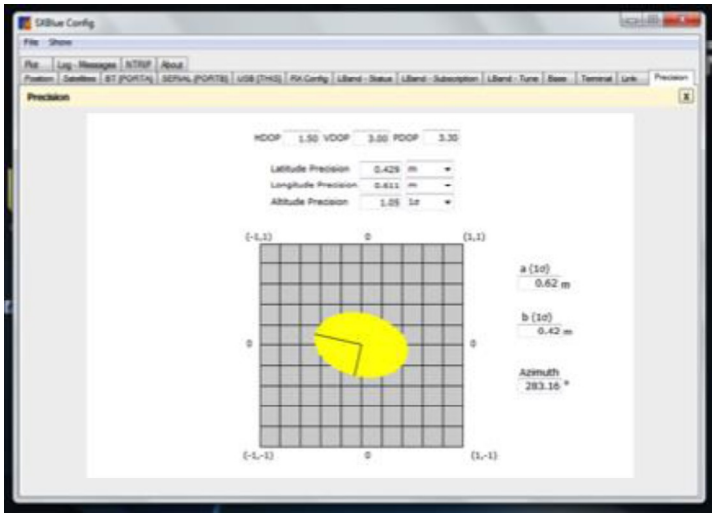


Figure B.24 SXBlue Config Precision tab. Latitude and longitude precision are given as one standard deviation, so this picture has a 95% error rate of 1.24 meters by 0.84 meters.

L band Status: Displays the type of SBAS (satellite based augmentation system) correction that is received and the quality. BER (bit error rate) should be low, in open skies near 0, if BER increases then the satellite signal is potentially being blocked and accuracy will suffer.

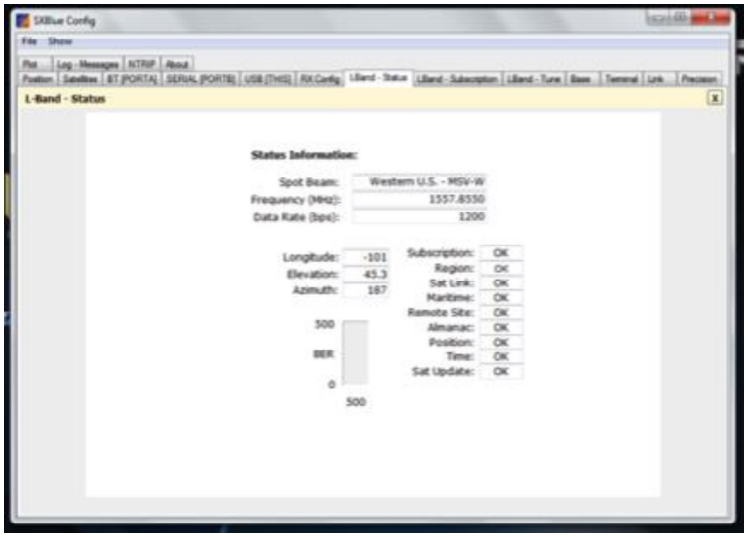


Figure B.25 LBand-Status tab. Note the lack of BER.

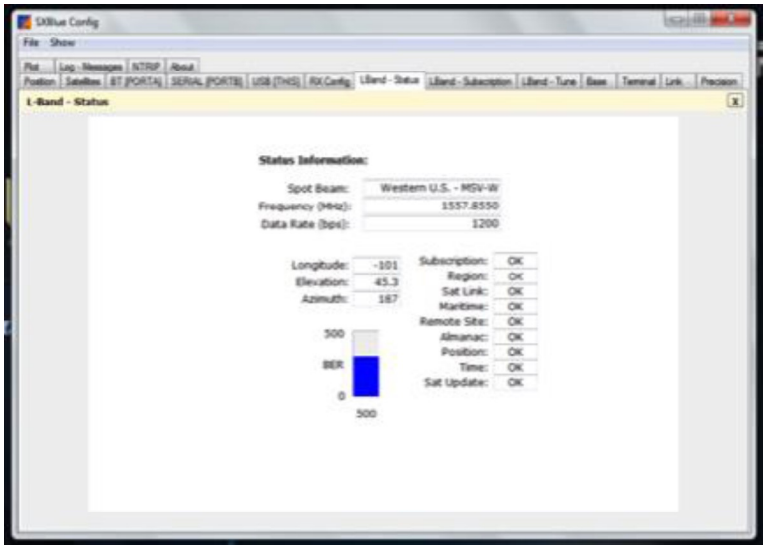


Figure B.26 LBand_status showing BER. This is usually indicative that the GPS antennae is blocked from overhead skies, or GPS quality is low. The user should be cautious of taking data if BER is high, and visually check or perform other data quality assurance processes to insure accuracy.

LBand-Subscription: Displays the Omnistar correction that is available. May not appear until Omnistar is activated.

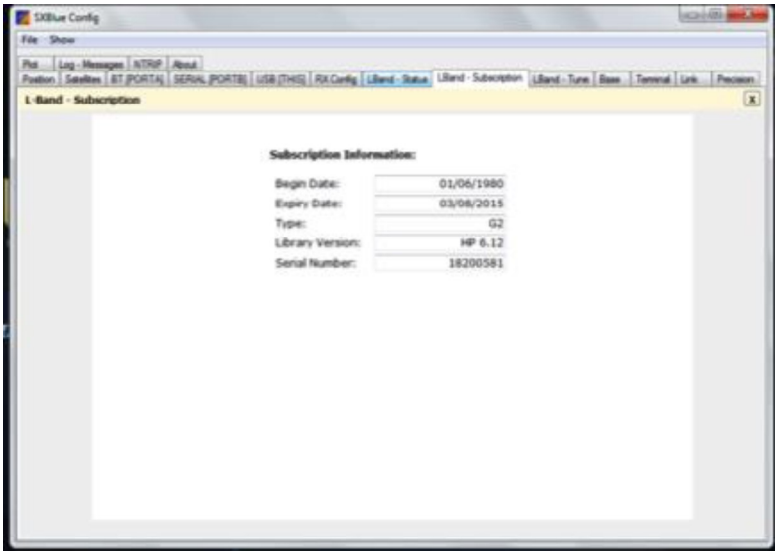


Figure B.27 LBand-Subscrition tab. Displays information about the purchased Omnistar correction.

5.2.5 Initialize Omnistar. Completed over the phone with Omnistar sales representatives. The GPS receiver must be turned on and in view of satellites to acquire the Omnistar correction. Once Omnistar is activated and the receiver locks onto the signal the LBand-Subscription tab will display updated information.

5.2.6 A. To close SXBlue Config without saving any configurations, the default option, click on the X and choose “Disconnect w/o Save”. This will keep the current settings the same.

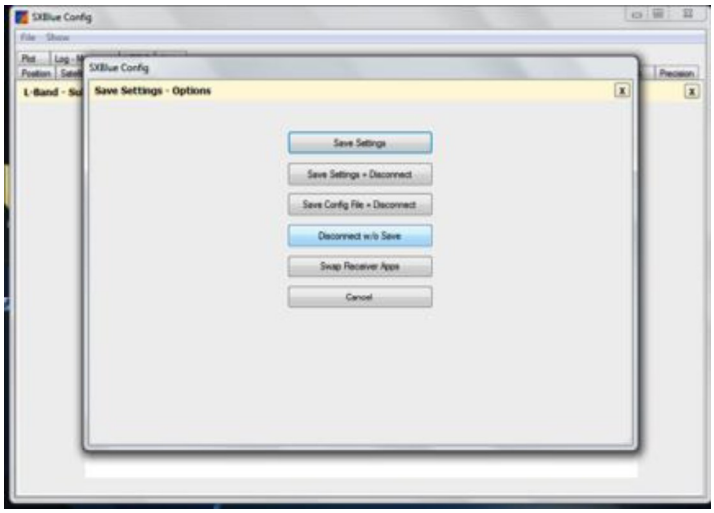


Figure B.28 Closing the SXBlue Config without saving.

B. If a configuration has been changed, ie. output rate or type, and these changes are to be permanent, click the X and then choose “Save Settings + Disconnect”. This option is used rarely,

usually after setting up a new reference point Section 5.3. The next time the system is opened the changed settings will be used.

5.3 New Experiment Location

This section introduces SXBlue HPSeed program that allows a user to setup a known reference point. Using a known reference point will drastically decrease the time of convergence to <10 cm accuracy from greater than 30 minutes to less than 4 minutes. Using this method will save much time for any experiment that will be phenotyped more than once. A reference point should be non-moveable and constant, examples fence post, reference stakes than won't be moved, etc. The reference must remain in the same location, because convergence occurs to the reference. If the reference is moved, convergence occurs but to an inaccurate location.

5.3.1 Start the GPS. From a cold start, it can take up to 30 minutes to converge to 10x10 cm accuracy with Omnistar.

5.3.2 Check the accuracy using SXBlue Config application, and the precision tab.

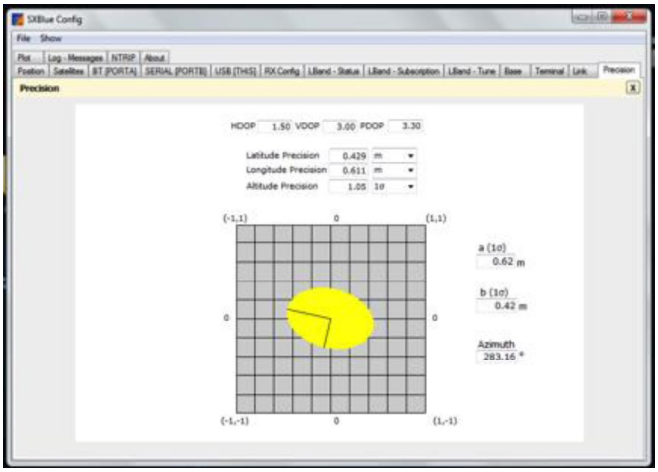


Figure B.29 Checking the accuracy. In this example more time is needed before saving a reference point.

5.3.3 When the accuracy is <0.05m (<5cm) in the precision tab of the of the SXBlue Config, close the SXBlue Config Application Section 5.2.6 A.. A reference point can be seeded with this level of precision. The SXBlue Config program must be closed to allow the SXBLue HPSeed Program to interface with the GPS.

5.3.4 Place the GPS receiver on a reference point. Be careful not to cover the antennae, or GPS signal will be lost. If ever in doubt about the quality of signal or if signal has been lost

use the SXBlue Config application checking the Position tab for type of Diff Used and the precision tab (Figure B.21 and Figure B.23).

5.3.5 Launch the SXBlue HPseed program.

5.3.6 From the drop down menu select the correct port and click Connect.

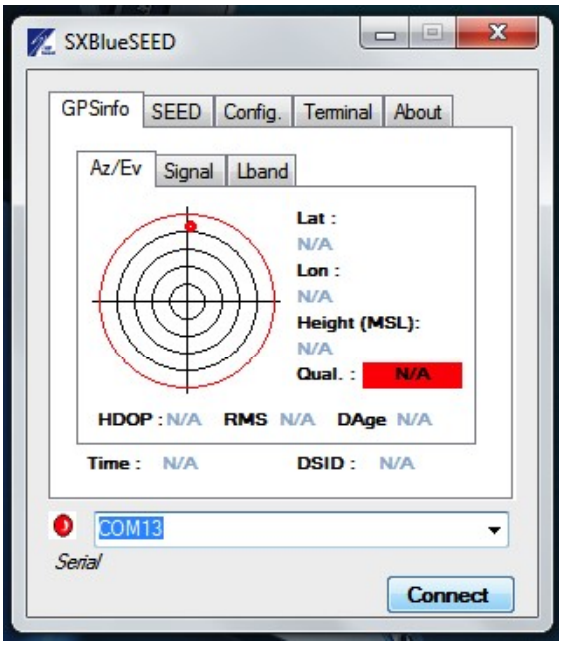


Figure B.30 Selecting correct COM port for GPS. COM port is system specific found in Section 4.2.2.

5.3.6 A Virtual COM is created, click OK.

5.3.7 Enter the correct baud rate and click OK.

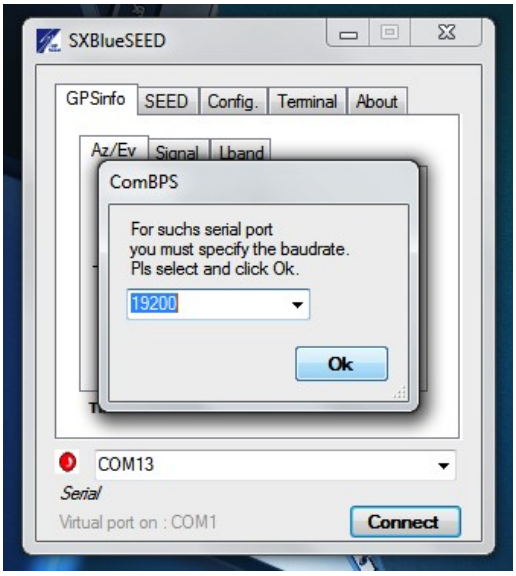


Figure B.31 Selecting correct baud rate (19200).

5.3.8 A window with GSPSinfo, SEED, Config, Terminal, About opens. The GPSInfo is the default tab. When signal is being received the color to the left of the COM port number will be green.

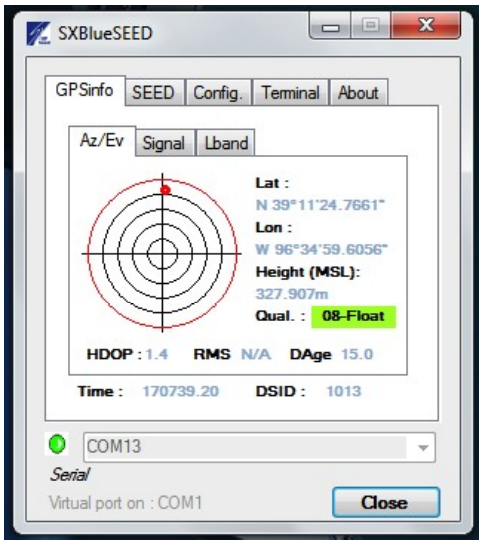


Figure B.32 Home screen of SXBLue HPSeed. Signal is being received as information is displayed and highlighted in green.

5.3.9 Click on the SEED tab and two sub tabs (Log and Goto) will be visible.

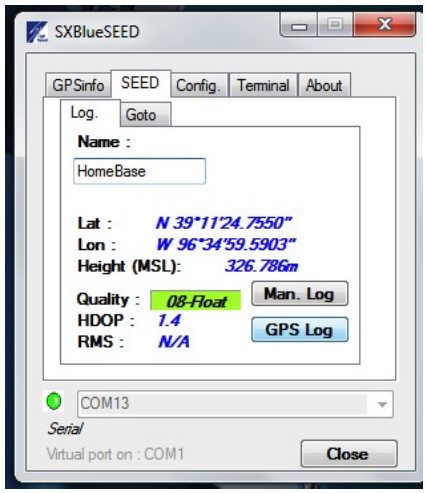


Figure B.33 Seed tab of the SXBlue HPSeed Program, with the sub-tabs Log and Goto. Current screen is on the log sub-tab.

5.3.10 In the Log. Tab, enter a name for the reference station (HomeBase is the example used) and click GPS Log. This will collect readings from the current location, where the GPS

antennae is and save the location. A popup box stating the type of reference data saved may appear, click Okay or next if it does.

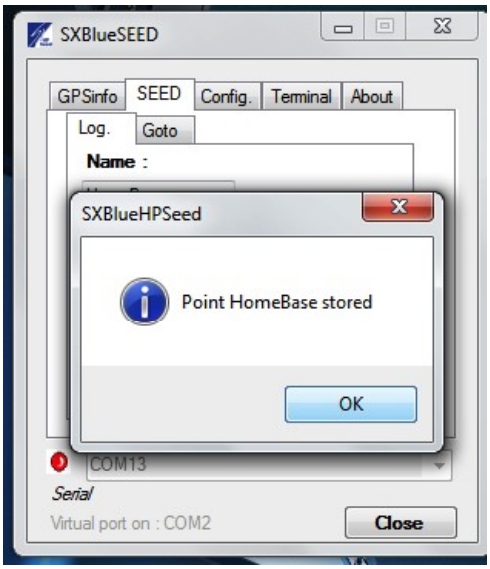


Figure B.34 Dialog box saying the reference (HomeBase) was correctly saved.

5.3.11 The new reference point has been saved. To exit SXBlue HPSeed click close connection and then the X to close the program. SXBlue HPSeed is prone to crash if not closed in this order, the port connection must be closed before closing the program. If not the GPS port will not be available to other programs.

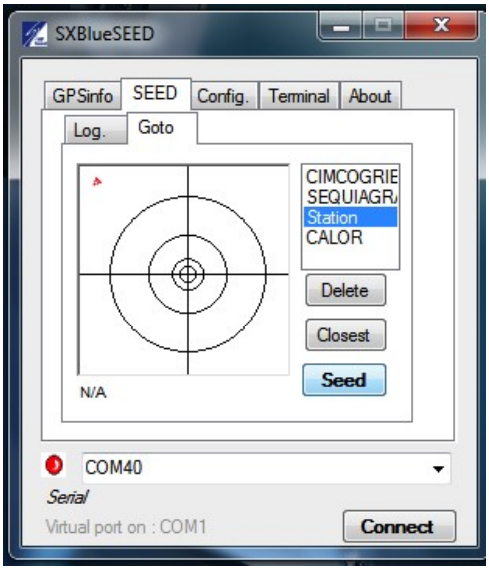


Figure B.35 SXBlue HPSeed after closing the port connection to the GPS, but before exiting the program.

5.3.12 To save the reference point, SXBlue HPSeed turns on various NMEA strings. These must be turned off, or they will interfere with data collection. Open and connect the GPS to SXBlue Config (Section 5.2) and go to the USB(THIS) tab.

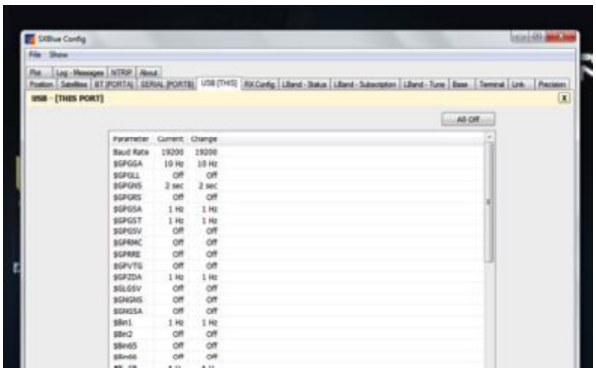


Figure B.36 SXBlue Config USB(THIS) tab showing NMEA strings turned on. Examples are $GPGGA 10 Hz, $GPGNS 2 sec, $GPZDA 1 Hz.

5.3.13 Highlight and select from the drop down menu “off” for all strings other than $GPGGA $GPZDA. ($GPGGA and $GPZDA should be set to 10 Hz).

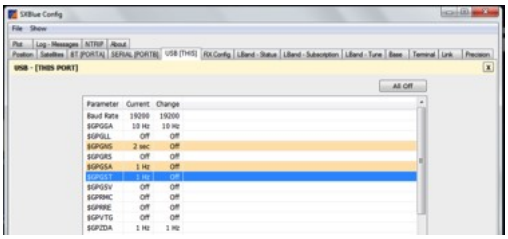


Figure B.37 Turning off NMEA strings in SXBlue Config. $GPGNS has changed from 2 sec to off, $GPGSA and $GPGST are now off.

5.3.14 Disconnect and save settings (Section 5.2.6B). This will save the settings for only the $GPGGA and $GPZDA strings to be exported.

Figure B.38 Saving the new settings when exiting.

5.4 Field GPS Setup

Note: Must have completed GPS setup.

5.4.1 Plug in all USB components as outlined in Initial Phenocorn Startup. TIP: place different colored tape on each USB and computer port.

5.4.2 Turn on the GPS and allow time to find GPS satellites and correction, normally 2-3 minutes. The SXBlue GPS receiver will have four LED lights that illuminate when GPS, DGPS, and DIFF (SBAS-Omnistar correction have been located) Place the GPS receiver at the seeded reference point (Section 5.2). If the GPS receiver is not at the reference point, the program will seed the point but it will not be accurate.

5.4.3 Open the SXBlue Config (Section 5.3) and check the Diff Used in the Position tab and the precision from the Precision tab. Diff Used should be "OMNIHP" or "OMNIG2", if Omnistar is not used wait for the signal to be found. Caution: If Omnistar correction isn't received, and further steps are taken, the receiver will drift and readings will not be accurate.

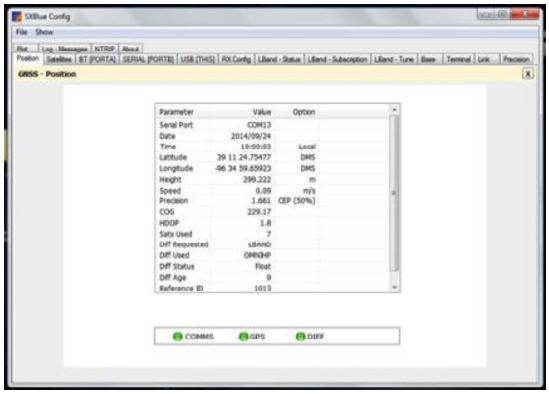


Figure B.39 Using the Position tab in SXBlue Config to verify the type of Diff Used. OMNIHP is this example, and the Diff Status is "Float".

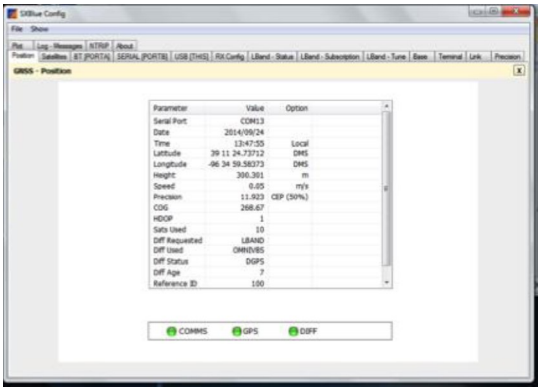


Figure B.40 Startup before OmnistarHP tracking occurs. Dif Status is "DGPS". If following steps are taken before allowing the signal to correct to OMNIHP, GPS readings will drift.

5.4.4 Close the SXBlue Config by choosing “disconnect without save” (Section 5.2.6 A.).

SXBlue Config must be closed to provide access to the GPS port.

5.4.5 Open and connect the GPS to SXBlue HPSeed (Section 5.3.55.3.5).

5.4.6 Click on the Seed tab and then Goto tab.

5.4.7 From the list select reference point of reference (example: HomeBase) and click seed.

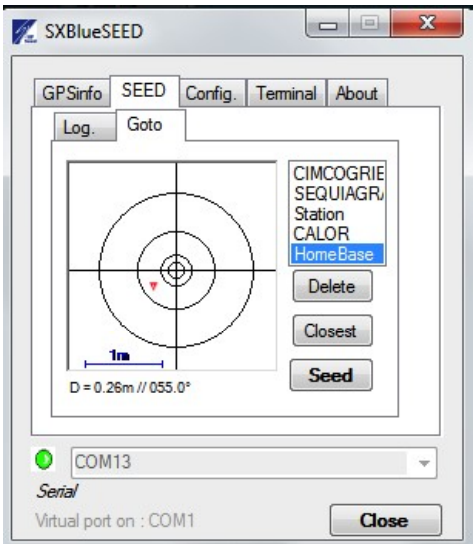


Figure B.41 Selecting the reference point of interest HomeBase.

5.4.8 Click seed with the reference point highlighted. When the triangle moves to the center of the bulls eye, the position has been located with <10cm accuracy.

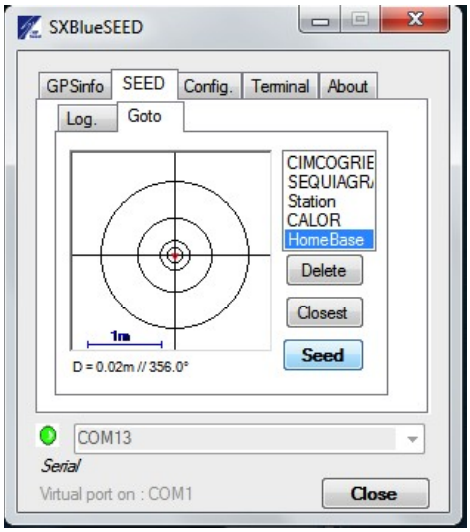


Figure B.42 SXBlue HPSeed after seeding the reference point. Position, red cursor, is directly over the perpendicular x, y lines. D, distance from point is <5cm (0.02m).

5.4.9 The program has seeded location of the antennae to the reference point. Click disconnect and then close SXBlue Seed (Section 5.3.11). This must be closed correctly or the program will crash.

5.4.10 To verify correction position, open SXBlue Config (Section 5.2.2) and check the position Diff Used. Diff Used should be "OMNIHP" or "OMNIG2". The Diff Status should also be "Fixed". The precision should be less than 0.05 m (<5cm). If these criteria are not met repeat the seeding process. Even if precision is high and Diff Used is not locked, the GPS readings will drift resulting in inaccuracies.

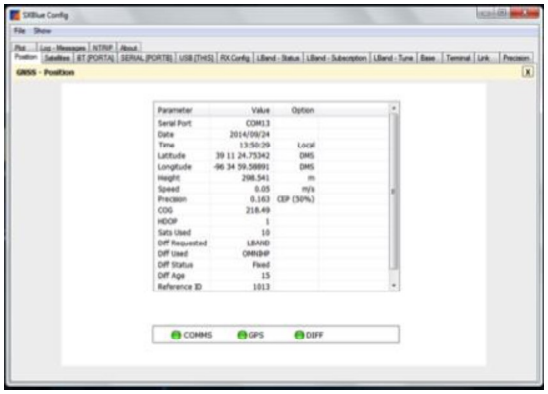


Figure B.43 Verifying that the OMNIHP and Diff Status is Fixed.

5.4.11 Close the SXBlue Config (Section 5.2.6 A.).

6. Phenocorn Field Setup

6.1.1 The GPS should be turned on and have corrected signal (Section 5.4.1-5.4.11).

6.1.2 All hardware components should be connected to the laptop (Section 5.2).

6.1.3 Create a folder in which to save the image data. This could be saved anywhere on the computer. This folder only needs set up once, subsequent phenotyping runs can be saved into the same folder/directory.

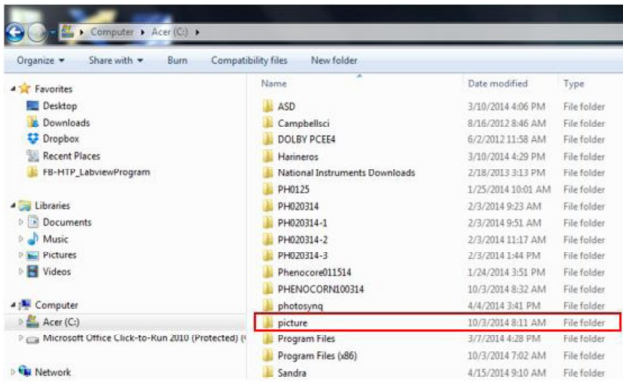


Figure B.44 Creating a folder (example picture) to save image files.

6.1.4 Launch the Phenocorn icon (phenocorn100314) from the Desktop.

6.1.5 When prompted, choose the location and file name that the data will be saved to including the file type extension, example.txt. The text file containing IRT, NDVI, and GPS reference will be saved in this file name. Click OK. If a name is not entered the default GreenSeekerData will be used.

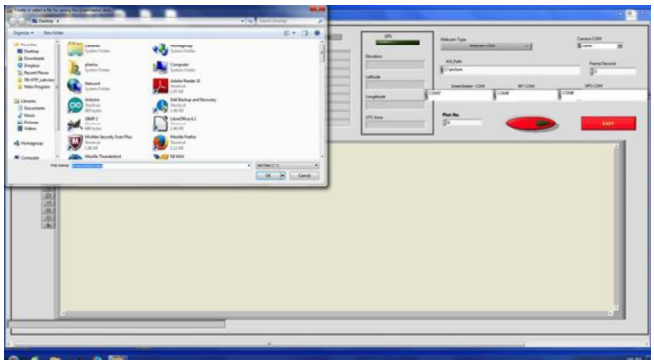


Figure B.45 Launching the Phenocorn program, and choosing the location to save the data file, including extension name.

6.1.6 Within the graphical user interface select the correct COM port settings for each instrument. Use COM ports found in Table B.1. On one particular computer, we found the following settings which are used for examples, our experience shows that COM ports can vary widely between computers.

GreenSeeker	COM15
IRT	COM12
GPS	COM14

When the GPS COM is correct, the GUI will display current GPS location, elevation, and time without recording this data. **TIP: Check the GPS is functioning by observing that the output is changing.**

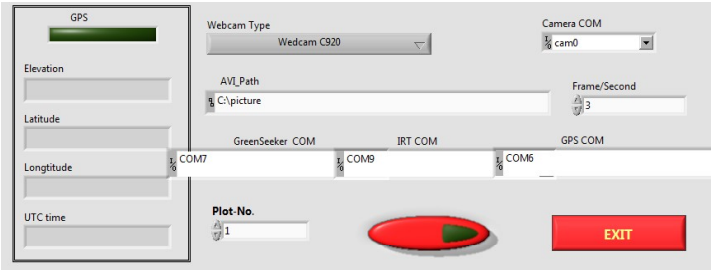


Figure B.46 Selecting the correct COM port setting for each sensor: GreenSeeker, IRT, and GPS. This is initial screen and COM numbers are not set correctly.

6.1.7 Enter the absolute path (ie. C:\Users\phemu\Desktop|PhenocornPhoto) to save the video files. This path will be to the folder made in Section 6.1.3, and this file is separate from the text data file. If no path information is entered the video will not be saved. TIP: Find the folder (using file explorer, etc.) that was created for the photo data and copy and paste the path, this saves typing and increases path accuracy.

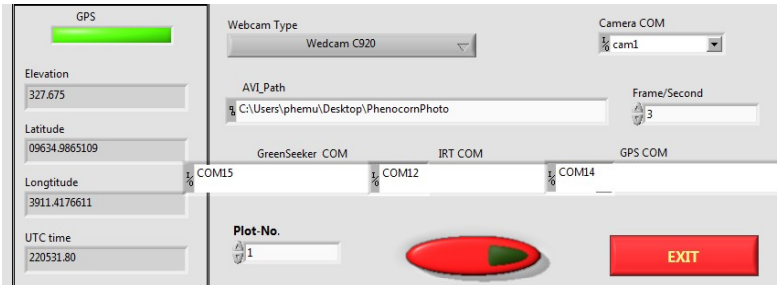


Figure B.47 Entering path to save video.

6.1.8 Click the red oval button. The button will turn green and the program is now ready to collect data. TIP: move the mouse of the button, this prevents accidental clicking and stopping the program from recording data.

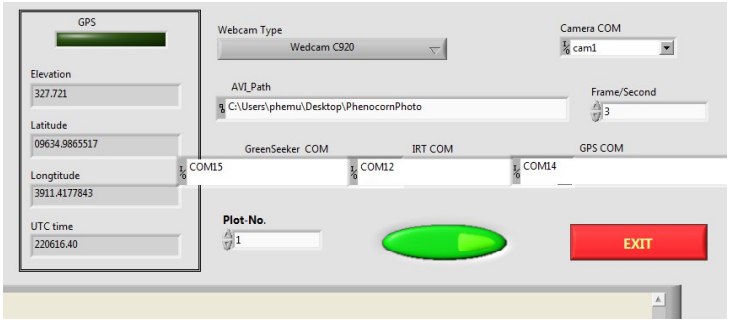


Figure B.48 Phenocorn Software ready to record data. The oval button is green indicating data is ready to be collected.

6.1.9 Press the GreenSeeker button, and data will begin to record.

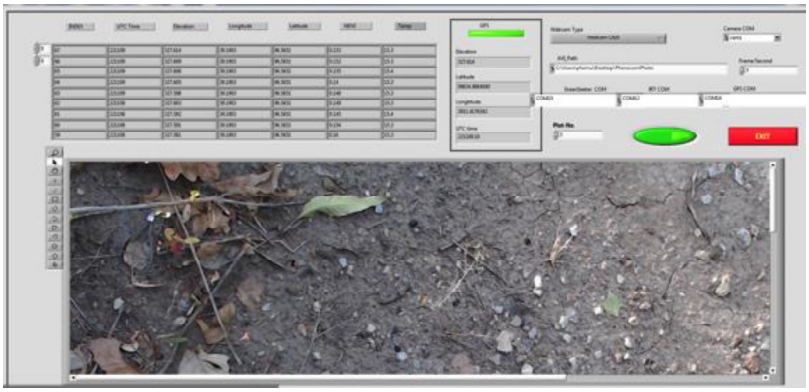


Figure B.49 GUI interface while recording real time data, the picture view also appears in the GUI, in this example the camera was over bare soil and leaves.

During data collection, the computer lid can be shut to save energy. Occasional checks of the data, observing that GPS, NDVI, and CT readings are changing is recommended. **TIP: Checking data before taking readings of the experiment and after finishing is a good check to make sure data is being read and saved into the file properly. If there are problems, post processing will have to be completed to remove bad data. See Troubleshooting section.**

6.1.10 After collecting data, exit the program. The data is saved automatically, and the program is robust to shutting down from the X (close the program), clicking the green button to turn off measurements and then exiting, or clicking the exit button.

6.1.11 Data can be directly analyzed on the computer, or transferred to other computers for analysis.

7. Phenocart

The original Phenocorn design was carried by hand through field plots; however, the Phenocorn is approximately 12 kg with all equipment. The Phenocart is one solution to allow the user to carry less weight in the field. The following images show how a bicycle has been converted into a one-wheel cart that can carry the Phenocorn. This provides a lot of flexibility for any type of planting system and crops.

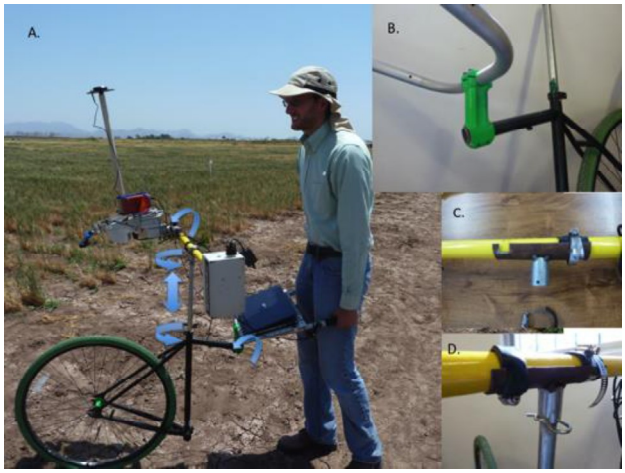


Figure B.50 PhenoCart to carry the Phenocorn in the field. A. Completed PhenoCart with arrows showing adjustable areas of the cart. B. Attaching a handle to the cart. C. Bracket to hold the GreenSeeker sensor using hose clamps. D. Bracket and GreenSeeker attached to the cart.

Figure B.51 Phenocart and Phenocorn in bed planted wheat.

8. Tips for Successful Phenotyping

8.1.1 Charge equipment daily. The Phenocorn operates with three separate batteries, computer, GPS, and GreenSeeker, failure of any one battery will disrupt phenotyping.

8.1.2 Carry adequate tools for adjustment of equipment. Useful equipment include: flathead and Philips screwdriver, wrench or sockets to adjust GreenSeeker and PhenoCart, zip ties, duct tape or electrical tape.

8.1.3 Take time to learn about your particular GPS unit. Failure of GPS, due to lack of convergence, multipath, etc. will result in imprecise data.

8.1.4 Make sure GPS signal is fixed for signal correction.

9. Additional Resources

Trimble: http://www.trimble.com

GreenSeeker User manual:

http://www.nue.okstate.edu/Hand_Held/GS_HandHeld_Manual_rev_K[1].pdf

OmniStar: http://www.omnistar.com

SXBlue: http://www.sxbluegps.com

SXBlue User manuals: http://www.sxbluegps.com/download/

MicroEpsilon CT series manual:

http://www.instrumart.com/products/32312/micro-epsilon-ct-series-infrared-thermometer

10. Troubleshooting

Problem: The computer runs out of battery.

Solutions: Storing the HTP data, especially images, is an energy intensive process. Turn off any unused processes and programs. Disable the wireless adapter, dim the output screen, and changing energy savings options, (when the computer sleeps on low battery) are all options that will provide more time in the field.

Problem: The computer goes to sleep.

Solution: Change the power options, if phenotyping with the computer lid closed, change to an option that does not force sleep when the computer lid is closed.

Problem: Connection to the USB(s) is lost. Usually associated with a noise similar to improperly unplugging a USB memory stick from a computer.

Solutions: Look for a USB connection that is loose. Replacing USB cables also helps, if a connection is repeatedly being lost. Depending on computer position, securing the USB connection so there is no movement can be useful.

Problem: SXBlue HPSeed crashes on closing.

Solutions: SXBlue HPSeed must be closed in the following order 1. Close the connection to the GPS by clicking Disconnect. 2. Close the program by clicking on the X. If it is closed out of order, clicking on the X while still connected to the GPS, the program effectively

crashes and will not allow other programs to access the GPS. If this occurs force quit the SXBlue HPSeed program by using Ctrl-Alt-Del and closing from the task manager.

Problem: The collected data have one (or more columns) that did not change in the file.

Solution: One of the sensor connections was malfunctioning, so the program writes the last, good, (before the connection malfunction) data point until the end of data collection. In the field, before turning off the program run a check by collecting data and watching the screen to see if data is changing. If the data stream is changing then there has not been an issue. After data collection examine the end of the file for irregularities. Depending on the sensor that malfunctioned either the column, containing the bad data, after the malfunction can be deleted (IRT, and NDVI), or all rows and columns after the malfunction need to be deleted (GPS) because there is no known location for these points.

Problem: The data look good, but instead of 10Hz there is irregularity, ie many good readings and then a pause in data collection.

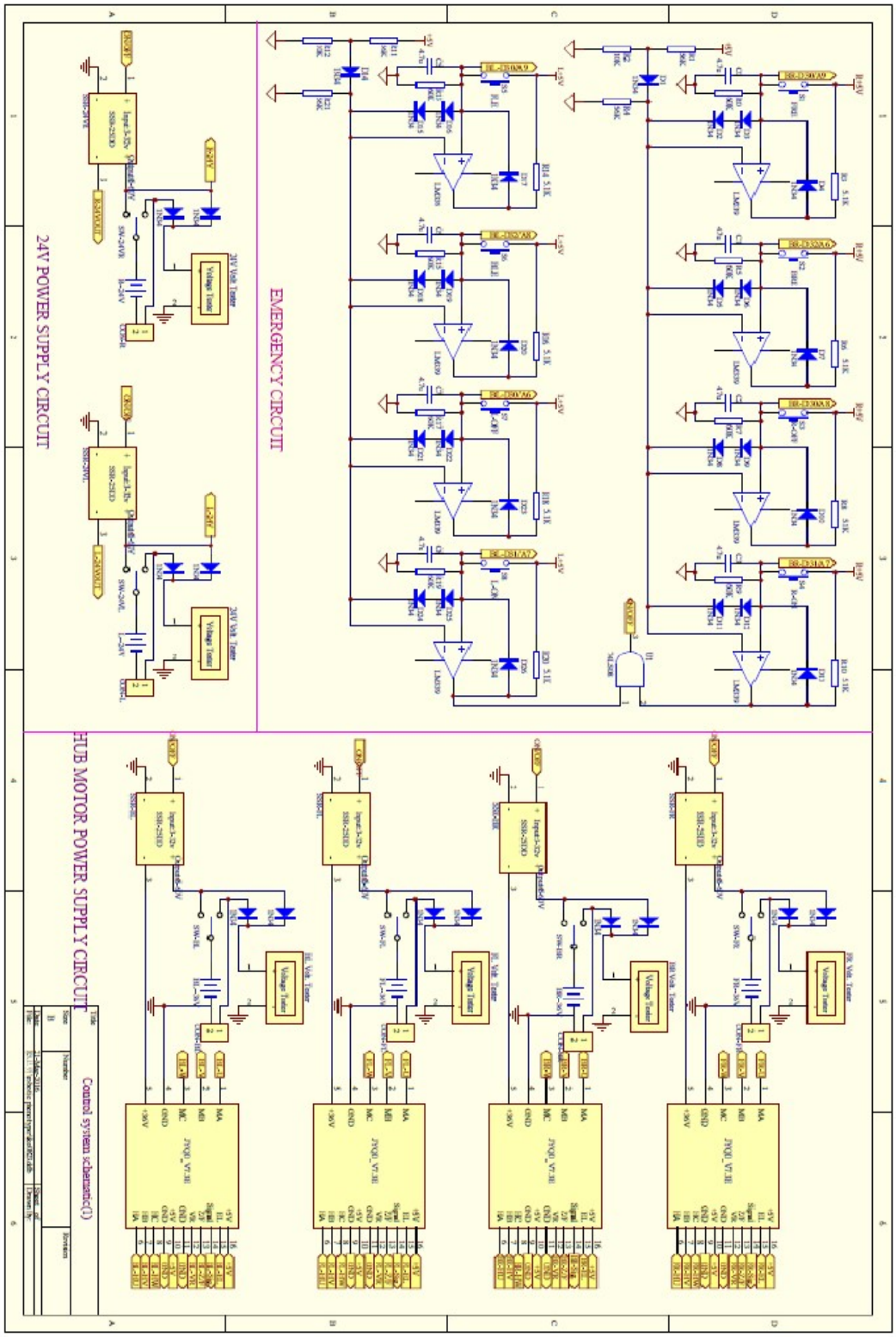
Solution: Different NMEA strings turned on at different rates will cause data reading irregularities. The LabView program will not continue its loop until all data NMEA strings have been exported thus one string exporting at 1Hz will slow the overall data collection. Only the $GPGGA string needs to be turned on at 10 Hz, all other NMEA output can be turned off. Use the SXBlue Config to change any other NMEA strings to off and save the settings. This usually occurs after setting a new reference point (Section 5.3.12)

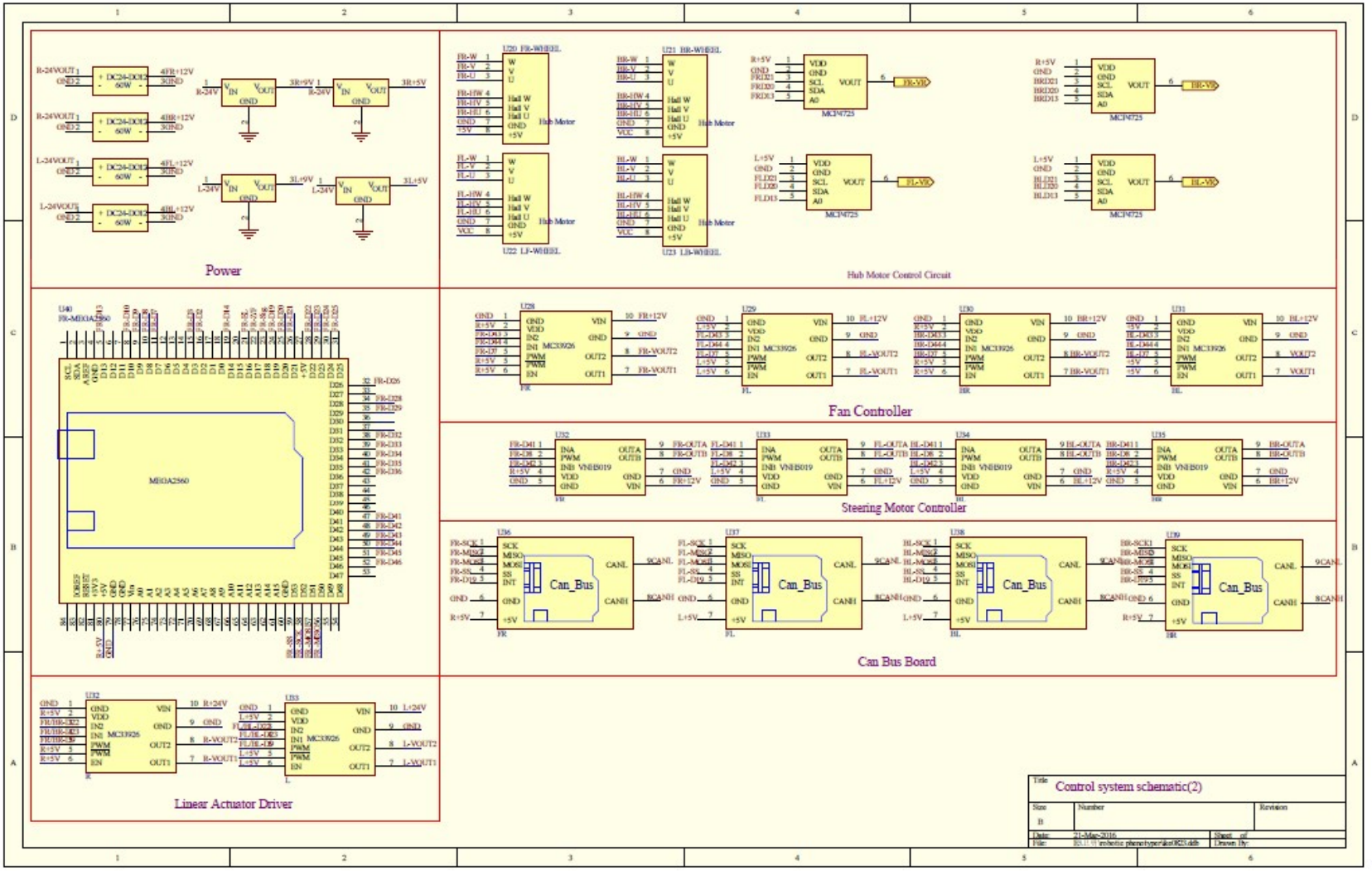
11. Hardware Equipment and Specifications

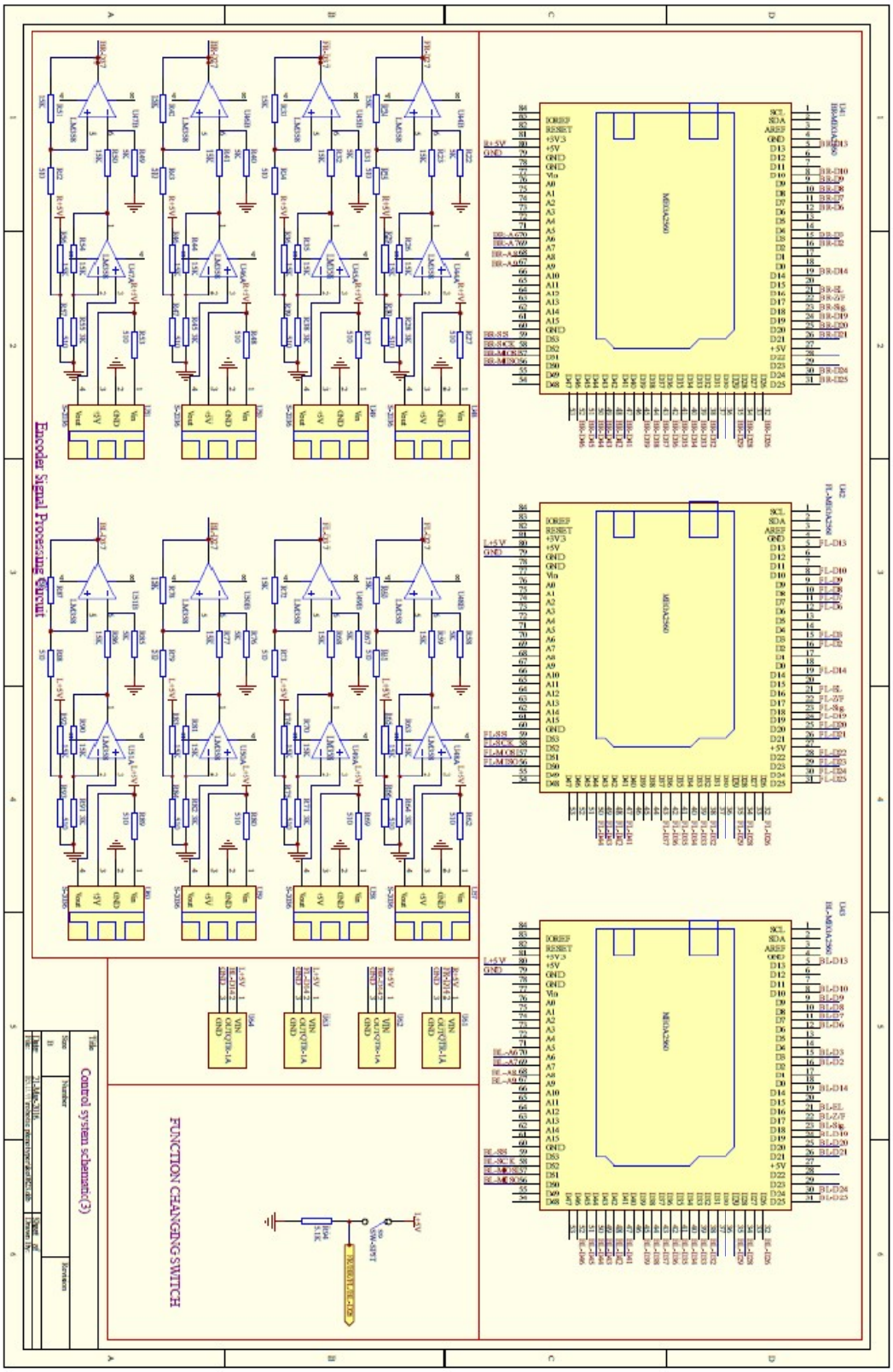
Table B.2 Phenocorn hardware components and technical specifications.

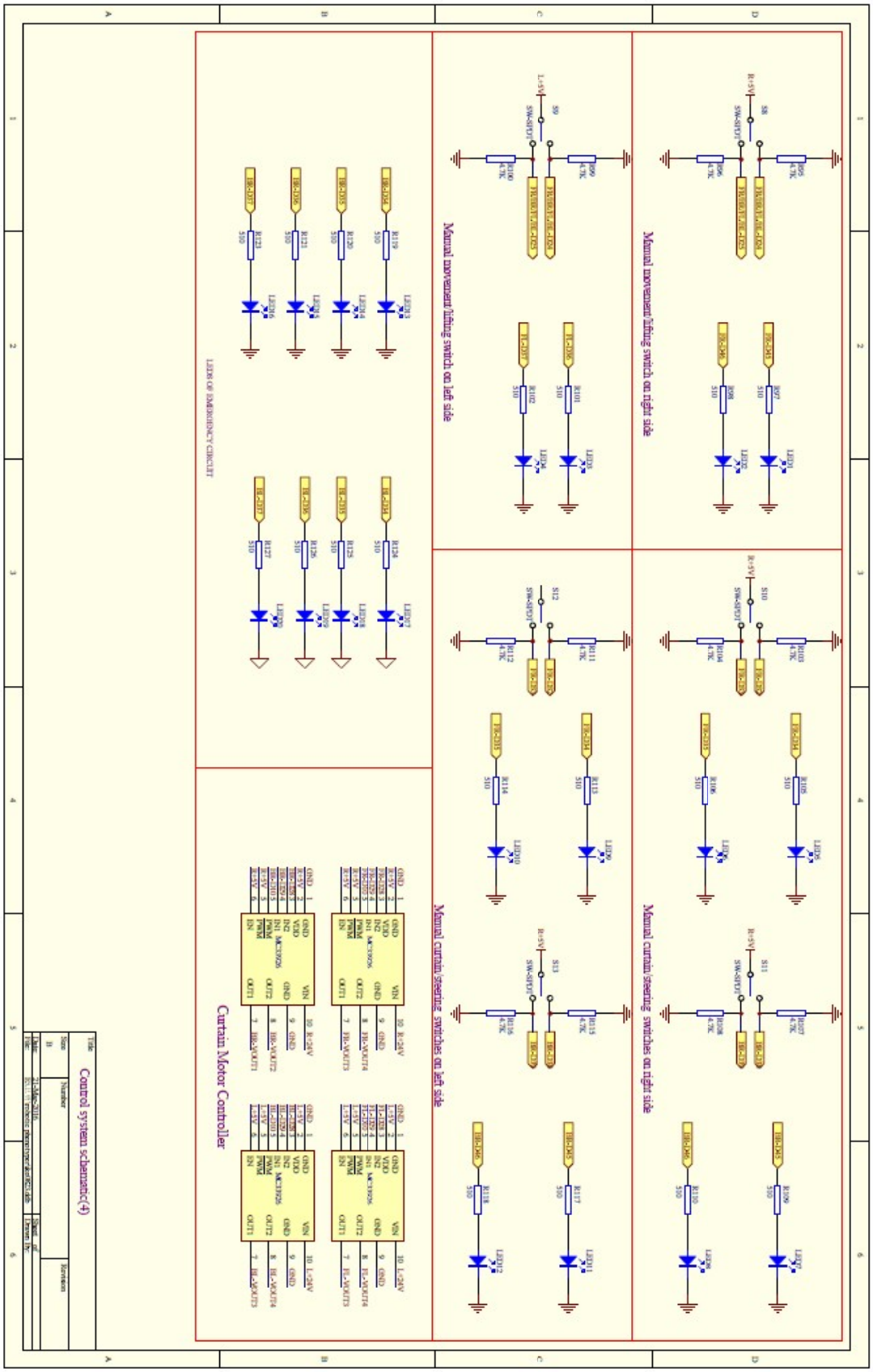
Instrument	Company	Field of View	Output	Output Type	Other Specifications
Aspire 4830	Acer Inc., San Jose, CA				Windows 7, 2.4 Ghz process, 8 GB RAM, 64 bit
GreenSeeker	Trimble, CA	1cmX60cm	10 Hz/ 38400 baud	RS-232	RS-232 to USB converter used
SXBlue III-L	Geneq, Montreal, Canada		10 Hz / 19200 baud	USB	Omnistar HP for 95% accuracy <= 10cm
CT Series Thermometer	MicorEpsilon, Raliegh, NC	1:2	10 Hz/ 9600 baud	analog	Analog to Digital (AD) converter, Temperature resolution 0.1 °C, System accuracy: ±1 °C
C920 Logitech			~3 Hz	USB	HD 1080p

Appendix C-Control system schematic

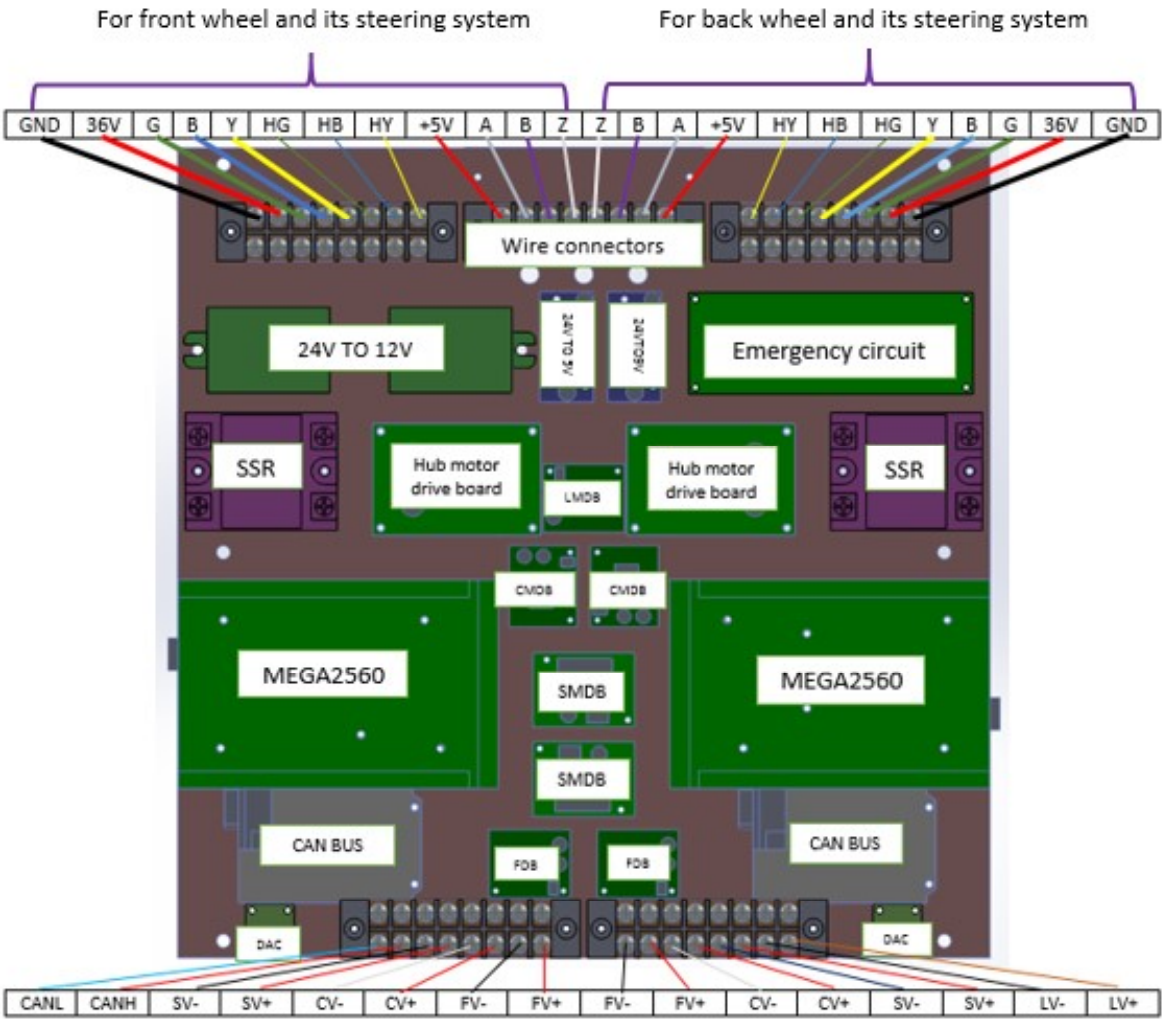



Power
Hub Motor Control Circuit
Fan Controller
Steering Motor Controller
Can Bus Board
Linear Actuator Driver
Control system schematic(2)


Encoder Signal Processing Circuit
FUNCTION CHANGING SWITCH
Control system schematic(3)


Manual movement lifting switch on right side
Manual movement lifting switch on left side
Manual curtain steering switches on right side
Manual curtain steering switches on left side
Curtain Motor Controller
Control system schematic(4)

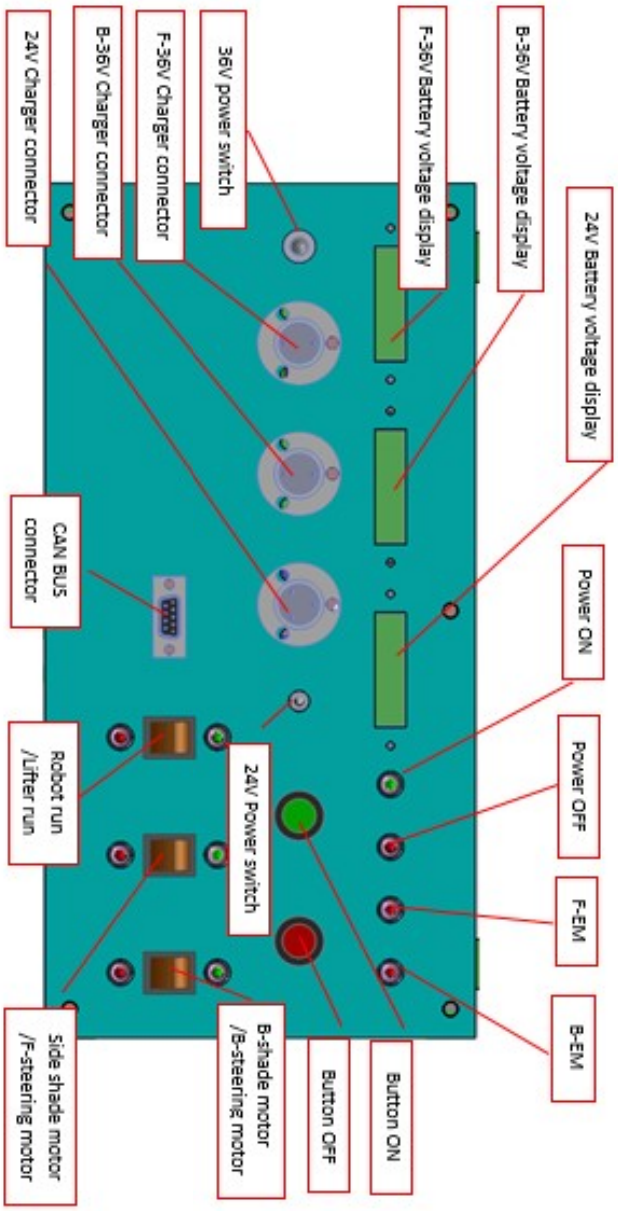
Appendix D-Control system wiring diagram



Terminal block specification

Item	Specification	Item	Specification	Item	Specification
GND	System ground	+5V	5v power supply	CV-	Terminal blocks for shade motor
36V	Wheel power	A	Encoder sensor A	CV+	
G	Thick green wire	B	Encoder sensor B	FV-	Terminal blocks for cooling fan
B	Thick blue wire	Z	Sensor for zero position	FV+	
Y	Thick yellow wire	CANL	CAN LOW	LV-	Terminal blocks for lifter motor
HG	Thin green wire for hall	CANH	CAN HIGH	LV+	
HB	Thin blue wire for hall	SV-	Terminal blocks for steering motor		
HY	Thin yellow wire for hall	SV+			

Appendix E-Control system panel



Appendix F-Control system C code

```
#include <EEPROM.h>
#include <Math.h>
#include "mcp_can.h"
#include <SPI.h>
#include <stdio.h>
#include <QTRSensors.h>
#include <Wire.h>
#include <Adafruit_MCP4725.h>

#define CAN_ID 0X00
#define Minimum_pwm 2000
#define INT8U unsigned char
#define UINT unsigned int
#define NUM_SENSORS   1            // number of sensors used
#define TIMEOUT       3000         // waits for 2500 microseconds for sensor outputs to go low
#define EMITTER_PIN   43           // emitter is controlled by digital pin 43
#define kP    5
#define kI    3
#define kD   3                      // for while constant speed control

Adafruit_MCP4725 dac;             // 12-bit Digital to Analog converter for controlling wheel speed
                                  //
QTRSensorsRC qtrrc((unsigned char[]) {14},
  NUM_SENSORS, TIMEOUT, EMITTER_PIN);
unsigned int sensorValues[NUM_SENSORS];

/* Dinfine interface of HUB-MOTOR */
const int HM_CO_IN       = 17;       //OUTPUT PIN
const int HM_EL           =16;       //OUTPUT PIN
const int HM_SP           = 18;      //INPUT PIN
const int HM_SP_CON    = 6;         //OUTPUT PIN


/*Define interface of Steering Motor */
const int SM_DIRA       =41;         //OUTPUT PIN
const int SM_DIRB       =42;         //OUTPUT PIN
const int SM_SP_CON   = 8;           //OUTPUT PIN

/*Define interface of ENCODER */
const int AXORB         =1;          // A^B
const int ENCO_A        =3;          //OUTPUT PIN
const int ENCO_B        =2;          //OUTPUT PIN
const int ZERO_P        = 14;        //OUTPUT PIN

/*Define interface of Lifter Motor */
const int LM_DIRA          =22;   //OUTPUT PIN
const int LM_DIRB          =23;    //OUTPUT PIN
const int LM_SP_CON      = 9;    //OUTPUT PIN
```

```
/*Define interface of Curtain Motor */
const int CM_DIRA          =28;   //OUTPUT PIN
const int CM_DIRB          =29;    //OUTPUT PIN
const int CM_SP_CON      =10;    //OUTPUT PIN

/*Define interface of Brake Motor */
const int BM_DIRA        =43;   //OUTPUT PIN
const int BM_DIRB        =44;    //OUTPUT PIN
const int BM_SP_CON   =7;     //OUTPUT PIN

/*Define Address A0 about MCP4725 */

const int MCP4725_A0=13;

const int MOTORS_PIN[5][3]={  {HM_CO_IN,HM_EL,HM_SP_CON},
                              {SM_DIRA,SM_DIRB,SM_SP_CON},
                              {CM_DIRA,CM_DIRB,CM_SP_CON},
                              {BM_DIRA,BM_DIRB,BM_SP_CON},
                              {LM_DIRA,LM_DIRB,LM_SP_CON},
                             };
/*Define interface of KEYS */

const int KEYI1          =30;   //INPUT PIN
const int KEYI2          =31;   //INPUT PIN
const int KEYI3          =32;   //INPUT PIN
const int KEYI4          =33;   //INPUT PIN
const int KEYI5          =38;   //INPUT PIN
const int KEYI6          =39;   //INPUT PIN
const int KEYI7          =24;   //INPUT PIN
const int KEYI8          =25;   //INPUT PIN
const int KEYI9          =26;   //INPUT PIN
const int LED1           =34;   //OUTPUT PIN
const int LED2           =35;   //OUTPUT PIN
const int LED3           =36;   //OUTPUT PIN
const int LED4           =37;   //OUTPUT PIN
const int LED5           =45;   //OUTPUT PIN
const int LED6           =46;   //OUTPUT PIN
const int EMK1           =A6;
const int EMK2           =A7;
const int EMK3           =A8;
const int EMK4           =A9;
const int keys[8]={KEYI1,KEYI2,KEYI3,KEYI4,KEYI5,KEYI6,KEYI7,KEYI8};
const int emks[4]={EMK1,EMK2,EMK3,EMK4};
const int LEDS[6]={LED1,LED2,LED3,LED4,LED5,LED6};
const int S_Threshold[4]={200,200,200,150};
const int angle_co[4]={-1,1,-1,1};
INT8U BAddress=0;
INT8U command_t=0;
UINT  HMPWM= 0;
UINT  desired_sp=0;
```

```
boolean desired_dir=0;
INT8U report;
INT8U Lifter_on_off;
int  encoderPos=0;
int  DencoderPos=0;
int  encoderPosFlag=0;
boolean Can_get_data = false;
boolean ON_FLAG = false;
boolean OFF_FLAG = false;
boolean Test_run=false;
boolean Stop_flag=false;
INT8U Test_time=0;
float speed_pi_p=0;
float speed_pi_n=0;
float Error=0;
float Integral=0;
float Derivative=0;
float speed_present=0;
float speed_preview=0;
const INT8U BoardA[6][2]={{'F','R'},{'B','R'},{'F','L'},{'B','L'},
                          {'A','L'},{'S','T'},{'F','A'},{'B','A'}
                         };                                  // Board Addresses

const INT8U Actuator_T[3]={'W','S','?'};                     // W----WHEELS;
                                                             // S----STEERING MOTOR

const INT8U Direction[3][2]={{'F','B'},{'L','R'},{'S','W'}}; // F----Forward running;
                                                             // B----Backward running;
                                                             // L---left turn;
                                                             // R----right turn;


/* define arrays for saving the command parameters;          */
/* after resetting the parameters, the action code is set zero.*/

UINT   STATUS[2][3]={{0,0,0},{0,0,0}};       // Action Code 0-waiting, 1-reset;
                                             // Displacement direction: 0-Corotation,
                                             //                         1-Inversion;
                                             // Speed;
                                             // 0-----wheel     1------steering motor

/* Define arrays for saving the reported data */

float Speed[2]={0,0};                        //Displacement direction: 0-Corotation,
                                             //                         1-Inversion;
                                             //Speed, num/min;
int   angle=0;                               //Turnning angle of steering motor;
                                             //Positive angle-----right turn;
                                             //Negative angle-----left turn;
int d_time[4]={250,750,1250,1750};
boolean reply=0;
```

```
INT8U Flag_Recv = 0;
INT8U len = 0;
INT8U buf[8];
INT8U Rbuf[8];
INT8U j=0;

/* Interrupt programs */
void MCP2515_ISR()                          //Can bus interrupt program,  Interrupt 4
{
    Flag_Recv = 1;
    CAN.readMsgBuf(&len, Rbuf);
}

void counting(void)                         //Encoder interrupt program. Interrupt 1
{
        if (digitalRead(ENCO_B) == LOW)
                {
                        encoderPos--;
                }
        else
                {
                        encoderPos++;
                }
        angle=3*pow(-1,BAddress)*encoderPos*angle_co[BAddress];

}

void EEPROM_INI(void)
{
        int address;
        BAddress=EEPROM.read(0);
        if(BAddress==255)
                        BAddress=0;
        report=EEPROM.read(1);
        if(report==255)
                        report=1;
       Serial.print("BOARD ADDRESS = ");
       Serial.write(BoardA[BAddress][0]);
       Serial.write(BoardA[BAddress][1]);
       Serial.println();
}

void stop_all_motors()
{
       INT8U i;
       for(i=0;i<5;i++)
       {
                       motors_s(i);
        }
}
```

```
void setup()
{
        CAN.begin(CAN_500KBPS);                       // init can bus : baudrate = 500k
        attachInterrupt(4, MCP2515_ISR, FALLING);     // start interrupt
        Serial.begin(115200);
        EEPROM_INI();

        pinMode(SM_DIRA, OUTPUT);                     //ABOUT STEERING MOTOR PINS
        pinMode(SM_DIRB, OUTPUT);
        pinMode(SM_SP_CON, OUTPUT);
        pinMode(LM_DIRA, OUTPUT);                     //ABOUT LIFTER MOTOR PINS
        pinMode(LM_DIRB, OUTPUT);
        pinMode(LM_SP_CON, OUTPUT);
        pinMode(CM_DIRA, OUTPUT);                     //ABOUT CURTAIN MOTOR PINS
        pinMode(CM_DIRB, OUTPUT);
        pinMode(CM_SP_CON, OUTPUT);
        pinMode(BM_DIRA, OUTPUT);                     //ABOUT BRAKE MOTOR PINS
        pinMode(BM_DIRB, OUTPUT);
        pinMode(BM_SP_CON, OUTPUT);
        pinMode(HM_CO_IN, OUTPUT);                    //ABOUT HUBMOTOR PINS
        pinMode(HM_EL, OUTPUT);
        pinMode(HM_SP_CON, OUTPUT);
        pinMode(HM_SP, INPUT);
        pinMode(LED1, OUTPUT);                        //ABOUT LEDS PINS
        pinMode(LED2, OUTPUT);
        pinMode(LED3, OUTPUT);
        pinMode(LED4, OUTPUT);
        pinMode(LED5, OUTPUT);
        pinMode(LED6, OUTPUT);

        pinMode(KEYI1, INPUT);                        //ABOUT KEYS PINS
        pinMode(KEYI2, INPUT);
        pinMode(KEYI3, INPUT);
        pinMode(KEYI4, INPUT);
        pinMode(KEYI5, INPUT);
        pinMode(KEYI6, INPUT);
        pinMode(KEYI7, INPUT);
        pinMode(KEYI8, INPUT);
        pinMode(KEYI9, INPUT);
       //ABOUT ENCODER PINS
        pinMode(AXORB,INPUT);
        pinMode(ENCO_A, INPUT);
        pinMode(ENCO_B, INPUT);
        pinMode(ZERO_P, INPUT);
        pinMode(EMK1, INPUT);
        pinMode(EMK2, INPUT);
        pinMode(EMK3, INPUT);
        pinMode(EMK4, INPUT);
       pinMode(MCP4725_A0, OUTPUT);                   //About MCP4725
       digitalWrite(MCP4725_A0, 0);
       dac.begin(0x62);
```

```
	attachInterrupt(1, counting, RISING);
	Lifter_on_off=0;
	stop_all_motors();
	dac.setVoltage(0, false);
	angle=0;
	encoderPos=0;
	desired_sp=0;
}

void motors(int type,boolean Dir, int Duty)
{
	 if(type==0&digitalRead(KEYI9)==0)
	 {
		digitalWrite(MOTORS_PIN[type][0], Dir);
		digitalWrite(MOTORS_PIN[type][1],1);
		//analogWrite(MOTORS_PIN[type][2], Duty);
		dac.setVoltage(Duty, false);
	 }
	 else
	 {
		digitalWrite(MOTORS_PIN[type][0], Dir);
		digitalWrite(MOTORS_PIN[type][1], !Dir);
		analogWrite(MOTORS_PIN[type][2], Duty);
	 }


}
void motors_s(int type)
{
	 if(type==0)
	 {
		digitalWrite(MOTORS_PIN[type][0], 1);
		digitalWrite(MOTORS_PIN[type][1], 0);
		dac.setVoltage(0, false);
		desired_sp=0;
	 }
	 else
	 {
		digitalWrite(MOTORS_PIN[type][0], 0);
		digitalWrite(MOTORS_PIN[type][1], 0);
		analogWrite(MOTORS_PIN[type][2], 0);
	 }

}

// Detecting speed

float V_num(void)
{
	unsigned long  duration;
```

```
            float speed_min;
            duration = pulseIn(HM_SP, HIGH);
             if(duration!=0)
                        speed_min=60000000/(72*duration);
             else
                         speed_min=0;
             return speed_min;
}

float average_s()
{
          float sp[5],t_sp=0;
          for(char i=0;i<5;i++)
          {
                       sp[i]=V_num();
                       delay(100);
          }
          for(char i=0;i<5;i++)
          {
                       for(char j=i;j<5;j++)
                       {
                                  if (sp[i]<sp[j])
                                  {
                                             t_sp=sp[i];
                                             sp[i]=sp[j];
                                             sp[j]=t_sp;
                                  }
                       }
           }

            t_sp=(sp[1]+sp[2]+sp[3])/3;
            Speed[1]=t_sp;
            Speed[0]=STATUS[0][1];
            return(t_sp);
}

void mistake()
{
            INT8U STOP_B[8]={'S','T','O','P','-','r','o','E'};
            STOP_B[5]=BoardA[BAddress][0];
            STOP_B[6]=BoardA[BAddress][1];

           Serial.print("STOP_B = ");
           for(int j = 0; j<8; j++)
               {
                          Serial.write(STOP_B[j]);
               }
           CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B);

}
```

```
void myPID(INT8U dir)
{
        INT8U i=0;
        speed_present=average_s();
        if(speed_present==0)        speed_preview=0;
        Error=(desired_sp-speed_present)/60;
        Integral=Error;
        Derivative=(speed_preview-speed_present)/180;
        float drive_num=90*(kP*Error+kI*Integral+kD*Derivative);
        HMPWM=int(drive_num);
        i=0;
         while(speed_present<5){
                 if(HMPWM>Minimum_pwm)
                 {
                         mistake();
                         desired_sp=0;
                         motors_s(0);
                         break;
                 }
                 if(i!=0)
                {
                        delay(3000);
                }
                i++;
                motors(0,dir,HMPWM);
                HMPWM= HMPWM+100;
                speed_present=V_num();
        }
       Serial.print("HMPWM= ");Serial.println(HMPWM);
}

void const_speed(INT8U dir)
{
        float speed_min;
        float Delt_speed,RATE;
        speed_min=average_s();
        speed_preview= speed_min;
        if(desired_sp!=0)
        {
                Delt_speed=desired_sp-speed_min;
                RATE=abs(Delt_speed)/1.5;
                if(RATE>1&RATE<10)
                {
                        if(Delt_speed>0)
                                HMPWM=HMPWM+8;
                        else
                                HMPWM=HMPWM-8;
                        HMPWM=constrain(HMPWM,0,4095);
                         motors(0,dir,HMPWM);
                }
                else
```

```
                {
                        if(RATE>10)
                        {
                                if(Delt_speed>0)
                                        HMPWM=HMPWM+100;
                                else
                                        HMPWM=HMPWM-100;
                                HMPWM=constrain(HMPWM,0,4095);
                                 motors(0,dir,HMPWM);
                         }
                }
        }

        delay(100);
        Serial.print("HMPWM= ");Serial.println(HMPWM);
        delay(100);
        Serial.print("SPEED= ");Serial.println(speed_min,1);

}

void print_parameter()
{
          Serial.print("encoderPos= ");Serial.println(encoderPos);
          delay(10);
          Serial.print("Angle= ");Serial.println( angle);
          delay(10);
          qtrrc.read(sensorValues);
          Serial.print("sensorValues[0] = "); Serial.println(sensorValues[0]);
}

void Turn_to_Zero(boolean s_dir)
{
          qtrrc.read(sensorValues);
          Serial.print("sensorValues[0] = "); Serial.println(sensorValues[0]);
          motors(1,s_dir,120);
          while(sensorValues[0]<S_Threshold[BAddress])
           {
                    delay(20);
                    qtrrc.read(sensorValues);
                    Serial.print("sensorValues[0] = "); Serial.println(sensorValues[0]);
            }
          motors_s(1);
          encoderPos=0;
          angle=0;

}

void Step_Turn(boolean s_dir)
{
          int pre_encoderPos;
          int delta_step;
```

```
        pre_encoderPos=encoderPos;
        delta_step=abs(abs(pre_encoderPos)-abs(encoderPos));
        motors(1,s_dir,120);
        while(delta_step<1)
          {
                delta_step=abs(abs(pre_encoderPos)-abs(encoderPos));
                delay(10);
          }
        motors_s(1);
}


void Turn_Angle(boolean s_dir,int s_angle)
{
        int pre_encoderPos;
        int delta_step;
        pre_encoderPos=encoderPos;
        delta_step=abs(abs(pre_encoderPos)-abs(encoderPos)+1)*3;
        motors(1,s_dir,120);
        while(delta_step<s_angle)
          {
                delta_step=abs(abs(pre_encoderPos)-abs(encoderPos)+1)*3;
                delay(10);
          }
        motors_s(1);
}

void set_BAddress()
{
         if(Serial.available()>0)
         {
                Serial.readBytes(buf, 8);
                delay(100);
                Serial.write(buf,8);
                delay(100);
                Serial.println(':');
                switch(buf[0])
                {
                        case 'A':case 'a':
                                BAddress=buf[1]&0x0f;
                                EEPROM.write(0,BAddress);
                                delay(100);
                                BAddress=0;
                                EEPROM_INI();
                        break;
                        case '?':
                                delay(100);
                                BAddress=0;
                                EEPROM_INI();
                                Serial.print("Report ");
                                if(report==0)
```

```
            {
                    Serial.println("OFF !");
            }
             if(report==1)
            {
                    Serial.println("ON !");
            }
break;
case 'R':case 'r':
            report=buf[1]&0x0f;
            EEPROM.write(1,report);
            delay(100);
            report=0;
            EEPROM_INI();
            Serial.print("Report ");
            if(report==0)
            {
                    Serial.println("OFF !");
            }
             if(report==1)
            {
                    Serial.println("ON !");
            }
break;

// Following code is used to test the encoder;

case 'B':case 'b':
             //Clear encoderPos and DencoderPos
             encoderPos=0;
             angle=0;
             print_parameter();
             delay(1000);
             print_parameter();
break;

case 'L':case 'l':
              // trun left to the zero position, 1---left turn;
              //                                 0----right turn
              Turn_to_Zero(1);
              print_parameter();
break;

case 'M':case 'm':
              // trun right to the zero position, 1---left turn;
              //                                  0----right turn
              Turn_to_Zero(0);
              print_parameter();
break;

case 'C':case 'c':
```

```
                                        // Step left turn, 1---left turn;
                                        //                 0----right turn
                                         Step_Turn(1);
                                         print_parameter();
                                break;

                                case 'D':case 'd':
                                        // Step right turn, 1---left turn; 0----right turn
                                         Step_Turn(0);
                                         print_parameter();
                                break;

                                case 'E':case 'e':
                                         Turn_Angle(1,90);
                                         print_parameter();
                                break;

                                case 'F':case 'f':
                                         Turn_Angle(0,90);
                                         print_parameter();
                                break;
                               // End of testing encoder code
                                default: break;
                        }

                }
}

void clear_buf()
{
        INT8U i;
        for(i=0;i<8;i++)
        {
                buf[i]=0;
        }
}

void save_data(INT8U j)
{
        INT8U i;
        for(i=0;i<8;i++)
        {
                if(j==1) buf[i]=Rbuf[i];
                Rbuf[i]=0;
        }
}

INT8U check_board()
{
          INT8U i,j;
          for(i=0;i<6;i++)
```

```
            {
                    if((Rbuf[0]==BoardA[i][0])&(Rbuf[1]==BoardA[i][1]))
                    {
                            j=i+1;
                            break;
                    }
                    else  j=0;
             }
           delay(100);
           Serial.write(BoardA[BAddress][0]);Serial.write(BoardA[BAddress][1]);
           Serial.println(" GETS DATA!");
           Serial.print("data len = ");Serial.println(len);
           Serial.print("j = ");Serial.println(j);
           Serial.print("Can Send Ok\r\n");
           delay(100);
           return j;
}

void set_parameter()
{
            INT8U i, j,k;
            switch(buf[2])
            {
                    case 'W':  case 'w': i=0; break;
                    case 'S':  case 's': i=1; break;
                    case '?': reply=1;goto loop4; break;
                    default: goto loop1; break;
            }
           switch(buf[3])
            {
                    case 'R': STATUS[i][1]=0; break;     // 1-----Forward running or right turn;
                    case 'L': STATUS[i][1]=1; break;     // -1----Backward running or left turn;
                    case 'F': STATUS[i][1]=1; break;     // 1-----Forward running or right turn;
                    case 'B': STATUS[i][1]=0; break;     // -1----Backward running or left turn;
                    default: goto loop2; break;
            }
           switch(buf[4])
            {
                    case 'S':  STATUS[i][0]=1; break;    // 1------Starting
                    case 'W':  STATUS[i][0]=0; break;  // 0-----Waiting
                    default: goto loop3; break;
            }
            STATUS[i][2]=get_s_v();
            loop4:loop3:loop2:loop1:
            clear_buf();
}

void answer()
{
            INT8U STOP_B[8]={'#','L','L','-','-','0','K','!'};
            STOP_B[1]=BoardA[BAddress][0];
```

```
            STOP_B[2]=BoardA[BAddress][1];
           Serial.print("STOP_B = ");
            for( j = 0; j<8; j++)
               {
                        Serial.write(STOP_B[j]);
               }
           CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B);
}

void take_parameter()
{
            INT8U i,j,k;
            i=check_board();
            switch(i)
            {
                        case 0:
                                    save_data(0);
                        break;
                        case 1:case 2:case 3:case 4:
                        if(i==(BAddress+1))
                        {
                                     save_data(1);
                                     set_parameter();
                                     //answer();
                        }
                        break;
                        case 5:
                                    save_data(1);
                                    set_parameter();
                        break;
                        case 6:
                                    stop_all_actuators();
                                    save_data(0);
                         break;
                        default:
                                    save_data(0);
                        break;
            }
}

void stop_all_actuators()
{
           INT8U STOP_B[8]={'#','L','L','S','T','O','P','!'};
           STOP_B[1]=BoardA[BAddress][0];
           STOP_B[2]=BoardA[BAddress][1];
           stop_all_motors();
           delay(d_time[BAddress]);
           CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B);
           Stop_flag=1;
}
```

```
void check_address()
{
          if(Flag_Recv)                 // check if get data
           {
                    Flag_Recv = 0;   // clear flag
                    Serial.println("CAN_BUS GET DATA!");
                    Serial.print("data len = ");Serial.println(len);
                    for(int i = 0; i<len; i++)    // print the data
                    {
                              Serial.write(Rbuf[i]);
                    }
                    Serial.println();
                    take_parameter();
           }

}

int get_s_v()
{
          int SPEED_V=0;
          if((buf[5]&0x0f==0)&(buf[6]&0x0f==0)&(buf[7]&0x0f==0))
          {
                    SPEED_V=0;
          }
          else
          {
                   SPEED_V=int(buf[5]&0x0f)*100+int(buf[6]&0x0f)*10+int(buf[7]&0x0f);
          }
          Serial.print("SPEED_V = ");Serial.println(SPEED_V);
          return(SPEED_V);
}

void RUNNING()
{
           INT8U i;
           for(i=0;i<2;i++)
           {
                     if(STATUS[i][0]==1)
                        {
                                  if(i==0)
                                   {
                                            desired_sp=STATUS[i][2]%100;
                                            if(BAddress<2) desired_dir=!boolean(STATUS[i][1]);
                                            else  desired_dir=boolean(STATUS[i][1]);
                                             if(desired_sp==0)
                                             {
                                                      motors_s(0);
                                                      motors_s(3);
                                             }
                                             else
```

```
                                        {
                                                Test_time=STATUS[i][2]/100;
                                                if(Test_time!=0){Stop_flag=0;}
                                                motors(3,1,255);
                                                delay(1000);
                                                myPID(desired_dir);
                                        }
                                }
                                else
                                {
                                        if(STATUS[i][2]!=0)
                                                Turn_Angle(STATUS[i][1],STATUS[i][2]);
                                        else    Turn_to_Zero(STATUS[i][1]);
                                }
                                STATUS[i][0]=0;
                                STATUS[i][2]=0;
                        }
                }

}

void sent_info(INT8U i,INT8U j)
{
        INT8U STOP_B[8]={'#','T','L','E','T','O','P','!'};
        INT8U STOP_B1[8]={'S','T','O','P','A','L','L','!'};
        if(ON_FLAG&OFF_FLAG)
        {
                ON_FLAG=0;
                OFF_FLAG=0;
                STOP_B[1]=BoardA[BAddress][0];
                STOP_B[2]=BoardA[BAddress][1];
                 switch(j)
                 {
                        case 'O':
                                STOP_B[4]='P'; STOP_B[5]='O'; STOP_B[6]='n';
                                CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B);
                        break;
                        case 'C': case 'c':
                                STOP_B1[4]='P'; STOP_B1[5]='O';
                                STOP_B1[6]='f';STOP_B1[7]='f';
                                CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B1);
                        break;
                        case 'B':
                                STOP_B1[4]='B'; STOP_B1[5]='J';
                                STOP_B1[6]='a';STOP_B1[7]='m';
                                CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B1);
                        break;
                        case 'F':
                                STOP_B1[4]='F'; STOP_B1[5]='J';
                                STOP_B1[6]='a';STOP_B1[7]='m';
                                CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B1);
```

```
                        break;
                        case 'T':
                                STOP_B1[4]='T'; STOP_B1[5]='e';
                                STOP_B1[6]='s';STOP_B1[7]='t';
                                CAN.sendMsgBuf(CAN_ID, 0, 8, STOP_B1);
                                Stop_flag=1;
                        break;
                        default:break;
                }
        }
}

void manul_curtain(INT8U n, INT8U dir,INT8U ACT,INT8U duty)
{
        float sp;
        digitalWrite(LEDS[n],1);motors(ACT,dir,duty);
        while(digitalRead(keys[n]));
        digitalWrite(LEDS[n],0);
        motors_s(ACT);
}

void manul_steering(INT8U n, INT8U dir,INT8U ACT,INT8U duty)
{
        float sp;
        digitalWrite(LEDS[n],1);
        motors(ACT,dir,duty);
        while(digitalRead(keys[n]))
        {
                qtrrc.read(sensorValues);
                Serial.print("sensorValues[0] = "); Serial.println(sensorValues[0]);
                delay(10);
                Serial.print("encoderPos = "); Serial.println(encoderPos);
                delay(10);
                Serial.print("Angle = ");Serial.println(angle);
                delay(10);
         }
        digitalWrite(LEDS[n],0);
        motors_s(ACT);
}

void manul_movement(INT8U n, INT8U dir,INT8U ACT,int duty)
{
        float sp;
        INT8U N0=0;
        speed_present=0;
        speed_preview=0;
        Integral=0;
         if(BAddress==1|BAddress==3)
        {
                N0=4;
        }
```

```
            digitalWrite(LEDS[n],1);
            desired_sp=60;
            motors(3,1,255);
            myPID(dir);
            delay(2000);
            while(digitalRead(keys[n+N0]))
            {
                    const_speed(dir);
                    delay(3000);
            }
            digitalWrite(LEDS[n],0);
            motors_s(ACT);
            motors_s(3);
            delay(3000);
}

INT8U keyscan()
{
        INT8U key_value=0;
        for(INT8U i=0;i<6;i++)
        {
                key_value=(key_value<<1)+digitalRead(keys[i]);
        }
        return key_value;
}

void manually_runA(INT8U key_value)
{
        boolean dir=1;
        if(BAddress==2)         dir=!dir;
          if(digitalRead(KEYI9)==1)
          {
                    switch(key_value)
                    {
                            case 0x20:manul_steering(0, 1,1,150);break;
                            case 0x10:manul_steering(1, 0,1,150);break;
                            case 0x02:manul_curtain(4, 1,4,255);break;
                            case 0x01:manul_curtain(5, 0,4,255);break;
                            default: break;
                    }
          }
          else
          {
                    switch(key_value)
                   {
                            case 0x20:manul_curtain(0, 1,2,255);break;
                            case 0x10:manul_curtain(1, 0,2,255);break;
                            case 0x02: manul_movement(4, dir,0,Minimum_pwm);break;
                            case 0x01: manul_movement(5, !dir,0,Minimum_pwm);break;
                            default: break;
                   }
```

```
            }
          desired_sp=0;
}

void manually_runB(INT8U key_value)
{
            boolean dir=1;
           if(BAddress==3) dir=!dir;
            if(digitalRead(KEYI9)==1)
            {
                        switch(key_value)
                        {
                                    case 0x80:
                                                if(BAddress==3)
                                                {
                                                            digitalWrite(LEDS[1],0);
                                                            digitalWrite(LEDS[0],1);
                                                            stop_all_motors();sent_info(0,'C');
                                                }
                                                else
                                                {
                                                            digitalWrite(LEDS[1],0);
                                                            digitalWrite(LEDS[2],1);
                                                            stop_all_motors();
                                                            sent_info(0,'B');
                                                 }
                                                OFF_FLAG=1;
                                     break;
                                     case 0x40:
                                                digitalWrite(LEDS[1],1);
                                                digitalWrite(LEDS[0],0);
                                                digitalWrite(LEDS[2],0);
                                                digitalWrite(LEDS[3],0);
                                                sent_info(0,'O');
                                                ON_FLAG=1;
                                    break;
                                    case 0x20:
                                                if(BAddress==3)
                                                {
                                                            digitalWrite(LEDS[1],0);
                                                            digitalWrite(LEDS[2],1);
                                                            stop_all_motors();
                                                            sent_info(0,'B');
                                                }
                                                 else
                                                {
                                                            digitalWrite(LEDS[1],0);
                                                            digitalWrite(LEDS[0],1);
                                                            stop_all_motors();
                                                            sent_info(0,'C');
                                                }
```

```
                        OFF_FLAG=1;break;
                    case 0x10:
                        digitalWrite(LEDS[1],0);
                        digitalWrite(LEDS[3],1);
                        stop_all_motors();
                        sent_info(0,'F');
                        OFF_FLAG=1;
                    break;
                    case 0x48:
                         manul_steering(4, 1,1,150);
                    break;
                    case 0x44:
                        manul_steering(5, 0,1,150);
                    break;
                    case 0x42:
                       manul_curtain(2, 1,4,255);
                    break;
                    case 0x41:
                      manul_curtain(3, 0,4,255);
                     break;
                     default: break;
            }
}
else
{
          switch(key_value)
          {
                case 0x80:
                if(BAddress==3)
                {
                        digitalWrite(LEDS[1],0);
                        digitalWrite(LEDS[0],1);
                        stop_all_motors();sent_info(0,'C');
                }
                else
                {
                        digitalWrite(LEDS[1],0);
                        digitalWrite(LEDS[2],1);
                        stop_all_motors();sent_info(0,'B');
                }
                 OFF_FLAG=1;
                 break;
              case 0x40:
                 digitalWrite(LEDS[1],1);
                 digitalWrite(LEDS[0],0);
                 digitalWrite(LEDS[2],0);
                 digitalWrite(LEDS[3],0);
                 sent_info(0,'O');
                 ON_FLAG=1;
             break;
            case 0x20:
```

```
                        if(BAddress==3)
                        {
                                digitalWrite(LEDS[1],0);
                                digitalWrite(LEDS[2],1);
                                stop_all_motors();
                                sent_info(0,'B');
                        }
                        else
                        {
                                digitalWrite(LEDS[1],0);
                                digitalWrite(LEDS[0],1);
                                stop_all_motors();
                                sent_info(0,'C');
                        }
                         OFF_FLAG=1;
                    break;
                    case 0x10:
                        digitalWrite(LEDS[1],0);
                        digitalWrite(LEDS[3],1);
                        stop_all_motors();
                        sent_info(0,'F');
                        OFF_FLAG=1;
                    break;
                    case 0x48:
                           manul_curtain(4, 0,2,255);
                    break;
                    case 0x44:
                            manul_curtain(5, 1,2,255);
                    break;
                    case 0x42:
                            manul_movement(2, dir, 0,Minimum_pwm);
                            desired_sp=0;
                    break;
                    case 0x41:
                            manul_movement(3, !dir,0,Minimum_pwm);
                           desired_sp=0;
                    break;
                    default: break;
                   }
            }

}

void keys_detect()
{
        INT8U i,j;
        INT8U key_value=0;
        if(BAddress==0|BAddress==2)
        {
                key_value=keyscan()&0x33;
                delay(200);
```

```
                key_value=keyscan()&0x33;
                if(key_value!=0)
                {
                        manually_runA(key_value);
                }

        }
        else
        {
                 for(i=0;i<8;i++)
                {
                        if(i<4)
                        {
                                boolean BSL;
                                if(analogRead(emks[i])<850) BSL=0;
                                 else BSL=1;
                                 key_value=(key_value<<1)+BSL;
                        }
                        else
                        {
                               key_value=(key_value<<1)+digitalRead(keys[i]);
                        }

                }

                if(key_value!=0)
                {
                        manually_runB(key_value);
                }

        }
}

void can_send_data()
{
        INT8U DATA_B[8];
        int speed_v;
        if(reply==1){
                DATA_B[0]='#';
                DATA_B[1]=BoardA[BAddress][0];
                DATA_B[2]=BoardA[BAddress][1];
                DATA_B[3]='S';
                if(STATUS[0][1]==1) DATA_B[4]='F';
                else DATA_B[4]='B';
                speed_v=int((Speed[1]+0.5));
                DATA_B[5]=speed_v/100+0x30;
                DATA_B[6]=(speed_v%100)/10+0x30;
                DATA_B[7]=speed_v%10+0x30;
                CAN.sendMsgBuf(CAN_ID, 0, 8, DATA_B);
                delay(100);
                DATA_B[3]='A';
```

```
                if(angle==0)
                {
                        DATA_B[4]=0x30;
                        DATA_B[5]=0x30;
                        DATA_B[6]=0x30;
                        DATA_B[7]=0x30;
                 }
                else
                {
                        speed_v=abs(angle);
                        if(angle<0)  DATA_B[4]='-';
                        else        DATA_B[4]='+';
                        DATA_B[5]=speed_v/100+0x30;
                        DATA_B[6]=(speed_v%100)/10+0x30;
                        DATA_B[7]=speed_v%10+0x30;
                }
                CAN.sendMsgBuf(CAN_ID, 0, 8, DATA_B);
                reply=0;
        }
}

void check_data()
{
          INT8U i,k,j;
          INT8U NUM=0;
          int speed_v;
         if(report){
                        for(i=0;i<2;i++)
                        {
                                Serial.print("Actuator type = ");
                                switch(i)
                                {
                                        case 0:
                                                Serial.println("Wheel");
                                        break;
                                        case 1:
                                                Serial.println("Steering system");
                                        break;
                                 }

                               for(k=0;k<3;k++)
                               {
                                        NUM=STATUS[i][k]/100;
                                        Serial.print(NUM);
                                        NUM=(STATUS[i][k]%100)/10;
                                        Serial.print(NUM);;
                                        NUM=STATUS[i][k]%10;
                                        Serial.print(NUM);
                                        delay(100);Serial.print('\t');
                               }
                               delay(100);Serial.println();
```

```
                        }

        }
        report=0;
}


void delay_t(INT8U j)
{
        INT8U i;
        float s;
        for(i=0;i<j;i++)
        {
                delay(100);
                if(i%10==0)
                {
                        s=average_s();
                }
                if(Flag_Recv)
                {
                        check_address();
                        RUNNING();
                }
                keys_detect();
        }
}

void loop()
{
        char serial_flag=0;
        char i,k;
         if(Flag_Recv)
        {
                check_address();
                RUNNING();
        }
         set_BAddress();
         check_data();
         if(desired_sp!=0)
         {
              if(Test_time==0)
                   {
                        const_speed(desired_dir);
                        delay_t(10);
                   }
                   else
                   {
                        for(i=0;i<Test_time;i++)
                        {
                                const_speed(desired_dir);
```

```
                                        delay_t(10);
                                }
                                desired_sp=0;
                                motors_s(0);
                                motors_s(3);
                                if(Stop_flag==false)
                                {
                                        sent_info(0,'T');
                                }
                        }
                }
                keys_detect();
                can_send_data();
                for(j=0;j<8;j++)
                {
                        buf[j]=0;
                }
}
```

Appendix G-Matlab code of OTSU method

```
% ========================================================================
% Image segmentation with OTSU method
% This code was modified based on some literatures.
% ========================================================================
close all;
clear all;
clc;
warning off;
imagename=input('Enter the file name:','s');
SE = strel('diamond',4);
Image = imread(imagename);
SImage=imresize(Image,0.2);
I_gray=rgb2gray(SImage);
BW=im2bw(I_gray);
figure, imshow(I_gray)
timestring=gettime();
figure,imshow(SImage),title(['Original image (Name:',imagename,')','
',timestring]);
I_double=double(I_gray);
[wid,len]=size(I_gray);
colorlevel=256;
hist=zeros(colorlevel,1);

for i=1:wid
    for j=1:len
        m=I_gray(i,j)+1;
        hist(m)=hist(m)+1;
    end
end
hist=hist/(wid*len);
miuT=0;
for m=1:colorlevel
    miuT=miuT+(m-1)*hist(m);
end
xigmaB2=0;

for mindex=1:colorlevel
      threshold=mindex-1;
      omega1=0;
      omega2=0;
      for m=1:threshold-1
            omega1=omega1+hist(m);
      end
      omega2=1-omega1;
      miu1=0;
      miu2=0;
      for m=1:colorlevel
            if m<threshold
                  miu1=miu1+(m-1)*hist(m);
            else
                  miu2=miu2+(m-1)*hist(m);
            end
      end
```

```
      xigmaB21=omega1*(miu1-miuT)^2+omega2*(miu2-miuT)^2;
      xigma(mindex)=xigmaB21;
      if xigmaB21>xigmaB2
            finalT=threshold;
            xigmaB2=xigmaB21;
      end
end
fT=finalT/255
T=graythresh(I_gray)

for i=1:wid
      for j=1:len
            if I_double(i,j)>finalT
                  bin(i,j)=1;
            else
                  bin(i,j)=0;
            end
      end
end
timestring=gettime();
figure,imshow(bin),title(['Segmented Image (Name:',imagename,')',' with OTSU
method','    ',timestring]);
Cover=1-sum(bin(:))/(wid*len)
[gray_im]=E_objectives(SImage, bin,1,0);
figure,imshow(gray_im),title(['Objectives Image (Name:',imagename,')','with
OTSU method']);
```

Appendix G-Matlab code of K-Means method

```
% ========================================================================
% Image segmentation with K-Means
% This code was modified based on some literatures.
% ========================================================================
close all;
clear all;
clc;
imagename=input('Enter the file name:','s');
C_Segments=2;
Image = imread(imagename);
SImage=imresize(Image,0.2);
I_gray=rgb2gray(SImage);
timestring=gettime();
figure,imshow(SImage),title(['Original Image (Name:',imagename,')','
',timestring]);
img_gray=rgb2gray(SImage);
[m,n]=size(img_gray);
T=graythresh(img_gray);
img_bw=im2bw(img_gray,T);
GraySeg= reshape(img_gray(:, :), m*n, 1);
cGray=kmeans(double(GraySeg), 2);
rGray= reshape(cGray, m, n);

[wid,len]=size(rGray);
bin=rGray-1;
y=floor(m/2);
for i=1:n
    if bin(y,i)==1
        x1=i;
        break;
    end
end
for i=1:n
    if bin(y,i)==0
        x2=i;
        break;
    end
end
timestring=gettime();
figure,imshow(bin),title(['Segmented Image (Name:',imagename,')',' with K-
Means method ','    ',timestring]);
Cover=1-sum(bin(:))/(m*n)
[gray_im]=E_objectives(SImage, bin,1,0);
figure,imshow(gray_im),title(['Objectives Image (Name:',imagename,')',' with
K-Means method']);
```

Appendix H-Matlab code of ME (Maximum Entropy) method

```
% ========================================================================
% Image segmentation with Maximun Entropy
% This code was modified based on some literatures.
% ========================================================================
close all;
clear all;
clc;
warning off;
imagename=input('Enter the file name:','s');
BW1=imread(imagename);
SImage=imresize(BW1,0.2);
a=rgb2gray(SImage);
[wid,len]=size(a);
timestring=gettime();
figure,imshow(SImage),title(['Original Image (Name:',imagename,')','
',timestring]);
count=imhist(a);
[m,n]=size(a);
N=m*n;
L=256;
count=count/N;
for i=1:L
      if count(i)~=0
            st=i-1;
            break;
      end
end
for i=L:-1:1
      if count(i)~=0
            nd=i-1;
            break;
      end
end
f=count(st+1:nd+1);
size(f)
E=[];

for Th=st:nd-1
      av1=0;
      av2=0;
      Pth=sum(count(1:Th+1));

      for i=0:Th
            av1=av1-count(i+1)/Pth*log(count(i+1)/Pth+0.00001);
      end

      for i=Th+1:nd-1
            av2=av2-count(i+1)/(1-Pth)*log(count(i+1)/(1-Pth)+0.00001);
      end
      E(Th-st+1)=av1+av2;
end
position=find(E==(max(E)));
finalT=st+position-1; %finalT is the gray threshold
```

```
for i=1:wid   % Image binarization
      for j=1:len
            if a(i,j)>finalT
                  bin(i,j)=1;
            else
                  bin(i,j)=0;
            end
      end
end
timestring=gettime();
figure,imshow(bin),title(['Segmentation Image (Name:',imagename,')',' with
ME(Maximum Entropy) method','    ',timestring]);
Cover=sum(bin(:))/(wid*len)
[rgb_im]=E_objectives(SImage,bin,1,0);
figure,imshow(rgb_im),title(['Objectives Image (Name:',imagename,')',' with
ME(Maximum Entropy) method']);
```

Appendix I-Matlab code of RGB-based GF (Green Feature) method

```
% ====================================================================
% Image segmentation with GF(Green Feature) method
% This code was modified based on some literatures.
% ====================================================================
function G=ColourA(Image)
fR=double(Image(:,:,1));  % RED
fG=double(Image(:,:,2));  % GREEN
fB=double(Image(:,:,3));  % BLUE
[x,y]=size(Image);
EG=2*fG-fR-fB;
AV=0;
range1=find(EG>=AV);
range2=find(EG<AV);
GEG=EG;
GEG(range1)=1;
GEG(range2)=0;
Dgr=(fG-fR)./(fG+fR);
range1=find(Dgr>0);
range2=find(Dgr<=0);
GGR=Dgr;
GGR(range1)=1;
GGR(range2)=0;
Dgb=(fG-fB)./(fG+fB);
range1=find(Dgb>0);
range2=find(Dgb<=0);
GGB=Dgb;
GGB(range1)=1;
GGB(range2)=0;
G=(GEG.*GGR).*GGB;
```

Appendix J-Matlab code of LAB-based GF (Green Feature) method

```
% ====================================================================
% Image segmentation with LABGF(Green Feature) method
% This code was modified based on some literatures.
% ====================================================================
close all;
clear all;
clc;
warning off;
imagename=input('Enter the file name:','s');
BW1=imread(imagename);
SImage=imresize(BW1,0.2);
timestring=gettime();
figure,imshow(SImage),title(['Original Image (Name:',imagename,')','
',timestring]);
labim=rgb2lab(SImage);
G=labim(:,:,2);
[wid,len]=size(G);
AV=-6;
range1=find(G<=AV);
range2=find(G>AV);
bin=G;
bin(range1)=1;
bin(range2)=0;
timestring=gettime();
figure,imshow(bin),title(['Segmentation Image (Name:',imagename,')',' with
LAB-based GF method','    ',timestring]);
Cover=sum(uint8(bin(:)))/(wid*len)
[rgb_im]=E_objectives(SImage,bin,1,0);
figure,imshow(rgb_im),title(['Objectives Image (Name:',imagename,')',' with
LAB-based GF method']);
```

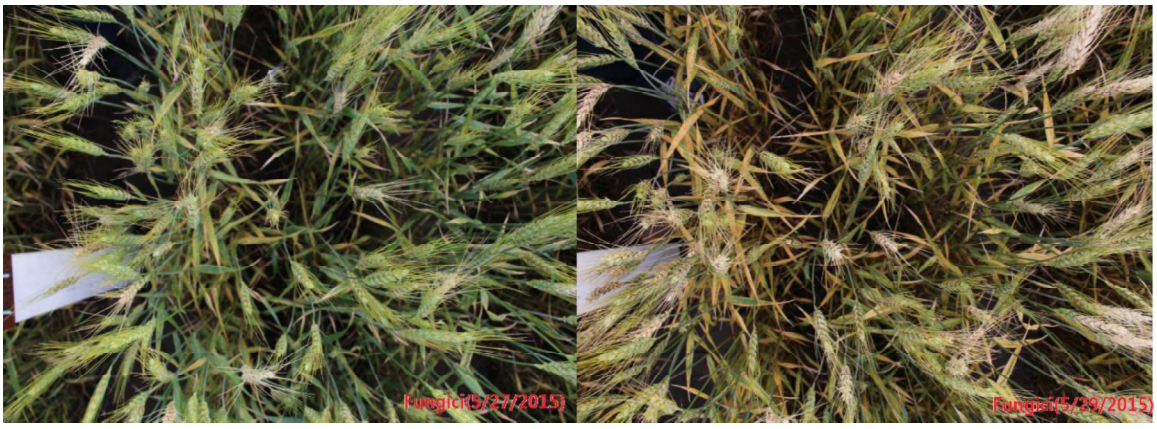
Appendix K-Wheat growth status

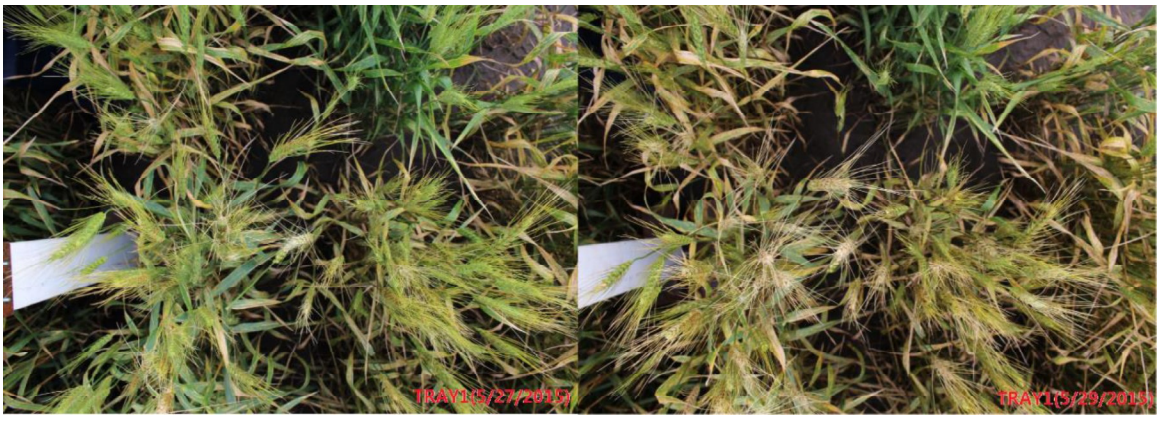
1. Pictures taken on May 16th 2016

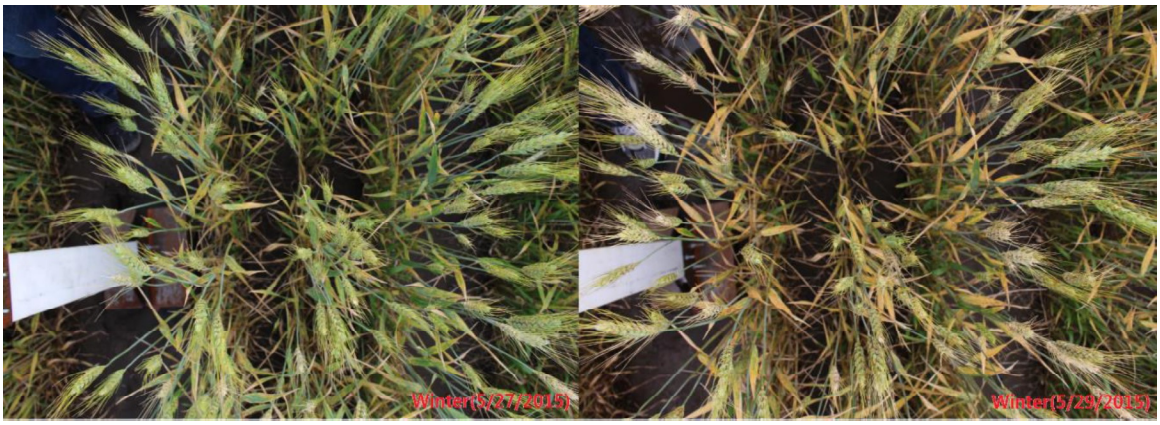

Winter(5/16/2015)


U6837(5/16/2015)

2. Pictures taken on May 27th and 29th 2015






Winter(5/27/2015)
Winter(5/29/2015)


U6837(5/27/2015)
U6837(5/29/2015)


TRAY9(5/27/2015)
TRAY9(5/29/2015)

Appendix L-Selection of Texture Features for wheatear counting

622212515FWH20150522-2SP1X500SZ200X300RGBOriginal

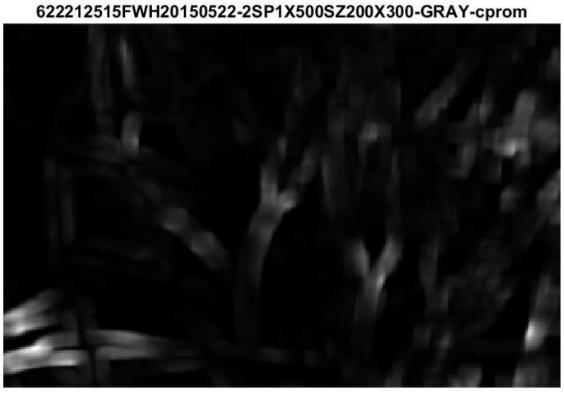
622212515FWH20150522-2SP1X500SZ200X300-GRAY-autoc

622212515FWH20150522-2SP1X500SZ200X300-GRAY-contr

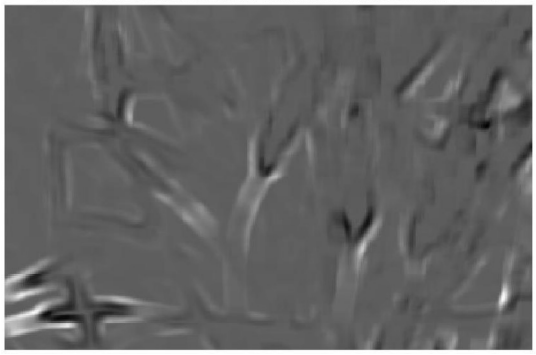
622212515FWH20150522-2SP1X500SZ200X300-GRAY-corrm

622212515FWH20150522-2SP1X500SZ200X300-GRAY-corrp

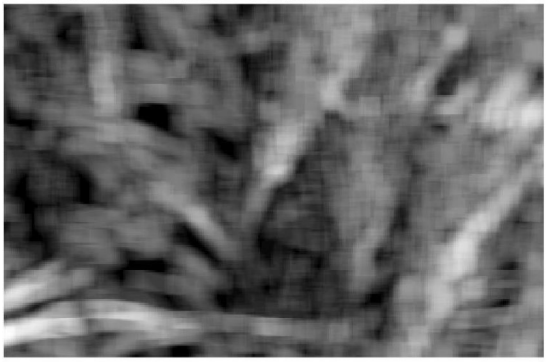
622212515FWH20150522-2SP1X500SZ200X300-GRAY-cprom



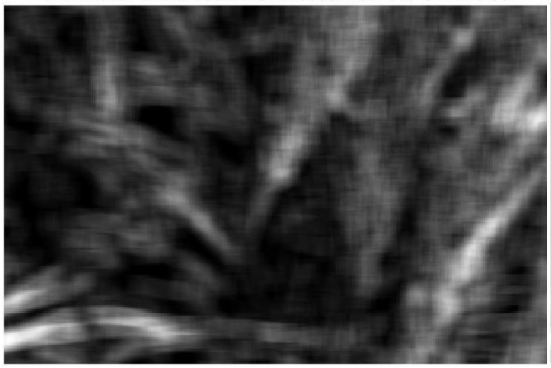
622212515FWH20150522-2SP1X500SZ200X300-GRAY-cshad



622212515FWH20150522-2SP1X500SZ200X300-GRAY-denth

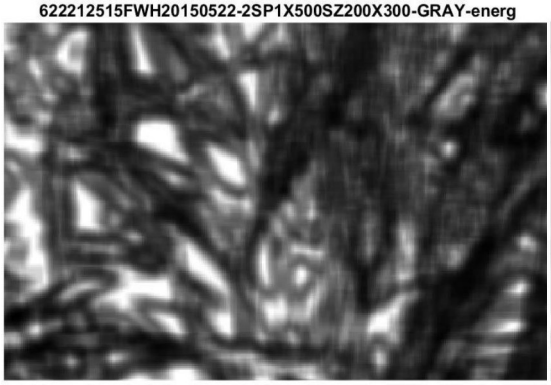


622212515FWH20150522-2SP1X500SZ200X300-GRAY-dissi

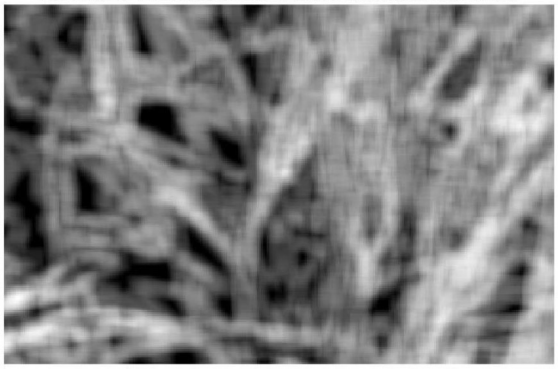


622212515FWH20150522-2SP1X500SZ200X300-GRAY-dvarh

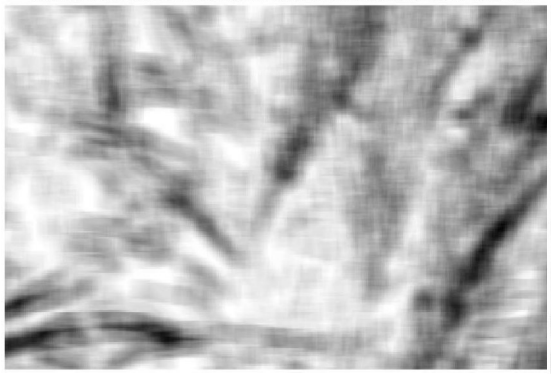
622212515FWH20150522-2SP1X500SZ200X300-GRAY-energ



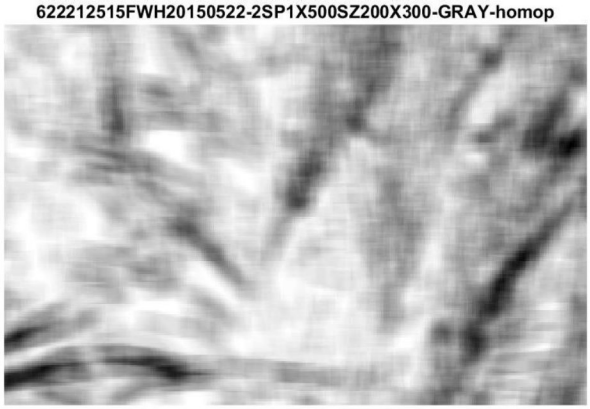
622212515FWH20150522-2SP1X500SZ200X300-GRAY-entro



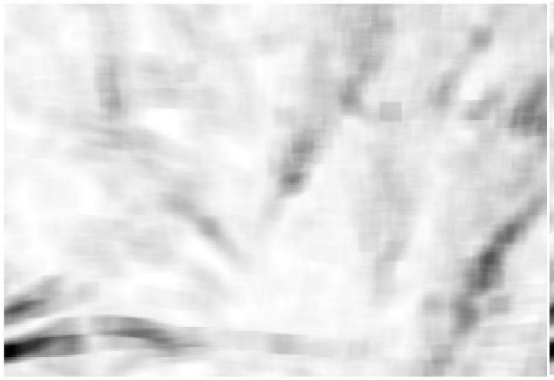
622212515FWH20150522-2SP1X500SZ200X300-GRAY-homom



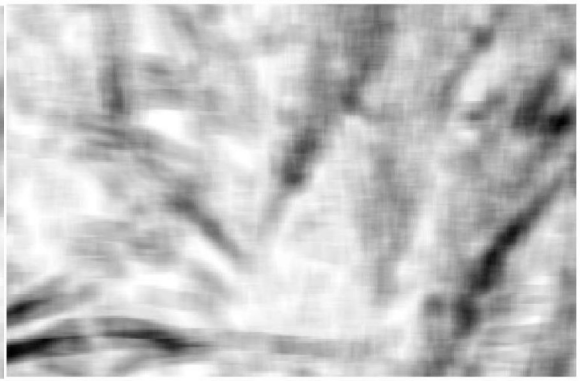
622212515FWH20150522-2SP1X500SZ200X300-GRAY-homop



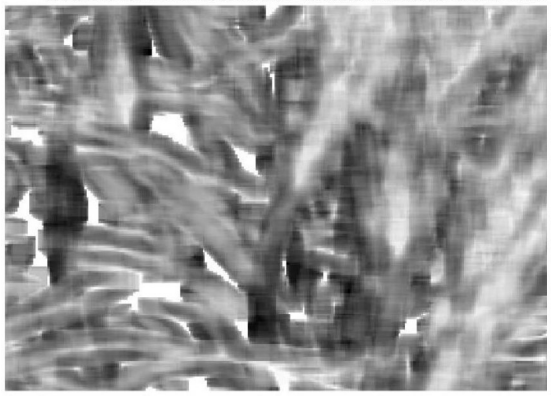
622212515FWH20150522-2SP1X500SZ200X300-GRAY-idmnc



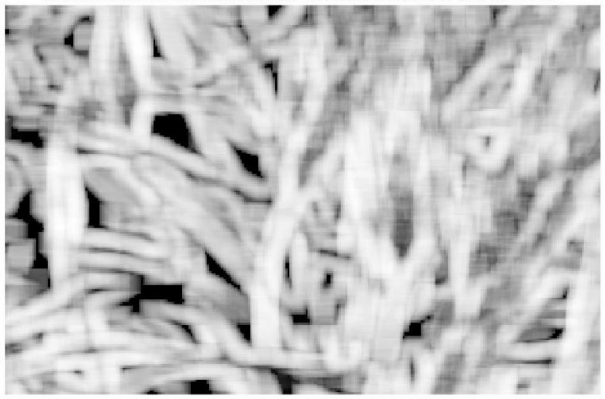
622212515FWH20150522-2SP1X500SZ200X300-GRAY-indnc



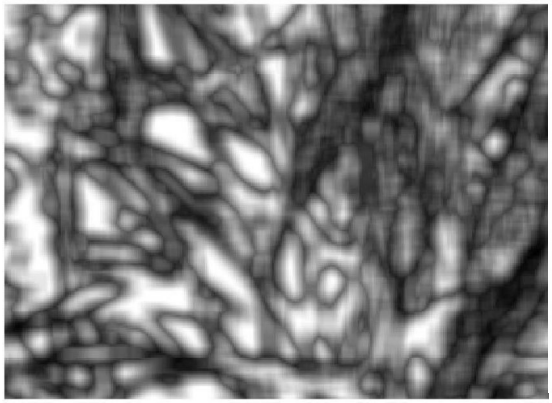
622212515FWH20150522-2SP1X500SZ200X300-GRAY-inf1h



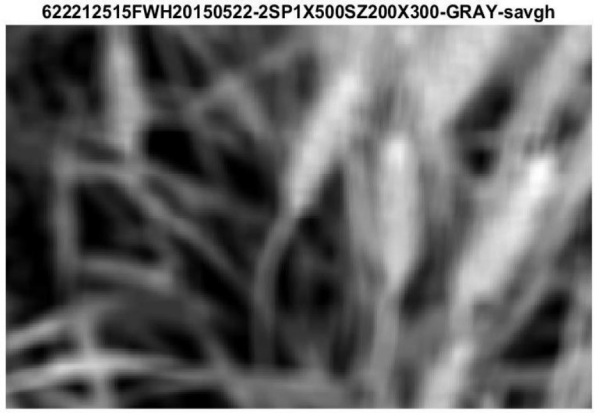
622212515FWH20150522-2SP1X500SZ200X300-GRAY-inf2h



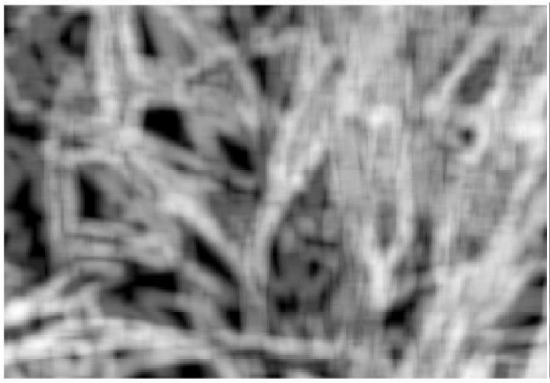
622212515FWH20150522-2SP1X500SZ200X300-GRAY-max



622212515FWH20150522-2SP1X500SZ200X300-GRAY-savgh

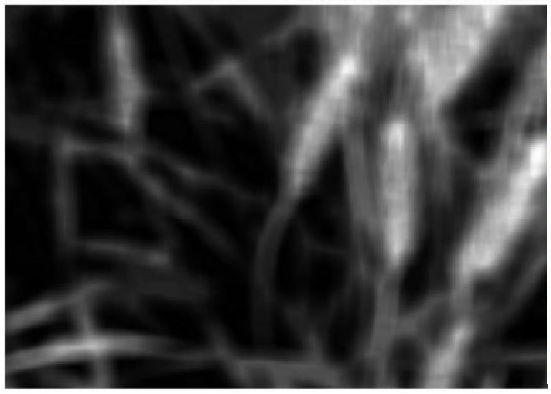


622212515FWH20150522-2SP1X500SZ200X300-GRAY-senth



622212515FWH20150522-2SP1X500SZ200X300-GRAY-sosvh

622212515FWH20150522-2SP1X500SZ200X300-GRAY-svarl



622212515FWH20150522-2SP1X500SZ200X300RGBOriginal

Appendix M-Matlab Code for estimating PVC

```
% ========================================================================%
% For estimating PVC(Percent Vegetation Coverage)                         %
% Author: Yong Wei                                                        %
% ========================================================================%
close all;
clear all;
clc;
warning off;

jpgPath=input('Enter the file folder name:','s');
FileList=dir(jpgPath);
ff=1;
num=1;
for rr=1:length(FileList)

      if(FileList(rr).isdir==1&&~strcmp(FileList(rr).name,'.')&&~strcmp(FileL
      ist(rr).name,'..'))
              fileFolder{ff}=[FileList(rr).name];
              ff=ff+1;
      end
end
folder_num=size(fileFolder);
for j=6:folder_num(2)
    Fpath=char(strcat(jpgPath,'\', fileFolder(j),'\*.JPG'));
    filename=dir( Fpath);
    file_length=length(filename);
    Fname=char(strcat(jpgPath,'\', fileFolder(j)));
    FileN=['c:\whdata0628\',char( fileFolder(j)),'Data'];

    a=['mkdir ' FileN];
    system(a)
    for i=1:file_length
        imagename=[Fname,'\',filename(i).name];
        idx=strfind(imagename,'2015');
        Time=imagename(idx:idx+7);
        Time_n=str2num(Time)-20150000;
        Image = imread(imagename);
        [Gimage,GIMF,Cover]=LABRGBGF(Image,-6);
        figure(1);
        imname=['BW','-',filename(i).name];
        imshow(GIMF);title(imname);
        path=[FileN,'\',imname,'.jpeg'];
        print(1,'-djpeg',path);
        figure(2);
        imname=['G','-',filename(i).name];
        imshow(Gimage);title(imname);
        path=[FileN,'\',imname,'.jpeg'];
        print(2,'-djpeg',path);
        coverage1(i,1)=Time_n;
        coverage1(i,2)=Cover*100;
    end
      figure(3),plot(coverage1(:,1),coverage1(:,2),'+');
      imname=[char( fileFolder(j)),' Wheat Growth Curve'];
```

```
      title(imname,'FontName','Times New
      Roman','FontWeight','Bold','FontSize',16)
      xlabel('Time(MMDD)','FontName','Times New Roman','FontSize',14)
      ylabel('Green Coverage(%)','FontName','Times New
      Roman','FontSize',14,'Rotation',90)
      set(gca,'FontName','Times New Roman','FontSize',14)
      path=[FileN,'\',imname,'.jpeg'];
      print(3,'-djpeg',path);
      imname=[char( fileFolder(j)),' Green Coverage'];
      path=[FileN,'\',imname,'.xls'];
      xlswrite(path,coverage1);
end
% ========================================================================%
function [rgb_im,bin,Cover]=LABRGBGF(BW1,AV)

      SImage=imresize(BW1,0.2);
      labim=rgb2lab(SImage);
      G=labim(:,:,2);
      [wid,len]=size(G);
      range1=find(G<=AV);
      range2=find(G>AV);
      bin1=G;
      bin1(range1)=1;
      bin1(range2)=0;
      bin2=ColourA(SImage);
      bin=bin1.*bin2;
      Cover=sum(uint8(bin(:)))/(wid*len);
      [rgb_im]=E_objectives(SImage,bin,1,0);
% ========================================================================%
function [Image]=E_objectives(SImage, bin,C_bit,gray_c)
[y,x]=size(bin);
if C_bit==1
    re_bin=bin;
else
    re_bin=~bin;
end

for i=1:3
    for j=1:y
        for k=1:x
            if re_bin(j,k)==0
              SImage(j,k,i)=255;
            end
        end
    end
end
if gray_c==1
      Image=rgb2gray(SImage);
else
      Image=SImage;
end
```

Appendix N-Matlab code for counting wheatheads

```
% ========================================================================%
% For counting  wheatheads                                                %
% Author: Yong Wei                                                        %
% ========================================================================%
close all;
clear all;
clc;
Param=['autoc';'contr';'corrm';'corrp';'cprom';'cshad';'dissi';'energ';
    'entro';'homom';'homop';'maxpr';'sosvh';'savgh';'svarh';'senth';'dvarh';
    'denth';'inf1h';'inf2h';'homom';'indnc';'idmnc'];

FolderPath=input('Creat NEW FOLDER PATH TO STORE THE PROCESSING
RESULTS:','s');
check=dir(FolderPath);
FLength=length(check);
if FLength~=0
    fprintf('The folder has Existed!');
    Decision=input('Do you use it[yes/no]:','s');
    switch( Decision)
         case 'YES'
             fprintf('Now you are using the Existed folder!');
         case  'yes'
             fprintf('Now you are using the Existed folder!');
         case 'NO'
             FolderPath=input('Creat another NEW FOLDER PATH TO STORE THE
             PROCESSING RESULTS:','s');
         case  'no'
             FolderPath=input('Creat another NEW FOLDER PATH TO STORE THE
             PROCESSING RESULTS:','s');
    end
end
mkdir(FolderPath);
jpgPath=input('Enter the PROCESSED IMAGE PATH:','s'); %Input the storage path
of the processed picture
Image = imresize(imread(jpgPath), 0.25);
figure,imshow(Image)
[X,Y]=size(Image(:,:,1));
fprintf('Picture Size: %s\n',[ num2str(X),'X',num2str(Y)]);
flag=1;
while flag==1
      Image_Size=[input('Enter the selected size(Format:BPX-YSZX-
      Y):','s'),'E'];
      FORNUMSXY=['P',TakePN1(Image_Size,'P','S'),'S'];
      SX_NUM=TakePN1(FORNUMSXY,'P','-');
      SY_NUM=TakePN1(FORNUMSXY,'-','S');
      FORNUMSIZE=['Z',TakePN1(Image_Size,'Z','E'),'E'];
      XSIZE_NUM=TakePN1(FORNUMSIZE,'Z','-');
      YSIZE_NUM=TakePN1(FORNUMSIZE,'-','E');
      sx=str2num(SX_NUM);
      sy=str2num(SY_NUM);
      xsize=str2num(XSIZE_NUM);
      ysize=str2num(YSIZE_NUM);
      imgcut=snipimg(sx,sy,xsize,ysize,Image,'RGB');
      figure(1),imshow(imgcut);
```

```
      Decision=input('Do you use it[yes/no]:','s');
          switch( Decision)
               case  'yes'
                   flag=0;
               case  'no'
                   flag=1;
               case  'YES'
                   flag=0;
               case  'NO'
                   flag=1;
          end
end
imgcutYW=snipimg(sx,sy,xsize-11,ysize-11,Image,'RGB');
[P_name]=TakePN(jpgPath,'\','.');
imname=[P_name,'SP',num2str(sx),'X',num2str(sy),'SZ',num2str(xsize),'X',num2s
tr(ysize)];
simname=[imname,'RGB','Original'];
figure(1),imshow(imgcutYW);title(simname);
path=[FolderPath,'\',simname,'.jpeg'];
print(1,'-djpeg',path);
weather=input('Enter weather condition(sunny/cloudy):','s');
TextureTraits= TextureA(imgcut,11,11,1,0);


Texturepath=[FolderPath,'\TextureTraits(23)'];
mkdir(Texturepath);
for i=1:23
    simname=[imname,'-GRAY-',Param(i,:)];
    figure(1), imshow(TextureTraits(:,:,i));title(simname);
    path=[Texturepath,'\',simname,'.jpeg'];
    print(1,'-djpeg',path);
    bwDGIM=im2bw(TextureTraits(:,:,i));
    simname=[imname,'-BW-',Param(i,:)];
    figure(1),imshow( bwDGIM);title(simname);
    path=[Texturepath,'\',simname,'.jpeg'];
    print(1,'-djpeg',path);
end
switch(weather)
         case  'sunny'
             TH=35;
             b=im2bw(TextureTraits(:,:,1)+TextureTraits(:,:,15));
             h = strel('disk',5);
             D=imerode(b,h);
             h = fspecial('disk',5);
             D = imfilter(D,h);
         case  'SUNNY'
             TH=35;
             b=im2bw(TextureTraits(:,:,1)+TextureTraits(:,:,15));
             h = strel('disk',5);
             D=imerode(b,h);
             h = fspecial('disk',5);
             D = imfilter(D,h);
         case  'cloudy'
             TH=25;
             A=im2bw(TextureTraits(:,:,1));
             B=im2bw(TextureTraits(:,:,13));
```

```
            C=im2bw(TextureTraits(:,:,15));
            b=A|B|C;
             h = strel('disk',3);
             b=imdilate(b,h);
             h = strel('disk',3);
             D=imerode(b,h);
        case  'CLOUDY'
            TH=25;
            A=im2bw(TextureTraits(:,:,1));
            B=im2bw(TextureTraits(:,:,13));
            C=im2bw(TextureTraits(:,:,15));
            b=A|B|C;
            h = strel('disk',3);
             b=imdilate(b,h);
             h = strel('disk',3);
             D=imerode(b,h);
end
simname=[imname,'-BW-','Synthe'];
D= imfill( D,'holes');
figure(1),imshow(D);title(simname);
path=[FolderPath,'\',simname,'.jpeg'];
print(1,'-djpeg',path);
WB=uint8(D);
WB=255-255*WB;
R=imgcutYW(:,:,1)+WB;
G=imgcutYW(:,:,2)+WB;
B=imgcutYW(:,:,3)+WB;
Gimage=cat(3,R,G,B);
simname=[imname,'-RGB-',' wheatheads'];
figure(1),imshow(Gimage);title(simname);
path=[FolderPath,'\',simname,'.jpeg'];
print(1,'-djpeg',path);

Blockpath=[FolderPath,'\WheatearBlock'];
mkdir(Blockpath);
[L,num]=bwlabel(D,8);
counter=0;
for i=1:num
    [r,c] = find(L == i);
    RDRange=max(r)-min(r)+1;
    CDRange=max(c)-min(c)+1;
    a=zeros(RDRange,CDRange);
    rowmin=min(r);
    colmin=min(c);
    numsize=size(r);
    if numsize(1)>81
          counter=counter+1;
          for j=1:numsize
                a(r(j)-rowmin+1,c(j)-colmin+1)=D(r(j),c(j));
          end
          imname=['N',num2str(2*counter-
          1),'EP',num2str(rowmin),'X',num2str(colmin),'SZ',num2str(RDRange),'X
          ',num2str(CDRange),'BW'];
          figure(1),imshow(a);
          title(imname);
          path=[Blockpath,'\',imname,'.jpeg'];
          print(1,'-djpeg',path);
```

```
            b=zeros(RDRange,CDRange,3);
            for k=1:RDRange
                    for j=1:CDRange
                          b(k,j,1)=Gimage(k+rowmin-1,j+colmin-1,1);
                          b(k,j,2)=Gimage(k+rowmin-1,j+colmin-1,2);
                          b(k,j,3)=Gimage(k+rowmin-1,j+colmin-1,3);
                    end
            end

             fR=uint8(b(:,:,1));  % RED
             fG=uint8(b(:,:,2));  % GREEN
             fB=uint8(b(:,:,3));  % BLUE
             WB=uint8(a);
             R=fR.*WB;
             G=fG.*WB;
             B=fB.*WB;
            % Form the RGB image using the CAT operator.
            imgrec=cat(3,R,G,B);%

              imname=['N',num2str(2*counter),'EP',num2str(rowmin),'X',num2str(c
              olmin),'SZ',num2str(RDRange),'X',num2str(CDRange),'RGB'];
              figure(1),imshow(imgrec);
              title(imname);
              path=[Blockpath,'\',imname,'.jpeg'];
              print(1,'-djpeg',path);
    end
end
Features=Extract_ears0622(Blockpath);
Morphology=Skeleton0701(Blockpath,Features,TH);
Image_labeled=LabelNum(Morphology(:,:,1),imgcut);
Earnum=sum(Morphology(:,8,1));
[P_name]=TakePN(jpgPath,'\','.');
imname=[P_name,'SP',num2str(sx),'X',num2str(sy),'SZ',num2str(xsize),'X',num2s
tr(ysize)];
simname=[imname,'  wheatheads Num(',num2str(Earnum),') with JPC method'];
figure(1),imshow(Image_labeled);title(simname);
path=[FolderPath,'\',simname,'.jpeg'];
print(1,'-djpeg',path);
Image_labeled=LabelNum(Morphology(:,:,2),imgcut);
Earnum=sum(Morphology(:,8,2));
simname=[imname,'  wheatheads Num(',num2str(Earnum),') with WSI method'];
figure(1),imshow(Image_labeled);title(simname);
path=[FolderPath,'\',simname,'.jpeg'];
print(1,'-djpeg',path);
Image_labeled=LabelNum(Morphology(:,:,3),imgcut);
Earnum=sum(Morphology(:,8,3));
simname=[imname,'  wheatheads Num(',num2str(Earnum),') with WCM method'];
figure(1),imshow(Image_labeled);title(simname);
path=[FolderPath,'\',simname,'.jpeg'];
print(1,'-djpeg',path);
fprintf('DONE!\n');
```

```
function [out] = GLCM_Features4(glcmin,pairs)
%
%
% This is an update of GLCM_Features2 (vectorized) without ismember()
%
% GLCM_Features2 helps to calculate the features from the different GLCMs
% that are input to the function. The GLCMs are stored in a i x j x n
% matrix, where n is the number of GLCMs calculated usually due to the
% different orientation and displacements used in the algorithm. Usually
% the values i and j are equal to 'NumLevels' parameter of the GLCM
% computing function graycomatrix(). Note that matlab quantization values
% belong to the set {1,..., NumLevels} and not from {0,...,(NumLevels-1)}
% as provided in some references
% http://www.mathworks.com/access/helpdesk/help/toolbox/images/graycomatrix
% .html
%
% This vectorized version of GLCM_FEatures1.m reduces the 19 'for' loops
% used in the headlier code to 5 'for' loops
% http://blogs.mathworks.com/loren/2006/07/12/what-are-you-really-measuring
% /
% Using tic toc and cputime as in above discussion
%
% Although there is a function graycoprops() in Matlab Image Processing
% Toolbox that computes four parameters Contrast, Correlation, Energy,
% and Homogeneity. The paper by Haralick suggests a few more parameters
% that are also computed here. The code is not fully vectorized and hence
% is not an efficient implementation but it is easy to add new features
% based on the GLCM using this code. Takes care of 3 dimensional glcms
% (multiple glcms in a single 3D array)
%
% If you find that the values obtained are different from what you expect
% or if you think there is a different formula that needs to be used
% from the ones used in this code please let me know.
% A few questions which I have are listed in the link
% http://www.mathworks.com/matlabcentral/newsreader/view_thread/239608
%
%
%
% Features computed
% Autocorrelation: [2]                        (out.autoc)
% Contrast: matlab/[1,2]                      (out.contr)
% Correlation: matlab                         (out.corrm)
% Correlation: [1,2]                          (out.corrp)
% Cluster Prominence: [2]                     (out.cprom)
% Cluster Shade: [2]                          (out.cshad)
% Dissimilarity: [2]                          (out.dissi)
% Energy: matlab / [1,2]                      (out.energ)
% Entropy: [2]                                (out.entro)
% Homogeneity: matlab                         (out.homom)
% Homogeneity: [2]                            (out.homop)
% Maximum probability: [2]                    (out.maxpr)
% Sum of squares: Variance [1]                (out.sosvh)
% Sum average [1]                             (out.savgh)
% Sum variance [1]                            (out.svarh)
% Sum entropy [1]                             (out.senth)
% Difference variance [1]                     (out.dvarh)
% Difference entropy [1]                      (out.denth)
```

```
% Information measure of correlation1 [1]   (out.inf1h)
% Informaiton measure of correlation2 [1]   (out.inf2h)
% Inverse difference (INV) is homom [3]     (out.homom)
% Inverse difference normalized (INN) [3]   (out.indnc)
% Inverse difference moment normalized [3]  (out.idmnc)
%
% The maximal correlation coefficient was not calculated due to
% computational instability
% http://murphylab.web.cmu.edu/publications/boland/boland_node26.html
%
% Formulae from MATLAB site (some look different from
% the paper by Haralick but are equivalent and give same results)
% Example formulae:
% Contrast = sum_i(sum_j(  (i-j)^2 * p(i,j) ) ) (same in matlab/paper)
% Correlation = sum_i( sum_j( (i - u_i)(j - u_j)p(i,j)/(s_i.s_j) ) ) (m)
% Correlation = sum_i( sum_j( ((ij)p(i,j) - u_x.u_y) / (s_x.s_y) ) ) (p[2])
% Energy = sum_i( sum_j( p(i,j)^2 ) )           (same in matlab/paper)
% Homogeneity = sum_i( sum_j( p(i,j) / (1 + |i-j|) ) ) (as in matlab)
% Homogeneity = sum_i( sum_j( p(i,j) / (1 + (i-j)^2) ) ) (as in paper)
%
% Where:
% u_i = u_x = sum_i( sum_j( i.p(i,j) ) ) (in paper [2])
% u_j = u_y = sum_i( sum_j( j.p(i,j) ) ) (in paper [2])
% s_i = s_x = sum_i( sum_j( (i - u_x)^2.p(i,j) ) ) (in paper [2])
% s_j = s_y = sum_i( sum_j( (j - u_y)^2.p(i,j) ) ) (in paper [2])
%
%
% Normalize the glcm:
% Compute the sum of all the values in each glcm in the array and divide
% each element by it sum
%
% Haralick uses 'Symmetric' = true in computing the glcm
% There is no Symmetric flag in the Matlab version I use hence
% I add the diagonally opposite pairs to obtain the Haralick glcm
% Here it is assumed that the diagonally opposite orientations are paired
% one after the other in the matrix
% If the above assumption is true with respect to the input glcm then
% setting the flag 'pairs' to 1 will compute the final glcms that would
result
% by setting 'Symmetric' to true. If your glcm is computed using the
% Matlab version with 'Symmetric' flag you can set the flag 'pairs' to 0
%
% References:
% 1. R. M. Haralick, K. Shanmugam, and I. Dinstein, Textural Features of
% Image Classification, IEEE Transactions on Systems, Man and Cybernetics,
% vol. SMC-3, no. 6, Nov. 1973
% 2. L. Soh and C. Tsatsoulis, Texture Analysis of SAR Sea Ice Imagery
% Using Gray Level Co-Occurrence Matrices, IEEE Transactions on Geoscience
% and Remote Sensing, vol. 37, no. 2, March 1999.
% 3. D A. Clausi, An analysis of co-occurrence texture statistics as a
% function of grey level quantization, Can. J. Remote Sensing, vol. 28, no.
% 1, pp. 45-62, 2002
% 4. http://murphylab.web.cmu.edu/publications/boland/boland_node26.html
%
%
% Example:
%
```

```
% Usage is similar to graycoprops() but needs extra parameter 'pairs' apart
% from the GLCM as input
% I = imread('circuit.tif');
% GLCM2 = graycomatrix(I,'Offset',[2 0;0 2]);
% stats = GLCM_features4(GLCM2,0)
% The output is a structure containing all the parameters for the different
% GLCMs
%
% [Avinash Uppuluri: avinash_uv@yahoo.com: Last modified: 04/05/2010]

% If 'pairs' not entered: set pairs to 0
if ((nargin > 2) || (nargin == 0))
   error('Too many or too few input arguments. Enter GLCM and pairs.');
elseif ( (nargin == 2) )
    if ((size(glcmin,1) <= 1) || (size(glcmin,2) <= 1))
       error('The GLCM should be a 2-D or 3-D matrix.');
    elseif ( size(glcmin,1) ~= size(glcmin,2) )
        error('Each GLCM should be square with NumLevels rows and NumLevels
cols');
    end
elseif (nargin == 1) % only GLCM is entered
    pairs = 0; % default is numbers and input 1 for percentage
    if ((size(glcmin,1) <= 1) || (size(glcmin,2) <= 1))
       error('The GLCM should be a 2-D or 3-D matrix.');
    elseif ( size(glcmin,1) ~= size(glcmin,2) )
       error('Each GLCM should be square with NumLevels rows and NumLevels
cols');
    end
end


format long e
if (pairs == 1)
    newn = 1;
    for nglcm = 1:2:size(glcmin,3)
        glcm(:,:,newn)  = glcmin(:,:,nglcm) + glcmin(:,:,nglcm+1);
        newn = newn + 1;
    end
elseif (pairs == 0)
    glcm = glcmin;
end

size_glcm_1 = size(glcm,1);
size_glcm_2 = size(glcm,2);
size_glcm_3 = size(glcm,3);

% checked
out.autoc = zeros(1,size_glcm_3); % Autocorrelation: [2]
out.contr = zeros(1,size_glcm_3); % Contrast: matlab/[1,2]
out.corrm = zeros(1,size_glcm_3); % Correlation: matlab
out.corrp = zeros(1,size_glcm_3); % Correlation: [1,2]
out.cprom = zeros(1,size_glcm_3); % Cluster Prominence: [2]
out.cshad = zeros(1,size_glcm_3); % Cluster Shade: [2]
out.dissi = zeros(1,size_glcm_3); % Dissimilarity: [2]
out.energ = zeros(1,size_glcm_3); % Energy: matlab / [1,2]
out.entro = zeros(1,size_glcm_3); % Entropy: [2]
```

```
out.homom = zeros(1,size_glcm_3); % Homogeneity: matlab
out.homop = zeros(1,size_glcm_3); % Homogeneity: [2]
out.maxpr = zeros(1,size_glcm_3); % Maximum probability: [2]

out.sosvh = zeros(1,size_glcm_3); % Sum of sqaures: Variance [1]
out.savgh = zeros(1,size_glcm_3); % Sum average [1]
out.svarh = zeros(1,size_glcm_3); % Sum variance [1]
out.senth = zeros(1,size_glcm_3); % Sum entropy [1]
out.dvarh = zeros(1,size_glcm_3); % Difference variance [4]
%out.dvarh2 = zeros(1,size_glcm_3); % Difference variance [1]
out.denth = zeros(1,size_glcm_3); % Difference entropy [1]
out.inf1h = zeros(1,size_glcm_3); % Information measure of correlation1 [1]
out.inf2h = zeros(1,size_glcm_3); % Informaiton measure of correlation2 [1]
%out.mxcch = zeros(1,size_glcm_3);% maximal correlation coefficient [1]
%out.invdc = zeros(1,size_glcm_3);% Inverse difference (INV) is homom [3]
out.indnc = zeros(1,size_glcm_3); % Inverse difference normalized (INN) [3]
out.idmnc = zeros(1,size_glcm_3); % Inverse difference moment normalized [3]

glcm_sum  = zeros(size_glcm_3,1);
glcm_mean = zeros(size_glcm_3,1);
glcm_var  = zeros(size_glcm_3,1);

% http://www.fp.ucalgary.ca/mhallbey/glcm_mean.htm confuses the range of
% i and j used in calculating the means and standard deviations.
% As of now I am not sure if the range of i and j should be [1:Ng] or
% [0:Ng-1]. I am working on obtaining the values of mean and std that get
% the values of correlation that are provided by matlab.
u_x = zeros(size_glcm_3,1);
u_y = zeros(size_glcm_3,1);
s_x = zeros(size_glcm_3,1);
s_y = zeros(size_glcm_3,1);

% checked p_x p_y p_xplusy p_xminusy
p_x = zeros(size_glcm_1,size_glcm_3); % Ng x #glcms[1]
p_y = zeros(size_glcm_2,size_glcm_3); % Ng x #glcms[1]
p_xplusy = zeros((size_glcm_1*2 - 1),size_glcm_3); %[1]
p_xminusy = zeros((size_glcm_1),size_glcm_3); %[1]
% checked hxy hxy1 hxy2 hx hy
hxy  = zeros(size_glcm_3,1);
hxy1 = zeros(size_glcm_3,1);
hx   = zeros(size_glcm_3,1);
hy   = zeros(size_glcm_3,1);
hxy2 = zeros(size_glcm_3,1);

corm = zeros(size_glcm_3,1);
corp = zeros(size_glcm_3,1);

for k = 1:size_glcm_3

    glcm_sum(k) = sum(sum(glcm(:,:,k)));
    glcm(:,:,k) = glcm(:,:,k)./glcm_sum(k); % Normalize each glcm
    glcm_mean(k) = mean2(glcm(:,:,k)); % compute mean after norm
    glcm_var(k)  = (std2(glcm(:,:,k)))^2;

    for i = 1:size_glcm_1
```

```
        for j = 1:size_glcm_2
            p_x(i,k) = p_x(i,k) + glcm(i,j,k);
            p_y(i,k) = p_y(i,k) + glcm(j,i,k); % taking i for j and j for i
            %if (ismember((i + j),[2:2*size_glcm_1]))
                p_xplusy((i+j)-1,k) = p_xplusy((i+j)-1,k) + glcm(i,j,k);
            %end
            %if (ismember(abs(i-j),[0:(size_glcm_1-1)]))
                p_xminusy((abs(i-j))+1,k) = p_xminusy((abs(i-j))+1,k) +...
                    glcm(i,j,k);
            %end
        end
    end

end

% marginal probabilities are now available [1]
% p_xminusy has +1 in index for matlab (no 0 index)
% computing sum average, sum variance and sum entropy:


%Q    = zeros(size(glcm));

i_matrix  = repmat([1:size_glcm_1]',1,size_glcm_2);
j_matrix  = repmat([1:size_glcm_2],size_glcm_1,1);
% i_index = [ 1 1 1 1 1 .... 2 2 2 2 2 ... ]
i_index   = j_matrix(:);
% j_index = [ 1 2 3 4 5 .... 1 2 3 4 5 ... ]
j_index   = i_matrix(:);
xplusy_index = [1:(2*(size_glcm_1)-1)]';
xminusy_index = [0:(size_glcm_1-1)]';
mul_contr = abs(i_matrix - j_matrix).^2;
mul_dissi = abs(i_matrix - j_matrix);
%div_homop = ( 1 + mul_contr); % used from the above two formulae
%div_homom = ( 1 + mul_dissi);

for k = 1:size_glcm_3 % number glcms

    out.contr(k) = sum(sum(mul_contr.*glcm(:,:,k)));
    out.dissi(k) = sum(sum(mul_dissi.*glcm(:,:,k)));
    out.energ(k) = sum(sum(glcm(:,:,k).^2));
    out.entro(k) = - sum(sum((glcm(:,:,k).*log(glcm(:,:,k) + eps))));
    out.homom(k) = sum(sum((glcm(:,:,k)./( 1 + mul_dissi))));
    out.homop(k) = sum(sum((glcm(:,:,k)./( 1 + mul_contr))));
    % [1] explains sum of squares variance with a mean value;
    % the exact definition for mean has not been provided in
    % the reference: I use the mean of the entire normalized glcm
    out.sosvh(k) = sum(sum(glcm(:,:,k).*((i_matrix - glcm_mean(k)).^2)));
    out.indnc(k) = sum(sum(glcm(:,:,k)./( 1 + (mul_dissi./size_glcm_1) )));
    out.idmnc(k) = sum(sum(glcm(:,:,k)./( 1 + (mul_contr./(size_glcm_1^2)))));
    out.maxpr(k) = max(max(glcm(:,:,k)));

    u_x(k)       = sum(sum(i_matrix.*glcm(:,:,k)));
    u_y(k)       = sum(sum(j_matrix.*glcm(:,:,k)));
    % using http://www.fp.ucalgary.ca/mhallbey/glcm_variance.htm for s_x
```

```
 % s_y : This solves the difference in value of correlation and might be
 % the right value of standard deviations required
 % According to this website there is a typo in [2] which provides
 % values of variance instead of the standard deviation hence a square
 % root is required as done below:
 s_x(k)  = (sum(sum( ((i_matrix - u_x(k)).^2).*glcm(:,:,k) )))^0.5;
 s_y(k)  = (sum(sum( ((j_matrix - u_y(k)).^2).*glcm(:,:,k) )))^0.5;

corp(k) = sum(sum((i_matrix.*j_matrix.*glcm(:,:,k))));
corm(k) = sum(sum(((i_matrix - u_x(k)).*(j_matrix -
 u_y(k)).*glcm(:,:,k))));

out.autoc(k) = corp(k);
out.corrp(k) = (corp(k) - u_x(k)*u_y(k))/(s_x(k)*s_y(k));
out.corrm(k) = corm(k) / (s_x(k)*s_y(k));

out.cprom(k) = sum(sum(((i_matrix + j_matrix - u_x(k) - u_y(k)).^4).*...
             glcm(:,:,k)));
out.cshad(k) = sum(sum(((i_matrix + j_matrix - u_x(k) - u_y(k)).^3).*...
             glcm(:,:,k)));




out.savgh(k) = sum((xplusy_index + 1).*p_xplusy(:,k));
% the summation for savgh is for i from 2 to 2*Ng hence (i+1)
out.senth(k) =  - sum(p_xplusy(:,k).*...
         log(p_xplusy(:,k) + eps));

 % compute sum variance with the help of sum entropy
 out.svarh(k) = sum((((xplusy_index + 1) - out.senth(k)).^2).*...
     p_xplusy(:,k));
     % the summation for savgh is for i from 2 to 2*Ng hence (i+1)

 % compute difference variance, difference entropy,
 % out.dvarh2(k) = var(p_xminusy(:,k));
 % but using the formula in
 % http://murphylab.web.cmu.edu/publications/boland/boland_node26.html
 % we have for dvarh
 out.denth(k) = - sum((p_xminusy(:,k)).*...
     log(p_xminusy(:,k) + eps));
 out.dvarh(k) = sum((xminusy_index.^2).*p_xminusy(:,k));

 % compute information measure of correlation(1,2) [1]
 hxy(k) = out.entro(k);
 glcmk  = glcm(:,:,k)';
 glcmkv = glcmk(:);

 hxy1(k) =  - sum(glcmkv.*log(p_x(i_index,k).*p_y(j_index,k) + eps));
 hxy2(k) =  - sum(p_x(i_index,k).*p_y(j_index,k).*...
     log(p_x(i_index,k).*p_y(j_index,k) + eps));
  hx(k) = - sum(p_x(:,k).*log(p_x(:,k) + eps));
  hy(k) = - sum(p_y(:,k).*log(p_y(:,k) + eps));
```

```
    out.inf1h(k) = ( hxy(k) - hxy1(k) ) / ( max([hx(k),hy(k)]) );
    out.inf2h(k) = ( 1 - exp( -2*( hxy2(k) - hxy(k) ) ) )^0.5;

    %     eig_Q(k,:)    = eig(Q(:,:,k));
    %     sort_eig(k,:)= sort(eig_Q(k,:),'descend');
    %     out.mxcch(k) = sort_eig(k,2)^0.5;
    % The maximal correlation coefficient was not calculated due to
    % computational instability
    % http://murphylab.web.cmu.edu/publications/boland/boland_node26.html


end



%       GLCM Features (Soh, 1999; Haralick, 1973; Clausi 2002)
%           f1. Uniformity / Energy / Angular Second Moment (done)
%           f2. Entropy (done)
%           f3. Dissimilarity (done)
%           f4. Contrast / Inertia (done)
%           f5. Inverse difference
%           f6. correlation
%           f7. Homogeneity / Inverse difference moment
%           f8. Autocorrelation
%           f9. Cluster Shade
%          f10. Cluster Prominence
%          f11. Maximum probability
%          f12. Sum of Squares
%          f13. Sum Average
%          f14. Sum Variance
%          f15. Sum Entropy
%          f16. Difference variance
%          f17. Difference entropy
%          f18. Information measures of correlation (1)
%          f19. Information measures of correlation (2)
%          f20. Maximal correlation coefficient
%          f21. Inverse difference normalized (INN)
%          f22. Inverse difference moment normalized (IDN)
```

```
function Features=Extract_ears0622(Blockpath)

Fname=Blockpath;
% Fname='C:\ wheatheads0622_3\WheatearBlock';
Fpath=[Fname,'\*.jpeg'];
filename=dir(Fpath);
file_length=length(filename);
Features=zeros(file_length/2,4);
for i=1:file_length/2
      k=0;
      PN=0;
      while PN~=2*i
            k=k+1;
            imagename=[Fname,'\',filename(k).name];
            P_NUM=TakePN1(filename(k).name,'N','E');
            PN=str2num(P_NUM);
      end
      Imagergb = imread(imagename);
      % imshow(Imagergb)
      [imagergb_cut,~,~,~,~]=Cutting_Im(Imagergb);
      %  figure(3),imshow(imagergb_cut)
      I_gray=rgb2gray(imagergb_cut);
      BW=im2bw(I_gray);
      Sumbw1=sum(BW(:));
      % figure(1),imshow(BW)
      % imgrec=snipimg(SX,SY,15,15,imagergb_cut,'RGB');
      % figure,imshow(imgrec)
      G=ColourA(imagergb_cut);
      % figure,imshow(G)
      BW1=uint8(G);
      BW2=logical(BW1);
      EG=BW2.*BW;
      % figure(2),imshow(EG);
      Sumbw2=sum(EG(:));
      Per=100*(Sumbw1-Sumbw2)/Sumbw1;

       Features(i,2)=Sumbw1;
       Features(i,3)=Sumbw2;
       Features(i,4)=Per;
       Features(i,1)=i;
       % end
end

path=[Fname,'\NoGreenCoverage.xls'];
xlswrite(path,Features,1);
```

```
function Morphology=Skeleton0701(Blockpath,Features,TH)
[myfit,stats,NUM1]=wtearscm();
Fname=Blockpath;
Fpath=[Fname,'\*.jpeg'];
filename=dir(Fpath);
file_length=length(filename);
Morphology=zeros(file_length/2,8,3);
AREA=zeros(file_length/2,13);
Number=0;
for i=1:file_length/2
      if Features(i,4)<TH
            Number=Number+1;
            k=0;
            PN=0;
            while PN~=2*i-1
                  k=k+1;
                  imagename=[Fname,'\',filename(k).name];
                  P_NUM=TakePN1(filename(k).name,'N','E');
                  P_NAM=TakePN1(filename(k).name,'E','.');
                  PN=str2num(P_NUM);
            end
             Imagebw = imread(imagename);
            STATS=countingwte(Imagebw);
            Morphology(Number,8,2)=STATS(1,13);
            AREA(Number,:)=STATS;
            NUM(1,1)=STATS(1,2);
            NUM(1,2)=STATS(1,3);
            NUM(1,3)=STATS(1,4);
            NUM(1,4)=STATS(1,7);
            NUM(1,5)=STATS(1,9);
            NUM(1,6)=STATS(1,11);
            Morphology(Number,8,3)=wtecum(NUM,myfit);
             %imshow(Imagebw)
             [imagebw_cut,~,~,~,~]=Cutting_Im(Imagebw);
            I_gray=rgb2gray(imagebw_cut);
            BW=im2bw(I_gray);
            BW=imfill(BW,'holes');
            RBW=~BW;
             RBW=imfill(RBW,'holes');
             BW=~RBW;
             %figure,imshow(BW)
            h = fspecial('disk',20);
            FBW = imfilter(BW,h);
            FBW1 =FBW;
            figure(1),imshow(FBW)
            counter=0;
            [Heteromor,~,~]=D_Heteromorphism(FBW,15);
            counter=counter+Heteromor;
            [Heteromor,~,~]=D_Heteromorphism(FBW,30);
            counter=counter+Heteromor;
            [Heteromor,~,~]=D_Heteromorphism(FBW,35);
            counter=counter+Heteromor;
            if counter==3
                Heteromor=1;
            else
                Heteromor=0;
            end
```

```
                FORNUM=[P_NAM,'E','.'];
                FORNUMSXY=['P',TakePN1(FORNUM,'P','S'),'S'];
                SX_NUM=TakePN1(FORNUMSXY,'P','X');
                SY_NUM=TakePN1(FORNUMSXY,'X','S');
                FORNUMSIZE=['Z',TakePN1(FORNUM,'Z','B'),'B'];
                XSIZE_NUM=TakePN1(FORNUMSIZE,'Z','X');
                YSIZE_NUM=TakePN1(FORNUMSIZE,'X','B');
                Morphology(Number,1,1)=i;
                Morphology(Number,2,1)=Heteromor+1;
                Morphology(Number,3,1)=str2num(SX_NUM);
                Morphology(Number,4,1)=str2num(SY_NUM);
                Morphology(Number,5,1)=str2num(XSIZE_NUM);
                Morphology(Number,6,1)=str2num(YSIZE_NUM);
                Morphology(Number,7,1)=Morphology(Number,5,1)*Morphology(Number,6
                ,1);
                if Heteromor+1==1
                    if Morphology(Number,7,1)>20000
                        Morphology(Number,8,1)=2;
                    else
                        Morphology(Number,8,1)=1;
                    end
                else
                   h = fspecial('disk',10);
                   FBW = imfilter(BW,h);
                   Skeleton=bwmorph(FBW,'thin',100);
                   imname=['S',num2str(i),'E',P_NAM];
                   figure(1),imshow(Skeleton);
                   title(imname);
                   path=[Fname,'\',imname,'.jpeg'];
                   print(1,'-djpeg',path);
                   a=bwmorph(Skeleton,'branchpoints');
                   [~,num]=bwlabel(a,8);
                   Morphology(Number,8,1)=num+1;
                end
                Morphology(Number,1:7,2)=Morphology(Number,1:7,1);
                Morphology(Number,1:7,3)=Morphology(Number,1:7,1);
                AREA(Number,1)=Morphology(Number,1,1);

        end
end
path=[Fname,'\ wheatheads.xls'];
xlswrite(path,Morphology(:,:,1),1);
xlswrite(path,Morphology(:,:,2),2);
xlswrite(path,Morphology(:,:,3),3);
xlswrite(path,AREA,4);
xlswrite(path,myfit,5);
xlswrite(path,stats,6);
xlswrite(path,NUM1,7);
imname=['No-Green Coverage Stem'];
x=1:file_length/2;
figure(1), stem(x,Features(:,4));
title(imname);
xlabel('Number of suspected wheatear block');
ylabel('No-Green coverage(%)');
path=[Fname,'\',imname,'.jpeg'];
print(1,'-djpeg',path);
```

```
function AREA=countingwte(image)
AREA=zeros(1,13);
[imgcut,~,~,~,~]=Cutting_Im(image);
I_gray=rgb2gray(imgcut);
BW=im2bw(I_gray);
% figure, imshow(BW)
 [x,y]=size(BW);
% sump=sum(BW(:))/(x*y)
BWS=zeros(x+200,y+200);
BWS(102:x+97,102:y+97)=BW(2:x-3,2:y-3);
% figure, imshow(~BWS)
STATS = regionprops(BWS,'all');
AREA(1,2)=getfield(STATS,'Area');
AREA(1,3)=getfield(STATS,'MajorAxisLength');
AREA(1,4)=getfield(STATS,'MinorAxisLength');
AREA(1,5)=getfield(STATS,'Eccentricity');
AREA(1,6)=getfield(STATS,'Orientation');
AREA(1,7)=getfield(STATS,'ConvexArea');
AREA(1,8)=getfield(STATS,'EquivDiameter');
AREA(1,9)=getfield(STATS,'Solidity');
AREA(1,10)=getfield(STATS,'Extent');
AREA(1,11)=getfield(STATS,'Perimeter');
AREA(1,12)=getfield(STATS,'PerimeterOld');
AREA(1,1)=1;
a=getfield(STATS,'ConvexArea');
b=getfield(STATS,'Area');
% cimage=getfield(STATS,'ConvexImage');
% figure, imshow(~cimage)
AREA(1,13)=fix(((a-b)/b)*10)+1;
```

```
function [myfit,stats,NUM]=wtearscm()

GCCD='WheatearCountingModelDatasbase0704.xls';
NUM=xlsread(GCCD,1);
[xs,~]=size(NUM);
% Y=zeros(xs,1);
X=zeros(xs,16);
Y=NUM(:,8);

X(:,2)=NUM(:,2)./(NUM(:,3).*NUM(:,4));
X(:,1)=1;
X(:,3)=(NUM(:,5)-NUM(:,2))./NUM(:,2);
X(:,4)=NUM(:,6);
X(:,5)=NUM(:,7)./(2*(NUM(:,3)+NUM(:,4)));
X(:,6)=X(:,2).*X(:,3);
X(:,7)=X(:,2).*X(:,4);
X(:,8)=X(:,2).*X(:,5);
X(:,9)=X(:,3).*X(:,4);
X(:,10)=X(:,3).*X(:,5);
X(:,11)=X(:,4).*X(:,5);
X(:,12)=(X(:,2).*X(:,3)).*X(:,4);
X(:,13)=(X(:,2).*X(:,3)).*X(:,5);
X(:,14)=(X(:,3).*X(:,4)).*X(:,5);
X(:,15)=(X(:,4).*X(:,5)).*X(:,2);
X(:,16)=((X(:,2).*X(:,3)).*X(:,4)).*X(:,5);

[myfit,~,~,~,stats]=regress(Y,X,0.05);
NUM(:,9)=round(X*myfit);
```

```
function wheatear=wtecum(NUM,myfit)

X(1,2)=NUM(1,1)./(NUM(1,2).*NUM(1,3));
X(1,1)=1;
X(1,3)=(NUM(1,4)-NUM(1,1))./NUM(1,1);
X(1,4)=NUM(1,5);
X(:,5)=NUM(:,6)./(2*(NUM(:,2)+NUM(:,3)));

X(1,6)=X(1,2).*X(1,3);
X(1,7)=X(1,2).*X(1,4);
X(1,8)=X(1,2).*X(1,5);

X(1,9)=X(1,3).*X(1,4);
X(1,10)=X(1,3).*X(1,5);

X(1,11)=X(1,4).*X(1,5);

X(1,12)=(X(1,2).*X(1,3)).*X(1,4);
X(1,13)=(X(1,2).*X(1,3)).*X(1,5);
X(1,14)=(X(1,3).*X(1,4)).*X(1,5);
X(1,15)=(X(1,4).*X(1,5)).*X(1,2);

X(1,16)=((X(1,2).*X(1,3)).*X(1,4)).*X(1,5);

wheatear=round(X*myfit);
if wheatear==0
    wheatear=1;
end
```

```
function Image_labeled=LabelNum(Morphology,image_cut)
 [Block,~]=size(Morphology);
Image_labeled=image_cut;
for i=1:Block
     xs=fix(Morphology(i,3)+0.5*Morphology(i,5)-5);
    if xs<1
           xs=Morphology(i,3);
    end
    ys=fix(Morphology(i,4)+0.5*Morphology(i,6)-5);
    if ys<1
           ys=Morphology(i,4);
    end
    for j=1:Morphology(i,8)
          Image_labeled=ImLabeled(Image_labeled,xs,ys+(j-1)*15,3);
    end
end
```