2950 Niles Road, St. Joseph, MI 49085-9659, USA
269.429.0300 fax 269.429.3852 hq@asabe.org www.asabe.org

An ASABE Meeting Presentation
DOI: https://doi.org/10.13031/aim.201800831
Paper Number: 1800831

Deep Learning based Real-time GPU-accelerated Tracking and Counting of Cotton Bolls under Field Conditions using a Moving Camera

Kadeghe Fue[1,2], Wesley Porter[2], Glen Rains[1,2]
[1]College of Engineering, University of Georgia, Athens, GA
[2]College of Agriculture and Environmental Sciences, University of Georgia, Tifton, GA

Written for presentation at the
2018 ASABE Annual International Meeting
Sponsored by ASABE
Detroit, Michigan
July 29-August 1, 2018

ABSTRACT. ***Robotic harvesting involves navigation and environmental perception as first operations before harvesting of the bolls can commence. Navigation is the distance required for a harvester's arm to reach the cotton boll while perception is the position of the boll relative to surrounding environment. These two operations give a 3D position of the cotton boll for picking and can only be achieved by detection and tracking of the cotton bolls in real-time. It means detection, tracking and counting of cotton bolls using a moving camera allows the robotic machine to harvest easily. GPU-accelerated deep neural networks were used to train the convolution networks for detection of cotton bolls. It was achieved by using pretrained tiny yolo weights and DarkFlow, a framework which translates YOLOv2 darknet neural networks to TensorFlow. A method to connect tracklets using vectors that are predicted using Lucas-Kanade algorithm and optimized using robust L-estimators and homography transformation is proposed. The system was tested in defoliated cotton plants during the spring of 2018. Using three video treatments, the counting performance accuracy was around 93% with standard deviation 6%. The system average processing speed was 21 fps in desktop computer and 3.9 fps in embedded system. Detection of the system achieved an accuracy and sensitivity of 93% while precision was 99.9% and F1 score was 1. The Tukey's test showed that the system accuracy and sensitivity was the same when the plants were rearranged. This performance is crucial for real-time robot decisions that also measure yield while harvesting.***

Keywords. Boll counting, CNN, Cotton Bolls, Cotton counting, Cotton harvesting, DarkFlow, Darknet, Deep learning, GPU, machine vision, TensorFlow, YOLO.

Introduction

Cotton is a very important crop in the United States that utilizes large and expensive machines to harvest in a once-through system at a time when many of the bolls have been open for several weeks. Consequently, farmers suffer losses in quantity and quality because of the indeterminate ripening of the cotton fruit (boll) (UGA Cotton Team, 2017). It is imperative for United States to adopt new modern technologies and improve harvesting capacities. Current technologies which are heavy, expensive and difficult to maintain can be unprofitable as the current harvest system can cost over $600k (Fue et al., 2018). However, harvesters are expensive because of the heavy mechanical equipment used to make them; and they work only in defoliated cotton plants hence early harvesting is almost impossible (UGA Cotton Team, 2017). New harvesting technologies maypave a way to more effective and cost-efficient methods. Also, any technology to be invented needs to be able to harvest as the bolls open to reduce current losses and preserve cotton quality (UGA Cotton Team, 2017; Fue et al., 2018). However, for a machine to harvest bolls as they open requires smaller machines that do not increase compaction, destroy plant branches carrying bolls that open later and that can discriminate bolls ready to pick from cotton

canopy early in the season. A possible approach is to develop small but effective robots that can be deployed as an "army of bots" to harvest on a continuous basis as bolls open. These robots could be developed such that they are also able to harvest a diverse number of crops and remain active for longer periods of the year. The need to selectively collect cotton bolls in space with a robotic system requires very effective research on machine vision algorithms that will guide the end effector of the robotic arm (Bloch et al., 2017). Machines need to "perceive" stereoscopically the depth of field to an object so that the horizontal and vertical distances can be dynamically determined in real-time. This characteristic is essentially important for machine vision systems to locate the bolls and store 3D locations while directing a robotic arm to their location for removal. Previously, Fue et al. (2018) used color segmentation to determine the boll locations in real-time.

Color segmentation is challenging in field conditions with heavily changing illumination and dense occlusion while using a moving camera. Color space techniques like changing the obtained RGB images to Cyan-magenta-Yellow (CMY), Luminance YUV, Hue Luminance Saturation (HLS) and grey images would be effective enough if illumination and occlusion were limited (Li et al., 2017, Cheng et al., 2001 & Gauch and Hsia, 1992). This makes color selection a very difficult task (Gauch and Hsia, 1992). Also, image noise can be corrected and improved by increasing the image smoothness and brightness using equalization techniques that can be effective under certain conditions (Cheng et al., 2001). Also, some statistical and machine learning methods exist that are used to classify objects in images such as naïve Bayes classifier, logistic regression and conventional artificial neural networks (Choi et al, 2015 and Choi et al., 2016). All these techniques can improve image detection and classification, but they are limited in discriminative power and cannot differentiate bolls whenever the color appears similar or occluded by each other which is a common occurrence in the field (Fue et al., 2018, Choi et al, 2015 and Choi et al., 2016). So, it is imperative to investigate alternative methods for detection such as deep neural networks which has proved to be very effective, even in challenging lighting conditions (Kamilaris and Prenafeta-Boldú, 2018).

Deep neural network uses a lot of data to learn detailed representation of features while traditional machine learning uses just limited data to learn the features. It is an analytical approach developed to mimic the biological nervous system of most animals (Kamilaris and Prenafeta-Boldú, 2018). Deep learning algorithms have recently been used to solve many machine vision problems. One key to this surge is the ease with which the algorithms can be implemented using open-source deep learning application frameworks such as TensorFlow, Keras, Theano, Caffe, Torch and others. Deep learning (DL) can be implemented using various architectures like deep neural networks, recurrent neural networks and deep belief networks. All these architectures can be learned by supervised, semi-supervisedor unsupervised models (Kamilaris and Prenafeta-Boldú, 2018). These DL networks consist of input and output layers while in between there are several hidden layers (Kamilaris and Prenafeta-Boldú, 2018). Adding layers can improve model accuracy, but it also slows the processing speed of the model (Redmon and Farhadi, 2017). This is because the networks are made up of several neurons and each neuron has a learnable bias and weight. Each neuron receives the inputs, processes it and gives output to the next neuron and may follow non-linearity. In machine vision applications, DL networks need to be trained using known images. (Redmon and Farhadi, 2017; Kamilaris and Prenafeta-Boldú, 2018).

In this research, supervised convolution deep learning neural network architecture was deployed to detect and recognize the cotton bolls using state of art detector, YOLO v2 model (Redmon and Farhadi, 2017; Redmon, 2016 and Redmon et al., 2016). A similar investigation approach using deep learning to detect cotton bolls was done before (Li et al., 2017) and proved to outperform most of the existing methods of cotton boll classification. However, the approach used a slower technique called deep fully convolutional neural network to do semantic segmentation which is not a good candidate for robotic use. Fortunately, the advancement of graphical processing units (GPUs) to speed up processing in computers provides a method to implement DL systems.

The current GPU technology allows quick training of the deep learning algorithms and is supported by several frameworks such as Tensorflow, Keras, Caffe2, NVIDIA Caffe, MXNet and pyTorch . TensorFlow was used to train the DL network deployed in this study. This was achieved by using the DarkFlow framework which translates the pretrained weights from darknet to TensorFlow. Darknet is a neural network framework implemented in C and cuda (Redmon et al, 2017). YOLOv2 classification is based on modified Darknet-19 which has 19 convolutional layers and 5 max pooling layers (Redmon and Farhadi, 2017). In this research, the deep learning model was retrained using data that was manually annotated for cotton bolls. And then, color segmentation was used to leverage any missed bolls. Then, the algorithm to track and count bolls was developed.

Therefore, the main objective of this study was to detect, track and count the cotton bolls under field conditions. The specific objectives achieved were to;

1. Develop the model to detect cotton bolls in real-time by combining DL techniques, color segmentation, optical flow and statistical tracking methods.
2. Detect cotton bolls from a mobile camera using model developed in objective 1
3. Count the bolls as they cross the out of scene line (CoSL) of the camera. This is the bottom horizontal line that can be assumed as the maximum displacement position a cotton-picking end effector could reach.

Materials and Methods

Experimental Set-up (Training Dataset)

An experiment was set up at the University of Georgia (UGA) Tifton campus grounds (N Entomology Dr, Tifton, GA, 31793) at (31˚28'N 83˚31'W). The chosen location was open to direct sunlight to simulate field conditions. Twelve defoliated cotton plants were cut from a nearby farm and put in pots. The plants were placed in 2 columns of 6 rows, 91.4 cm between the center of the stalk and each stalk 61 cm from the next. A moving camera Samsung Galaxy S6 Edge Plus (Samsung Electronics, Seoul, South Korea) at less than 3 mph and at least 1.5m above the ground took 720p images of the plants facing downward at the rate of 30 fps. Samsung camera has f/1.9 aperture and has 16 Mega pixels lens. Other training images were taken using ZED camera (Stereo labs Inc, San Francisco, CA, USA) that is attached to an embedded system (NVIDIA Jetson TX2 development kit, Nvidia Corp., Santa Clara, CA, USA) that is mounted on a research red rover which is moving at less than 3 mph (Rains, 2015). ZED camera has f/1.9 aperture and has a 16 Mega pixels lens. The ZED camera was at least 1.5m above the ground taking 720p images as the rate of 15 fps. Both were tilted (facing downward) at angle less than 90^0 but greater than 80^0 from horizontal Getting variable angles with Samsung was important for the developed algorithm to have a very robust detection capability. All the images were used to train the deep learning algorithms. It is important to use a mixture of images from different cameras to enhance the model detection effectiveness.

Experimental Set-up (Testing Dataset)

After training, the 12 cotton plants were replaced. On April 12th, 2018, three test video images (3 treatments) were taken using a moving Samsung camera. In each of the test videos taken, the 12 plants were randomized and rearranged so that the model can validate the counting algorithms. In testing the model, 9 parameters were obtained and calculated. Detailed images were chosen. Since each video has more than 600 images, it was decided to select 20 images from each video for detailed analysis. Frame numbers in sequence of 50 frames were chosen. Speed of processing images with the embedded system was measured using frame per second (fps) and average frame per second (afps). The reported bolls counted was also collected. Using manual methods, true positives (TP), false positives (FP), false negatives (FN) and multiple detections (MD) were collected (Sokolova, 2006). MD means the true positives are detected twice or more on the same boll. True negatives were not determined because the system was not made to find true negatives but to ignore them. After that, the sensitivity or recall, accuracy, precision and the weighted average (F1 score) of the model algorithms were determined as follows (Sokolova, 2006):

$$\begin{aligned} sensitivity\ (recall)\ &= \frac{TP}{TP+FN} \\ accuracy &= \frac{TP}{TP+FP+FN} \\ precision &= \frac{TP}{TP+FP} \\ F1\ score &= \frac{2*recall*precision}{recall+precision} \end{aligned} \tag{1}$$

Image Processing System and Extraction

From a Samsung Galaxy cell phone, the videos were transferred to a desktop Lenovo Legion Y520 (NVIDIA GeForce GTX-1050 Ti graphics running an Intel i5 7th generation CPU) with Windows 10 installed. The images obtained using ZED camera were also transferred to the Lenovo desktop that had the TensorFlow framework installed. Training data and testing data were both loaded to the Lenovo. Later, the videos were loaded onto the Ubuntu operated Jetson TX2 embedded computer to compare the speed of processing between the Jetson TX2 and Lenovo Y520. The comparison was studied to help optimize the performance of the Jetson TX2.

Data Training and Testing using TensorFlow Framework

In this study, convolutional deep learning (CNNs) model YOLO v2 ((Redmon and Farhadi, 2017)) is deployed for boll detection. Tracking is performed by the OpenCV Lucas-Kanade algorithm (Lucas and Kanade, 1981) and L-estimators and homography transformation are used to detect outliers of the Lukas-Kanade algorithm.

Twelve plants were used for training and another group of 12 plants were used for testing. Four hundred eighty-six images with 7498 bolls were used to train the tiny YOLOv2 model. The image taken for training were of different quality from 360p

(31 images), 720p (404 images), to 1080p (51 images). This gives a CNN model a chance to learn different image size challenges of the images. Also, to improve the model detector, the images were taken at different times of the day. Some of the images were taken during the morning (212 images), some afternoon (121 images) and some during the evening (97 images). Some images taken during the daylight, sun was occluded to create shadow over the images and create varying illumination. Unfortunately, most of the images were taken from defoliated plants. Very few images (56 images) from online were also included on training data. In later stages, when the model will be tested in nondefoliated plants, more samples will be taken to improve the model.

A program that produces bounding boxes around the cotton boll was developed and used to determine each cotton boll selection in the images. The program accepts the image and allows the user to create bounding boxes around the objects (cotton bolls). Then, the program generates an annotation file that gets all the information of the cotton bolls present in that picture. Using trained weights called "tiny yolo" provided by Darkflow, we were able to retrain the model by modifying the configurations to accept only one label and reduce the number of filters to 30 with one class and one label. The filters were

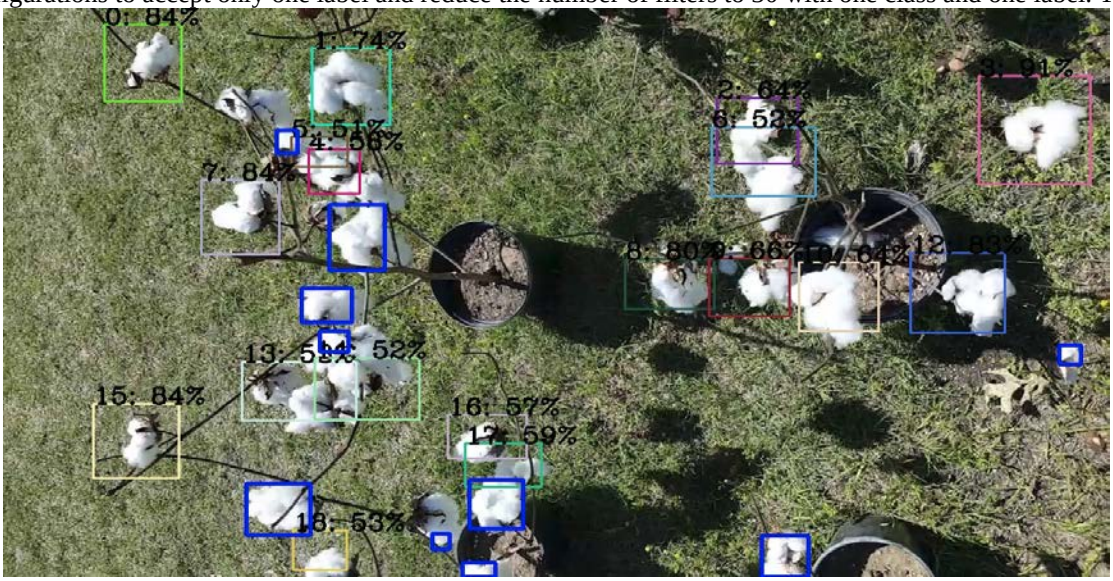

Figure 1. Detection of the cotton boll using both color segmentation (blue boxes) and deep learning algorithm (other color boxes). The deep learning can show percentage of detection and only 50% above were considered enough to be boll and hence shown. All the percentages are above 50%.

modified, labels and configuration file to reflect the one class "cotton_boll". The trained model "cotton boll" which recognize only cotton bolls was then tested using 3 different videos. After training, the generated load file and model files were frozen to form .pb and .meta files to allow easy deployment between desktop and Jetson.

Later, the videos were transferred to Jetson and TensorFlow was installed onto the Jetson TX2. Jetson TX2 was accessed using secure shell (SSH) instead of High-Definition Multimedia Interface (HDMI) screen so that the GPU usage could be reduced and restrict RAM for image processing only. However, the Jetson TX2 can be optimized with the TensorRT inference engine in the future to speed up deep learning inference.

Detection, Redetection and Tracking of Cotton Bolls

After the detection of bolls using TensorFlow, some of the previously missed bolls need to be detected. The color segmentation algorithm detects the boll by differentiating the color of the bolls to get any missed boll. The cotton boll color segmentation task involved four steps (Fue et al., 2018 & Gong and Sakauchi,1995):

1. Grab an image frame,
2. Using RGB color threshold, separate each RGB channel of the image. For cotton bolls, the white components of the image can be masked (All the red, green, and blue channels set above 240).
3. Subtract the image background from original image.
4. Remove all the region where the contours are less than value M. Value M can be determined by estimating the number of pixels defining the smallest boll (In this research, M was set to 10 pixels).

Then, all the regions that were detected by the deep learning algorithm are subtracted. Figure 1 shows blue bounding boxes of the bolls missed by YOLOv2 but were detected using color segmentation. YOLOv2 detected bolls are presented by using other color bounding boxes. You can evidently see how color segmentation cannot be effective in detecting the whole boll if there is an occlusion or lighting change (e.g. shadows). When using color segmentation alone, it is extremely difficult to detect nearby bolls which may occlude each other. Also, since bolls are located different distances from the ground and the camera is above, lower bolls tend to appear when they first appeared in frames and become occluded as the

camera moves near the bolls. This may become a very difficult task hence when occluded, the system determined and established a dummy boll to exist together with the upper boll. For multiple detections in YOLOv2 and color segmentation, non-maximum suppression (NMS) was applied to remove any repetition. NMS uses gradient direction where all pixels or bounding boxes that are not part of the local maxima are excluded. Figure 2 shows the nearby bolls detected using YOLOv2 and color segmentation. All the bolls are detected, and the bounding boxes predict the right position of the bolls.

Figure 2. Detection of near and occluded cotton bolls. The bounding boxes (light pink, purple, magenta, dark green, sea green) present 5 bolls occluded over each other and the system is trying to show all of them.

The technique is established that detect the bolls as "dummy" blobs and store them in a vector related to the missed boll. Figure 3 shows a sky blue bounding box with a feature which disappears in frames before returning. The bounding box continues to be predicted out of scene. When the boll (tracking feature) reappears then the "dummy" is authenticated as genuine. This technique provides a method to "remember" out of scene (lost) bolls.

Remembering bolls as they go in and out of the camera scene is achieved by assigning features of the detected boll to a Lucas-Kanade (LK) algorithm which uses a mathematical technique to predict optical flow of the boll sparse features (Lucas and Kanade, 1981). Since, Lucas-Kanade cannot accurately provide flow information to the texture-less neighbors of the selected features, a guided feature detection is proposed that redirects the Lucas-Kanade to the blob detected that gives the LK a clue of the position of the boll using the nearby detected good features. The nearby detected features were captured using good features to track algorithm also known as Shi-Tomasi Corner Detector (Shi & Tomasi, 1994) implemented in OpenCV. Since LK uses displacement of image contents between two frames, then LK determined movement of all these features. The system only considered features enclosed within the bounding boxes to achieve real-time movement of the boll features. This reduced the number of points to track and allowed the system to determine occlusion between two bolls. A tracked boll is connected using a tracklet in each image. A tracklet is a line that tracks and connects the feature transformation from one image to the next one. In Figure 3, the context diagram shows a clear movement of the feature and cotton boll. As the feature disappears from the frame scenes, the algorithm can still predict and track in background so that the bolls positions cannot be lost.

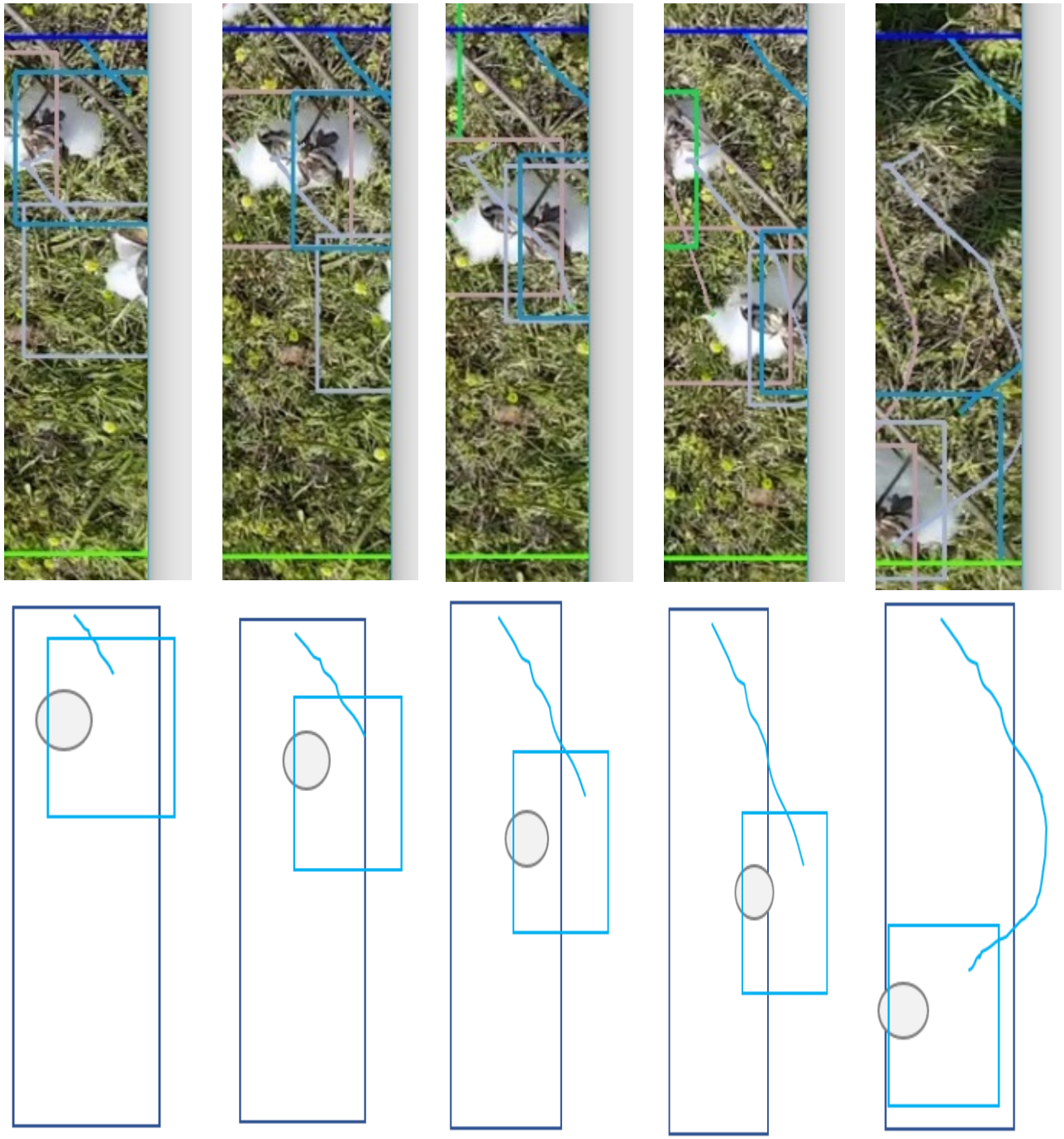

Figure 3. Image vs Context. Boll tracked with sky-blue bounding box (disappeared and reappeared). The tracklet tracks a feature near the boll. This feature happens to disappear at image number 2 and reappears at image number 5. This feature was randomly selected using Shi-Tomasi corner feature detector. In context diagram, the round object is the cotton boll while the sky-blue lines present the bounding box and a feature tracklet.

Whenever the LK gets the outlier, the algorithm assumes a displacement of points as a Gaussian distribution; hence robust L-estimator method (Rousseeuw, & Croux, 1992) and homography transformation matrix (Chum et al., 2005) are determined and used to check if the predicted point is within 99.3% distribution by determining the 1.5 IQR (interquartile range) in classical boxplots. The homography transformation matrix was obtained by choosing only feature transformations that are near the detected bolls since many points may heavily affect the system performance. Mathematically for Gaussian distribution (Eqn. 2), probability density function is defined to be normal when the IQR covers 50% distribution (Central region) and anything out of the 1.5 IQR is an outlier. Within this technique, we can change the prediction to be within 1.5 IQR. The Figure 3 shows the tracklet of the disappear boll features, were lost due to boll being out of scene but eventually they were restored when the boll came back.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad (2)$$

In the gaussian equation above, f(x) can be solved if standard deviation (σ) of the feature displacements is found and the mean (μ) of the displacements.

So, in Eqn. 3, the predicted point x can only be validated if it is not detected as an outlier compared to all other points.

$$f(x) = \begin{cases} x, \ if \ min(CR) - 1.5 * IQR < x < max(CR) + 1.5 * IQR \\ min(CR) - 1.5 * IQR, \ if \ x < min(CR) - 1.5 * IQR \\ max(CR) + 1.5 * IQR, \ if \ x > max(CR) + 1.5 * IQR \end{cases} \quad (3)$$

Tracking and Counting Cotton Bolls

Each image was divided into three parts horizontally (Figure 4). Since images are 720p, the first part are pixels from 0 to 360 (middle of the image [blue line]), the second division (CoSL) is at 700 (green line) that divides each image into two other parts. The first part is considered as the first appearance of bolls where the system tries to establish the tracklets accordingly; while after the boll passes the blue line, it considers the established tracklet is valid and the system should force the tracklet to exist even when the boll is not detected. The third part is 10 centimeters from the camera base were the robotic arm will be placed. The system counted the bolls when they passed the green horizontal line. Table 1 summarizes the proposed algorithm using the pseudocode and flowchart of the sequence of events.

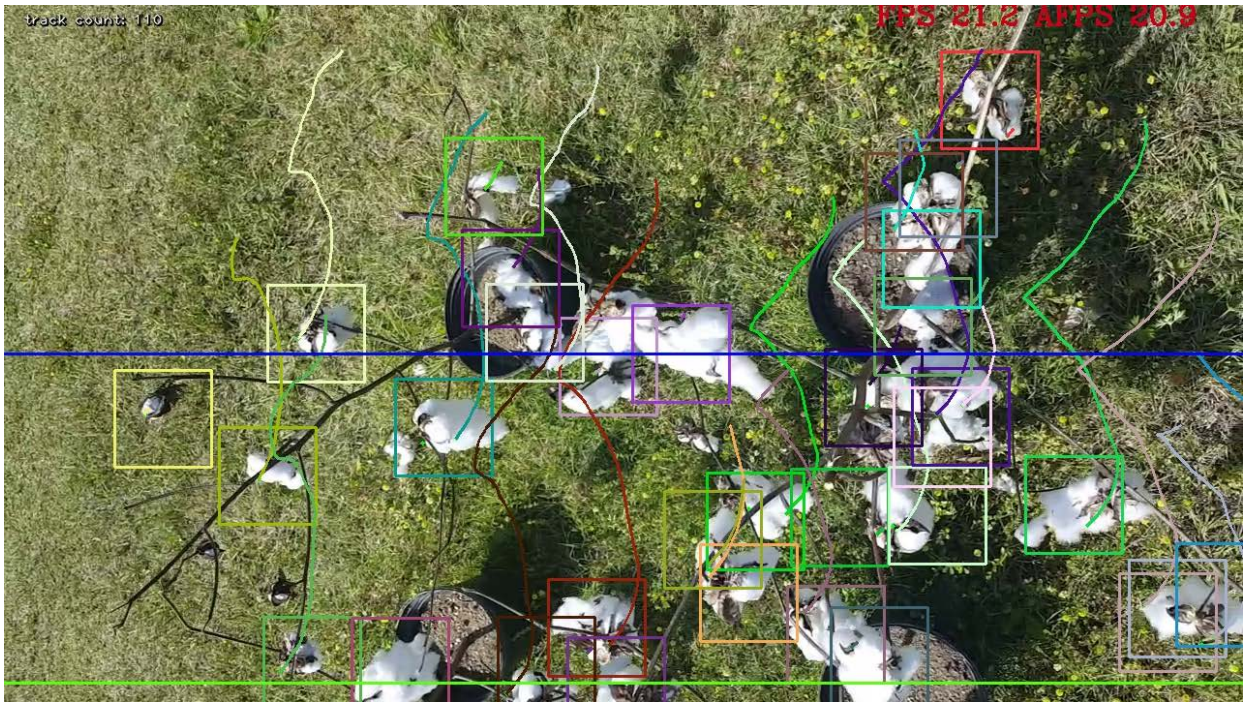

Figure 4. Bolls bounding boxes crossing blue line and green line (CoSL). Each boll that is initially detected cannot have a tracklet, but the previous detected bolls have a tracklet that is terminated inside the bounding box from the first position it was tracked.

The proposed algorithm (Table 1) can be summarized into 4 parts of processing; input, output, detection and tracking/counting of cotton bolls. The input is an image received from the camera frame and output is an image that consists of bounding boxes, boll position and boll counting. In this process, the image I_t received can be matched with previous image I_{t-1} using LK algorithm together with homography transformation. The system processed images by predicting the position of the bolls, determine the missed bolls, remove the multiple detections and determine the disappeared bolls. After that, using Horizontal line Crossing positions (CoSL), the system counts all the bolls that pass the line.

Table 1. Proposed algorithm pseudocode

Algorithm. Proposed tracking and counting approach: iteration at time t
Input:
Image I_t
Previous Image I_{t-1}
Model prediction of objects in I_{t-1} using LK algorithm, target positions P_t
Current position of objects predicted by YOLOv2, Ct
Output:
Matched positions M_t of Ct and P_t
Missed positions St from Ct and P_t, given as St = Pt - M_t
New positions Nt from Ct
Horizontal line Crossing positions (CoSL) H_t
Current boll detection and tracking:
1: Determine if the predicted image P_t is matched with current image Ct
2: Detect the bolls using deep learning YOLOv2
3: Determine the missed bolls using image segmentation and include them.
4: Perform non-maximum suppression to remove all the bounding boxes that do not have multiple detections
5: Determine if the bolls have the first appearance in the current image N_t and establish a tracklet
6: Determine if the previous boll has crossed the horizontal line H_t or is just outside the scene S_t
7: Determine if the boll was detected but disappeared using homography transformation matrix obtained by using LK output tracklets.
8: Determine the very short or very long tracklets and modify by using LK algorithm which utilizes the classical boxplots.
Current boll counting:
9: Determine and count the number of the bolls crossing the horizontal line, H_t and log the information

Results and Discussion

In Table 2, several parameters are reported from the experiments. All three treatments with 20 frames each are investigated. Sixty image frames from 3 videos are chosen and in interval to avoid any biasness. The frames were chosen in intervals of 50 from frame number 50 and 51 to 500 and 501 respectively. Since the system algorithms refreshes and get new points using feature detector every 5 frames, frames divisible by 5 will reflect the system speed and performance when getting new points. Bolls were manually counted, and 131 cotton bolls were found. The general algorithm performance parameters (Table 2) were recorded for each video. These parameters were loading time of the neural network, neural network building time, average fps and counting output of each video. Using the counting output, the system performance, error and accuracy was determined. The system accuracy was around 94% and speed of processing was approximately 21 fps for Lenovo and 4 fps for Jetson TX2.

Table 2. General algorithm performance parameters

	1st Video	2nd Video	3rd Video	Average
Loading time (sec)	6.75E-05	1.80E-04	7.01E-05	1.06E-04
Net building time (sec)	3.32	7.60	3.46	4.79
Average FPS (Lenovo)	21.40	22.50	21.10	21.67
Average FPS (Jetson TX2)	3.9	4.0	4.0	3.9
Cotton boll Count	141	135	140	138.67
% Performance in counting	107.63	103.05	106.87	105.85
% Error in counting	7.63	3.05	6.87	5.85
% Accuracy in counting	92.37	96.95	93.13	94.15

Apart from general performance, the system was investigated in detail to compare the manual detection against automated performance. The frames chosen were checked for true positives (TP), false positives (FP), false negatives (FN) and multiple detections (MD). Figure 5 shows a worst-case example where multiple detection (MD) of two bolls was made using four bounding boxes (6,7,8 and 9).

The system sensitivity was 93% with standard deviation of 6% while the accuracy was almost the same because the false positives are negligible (Table 3). Also, there were no observed differences in the three treatment videos as p-value for sensitivity and accuracy obtained using Tukey's test was 0.6087 and 0.6481, respectively.

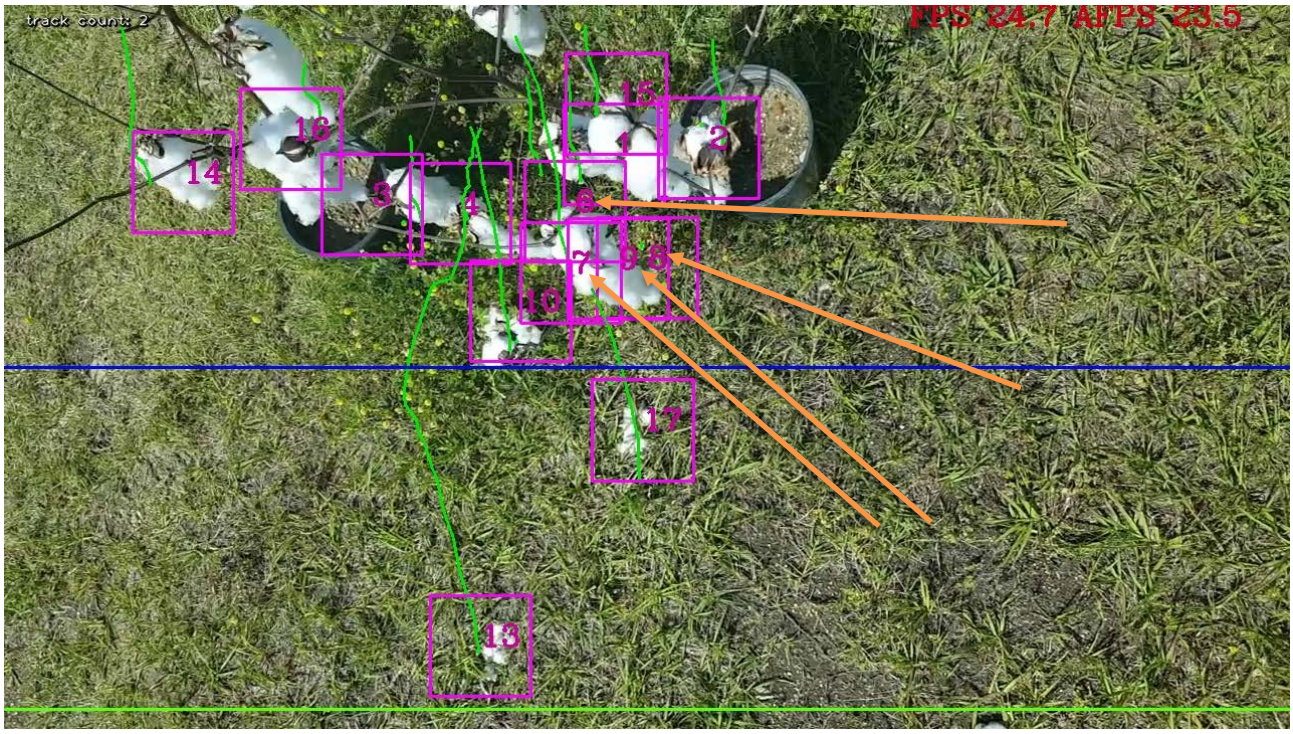

Figure 5. Multiple detections for boll numbered 6, 7,8 and 9 . This means the system was counting the same boll twice or trice leading to counter errors. The yellow arrows point to the four bounding box numbers 6,7,8 and 9. There is only two bolls that are represented by four bounding boxes.

Table 3. Detailed performance measurement using manual inspection over automatic detection

	TP	FP	FN	MD	SENSITIVITY (%)	ACCURACY (%)	PRECISION (%)	F1
1st Video	27.1 (8.05)	0.1 (0.22)	2 (1.62)	1.5 (1.05)	93.4[a] (5.43)	93.3[a] (5.6)	99.8 (0.68)	1 (0.03)
2nd Video	22.3 (9.78)	0 (0)	1.6 (1.19)	1.2 (0.88)	93[a] (6.03)	93[a] (6.03)	100 (0)	1 (0.03)
3rd Video	23 (9.31)	0.1 (0.22)	1.3 (1.42)	0.8 (0.62)	94.8[a] (5.9)	94.6[a] (5.93)	99.9 (0.64)	1 (0.03)

The system performance speed decreased as the number of bolls increased. From the observation in Figure 6, the system starts at high speed at around 24 fps and decreases to 19 fps before coming back to 24 fps as it finishes the rows. The same happened using the Jetson TX2 where the speed started at 4.8 fps and decreased to 4 fps (Figure 6). The system becomes slow because of the vast number of vectors (more than 5000) created for tracking. Vectors were used to store the tracklets. They were cleared whenever they reach 50 arrays to increase the performance. This is important especially when the system is deployed in embedded systems because the GPU and CPU share the RAM system that is also used to store the tracklets compared to desktop systems that have a dedicated RAM for CPU and a different RAM for GPU.

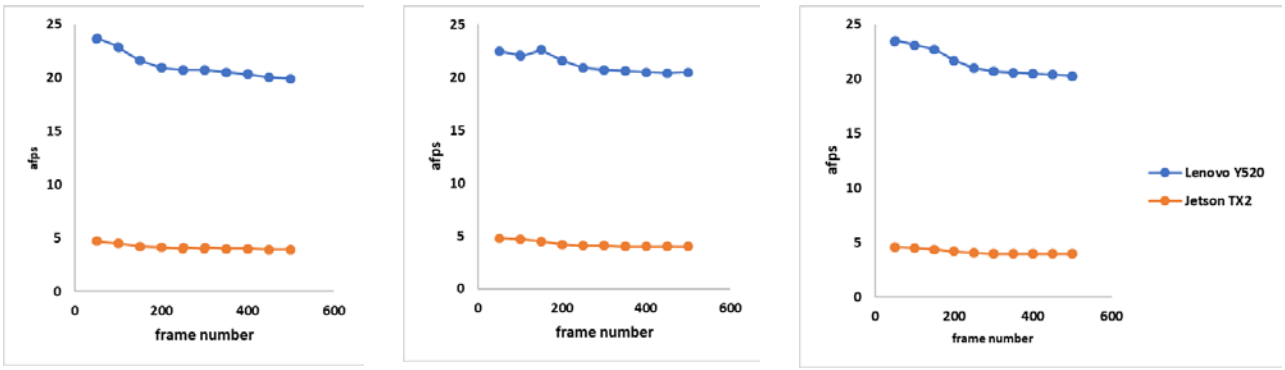

Figure 6. Comparison between LenovoY520 and Jetson TX2 system

Conclusion

In this study, a machine vision system method that detected, tracked and counted cotton bolls in real time was developed. The system tracked and counted cotton bolls at 93% detection accuracy and an average processing speed of 21 fps and 4 fps for Lenovo Y520 and Jetson TX2, respectively. Sensitivity and accuracy were the same for all three tests using the Tukey's test. However, for this experiment, precision and F1 score may be misleading the results as they were 99.9% and 1, respectively. This is because the experiments carried out in this research had negligible false positives and mostly suffered from false negatives. The system developed used two unified techniques (deep learning methods and color segmentation) to detect cotton bolls and used three other techniques (Lucas-Kanade algorithms, robust L-estimators and homography transformations) to track lost tracklets. To increase the system accuracy, alternative deep learning methods and color segmentation methods may be used to improve false negatives. However, the Jetson TX2 performance may be accelerated using TensorRT which is a C++ library that can facilitate high performance inference on NVIDIA GPUs. The envisioned embedded system will use stereo cameras to determine 3D position of the boll. Likely, if a speed of more than 10 fps is achieved in Jetson, then, the system will be ready for robotic operation and harvesting.

Acknowledgements

The authors are very thankful for project support from Cotton Incorporated, Agency Assignment #17-038. We also thank Mr. Ricky Fletcher and Mr. Gary Burnham for their helpfulness and technical support in building camera platform and collection of data.

References

University of Georgia Cotton Team. (2017). Georgia cotton production guide. University of Georgia Cooperative Extension Service. Retrieved from http://www.ugacotton.com/production-guide/

Bloch, V., Bechar, A., & Degani, A. (2017). Development of an environment characterization methodology for optimal design of an agricultural robot. Industrial Robot: An International Journal, 44(1), 94-103. https://doi.org/10.1108/IR-03-2016-0113

Chum, O., Pajdla, T., & Sturm, P. (2005). The geometric error for homographies. Computer Vision and Image Understanding, 97(1), 86-102. https://doi.org/10.1016/j.cviu.2004.03.004

Rains, G., B. Bazemore, K. Ahlin, A. Hu, N. Sadegh, & McMurray, G. (2015). Steps towards an Autonomous Field Scout and Sampling System. ASABE Paper No. 15219077. St. Joseph, MI: ASABE.

Fue, K., Porter, W., and Rains, G. (2018). Real-Time 3D Measurement of Cotton Boll Positions Using Machine Vision Under Field Conditions. BWCC Paper No. 18385. Cordova, TN: NCC

Li, Y., Cao, Z., Xiao, Y., & Cremers, A. (2017). DeepCotton: in-field cotton segmentation using deep fully convolutional network. Journal of Electronic Imaging, 26(5). https://doi.org/10.1117/1.JEI.26.5.053028

Lucas, B. & Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In Proceedings of the 7th international joint conference on Artificial intelligence. 2, pp. 674-679. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Gauch, J. M., & Hsia, C. W. (1992). Comparison of three-color image segmentation algorithms in four color spaces. In Visual Communications and Image Processing Vol. 1818, pp. 1168-1182. Bellingham, Washington: SPIE

Gong, Y., & Sakauchi, M. (1995). Detection of regions matching specified chromatic features. Computer vision and image understanding, 61(2), pp. 263-269. https://doi.org/10.1006/cviu.1995.1018

Cheng, H. D., Jiang, X. H., Sun, Y., & Wang, J. (2001). Color image segmentation: advances and prospects. Pattern recognition, 34(12), 2259-2281. https://doi.org/10.1016/S0031-3203(00)00149-7

Choi, D., Lee, W., Ehsani, R., Schueller, J., & Roka, F. (2016). Detection of dropped citrus fruit on the ground and evaluation of decay stages in varying illumination conditions. Computers and Electronics in Agriculture, 127, 109-119. https://doi.org/10.1016/j.compag.2016.05.020

Choi, D., Lee, W., Ehsani, R., & Roka, F. (2015). A machine vision system for quantification of citrus fruit dropped on the ground under the canopy. Transactions of the ASABE, 58(4), 933-946. https://doi.org/10.1016/j.compag.2016.05.020

Kamilaris, A., & Prenafeta-Boldú, F. (2018). Deep learning in agriculture: A survey. Computers and Electronics in Agriculture, 147, 70-90. https://doi.org/10.1016/j.compag.2018.02.016

Redmon, J., & A. Farhadi, A. (2017). YOLO9000: better, faster, stronger. arXiv preprint .

Redmon, J. (2016). Darknet: Open source neural networks in c. Pjreddie. com.[Online]. Available: https://pjreddie. com/darknet/. Retrieved from 21-April-2018

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. CVPR Paper No. 16526932. Piscataway, NJ: IEEE.

Rousseeuw, P. J., & Croux, C. (1992). Explicit scale estimators with high breakdown point. In Book Statistical analysis and related methods (pp. 77-92). Dodge, Amsterdam: North-Holland. Retrieved from https://feb.kuleuven.be/public/u0017833/PDF-FILES/l11992.pdf

Shi, J., & Tomasi, C. (1994). Good features to track. CVPR Paper No. 4764232. Piscataway, NJ: IEEE.

Sokolova, M., Japkowicz, N., & Szpakowicz, S. (2006). Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In *Australasian joint conference on artificial intelligence* (pp. 1015-1021). Berlin, Heidelberg: Springer.