SEMI-AUTOMATIC TOOL TO COUNT MOSQUITO EGGS IN OVITRAP STICK IMAGES

Charles Beumier\(^1\), Jorge Rubio\(^2\), Veronica Andreo\(^2\)-\(^3\), Claudio Guzman\(^4\), Ximena Porcasi\(^2\), Carlos Marcelo Scavuzzo\(^2\), Michal Shimoni\(^1\)

\(^1\)Signal & Image Centre, Belgian Royal Military Academy, Brussels, Belgium
\(^2\)Instituto de Altos Estudios Espaciales “Mario Gulich” – CONAE, Cordoba, Argentina
\(^3\)CONICET (Consejo Nacional de Investigaciones Cientificas y Tecnicas), Argentina
\(^4\)Direccion de Epidemiologia de Cordoba

ABSTRACT

Oviposition measurement with ovitraps is one of the most widely used methods to monitor \textit{Aedes aegypti} mosquito activity in the world. Egg counting is however very time consuming. This paper presents the semi-automatic counting of mosquito eggs laid on ovitrap sticks in images acquired by cellular phones. In Cordoba, Argentina, 150 ovitraps were distributed in the city to measure the evolution of the \textit{Aedes aegypti} population, estimated indirectly by the number of laid eggs. An important increase in the counts is a potential indicator of an imminent outbreak, alerting the health services to warn the population and recall the good sanitary practices. Bringing image processing to this project is a way to relieve the technician from the tedious egg counting behind a magnifier and to reduce the count errors due to distraction or fatigue. We developed a fast semi-automatic counting solution with tools to focus on the useful image area, to show the confidence of automatic count numbers and to handle the collection of results.

Index Terms— \textit{Aedes aegypti}, monitoring, semi-automatic egg counting, GUI, computer vision

1. INTRODUCTION

\textit{Aedes aegypti} mosquito is a sanity threat since it can transmit viruses such as the dengue, zika, chikungunya and yellow fever. It is particularly important to be aware of its spread and to know when and where there is a high reproduction rate [1].

Due to the frequent Dengue outbreaks registered in the Americas and Argentina [2], Cordoba is one of several cities in which the health ministry implemented a monitoring solution. This consists of 150 sites selected in the city to place ovitraps. An ovitrap [3] is typically made of a wooden stick (a tongue depressor) standing in a cup partly filled with water, what represents an attractive place for an \textit{Aedes aegypti} mosquito to lay eggs. This normally happens on the wood close to the water level. In times of high abundance and high levels of reproductive activity, the task of counting eggs can be very time consuming, with a consequent increase in the probability of making mistakes. Therefore the Gulich Institute was asked to automate counting of mosquito eggs in ovitrap stick images.

Initiatives in counting mosquito eggs automatically are not new. In 2012, a solution called \textit{EggCounter} was developed and made available as a java application [4]. It is semi automatic in the sense that the automatic count of each detected dark blob can be edited by the user (a blob can hold several adjacent eggs). In 2016, the \textit{ICount} [5] solution was proposed and distributed as a Windows executable file. This system was developed for a laboratory controlling the mosquito reproduction and a scanner which digitizes sandpaper strips on which eggs are laid. Both systems provide a semi-automatic way to improve egg detection by adapting one or several thresholds.

In the current project, we planned to reduce human intervention so as to make egg counting nearly automatic. Automatic procedures are provided for the two most error prone steps: the stick detection for correct area delineation and the threshold estimation for correct egg segmentation. As a fallback solution or for optimization, the user can manually adjust the related parameters. The system is intended to be used every week for 150 mosquito traps. In burst periods the count on a stick can reach several hundred eggs [6, 7]. Automatism is welcome to speed up the procedure and let the person concentrate on difficult cases. Compared to [4, 5] our development considered eggs with lower image resolution obtained by cellular phones.

Ovitrap sticks can be damaged or eggs can fall off when dry. Taking pictures of the sticks are necessary for the counting program but they are also useful for long-term archiving of data. An ovitrap contains more than just a number of eggs. The egg distribution over the stick and the threshold estimation for correct egg segmentation.

Demonstrated in [4] for the \textit{Icount} system, the \textit{ICount} system provides a semi automatic way to improve egg detection by adapting one or several thresholds. In the current project, we planned to reduce human intervention so as to make egg counting nearly automatic. Automatic procedures are provided for the two most error prone steps: the stick detection for correct area delineation and the threshold estimation for correct egg segmentation. As a fallback solution or for optimization, the user can manually adjust the related parameters. The system is intended to be used every week for 150 mosquito traps. In burst periods the count on a stick can reach several hundred eggs [6, 7]. Automatism is welcome to speed up the procedure and let the person concentrate on difficult cases. Compared to [4, 5] our development considered eggs with lower image resolution obtained by cellular phones.

Ovitrap sticks can be damaged or eggs can fall off when dry. Taking pictures of the sticks are necessary for the counting program but they are also useful for long-term archiving of data. An ovitrap contains more than just a number of eggs. The egg distribution over the stick and more precisely the level at which eggs were laid may reveal some facts. New computer programs might highlight new information and tests on archived images could become an added value.
2. METHODOLOGY

Each ovitrap consisted of a black 1000 ml plastic container filled with 250 ml of water and a wooden tongue depressor (13 x 2 cm) held vertically to serve as substrate for mosquito oviposition. Eggs laid in each wooden paddle were counted under magnifying glass. All ovitraps were replaced weekly.

Stick images were captured in the lab with 2 cellular phones (12 and 16 Mpixel), in controlled conditions. The sticks were laid on a uniform background. A minimal distance had to be maintained to ensure image sharpness. The typical egg length in images varied from 12 to 20 pixels, what is quite challenging for image processing. More than 20 images were used for development and tests.

From the observation of mosquito eggs images, we identified two main challenges in optimal image conditions, supposing that sticks are in focus with sufficient contrast.

The first challenge is the separation between egg and non-egg regions. The eggs are dark and easily separated from the bright background of the sticks. But the stick might contain dirty parts, might be rot from the water it stays in a week long and might be covered by insects or excrements.

The second challenge concerns egg counting. For isolated eggs, detection and counting can be straightforward from dark object delineation. For eggs grouped into clusters egg separation can be difficult so that their number is estimated from the cluster area, as proposed in [4]. The detection and the area of clusters however depend on the threshold used to find them.

We propose in this paper an image processing solution dealing with the challenges of non-egg removal and egg count estimation in clusters. We implemented a focusing technique to restrict the search to the stick and a multi-thresholds procedure to estimate the confidence in the count returned so that the operator decides whether to check and update the count or not.

3. IMAGE PROCESSING AND GUI

The proposed solution uses the openCV [8] library for image processing and the Qt toolkit for user interaction. First a region of interest is located around the stick. Then dark objects are delineated and eggs are counted based on the objects area. A graphical user interface (GUI) allows for the visualization and potential modification of the results by the operator.

3.1. Region of interest

To detect the stick thanks to its specific color, the openCV cv2.inrange function is used in the HSV color space to keep pixels with a red hue (-30 to 30), a moderate saturation (30 to 120) and a high value (100 to 255). This segmentation is trivial since stick images were captured with a controlled background. The largest contour of the mask created by cv2.inrange should contain all the eggs.

A bounding box enclosing this largest contour is shown to the user. It can be manually adapted for correction or refinement. The bounding box avoids false detections outside the stick. It also enables a large reduction in the number of pixels to be processed by the following steps.

From the known stick width (20 mm) and the stick bounding box we derive an estimate of pixels per mm.

3.2. Object detection

The common method to find objects in openCV consists in thresholding (cv2.threshold()) the image and finding the contours thanks to cv2.findContours().

A low threshold is expected since the eggs are dark. Its value depends mainly on the illumination conditions. Contours are fitted to an ellipse thanks to cv2.fitEllipse(), so that the position x, y and semi axis lengths a, b (b > a) are known. Contours with a length (2*b) shorter than 0.3 mm (in pixels) are rejected since the eggs are longer than about 0.5 mm. Large contours will also be filtered out, but once the equivalent area of one egg has been estimated.

The area of each contour is computed by cv2.contourArea(). A single egg area value is estimated as the median area of contours which have a length in the range 0.4-0.8 mm, a little larger than the egg length range. This supposes that isolated eggs are present in the image what is normally observed with Aedes mosquitoes [9] and always for our dataset.

The threshold value has an influence on the number and size of the detected contours. Increasing the threshold enlarges a dark blob until it is no longer separated from its neighborhood and disappears. We propose to apply several threshold values to solve the problem of threshold selection. As a speed up, only the pixels in the bounding box are considered. For each threshold, eggs are counted for the accepted contours as explained in the next subsection.

3.3. Counting eggs

Egg counting is based on the contour area. This area is divided by the estimated egg area to give an integer count for the contour. Egg clusters are often overestimated due to the area enclosed in between eggs. Therefore we applied a 0.85 empirical reduction factor for clusters equivalent to more than 5 eggs. A region larger than the equivalent of 25 eggs has been considered as a false egg cluster by default and is rejected. Such large areas are typically due to dust, insects or wood defects.
3.4. Threshold selection

The total sum of each egg count in blobs is computed for each threshold value in the range $[THR-50, THR+50]$, with $THR$ being the default base threshold (e.g. 75). These 101 total sums form a profile that the user can appreciate in a glance. A successful estimate normally displays a plateau with rather constant count. Considering all ranges of 15 consecutive threshold values, the one with minimal count variance is selected. The central threshold of this range and its median count are returned. The user can accept this count or adapt the threshold, possibly after a visual evaluation of the blob counts now superimposed on the image. He may also decide to update these blob counts individually.

3.5. User interface

The user interface is an important part of the development. First a bounding box focusing on the stick (see Figure 1) is displayed on the image and can be adapted manually. Secondly, the profile of counts is shown for interpretation of the threshold robustness. And finally, if accuracy is desired and time is available, the egg count of each contour is displayed nearby on the image for visual evaluation or edition (Figure 1).

After acceptance of a total egg count by the user, the results file is updated. If the image filename is new, a new line is created with that name and total count. If not, the count for this filename is updated. Storing additionally the image location and count of each contour would allow for later correction and research, like for instance the study on egg distribution on the stick.

4. RESULTS AND DISCUSSION

The software solution was developed in python 2.7, with openCV 3.2 for image processing and PyQt4 for the GUI. It was then ported to python 3.5, openCV 4.2 and PyQt5. It ran on a laptop computer, handling an image in less than one second in automatic mode.
The automatic bounding box normally fits the stick well if the background color is not too similar to the stick. If not, it should be refined to offer a correct estimation of the pixel to mm ratio, although a rough value suffices.

The particularity of the approach is to test dark object segmentation for a range of thresholds. The profile of counts versus thresholds reveals the confidence in the guess through the stability of the count. The range of considered thresholds can be adapted by the base threshold parameter, typically once for a bunch of images captured in similar conditions.

In a preparatory study [10] the egg number estimated by image processing differed from visual inspection by about 3%. For sticks containing 500 eggs, the software reduced by about 4 the time spent to estimate the count.

In the tests that we performed with the images provided by ICount [5], we achieved a perfect estimation, with a high confidence (plateau). The images have an excellent quality and the eggs are about 25 pixel long. With our more challenging dataset, the estimation was in nearly all cases very close to the ground truth, showing a difference of up to 4 eggs for images containing between 20 and 80 eggs. However, even an observer has difficulties in counting eggs in some clusters of our dataset images. Also better estimates can be achieved by threshold refinement, checking the individual counts. The most difficult case concerns an image with 12-15 pixel long eggs, on a stick full of dark dots and darkened by rot, and with many blurred clusters. In this case, the refinement of the bounding box around the area with eggs was necessary, otherwise no plateau was present. For sticks with large dark spots with no egg, the only way to achieve a correct estimate is to edit the spots counts.

We are confident that a better accuracy can be achieved with larger image resolution (eggs of min 25 pixels long). Larger egg sizes can be captured by smartphones with closer shooting thanks to a removable lens. In any case, this current tool even if not yet highly accurate can reduce the operator time load and significantly enhance the response time to health authorities. It can also safeguard biological data otherwise lost.

5. CONCLUSIONS

For counting mosquito eggs on ovitraps, we first applied automatic bounding box detection, enclosing the stick to reduce false counts and fasten processing. Then we detected the darkest objects by thresholding and contour extraction, estimating the number of eggs in each object from its area. This was repeated for a range of threshold values to show the profile of count versus threshold. The program issues the probable count from the profile and the user decides from its shape to accept the value, to edit the bounding box, to check individual egg counts or edit them.

The tests with high quality egg images [5] and with a challenging low resolution dataset showed the adequacy for practical use. The solution can be made automatic for an acceptable image quality and friendly background if we can find a valid merit function to highlight the best counts among different thresholds. The storage of results as a count per image file with updating cases database correction. Adding the position and counts of individual objects opens perspective for new research.

6. ACKNOWLEDGMENTS

This research is conducted in collaboration with the EC-H2020 EOXPOSURE project, grant agreement No 734541 (http://www.h2020-eoxposure.eu). The internship of Charles Beumier was funded by EOXPOSURE and by the Belgian Ministry of Defence. Charles is grateful for the warm welcome and fruitful discussions at the Gulich Institute.

7. REFERENCES

[1] WHO, 2020. Dengue and severe dengue. URL: https://www.who.int/news-room/fact-sheets/detail/dengue-and-severe-dengue.
[2] PHAO, 2020. Actualizacion epidemiologica dengue y otras arbovirosis. Washington, D.C. OPS/OMS. URL: https://www.paho.org/es/file/66649/download?token=ysSF9lr.
[3] C. Zettel, P. Kaufman, “Yellow fever mosquito Aedes aegypti”, Parasites & Vectors (2016), 9:590.
[4] G. Gusmão, S. Machado, M.A. Rodrigues, “A new algorithm for segmenting and counting Aedes aegypti eggs in ovitraps”. In Principe JC Engineering in Medicine and Biology Society, 2009 EMBC 2009, Annual International Conference of the IEEE, Minneapolis, p.6714-6717.
[5] J. Gaburro, J.B. Duchemin, P. Paradkar, S. Nahavandi, A. Bhatti, “Assessment of ICount software, a precise and fast egg counting tool for the mosquito vector Aedes aegypti”, Parasites & Vectors (2016), 9:590.
[6] M. Chanampa, “Distribucion y abundancia de Aedes aegypti en la provincial de Salta: asociacion con factores ambientales”, PhD Thesis, Universidad Nacional de Cordoba, Argentina, 2019.
[7] V. Andreo, X. Porcasi, C. Rodriguez, L. Lopez, C. Guzman, C. M. Scavuzzo, “Time Series Clustering Applied to Eco-Epidemiology: the case of Aedes aegypti in Cordoba, Argentina.” XVIII Workshop on Information Processing and Control (RPIC) 2019
[8] G. Bradski, “The openCV library”, Dr Dobb’s J. Software Tools, 2000.
[9] C. Zettel, P. Kaufman, “Yellow fever mosquito Aedes aegypti”, https://edis.ifas.ufl.edu/pdfs/IN/IN79200.pdf, Feb 2009, reviewed March 2019.
[10] J. Rubio, V. Andreo, X. Porcasi, C. Guzmán, L. Lopez “Desarrollo de una aplicacion para el conteo de huevos en ovitrampas”, XIX Simposio Internacional de Enfermedades Desatendidas. Buenos Aires 2019.