Received 26 February 2020; revised 31 May 2020 and 2 July 2020; accepted 14 July 2020. Date of publication 10 August 2020; date of current version 2 September 2020. The review of this article was arranged by Associate Editor S. Kaarthik.

Digital Object Identifier 10.1109/OJIA.2020.3015253

High-Accuracy Adaptive Low-Cost Location Sensing Subsystems for Autonomous Rover in Precision Agriculture

SAMUEL J. LEVOIR [1,2], PETER A. FARLEY [1], TAO SUN [3], AND CHONG XU [1] (Senior Member, IEEE)

[1]School of Engineering, University of St. Thomas, St Paul, MN 55105, USA
[2]Open Systems International, Medina, MN 55340, USA
[3]School of Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

CORRESPONDING AUTHOR: CHONG XU (e-mail: chong.xu@stthomas.edu)

ABSTRACT With the prosperity of artificial intelligence, more and more jobs will be replaced by robots. The future of precision agriculture (PA) will rely on autonomous robots to perform various agricultural operations. Real time kinematic (RTK) assisted global positioning systems (GPS) are able to provide very accurate localization information with a detection error less than ±2 cm under ideal conditions. Autonomously driving a robotic vehicle within a furrow requires relative localization of the vehicle with respect to the furrow centerline. This relative location acquisition requires both the coordinates of the vehicle as well as all the stalks of the crop rows on both sides of the furrow. This extensive number of coordinate acquisitions of all the crop stalks demand onerous geographical survey of entire fields in advance. Additionally, real-time RTK-GPS localization of moving vehicles may suffer from satellite occlusion. Hence, the above-mentioned ±2 cm accuracy is often significantly compromised in practice. Against this background, we propose sets of computer vision algorithms to coordinate with a low-cost camera (50 US dollars), and a LiDAR sensor (1500 US dollars) to detect the relative location of the vehicle in the furrow during early, and late growth season respectively. Our solution package is superior than most current computer vision algorithms used for PA, thanks to its improved features, such as a machine-learning enabled dynamic crop recognition threshold, which adaptively adjusts its value according to the environmental changes like ambient light, and crop size. Our in-field tests prove that our proposed algorithms approach the accuracy of an ideal RTK-GPS on cross-track detection, and exceed the ideal RTK-GPS on heading detection. Moreover, our solution package neither relies on satellite communication nor advance geographical surveys. Therefore, our low-complexity, and low-cost solution package is a promising localization strategy as it is able to provide the same level of accuracy as an ideal RTK-GPS, yet more consistently, and more reliably, as it requires no external conditions or hassle of the work demanded by RTK-GPS.

INDEX TERMS Computer vision, machine learning, navigation, precision agriculture, crop row detection, localization, LiDAR.

I. INTRODUCTION

Precision agriculture (PA) has become a trending research topic due to the increasing demands for environmental protection, cost reduction and yield enhancement [1], [2]. The current aggregational approaches of irrigation, weeding and insect control inevitably use more resources than needed. Additionally, the extensive use of herbicide and pesticide has posed potential danger to animals and human beings by leaving toxic chemical components built up in the soil to a level that eventually infiltrate crops and livestock.

Driven by the above limitations of current agricultural methodologies, the University of St. Thomas is cooperating with Rx Robotics and Molitor Brothers Farm to propose a new set of solutions to precisely manage the farm fields to achieve

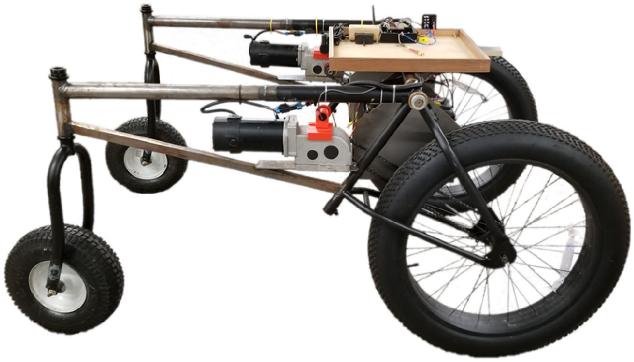


FIGURE 1. Rx Robotics autonomous horticultural rover, the 'bikebot'. The rover is currently an electrically powered wheeled vehicle, which will be remodeled with tracks.

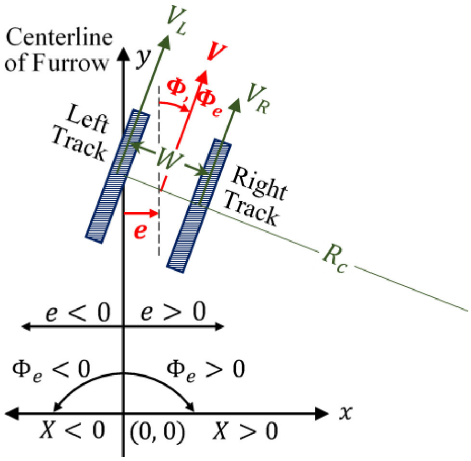


FIGURE 2. Illustration of cross-track error e and heading error Φ_e, as well as the geometric relationship of the motion of the rover, when the furrow is assumed to be straight around the rover.

higher productivity at a lower cost while also preserving the environment. The PA system utilizes small-scale autonomous vehicles equipped with robotic arms to collect field samples and performs various agricultural operations. The collected samples will be used to evaluate the soil health and plant conditions, and employed to derive actions on plants or soil via machine learning algorithms such as neural networks [3], [4]. Moreover, the vehicle will also be used to co-plant cover crops to suppress weed growth instead of herbicides, maintain the nitrates from being leached, and eliminate the cost incurred by the expensive hybrid seeds for herbicide tolerance.

The robotic vehicle should be able to self-navigate through the narrow furrows, and self-charge through a dock, so that it is able to work 24/7/365 with little to no human interventions. We have designed and implemented an electrically powered rover, as shown in Figure 1, to serve as the above-mentioned vehicular platform. The papers [5] detailed the dynamic inversion based navigation algorithm employed by the rover. More specifically, the navigation subsystem requires the cross-track and heading errors of the rover with respect to the centerline of the furrow as the input variables. As shown in Figure 2, the cross-track error e is defined as the normal distance from the centerline of the furrow to the rover center, with right offset being positive. The heading error Φ_e is the angular difference between the heading of the rover and heading of the furrow. The heading error is positive if the rover is skewed clockwise from the heading of the centerline of the furrow.

To detect the relative location of the rover with respect to the furrow precisely, the sensing subsystem can either use a Global Navigation Satellite System (GNSS), or camera and/or light reflective based sensors, such as LiDAR, sonar, radar, etc. Real time kinematic (RTK) techniques can further enhance the accuracy of GNSS. Nevertheless, to calculate the relative position of the rover with respect to the furrow, not only are the coordinates of the rover required, but those of each point constituting the furrow are also required. This *a priori* information calls for large amount of work on geographical survey in advance. Besides, the aforementioned RTK accuracy requires a constant connection to a large number of satellites, which cannot be obtained during occlusion of GNSS satellites. Admittedly, the Kalman filter and its variants can be used to fuse the GNSS data with inertial measurement unit (IMU) measurements to estimate the rover positions [29]. Chiang *et al.* [30] reported a detection error of 8∼18 cm when the GPS signal is lost for 60 seconds during highway travel. All other similar fusion positioning systems reported higher detection error [29].

Nevertheless, navigation in precision agriculture requires a higher accuracy on localization. Hence, we propose to use a low-cost camera and LiDAR as the sensors to detect the relative location of the rover with respect to the furrow, during the early and late growth season of the crops respectively. The raw image data acquired by the camera or LiDAR are processed using computer-vision algorithms, through which the relative geographical location is extracted. This process relies only on local devices (no satellite connection as GPS) and hence is more reliable. The relative location is obtained without any advance geographical survey and hence is more convenient to use.

These advantages have motivated the computer vision technologies to mushroom since 1980s [6]. It has been extensively used for indoor robotic navigation and outdoor transportation.

The NAVLAB1 [7]–[10] has used the red, green, blue (RGB) pyramid and surface texture to distinguish the road and non-road pixels, and Hough line approach to find the exact edge lines. The NAVLAB1 also employs ALVINN (Autonomous Land Vehicle In a Neural Network), which is a Gaussian distribution enabled continuous neural network to determine the steering angle corresponding to a captured video image [11]. It realizes road edge detection and navigation via a neural network, whose input is the raw image and output is the steering angle. Another leading work in computer-vision based road-following was done by Turk *et al.* [12], who designed a system called VITS (VIsion

Task Sequencer). It is able to combinatively consider the factor of current position, speed, heading, etc., to make a projection of the road boundary and create a road map for the vehicle. Equally prominent is the research conducted for vehicles travelling on the 'Autobahns' [13], [14] and all the derived work [15], [16], including the EUREKA-project Prometheus [17], [18], which aimed at improving the traffic safety by developing a system to warn the drivers when they are off-track. Other distinguished research work focusing on computer-vision based road following include [19], [27], [31], [32], [36].

Admittedly, computer vision has been extensively used in self-driving and driver assistance systems in transportation as stated above. Nevertheless, operation in precision agriculture poses new challenges to localization of autonomous navigation for the following reasons:

- The furrow is much narrower than a highway road, hence allowing for less trajectory oscillation.
- The agricultural vehicle is uniquely equipped with two tracks as to reduce pressure to the soil, while most vehicles in transportation have wheeled structures.
- The aforementioned two major differences evoke a different navigation algorithm [5], which relies on high-precision detection of cross-track (lateral) distance and heading values of the rover with respect to the furrow centerline to produce a narrowly bounded trajectory without damaging the crops.
- Unlike most roads, the furrows do not have painted lanes or physically existent boundaries. The scattered bottom of each crop stalk forms an imagined boundary of the furrow. Detection of these imagined boundaries are interfered with the ever growing crop leaves and ever changing ambient light.
- Unlike in transportation, there are much more furrows present in one picture in agricultural fields. The irrelevant furrows may lead to compromised localization accuracy, if all the furrows are not equidistant and parallel.
- The challenges presented in the above two items calls for a more robust and effective crop row detection algorithm.
- When the crops are taller than a certain height, the computer vision based algorithm may not be efficacious to detect the crop rows and a LiDAR will be used instead of a camera.

These new challenges posed by precision agriculture have inspired a large amount of research in crop row detection and autonomous rover localization. In 1987, Searcy *et al.* [37] presented the primary steps a computer-vision based crop-row sensing subsystem needs to include and serves as the foundation followed by many other researchers [38]–[40]. Most monocular based computer vision solutions contain the following fundamental steps: crop recognition, furrow edge formulation, perspective (camera-view to top-view) conversion, cross-track and heading calculation. Crop recognition can be done either via pixel segmentation or column-wise comparison. Pixel segmentation is a series of operations, where a threshold of a color (normally green) index is set to filter out the crop pixels and a black-and-white (BW) binary image is consequentially generated [45]. The common furrow edge formulation methods following segmentation approach are the Hough transform [48] and its variations [42], [46], [47], where a best line of fit will be generated penetrating as many crop pixels as possible. However, segmentation with a constant threshold is not very robust against inevitable environmental changes, such as crop size and light conditions. To combat this challenge, an opening operation may be conducted to remove the small bright pixels in the foreground [42] before a constant threshold is applied.

Another commonly used crop row recognition approach is using column-wise comparison to decide the location of the crop edges in an image. When this method is applied, the by-column change of a certain indicator, normally the green value or a linear combination of RGB values, is recorded. One branch of this method is to sum the indicator per column and locate the crop rows either by finding the peaks of the achieved waveform [41], [43] or the median value of the row histogram [44]. Another branch divides the whole image into strips and finds the furrow edge points per strip followed by a linear regression to figure out the complete furrow edge [40]. Nevertheless, all these algorithms have assumed a known row spacing [51]. Winterhalter *et al.* [51] proposed a Pattern Hough and Pattern RANSAC based crop row detection and localization algorithm, which detect all the rows in 3D camera or laser images simultaneously by exhaustively searching through all the possible combinations of the normal distances and angles of all the supposedly parallel and equidistant crop rows. However, this method has relatively high computation complexity due to the exhaustive searching process. Besides, more advanced sensors such as stereo cameras [49], infrared cameras [42], [50], or 3D laser scanners [51] may also be used to provide more informative raw images to facilitate the localization process.

Standing on the shoulders of all the above-mentioned and many other giants, we propose an alternative approach to detect the lateral and heading values of the rover in a low complexity manner with low cost 2d camera ($50) and LiDAR ($1500). The major differences between our findings and the above-mentioned previous excellent achievements lie in the following aspects. First, during early growth stage, we set a dynamic threshold to segment the crops, which is able to change with environment. The dynamic threshold is derived from two inputs that can be read directly from the images. These two inputs are indicative of the two most influential varying environmental factors, the crop volume and the ambient light intensity respectively. The exact formula of the threshold as a function of the two inputs is obtained via a heuristic machine-learning approach. We denote this process as the *adaptive RGB filtering algorithm* in our paper.

Second, while most of the aforementioned papers presented results only when the rover is aligned with the furrow, we did extensive field tests with a large variety of preset cross-track and heading offsets ranging from -16 cm to $+16$ cm and -20° to $+20^\circ$, respectively. Through the test, we found the detection results to be increasingly inaccurate as the cross-track offset gets bigger. The worsened detection error in misaligned positions has led us to find out the non-negligible interference imposed by the body of the tall crops. We quantify the negative effect as a function of an input from the image via heuristic machine learning approach and compensate for the interference with the aid of the formula. This process is denoted as the *anti overinflation process* in our paper.

Third, we fuse the information from the two track tachometers with the previous location detection result to estimate the current location of the rover, and calculate the predicted pixel position of the centerline. This will help us filter out the unwanted furrows, which the rover is not currently between. We name this strategy the *triangle filtering* in our paper. Fourth, rather than an ordinary linear regression approach, we develop an *inverted linear regression approach* to more accurately obtain the furrow edge line. Last but not least, during the late growth stage, the crop rows are detected through horizontal scanning starting from the estimated centerline of the furrow in the 2d point cloud obtained by the LiDAR sensor.

All of the above-mentioned meticulous operations result in a greatly improved localization precision. Our field tests indicate that our solution achieves an average cross-track and heading detection error of 0.09 cm and 0.43° with a standard deviation of 2.53 cm and 0.83° under all light and crop-size circumstances and all possible preset positions during early growth stage, and an average detection error of 0.83 cm and 0.37° with standard deviation of 1.38 cm and 0.94° for cross-track and heading respectively during late growth stage.

This paper presents the methodologies and results of the research continued from the IEEE proceeding paper [52].

The major improvements from the proceeding paper to this treatise are reflected in the following aspects:

- This paper introduces the *adaptive RGB filtering*, which quantitatively evaluates the dynamic threshold that changes based on the crop volume and ambient light condition.
- *Triangle filtering* is developed and presented in this paper, which can remove the irrelevant interfering crop rows.
- *Inverted linear regression* and Deming regression are introduced to acquire the crop row edges.
- *Anti overinflation adjustment* is introduced to compensate for the detection error incurred by the crop height.
- While the conference paper only focused on the early growth stage, we introduce a LiDAR based low-complexity localization algorithm for late growth stage in this treatise.
- While the conference paper only presented results from preliminary test done with tapes on floor imitating the crop rows, we did extensive field tests throughout the early and late growth stage of the crops, during different times of the day, with a large number of combinations of preset cross-track and heading values.
- On top of that, six combinations of two different crop recognition approaches and three furrow edge formulation methods are adopted to process the experimental data and the results are compared.

The rest of this paper is organized in the following way: The algorithm for early growth season will be introduced in Section II. The proposed triangle filtering, the adaptive RGB filtering algorithm, the anti-overinflation correction will be introduced in Section II-B, II-C1 and II-E. Late growth localization approach will be introduced in Section III. The results of all six combinations of the algorithms introduced in this paper will be presented in Section IV. The paper will be concluded in Section V.

II. EARLY GROWTH SUBSYSTEM

The crops' early growth stage is defined as the time from when seeds germinate from the ground till when the height of the crops are approximately 45 cm. During this stage, the color of the crops is visually distinguishable from the dirt around it, and the stocks of crops are not tall enough to reflect the beams emanated from the light-reflective sensors such as radar, sonar or LiDAR installed on the rover. Hence, image-based sensors like a camera have a better opportunity to capture the edges of the furrow.

Every 0.02 seconds, an image of the field is taken with a camera facing the direction of the rover's heading and located on top of the rover. To best capture the information, we tilt the camera with such an angle that the center of the camera will face the point on the ground that is 2.25 meters in front. This set up will ensure enough crop pixels present in the obtained picture, yet is not that long to undermine the assumption that the furrow edges appearing in the images are two straight lines.

The overall flow chart of the entire computer-vision based algorithm for early growth season is presented in Figure 3. More details of each process will be given in the following subsections.

A. IMAGE CAPTURE AND UNDISTORTION

All cameras are subject to symmetrical radial distortion and tangential distortion [21], [22] caused by deviation from rectilinear projection and non-parallel alignment between the lens and the image recording component [23]. The effects of these distortions can be clearly observed in the raw image of a chessboard taken with the camera used by the same group of authors in this treatise, as seen in Figure 2(a) of [52], while the corrected image of it is shown in Figure 2(b) of [52].

On the one hand, the adjustments needed to apply to any pixel in the raw image caused by radial distortion can be formulated as [24]–[26]:

$$\mathbf{C}_{\mathrm{r}} = \begin{bmatrix} C_{\mathrm{r}x} \\ C_{\mathrm{r}y} \end{bmatrix} = \begin{bmatrix} \delta_x(K_1 r^2 + K_2 r^4 + \cdots) \\ \delta_y(K_1 r^2 + K_2 r^4 + \cdots) \end{bmatrix}, \tag{1}$$

where $C_{\mathrm{r}x}$ and $C_{\mathrm{r}y}$ are the amount of revision needed to apply to pixel $(x_{\mathrm{d}}, y_{\mathrm{d}})$ in the raw distorted image in both axes, to correct for the radial distortion.

In the equation, the location of the pixel $(x_{\mathrm{d}}, y_{\mathrm{d}})$ in the raw image is alternatively defined by δ_x and δ_y, the horizontal and

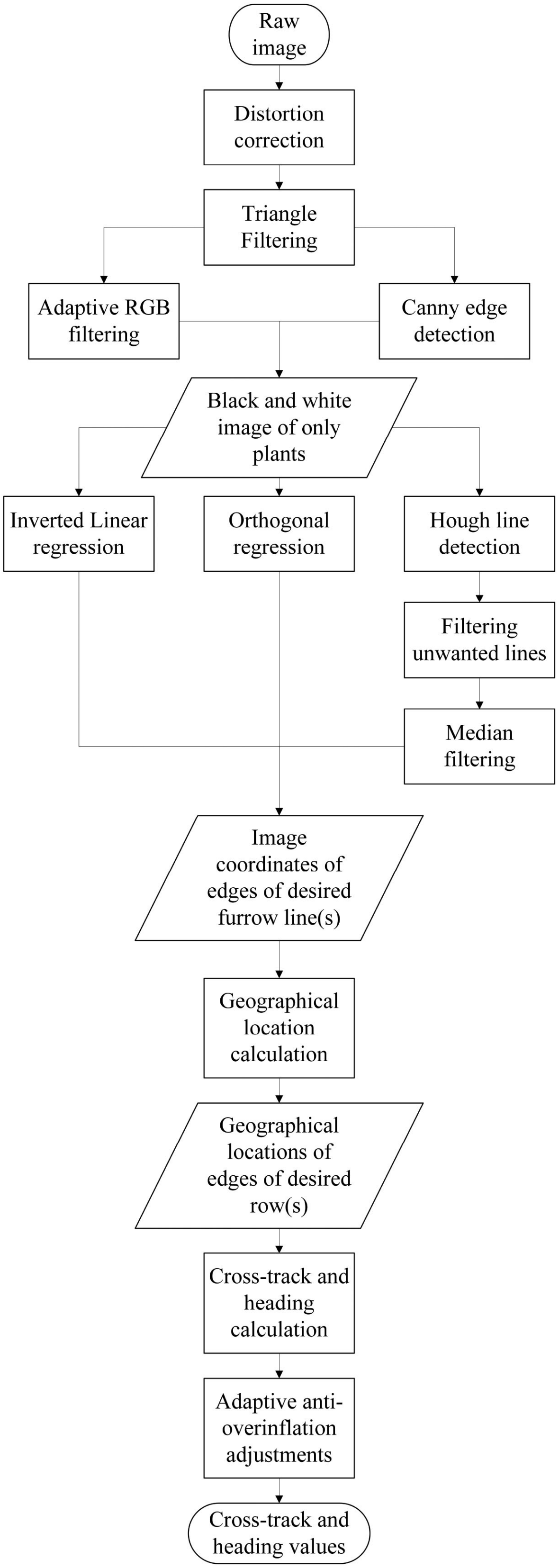

Raw image
Distortion correction
Triangle Filtering
Adaptive RGB filtering
Canny edge detection
Black and white image of only plants
Inverted Linear regression
Orthogonal regression
Hough line detection
Filtering unwanted lines
Median filtering
Image coordinates of edges of desired furrow line(s)
Geographical location calculation
Geographical locations of edges of desired row(s)
Cross-track and heading calculation
Adaptive anti-overinflation adjustments
Cross-track and heading values

FIGURE 3. A flowchart outlining all the possible steps to process the raw image taken by the camera to get the cross-track error and heading error during the early growth season. Please note that ovals are input/outputs, rectangles are processes, and rhombuses are data.

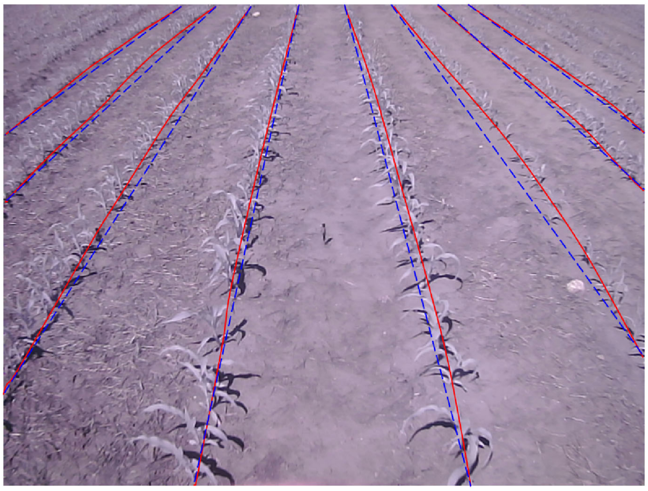


FIGURE 4. Distorted raw image taken with ELPUSBFHD01 M in the corn field. The red curves and blue dashed straight lines were both manually added by the authors. The red curves delineating the crop rows were created by connecting each contact point between the corn stalks and the dirt, while the blue dashed straight lines were created by directly connecting the lower and upper ending points of each crop row in the image with a straight line. Obviously, the blue straight lines are not on the crops.

vertical distances from the pixel to the center of the distorted image, namely $\delta_x = x_\mathrm{d} - x_\mathrm{c}$, $\delta_y = y_\mathrm{d} - y_\mathrm{c}$. The radial distance from the pixel to the center of the distorted image is then denoted as r, where $r = \sqrt{\delta_x^2 + \delta_y^2}$.

On the other hand, the adjustment caused by the tangential distortion can be formulated as:

$$\mathbf{C}_\mathrm{t} = \begin{bmatrix} C_{\mathrm{t}x} \\ C_{\mathrm{t}y} \end{bmatrix} = \begin{bmatrix} P_1(r^2 + 2\delta_x^2) + 2P_2\delta_x\delta_y \\ 2P_1\delta_x\delta_y + P_2(r^2 + 2\delta_y^2) \end{bmatrix}. \tag{2}$$

In both Equations (1) and (2), K_n and P_n are the nth order radial and tangential distortion coefficients, which can be obtained through a training process. Finally, the total amount of adjustments needed to be done to a pixel in the distorted raw image is the summation of $\mathbf{C}_\mathrm{r}$ and $\mathbf{C}_\mathrm{t}$. That is,

$$\mathbf{C} = \mathbf{C}_\mathrm{r} + \mathbf{C}_\mathrm{t} = \begin{bmatrix} C_x \\ C_y \end{bmatrix} = \begin{bmatrix} C_{\mathrm{r}x} + C_{\mathrm{t}x} \\ C_{\mathrm{r}y} + C_{\mathrm{t}y} \end{bmatrix}. \tag{3}$$

Interested readers are welcome to refer to the IEEE proceeding paper [52] for more details about the application process of the correction algorithm.

As an example, Figure 4 is a raw image taken with the camera. As can be seen from the picture, the crop rows are delineated with the red bulged curves, while the blue straight lines are depicted as benchmarks. Clearly, the crop rows in the raw image contort from their original straight-line shape. After applying Equations (1)∼(3), the raw image can be rectified as shown in Figure 5. Now the crop rows are recovered to their original shape as being depicted by straight lines. Last but not the least, as the objects in the distorted raw image swell out as compared to their actual shape, the rectified undistorted images are hence ‘shrunk back’ to the center, leaving some

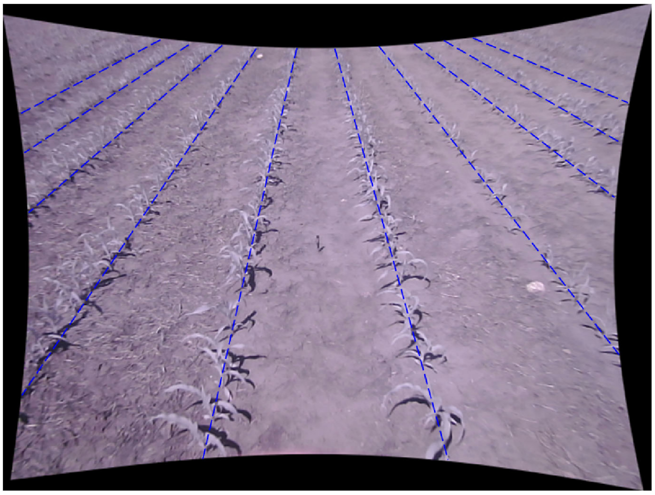


FIGURE 5. Rectified image of Figure 4. The blue dashed straight lines were manually added by the authors. The blue dashed straight lines delineating the crop rows were created by directly connecting the lower and upper ending points of each crop row in the image with a straight line.

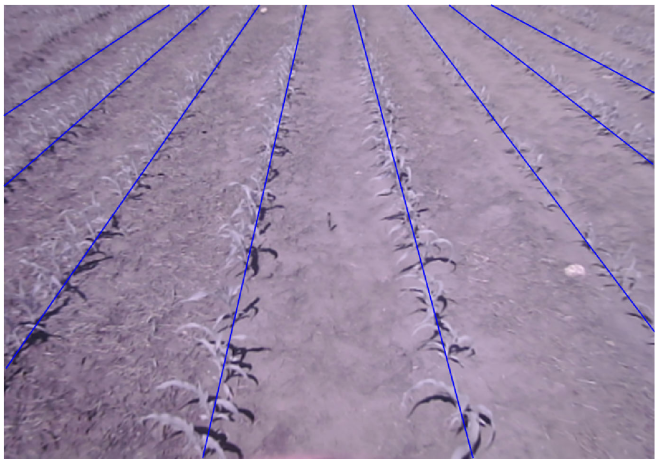


FIGURE 6. Cropped image of Figure 5. The blue dashed straight lines were manually added by the authors. The blue dashed straight lines delineating the crop rows were created by directly connecting the lower and upper ending points of each crop row in the image with a straight line.

blackout areas on the edges of the image, as shown in Figure 5. To reduce the interference from these blackout areas to the true information conveyed by the images, the images need to be cropped as shown in Figure 6.

B. TRIANGLE FILTERING AND CENTERLINE CALCULATION

Intrinsically, there are multiple crop rows in each distortion-corrected and cropped picture as shown in Figure 6. However, only the furrow traversed by the rover is of interest and all the location detection algorithms introduced in this treatise are developed to prevent the rover from running over the crop row on either side of the interested furrow. Our experiments have indicated that the inclusion of crop rows other than these two rows of concern in the image interfere with, rather than facilitate, the location detection process. Hence, as shown in Figure 3, an extra step called 'triangle filtering' is applied to filter out the interfering crop rows before any approach is invoked to identify the crop row pixels in the image.

To do this, *a priori* location information of the rover is required to estimate the pixel location of the centerline in the image. To be more exact, the pixel location of the centerline in the image is determined by the cross-track and heading errors of the furrow. Hence, we will estimate the *a priori* cross-track error $\check{e}_k$ and heading error $\check{\Psi}_{e,k}$ of the rover at the current moment using dead reckoning method, based on the location of the rover at the previous moment and measurements of the track speeds at the current moment.

More quantitatively, the heading rate of the rover in radian/sec can be calculated as [5]:

$$\dot{\Psi} = \frac{V_{L,\text{act}} - V_{R,\text{act}}}{W}, \tag{4}$$

where $V_{L,\text{act}}$ and $V_{R,\text{act}}$ are the measurements of the actual left and right track speed in m/s obtained with encoders, and W is the width of the rover. Hence, the estimated heading of the rover at the current moment can be calculated as

$$\Psi_k = \Psi_{k-1} + \dot{\Psi} \cdot dt, \tag{5}$$

where dt is the sampling period. According to the rover kinematics [5], the fraction of the rover speed on x and y Cartesian axes can be computed as:

$$\begin{aligned} V_k &= \frac{V_{L,\text{act}} - V_{R,\text{act}}}{2} \\ V_x &= V_k \sin(\Psi_k) \\ V_y &= V_k \cos(\Psi_k), \end{aligned} \tag{6}$$

where V_k is the computed rover speed along its heading at the current moment. Therefore, the location coordinates of the rover at the current moment (X_k, Y_k) can be derived as:

$$\begin{aligned} X_k &= X_{k-1} + V_x \cdot dt \\ Y_k &= Y_{k-1} + V_y \cdot dt \end{aligned} \tag{7}$$

As a result, the *a priori* estimation of the cross-track error and heading error of the rover at the current moment can be concluded as:

$$\begin{aligned} \check{e}_k &= X_k - X_0 \\ \check{\Psi}_{e,k} &= \Psi_k - \Psi_{\text{row}}, \end{aligned} \tag{8}$$

where X_0 and Ψ_{row} are the x coordinate and heading of the furrow centerline. Here we assume that $X_0 = 0$ and $\Psi_{\text{row}} = 0$ for simplicity. That is, we assume that the centerline of the furrow is aligned with the y axis. This assumption doesn't affect the development and application of the location detection algorithms introduced in this treatise as the detection results, namely the cross-track and heading errors are relative position information of the rover with respect to the centerline of the furrow.

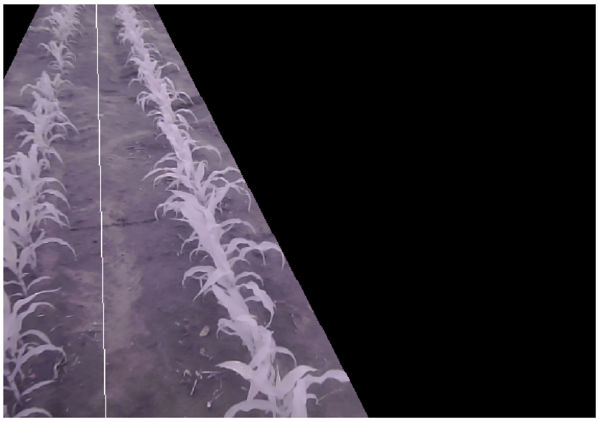


FIGURE 7. An example of applying the triangle filtering to the distortion-corrected and cropped image. The areas filtered out are masked in black, while the estimated centerline is depicted in white.

Now to decide the pixel location of the centerline in the image captured by a camera located $\check{e}_k$ meters away, and facing $\check{\Psi}_{e,k}$ degrees away from the centerline of the furrow, we went through a set of images with known cross-track and heading errors and recorded the pixel locations of the centerlines' intersections with the top and bottom edges of each image, which are denoted as P_t and P_b respectively. We then plotted these sets of P_t and P_b values against the known heading and cross-track values to determine the relationship between them. Then, by fitting a line to the plotted data, the following empirical equations are obtained to locate the centerline in the picture:

$$P_b = 0.5N_c - 7.5\check{\Psi}_{e,k} - 4.5\check{e}_k$$
$$P_t = 0.5N_c \times (1 - 0.02\check{\Psi}_{e,k}) - 3.75\check{\Psi}_{e,k} - 0.75\check{e}_k \quad (9)$$

where P_b (and P_t) is the column index of the intersection between the centerline and the bottom (and top) edge of the image. To be more exact, both $(P_b - 1)$ and $(P_t - 1)$ are the number of pixels between the intersections and the left edge of the image. In Equation (9), N_c is the total number of columns in the distortion-corrected and cropped images. It may be worthy to mention that the numbers in Equation (9) are dependent on the down angle of the camera, which is 31.37 degrees below horizon here.

An example of the estimated centerline can be seen in Figure 7. After the estimated centerline is calculated, any pixels beyond a certain distance f_r from the centerline on both sides at the rth row will be filtered out from the image. As the camera is mounted on the rover with a fixed height and angle facing the ground [52], this set of distances $f_r, \forall r = 1, \ldots, N_r$ should also be fixed, where N_r is the total number of rows in the image. Hence, starting from the top of the image with an initial value of $f_1 = 55$ pixels, the width of the filter on the rth row of the image will be $f_r = 2(f_1 + r - 1)$. This gradual increase of the width causes the filtered area to resemble a triangle. Hence we name this process the 'triangle filtering' process. An example of applying the triangle filtering approach can be seen in Figure 7.

C. LOCATING THE PLANTS IN THE IMAGE

The next step in the algorithm is to locate the pixels in the image that represents the crops, as shown in the total flow chart in Figure 3. Two approaches have been developed and applied to complete this step: the self-adaptive red-green-blue (RGB) filtering and the Canny edge detection algorithm.

1) ADAPTIVE RGB FILTERING

The adaptive RGB filtering algorithm is able to pick out the pixels representing the crops despite the change of crop size and brightness level. Each pixel in a color image taken by the camera is associated with a set of red, green, and blue values ranging from 0 to 255 according to its color. The set of pixels in an image outputted from triangle filter can be defined as

$$\mathbb{P} = \{\mathbf{P}_i\}, \quad \forall i = 1, 2, \ldots, |\mathbb{P}|, \quad (10)$$

where $\mathbf{P}_i$ is a vector with

$$\mathbf{P}_i = \left[P_{i,r}, P_{i,g}, P_{i,b}\right]$$
$$\text{with } P_{i,r}, P_{i,g}, P_{i,b} \in [0, 255], \quad \forall i = 1, 2, \ldots, |\mathbb{P}|. \quad (11)$$

To distinguish the crops from the dirt in the image, the system should find a green threshold value γ such that all the pixels in the image having green values $P_{i,g} > \gamma$ constitute the crops, and all the other pixels having $P_{i,g} < \gamma$ constitute dirt. Nevertheless, due to variations in brightness level, crop height, crop color and location of the rover associated with each picture, having one constant green threshold value γ for all the pictures cannot optimize the performance of the sensing subsystem. Therefore, an adaptive RGB filtering algorithm is proposed to accommodate these variations. The general flow-chart of the proposed adaptive algorithm is shown in Figure 8.

First, an initial low green threshold $\gamma_0 = 140$ is applied to the images to filter out all the pixels with $P_{i,g} < \gamma_0$. $\gamma_0 = 140$ was selected empirically by looking at individual crop pixels across all testing days having different brightness levels. γ_0 was what we observed to be slightly lower than the lowest green value for a plant pixel on the darkest day. Removing all of the pixels with green values lower than γ_0 will locate all of the crops in the images in most cases, as well as some other undesired objects, such as brighter spots in the soil. To exclusively locate the crops, this filtering process needs to be continuously repeated with an increasingly bigger γ value until the black and white (BW) mean value of the corresponding BW image is below the BW mean threshold. Then, all white pixels in the resulted BW image will be recognized as crop pixels.

Below we will introduce the method to determine the BW mean threshold of each picture. To facilitate our discussion, we denote the set containing all the pixels with a green value bigger than γ as

$$\mathbb{P}_{>\gamma} = \left\{\mathbf{P}_i : \mathbf{P}_i \in \mathbb{P} \bigcap P_{i,g} > \gamma\right\}$$
$$\forall i = 1, 2, \cdots |\mathbb{P}|. \quad (12)$$

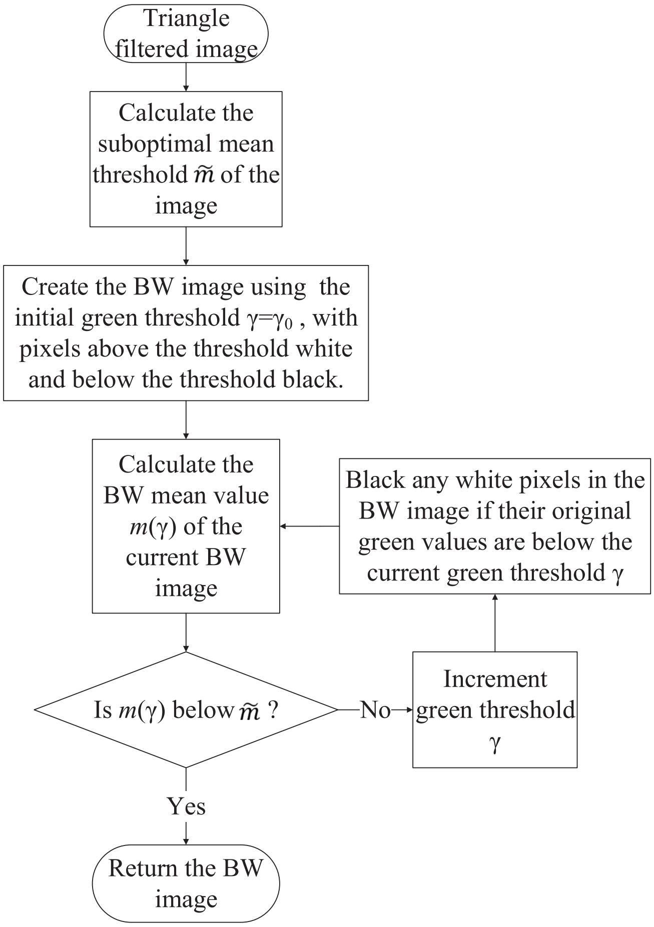

Triangle filtered image
Calculate the suboptimal mean threshold $\tilde{m}$ of the image
Create the BW image using the initial green threshold γ=γ₀ , with pixels above the threshold white and below the threshold black.
Calculate the BW mean value $m(\gamma)$ of the current BW image
Black any white pixels in the BW image if their original green values are below the current green threshold γ
Is $m(\gamma)$ below $\tilde{m}$?
No
Increment green threshold γ
Yes
Return the BW image

FIGURE 8. The flowchart of adaptive RGB filtering algorithm.

To further visualize the filtering output of applying $P_{i,g} > \gamma$ to the image $\mathbb{P}$, we turn the original picture $\mathbb{P}$ into a BW image $\mathbb{P}_{\mathrm{BW},\gamma}$, with all the pixels in set $\mathbb{P}_{>\gamma}$ marked in white and all the pixels in set $\mathbb{P}_{<\gamma}$ in black. More quantitatively,

$$\mathbb{P}_{\mathrm{BW},\gamma} = \mathbb{P}_{\mathrm{W},\gamma} \bigcup \mathbb{P}_{\mathrm{B},\gamma} \tag{13}$$

where

$$\begin{aligned} &\mathbb{P}_{\mathrm{W},\gamma} = \left\{P_{\mathrm{W},\gamma,i} = 255\right\} \quad && \forall i : \mathbf{P}_{i,g} \geq \gamma \bigcap \mathbf{P}_i \in \mathbb{P} \\ &\mathbb{P}_{\mathrm{B},\gamma} = \left\{P_{\mathrm{B},\gamma,i} = 0\right\} \quad && \forall i : \mathbf{P}_{i,g} < \gamma \bigcap \mathbf{P}_i \in \mathbb{P} \\ & && i = 1, 2, \ldots |\mathbb{P}| \end{aligned} \tag{14}$$

Figure 9 is an unprocessed raw image of the corn field. After applying distortion correction, triangle filtering and $P_{i,g} > \gamma$ filtering to Figure 9, the achieved BW image $\mathbb{P}_{\mathrm{BW},\gamma}$ is depicted in Figure 10. More exactly, all the pixels that have $P_{i,g} > \gamma$ are represented as white pixels in Figure 10, and the rest are in black.

The mean value of the BW image $\mathbb{P}_{\mathrm{BW},\gamma}$ is called the BW mean value and is denoted as m, and $m \in [0, 255]$. Ideally, every single pixel of the crop should be highlighted in white and all other pixels should be in black. In this case, the resulting BW mean value would be the optimal value for BW mean threshold of the image. More quantitatively, the BW mean value of a BW image achieved after applying filter $P_{i,g} > \gamma$ is defined as

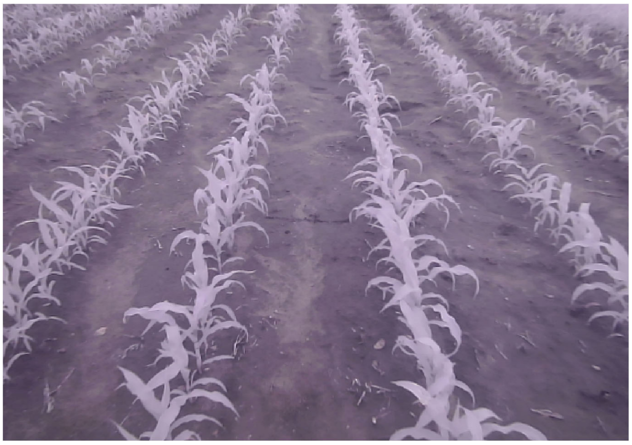


FIGURE 9. An unprocessed raw image taken in the field, as a reference for post-process pictures.

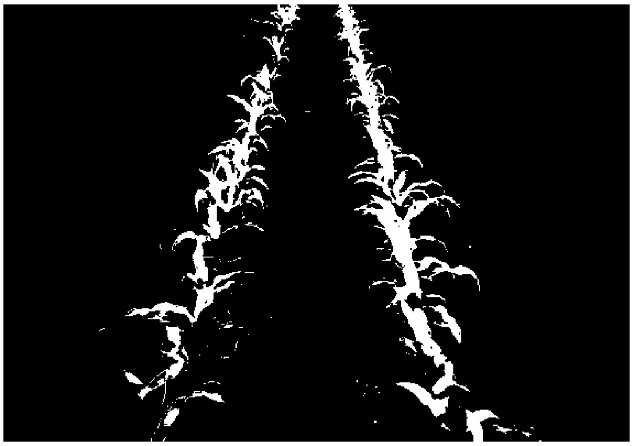


FIGURE 10. The image of the raw picture in Figure 9 after distortion correction, triangle filtering and RGB filtering.

$$m(\gamma) = \frac{\sum_{i=1}^{|\mathbb{P}|} P_{\mathrm{BW},\gamma,i}}{|\mathbb{P}_{\mathrm{BW},\gamma}|} \tag{15}$$

As the BW mean m can be any real number $\in [0, 255]$, exhaustively searching for the optimal value of it can be time consuming and therefore not considered in real-time robotic systems. We hereby propose a low-complexity algorithm to find the suboptimal value of the BW mean and set it as the mean threshold, which is denoted as $\tilde{m}$ in this treatise. The two major factors that affect the value of $\tilde{m}$ are the volume of the crops and the ambient brightness. With denoting the crop-volume factor as α and the brightness factor as β, we assume that the BW mean threshold of each picture can be represented by

$$\tilde{m} = \alpha \times \beta. \tag{16}$$

What we are trying to calculate with Equation (16) is the sub-optimal mean threshold, or in other words, the mean pixel value of the final BW image returned by the adaptive RGB filtering algorithm (see flowchart in Figure 8). One may notice that this mean value is essentially the number of crop pixels divided by the total number of pixels in the image. As the number of pixels in each image is the same, the sub-optimal mean threshold $\tilde{m}$ is directly proportional to the number of crop pixels.

A rough estimate of how many plant pixels there are in an image is calculated by counting how many pixels have a green value above a fixed constant γ_h. The high green threshold $\gamma_h = 160$ is selected empirically, such that $\mathbb{P}_{>\gamma_h}$ will never include dirt in any circumstance. Thus, given the same level of brightness, $|\mathbb{P}_{>\gamma_h}|$ should be proportional to the volume of the crops, or the number of pixels representing the crop rows. Thus, we define the crop volume factor as the number of pixels with green value higher than γ_h, i.e.

$$\alpha = |\mathbb{P}_{>\gamma_h}|. \tag{17}$$

Hereby, $|\mathbb{P}_{>\gamma_h}|$ is actually an indicator of the crop volume factor α, and $|\mathbb{P}_{>\gamma_h}|$ can be easily obtained with any given image. Notice, when the brightness level is the same, α is determined by the age of the crops since sprout.

However this is not a perfect solution to calculate the amount of crops in the image. On the brighter days, other things in the image such as rocks, fallen leaves, and miscellaneous debris are more likely to fall into this count of pixels with a green value. Similarly, on the pictures we took in the evenings, we found that some of the crop pixels didn't make it above this threshold. To compensate for this, we multiplied α by a variable β which is proportional to overall brightness of the image, as shown in Equation (16).

Similar to the indicator of the crop-volume factor α, we should also select an easily obtained property associated with each image $\mathbb{P}$ as the indicator of brightness factor β. Intrinsically, the average green value $\bar{g}$ of all the pixels in the undistorted color image $\mathbb{P}$ should be positively correlated to the brightness level given the same crop size. Here, we define the average green value of the undistorted image as

$$\bar{g} = \frac{\sum_{i=1}^{|\mathbb{P}|} P_{i,g}}{|\mathbb{P}|}, \tag{18}$$

and the brightness factor can then be written as a monotonic function of $\bar{g}$:

$$\beta = f(\bar{g}). \tag{19}$$

In order to derive the exact form of the function $f()$, non-linear regression is used on the test data obtained through training process as shown in Figure 17.

To train the system, we collected pictures on five different days spreading out the early growth season of the crops during morning, afternoon and after sunset. This way, we will have pictures varying both on crop heights and brightness level. On each test day, we took pictures with various preset cross-track and heading offsets. The test on each day normally lasts for 2 hours and contains 100∼150 pictures. It is reasonable to assume that the crop volume and the brightness level on the same testing day are relatively the same. Hence, to reduce the complexity of calculation, we assume the optimal value of BW mean threshold on the same day are the same, and we denote the BW mean threshold on the dth day as $\tilde{m}_d$. Correspondingly, we denote the average crop-volume factors on the dth testing day as $\bar{\alpha}_d$. Thus, Equation (16) can be modified as:

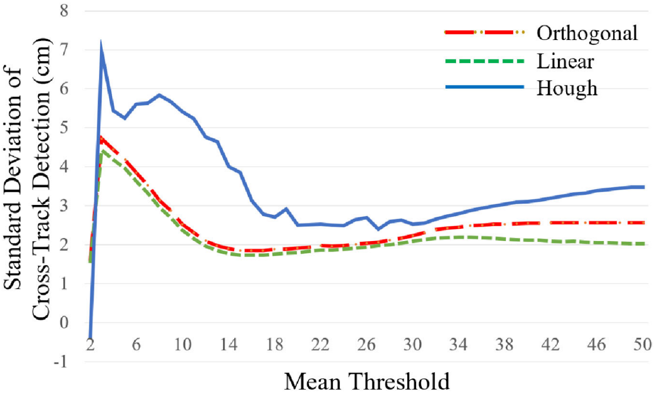


FIGURE 11. Standard deviations of the cross-track detection error on testing Day 2 at different mean threshold values.

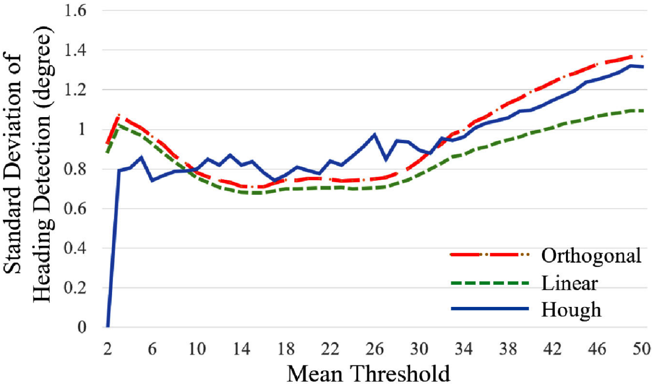


FIGURE 12. Standard deviations of the heading detection error on testing Day 2 at different mean threshold values.

$$\tilde{m}_d = \alpha_d \times \beta_d, \tag{20}$$

for training purpose.

We then search for the optimal value $\tilde{m}_d$ for BW mean threshold on each day with maximum likelihood methodology. More exactly, the standard deviation and average of the detection errors for cross-track and heading of every picture are calculated by using a large range of BW mean thresholds (1 to 50 with increments of 1). Figures 11∼14 show an example of these values on Day 2. These charts were then visually inspected to find the min, max, and optimal BW mean threshold candidates for each day, which can be seen in Figure 15. The mean threshold that results in the smallest mean and standard deviation of detection error for all six

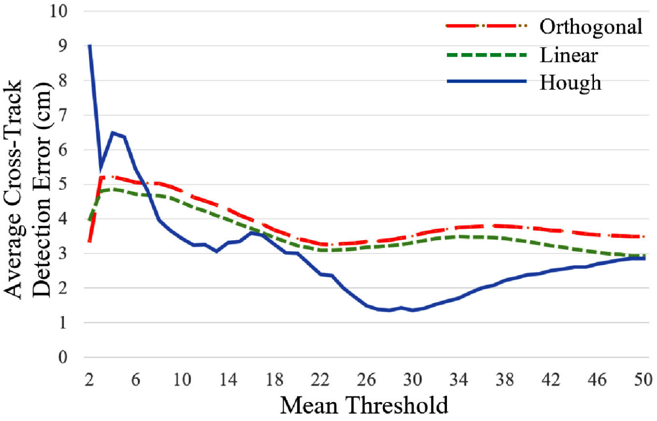


FIGURE 13. Average of absolute cross-track detection error on testing Day 2 at different mean threshold values.

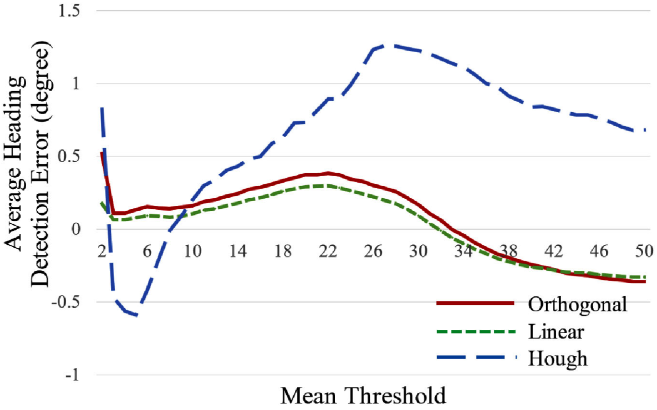


FIGURE 14. Average heading detection error on testing Day 2 at different mean threshold values.

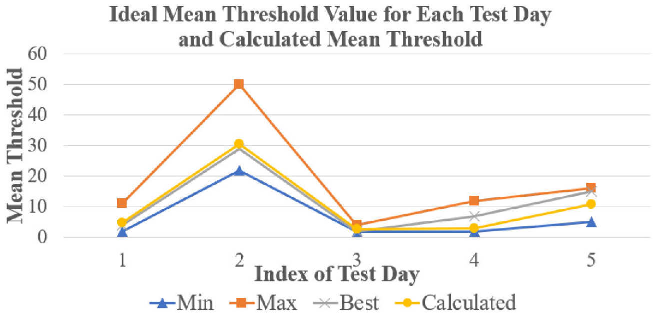


FIGURE 15. The minimum, maximum, optimal and calculated suboptimal mean threshold for all the pictures taken on each testing day.

possible detection algorithms is selected as the ultimate BW mean threshold on that day, while the min and max were the bounds on range we considered to give acceptable results. For reference, we chose the following values: Minimum = 22, maximum = 50, optimal = 29 for testing Day 2 according to Figures 11∼14.

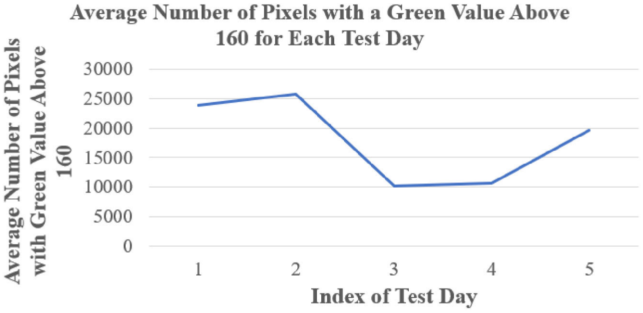


FIGURE 16. The average crop-volume factor $\bar{\alpha}_d$ on each testing day.

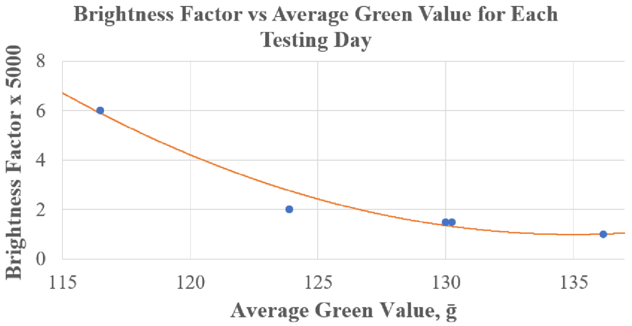


FIGURE 17. The nonlinear regression of brightness factor over average green value, using the calculated β_d and $\bar{g}_d$ on each testing day.

Thus $\bar{\alpha}_d$ can be calculated as

$$\bar{\alpha}_d = \frac{\sum_{i=1}^{N_p} |\mathbb{P}_{>\gamma_h,i}|}{N_p}, \tag{21}$$

where N_p is the total number of pictures taken on the dth testing day, and variable i refers to the index of the pictures taken on the same day. The average crop-volume factor $\bar{\alpha}_d$ on each testing day is plotted in Figure 16. Figures 15 and 16 show that the optimal mean threshold $\tilde{m}_d$ and $\bar{\alpha}_d$ on each day are positively correlated, yet not directly proportional. As predicted in Equation (16), the non-constant scaling factor between $\tilde{m}_d$ and $\bar{\alpha}_d$ is the brightness factor β_d. Next, to derive the formula for the brightness factor β, we calculate the theoretically expected β_d value on each day. According to Equation. (16), we should have

$$\beta_d = \tilde{m}_d / \alpha_d. \tag{22}$$

In order to derive the exact form of the function $\beta = f(\bar{g})$ as shown in Equation (19), the value of β_d and $\bar{g}_d$ are plotted in Figure 17, where $\bar{g}_d$ represents the average green value of all the undistorted images taken on the dth testing day.

The function of $\beta = f(\bar{g})$ is generated through nonlinear regression and recorded as

$$\beta = f(\bar{g}) = \frac{1}{5000}\left[\frac{(\bar{g}-135)^2}{70} + 1\right]. \tag{23}$$

One thing maybe worthy of note is that the brightness factor β is actually monotonically decreasing with the average green value $\bar{g}$. This is understandable, as given the same volume of crops, the same number of pixels should be selected to represent the crops, and hence the BW mean threshold $\tilde{m}$ should be the same regardless of the brightness level. Nevertheless, on a brighter day, with the whole image brighter, more pixels will have their green values above γ_h, resulting in a bigger α value. Hence, according to Eq. (16), to achieve the same BW mean threshold $\tilde{m}$ with a bigger α, the brightness factor β needs to be lower on a brighter day. As a conclusion, the brightness level and the brightness factor β are negatively correlated.

To make sure that an abnormally bright or dark picture wouldn't cause overly large or small $\tilde{m}$ values, boundaries are imposed on each side of the formula for β. The upper boundary for β is set at $\bar{g} = 110$, because $\bar{g} = 110$ is slightly darker than the darkest testing scenario after sunset. We want to keep the range close to the value range used for non-linear regression to avoid possible inaccuracies from extrapolation. The lower boundary of β is set to be $1/5000$ at $\bar{g} = 135$ for a couple of reasons. First, (135, 1/5000) is the vertex of the parabola shown in Figure 17. As stated before, the brightness factor function $\beta = f(\bar{g})$ needs to be monotonically decreasing. Hence, the upper bound of the value range of $\bar{g}$ cannot exceed the value of the independent variable at the vertex of the parabola. Second, we could have created another equation of β for $\bar{g} > 135$. Nevertheless, this will cause the β values to be less than $1/5000$ and further incur insufficient pixels to be selected as crops. In addition to the aforementioned boundaries set to β, we have also imposed minimum and maximum values allowed for $\tilde{m}$ to be 1 and 50 respectively. This is to further safeguard the system against selecting far too few or too many plant pixels.

All in all, the brightness factor β can be defined piecewise in terms of average green value $\bar{g}$ as:

$$\beta = \begin{cases} \dfrac{10}{5000}, & \text{if } \bar{g} \in (0, 110] \\ \dfrac{1}{5000}\left[\dfrac{(\bar{g}-135)^2}{70} + 1\right] & \text{if } \bar{g} \in (110, 135] \\ \dfrac{1}{5000}, & \text{if } \bar{g} \in (135, 255) \end{cases} \tag{24}$$

Based on Eq. (20) and (24), the calculated BW mean thresholds $\tilde{m}_d$ on each testing day are shown in Figure 15 as the yellow line with dot markers. As can be seen from the figure, the calculated BW mean threshold values are very close to their optimal counterparts. During the actual application of the adaptive RGB filtering to detect the crop pixels, the mean threshold $\tilde{m}$ of each picture will be calculated using Equation (16) with (17) and (24).

One thing worthy of note is that, equations in Sec. II-C1 are developed in the context of corn field. Nevertheless, the formats of all the equations and steps of the algorithms will maintain the same for different types of crops. However, the thresholds and constants used in various steps and equations, including γ_0, γ_h and the constants in Eq. (24), will need to be recalculated using the same methodologies for different types of crops through training process.

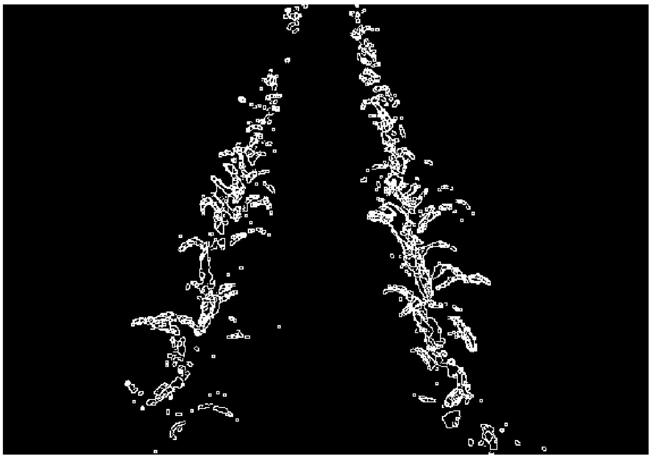


FIGURE 18. The result of applying triangle filtering and Canny edge detection on the image in Fig. 9.

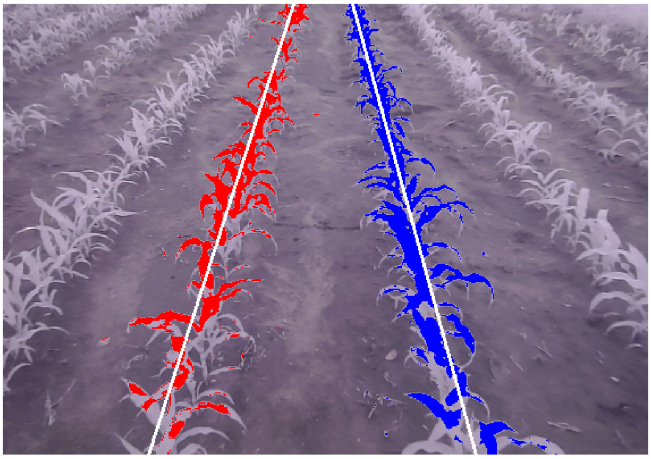


FIGURE 19. The resultant furrow edge lines after linear regression is applied to all the detected plant pixels in the image of Fig. 9. All plant pixels on the left side are highlighted in red and the right side in blue. The resultant furrow edge lines are marked in white.

2) CANNY EDGE DETECTION

A comprehensive description of the Canny edge detection algorithm was addressed in [52], which will not be repeated here. Fig. 18 shows the Canny edge detection result after applying triangle filtering on the input image shown in Fig. 9.

D. CALCULATING THE FURROW LINES

After all the plant pixels have been picked out, the system classifies the marked pixels into left and right sides based on each pixel's location in relation to the assumed centerline calculated earlier (see Equation (9) and Section II-B). As shown in Fig. 19 and 20, the left side and right side categories are colored in red and blue for display. Then the left and right

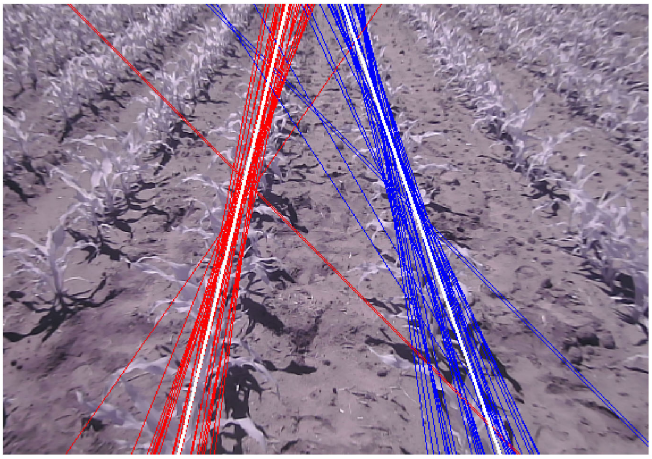


FIGURE 20. All the resultant Hough lines detected in the image of Fig. 9. Hough lines on the left side are highlighted in red and the right side in blue. The final furrow edge lines are marked in white.

furrow lines are obtained using either the regression method or Hough line detection.

1) INVERTED LINEAR REGRESSION

For regression approaches, the system defines each marked pixel as a point with (x, y) coordinates in a Cartesian plane having the left top corner of the image as the origin. Then either linear regression or orthogonal (Deming) regression is applied to get a line of best fit in the form of

$$y = mx + b \tag{25}$$

for both the left and right rows of crops.

The formulae to determine the slope m and intercept b for a conventional linear regression can be found using the least mean square error approach as [53]:

$$m = \frac{N\sum_{i=1}^{N}(x_i y_i) - \sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i}{N\sum_{i=1}^{N}(x_i^2) - (\sum_{i=1}^{N} x_i)^2}$$

$$b = \frac{\sum_{i=1}^{N} y_i - m\sum_{i=1}^{N} x_i}{N} \tag{26}$$

While the conventional linear regression only minimizes the mean square errors from the plant pixel points to the achieved furrow line in the vertical y direction, the result in our application is actually more sensitive to the errors in x direction. Therefore, the x and y coordinates for each plant pixel are swapped before calculating the slope and intercept, so that horizontal distance is taken into account rather than vertical. After using Equation (26), the x and y coordinates of the resultant line is inverted back.

2) DEMING REGRESSION

Another regression approach, namely the Deming (orthogonal) regression, minimizes the normal distances from each plant pixel to the achieved furrow edge line in Equation (25). Correspondingly, the parameters can be evaluated using the following equations [54]:

$$s_{xx} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}$$

$$s_{xy} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

$$s_{yy} = \frac{\sum_{i=1}^{N}(y_i - \bar{y})^2}{N-1}, \tag{27}$$

and the slope and intercept in Equation (25) can be obtained as:

$$m = \frac{s_{yy} - s_{xx} + \sqrt{(s_{yy} - s_{xx})^2 + 4s_{xy}^2}}{2s_{xy}}$$

$$b = \frac{\sum_{i=1}^{N} y_i - m\sum_{i=1}^{N} x_i}{N} \tag{28}$$

3) HOUGH LINE DETECTION

Hough line detection is a process which exhaustively draws lines on the image and then records all the lines colliding with at least a given number of marked pixels [52]. In the proceeding paper [52], after all of the Hough lines are gathered, any lines within 45 degrees of the horizontal are filtered out [52]. For each of the remaining lines, the intersection between the lines and the top and bottom of the image are both calculated. Next, a median of these coordinates is calculated for top and bottom, and the resulting line drawn between these two points is the final furrow line. This process is carried out independently for each side of the furrow based on the assumed centerline obtained earlier.

Nevertheless we had problems with far too many Hough lines being drawn, causing very inaccurate results. So we improve the Hough line detection algorithm by repeating the Hough line process in a loop, continuously increasing the number of points a line must connect until the number of lines is below a certain threshold $N_h = 70$. This threshold was found empirically by looking at the training data to get a general range we expected it to be in, and then testing values within that range to see what gave the best results. In addition to this, if there are less than four lines on one side, that side is ignored as it doesn't have enough data to produce an accurate result. Hence, the final calculation is done with only the furrow line from the other side.

E. ADAPTIVE ANTI-OVERINFLATION ADJUSTMENTS

Ideally, the calculated furrow edges should be lines connecting the base point of each crop stalk on the ground, such as the black lines in Figures 21 and 22. Obviously, the base lines of the crop stalks are not always the visual center of the crop rows. Nevertheless, all the approaches used to acquire the furrow edge lines discussed in Sections II-C2 and II-D calculate each furrow line by generating a line that is roughly the centroid of all the recognized plant pixels on one side. This causes the obtained furrow lines to be drawn on the leaves of the plants, rather than at the base of them, as delineated as

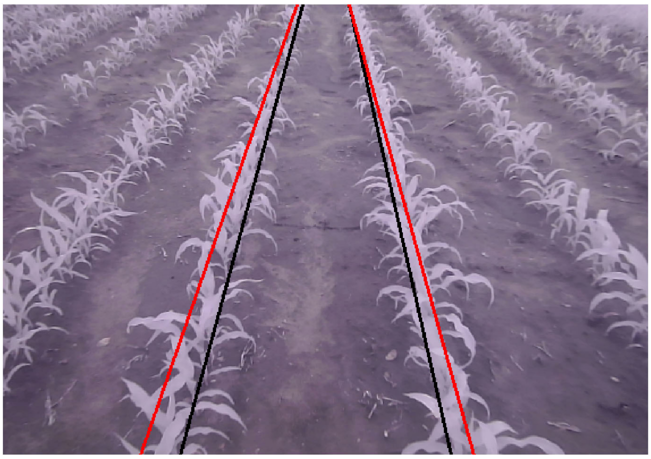


FIGURE 21. The black lines were drawn on this image manually. They show the ideal case of where the final furrow lines should be placed, at the base of the crops. The red lines show the actual output from the system, where the lines are drawn on top of the crops rather than the base.

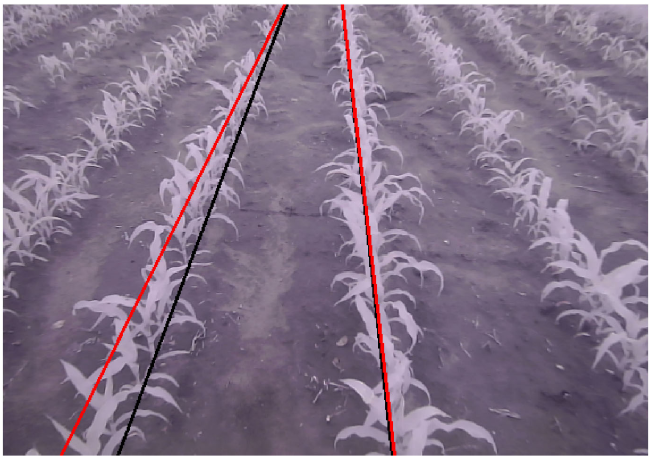


FIGURE 22. As in figure 21, the black lines show the ideal case and the red show actual output of the system. This image has a known cross-track value of 16 cm, causing the furrow line on the right to be much closer to ideal than the one on the left due to the change in angles from the large cross-track value.

red lines in Figures 21 and 22. Hence, there will always be a gap between the actual furrow edges and the calculated furrow edges. This gap can be remarkable, when the crop height is non-negligible.

On one hand, when the camera is located in the center of the furrow, i.e. when the cross-track error is close to 0, the distance between the actual furrow edges and the calculated furrow edges have the same absolute value but opposite signs on each side, causing them to cancel each other out. As a result, the furrow looks slightly wider then it actually is, but the detected cross-track error is still close to 0.

On the other hand, if the cross-track error of the rover gets larger, mistaking the lines on top of the crops as the true furrow edges can cause additional errors to the detection results of the cross-track locations. Figure 22 shows this scenario where the cross-track error is known to be positive 16 cm. That is, the camera is 16 cm to the right from the centerline of the furrow. The relatively big cross-track error on right side causes the calculated furrow line to appear much closer to the actual furrow edge on the right side, yet the distance between the calculated furrow line and the actual furrow edge on the left side is much bigger. Consequently, this leads the sensing subsystem to assume that the camera is much further away from the left side and the centerline of the furrow than it actually is, causing an overinflated cross-track error.

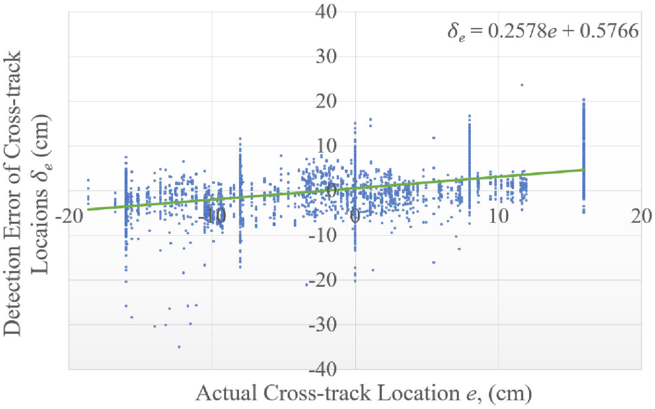


FIGURE 23. This chart shows the cross-track inaccuracy caused by the final furrow lines being drawn on top of the plants rather than at the base. The green line was created by linearly regressing all the blue data in the figure. The equation for the green line is included at the upper right corner of the chart.

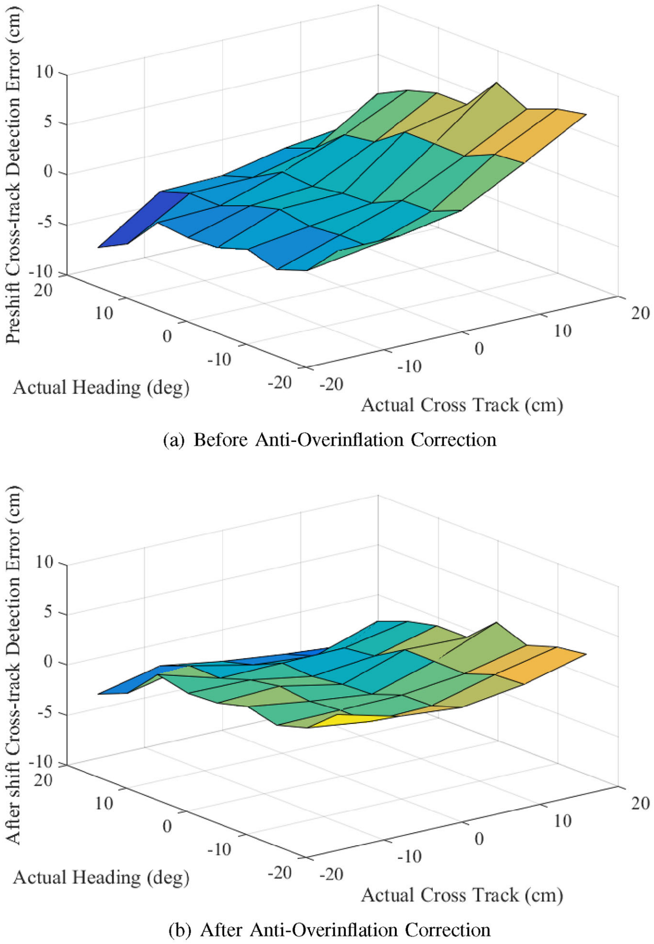
The same phenomenon can be evidenced from the collective results shown in Figures 23 and 24(a). After the 2D furrow edge lines shown in Figures 21 and 22 are converted to 3D lines with geographical coordinates on the ground, the detected cross-track error $\hat{e}$ and the heading error $\hat{\Phi}_e$ of the rover can be calculated using the approached introduced in [52]. The detection errors of the cross-track locations versus the actual cross-track locations of all the test samples are plotted in Figure 23. In order to better represent the massive number of test results, a line of best fit is drawn as a green line in the figure too. As can be seen in the figure, the bigger the preset actual cross-track distance from the center line, the bigger the detection error, which is supported by the positive slope of the line of best-fit of all the results in Figure 23. This indicates that the detection results of the cross-track distance is indeed 'overinflated' as analyzed theoretically above.

In order to account for this overinflation, an empirical equation is derived to quantify the line of best fit in Figure 23. According to the formula shown in Figure 23, we can then establish the equation to achieve the compensated cross-track error e_c from the detected cross-track error $\hat{e}$ as follows:

$$\because \delta_e = 0.2578e + 0.5766$$

$$\therefore \hat{e} = e + 0.2578e + 0.5766 = 1.2578e + 0.5766$$

$$\therefore e_c = \frac{\hat{e} - 0.5766}{1.2578}, \tag{29}$$



(a) Before Anti-Overinflation Correction

(b) After Anti-Overinflation Correction

FIGURE 24. A 3D surface showing the average cross-track detection errors across all testing days at different rover locations (cross-track and heading offsets) (a) before and (b) after anti-overinflation corrections.

where δ_e is the detection error of the cross-track location, and hence is defined as $\delta_e = \hat{e} - e$.

Applying Equation (29) to all the cross-track location detection results shown in Figure 24(a), we achieved the anti-overinflation adjusted cross-track detection results as shown in Figure 24(b). As can be seen in Figure 24(a), there is a distinct correlation between cross-track locations and cross-track detection errors. Nevertheless, after applying the anti-overinflation correction, the 3D surface is much less sloped, meaning the correlation between cross-track locations and cross-track detection errors has been significantly reduced. With a comparison between both figures, the proposed anti-overinflation process has largely reduced the cross-track detection errors from ranging between $-10 \sim +10$ cm to $-3 \sim +2$ cm.

III. LATE GROWTH SUBSYSTEM

As the crops get larger, the algorithms introduced for the early growth season in Section II become increasingly inaccurate and inefficient, mostly due to the issue caused by tall crop stalks as discussed in Section II-E. So once the crops are above 45 cm (a foot and a half), we consider them to be in the late growth stage, in which a LiDAR (Light Detection And Ranging) sensing system is used instead of the camera. As now the crops are tall enough to reflect the laser beams transmitted by the LiDAR sensor, producing a highly accurate point cloud for locating crops.

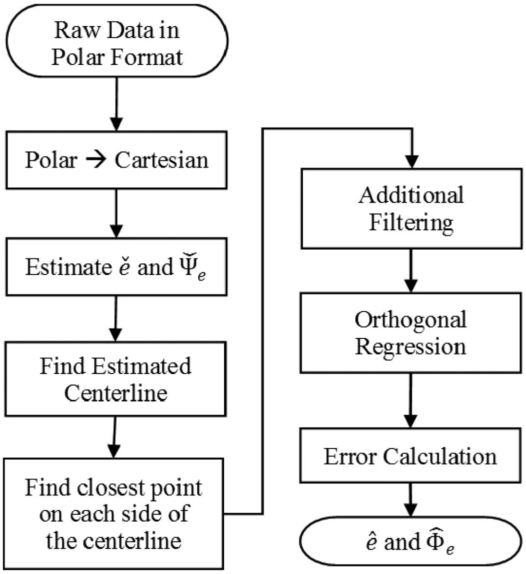

Raw Data in Polar Format
Polar → Cartesian
Estimate $\check{e}$ and $\breve{\Psi}_e$
Find Estimated Centerline
Find closest point on each side of the centerline
Additional Filtering
Orthogonal Regression
Error Calculation
$\hat{e}$ and $\widehat{\Phi}_e$

FIGURE 25. Flowchart of the row sensing subsystem algorithms to detect the cross-track error $\hat{e}$ and heading error $\hat{\Psi}_e$ during late-growth season.

The LiDAR sensor is mounted on the center of the rover, with zero degree facing directly forward. Each time the LiDAR scans, it measures a distance to the nearest obstacle every quarter of a degree for its 270 degree field of view, resulting in 1080 distances with its angle index written to a csv file. This csv file is then processed by our algorithm in C to determine the rover's heading and cross-track locations. The flow chart of the late growth error-detection algorithm is shown in Figure 25. One thing worthy of note is that unlike the process for early growth season as shown in Figure 3, there is no 2D to 3D coordinate conversion process in the location detection algorithm for late growth season. As the raw data obtained with LiDAR during late growth season naturally comprise the geographical distances and angles of any detected object. After the raw data are read in, they are then converted to Cartesian format as shown in Figures 26 and 27.

Ideally, we want all the data points to represent the crop stalks on both sides of the furrow of interest traversed by the rover. Thus, the data points can be used to define the furrow edges and further obtain the true centerline of the furrow for calculation of cross-track and heading errors of the rover. Nevertheless, as can be seen in Figure 26 and 27, some data points are not from the furrow of interest, but the crops or objects outside it. This is because some laser beams emanated from the LiDAR travel through the gaps in the leaves of the desired

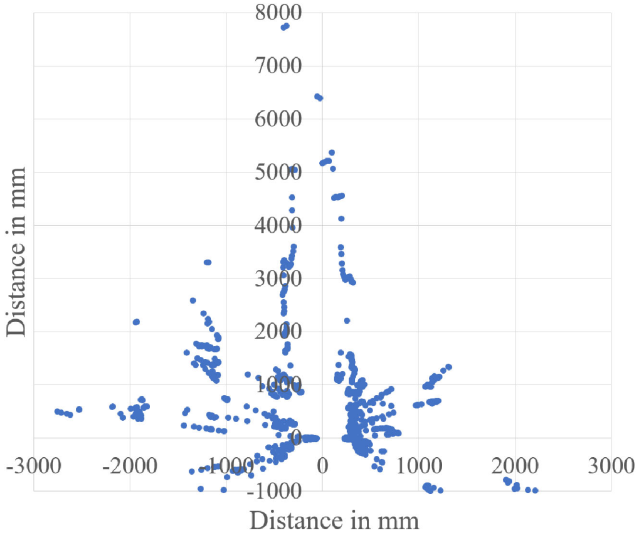


FIGURE 26. An example of the raw data collected by the LiDAR sensor represented in Cartesian format. This set of data has a known cross-track error of 0 cm and a heading error of 0 degree.

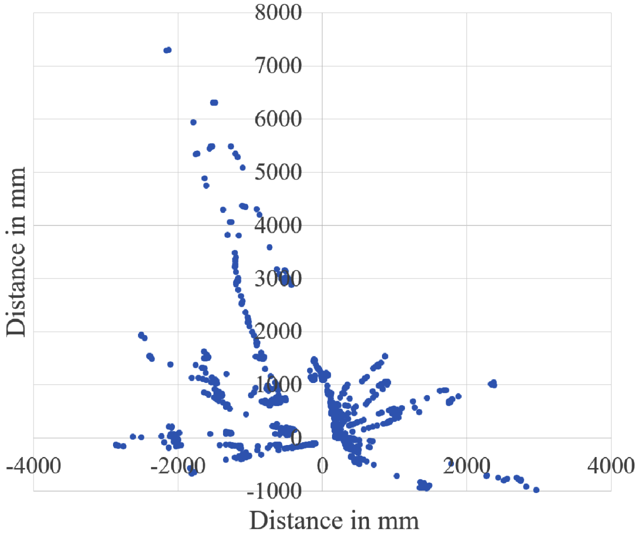


FIGURE 27. Another example of the raw data collected by the LiDAR sensor represented in Cartesian format. This set of data has a known cross-track error of 8 cm and a heading error of 15 degrees.

crop rows and are reflected back from the objects behind them. These undesired data points need to be filtered out to avoid interfering in the process of determining the centerline of the interested furrow.

To ferret out the desired data points from the whole set, the first step is to calculate the estimated centerline based on the estimated cross-track and heading errors $\check{e}_k$ and $\check{\Psi}_{e,k}$. The process of estimating the *a priori* cross-track and heading errors is the same as used in early growth season, which were detailed in Section II-B from Equation (4) to (8). The equation of the estimated centerline during late-growth season can then

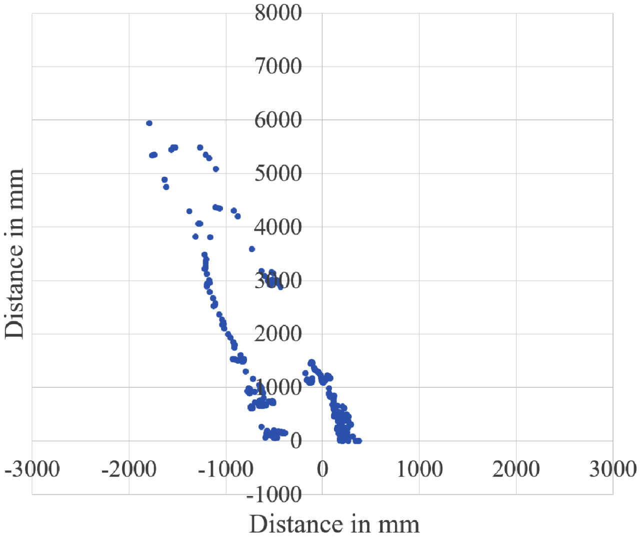


FIGURE 28. The kept data points after the closest points on each side of the estimated centerline are found and the additional filtering process is applied to the data in Figure 27.

be quantified as:

$$x = m \times y + b \tag{30}$$

where:

$$b = -\check{e}_k \text{ and } m = -\tan\left(\check{\Psi}_{e,k}\right) \tag{31}$$

The raw Cartesian data are then broken up into discrete sets based on their y value, where each set contains all the points having their y values within a 1 mm range. For example, the first set contains all points with y values in [0,1) mm and the next set would contain all points with y values in [1,2) mm. This continues until $y = 6$ m, resulting in 6000 sets of points. For each set, the algorithm finds the closest points from the estimated centerline in the horizontal (x) direction on both sides. If either or both of the points are within 450 mm of the estimated centerline in the x direction, they are kept as desired data points, with all other points in the set thrown out. The value of the threshold, i.e. 450 mm, is an empirical value determined by the width of the furrows, which is 30 inches (762 mm) here. The threshold of 450 mm is set so that if a data point is further than 450 mm away from the centerline, it is likely not from the crops of interest, but something behind them, such as the crops in another row. The value of this threshold should be readjusted when the algorithm is applied to a crop field having different furrow width. An example of the retained data points can be seen in Figure 28. After this additional filtering process is completed, a line of best fit is created on each side of the estimated centerline using orthogonal regression. Lastly, the slope and x-intercept of each furrow edge is utilized to calculate the final heading and cross-track values.

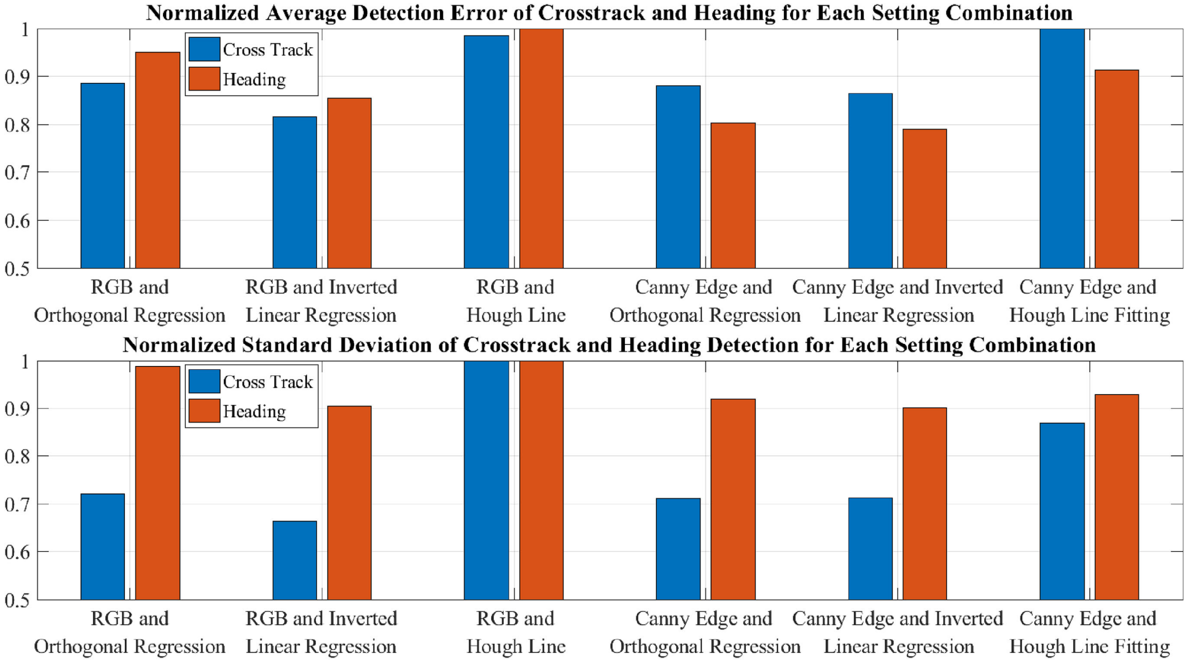


FIGURE 29. Please note values have been normalized for easier comparison and display. A vertical value of 1 corresponds to the following values: average *absolute* cross-track detection error is 2.73 cm, average *absolute* heading detection error is 0.86°, standard deviation of cross-track detection error is 3.73 cm, and standard deviation of heading detection error is 0.92°.

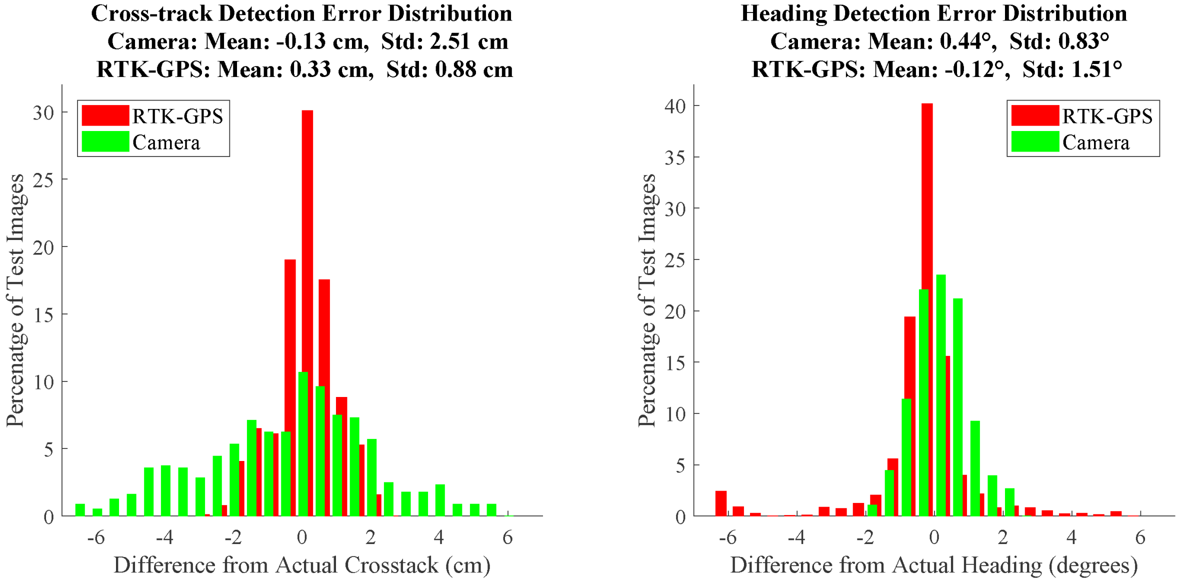
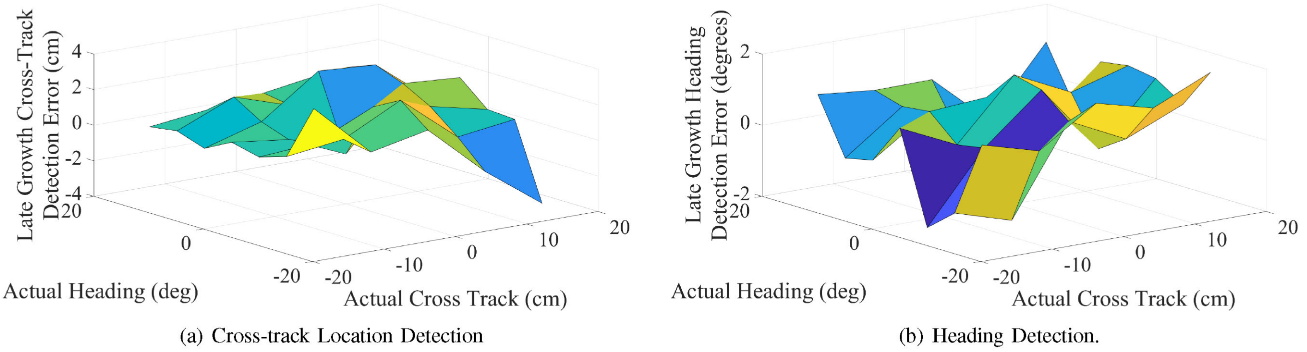


FIGURE 30. Comparison of histograms of the cross-track and heading detection errors, when a camera and an RTK-GPS are respectively used as the sensor during the early growth season. When the camera is used, the adaptive RGB filtering and inverted linear regression are jointly adopted as parts of the computer vision algorithm.

IV. RESULTS

In this section, Figures 29 and 30 present the experimental results during the early growth season, while Figures 31 and 32 present the results during the late growth season. Moreover, Figure 33 displays the closed-loop route taken by the rover given an offtrack initial position, when the navigation algorithm introduced in [5] and the computer vision algorithms introduced in this paper are jointly applied. In our experiment, a camera with a part number ELP-USB30W04MT-RL36, a LiDAR with a part number UST-10LX and a GPS with a part number Emlid Reach M+ were used to generate the results.



(a) Cross-track Location Detection (b) Heading Detection.

FIGURE 31. A 3D surface showing the (a) cross-track and (b) heading detection errors over all possible rover locations during late growth season.

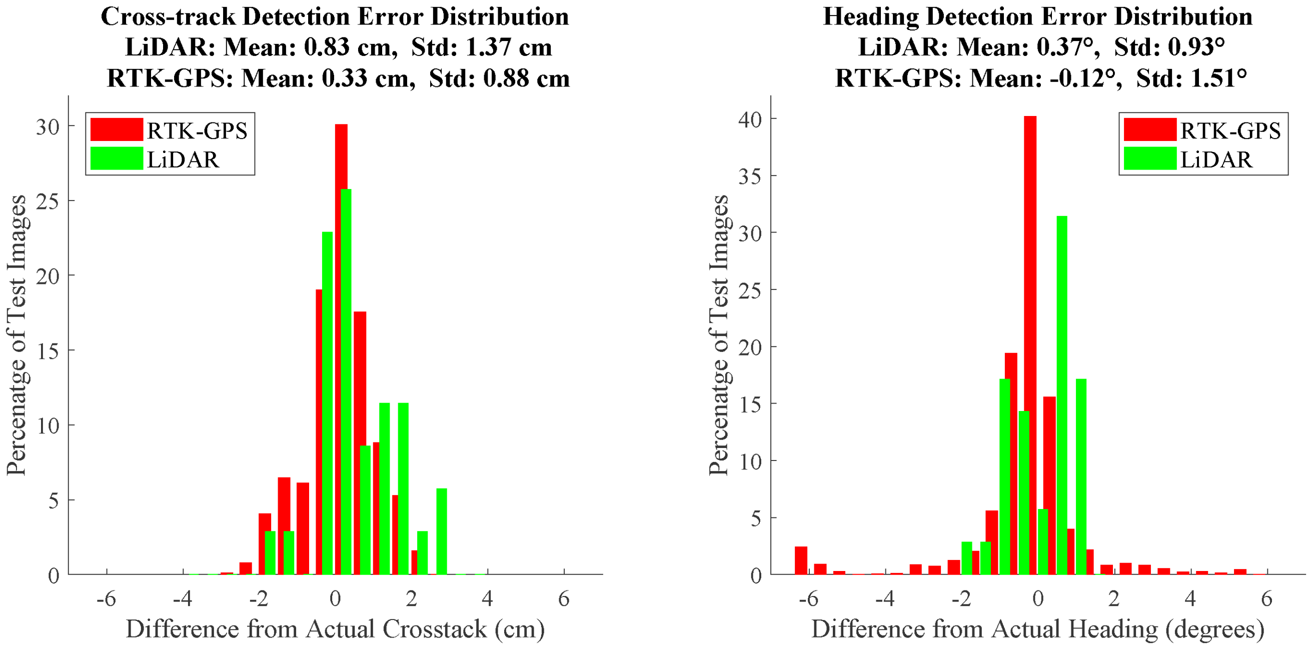


FIGURE 32. Comparison of histograms of the cross-track and heading detection errors when a LiDAR and an RTK-GPS are respectively used as the sensor during late growth season.

A. EARLY GROWTH SEASON

According to the flowchart in Figure 3, for early growth season, there are 2 × 3 possible ways to implement the computer-vision based algorithm to calculate the cross-track and heading errors, with choices between 'RGB filtering' and 'Canny edge detection,' and choices among 'inverted linear regression,' 'orthogonal regression' and 'Hough line detection'.

In order to know which combination will achieve the best results, all six combinations were implemented to process 575 raw pictures we collected at the Molitor Bros Farm in Minnesota. These pictures were taken at different times on different days with different weather and light conditions throughout corn's early growth season. The normalized average detection error and standard deviation of each combination is presented in Figure 29.

The results indicate that regression was generally more accurate than Hough line fitting. This is due to the fact that regression does a better job when one side of the furrow is brighter that the other. In this scenario, more pixels will be marked as plants on one side, causing almost all of the Hough lines to be drawn on that side, with only a few lines left on the other side. On the other hand, the creation of the line of best fit by regression is not influenced by the brightness difference between two sides.

Also, both forms of regression showed similar results, but the inverted linear regression turned out to be slightly more accurate than orthogonal regression. The inverted linear regression approach only tries to minimize the least mean square errors from the resultant furrow edge lines to the scattered pixel points in the horizontal direction (see Section II-D). Due to the narrow width of the furrow, all the possible headings a rover can have without damaging the crops are limited in $(-25, 25)$ degrees. The narrow heading range causes all furrow edge lines to span from bottom to top of the picture

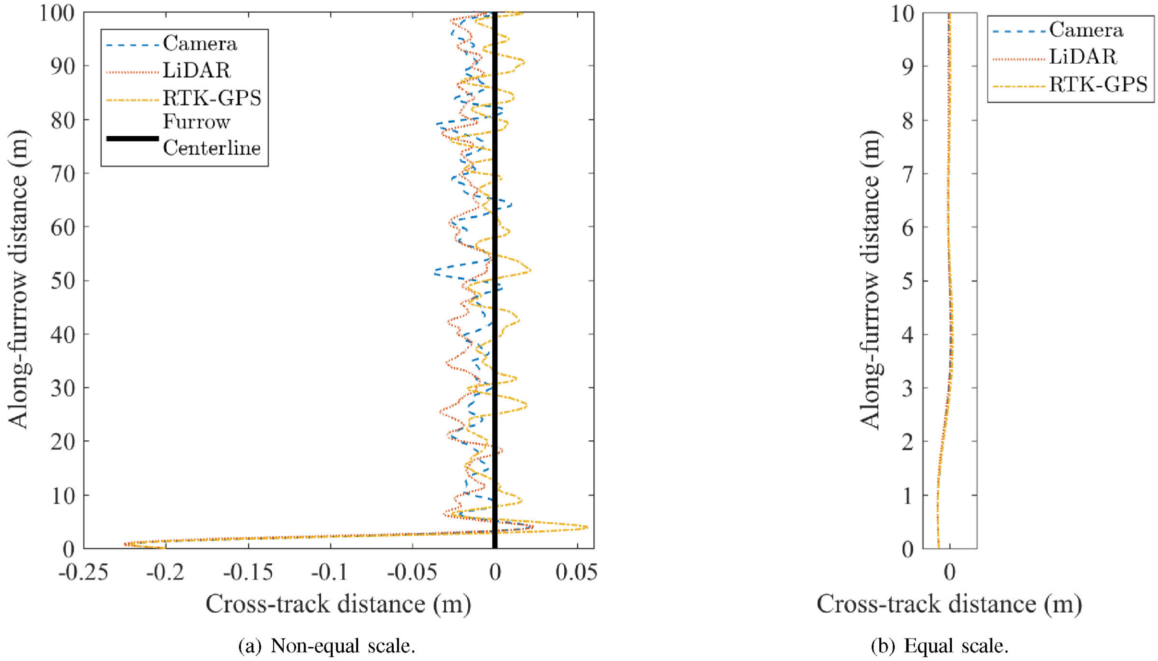


FIGURE 33. Top view of the paths of the rover in the furrow when a low-cost camera, a LiDAR and an RTK-GPS are used as the sensor respectively. The initial position of the rover is a cross-track error of 20 cm to the left of and a heading of 2° counter-clockwise from the centerline. Parameters of the dynamic-inversion based navigation algorithm are as follows: the damping ratio $\zeta = 0.8$ and the decay length $Y_0 = 1.5$m. Please refer to [5] for more details of the navigation algorithm. The horizontal and vertical axes in (a) do not have equal scales, while those in (b) have equal scales.

in vertical direction, while only taking up a short span in horizontal direction, as can be evidenced in Figures 19∼22. Therefore, minimizing the least mean square error in the horizontal direction is more important to final detection results of cross-track and heading locations, as compared to a normal distance between the scattered plant pixel points to the achieved furrow edge lines.

As can be seen in Figure 29, the adaptive RGB filtering and inverted linear regression produced the best results, so we have included a histogram of the distribution of cross-track and heading detection errors achieved by RGB filtering and inverted linear regression in Figure 30.

One thing worthy to note is that the camera used to take all the pictures had a problem causing the images to appear grey and monotone. We recognized this after the first few tests, but by the time the manufacturer got back to us with a replacement the early growth season was over. With a largely reduced color gradient, the proposed algorithms still provide the sensing subsystem with a detection accuracy close to GPS. It is hence safe to say that the accuracy will be enhanced, if a better camera is used.

B. LATE GROWTH SEASON

It can be seen from the late-growth results in Figure 31 that the cross-track detection error is bounded within -2 cm to $+2$ cm for all the preset locations of the rover, except a very slanted position at $+16$ cm and -20°, while the heading detection error is bounded within -2° to $+1.5^\circ$. An improvement in standard deviation of cross-track values compared to early growth is seen from 2.53 cm to 1.377 cm, while standard deviation of heading offset increases slightly from 0.83° to 1.2056°. Average absolute cross-track detection error goes from 1.998 cm to 1.206 cm, while average absolute heading error goes from 0.7379° to 0.8631° compared to early growth. The system is also much simpler and more immune to the interference from ambient light and crop heights.

C. OPEN AND CLOSED LOOP PERFORMANCE COMPARISONS

The open-loop performance of the proposed computer vision algorithms are compared with RTK-GPS during both the early and late growth seasons in Figures 30 and 32, while the RGB filtering and inverted linear regression are jointly used for early growth detection.

From both figures we can see that the accuracy of the RTK-GPS is slightly better than both the low-cost camera ($50) and the LiDAR when the proposed computer vision algorithm is used to detect the cross-track locations, as evidenced by a smaller standard deviation achieved by the RTK-GPS. However, using standard deviation as the metric, our computer vision algorithms outperform the RTK-GPS in terms of heading detection.

The detection accuracy of the RTK-GPS presented in Figures 30 and 32 is only achievable during ideal conditions, when constant high-quality satellite connection and accurate geographical survey of all the crop stalks in the field are available. Unfortunately, these two assumptions are challenging to meet in current agriculture environments, and hence the accuracy of RTK-GPS promised in Figures 30 and 32 are hard to achieve. On the contrary, the proposed computer vision algorithms are not dependent on satellite signals or onerous geographical survey. Therefore the accuracy of the proposed computer vision algorithms as presented in Figures 30 and 32 is easier to achieve and for a longer period of time.

Moreover, the gap of the detection accuracies between the RTK-GPS and the proposed computer vision algorithms observed in Figures 30 and 32 is diminished in the closed-loop simulation results shown in Figure 33. It may be worthy of note that the horizontal and vertical axis of Figure 33(a) are not equally scaled. The horizontal scale is much bigger than the vertical scale, hence the horizontal fluctuation of each path and the differences among the three paths are exaggerated in Figure 33(a). If the same scales are applied (see Figure 33(b)), the differences become negligible and all three paths are straightly aligned with the centerline of the furrow after a small vibration. This proves that the proposed computer vision algorithms, when utilized with a low-cost 2D camera or a LiDAR during early or late growth season, are able to provide the rover with the same route as the RTK-GPS, yet at a much lower realization complexity.

V. CONCLUSION

This paper continues the work published in [52] and introduces advanced computer vision algorithms to detect the location of the rover when it is travelling inside a furrow. The proposed solution package utilizes a low-cost camera and a LiDAR sensor during early and late growth season respectively. Each sensor works with its own set of computer vision algorithms. The proposed computer vision algorithms for early-growth season are able to adaptively adjust the threshold to segment the crops in the picture according to the ambient light intensity and crop size at the time when the picture is taken. Six combinations of crop recognition and furrow edge formulation algorithms are proposed and applied to process the pictures taken at Molitor Bros Farm for optimal selection. The combination of the adaptive RGB filtering and inverted linear regression turns out to provide the most accurate detection results. The proposed computer vision algorithm for the late-growth season is low-complexity and independent of environmental changes too.

The in-field tests at Molitor Bros Farm have shown that the proposed computer vision algorithms are able to provide a standard deviation of 2.51 cm and 0.83° for cross-track and heading detection during early growth season, and a standard deviation of 1.37 cm and 0.93° during late growth season. These are very close to the accuracy achieved by the RTK-GPS under ideal conditions, with its standard deviations of 0.88 cm and 1.51°. The closed-loop simulation further proves that there is no noteworthy difference in the path made by the rover in the furrow, no matter whether the proposed sensing systems or the RTK-GPS is employed. The advantage of the proposed computer vision algorithms over RTK-GPS is their reliability and facility. That is, while the RTK accuracy will degrade significantly during reduced GPS coverage and requires onerous advance geographical survey, the proposed computer vision algorithms are expected to achieve the above-mentioned accuracy over all kinds of weather conditions without extra work.

ACKNOWLEDGMENT

The authors would like to express their deep gratitude to Mr. Brian Molitor and Molitor Brothers Farm (Cannon Falls, Minnesota) for their enormous support to this project. Special thanks also go to Mr. Scott Morgan and Rx Robotics (Redwing, Minnesota) for their consistent financial support.

REFERENCES

[1] A. B. McBratney, B. Whelan, and T. Ancev, “Future directions of precision agriculture,” *Precis. Agriculture*, vol. 6, pp. 7–23, 2005.

[2] A. B. McBratney and M. J. Pringle, “Estimating average and proportional variograms of soil properties and their potential use in precision agriculture,” *Precis. Agriculture*, vol. 1, no. 2, pp. 125–152, Sep. 1999.

[3] J. A. Howard and C. W. Mitchell, *Phytogeomorphology*, Hoboken, NJ, USA: Wiley, 1985.

[4] T. C. Kaspar, T. S. Colvin, B. Jaynes, D. L. Karlen, D. E. James, and D. W Meek, “Relationship between six years of corn yields and terrain attributes,” *Precis. Agriculture*, vol. 4, pp. 87–101, 2003.

[5] J. Hammer and C. Xu, “Dynamic inversion based navigation algorithm for furrow-traversing autonomous crop rover,” in *Proc. IEEE Conf. Control Technol. Appl.*, Aug. 19–21, 2019, pp. 53–58.

[6] G. N. DeSouza and A. C. Kak, “Vision for mobile robot navigation: A survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 237–267, Feb. 2002.

[7] D. A. Pomerleau, “ALVINN: An autonomous land vehicle in a neural network,” Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU-CS-89–107, 1989.

[8] C. Thorpe, T. Kanade, and S. A. Shafer, “Vision and navigation for the Carnegie-Mellon Navlab,” in *Proc. Image Understand Workshop*, 1987, pp. 143–152.

[9] C. Thorpe, M. H. Herbert, T. Kanade, and S. A. Shafer, “Vision and navigation for the Carnegie-Mellon Navlab,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 3, pp. 362–372, May 1988.

[10] T. M. Jochem, D. A. Pomerleau, and C. E. Thorpe, “Vision-based neural network road and intersection detection and traversal,” in *Proc. IEEE Conf. Intell. Robots Syst.*, Aug. 1995, vol. 3, pp. 344–349.

[11] D. A. Pomerleau, “Efficient training of artificial neural networks for autonomous navigation,” *Neural Comput.*, vol. 3, pp. 88–97, 1991.

[12] M. A. Turk, D. G. Morgenthaler, K. D. Gremban, and M. Marra, “VITS–A vision system for autonomous land vehicle navigation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 3, pp. 342–361, May 1988.

[13] E. D. Dickmanns and A. Zapp, “A curvature-based scheme for improving road vehicle guidance by computer vision,” in *Proc. SPIE–Int. Soc. Photo-Opt. Instrum. Engineers*, Oct. 1986, vol. 727, pp. 161–168.

[14] E. D. Dickmanns and A. Zapp, “Autonomous high speed road vehicle guidance by computer vision,” in *Proc. 10th World Congr. Autom. Control*, 1987, vol. 1, pp. 221–226.

[15] E. D. Dickmanns, “Computer vision and highway automation,” *Veh. Syst. Dyn.*, vol. 31, no. 5, pp. 325–343, 1999.

[16] E. D. Dickmanns and B. Mysliwetz, “Recursive 3D road and relative egostate recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 199–213, Feb. 1992.

[17] V. Graefe and K. Kuhnert, “Vision-based autonomous road vehicles,” in *Vision-Based Veh. Guidance*, I. Masaki, Ed., Berlin, Germany: Springer-Verlag, 1992, pp. 1–29.

[18] U. Regensburger and V. Graefe, “Visual recognition of obstacles on roads,” in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 1994, pp. 980–987.

[19] A. M. Waxman *et al.*, “A visual navigation system for autonomous land vehicles,” *IEEE Trans. Robot. Autom.*, vol. 3, no. 2, pp. 124–141, Apr. 1987.

[20] S. K. Kenue, "LANELOK: Detection of lane boundaries and vehicle tracking using image-processing techniques–Parts I and II: Template matching algorithms," in *Proc. SPIE*, Mobile Robots IV, vol. 1195, Mar. 1990, doi: 10.1117/12.969886.

[21] C. C. Slama, *Manual of Photogrammetry*. 4th ed., Falls Church, Virginia: American Society of Photogrammetry, 1980.

[22] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 1997, pp. 1106–1112.

[23] P. van Walree, "Distortion," Photographic optics, Jan. 2009. [Online]. Available: https://web.archive.org/web/20090129134205/toothwalker.org/optics/distortion.html. Accessed on: Feb. 2, 2009.

[24] J. P. de Villiers, F. W. Leuschner, and R. Geldenhuys, "Centi-pixel accurate real-time inverse distortion correction," in *Proc. SPIE Optomechatronic Technol.*, Nov. 17, 2008, Paper 726611, doi: 10.1117/12.804771.

[25] D. C. Brown, "Decentering distortion of lenses," *Photogrammetric Eng.*, vol. 32, no. 3, pp. 444–462, May 1966.

[26] A. E. Conrady, "Decentred lens-systems," *Monthly Notices Roy. Astronomical Soc.*, vol. 79, pp. 384–390, 1919.

[27] Q. Li, L. Chen, M. Li, S.-L. Shaw, and A. Nchter, "A sensor-fusion drivable-region andlane-detection system for autonomous vehicle navigation in challenging road scenarios," *IEEE Trans. Veh. Technol.*, vol. 63, no. 2, pp. 540–555, Feb. 2014.

[28] Y. Wang, E. Teoh, and D. Shen, "Lane detection and tracking using B-Snake," *Image Vis. Comput.*, vol. 22, no. 4, pp. 269–280, Apr. 2004.

[29] D. Liu, B. Sheng, and F. Hou, "From wireless positioning to mobile positioning: An overview of recent advances," *IEEE Syst. J.*, vol. 8, no. 4, pp. 1249–1259, Dec. 2014.

[30] K. Chiang, H. Chang, C.-Y. Li, and Y.-W. Huang, "An artificial neuralnetwork embedded position and orientation determination algorithm forlow cost MEMS INS/GPS integrated sensors," *Sensors*, vol. 9, no. 4, pp. 2586–2610, Apr. 2009.

[31] V. Indelmanpini, G. Rivlin, and H. Rotstein, "Real-time vision-aided localization and navigation based on three-view geometry," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 3, pp. 2239–2259, Jul. 2012.

[32] T. B. Karamat, R. G. Lins, S. N. Givigi, and A. Noureldin, "Novel EKF-based vision/inertial system integration for improved navigation," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 1, pp. 116–125, Jan. 2018.

[33] E. Royer, M. Lhuillier, M. Dhome, and T. Chateau, "Towards an alternative GPS sensor in dense urban environment from visual memory," in *Proc. Brit. Mach. Vis. Conf.*, 2004, pp. 197–206.

[34] S. Segvic, A. Remazeilles, A. Diosi, and F. Chaumette, "A mappingand localization framework for scalable appearance-based navigation," *Comput. Vis. Image Understanding*, vol. 113, no. 2, pp. 172–187, Feb. 2009.

[35] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM:Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 6, pp. 1052–1067, Jun. 2007.

[36] A. Vu, A. Ramanandan, A. Chen, J. Farrell, and M. Barth, "Real-time computer vision/DGPS-aided inertial navigation system for lane-level vehicle navigation," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 899–913, Jun. 2012.

[37] J. Reid and S. Searcy, "Vision-based guidance of an agriculture tractor," *IEEE Control Syst. Mag.*, vol. 7, no. 2, pp. 39–43, Apr. 1987.

[38] J. Billingsley and M. Schoenfisch, "The successful development of a vision guidance system for agriculture," *Comput. Electron. Agriculture*, vol. 16, no. 2, pp. 147–163, 1997.

[39] N. D. Tillett, T. Hague, and S. J. Miles, "Inter-row vision guidance for mechanical weed control in sugar beet," *Comput. Electron. Agriculture*, vol. 33, no. 3, pp. 163–177, Mar. 2002.

[40] H. T. Søgaard and H. J. Olsen, "Determination of crop rows by image analysis without segmentation," *Comput. Electron. Agriculture*, vol. 38, no. 2, pp. 141–158, Feb. 2003.

[41] A. English, P. Ross, D. Ball, and P. Corke,"Vision based guidance for robot navigation in agriculture," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 1693–1698.

[42] B. Åstrand and A.-J. Baerveldt, "A vision based row-following system for agricultural field machinery," *Mechatronics*, vol. 15, no. 2, pp. 251–269, Mar. 2005.

[43] H. J. Olsen, "Determination of row position in small-grain crops by analysis of video images," *Comput. Electron. Agriculture*, vol. 12, no. 2, pp. 147–162, Mar. 1995.

[44] D. C. Slaughter, P. Chen, and R. G. Curley, "Vision guided precision cultivation," *Precision Agriculture*, vol. 1, no. 2, pp. 199–216, 1999.

[45] J. A. Marchant, "Tracking of row structure in three crops using image analysis," *Comput. Electron. Agriculture*, vol. 15, no. 2, pp. 161–179, Jul. 1996.

[46] T. Bakker *et al.*, "A vision based row detection system for sugar beet," *Comput. Electron. Agriculture*, vol. 60, no. 1, pp. 87–95, Jan. 2008.

[47] V. Leemans and M.-F. Destain, "Line cluster detection using a variant of the Hough transform for culture row localisation," *Image Vis. Comput.*, vol. 24, no. 5, pp. 541–550, 2006.

[48] J. A. Marchant, and R. Brivot, "Real-time tracking of plant rows using a hough transform," *Real-Time Image*, vol. 1, no. 5, pp. 363–371, Nov. 1995.

[49] M. Kise, Q. Zhang, and F. R. Más, "A stereovision-based crop row detection method for tractor-automated guidance," *Biosystems Eng.*, vol. 90, no. 4, pp. 357–367, 2005.

[50] A. Pérez, F. López, J. Benlloch, and S. Christensen, "Colour and shape analysis techniques for weed detection in cereal fields," *Comput. Electron. Agriculture*, vol. 25, no. 3, pp. 197–212, 2000.

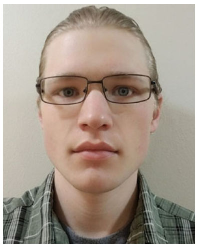
[51] W. Winterhalter, F. V. Fleckenstein, C. Dornhege, and W. Burgard, "Crop row detection on tiny plants with the pattern Hough transform," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3394–3401, Oct. 2018.

[52] D. Langan, R. Vraa, and C. Xu, "Machine-vision based location detection solutions for autonomous horticulture rover during early growth season," in *Proc. IEEE Int. Conf. Intell. Saf. Robot.*, Aug. 24–27, 2018, pp. 429–434.

[53] D. A. Freedman, *Statistical Models: Theory and Practice*, Cambridge, U.K.: Cambridge Univ. Press, 2009.

[54] W. E. Deming, *Statistical Adjustment of Data*. Hoboken, NJ, USA: Wiley, 1985.

SAMUEL J. LEVOIR received the B.S. degree in computer engineering from the University of St. Thomas in St. Paul, Minnesota USA in 2019. During his time at St. Thomas, his research focused on computer vision algorithms and machine learning techniques. These studies were applied to the project written about in this paper of navigating a rover through a crop field as well as a laser scanning system which created 3D models of braille documents. He also researched the development of efficient power usage algorithms in low voltage applications with a focus on dynamic voltage and frequency scaling. After graduating from St. Thomas, he has been employed at Open Systems International (OSI) as a Software Developer, writing code to help maintain the power grid.



PETER A. FARLEY received the B.S. degree in computer engineering from the University of St. Thomas, St. Paul, Minnesota, USA in 2020. His research interests include embedded systems, autonomous navigation, and operating system design. While an undergraduate at the University of St. Thomas, he designed software for automatic guidance and data collection and analytic in the field of precision agriculture. He also researched and designed a proof of concept system to detect anomalies in HVAC systems. He is currently working as a Software Engineer at Xirgo Technologies, where his work focuses on embedded software development in the field of vehicle and freight monitoring.

TAO SUN received the dual-B.S. degrees in electronic engineering and computer science from the Huazhong University of Science and Technology (HUST), China in 2004 and the Ph.D. degree in electronic engineering from the University of Southampton, UK in 2008. In 2010, he joined Research Laboratory of Electronics (RLE) at the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA as a Research Fellow. In 2012, he was employed as a System Engineer at Oracle Inc. USA in high performance ZFS storage appliances. In 2015, he joined Inter Systems Inc. USA as a Kernel Architect in distributed system development. He is currently a System Design and Management (SDM) Fellow at Computer Science & Artificial Intelligence Lab (CSAIL) and Sloan School of Management, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. Dr. Sun's research interests are in artificial intelligence (AI), internet-of-things (IOT), smart sensors and robotics. He has published over 50 papers and patents with over 1800 citations and i10-index of 19.

CHONG XU (Senior Member IEEE) received the B.S. degree in electrical engineering from the Beijing University of Posts and Telecommunications (BUPT), China in 2004, M.S. and the Ph.D. degree in electrical engineering from University of Southampton, UK in 2006 and 2009 respectively. In 2010, she joined Information Technology Laboratory at the National Institute of Standards and Technology (NIST), Department of Commerce, Gaithersburg, MD, USA as a Research Fellow. Her work at NIST includes researches on smart grid, scheduling/routing algorithms design and MIMO systems. Since 2012, she has been employed as a full-time Faculty Member of School of Electrical and Computer Engineering at the University of St. Thomas in St. Paul, Minnesota, USA. Dr. Xu's research interests include machine learning, internet-of-things (IoT), wireless communication, smart grid, robotics and heuristic optimization algorithms.