FIGHTING BACK ZIKA, CHIKUNGUNYA AND DENGUE: DETECTION OF MOSQUITO-BREEDING HABITATS USING AN UNMANNED AERIAL VEHICLE

G. S. Andrade†, T. M. Dias†, V. C. Alves‡, H. M. Alves Junior†, L. F. Pinheiro†, P. M. C. Silva†, R. S. G. Pontes†+, B. P. Silva, I. B. O. de Macedo*, G. M. Araujo†, T. M. Prego* and A. A. Lima**

†Departamento de Engenharia de Controle e Automação ,
Centro Federal de Educação Tecnológica Celso Suckow da Fonseca - CEFET/RJ
‡Departamento de Engenharia de Produção,
Centro Federal de Educação Tecnológica Celso Suckow da Fonseca - CEFET/RJ
*Curso Técnico de Telecomunicações,
Centro Federal de Educação Tecnológica Celso Suckow da Fonseca - CEFET/RJ
+Diretoria de Tecnologia da Informação,
Colégio Pedro II - CPII

ABSTRACT

Every year, thousands of people die from diseases such as dengue fever, Chikungunya and Zika. This public health problem is more evident in developing countries. All these illnesses have the same source in common: a mosquito scientifically known as Aedes Aegypti. In cities, the majority of mosquito-breeding habitats are man-made: bottles, tires, barrels, pots or any stagnant water. This project proposes a system to aid in determining mosquito-breeding habitats location by employing computer vision tools on aerial images. Tires and regions with stagnant water were selected as objects of study. For this, a dataset was created containing video sequences and telemetry data from an Unmanned Aerial Vehicle (UAV) and the respective manual annotation in different scenarios. The features extracted from the videos (through HSV color space, histograms and edge detection) were used to train a Random Forest classifier, resulting in an accuracy of 99% in the test set. The system is also capable of automatically determine the GPS coordinates of the possible mosquito breeding location with enough precision.

1. INTRODUCTION

The *Aedes aegypti* mosquito is being considered the most dangerous animal in the world, as it is rapidly spreading, posing a threat not only to South America, but Europe as well [1]. Each year, about 725,000 people die from mosquito-borne diseases, while people that die from homicide are around 475,000 [2].

The *Aedes aegypti* is an urban agent of daytime habits, typical of tropical and subtropical regions. This species of mosquito spreads numerous dangerous diseases throughout the Brazilian population, amongst them dengue fever, yellow fever, Chikungunya and Zika [3]. The last one, particularly, is responsible for the blatant increase in cases of microcephaly throughout brazilian territory [4] [5].

Also, suspected and confirmed yellow fever cases have been reported in South American countries such as Brazil, Colombia, Ecuador, Peru, Bolivia and Suriname [6]. In Brazil, the disease had already been eradicated, but recent news point to another outbreak, due to recent environmental disasters [7]. The last informative report of Ministério da Saúde, from Brazil, relates 724 confirmed cases in country, with 237 deaths, between 1st of July 2017 and 28th of February 2018. And still, 785 cases more were registered in this period [8]. The proliferation of the *Aedes aegypti* can potentialize contamination in urban centers [9].

The use of automated machines capable of fighting back the mosquito [10] or identifying the *Aedes aegypti* hatchers [11] [12] would be very important in terms of public health, since it would enable identification of foci in a non-intrusive way, allowing faster coverage of much larger areas. Unmanned aerial vehicles are up to the task, as these are equipped with several sensors, as well as high resolution cameras, which can cover a wide range of functions, including identification of possible mosquito foci. The UAVs are capable of filming or photographing objects at great distances, also serving as patrol, search, rescue and monitoring operations in large urban centers. Electronic and computational centers, which act as their "brains" on ground, gives the vehicles the exclusive task of capturing data to be processed externally.

This work proposes a method to assist in elimination of *Aedes aegypti* breeding sites, using UAVs to capture possible mosquito foci images even in hard to reach areas, reducing human exposure to long, monotonous or dangerous tasks, as well as providing less economical expenses and granting environmental benefits like reducing insecticide emanation. Computer vision algorithms are used to identify the possible breeding sites within the annotated image database. All supporting materials, including slides, database and codes are available at https://goo.gl/D1zhxw.

2. THE TEAM

The team in charge of developing the project is composed by three advisors and nine students: seven college students (six students undergraduate from Control and Automation Engineering and one student undergraduate from Production Engineering) and two high school students attending telecommunications technical course.

The project advisors are Gabriel Matos Araujo, Amaro Azevedo de Lima and Thiago de Moura Prego. Both are D. Sc. in Electrical Engineering and professors at CEFET/RJ.

The team composition aims to gather students' knowledge in various areas to solve the main problem. In addition, it brings high school students closer to undergraduate students who had similar realities in high school and followed in Electrical Engineering and related areas.

The tasks were divided as follows: the high school students participated in gathering images, database annotation and developing a solution for georeferencing the database; while the college students focused on the machine learning techniques to automatically identify the possible mosquito-breeding habitats in gathered images.

The execution planning was created and coordinated by the production engineering student, as well as gathering bibliography and statistical analysis about this public health problem.

3. HARDWARE DETAILS

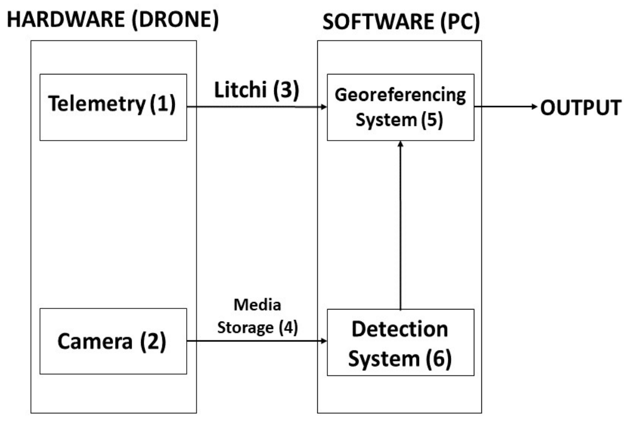

Fig 1: Hardware-Software diagram, showing how the UAV corresponds to the Detection System.

An UAV was used to capture several videos of tires, puddles and some other objects with water, like water tanks and pails at different heights, arrangements and environmental situations simulated at the campus of CEFET/RJ. For this a commercial UAV, model DJI Phantom Vision 2 Plus with 20 minutes of flight autonomy, was used.

For developing the intelligent system, it was needed to know some of the UAV's parameters like velocity, altitude and geographical positioning, so the team made use of the drone telemetry and the Litchi Android app, which will be covered in the following topics. Fig. 1 shows how the parameters collected by the hardware relate directly to the developed Software, which will be covered in Section 4.

3.1. Telemetry

The telemetry is obtained through the drone's flight controller. The flight controller returns real-time parameters in the DJI app. The main flight parameters are aircraft status, position, velocity and altitude, sensor information (Compass, IMU, Satellite positioning) and others [13]. The IMU (Inertial Measurement Unit), composed by an accelerometer and gyroscope, is used to measure linear acceleration and angular velocity [13], and was configured to adjust these parameters. The compass determines the heading of the aircraft relative to North; and satellite positioning relative to the aircraft body and relative to the ground [13]. Despite these useful information, the flight controller does not save these information, just show them in real-time, and that's why the Litchi app was used. Fig. 2 shows an internal view of the drone, highlighting the positioning of flight controller, board configuration, motor, Electronic Speed Control (ESC) and GPS.

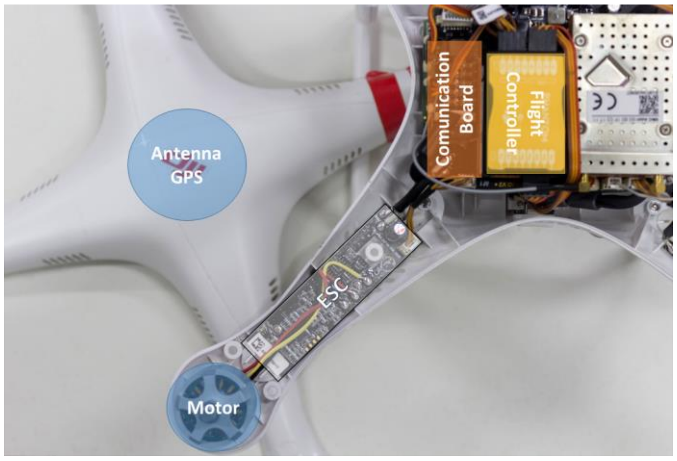

Fig 2: DJI Phantom 2 Plus drone hardware highlights.

3.2. Camera

The embedded camera has a resolution of 4384 x 3288 pixels. However, the videos were recorded using 1080p at 30 frames per second with the smallest FOV (Field of View) of 85 degrees to obtain less distorted videos [14]. The videos were made with the camera heading perpendicularly to the ground.

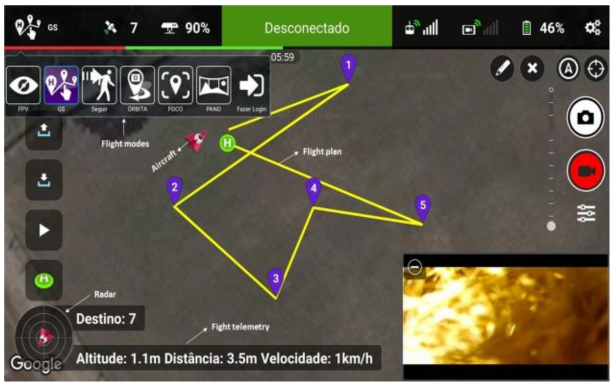

Fig 3: Litchi Android App interface showing flight plan.

3.3. Litchi app

The Litchi app is presented in Fig. 3, showing its capability to do an autonomous flight using a flight plan when it's operating in GPS flight mode. Litchi outputs a table with some intrinsic drone parameters including velocity, altitude, latitude, longitude among others. The GPS information is obtained from this table. However these parameters are sampled at approximately 10 samples/second, which is not enough to cover all frames provided in one video. Because of that, a linear interpolation is required in order to make a correspondence between the video frames and Litchi's table values.

3.4. Media storage

All collected data from DJI embedded camera was stored in SD cards. The dataset was then organized in specific folders to prepare for training and testing the detection system afterwards.

3.5. Georeferencing system

As mentioned in Section 3.3, the GPS signal from Litchi has a sample rate of 10 samples/second. Since the sample rate from the drone camera is 30 frames/second, linear interpolation was applied to GPS signal in order to associate a frame with a tire or puddle to a GPS coordinate. In this scheme, each frame has a respective coordinate.

It was decided to employ a linear interpolation because the GPS precision is not so high, besides we have chosen a constant low speed in a straight line trajectory for the drone in the flight plan, as can be seen in Fig. 3. So, we can assume that we have an approximate linear variation. Mathematically, for a linear variation in an interval [x_1 , x_2], the value of x in this interval can be obtained by the interpolation function $f(x)$ according to Eq. (1) [15].

$$f(x) = \frac{x - x_1}{x_2 - x_1} \cdot f(x_2) + \frac{x_2 - x}{x_2 - x_1} \cdot f(x_1). \qquad (1)$$

4. DETECTION SYSTEM

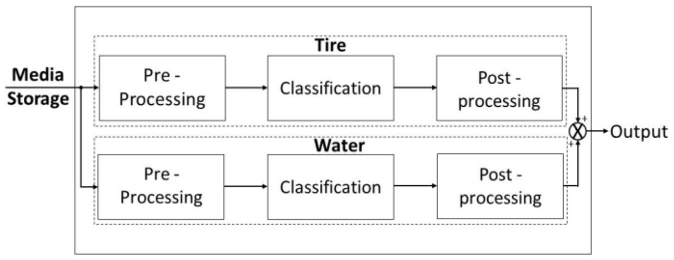

Fig 4: Developed Detection System Overview.

This section will cover methods and technologies used in the development of the detection system. The block diagram shown in Fig. 4 presents how the system works step by step, from reading the stored media to presenting video file with detected objects. Different methods were used for processing tires and water objects, as it can be seen in the schematics in Fig. 5 and Fig. 6.

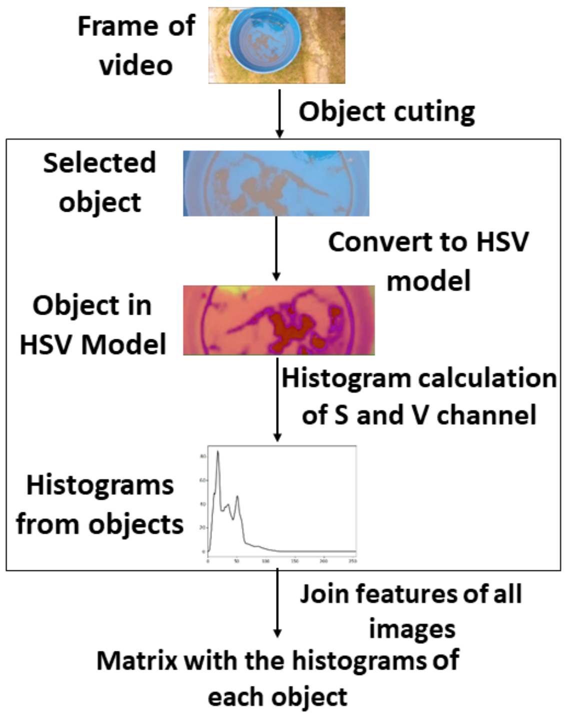
Frame of video
Object cuting
Selected object
Convert to HSV model
Object in HSV Model
Histogram calculation of S and V channel
Histograms from objects
Join features of all images
Matrix with the histograms of each object

Fig 5: Scheme that represents how the water detector algorithm works.

4.1. Construction of scenarios and acquisition of videos

The team worked on using an UAV to capture several videos of tires, puddles and some objects with water, like water tanks and pails at different heights, arrangements and environmental situations simulated at the campus of CEFET/RJ. For this it was used the commercial UAV described on the previous section. The team defined a model of flight plan with the following characteristics:

- A mobile app called Litchi was used to define the flight plan. In this application it is possible to mark some key-points to pre-establish the flight of the UAV. Then it follows the predetermined route, but usually cannot keep a perfect straight line because of strong winds.
- As so, the drone does take some time to align again with the route.
- The settings of the camera are configured manually, like the sensibility of the camera’s light sensor (ISO) and shutter release.
- The parameters also were set up. The altitude was determined to be constant throughout the course and the videos were recorded in three different altitudes: 5, 10 and 15 meters. The speed was set up at 7 km/h constant. Both parameters may vary along the course because of nature interference.

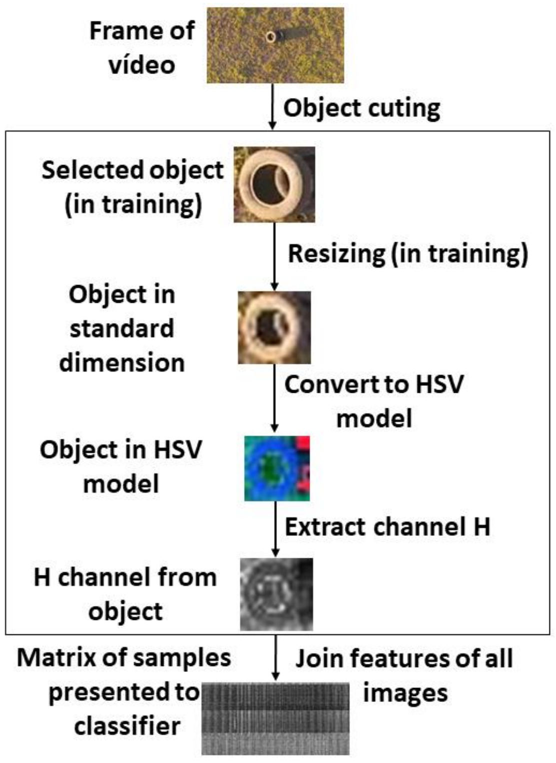

Fig 6: Scheme that represents how the tire detector algorithm works.

As a result, more than 19 gigabytes of videos in HD resolution were obtained. Fig. 7 shows an example of aerial image of randomly displaced tires taken using the UAV. These images were used to compose the database.

Every recorded video was subjected to the annotation process. For this task the software Zframer, developed at Laboratory of Signals, Multimedia and Telecommunications (SMT) of Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia (COPPE/UFRJ), was used. This software made possible to mark each object that is a possible breeding-place, so the detection system could be trained. In order to speed up the process, Zframer allows to interpolate annotations between frames. Fig. 8 shows the annotation process of tire images using Zframer.

The video annotations were saved in a text file (.txt) alongside with the video database that was read by the detection algorithm, composing the project dataset. This dataset, as well as other supporting material are available at https://goo.gl/D1zhxw.

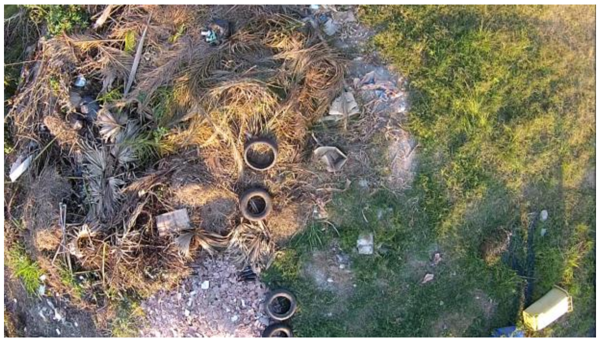
Fig 7: Aerial image of randomly displaced tires taken using an UAV

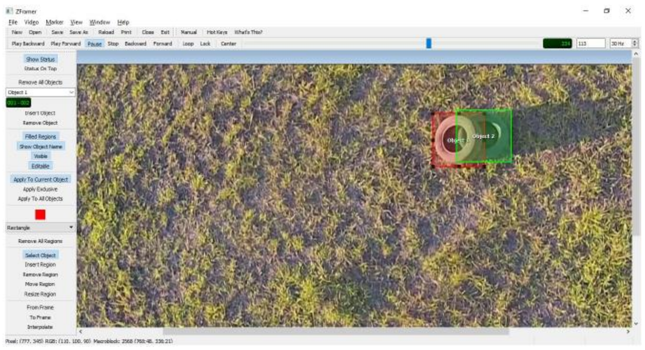
Fig 8: A video frame being annotated using Zframer

4.2. Pre-processing and feature extraction

4.2.1 Pre-processing for tire detection system

The pre-processing of tires consists in obtaining an image, the frame of a video from the dataset, and applying to it some techniques of image processing in order to get the features of the searched object. The steps of pre-processing are described in the following.

As a first action, the image border is removed in order to reduce the fish-eye effect caused by the drone's camera lens. Fish-eye cameras have a "super-wide" field-of-view (FOV) characteristic (e.g. cameras with field-of-view of up to 180 degrees). These lenses capture severe forms of distortion, the most evident being radial distortion. Several other distortions, such as uneven illumination and inaccurate estimation of the centre of distortion, should also be considered when using a fish-eye camera [16]. So, it is harder to detect objects in the border of such videos, where the distortions are more evident. For solving this problem, 15% of the video dimension was discarded in each border.

In order to obtain the training samples, the ground truth was used do cut all tires of all frames in the training set. These samples were rescaled to the same standard dimension of 20 x 20 pixels. Then these sample blocks are converted from RGB to HSV (Hue, Saturation and Value), a model that contains information about color, saturation and brightness of the pixel [17]. In the next step, only the Hue channel of image was considered, and this channel is reshaped to a vector with a dimension of 20^2 x 1. This operation is applied to all images and the result is joined in a matrix to train or classify the data.

4.2.2 Pre-processing for water detection system

The pre-processing to recognize water is based on a method proposed by [18]. Upon implementation of feature extraction, HSV was used. Each frame was converted from the RGB colorspace to HSV. This is so because the regions with water presents low saturation and high brightness values. As long as the water do not have a particular shape, it was decided to split the images into squares of 30 x 30 pixels in order to make the water detection process easier. Also, this size was determined based on the size of the video and because it makes possible to split the image into little blocks that could contain stagnant water. From each square obtained from the image, a histogram calculation was made in order to detect the intensity of every pixel in the channels S and V. The output of this calculation is an array concatenation of the two channels, which were used to feed the classifier and detect water spots, as shown in Fig. 5.

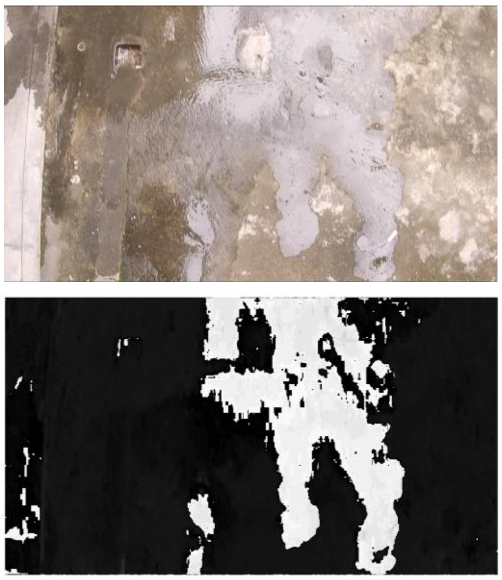

Fig 9: Water puddle converted to HSV.

4.3. Classification

4.3.1 Random Forest

Initially, several Machine Learning techniques were evaluated in our dataset to work as the data classifier. The best results were obtained by using a Random Forest algorithm. As illustrated in Fig. 10, Random Forest consists of a prediction algorithm that constructs an ensemble of prediction trees built with random vectors sampled independently with the same distribution for all trees in the Forest [19]. This combination of decision trees leads to models with high accuracy, stability and execution speed [20].

In this algorithm, the class of the instance is voted between all trees of the forest where each classifier contributes with a single vote for the assignation of the most frequent class, that is, the class of the instance [21].

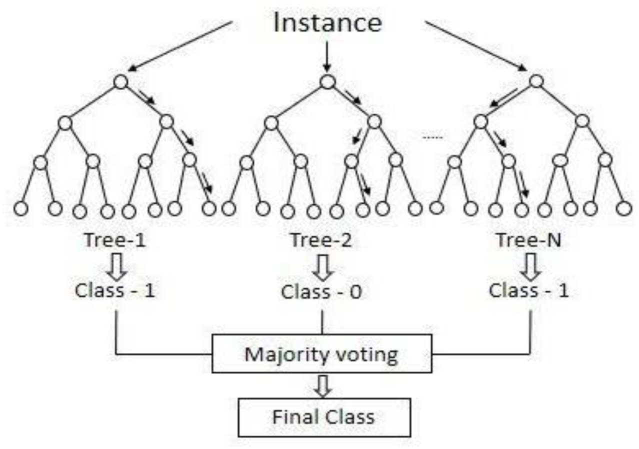

Fig 10: Decision tree forest scheme

4.3.2 Training

In training, 70% of the annotated videos from the constructed dataset were used. The classifier loads two arrays, one containing the values from feature space, and other containing the class label - a flag which says if that object is positive (a breeding-site), or negative (not a breeding-site). There are two different input arrays for the classifier, since there are two distinct methods for pre-processing data, but the training algorithm is the same. The number of trees is a free parameter of the algorithm. By setting it to T = 20, as shown in Fig. 11, it is possible to verify that the minimal number of trees T with the best results in the training set is around T = 5, as there is no considerable gain in using more than 5 trees.

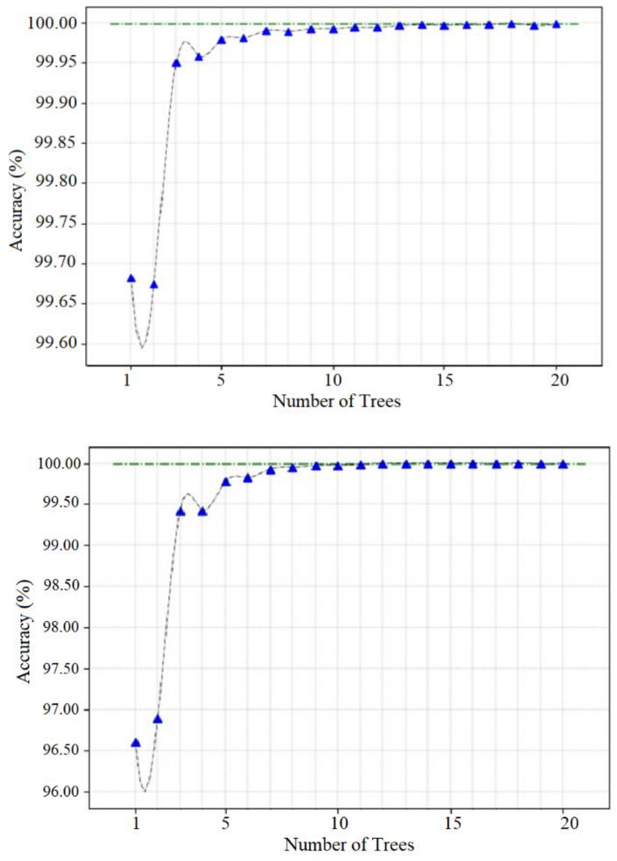

Fig 11: Training accuracy by number of trees. Green line depicts best training results. The first graph depicts Water Detector training and the second Tire Detector training.

4.3.3 Test

The remaining 30% of the annotated videos were used in test. These videos were submitted to the same feature extraction algorithms used in the training step. The samples in feature space feed the trained classifier, that can predict if the information is from a positive or a negative class.

4.4. Post-processing

After applying the classifier, some algorithms were used to execute correlation between objects and frames. Besides that, the ultimate steps are to apply inertia in object location over time and correlate the GPS coordinates with the video frames.

For applying the geometrical and temporal correlation as post-processing techniques, it is required to assume some premises about the system: (i) It is possible to model the signal with non-causal models; (ii) The movement of detected object is piecewise linear, and it is impossible for a real object to appear, disappear and reappear along time flow.

In the first step the correlation between detected objects in a frame is verified. To do so, the Euclidean distances between each pair of detections is calculated. If these distances are smaller than a threshold, it is considered that they belong to the same object and the average is applied to transform them in a single object.

In a second step, the correlation between objects in different and successive frames is verified. Euclidean Distance is also used, but in this case applied in a space with three dimensions. The rule applied to verify correlation between frames is the same as in step one.

In the next step, objects with less than thirty apparitions, that is, objects that appear for less than one second in the video, are discarded as false positive.

The fourth step consists of actions to add objects in a frame where there was a false negative. For this, a recursive function, whose objective is to determine the object location in the frame by average, applies. Considering n_1 and n_2 as frame indexes and $X(n)$ as the bounding box of the detected object, this recursive function can be obtained by employing the model of Eq. (2).

$$X\left[floor\left(\frac{n_1+n_2}{2}\right)\right] = ceil\left(\frac{X[n_1]+X[n_2]}{2}\right) \qquad (2)$$

In the next step, a technique was applied to reduce oscillations in the detected object position between frames and to smooth the frame transitions. Each bounding box is influenced by its last and future positions. This temporal correlation is obtained by using the model in Eq. (3).

$$X[n] = ceil\left(\frac{1}{5}\sum_{l=-2}^{2} X[n-l]\right) \tag{3}$$

The expected results of these techniques are to add non-detected objects between frames with detected objects and to correct oscillations over time.

After applying geometrical and temporal consistency between detected objects, the video frames are correlated with the GPS table stored in the dataset. After reading the table and calculating the interpolation, as described in Section 3.5, the system is able to extract latitude and longitude of the corresponding frame in which an object or a group of objects were identified. The system then uses the Gmplot python library to generate a map from the referenced region using Google Maps. The map is generated with a marker depicting the number of objects detected in that latitude and longitude keypoint.

5. CURRENT RESULTS

Concerning the image database, from all frames captured and annotated, 70% of the images were used for training and 30% for testing. The best training results for tire detection is using 5 trees, as shown in Fig. 11, as it is possible to verify that with 5 trees the accuracy is already above 99.95%. The same can be said about the water classifier.

Random Forest implementation for tire detection obtained 92.29% global accuracy and 86.21% of true positive rate. The water detector obtained above 90% global hits and the true positive rate was around 85%. Table 1 and 2 shows the confusion matrix for tire and water detection training, respectively.

	Predicted: NO	**Predicted: YES**	
ACTUAL: NO	TN = 1505	FP = 211	1716
ACTUAL: YES	FN = 25	TP = 1319	1344
	1530	1530	

Table 1: Confusion matrices for tire detection.

	Predicted: NO	**Predicted: YES**	
ACTUAL: NO	TN = 192632	FP = 32002	224634
ACTUAL: YES	FN = 11244	TP = 196588	207832
	203876	228590	

Table 2: Confusion matrices for water detection.

Fig. 12 shows an example of tire detection result depicting a red square around the tire and a map showing a marker of the actual geographic positioning where the specific frame was taken. The map also has a label with the number of objects detected in that region. Fig. 13 shows an example of water detection running in a frame of video depicting a water tank.

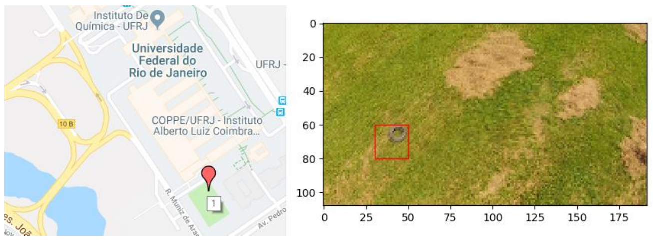
Instituto De Química - UFRJ
Universidade Federal do Rio de Janeiro
UFRJ
COPPE/UFRJ - Instituto Alberto Luiz Coimbra...
Av. Pedro
1
0
20
40
60
80
100
0
25
50
75
100
125
150
175

Fig 12: Tire detection frame, showing object geographic position and number of objects in label.

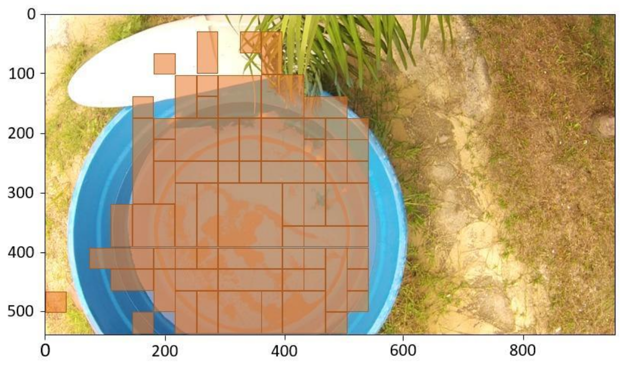

Fig 13:Water detection frame, showing detected water regions in a water tank.

The system was developed using Python, version 3. The OpenCV library was used to develop the pre-processing of water detector, because of its optimized algorithms for computer vision. In the pre-processing of the tire detection system, tools of Scikit Image library were used. The classifiers were developed using the Scikit Learn library, which contains efficient machine learning tools. In addition, some data science libraries, e.g. numpy and pandas, were also used.

6. FUTURE WORK

For future work, improving the accuracy of computer vision algorithms for object detection is suggested. New objects that can be a possible mosquito-breeding habitat could be identified, like bottles or vases. For achieving this, improving feature extraction and developing complimentary post-processing techniques would be necessary.

7. CONCLUSION

The team was able to develop an intelligent system to identify possible mosquito-breeding habitats in man-made objects. For achieving this objective, an image database consisting of various tire and water images in many different setups were collected using an UAV. This database was then properly annotated, and the sum of tire images collected thereafter were more than satisfactory to train and test the proposed system.

For developing the system, the computer vision technique known as Random Forest was used. Mosquito foci identifying and verification techniques were implemented, geographical positioning of the identified objects could be computed, and the overall obtained results were satisfactory.

For more details regarding this project, including access to a section of the database described in this article, visit https://goo.gl/D1zhxw.

8. REFERENCES

[1] M. Blasberg, H. Goos, and V. Hackenbroch, “Aedes Aegypti Mosquito Is World’s Most Dangerous Animal,” *SPIEGEL ONLINE*, 2016. [Online]. Available: http://www.spiegel.de/international/world/aedes-aegypti-mosquito-is-world-s-most-dangerous-animal-a-1103876.html. [Accessed: 29-Nov-2017].

[2] Bill Gates, “The Deadliest Animal in the World,” *Gatesnotes*, 2014. [Online]. Available: https://www.gatesnotes.com/Health/Most-Lethal-Animal-Mosquito-Week. [Accessed: 29-Nov-2017].

[3] N. Oliveira, “Aedes aegypti: conheça a história do mosquito no Brasil e suas características,” *EBC*, 2015. [Online]. Available: http://agenciabrasil.ebc.com.br/geral/noticia/2015-12/aedes-aegypti-conheca-historia-do-

mosquito-no-brasil-e-suas-caracteristicas. [Accessed: 29-Nov-2017].

[4] A. G. London, “Zika virus is a global public health emergency, declares WHO,” *BMJ*, vol. 352, 2016.

[5] M. Lahorgue Nunes *et al.*, “Microcephaly and Zika virus: a clinical and epidemiological analysis of the current outbreak in Brazil,” *J. Pediatr. (Versão em Port.*, vol. 92, no. 3, pp. 230–240, 2016.

[6] World Health Organization, “PAHO WHO | Yellow fever,” *Pan American Health Organization*, 2017. [Online]. Available: http://www.paho.org/hq/index.php?option=com_topics&view=article&id=69&Itemid=40784&lang=en. [Accessed: 29-Nov-2017].

[7] “Metade dos casos de febre amarela está na região afetada pela lama da Samarco — Rede Brasil Atual,” *Revista do Brasil*, 2017. [Online]. Available: http://www.redebrasilatual.com.br/revistas/126/metade-dos-casos-confirmados-de-febre-amarela-esta-na-regiao-afetada-pela-lama-da-samarco. [Accessed: 29-Nov-2017].

[8] Ministério da Saúde, “Febre amarela: Ministério da Saúde atualiza casos no país,” 2018. [Online]. Available: http://portalms.saude.gov.br/noticias/agencia-saude/42655-febre-amarela-ministerio-da-saude-atualiza-casos-no-pais. [Accessed: 04-Mar-2018].

[9] C. I. Paules and A. S. Fauci, “Yellow Fever — Once Again on the Radar Screen in the Americas,” *N. Engl. J. Med.*, vol. 376, no. 15, pp. 1397–1399, Apr. 2017.

[10] E. Ackerman, “Drones Distribute Swarms of Sterile Mosquitoes to Stop Zika and Other Diseases,” *IEEE Spectrum*, 2017. [Online]. Available: https://spectrum.ieee.org/robotics/drones/drones-distribute-swarms-of-sterile-mosquitoes-to-stop-zika-and-other-diseases. [Accessed: 29-Nov-2017].

[11] “Vistoria aérea com drone para combater a dengue,” 2016. [Online]. Available: http://www.dronefilmagemaerea.com/vistoria-aerea-com-drone-para-combater-a-dengue/. [Accessed: 29-Nov-2017].

[12] “Drones na guerra contra o Aedes,” *Revista Ecológico*, 2016. [Online]. Available: http://www.revistaecologico.com.br/materia.php?id=98&secao=1700&mat=1943. [Accessed: 29-Nov-2017].

[13] “Onboard SDK Documentation Home”. [Online]. Available: https://developer.dji.com/onboard-sdk/documentation/introduction/homepage.html. [Accessed: 29-Apr-2018].

[14] DJI. Phantom 2 Vision+ User Manual. V1.8. 2015.01.

[15] Pownuk, A. 2017. Why Linear Interpolation?, 43.

[16] C. Hughes, M. Glavin, E. Jones and P. Denny, "Review of geometric distortion compensation in fish-eye cameras," IET Irish Signals and Systems Conference (ISSC 2008), Galway, 2008, pp. 162-167. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4780947&isnumber=4780919

[17] C. Jos, D. Alamo, L. Arnaldo, R. Calla, W. Roberto, and R. Lov, “A novel approach for image feature extraction using HSV model color and filters wavelets,” XXXIX Lat. Am. Comput. Conf., pp. 1–7, 2013.

[18] M. G. Prasad, A. Chakraborty, R. Chalasani, and S. Chandran, “Quadcopter-based stagnant water identification,” 2015 5th Natl. Conf. Comput. Vision, Pattern Recognition, Image Process. Graph. NCVPRIPG 2015, pp. 1–4, 2015.

[19] L. Breiman, “Random Forests,” *Mach. Learn.*, vol. 45, pp. 5–32, 2001.

[20] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York, NY: Springer New York, 2009.

[21] V. F. Rodriguez-Galiano, B. Ghimire, J. Rogan, M. Chica-Olmo, and J. P. Rigol-Sanchez, “An assessment of the effectiveness of a random forest classifier for land-cover classification,” *ISPRS J. Photogramm. Remote Sens.*, vol. 67, pp. 93–104, 2012.