A Computer Vision Algorithm for Omnidirectional Bee Counting at Langstroth Beehive Entrances

Vladimir Kulyukin
Department of Computer Science
Utah State University
Logan, UT, USA
vladimir.kulyukin@usu.edu

Sai Kiran Reka
Department of Computer Science
Utah State University
Logan, UT, USA
saikiran.reka@aggiemail.usu.edu

***Abstract*—A computer vision algorithm is proposed for omnidirectional bee counting in images of Langstroth beehive entrances captured in situ with a miniature camera connected to a multi-sensor, solar-powered electronic beehive monitoring device. The algorithm consists of three stages: pre-processing, landing pad identification, and omnidirectional bee counting. In the pre-processing stage, an approximate image region where the landing pad is likely to be is cropped and the brightness of the cropped image adjusted. The landing pad identification is obtained through iterative reduction of the cropped image to the actual landing pad. Omnidirectional bee counts are computed by dividing the total number of bee pixels by the average number of pixels occupied by individual bees. The algorithm was evaluated on 1,781 images from two electronic beehive monitoring devices deployed in Langstroth beehives with live honeybees and achieved an accuracy of over 80 per cent compared to the ground truth obtained from human evaluators.**

Keywords—computer vision; contour analysis; color analysis; electronic beehive monitoring; sustainable computing

I. Introduction

The Apis millifera, also known as the Western honeybee, is responsible for one out of every three daily mouthfuls that the average U.S. resident eats [1]. Since 2006 honeybees have been disappearing from amateur and commercial apiaries. This trend has been called the colony collapse disorder (CCD) [2]. Other growing threats to the health of honeybee colonies include Varroa mites, American and European foulbrood, and nosema [3].

The high rates of colony loss threaten the world's food supply chains and necessitate continuous beehive monitoring. Unfortunately, continuous beehive monitoring cannot be done by human apiarists due to obvious problems with logistics and fatigue. However, recent advances in sensor technologies have made it possible to monitor many critical variables associated with honeybee health in situ. There is an emerging consensus among researchers and practitioners that significant scientific and practical insights will likely come from transforming traditional apiaries into smart worlds monitoring their status through multiple sensors, recognizing bee behavior patterns, and notifying all interested parties about deviations and anomalies. For example, NASA researchers believe that climate changes can be investigated through pollination data [4], because beehive data clusters may relate location and pollination timings to satellite data and ecosystem models.

Most approaches to electronic beehive monitoring (EBM), defined here as electronic capture and analysis of data from beehives over regular time intervals, depend on the grid for power and on the cloud for data transmission (e.g., [5, 6]). However, grid- and cloud-dependent EBM enlarges the electricity consumption and carbon footprints of cloud data centers which already account for two percent of overall U.S. electrical usage [7, 8]. According to the Smart 2020 forecast by the Climate Group of the Global e-Sustainability Initiative [9], so far quite accurate, the global carbon footprint of cloud data centers is expected to grow, on average, 7% per annum between 2002 and 2020. In 2010, McAfee, a U.S. computer security company, reported that the electricity required to transmit the trillions of spam e-mails annually is equivalent to powering two million U.S. homes and generates the same amount of greenhouse gas emissions as that produced by three million cars [10]. Consequently, there is a critical need to seek ecologically sustainable EBM solutions that use renewable power sources and minimally depend on the cloud for data transmission and analysis.

Since electronic beehive monitoring devices (EBMDs) increasingly use multiple sensors [11], principled answers must be sought to the question of what sensors should be included and why. While sensor accuracy is a significant factor, power consumption, field reliability, and ergonomics must also be considered. The latter consideration is critical for the broader acceptance of a specific sensor by amateur and commercial apiarists, regionally, nationally, and internationally, because ease of deployment plays an important role in technology adoption.

The position presented in this paper is that computer vision can contribute to solving various problems posed by electronic beehive monitoring (EBM). In particular, computer vision can be used to solve the bee counting problem because of recent advances in miniature cameras coupled with small computational devices such as Raspberry Pi (www.raspberrypi.org) or Arduino (www.arduino.cc) that are powered with solar.

The bee counting problem, well-known in apiary science, is the problem of obtaining accurate counts of bees entering or exiting a given beehive per unit of time. One of the variables monitored by human apiarists as they inspect their beehives is forager traffic. Foraging is an indicator of honeybee colony health, colony age structure, honey flow, pollination, and climate (e.g., [12]). Consequently, accurate estimates of forager traffic levels are important not only to apiarists but also to growers, climate scientists, and sustainable farmers. Forager traffic levels can be estimated by human observers with stopwatches. However, since human observation cannot be continuous, abrupt changes in forager traffic will likely be missed.

The remainder of this paper is organized as follows. In Section II, related work is presented. In Section III, hardware and software details of BeePi®, a solar-powered, multisensor EBMD, are presented and in situ data collection is described. In Section IV, a vision-based algorithm is presented for omnidirectional bee counting on landing pads of Langstroth beehives used by most apiarists in the U.S. [2, 3]. In Section V, the experiments are presented of evaluating the algorithm on over 1,781 images captured by two deployed BeePi EBMDs in North Logan, UT. In Section VI, the results of the experiments are discussed. Section VII summarizes the results and the findings of this investigation.

II. Related Work

Because of the importance of forager traffic counts, there have been multiple research and commercial attempts to automate bee counting at hive entrances. One of the first electrical bee counters was proposed by Lundie [13]. Lundie's design was subsequently adopted and improved upon by Faberge [14] through the production of electrical impulses generated by bees tripping a balance arm. Similar electrical bee counters were proposed by other researchers (e.g., Ericson et al. [15] and Liu et al. [16]).

In earlier electrical bee counting devices, no distinction was made between counting bees entering and exiting a hive. This problem was addressed in subsequent research through bi-directional bee counters. For example, Struye et al. [17] proposed a design for a bi-directional bee counter. This design was adopted by Lowland Electronics in Belgium to manufacture bi-directional bee counters in the 1990s. These devices count bees passing through special portals equipped with infrared (IR) sensors. Bees are counted when they cross infrared beams.

Dank and Gary [18] designed a box-like extension fixed at the hive entrance to estimate the forager traffic. Each bee passes through special tubes in the box attached to the entrance. The tubes are coated with paraffin so that bees cannot only crawl through them. A mesh bag at the end of the tubes is used to collect the bees and weigh them. The bees can escape from the mesh bag.

Bromenshenk et al. [6] designed and deployed bi-directional, IR bee counters in their multi-sensor SmartHive® system. The researchers found their IR counters to be more robust and accurate than capacitance and video-based systems. Since the IR counters required regular cleaning and maintenance, a self-diagnostic program was developed to check whether all of the emitters and detectors were functioning properly and the bee portals were not blocked by debris or bees.

In addition to IR devices, some researchers used radio frequency identification (RFID) to solve the bee counting problem. For example, Schneider et al. [19] investigated pesticide effects on honeybee colonies by exposing workers from a colony of approximately 2,000 bees to contaminated sugar syrup at a feeder. The effects of pesticide exposure were measured as the RFID-detected return rate of foragers from the feeder.

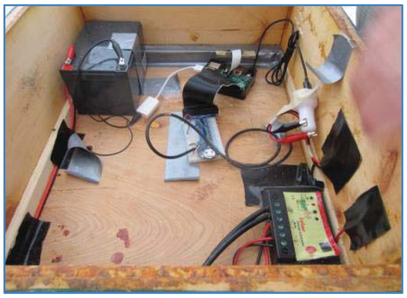

Figure 1. BeePi hardware components

III. In Situ Data Capture

A. Hardware

Images for this investigation were captured through a solar-powered, electronic beehive monitoring device (EBMD), called BeePi [20]. A fundamental objective of the BeePi design is reproducibility: other researchers and practitioners should be able to replicate our results at minimum cost and time commitments. The current BeePi hardware components are shown in Fig. 1: a Pi Model B+ 512MB RAM computer, a Pi T-Cobbler, a half-size breadboard, a DS18B20 temperature sensor, and a Pi camera. For solar harvesting, the Renogy 50 watts 12V monocrystalline solar panel was coupled with the Renogy 10 Amp PWM solar charge controller and the UPG 12V 12Ah F2 sealed lead acid AGM deep-cycle rechargeable battery.

All hardware components fit in a shallow Langstroth super, except for the solar panel that is placed on top of a beehive (see Fig. 2). The solar panels are tied to the hive supers with bungee cords. The Pi camera is placed outside to take static snapshots of the beehive's entrance, as shown in Fig. 3, with a plastic cover placed above it to protect it from the elements.

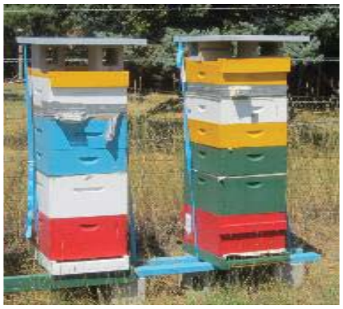

Figure 2. Solar panels on hive tops

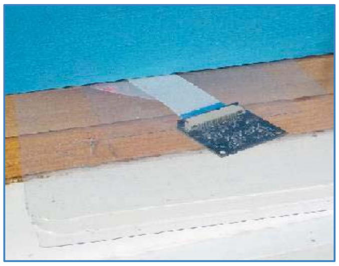

Figure 3. Pi camera looking down on landing pad

ISBN: 1-60132-442-1, CSREA Press ©

Four BeePi EBMDs were assembled in 2015 and deployed at two Northern Utah apiaries to collect 28GB of audio, temperature, and image data in different weather conditions [20]. Except for drilling narrow holes in inner hive covers for temperature sensor and microphone wires, no structural beehive modifications were done to the hives prior to deployment.

B. Software

All data collection is done in situ on the raspberry pi computer. The collected data are saved on a 16GB sdcard inserted into the pi computer. In situ data collection software is written in Python 2.7.

When the system starts, three data collection threads are spawned. The first thread collects temperature readings every 10 minutes and saves them in a text log. The second thread collects 30-second wav recordings every 15 minutes. The third thread saves PNG pictures of the beehive's landing pad every 15 minutes. The size of the image captured for the camera is 550KB with a resolution of 720 × 480 pixels.

A cronjob, i.e., an automated task that runs at specific intervals, monitors the threads and restarts them after hardware or software failures. For example, during a field deployment the camera of one of the EBMDs stopped functioning due to excessive heat. The cronjob kept periodically restarting the picture thread until the temperature went down and the camera started functioning properly again.

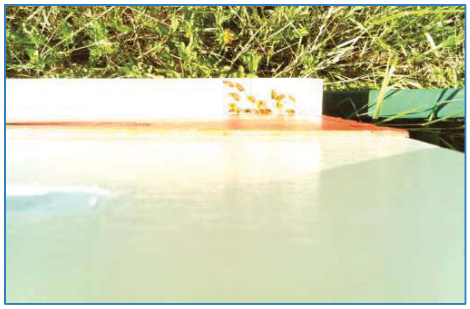

Figure 4. Sample captured image

IV. Vision-Based Bee Counting

The vision-based bee counting algorithm is omnidirectional in that it does not distinguish incoming and outgoing bee traffic. The reason why no directionality is integrated is two-fold. First, a robust vision-based solution to directionality will likely require video processing. Since the BeePi relies exclusively on solar power, in situ video capture and storage will reduce device operation times and make EBM less continuous. Second, omnidirectional bee counting can still be used as a valuable estimate of forager traffic so long as it accurately counts bees on landing pads.

The algorithm is implemented in JAVA with JDK 1.7 and the OpenCV 2.4.4 (www.opencv.org) bindings. The algorithm consists of three stages: pre-processing, landing pad identification, and omnidirectional bee counting. In the pre-processing stage, an approximate image region where the landing pad is likely to be is cropped and the brightness of the cropped region adjusted. The landing pad identification is obtained through iterative reduction of the cropped image to the actual landing pad. Omnidirectional bee counts are computed by dividing the total number of bee pixels by the average number of pixels occupied by individual bees obtained from camera calibration experiments.

A. Pre-Processing

Several in situ camera calibration experiments were conducted to estimate the coordinates of the image region where the landing pad is likely to be. The coordinates of the region are set in a configuration file and used in the algorithm to crop the region of interest. The lower image in Fig. 5 shows the output of the cropping step. Note that there may be some grass in the cropped image. The dimensions of the cropped region are intentionally set to be larger than the actual landing pad to compensate for camera swings in strong winds.

Image brightness varies greatly with the weather. When the sun is directly above the beehive, brightness is maximal. However, when the sun is obscured by clouds, captured images tend to be darker. Both cases have a negative impact on bee counting. To compensate for these two conditions, image brightness is dynamically adjusted to lie in (45, 95), i.e., the brightness index should be greater than 45 but less than 95. This range was experimentally found to yield optimal results. Fig. 6 illustrates how brightness adjustment improves omnidirectional bee counts. The upper image on the right in Fig. 6 shows a green landing pad extracted from the cropped image on the left without adjusted brightness. The lower image on the right in Fig. 6 shows a green pad extracted from the same image with adjusted brightness. Only four bees were identified in the upper image on the left whereas in the lower image eight bees were identified, which is closer to the twelve bees found in the original image by human counters.

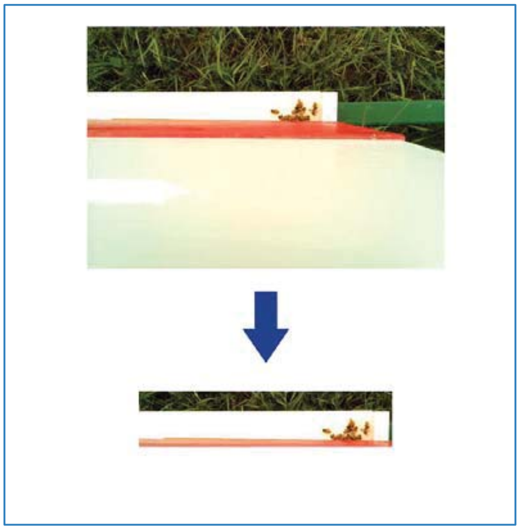

Figure 5. Cropping a landing pad region

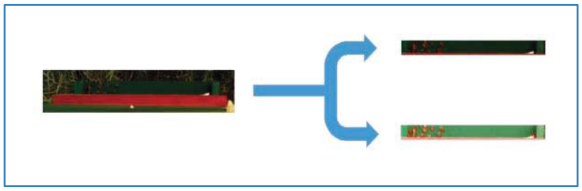

Figure 6. Adjusting image brightness

B. Landing Pad Identification

The three steps of the landing pad identification are shown in Fig. 7. The first step in identifying the actual landing pad in the approximate region cropped in the pre-processing stage is to convert the pre-processed RGB image to the Hue Saturation Value (HSV) format, where *H* and *S* values are computed with Equation (1).

$$H = \begin{cases} 60^\circ\left(\frac{G'-B'}{\Delta}\ mod\ 6\right) & if\ C_{max} = R', \\ 60^\circ\left(\frac{B'-R'}{\Delta} + 2\right) & if\ C_{max} = G', \\ 60^\circ\left(\frac{R'-G'}{\Delta} + 4\right) & if\ C_{max} = B'. \end{cases} \qquad (1)$$

$$S = \begin{cases} 0 & if\ C_{max} = 0, \\ \frac{\Delta}{C_{max}} & if\ C_{max} \neq 0. \end{cases}$$

In (1), R', G', B' are the *R*, *G*, *B* values normalized by 255, $C_{max} = max\{R', G', B'\}$, and $\Delta = C_{max} - min\{R', G', B'\}$. The value of *V* is set to C_{max}. In the actual implementation, the format conversion is done with the cvtColor() method of OpenCV. The inRange() method of OpenCV is subsequently applied to identify the areas of green or white, the two colors in which the landing pads of our beehives are painted. Noise is removed through a series of erosions and dilations. The white pixels in the output image represent green or white color in the actual image and the black pixels represent any color other than green or white.

To further remove noise from the image and reduce it as closely as possible to the actual landing pad, contours are computed with the findContours() method of OpenCV and a bounding rectangle is found for each contour. The bounding contour rectangles are sorted in increasing order by the *Y* coordinate, i.e., increasing rows. Thus, the contours in the first row of the image will be at the start of the list. Fig. 7 shows the bounding rectangles for the contours computed for the output image of step 3 in Fig. 7.

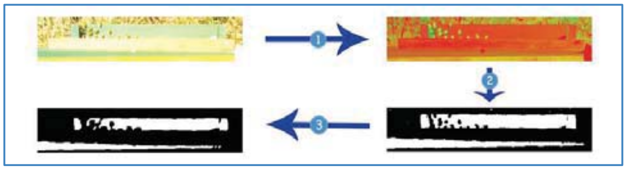

Figure 7. Landing pad identification steps: 1) HSV conversion; 2) color range identification; 3) noise removal

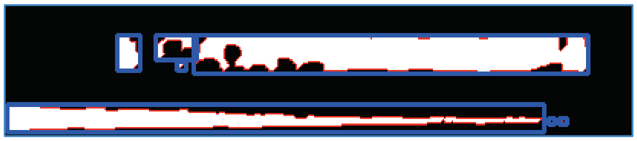

Figure 8. Bounding rectangles of found contours

Data analysis indicates that if the area of a contour is at least half the estimated area of the landing pad, the contour is likely to be part of the actual landing pad. On the other hand, if the area of a contour is less than 20 pixels, that contour is likely to be noise and should be discarded. In the current implementation of the algorithm, the estimated area of the green landing pad is set to 9,000 pixels and the estimated area of the white landing pad is set to 12,000 pixels. These parameters can be adjusted for distance.

Using the above pixel area size filter, the approximate location of the landing pad is computed by scanning through all the contours in the sorted list and finding the area of each contour. If the area is at least half the estimated size of the landing pad of the appropriate color, the *Y* coordinate of the contour rectangle is taken to be the average *Y* coordinate and the scanning process terminates. If the contour's area is between 20 and half the estimated landing pad area, the *Y* coordinate of the contour is saved. Otherwise, the current contour is skipped and the next contour is processed. When the first contour scan terminates, the average *Y* coordinate, $\mu(Y)$, is calculated by dividing the sum of the saved *Y* coordinates by the number of the processed contour rectangles.

After $\mu(Y)$ is computed, a second scan of the sorted contour rectangle list is performed to find all contours whose height lies in $[\mu(Y) - H, \mu(Y) + H]$, where *H* is half of the estimated height of the landing pad for the appropriate color. While the parameter *H* may differ from one beehive to another, as the alignment of the camera differs from one hive to another, it can be experimentally found for each beehive. For example, if the camera is placed closer to the landing pad, then *H* will have a higher value and if the camera is placed far from the landing pad, *H* will have a lower value. In our case, *H* was set to 20 for green landing pad images and to 25 and for white landing pad images.

A bounding rectangle is finally computed after the second scan to enclose all points in the found contours. To verify whether the correct landing pad area has been identified, the area of the bounding rectangle is computed. If the area of the bounding rectangle is greater than the estimated area of the landing pad, the bounding rectangle may contain noise, in which case another scan is iteratively performed to remove noise by decreasing *H* by a small amount of 2 to 4 units. In most of the cases, this extra scan is not needed, because the landing pad is accurately found. Fig. 9 illustrates the three steps of the contour analysis to identify the actual landing pad.

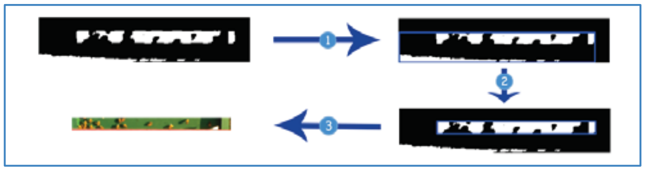

Figure 9. Contour analysis: 1) 1st contour scan; 2) 2nd contour scan; 3) pad cropping

Foreground and background pixels are separated on color. In particular, for green landing pads, the background is green and the foreground, i.e., the bees, is yellow; for white landing pads, the background is white and the foreground is yellow. All pixels with shades of green or white are set to 255 and the remaining pixels are set to 0. Three rows of border pixels of the landing pad image are arbitrarily set to 255 to facilitate bee identification in the next step. Fig. 10 shows the output of this stage. In Fig. 10, the green background is converted to white and the foreground to black. Since noise may be introduced, the image is de-noised through a series of erosions and dilation with a 2 x 2 structuring element.

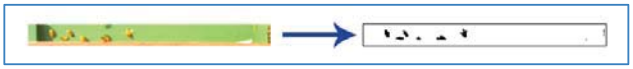

Figure 10. Background and foreground separation

C. Bee Counting

To identify bees in the image, the image from the previous stage is converted to grayscale and the contours are computed again. Data analysis suggests that the area of an individual bee or a group of bees vary from 20 to 3,000 pixels. Therefore, if the area of a contour is less than 20 pixels or greater than 3,000 pixels, the contour is removed.

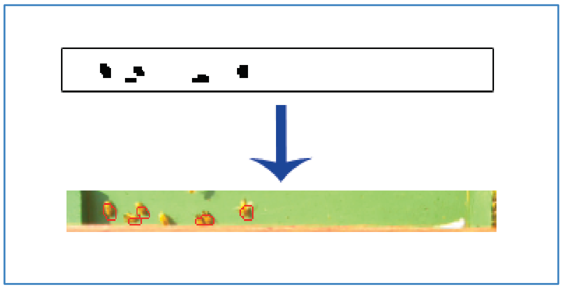

Figure 11. Omnidirectional bee counting

The area of one individual bee is between 35 and 100 pixels, depending on the distance of the pi camera from the landing pad. The green landing pad images were captured by a pi camera placed approximately 1.5m above the landing pad with the average area of the bee being 40 pixels. On the other hand, the white landing pad images were captured by a pi camera placed approximately 70cm above the landing pad where the average area of an individual bee is 100 pixels. To find the number of bees in green landing pad images, the number of the foreground pixels, i.e., the foreground area, is divided by 40 (i.e., the average bee pixel area on green landing pads), whereas, for the white landing pad images, the foreground area is divided by 100 (i.e., the average bee pixel area on white landing pads). The result is the most probable count of bees in the image. In the upper image in Fig. 11, five bees are counted by the algorithm. The lower image in Fig. 11 shows the found bees in the original image.

V. Experiments

A sample of 1,005 green pad images and 776 white pad images were taken from the data captured with two BeePi EBMDs deployed at two Northern Utah apiaries [20]. Each image has a resolution of 720 x 480 pixels and takes 550KB of space. To obtain the ground truth, six human evaluators were recruited. Each evaluator was given a set of images and asked to count bees in each image and record his or her observations in a spread sheet. The six spread sheets were subsequently combined into a single spread sheet.

Table I gives the ground truth statistics. The human evaluators identified a total of 5,770 bees with an average of 5.7 bees per image in images with green landing pads. In images with white landing pads, the evaluators identified a total of 2,178 bees with a mean of 2.8 bees per image.

Table II summarizes the performance of the algorithm ex situ on the same green and white pad images. The algorithm identified 5,263 bees out of 5,770 in the green pad images with an accuracy of 80.5% and a mean of 5.2 bees per image. In the white pad images, the algorithm identified 2,226 bees out of 2,178 with an accuracy of 85.5% and an average of 2.8 bees per image. The standard deviations of the algorithm were slightly larger than those of the human evaluators.

Table I. Ground Truth

Pad Color	Num Images	Total Bees	Mean	STD
Green	1,005	5,770	5.7	6.8
White	776	2,178	2.8	3.4

Table II. Accuracy (%) of the Algorithm

Pad Color	Num Images	Total Bees	Mean	STD	ACC
Green	1,005	5,263	5.2	7.6	80.5
White	776	2,178	2.8	4.1	85.5

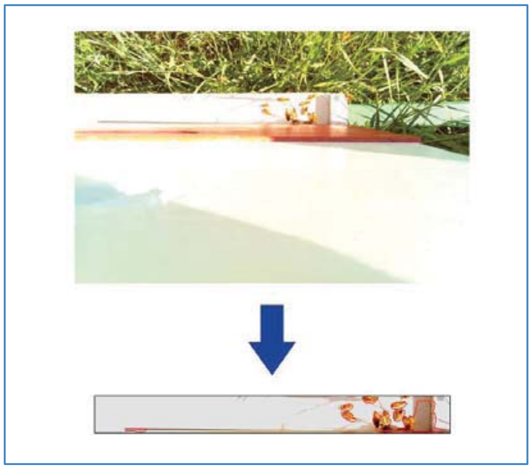

Figure 12. False positives

VI. Discussion

Our analysis of the results identified both true negatives and false positives. There appear to be fewer true negatives than false positives. The main reason for true negatives is the algorithm's conservative landing pad identification, which causes some actual bees to be removed from the image. The bees on the sides of the landing pad are also typically removed from the image. Another reason for true negatives is image skewness due to wind induced camera swings. If the landing pad is skewed, then a part of the landing pad is typically cropped out during the bounding rectangle computation. In some images, some actual bees were removed from images during image de-noising, which resulted in lower bee counts compared to human counts.

False positives were primarily caused by occasional shades, leaves, or blades of grass wrongly counted as bees. Fig. 12 gives an example of false positives. A human evaluator counted 9 bees in the upper image whereas the algorithm counted 28 bees on the landing pad (lower image in Fig. 12) cropped out of the upper image. The shade pixels on the right end of the cropped landing pad were counted as bees, which resulted in a much higher bee count than the human evaluator's count.

VII. Summary

A computer vision algorithm was presented for omnidirectional bee counting at Langstroth beehive entrances. The algorithm was evaluated on a total of 1,781 images and achieved an accuracy of over 80 per cent compared to the ground truth obtained from six human evaluators. The performance of the algorithm can be further improved through a more accurate identification of the landing pad's skew and image rotation before actual bee counting.

Acknowledgment

The first author expresses his gratitude to Neil Hattes who jumpstarted this project by donating a raspberry pi computer, a mother board, a temperature sensor, and a monitor. The in situ data collection software was originally developed and tested on this equipment. All bee packages and beehive woodenware used in this study were funded by the first author.

References

[1] Walsh, B. "A world without bees." Time, pp. 26-31, August 19, 2013.

[2] Conrad, R. *Natural beekeeping: organic approaches to modern apiculture*. Chelsea Green Publishing, White River Junction, Vermont, 2013.

[3] Crowder, L. & Harrell, H. *Top-bar beekeeping: organic practices for honeybee health*. Chelsea Green Publishing, White River Junction, Vermont, 2012.

[4] NASA HoneyBeeNet Project. http://honeybeenet.gsfc.nasa.gov/.

[5] Meikle, W.G. & Holst, H. "Application of continuous monitoring of honeybee colonies." Apidologie, vol. 46, pp. 10-22, 2015.

[6] Bromenshenk, J.J., Henderson, C.B., Seccomb, R.A., Welch, P.M., Debnam, S.E., & Firth, D.R. "Bees as biosensors: chemosensory ability, honey bee monitoring systems, and emergent sensor technologies derived from the pollinator syndrome." Biosensors, vol. 5, pp. 678-711, 2015.

[7] Walsh, B. "Your data is dirty: the carbon price of cloud computing." Time, April 2, 2014.

[8] Vaugh, A. "How viral cat videos are warming the planet." The Guardian, Sept. 25, 2015.

[9] "Smart2020: Enabling the low carbon economy in the information age." The Global e-Sustainability Initiative. Avail. at http://www.smart2020.org.

[10] Berners-Lee, M. & Clark, D. "What's the carbon footprint of … email?" The Guardian, October 7, 2010.

[11] McNeil, M.E.A. "Electronic beehive monitoring." American Bee Journal, August 2015, pp. 875 – 879.

[12] Gary, N.E. Chapter 8. "Activities and behavior of honey bees." In: Graham, J.M. (ed.) The hive and the honey bee, pp. 269–372, 1992.

[13] Lundie, A. E. "The flight activities of the honey bees", United States Department of Agriculture, Dept. Bull. No. 1328, 1925.

[14] Faberge, A.C. "Apparatus for recording the number of bees leaving and entering a hive". J. Sci. Instr., vol. 20, pp. 28–311, 1943.

[15] Erickson, E.H., Miller, H.H., & Sikkema, D.J. "A method of separating and monitoring honey-bee flight activity at the hive entrance". J. Apic. Res., vol. 14, pp. 119–125, 1975.

[16] Liu, C., Leonard, J., & Feddes, J.J. "Automated monitoring of flight activity at a beehive entrance using infrared light sensors". J. Apic. Res., vol. 29(1), pp. 20–27, 1990.

[17] Struye, M.H., Mortier, H.J., Arnold, G., Miniggio, C., & Borneck, R. "Microprocessor-controlled monitoring of honeybee flight activity at the hive entrance". Apidologie, vol. 25, pp. 384–395, 1994.

[18] Danka, R. G., & Gary, N.E. "Estimating foraging populations of honey bees (Hymenoptera: Apidae) from individual colonies." Journal of economic entomology, vol 80.2, pp. 544-547, 1987.

[19] Schneider, C.W., Tautz, J., Grünewald, B., & Fuchs, S. "RFID Tracking of sublethal effects of two neonicotinoid insecticides on the foraging behavior of Apis mellifera". PLoS ONE, vol. 7(1), e30023. doi:101371/journal.pone.0030023, 2012.

[20] Kulyukin, V., Putnam, M., & Reka, S. K. "Digitizing buzzing signals into A440 piano note sequences and estimating forage traffic levels from images in solar-powered, electronic beehive monitoring." In Edtrs. S. I. Ao, Oscar Castillo, Craig Douglas, David Dagan Feng, and A. M. Korsunsky, *Proceedings of International MultiConference of Engineers and Computer Scientists* (IMECS 2016): *International Conference on Computer Science*, vol. I, pp. 82-87, March 16-18, Kowloon, Hong Kong, IA ENG, ISBN: 978-988-19253-8-1;ISSN: 2078-0958, 2016.