Automated Crop Yield Estimation for Apple Orchards

Qi Wang, Stephen Nuske, Marcel Bergerman, Sanjiv Singh

The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
{wangqi, nuske, marcel, ssingh}@cmu.edu

Abstract   Crop yield estimation is an important task in apple orchard management. The current manual sampling-based yield estimation is time-consuming, labor-intensive and inaccurate. To deal with this challenge, we develop and deploy a computer vision system for automated, rapid and accurate yield estimation. The system uses a two-camera stereo rig for image acquisition. It works at nighttime with controlled artificial lighting to reduce the variance of natural illumination. An autonomous orchard vehicle is used as the support platform for automated data collection. The system scans the both sides of each tree row in orchards. A computer vision algorithm is developed to detect and register apples from acquired sequential images, and then generate apple counts as crop yield estimation. We deployed the yield estimation system in Washington state in September, 2011. The results show that the developed system works well with both red and green apples in the tall-spindle planting system. The errors of crop yield estimation are -3.2% for a red apple block with about 480 trees, and 1.2% for a green apple block with about 670 trees.

1 Introduction

Crop yield estimation is an important task in apple orchard management. Accurate yield prediction (by number of apples) helps growers improve fruit quality and reduce operating cost by making better decisions on intensity of fruit thinning and size of the labor force for harvest. It benefits packing industry as well, because managers can use estimation results to have an optimized capacity planning for packing and storage. Typical yield estimation is performed based on historical yields, weather conditions, and measurements manually taken in orchards. Workers conduct manual measurements by counting apples in multiple sampling locations. This process is time-consuming and labor-intensive. The limited sample size is usually not enough to reflect the yield distribution across an orchard, especially

in those with high spatial variability. Therefore, the current practice for yield estimation is inaccurate and has been an obstacle to the development of the industry.

Apple growers desire an automated system to conduct accurate crop yield estimation; however, there are no off-the-shelf tools serving this need. Researchers have been working on the development of related technologies for a few decades [1]. A wildly adopted solution to automated fruit yield estimation is to use computer vision to detect and count fruit on trees. Swanson et al. [2] and Nuske et al. [3] developed computer vision systems to estimate the crop yield of citrus and grape, respectively. However, there is no reported research leading to satisfactory yield estimation for apples.

Current efforts on apple yield estimation using computer vision can be classified into two categories: (1) estimation by counting apples and (2) estimation by detecting flower density. A few researchers have worked on the first category using color images [4-6], hyperspectral images [7], and thermal images [8]. Their common point is that they only deal with apple detection from a single or multiple orchard scenes; however, no further research is reported about yield estimation, which requires continuous detection and counting. Aggelopoulou et al. [9] worked on the second category. They sampled images of blooming trees from an apple orchard, and found a correlation between flower density and crop yield. However, this flower-density-based method is not persuasive because there are multiple unpredictable factors (such as weather conditions) during the long period between bloom and harvest, the correlation could vary year by year.

When conducting the apple-counting-based yield estimation, computer vision systems face three challenges due to the characteristics of orchard environments.

- Challenge 1: variance in natural illumination. It prevents from developing a reliable vision-based method to detect apples from an orchard scene.
- Challenge 2: fruit occlusion caused by foliage, branches, and other fruit.
- Challenge 3: multiple detections of same apple in sequential images. Unsuccessful registration of these detections will cause miscounting.

Our overall research goal is to design, develop, and deploy an automated system for rapid and accurate apple yield estimation. The system reduces labor intensity, and increases work efficiency by applying computer vision-based, fast data acquisition. Meanwhile, it improves prediction accuracy by conducting a large-scale data acquisition. At the initial stage of the research, we focus on two specific objectives: (1) build up system hardware and major algorithm modules for data acquisition and yield estimation; (2) conduct preliminary performance tests in an orchard.

2 System Overview

The hardware of the yield estimation system consists of three major parts (Fig. 1).

Part 1: a stereo rig. The stereo rig consists of two identical high-resolution monocular cameras, Nikon D300s (Nikon Inc., Melville, NY, USA) with wide-angle lenses (focal length: 11 mm). The camera is a consumer product with a low cost comparing to industrial or scientific imaging systems. The two cameras are mounted on an aluminum bar with a distance of about 0.28 m to form a stereo rig. A synchronizer triggers the two cameras synchronously at 1 Hz.
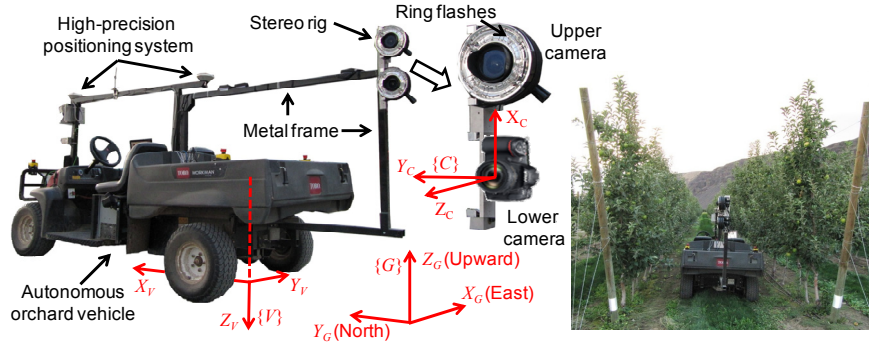

Fig. 1 The hardware of the crop yield estimation system. Three coordinates are used by the system: camera coordinates $\{C\}$, vehicle coordinates $\{V\}$, and ground coordinates $\{G\}$. $\{C\}$ originates at the focal point of the lower camera; $\{V\}$ originates at the projection of the central point of the rear axle of the vehicle on the ground; $\{G\}$ is a combination of the Universal Transverse Mercator coordinate system and elevation. The geometrical relationship between $\{C\}$ and $\{V\}$ is calibrated. The geometrical relationship between $\{V\}$ and $\{G\}$ is determined by the on-board positioning system.

Part 2: controlled illumination. The system is designed for night use to avoid interference from unpredictable natural illumination (Challenge 1 in Section 1). Ring flashes (model: AlienBees ABR800, manufactured by Paul C. Buff, Inc. Nashville, TN, USA) around the two lenses are used as active lighting during image acquisition. The power of each flash is set at 20 Ws. The two cameras are both set with aperture f/6.3, shutter speed 1/250 s, and ISO 400 for an optimal exposure of apple trees (about 2 m away from the cameras) under this controlled illumination.

Part 3: a support vehicle. An autonomous orchard vehicle [10] developed at Carnegie Mellon University is used as the carrying platform for automated data acquisition. The platform is able to travel through orchard aisles at a preset constant speed by following fruit tree rows. The speed is set at 0.25 m/s for our data acquisition. The stereo rig is attached to a frame at the rear of the vehicle (Fig. 1). Each tree row is scanned from both sides. The acquired sequential images provide multiple views of every tree from different perspectives to reduce fruit occlusion (Challenge 2 in Section 1). The on-board high-precision positioning system, POS LV manufactured by Applanix (Richmond Hill, Ontario, Canada), provides the geographic coordinates of the vehicle. The position and pose of the vehicle is used by the system software to calculate the geographic position of every detected ap-

ple. We use the global coordinates of apples to register the multiple detections of same apple to reduce over counting (Challenge 3 in Section 1).

The software of the crop yield estimation system has two major parts: (1) online processing, and (2) post-processing. The online processing controls the start and the stop of data acquisition. It is written in Python (Python Software Foundation). The software processes the acquired data off-line for apple detection, apple registration, and final apple count. Matlab 2010a (The MathWorks, Inc., Natick, MA, USA) is the programming language for post processing.

3 Apple Detection

The algorithm uses the following procedure to detect apples from an image. Firstly, it reads a color image (1072 × 712 pixels) acquired by the system and removes distortion. Then, it uses visual cues to detect regions of red or green apples in the image. Finally, it uses morphological methods to covert apple regions into apple counts in the image.

3.1 Detection of Red Apple Pixels

Under the controlled illumination, the red color of apples distinguishes from the colors of other objects in the orchard, such as ground, wires, trunks, branches, and foliage (Fig. 2a). The algorithm uses hue, saturation, and value in the HSV color space as visual cues for red apple detection. The hue values of red apple pixels are mainly in the ranges from 0° to 9° and from 349° to 360°. The hue values of other objects are out of the two ranges. It is necessary to exclude the background pixels during hue segmentation for red apple pixels because hue is undefined for pixels without any color saturation (white, grey, black colors), and for dark pixels close to black the saturation is low and the hue channel is an unreliable. Therefore, the procedure for red apple segmentation is: (1) segment pixels with $0° \leq hue \leq 9°$ or $349° \leq hue \leq 360°$; (2) remove (background) pixels with saturation ≤ 0.1 or value ≤ 0.1. After the processing, the regions of red apple are segmented from the image (Fig. 2b).

3.2 Detection of Green Apple Pixels

We use three visual cues: hue, saturation, and intensity profile to detect green apple pixels from an image. Analysis of 10 sample images shows that the hues of green apples and foliage are mainly in the range from 49° to 75°. We use this hue

range to segment green apples and foliage from an image. The dark background and most non-green objects are removed after the hue segmentation (Fig. 3b). Although they are both green, the apple pixels have a stronger green color which can be separated from leaves using the saturation channel. The algorithm uses a saturation threshold (≥ 0.8) to segment green apple pixels. Most foliage pixels are removed after the saturation segmentation (Fig. 3c); however, the central parts of most apples are removed as well because the camera flashes generate specular reflections at the central part of a green apple. In HSV color space, specular reflection has high brightness and low saturation. They cannot be detected as apple pixels by the saturation segmentation.
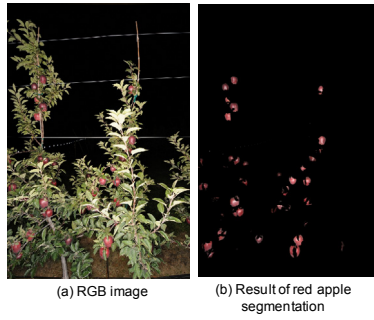

(a) RGB image   (b) Result of red apple segmentation

Fig. 2 Red apple segmentation using hue, saturation and value (brightness) as visual cues.

(a) RGB image   (b) Hue segmentation   (c) Saturation segmentation   (d) Detection of specular reflection   (e) Combination of the results from (c) and (d)
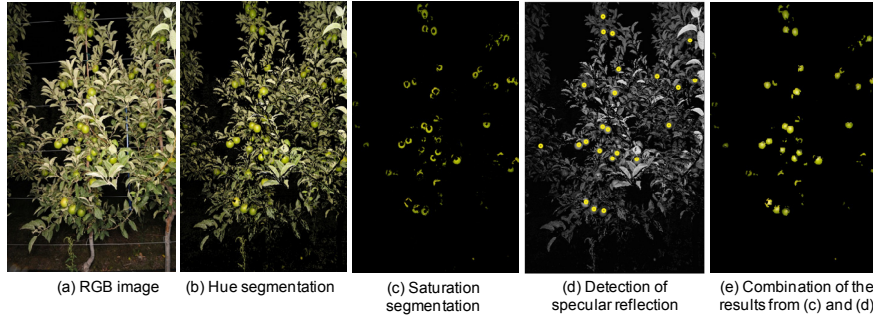
Fig. 3 (a) An image acquired by the crop yield estimation system in a green apple orchard. (b) The result of hue segmentation for green colors. (c) The result of saturation segmentation for the green color of apples. (d) The results of specular reflection detection (marked by circles). (e) The results of apple region detection using two visual cues: color and specular reflection.

The next step is to detect the apple regions with specular reflections. The rectangular area marked by dash lines in Fig. 4a is the minimum bounding rectangle of the apple area detected by the hue and saturation thresholding procedure, and extended by 25% to guarantee some apple pixels undetected by the saturation segmentation are included in the region. Fig. 4b is the light intensity (grayscale value) map of the rectangular region in Fig. 4a. The two apples have conical shape

in their intensity profile, which represents gradually descending light intensity from the peak to different directions. Compared to the apple regions, the foliage regions have more irregular intensity profiles. Based on these features, we use the shape of intensity profiles to detect specular reflections.

The algorithm detects specular reflections by searching for local maxima in the grayscale map (Fig. 4b). During the search, the size of the support neighborhood for the local maxima is 30% of the length of the short side of the rectangular region (Fig. 5a). The result of local maxima detection (Fig. 5a) includes the points with specular reflections, and also some points without specular reflections. To distinguish them, the algorithm checks the intensity profiles on four lines in a neighborhood (21 × 21 pixels) of each local maximum. As shown in Fig. 6, four lines go through a local maximum with slopes of 0°, 45°, 90°, and 135°, respectively. The intensity profiles around the specular reflection of an apple descend gradually from the local maximum to different directions (Fig. 6a). However, the intensity profiles around a local maximum, which is not a specular reflection, have irregular changes of grayscale values. Based on this difference, the algorithm uses the following procedure to decide whether a local maximum is a specular reflection. (1) It calculates the gradients of grayscale values between every two adjacent pixels on the four line segments. (2) It splits each line segment into two parts in the middle (at the local maximum). (3) It calculates a roundness score (R_i) by checking the signs of gradient in the eight parts. If the gradients in one part have the same sign, the roundness score of the part is one; otherwise, it is zero. (4) It calculates the total roundness score: $R = \sum_{i=1}^{8} R_i$. If $R \geq 4$, the local maximum is a specular reflection; otherwise, it is a false positive. The reason for using four rather than eight as a threshold is to make sure that the specular reflections on some partially-occluded apples can also be recognized. (5) It keeps specular reflection and removes false positives from the search region (Fig. 5b).

The algorithm overlies the results of saturation segmentation (Fig. 3c) with the pixels of specular reflection (Fig. 3d) and their surrounding neighborhoods (18 × 18 pixels). This combination yields a more complete detection of apple pixels in the image (Fig. 3e).
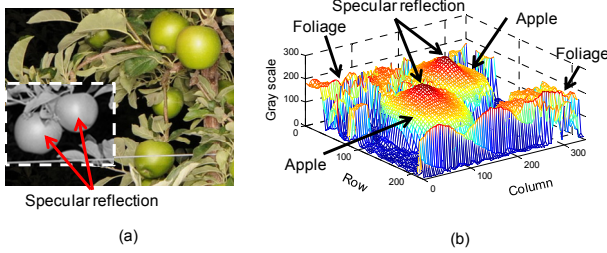

Fig. 4 (a) An example of specular reflection on the surface of apples. The rectangular region is transformed to grayscale image. (b) The mesh plot of the grayscale values of the rectangular region in (a).
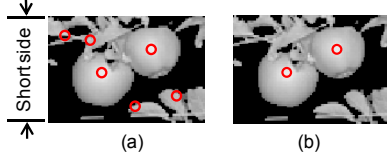
Fig. 5 (a) Local maxima (marked by circles) in a search region. (b) The results of detecting and removing local maxima that are not specular reflections.
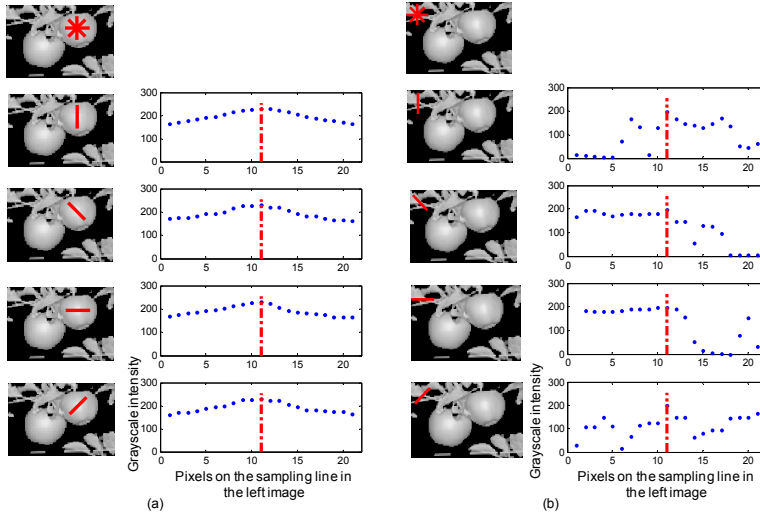

Fig. 6 (a) The intensity profiles around a local maximum, which is the specular reflection of an apple, in four directions. (b) The intensity profiles around a local maximum, which is not a specular reflection, in four directions.

3.3 Segmenting Individual Apples

The previous sections described how to detect the apple pixels in an image for both red and green apples. Here we describe morphological operations to convert these pixel regions into distinct individual apples.

Firstly, the software loads a binary image of apple regions. To realize apple counting, the software needs to determine the average diameter ($\overline{D}$) of apples in the loaded image. It calculates the eccentricity (E) of each apple region, and uses a threshold $0 < E < 0.6$ to find regions that are relatively round. These relatively round regions are usually the apples that have less occlusion and do not touch with other apples, which is convenient for determining apple diameter. A few small round regions are also detected. They are parts of some occluded apples and happen to be round in shape. Usually, they only account for a small portion of all the

relatively round regions. To remove the noise, the software calculates the area (S) of the relatively round regions and their average area ($\overline{S}$). It uses a threshold $S > \overline{S}$ to remove the noise with small area. Then, it calculates the length (in pixel) of the minor axis of the ellipse that has the same normalized second central moments as a remaining round apple region. The mean of the minor axis length of all the remaining regions is used as the average diameter of apples in the image.

Some apple regions contain two or more touching apples. The algorithm is able to detect them and split them into two apples. It calculates the length (L_{major}, in pixel) of the major axis of the ellipse that has the same normalized second central moments as a region. Any region with $L_{major} > 2\overline{D}$ is treated as a region with touching apples. It splits the major axis into two segments in the middle (at Point A in Fig. 7), and then uses the central points of each segment (Points B and C in Fig. 7) as the center of the two apples. It should be mentioned that the current version of software is designed to split this kind of region into only two apples. The design is based on the fact that apple clusters are usually thinned down to two apples in commercial orchards to keep the quality of individual apples in clusters. For orchards with larger clusters of apples we need to make improvements to deal with more apples per cluster.

Some apples are partially occluded by foliage or branches, and may be detected as multiple apple regions. The software calculates the distance between the centers (with the same definition as Point A in Fig. 7) of any two apple regions. Any pair with a distance less than $\overline{D}$ is treated as one apple. The midpoint of the two original centers is the new center of the apple.

In the end, the software records the locations of the centers of the remaining apple regions as the final detection of apples in the image.

4 Apple Registration from Multiple Images

Apple registration is required to deal with two critical issues in apple counting in an orchard. The first issue is to register apples that are detected multiple times. One apple can be seen up to seven times from one side of a tree row in the sequential images taken by the system. Some apples can be seen from both sides of a tree row. The second issue is to determine the apples that are detected from two opposite sides of a tree row, but belong to a same tree.

During continuous counting, the software detects apples in every frame of the image sequences taken by the two cameras of the stereo rig. The software applies the following procedure to calculate the global locations of detected apples, and use the locations to register apples.

Firstly, the software uses block matching to triangulate the 3D positions (in {C}) of all apples detected in the image sequences. The block matching is conducted in both ways between one pair of images taken by the binocular stereo rig.

When an apple detected in the lower image and an apple detected in the upper image are matches reciprocally, the software triangulates and records the 3D position (in $\{C\}$) of the apple center (in 2D image coordinates); otherwise, it discards the detection.

The software transforms apple locations from $\{C\}$ to $\{G\}$. The transformation provides the global coordinates for every detected apple.
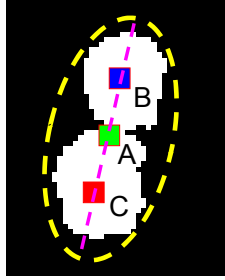

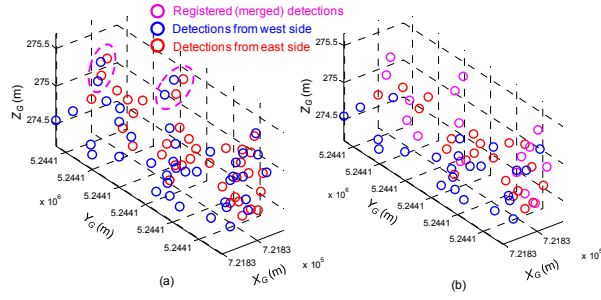**Fig. 7** The result of splitting a region with two touching apples

Fig. 8 (a) The apple detection results of three trees from two opposite sides of a row. (b) The results of registering apples detected from both sides.

The software merges the apples that are detected multiple times from one side of a tree row. It calculates the distance between every two apples in $\{G\}$, and then merges the apples with a distance less than 0.05 m from each other. The new location of the merged apple is obtained by averaging the locations of the multiple appearances of this apple. The software discards two kinds of detected apples as noise from the results: (1) apples that are detected only once in the sequential images; and, (2) dropped apples on the ground. Since the height of an apple on a tree is usually more than 0.3 m above the ground, apples with a height less than 0.3 m are treated as dropped apples.

The software uses global coordinates to register apples detected from both sides of a tree row, which requires precise positioning. However, we have noticed two issues in our apple positioning approach. First, when the vehicle returns on the opposite side of the row the GPS system has noticeably drifted in its reported height above ground. We have also noticed a bias in the stereo triangulation algorithm causing the apple location to be estimated closer to the camera. To solve these positioning problems, we calculate the GPS drift and stereo triangulation bias by locating objects on the orchard infrastructure, triangulating their position in world coordinates and repeating from the other side of the row. The error between the two reported locations of an object from each side of the row gives us a position correction term that we apply to the apple locations. The landmarks can be any stationary location such as the ends of posts, stakes, and wires. We used flagging tape that was placed every three trees and at present we manually record the landmark positions in the images, however this will be replaced in future iterations

of the system by an algorithm that can automatically detect the orchard infrastructure.

After correcting the apple locations, we merge the apples detected from both sides of the row. Fig. 8a shows an example in which some apples (marked by ovals) can be detected from both east and west sides of a row. To avoid double counting, the software calculates the distances between apples detected from one side and those detected from the other side. It merges apples within a distance of 0.16 m from each other. We use a lose threshold (0.16 m, about 2 times of apple diameter) to tolerate some errors of stereo triangulation. After this operation, the apples detected from both sides of a row are registered (Fig. 8b), and the software obtains a final apple count for the orchard.

5 Experiments and Results

The crop yield estimation system was deployed at the Sunrise Orchard of Washington State University, Rock Island, WA in September, 2011. The goals of the deployment are: (1) to evaluate the estimation accuracy of the current system, and (2) to discover issues that need to be improved for future practical applications.

5.1 Experimental Design

The experimental design includes four critical issues: apple variety, planting system of orchard, the area of orchard for data collection, and ground truth.

We selected two blocks of typical apple trees -- Red Delicious (red color) and Granny Smith (green color). These two popular commercial products are typical varieties in either red or green apples. Based on the suggestions of horticulturists, we selected "tall spindle" planting system (as shown in Fig. 1) for the field tests. The planting system is featured with high density of trees, thin canopy, and well-aligned straight tree row. It maximizes profitability through early yield, improved fruit quality, reduced spraying, pruning, and training costs. "Tall spindle" is being adopted by more and more apple growers, and is believed to be one of the major planting systems of apple orchard in the future, because of its ability to rapidly turn over apple varieties from those less profitable to those more profitable

The area of each block is about half acre. Specifically, there are 15 rows of red apple trees and 14 rows of green apple trees. Each row has about 48 trees. The ground truth of yield estimation is the human count conducted by professional orchard workers. Every tree row is split into 16 sections with three trees per section. The sections are marked by flagging tape that is used by the workers to count the number of apples in each section and likewise we force our algorithm to report the count for each section by manually marking the flags in the images. The numbers

are used as ground truth to evaluate the system's accuracy of crop yield estimation.

5.2 Results

The software processes the sequential images from the two blocks, and generates apple counts for every section (three trees). The results and analysis are presented as follows.

Fig. 9 shows the crop yield estimation and ground truth of the red apple block. In rows 1-10 that received regular fruit thinning, the computer count is close to the ground truth. The estimation errors of each row have a mean of -2.9%with a standard deviation of 7.1%. If we treat the 10 rows together, the estimation error is -3.2%. The numbers show that the crop estimation from the system is fairly accurate and consistent for rows 1-10. However, we undercount rows 11-15 by 41.3% because these trees were not fruit thinned (which would normally be conducted in most commercial orchards) leaving large clusters of apples. The larger clusters of apples cause two problems: (1) some apples are invisible and cannot be detected due to the occlusion caused by other apples nearby; (2) some fruit clusters consist of more than two apples, the current version of software can only split a cluster into two apples. Although the estimations per row are significantly below the ground truth, the standard deviation is small at 3.2%, which shows that the system performs consistently. Therefore, it is possible to calibrate the system, which will be discussed later.
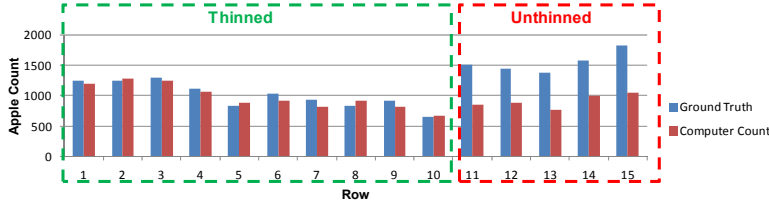

Fig. 9 Crop yield estimation and ground truth of the red apple block.

Similarly, the errors of the raw counts from the green apple block rows have a mean of -29.8% with a standard deviation of 8.1%. The trees in the green apple block have more foliage that those in the red apple block. The occlusion caused by foliage is thought to be the main reason for the undercount of green apples. Despite the large level of undercounting, the error is relatively consistent, making calibration for the raw counts possible. We perform calibration by selecting 10 random sections from the 224 sections in the green apple block, and conduct linear regression (with intercept = 0) between computer count and the ground truth (as shown in Fig. 10). The slope of the linear equation is the calibration factor. We run the method for 100 times, the average calibration factor is 1.4 with a standard

deviation of 0.1. The small standard deviation shows that the sample size of 10 sections is big enough to obtain a steady calibration. We use 1.4 as a calibration factor to correct the yield estimation in the green apple block. After the compensation (Fig. 11), the average yield estimation error at row level is 1.8% with a standard deviation of 11.7%. The compensated yield estimation error for the whole green apple block is 1.2%. We apply the same method to rows 11-15 in the red apple block, and find a calibration factor of 1.7. After the compensation, the average estimation error at row level is 0.4% with a standard deviation of 5.5%.
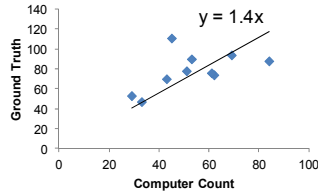

Fig. 10 A linear regression between computer counts for ten random green apple sections and the ground truth.
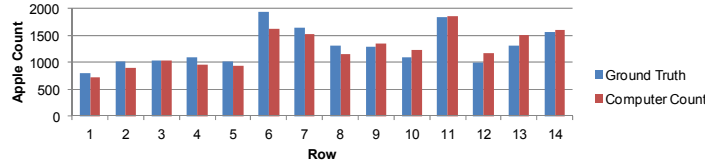

Fig. 11 Calibrated crop yield estimation and ground truth of the green apple block.

6. Discussion

The accuracy of the crop yield estimation system is subject to two major aspects: (1) how accurate it detects visible apples; (2) how accurate it estimates invisible apples. We discuss the two aspects in this section.

6.1 Detection Error of Visible Apples

In certain situations, the current software makes errors in detecting visible apples. For example, in our data set, the images of a few green apples are overexposed because these apples are much closer to the fleshes than the majority of the apples. As shown in Fig. 12a, a green apple in an overexposed image loses its original

color. A small amount of green apples have sunburn (red tint) on their skins (Fig. 12b). Green apples in an overexposed image or with sunburn cannot pass the (green) hue segmentation, and are undetectable in the image processing. New visual cues other than color should be considered in the future to deal with this problem. As mentioned earlier, the software has a limitation in dealing with fruit clusters comprised of more than two apples (Fig. 12c) and should be corrected in future iterations. False positive detections happen infrequently, but do occur in some situations as seen in Fig. 13. Future version of the software will look to reduce these false detections with a more strict set of image processing filters.
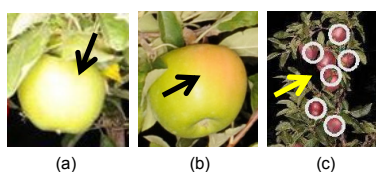

(a)   (b)   (c)

(a)   (b)   (c)

Fig. 12 Examples of visible apples that are not detected by the software. (a) An over-exposed image of a green apple. (b) A green apple with sunburn. (c) Apples missed in a large fruit cluster
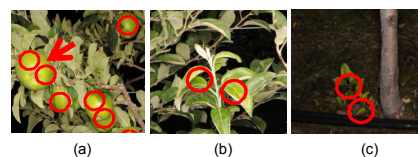
Fig. 13 Examples of false positive detections. (a) An apple with a short distance from the cameras. (b) Some new leaves with a color similar to green apples. (c) Weed with a color similar to green apples.

6.2 Calibration for Occluded Apples

The computer vision-based system cannot detect invisible apples that are occluded by foliage or other apples. As mentioned earlier, our solution is to calculate a calibration factor based on human sampling, and use the factor to predict the crop yield including invisible apples. The results show that the calibration method works well. In future work we will study the calibration procedure and evaluate accuracy versus sample size on larger orchard blocks. Too much sampling increases the cost of yield estimation; while, too less may harm the accuracy of estimation. We also will study if calibration factors can be used from prior years or from other orchards of similar varieties, similar ages and grown in similar styles.

6.3 Yield Maps

In addition to providing the grower with the total number of apples in an orchard, the system is also able to generate a yield map (Fig. 14) that provides information of the spatial distribution of the apples across the orchard. A yield map could be used for precision orchard management, enabling the grower to plan distribution

of fertilizer, irrigation, perform variable crop thinning, to improve operations by increasing efficiency, reducing inputs and increasing yield over time in underperforming sections.
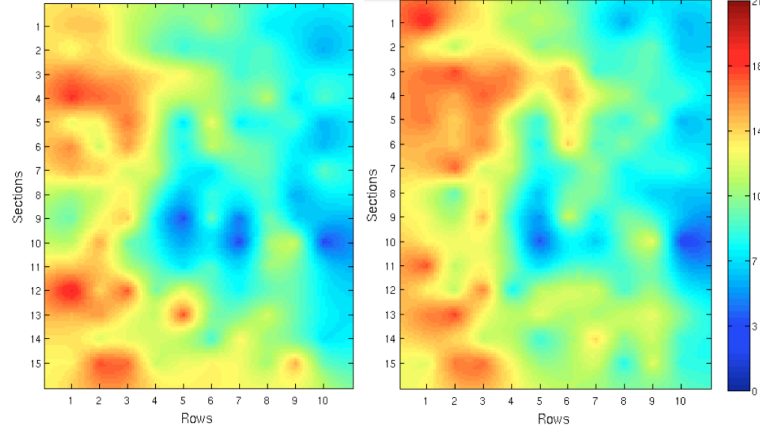

Fig. 14 High-resolution yield map representing spatial distribution of apples across the Red Delicious block. Left: Apple counts generated by our automated algorithm. Right: Ground truth apple counts. Our system provides a map that is a very close resemblance to the true state of the orchard**.**

7 Conclusions

Field tests show that the system performs crop yield estimation in an apple orchard with relatively high accuracy. In a red apple block with good fruit visibility, the error of crop yield estimation is -3.2% for about 480 trees. In a green apple block with significant fruit occlusion caused by foliage, we calibrate the system using a small sample of hand measurements and achieve an error of yield estimates of 1.2% for about 670 trees.

In future work we need to improve our system to deal with orchards with larger clusters of apples, which will require more precise and advanced methods to segment the apple regions within the images. We also improve the registration and counting algorithm to better merge apples detected from two sides of a row.

Acknowledgements

References

1. Jimenez AR, Ceres R, Pons JL (2000) A survey of computer vision methods for locating fruit on trees. Transactions of the American Society of Agricultural Engineers 43 (6):1911-1920

2. Swanson M, Dima C, Stentz A (2010) A multi-modal system for yield prediction in citrus trees. Paper presented at the 2010 Pittsburgh, Pennsylvania, June 20 - June 23, 2010,

3. Nuske S, Achar S, Bates T, Narasimhan S, Singh S Yield estimation in vineyards by visual grape detection. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011), 25-30 Sept. 2011, Piscataway, NJ, USA, 2011. 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011). IEEE, pp 2352-2358. doi:10.1109/iros.2011.6048830

4. Linker R, Cohen O, Naor A (2012) Determination of the number of green apples in RGB images recorded in orchards. Computers and Electronics in Agriculture 81 (0):45-57. doi:10.1016/j.compag.2011.11.007

5. Rakun J, Stajnko D, Zazula D (2011) Detecting fruits in natural scenes by using spatial-frequency based texture analysis and multiview geometry. Computers & Electronics in Agriculture 76 (1):80-88. doi:10.1016/j.compag.2011.01.007

6. Tabb AL, Peterson DL, Park J Segmentation of apple fruit from video via background modeling. In: 2006 ASABE Annual International Meeting, July 9, 2006 - July 12, 2006, Portland, OR, United states, 2006. 2006 ASABE Annual International Meeting. American Society of Agricultural and Biological Engineers, p ASABE

7. Safren O, Alchanatis V, Ostrovsky V, Levi O (2007) Detection of green apples in hyperspectral images of apple-tree foliage using machine vision. Transactions of the ASABE 50 (6):2303-2313

8. Stajnko D, Lakota M, Hočevar M (2004) Estimation of number and diameter of apple fruits in an orchard during the growing season by thermal imaging. Computers & Electronics in Agriculture 42 (1):31. doi:10.1016/s0168-1699(03)00086-3

9. Aggelopoulou A, Bochtis D, Fountas S, Swain K, Gemtos T, Nanos G (2011) Yield prediction in apple orchards based on image processing. Precision Agriculture 12 (3):448-456. doi:10.1007/s11119-010-9187-0

10. Hamner B, Bergerman M, Singh S Autonomous orchard vehicles for specialty crops production. In: American Society of Agricultural and Biological Engineers Annual International Meeting 2011, August 7, 2011 - August 10, 2011, Louisville, KY, United states, 2011. American Society of Agricultural and Biological Engineers Annual International Meeting 2011. American Society of Agricultural and Biological Engineers, pp 218-228

11. Bouguet JY (2008) Camera calibration toolbox for Matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/. Accessed 14 May 2012