5-2016

A Vision-Based Bee Counting Algorithm for Electronic Monitoring of Langsthroth Beehives

Sai Kiran Reka
Utah State University

A VISION-BASED BEE COUNTING ALGORITHM FOR ELECTRONIC MONITORING OF LANGSTHROTH BEEHIVES

by

Sai Kiran Reka

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Computer Science

Approved:

_____	_____
Dr. Vladimir Kulyukin Major Professor	Dr. Dan Watson Committee Member
_____	_____
Dr. Nicholas Flann Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2016

ABSTRACT

A Vision-Based Bee Counting Algorithm for Electronic Monitoring of Langsthroth Beehives

by

Sai Kiran Reka, Master of Science

Utah State University, 2016

Major Professor: Dr. Vladimir Kulyukin
Department: Computer Science

An algorithm is presented to count bee numbers in images of Langsthroth hive entrances. The algorithm computes approximate bee counts by adjusting the brightness of the image, cropping a white or green area in the image, removing the background and noise from the cropped area, finding the total number of bee pixels, and dividing that number by the average number of pixels in a single bee. On 1005 images with green landing pads, the algorithm achieved an accuracy of 80 percent when compared to the human bee counting. On 776 images with white landing pads, the algorithm achieved an accuracy of 85% compared to the human bee counting.

(46 pages)

PUBLIC ABSTRACT

A Vision-Based Bee Counting Algorithm for Electronic Monitoring of Langsthroth Beehives

by

Sai Kiran Reka, Master of Science

Utah State University, 2016

Major Professor: Dr. Vladimir Kulyukin
Department: Computer Science

Forager traffic is the number of bees entering or exiting the bee hive over a given period of time. To estimate forager traffic, beekeepers typically take a stopwatch and count the number of bees manually over a set period of time. Forager traffic is an important health indicator of the hive. Sudden changes in forager traffic levels show that there may be an anomaly inside the hive or in the environment. There have been several attempts to automate the estimation of forager traffic in Langsthroth hives with custom-built hardware devices. In this thesis, we argue that bee counting can be approached as a software problem. Specifically, an algorithm is proposed to estimate forager traffic using computer vision. The images are captured by a raspberry pi camera which is a part of our BeePi Electronic Bee Hive Monitor that fits in a shallow super on top of the Langsthroth hive.

ACKNOWLEDGMENTS

I would like to use this opportunity to express my deepest gratitude to my advisor, Dr. Vladimir Kulyukin, who had supported me throughout my research, encouraged me to think of possible solutions, supported and guided me during hard times. His experience and knowledge are secret ingredients to the success of the project.

I acknowledge my gratitude to my committee members, Dr. Dan Watson and Dr. Nicholas Flann for continuous support, reading my reports, commenting on my views and helping me understand difficult concepts.

Last but not least, I would like to thank my beloved family and friends for cheering me up and encouraging me to cross the hurdles and succeed in my academic pursuit.

Sai Kiran Reka

CONTENTS

Page

LIST OF FIGURES

Figure Page

CHAPTER 1

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]. Their overall health can be considered as an indicator of health of our eco-system. Since 2006 honeybees have been disappearing from amateur and commercial apiaries. This trend has been called the colony collapse disorder (CCD) [1]. According to an annual survey conducted by the Bee Informed Partnership and Apiary Inspectors of America [2], professional and amateur apiarists across the United States lost almost 40 percent of their beehives between April 2014 and April 2015.

Since seventy percent of the world's food production is dependent on pollination and major part of pollination is done by honeybees, such heavy honeybee colony losses threaten the world's food supply. The CCD is not the only malady affecting the Western honeybee. Other growing threats include Varroa mites [3], the deformed wing virus [4], American and European foulbrood [5, 6], and nosema [7].

A consensus is emerging among researchers and practitioners that electronic beehive monitoring (EBM) can help extract critical information on colony behavior and phenology without invasive beehive inspections [8]. Broadly speaking, the EBM is based on the assumption that beehives can be equipped with sensors to collect data on various critical variables such as forager traffic levels, temperature, audio, weight, etc. The collected data can be analyzed for patterns to predict anomalies and notify all relevant

parties about them. For example, NASA researchers believe that climate changes can be investigated through pollination data [9], because beehive data clusters may relate location and pollination timings to satellite data and ecosystem models.

One possible approach to the EBM is to leverage regional and national interests in beekeeping and turn it into a large citizen science project for amateur and professional apiarists to collect, analyze, annotate, and share sensor data from a network of deployed EBM devices (EBMDs). There are two fundamental issues that must be addressed: how the EBMDs are powered and what sensors EBMDs should have.

Under ideal circumstances, apiaries should be monitored continuously. Continuous EBM can be achieved by putting monitored apiaries or individual beehives on the grid. Unfortunately, making the EBM grid-dependent will enlarge the electricity consumption and carbon footprint of cloud computing. Data centers already account for 2 percent of overall U.S. electrical usage [10]. While there are R&D efforts, e.g., liquid cooling [11], due to pressure from various environmental groups, data center locations are selected for inexpensive electricity. For example, in 2010 a data center commissioned by Facebook signed a power agreement with PacificCorp, a utility company that generates the majority of its energy from coal-fired power stations [12].

It is not possible to provide infrastructure to carry electricity to remote areas where beehives are deployed. And, the beehives are not fixed to a particular spot. Beekeepers move the hives around a region for the bees to collect the honey. To make the hives mobile and get the hives connected to the grid, the bee yards should have power connectivity for every few meters. It takes lot of hardware and time to lay the framework.

According to the 2015 report by Greenpeace [13], the percentages of dirty (coal and nuclear) energy in the energy footprints of many U.S. corporations are significant: Google – 34%; Facebook – 39%; Microsoft – 40 %; IBM – 47%; Oracle – 61%; Amazon – 53%. While Apple claims to be 100% renewable, OS X accounts only for 9% of laptops and desktops and 11% of mobile devices worldwide. A recent trend among major cloud computing providers is not to report the components of their energy footprints. The Smart 2020 analysis forecast has predicted, so far quite accurately, that the global carbon footprint of the data centers and telecommunication networks will grow, on average, 7% and 5%, respectively, between 2002 and 2020 [14]. Consequently, it is of vital importance to seek renewable sources of power for the EBM.

The power consumed by raspberry pi per hour in idle mode without any I/O connected is 1.21 w. One sensor attached to the USB of the pi draws around 0.3 w per hour [37]. We have used 3 sensors, so approximately these sensors consume 0.9 w per hour. Any software running on the pi consumes more power and approximately it comes to 4 w per hour for raspberry pi B+ model computer. Running the pi 24 hours a day totals to 96 Watts and for a month it is 2880 w or 2.9 kw. On an average the cost a 1 kw in US is 12 cents [38]. So, we end up paying 36 cents per month. Generating 1kw power would release 2 pounds of carbon dioxide [39]. US produces 163,000,000 pounds of honey every year [35]. On an average each hive produces 30 – 60 pounds of honey every year [36]. So we can assume that there are around 3 million hives in US, approximating that each hive produces 50 pounds of honey. If we place those hives on grid, we consume 9 million kW power per month and if the power is generated from dirty energy resources,

we produce 18 million pounds of carbon dioxide per month and release it into environment

Since EBMDs are ad hoc sensor networks, principled answers must be found on what sensors should be included in these networks. While sensor accuracy is a significant factor, power consumption, reliability, and ergonomics must also be considered. The latter consideration, i.e., ergonomics, is almost completely overlooked in the EBM literature despite the fact that the ease of field deployment is critical for the broader acceptance of a specific sensor by all interested parties, especially beekeepers, regionally, nationally, and internationally. The position presented and argued in this thesis is that computer vision can contribute to the off-the-grid EBM. Specifically, it can contribute to solving the bee counting problem, a well-known problem, to which solutions can be used to estimate forager traffic.

The remainder of this paper is organized as follows. In Section II, related work is presented. In Section III, a vision-based algorithm is presented for counting bees on landing pads of Langstroth beehives. In Section IV, the experiments conducted on images are discussed. In Section V, the results are presented after evaluating the algorithm on over 1,000 images with Green Landing pad and 776 images with White landing pad, captured by two EBMDs deployed in the fields. In Section VI, the findings are summarized and conclusions are presented.

CHAPTER 2

RELATED WORK

The past several years have seen a growing interest in the EBM as members of the worldwide beekeeping community begin to realize that the zero tolerance mentality of the industrial model of beekeeping and its increasing demand for stronger chemical treatments may be insufficient to prevent the loss of bee colonies [15]. The expert panel discussion at the recent 2nd International Workshop on Hive and Bee Monitoring [16] concluded that there appears to be a potentially large, worldwide market for beehive monitoring technologies in the next decade. The current technological challenges are the use of renewable energy sources to power beehive monitors, the use of off-the-shelf hardware components to reduce assembly costs and to increase reliability and maintainability; and lack of data exchange standards and protocols due to competing commercial interests.

One of the symptoms beekeepers observe as they inspect their hives is forager traffic. Since foraging can be used as an indicator of colony health, colony age structure, honey flow, pollination, and climate [17, 18, 19], its accurate estimates are important not only for beekeepers but also for growers, climate scientists, and sustainable farmers. Forager traffic counts, i.e., numbers of bees entering or exiting a given hive over a given time interval, can be collected by human observers with stopwatches. The use of human observation cannot be continuous due to significant logistical requirements and fatigue, which may result in failure to detect abrupt bee traffic changes.

Because of the importance of forager traffic counts, there have been multiple research and commercial attempts to automate hive entrance bee counting. Lundie (1925) proposed an electrical bee counter [20]. Lundie's design was subsequently adopted and improved upon by Faberge (1943) proposed through the production of electrical impulses generated by bees tripping a balance arm [21]. Other researchers (e.g., Erickson et al. (1975) [22] and Liu et al. (1990)) [23] also proposed electrical bee counters.

Dank and Gary (1987) designed a box like extension fixed at the hive entrance to estimate the forager traffic. Each bee is passed through tubes in the box attached to the entrance. The tubes are coated with paraffin so that bees cannot fly and only allowed to crawl. A mesh bag at the end of the tubes is used to collect all the bees and weigh them. The bees are allowed to escape from the trap. This process requires bees to learn ways to enter the hive and exit the trap. [24]

A design for a bi-directional bee counter was proposed by Struye et al. and adopted by Lowland Electronics in Belgium to manufacture bi-directional bee counters in the 1990s [25]. The Lowland Electronics counters count bees passing through portals equipped with infrared sensors. A bee is counted when it crosses an infrared beam.

Several designs of hive entrance counters have been proposed that use the RFID technology. Schneider et al. (2012) [26] used RFID sensors to investigate sublethal pesticide effects on bee colonies by exposing workers from a small colony of about 2,000 bees to contaminated sugar syrup at a feeder. The effects of pesticide exposure were measured as the return rate of foragers from the feeder.

Marc, Godfried and Frans designed a bee traffic estimator using light emitting diodes and phototransistors. The bees are forced to pass through a tunnel. At the other end of the tunnel there is a LED and a phototransistor. If the bee passes through the light, it interrupts the light stream and the phototransistor detects the break and counts it as a bee. [27]

Bromenshenk et al. [28] designed and deployed bi-directional, infrared 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 king and the bee portals were not blocked by debris or bees.

Several commercial systems (e.g., Arnia [29] and HiveMind [30]) now offer an option for remote video hive monitoring. Arnia provides hive data acquisition in individual hives and access to the data through Internet-enabled devices. Arnia currently focuses on backyard beekeepers and small research projects.

CHAPTER 3

BEE COUNTING ALGORITHM

3.1 Overview

In this chapter, we describe our bee counting algorithm. The algorithm can be divided into three stages: pre-processing, landing pad identification, and bee counting. The pre-processing stage consists of cropping a part of the image where the landing pad is likely to be and adjusting the brightness of the image. The landing pad identification stage includes looking for the white or green area in the pre-processed image, cropping it out, and removing the background and the noise from the cropped area. The bee counting stage is finding the total number of bee pixels and dividing that number by the average number of pixels occupied by an individual bee obtained from camera calibration experiments.

3.2 BeePi: A Solar-Powered, Multi-Sensor Electronic Beehive Monitoring Device

The input images for the algorithm are captured by a solar-powered, electronic beehive monitoring device (EBMD), called BeePi [31]. BeePi is designed for the Langsthroth hive [31] used by many beekeepers worldwide. Four BeePi EBMDs were assembled and deployed at two Northern Utah apiaries to collect 28GB of audio, temperature, and image data in different weather conditions. Except for drilling narrow holes in inner hive covers for temperature sensors and microphone wires, no structural hive modifications are required for deployment.

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. Each BeePi consists of a raspberry pi computer, a miniature camera, a solar panel, a temperature sensor, a battery, a hardware clock, and a solar charge controller.

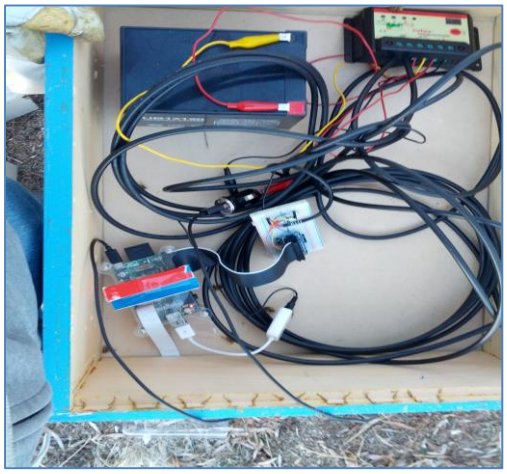

Figure 3.1. BeePi hardware components in a Langstroth super

The exact BeePi hardware components are shown in Fig. 1. We used the Pi Model B+ 512MB RAM models, Pi T-Cobblers, half-size breadboards, waterproof DS18B20 digital temperature sensors, and Pi cameras. For solar harvesting, we used the Renogy 50 watts 12 Volts monocrystalline solar panels, Renogy 10 Amp PWM solar charge controllers, Renogy 10ft 10AWG solar adaptor kits, and the UPG 12V 12Ah F2 sealed lead acid AGM deep-cycle rechargeable batteries.

The camera is placed outside to take static snapshots of the beehive's entrance, as shown in Fig. 2. A small piece of hard plastic was placed above the camera to protect it from the elements. Fig. 3 displays the solar panels on top of two beehives equipped with

BeePi EBMD at the Utah State University Organic Farm in North Logan, Utah. The solar panels were tied to the hive supers with bungee cords. All hardware fits in a shallow super, except for the solar panel placed on top of a hive (see Fig. 3).

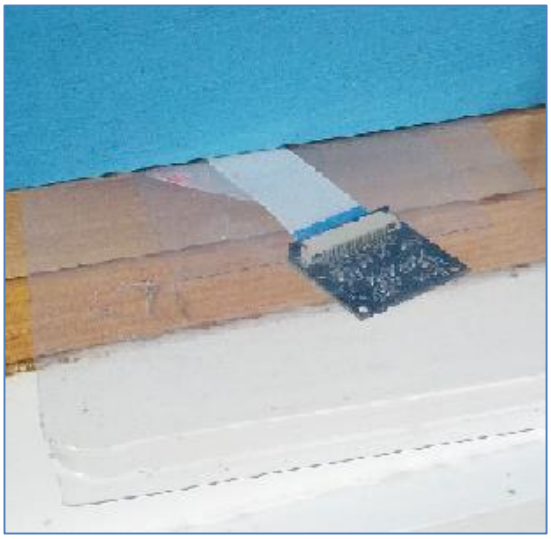

Figure 3.2. Covered BeePi camera

Figure 3.3. Solar panels on beehives

3.3 Pre-Processing Stage

The preprocessing phase can be divided into three steps. The first step is to crop the area of the image where the landing pad is likely to be and the second step is to adjust the brightness of the image.

3.3.1 Cropping Image

As the average size of the image being processed is 550 KB with an average resolution of 720 x 480 pixels, it is time-consuming to perform image operations on the whole image in situ. Hence, we conducted several in situ camera calibration experiments to find the most probable region of the image 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. Fig 3.4 and Fig 3.5 show the output of this stage. Note that there is some grass area cropped along with the landing pad. This is done to compensate for camera swings due to strong winds.

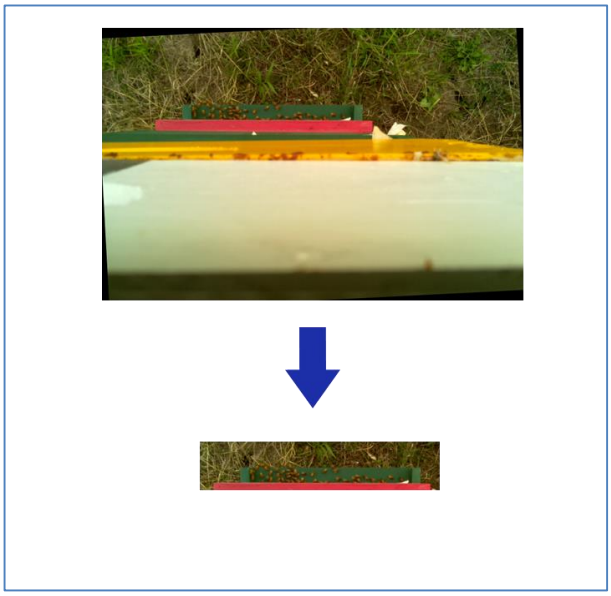

Figure 3.4 Cropping Green Landing pad

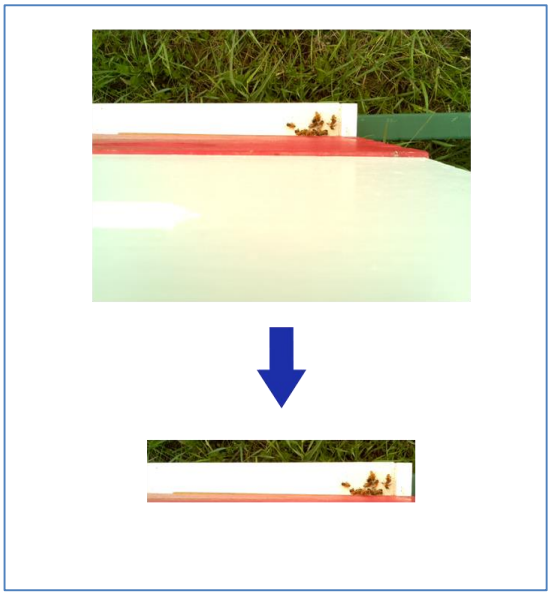

Figure 3.5 Cropping White Landing pad

3.3.2 Adjusting Brightness

The brightness of the image changes due to the sun or overcast weather. If the sun is directly above the beehive, then the brightness of the image is high and if the clouds obstruct sun, the image captured at that moment may have be darker. Both cases have a negative impact on bee counting. To compensate for these two conditions, the brightness of the image is adjusted to lie in (45, 95), i.e., the brightness should be greater than 45 but less than 95. This range was experimentally found to be yield optimal results.

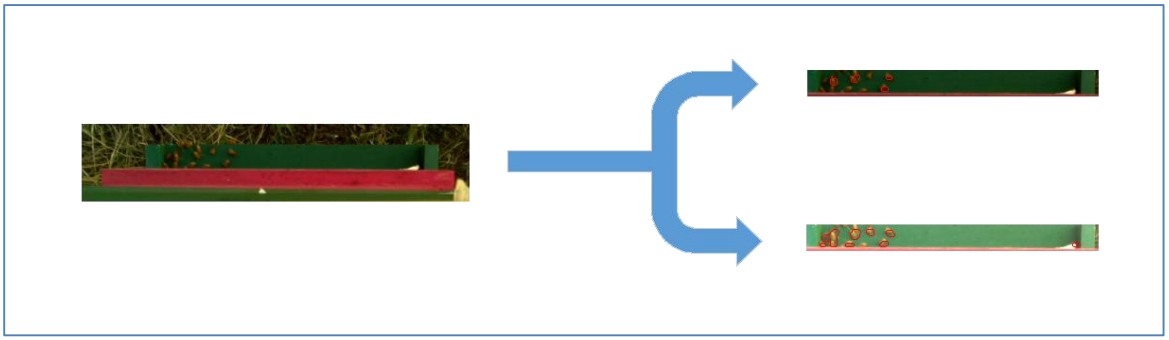

Figure 3.6 Identifying bees in actual and brightened images with green landing pad

In Figures 3.6 and 3.7 illustrate how brightness adjustments improve bee counts. In Fig 3.6, four bees are identified in the darker image and eight bees in the brighter image. In Fig 3.7, adjusting the brightness of the image increases the accuracy of identifying the landing pad, which results in increasing the number of counted bees.

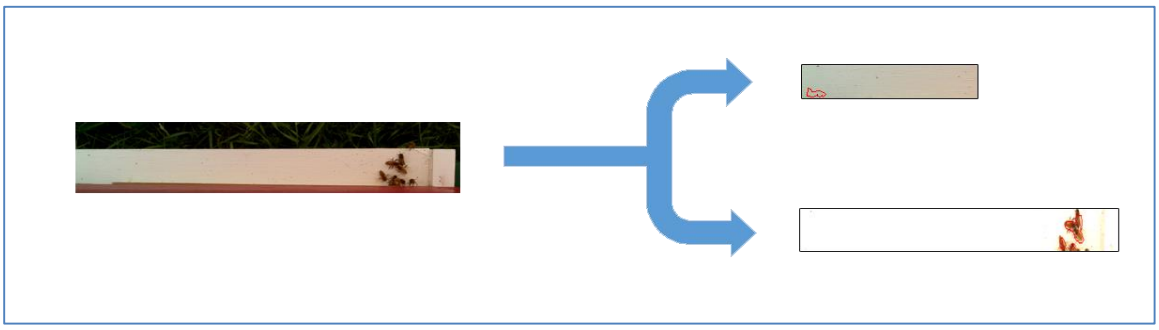

Figure 3.7 Identifying bees in actual and brightened images with white landing pad

3.4 Landing Pad Identification

This stage includes accurate identification of the landing pad as well as background and noise removal.

3.4.1 Landing Pad Identification

It is important to identify the exact region of the landing pad and cropping the region out, because cropping out larger region than the actual landing pad introduces a lot of noise and decreases the accuracy. On the other hand, cropping smaller regions than the actual landing pad area will remove bees from the image, thereby reducing the bee count. We assume that the landing pad is either green or white, because all screened bottom boards in the Langstroth beehives with deployed BeePi's are painted either green or white.

The first step in identifying the landing pad is to convert the image to Hue Saturation Value (HSV) format, which is done by the cvtColor method in OpenCV [33] that converts RGB to HSV. We have used OpenCV 2.4.4 in all our experiments. We have found that it is easier to identify specific color in HSV images than in RGB images. The inRange method of OpenCV is used to identify the areas with specified color.

Noise removal in the image is done by performing a series of erosions and dilations, as shown in Figures 3.8 and 3.9. The white pixels in the final image in Fig. 3.8 represent green color in the actual image and the black pixels represent any color other than green.

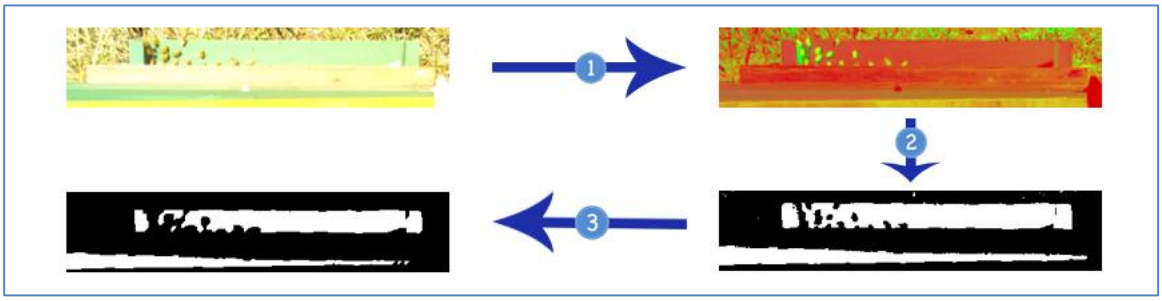

Figure 3.8 Identifying color. 1 – Convert to HSV, 2 – Identify Green, 3 – Remove Noise

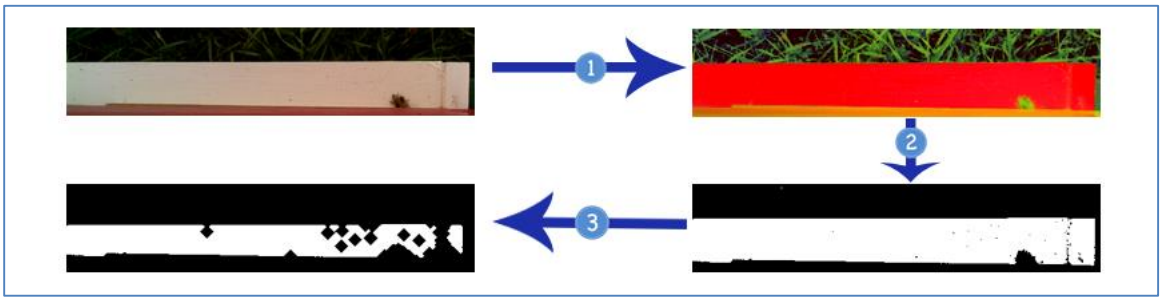

Figure 3.9 Identifying color. 1 – Convert to HSV, 2 – Identify White, 3 – Remove Noise

The next step is to crop the identified area from actual image. But to crop the region of interest, we need to find the bounding rectangle. To find bounding rectangle, we need to find the contours in the identified area and add the points in the contours to a list of points and to find the bounding rectangle, enclosing all the points in the list.

The input for this stage is the output of the third step of the previous stage, as shown in Fig 3.8 and Fig 3.9 in the images to the left of arrow 3. As can be seen in Figures 3.8 and 3.9, both output images have a lot of noise. For example, the image in Fig. 3.8 has a large area identified as green, but it is not part of the landing pad. If we add contours identified in this area and find a bounding rectangle, then we end up cropping larger area of image than the actual landing pad. Thus, we need to remove the green area and noise which are not a part of the landing pad.

To remove the noise and areas that are not part of the landing pad, we first find contours in the image and then find the coordinates of each contour by finding a bounding rectangle for each contour. The bounding rectangle gives the top left corner coordinates of the rectangle as well as its width and height. The found contours are sorted in increasing order by their Y coordinate, i.e., increasing rows. Thus, the contours in the first row of the image will be at the start of the list.

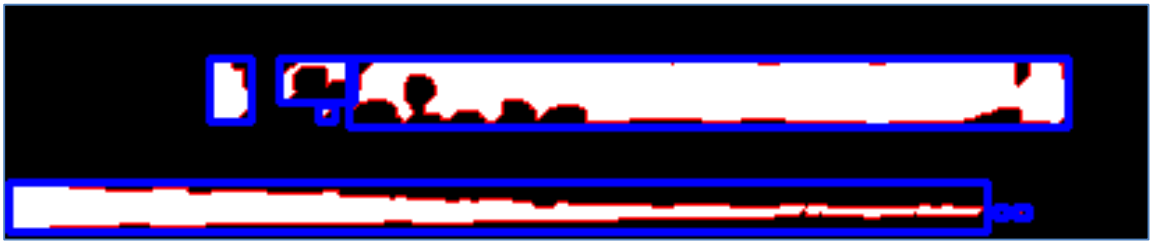

Figure 3.10 Bounding Rectangles drawn for all contours in image with green landing pad

Figure 3.11 Bounding Rectangles drawn for all contours in image with white landing pad

Experimental results show that there are large clusters of contours near the landing pad and if the contour area is at least half the estimated area of the landing pad, then the contour is definitely part of landing pad. On the other hand, the area of noise contours is estimated to be less than 20 pixels.

Using these filters, we compute the approximate location of the landing pad by scanning through all the contours in the sorted list and finding the area of each contour. If the area of contour is at least half the estimated size of the landing pad, then the Y coordinate of the contour is taken to be the average Y coordinate and the scanning process terminates. If the contour area is between 20 and half of estimated landing pad area, the Y coordinate of the contour is saved. Otherwise, the current contour is skipped and the next contour is processed. In other words, if the area of the contour is less than 20 pixels, they are considered as noise and discarded. When the first contour scan terminates, the average Y coordinate is calculated by dividing the sum of all Y coordinates by the number of the processed contours.

Another scan of the sorted list of contours is performed to find all the contours in the range [average Y – H, average Y + H], where H is half of the approximate height of the landing pad. H differs from one bee hive to another, as the alignment of the camera differs from one hive to another. But it can be experimentally found for each hive during the camera calibration experiments ex situ. For example, if the camera is placed close 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. Experimental results show that the H value for the green landing pad images is 20 and for the white landing pad images it is 25.

After getting all the contours satisfying the above conditions, we find a bounding rectangle that encloses all points in the contours. To verify whether we have identified the correct landing pad area, we calculate the area of the bounding rectangle. If the area of the bounding rectangle is greater than the estimated area of the landing pad, then the

bounding rectangle may contain noise, which is not part of the landing pad. So, we iteratively perform another scan to remove noisy contours 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 would been found in first scan. We have calculated the landing pad area experimentally and found that the green landing pad images has an area of 9000 and white landing pad images have an area of 12000. We used these numbers as expected areas.

Figures 3.12, 3.13, and 3.14 illustrate the process of identifying the landing pad area just described. In Fig 3.12, after the first iteration of noise removal, the blue rectangle is identified as landing pad. However, it exceeds the estimated area of landing pad. Thus, we process the image one more time to remove the noise. After the second iteration, the area of the identified landing pad is no greater than the estimated area of the landing pad. Consequently, we crop the identified area from the actual image and terminate.

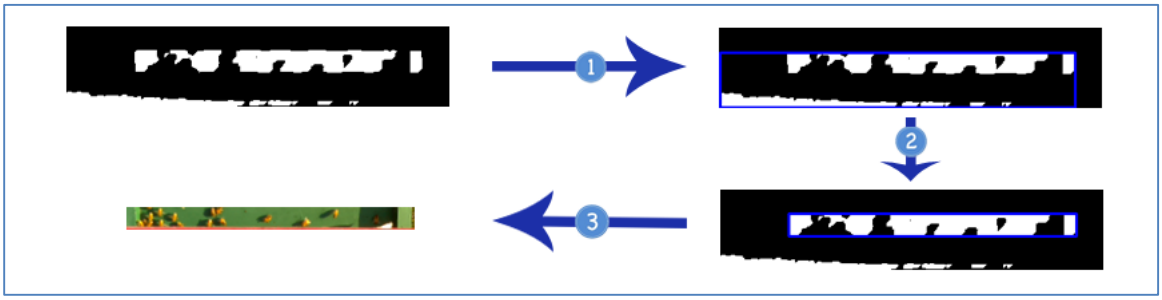

Figure 3.12 identifying green landing pad sample 1. 1 - 1st iteration of noise removal, 2 – 2nd iteration of noise removal, 3 – Crop landing pad

The same scenario can be seen in Fig 3.13. But in Fig 3.14, the landing pad is identified after the first iteration and there is no need for second iteration.

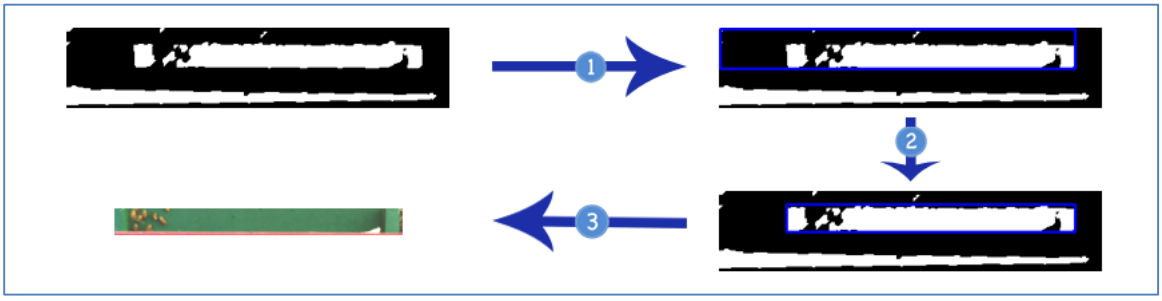

Figure 3.13 identifying green landing pad sample 2. 1 - 1st iteration of noise removal, 2 – 2nd iteration of noise removal, 3 – Crop landing pad

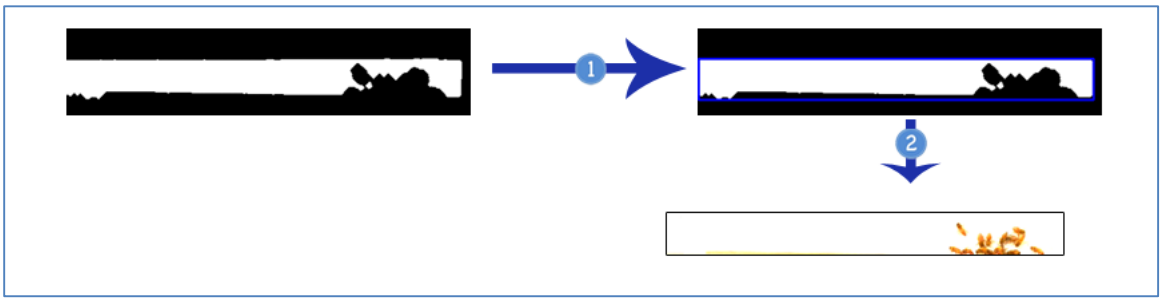

Figure 3.14 identifying white landing pad. 1 - 1st iteration of noise removal, 2-Crop landing pad

3.4.2 Background Removal

We can differentiate between foreground and background based on color. For green landing pads, the background is green and the bees are yellow and for white landing pads, the background is white and the foreground is yellow. Using this differentiation, all the pixels which have shades of green or white are changed to white (RGB value 255) and the remaining pixels are changed to black (RGB value 0). Experiments on color ranges, gave us the thresholds and ranges that can be used to

identify shades of green and shades of white while ignoring shades of yellow. Three rows of border pixels in the landing pad image are converted to white to facilitate easy identification of contours in the next step.

Figures 3.15 and 3.16 show the output of the background removal stage. In Fig. 3.15, the green background is converted to white and the bees are represented by the black pixels in the image. In Fig 3.16, the white background is converted to pure white and bees to black. Noise can be introduced in this phase. So we need to remove the noise.

Figure 3.15 removing background from green landing pad images.

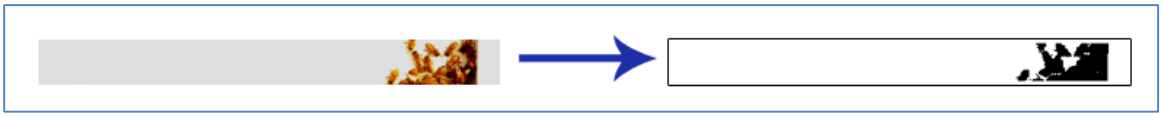

Figure 3.16 removing background from white landing pad images.

3.4.3 Noise Removal

Since noise introduced in the previous step may negatively impact bee counting accuracy, we need to de-noise the image by performing a sequence of erosion and dilation operations. We have used OpenCV implementations for erosion and dilation [34]. We define a structuring element of 2 x 2 pixels and use it to modify the neighboring pixels. Figures 3.17 and 3.18 show the output of this step.

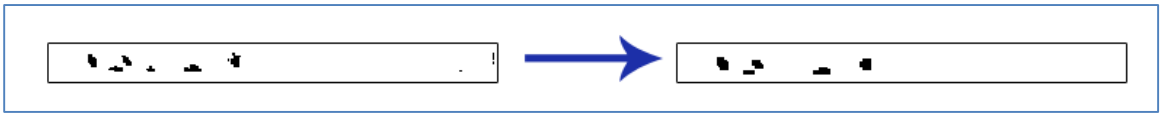

Figure 3.17 removing noise from green landing pad images.

Figure 3.18 removing noise from white landing pad images.

3.5 Bee Counting

The final stage consists of identifying the bees in the output image from the previous step, calculating the total area of the bees and dividing the area by estimated area of a bee.

3.5.1 Identifying and Counting Bees

To identify bees in the image, we convert the image to grayscale with the OpenCV method cvtColor. The OpenCV method findContours is used to find contours. We have to experiment with different parameter values to determine the best combination for a given dataset. Experimental results show that the area of a bee or a group of bees resides in (20, 3000), where both values are numbers of pixels. Thus, we scan all found contours to check if the area of the contour is between 20 and 3000 pixels. If the area is less than 20 pixels, the contour might be noise or part of a bee. If the area of the contour is greater than 3000, then part of the landing pad might have been identified as a bee group. In either case, we eliminate these contours and add the area of all the remaining contours.

The area of one individual bee is between 35 and 100 pixels, depending on the alignment of the Pi camera. If the camera is placed closer to the landing pad, the area of the bee would be greater and if the camera is far from the landing pad, then the area of the bee would be smaller. This can be easily observed in two sets of images. The green

landing pad images were captured by a Pi camera placed far from the landing pad and the experimental results show that the average area of the bee is 40. On the other hand, the white landing pad images were captured by a Pi camera placed closer to the landing pad and the average area of an individual bee is 100. Therefore, to find the number of bees in green landing pad images, we divide the total area by 40, whereas, for the white landing pad images, the total area is divided by 100. The result is the most probable count of bees in the image. Fig 3.19 and Fig 3.20 show the identified bees on a green landing pad and on a white landing pad, respectively.

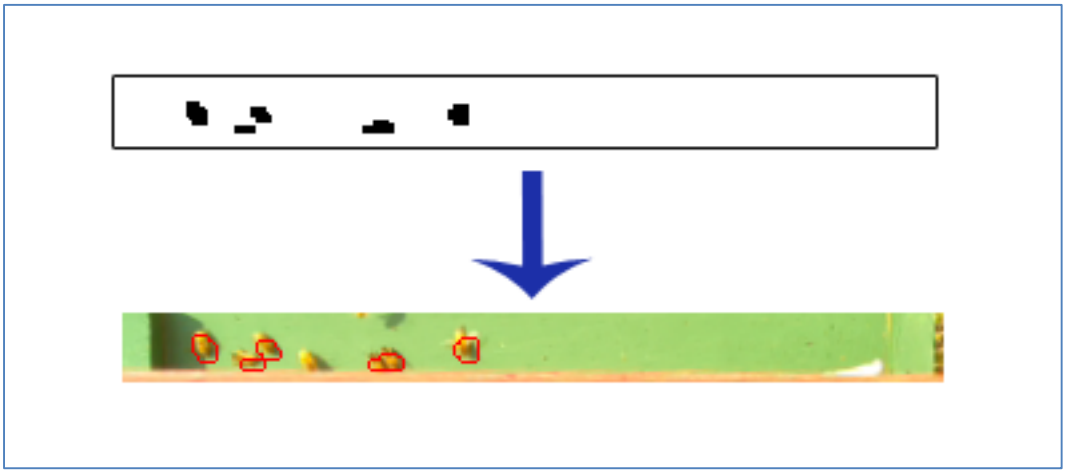

Figure 3.19 Identify bees in image with green landing pad and draw contours.

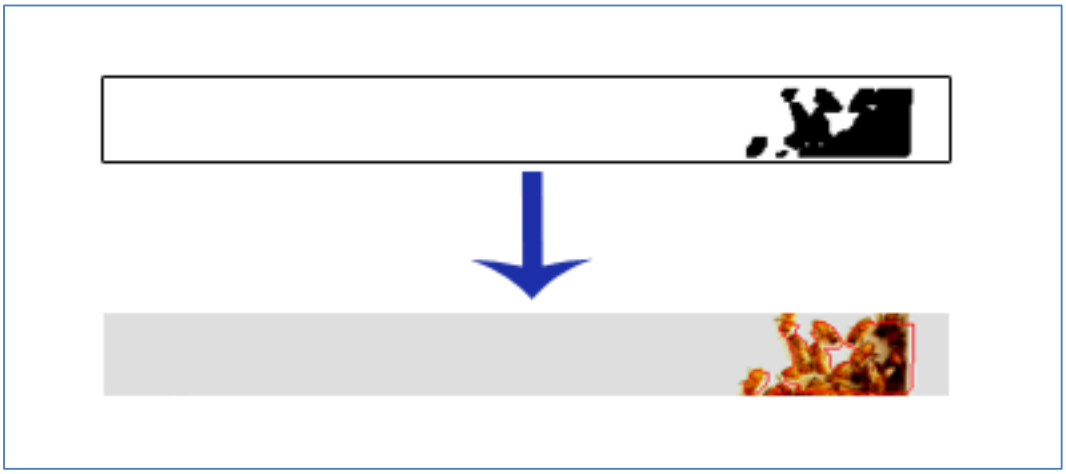

Figure 3.20 Identify bees in image with white landing pad and draw contours.

CHAPTER 4

EXPERIMENTS

4.1 Overview

This chapter describes the experiments we conducted to evaluate the accuracy of our algorithm. A sample of 1,005 images with green landing pads and 776 images with white landing pads was captured by two BeePi EBMDs deployed in two Norther Utah apiaries described in Section 3.2. The ground truth was obtained by four human evaluators who counted the number of bees on the landing pad in each image. Each evaluator was given a set of images and the instructions on how to count the bees.

4.2 Experimental Procedure

Each image has a resolution of 720 x 480 and takes 550 KB of memory. We recruited six human evaluators and provided them each with a set of images. They were instructed to count the number of bees on the landing pad and record their observations in an excel spread sheet. Fig 4.4 shows an example of the noted observations.

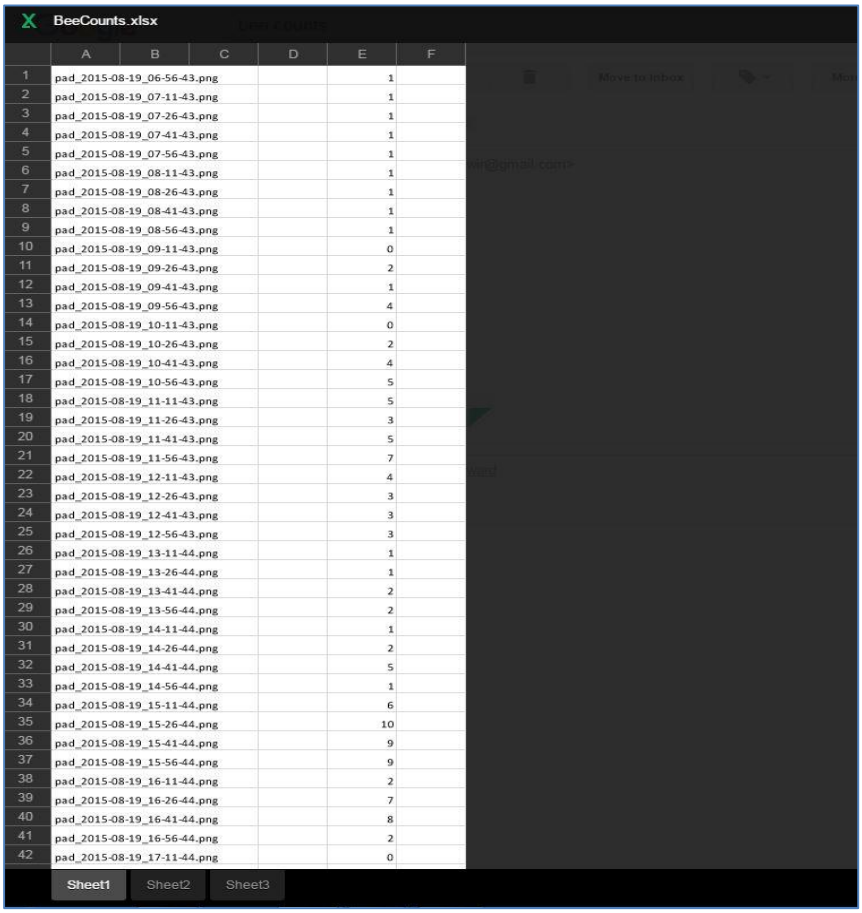

Figure 4.1 Recorded observations

All the results were then consolidated into a single spread sheet. These counts served us as the ground truth to evaluate our algorithm. Table 4.1 shows the ground truth for all images.

Table 4.1 Results of Human Evaluation

S. No	Type	Total Images	Total Bees	Mean	Standard Deviation
1	Green Pad	1005	5770	5.7	6.8
2	White pad	776	2178	2.8	3.4

To find the accuracy, the same images are given to the algorithm as input. The results are recorded in Table 4.2.

Table 4.2 Results of Algorithm

S. No	Type	Total Images	Total Bees	Mean	STD	Accuracy
1	Green Pad	1005	5263	5.2	7.6	80.5%
2	White pad	776	2226	2.8	4.1	85.5%

The accuracy of the algorithm comes to 80.5% for the green landing pad images and 85.5% for the white landing pad images. There are both false positives and true negatives due to improper cropping of the landing pad, part of bee on the landing pad, image brightness, and removal of bees of parts of bees as noise or background. For example, in Fig 4.5 there are 16 bees on the landing pad in the upper image, but the algorithm detects only 8 on the cropped and de-noised landing pad in the lower image. This is due to improper cropping of the landing pad.

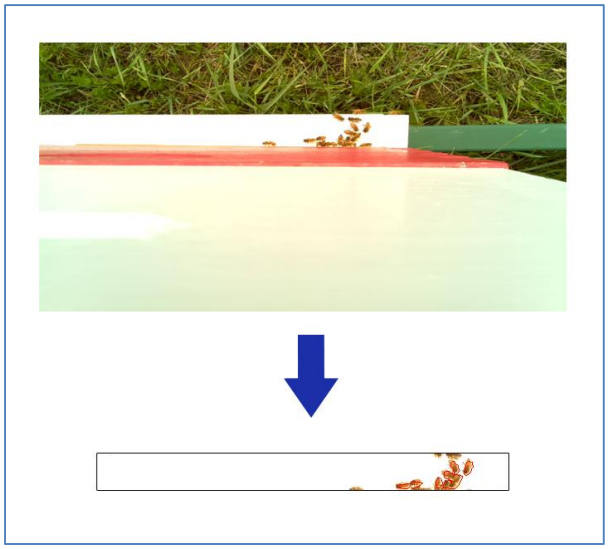

Figure 4.2. True negatives due to improper landing pad identification

There are many reasons for true negatives. We have observed some common reasons due to which there is a mismatch between the human evaluation and algorithm evaluated bee counts. We use Bounding rectangle to get the Region of Interest from the cropped image. The bounding rectangle is perfect rectangle without any skew angle. So if the landing pad is skewed, then a part of the landing pad would be cropped out. The bees can be on the landing pad or on the side walls of the hive or side walls of the landing pad. If the bee is on the side wall, then the area of the pixels identified as a bee would be less. To count the number of bees we first find out the total area identified as bees and divide it by the average area of a bee. So, the orientation of the bees can affect the count.

Removing background can remove parts of bees and also can introduce some noise. To remove the noise, we perform erosion and dilation. This may remove bees as well. The boundaries of the landing pad should be strict. We shouldn't compromise to get the whole landing pad, as there is a great chance of introducing noise due to grass and also due to the red box of the hive.

The algorithm sometimes identifies that there are bees on the landing pad even if there are no bees. These are false positives and can be caused due to various reasons. To identify the most of the bees, the brightness of the image should be in a particular range. If the brightness of the image is more than the required, then the brightness of the image is reduced. If it is less, then we increase the brightness of the image. If we change the brightness, we can end up changing some features of the images and the noise created while cropping, which cannot be eliminated totally is identified as a bee

As the brightness of the image is reduced, we ended up adding yellowish shade to

some parts of the image. We cannot eliminate yellow as they may represent bees. This can be taken care by making the background removal more robust by introducing more conditions to check each pixel and determine whether the pixel is a part of bee or not.

In Fig 4.6, there are only 11 bees but the algorithm identifies 28 bees in the landing pad. When the brightness of the image is reduced, it introduced shades of yellow to the landing pad and the background removal algorithms wouldn't remove pixels with shades of yellow.

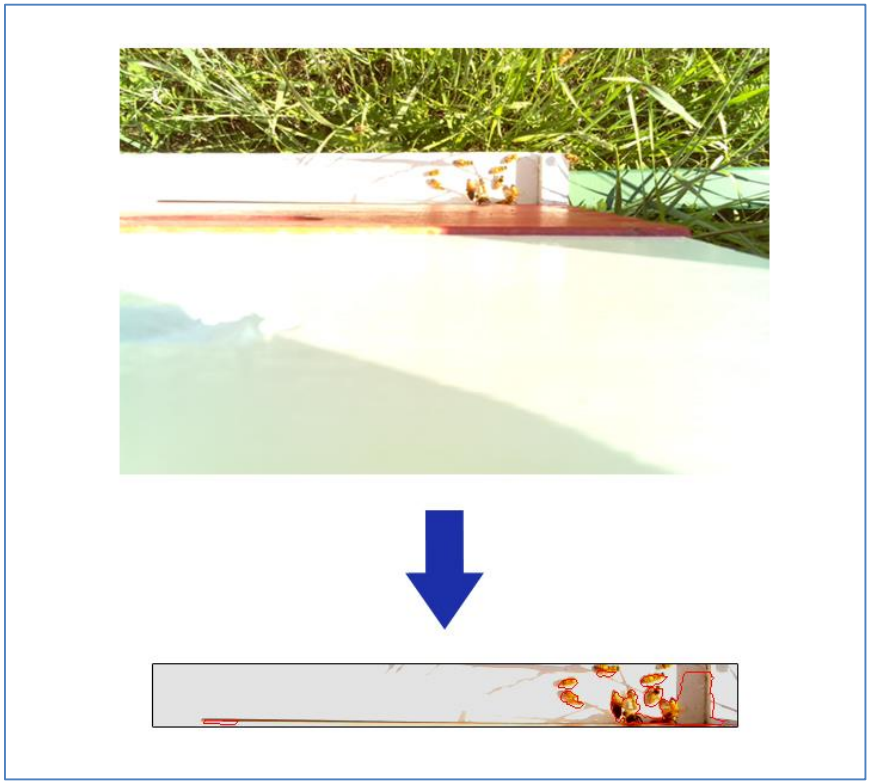

Figure 4.3 False positives

CHAPTER 5

RESULTS

The experiments show that the bee counting algorithm achieves an accuracy of 80.5% and 85.5% on green pad images and white pad images, respectively. Numbers in the Table 4.1 show the results of human evaluation of the images. We have used 1,005 images with green landing pad and 776 images with white landing pad. Human evaluators have identified a total of 5,770 bees with an average of 5.7 bees per image in images with green landing pad. In images with white landing pad, human evaluators identified 2,178 bees identified with a mean of 2.8 bees per image.

In Table 4.2, we have presented the results of our algorithm. Our algorithm had identified 5,263 bees out of 5,770 with an accuracy of 80.5% and an average of 5.2 bees per image in green landing pad images. On the white pad images, the algorithm had identified 2,226 bees out of 2,178 with an accuracy of 85.5% and 2.8 bees per image, on average.

The scatter plots in Fig 5.1 and Fig 5.2 shows the accuracy of the algorithm. The closer the points to the diagonal line, the more accurate the algorithm. If the points are far from the diagonal line they are consider outliers and the algorithmic count differs from the human evaluator count. Scatter plots for both the implementations are given below.

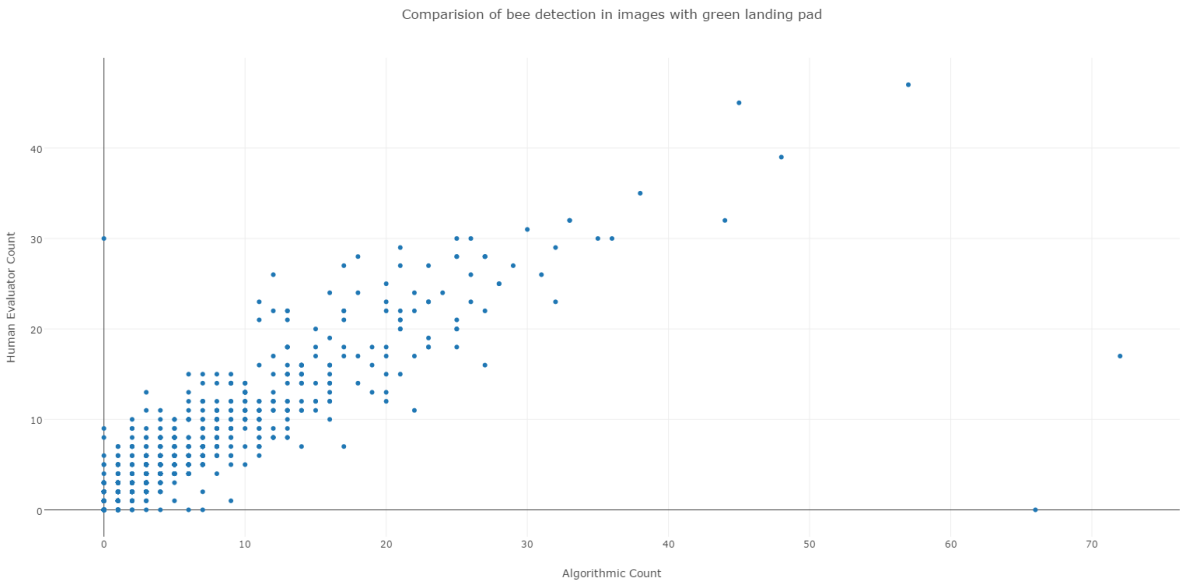

Figure 5.1 Comparison of actual and identified counts of bees in green pad images

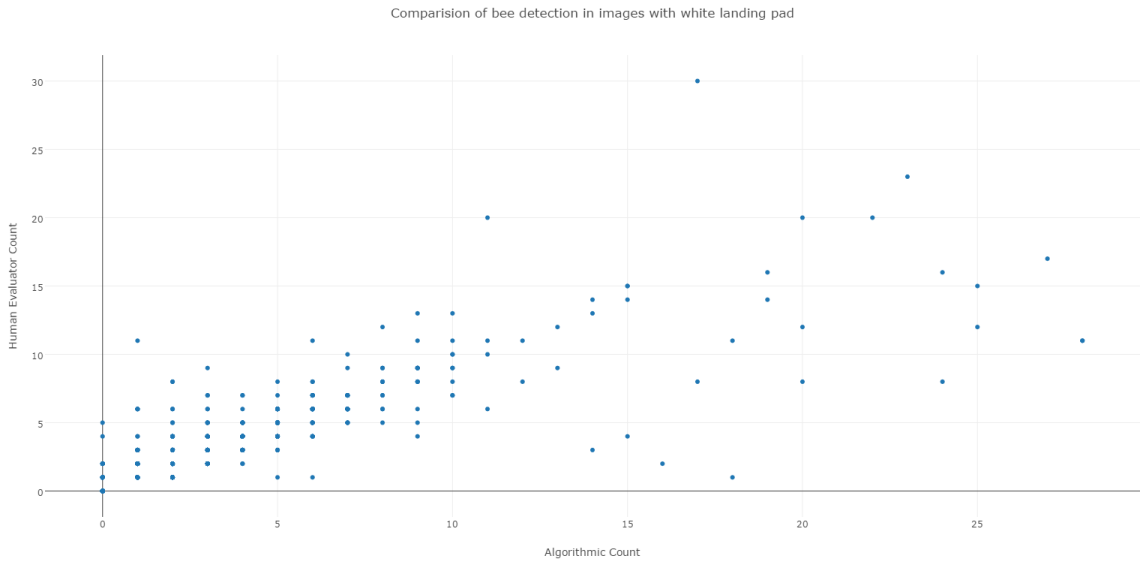

Figure 5.2 Comparison of actual and identified counts of bees in white pad images

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

An algorithm was presented to count bee numbers in images of Langsthroth hive entrances. The algorithm computes approximate bee counts by adjusting the brightness of the image, cropping a white or green area in the image, removing the background and noise from the cropped area, finding the total number of bee pixels, and dividing that number by the average number of pixels in a single bee.

Our experiments on a sample of 1,005 images with green landing pads and 776 images with white landing pads indicate that our algorithm have 80.5% and 85.5% accuracy compared to the ground truth of human evaluation. Human evaluators have identified 5,770 bees in 1,005 images and the algorithm identified 5,263 bees. In 776 white landing pad images, human evaluators counted 2,178 bees, of which the algorithm identified 2,226 bees.

In our future work, we propose to deploy the project onto the raspberry pi, which is part of the BeePi EBMD hardware [31] and calculate the number of bees in situ and save the count in a text log.

We also plan to automate image alignment so that the landing pad in each image is aligned horizontally to improve bee counting accuracy.

REFERENCES

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

[2] “Honeybees are dying, and scientists still don't know why” [Online]. Available : http://www.cnbc.com/2015/05/14/honeybees-are-dying-and-us-fruit-and-nut-crops-may-suffer.html.

[3] “Varroa destructor”[Online]. Available: https://en.wikipedia.org/wiki/Varroa_destructor

[4] “Deformed wing virus” [Online]. Available: https://en.wikipedia.org/wiki/Deformed_wing_virus

[5] “American foulbrood” [Online]. Available: https://en.wikipedia.org/wiki/American_foulbrood

[6] “European Foulbrood: A Bacterial Disease Affecting Honey Bee Brood” [Online]. Available: http://articles.extension.org/pages/23693/european-foulbrood:-a-bacterial-disease-affecting-honey-bee-brood.

[7] “Nosema apis” [Online]. Available: https://en.wikipedia.org/wiki/Nosema_apis.

[8] M. T. Sanford. "2nd international workshop on hive and bee monitoring", *American Bee Journal*, Dec 2014, pp. 1351-1353.

[9] “Honey Bees Turned Data Collectors Help Scientists Understand Climate Change” [Online]. Available : http://www.nasa.gov/topics/earth/features/beekeepers.html

[10] J. G. Koomey "*Growth in Data Center Electricity Use 2005 to 2010*". Analytics Press, 2011

[11] "What's Stopping Liquid Cooling?" [Online]. Available : http://www.datacenterjournal.com/whats-stopping-liquid-cooling/

[12] "Facebook: Power use in Prineville data center more than doubled last year" [Online]. Available: http://www.oregonlive.com/siliconforest/index.ssf/2013/07/facebook_power_use_in_prinevil.html

[13] G. Cook, D. Pomerantz, K. Rohrbach and B. Johnson "*A Guide to Building the Green Internet*" Greenpeace, Washington, DC, May 2015.

[14] P. Delforge and J. Whitney "*Data Center Efficiency Assessment*" Natural Resources Defense Council, New York, NY, IP:14-08-a, Aug 2014

[15] D. A. Mezquida and J. L. Martínez "Short communication. Platform for bee-hives monitoring based on sound analysis. A perpetual warehouse for swarm's daily activity" *Spanish Journal of Agricultural Research*, Vol 7, Issue 4, pp 824-828.

[16] M. T. Sanford. "2nd international workshop on hive and bee monitoring," *American Bee Journal*, December 2014, pp. 1351-1353.

[17] W. G. Meikle , N. Holst "Application of continuous monitoring of honeybee colonies" *Apidologie*, Volume 46, Issue 1, pp 10-22, Jan 2015

[18] N. E. Gary, Activities and behavior of honey bees. In: The Hive and The Honey Bee (Ed. J.M. Graham), Dadant and Sons, Hamilton, IL, pp. 269-372, 1992.

[19] J. L. Gould, C. G. Gould “The Honey Bee”. New York, NY, Scientific American Library, 1988.

[20] A. E. Lundie, “The flight activities of the honey bees”, United States Department of Agriculture, Dept. Bull. No. 1328:1-38, 1925.

[21] A. C. Faberge, “Apparatus for recording the number of bees leaving and entering a hive”. *Journal of Scientific Instruments*. Vol 20, pp 28–311, Feb 1943.

[22] E. H. Erickson, H. H. Miller, D. J. Sikkema, “A method of separating and monitoring honey-bee flight activity at the hive entrance”. *Journal of Apicultural Research.*Vol 14, pp 119–125, 1975.

[23] C. Liu, J. J. Leonard, J. J. Feddes “Automated monitoring of flight activity at a beehive entrance using infrared light sensors”. *Journal of Apicultural Research.* Vol. 29, 1990, DOI:10.1080/00218839.1990.11101193

[24] R. G. Danka, and N. E. Gary, Estimating foraging populations of honey bees (Hymenoptera: Apidae) from individual colonies. *Journal of economic entomology*, Vol 80, Issue 2, pp.544-547, 1987

[25] M. H. Struye, H. J. Mortier, G. Arnold, C. Miniggio, R. Borneck, “Microprocessor-controlled monitoring of honeybee flight activity at the hive entrance”. *Apidologie*, Vol 25, pp: 384-395, 1994.

[26] C. W. Schneider, J. Tautz, B. Grünewald, S. Fuchs, “RFID Tracking of sublethal effects of two neonicotinoid insecticides on the foraging behavior of Apis mellifera”. *PLoS ONE,* Vol 7, e30023. doi:101371/journal.pone.0030023, 2012.

[27] M. Strye, G. Borremans, F. J. Jacobs: "Monitoring honey bees the design of a computer operated bee counter", *Nederlandse Entomologische Vereniging* Amsterdam, Vol 2, 1991.

[28] J. J. Bromenshenk , C. B. Henderson , R. A. Seccomb , P. M. Welch, S. E. Debnam and D. R. Firth. "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.

[29] "Arnia" [Online]. Available: http://www.arnia.co.uk/

[30] "HiveMind" [Online]. Available http://hivemind.co.nz/

[31] V. Kulyukin, M. Putnam, and S. K. Reka. "Digitizing Buzzing Signals into A440 Piano Note Sequences and Estimating Forage Traffic Levels from Images in Solar-Powered, Electronic Beehive Monitoring", *International Conference on Computer Science*, Kowloon, Hong Kong, Vol. I, pp. 82-87, 2015.

[32] L. L. Langstroth. "Langstroth on the Hive and the Honey Bee: A Bee Keeper's Manual". Dodo Press: UK, 2008; orig. published in 1853.

[33] "OpenCV" [Online]. Available: http://opencv.org/

[34] "Eroding and Dilating" [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html

[35] "Beekeeping in the United States" [Online]. Available : https://en.wikipedia.org/wiki/Beekeeping_in_the_United_States

[36] “How much honey does a hive produce in a year” [Online]. Available: https://www.quora.com/How-much-honey-does-a-hive-produce-in-a-year?share=1

[37] “RasPi power usage measurements ALL Models” [Online]. Available: https://www.raspberrypi.org/forums/viewtopic.php?f=2&t=6050

[38] “The Price Of Electricity In Your State” [Online]. Available: http://www.npr.org/sections/money/2011/10/27/141766341/the-price-of-electricity-in-your-state

[39] “How much carbon dioxide is produced per kilowatthour when generating electricity with fossil fuels?” [Online]. Available: https://www.eia.gov/tools/faqs/faq.cfm?id=74&t=11

APPENDICES

APPENDIX A

```
1. FUNCTION DetectBeeTraffic(folderPath, rowStart, rowEnd, colStart, colEnd)
2. Images[]  = Read Images from folderPath
3. count = {}
4. For i=0 to Count of Images -1
5.      Images[i] = Submat(rowStart, rowEnd, colStart, colEnd)
6.      AdjustBrightness(Images[i])
7.      HSV = Convert to HSV(Images[i])
8.      BinImg = IdentifyColor(Color, HSV)
8a.     RemoveNoise(BinImg)
9.      Pad = FindLandingPad(BinImg)
10.     BinImg2 = RemoveBackground(Pad)
11.     Remove Noise in Binary Image
12.     Contours = IdentifyContours(BinImg2)
13.     totalArea = 0;
14.     For Each Contour in Contours
15.             If Contour.area  > 20 && Contour.area < 3000
16.                     totalArea = totalArea + Contour.area
17.             End If
18.     End For
19.     count[i] = Round(totalArea/Average area of a bee)
20. End For
21. Return count

22. FUNCTION AdjustBrightness(Img)
23.     brightness  = getAverageLuminance(Img)
24.     If luminance < 45
25.             IncreaseLuminance(Img)
26.     Else If luminance > 90
27.             ReduceLuminance(Img)
28.     End If
29. Return Img


30. FUNCTION FindLandingPad(White)
31.     Contours = IdentifyContours(White)
31a.    Find the bounding rectangle for each
32.     SortContoursByBoundingRectangleY(Contours
```

```
33.     Filter those rectangles whose area is below a threshold
34.     Rect = BoundingRect(Remaining Contours)
35.     If Rect.area > 12000
36.             Filter the Outliers in the contour list again
37.             Rect = BoundingRect(Contours)
38.      End If
39. Return Rect

40.  FUNCTION  RemoveBackground(Pad)
41.     source = {}{}
42.     For r =0 to Pad. Rows
43.             For c =0 to Pad.Cols
44.                     pixel = Pad.getPixel(r,c)
45.                     If r<=3||col<=3||r> Pad.Rows – 4 || c >= Pad.Cols – 3
46.                             pixel = 255
47.                     Else If pixel[0] > = 120 && pixel[1] > = 150 && pixel[2] > = 180
48.                             pixel =255
49.                     Else
50.                             pixel = 0
51.                     End If
52                      source[r][c] = pixel;
53.             End For
54.     End For
55. Return source
```

Pseudocode of the Bee Counting Algorithm