IoT–Based System for Real-time Monitoring and Insect Detection in Vineyards

Sintija Stevanoska†
Faculty of Computer Science and Engineering
Ss. Cyril and Methodius
Skopje, North Macedonia
sintija.stevanoska@gmail.com

Danco Davcev
Faculty of Computer Science and Engineering
Ss. Cyril and Methodius
Skopje, North Macedonia
danco.davcev@finki.ukim.mk

Elena M. Jovanovska
Faculty of Computer Science and Engineering
Ss. Cyril and Methodius
Skopje, North Macedonia
jovanovska.elena14@gmail.com

Kosta Mitreski
Faculty of Computer Science and Engineering
Ss. Cyril and Methodius
Skopje, North Macedonia
kosta.mitreski@finki.ukim.mk

ABSTRACT

The Internet of Things (IoT) is a relatively new concept with a number of potential uses in agriculture. In this work we propose a system based on IoT for early detection of wine moth infestation, as well as monitoring the number of pests at a particular part of the season. This leads to optimization of the use of pesticides in the vineyards. The wine moths are caught using a pheromone trap with a camera attached to it. The camera is used for real-time monitoring of the trap. The output of the program is an image that indicates how many pests are currently caught on the trap. We have implemented a prototype in one of the vineyards in a Macedonian winery near Skopje City.

KEYWORDS

Internet of Things (IoT), Real-time monitoring system, Pests, Pesticides, Image Recognition, Vineyards

ACM Reference format:
Sintija Stevanoska, Elena M. Jovanovska, Kosta Mitreski, Danco Davcev. IoT–Based System for real-time monitoring and Insect Detection in vineyards. *MobiWac'20,* November 16–20, 2020, Alicante, Spain © 2020 Association for Computing Machinery. ACM ISBN 978-1-4503-8119-2/20/11, https://doi.org/10.1145/3416012.3424634

*IoT -Based System for Real-time Monitoring and Insect Detection in Vineyards
†Sintija Stevanoska

MobiWac'20, November 16–20, 2020, Alicante, Spain
©2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8119-2/20/11...$15.00
https://doi.org/10.1145/3416012.3424634

1 Introduction

The IoT has evolved with the development of wireless technologies, micro-electro-mechanical systems, micro-services and the Internet. It enables real-time overview of the system's state, enabling those responsible for the systems to react in a timely manner. It helps in so many different areas to ease our lives from helping the elderly using sensors that monitor and notify if they fall [20] to improvements in agriculture using sensors for optimizing the use of pesticides and save the crops and soil form harm [2].

The agricultural sector is one of the most important sectors throughout the entire history of humankind, largely impacting the quality of life. As such, this sector needs to be resilient to external factors, and that's exactly where ICT can be used - to solve a large portion of the problems that farmers face. Consequently, agronomy is one of the sectors that can benefit from utilizing IoT solutions. Besides being used for soil, animal and plant monitoring, IoT can be applied for resource rationalization and monitoring of the negative impact agriculture has on the environment.

Precision Farming is a modern approach that uses IoT technologies to optimize resources, in order to obtain high grape yield while reducing operational costs [7].

IoT can transform many aspects of today's agriculture through: **Sensor-based data collection**-sensor systems are the most important part of this approach. **Agricultural drones**. **Smart Greenhouse**-involves the use of IoT technologies to automatically control the climate inside a greenhouse, without the need of human intervention. **Predictive analysis for smart farming**-the key player in precise agriculture.

Data collected through sensors is used to perform predictive analysis. Forecast data include parameters such as soil and air moisture, temperature, solar radiation, amount of rainfall and pressure.

In this work, we propose an IoT-based system for data collection through several cameras placed in the vineyard. The camera is attached next to a pheromone trap and it is used to detect the insects caught on the trap. Object detection is done by a machine vision technique that locates object instances in digital images or videos [3]. The goal of automated object detection is to replicate human intelligence: when humans look at an image, they can recognize objects in a matter of seconds.

In section 2 we explain several related works relevant to our work, in section 3 we describe the proposed system for early detection of wine moth infestation, as well as monitoring the number of pests. The implementation of the prototype was in a Macedonian winery near Skopje City and it is presented in section 4. Section 5 concludes the paper.

2 Related Work

In order to understand the necessities for improvement and similar projects done in Precision Farming, we reviewed several related works. There are several approaches to object detection on images. Agricultural monitor of disease and insect pests is a very complicated process, and involves many different phases and different actors. Every phase involves many kinds of operation, and every operation involves many technologies, includes data collection, image processing, data mining and fusion, pattern recognition, etc [2].

A recent investigation has illustrated IoT based Borer insect detection in tomatoes in India. The authors have used a robot attached to a wireless web camera and Azure cloud service. The web camera takes video of tomato plantation area in real-time and sends the video data to the Java enabled Software-as-aService (SaaS) where unripe tomato and border detection is done. The information is then processed by the data base stored at the Azure cloud for matching with appropriate pesticide amalgamation [16].

In another paper, an automatic method for monitoring pests from trap images is described. It uses a sliding window-based detection pipeline, where a convolutional neural network is applied to image patches at different locations to determine the probability of containing a specific pest type. Image patches are then filtered by non-maximum suppression and thresholding, according to their locations and associated confidences, to produce the final detections. Qualitative and quantitative experiments demonstrate the effectiveness of the proposed method on a codling moth dataset [17].

Sciarretta and Calabrese [19] discuss monitoring systems that can be equipped with software for image interpretation and identification of the caught target insects (fully automated system) or a semi-automated approach where a remote operator can count the trapped insects by watching the images coming from the e-trap. They indicate that the software can integrate a decision support system (DSS) module, which provides information on the risk of infestation and the actions to be taken.

The novelty of our work is that a high degree of automation has been achieved, with increased the precision in relation to the above stated solutions [19].

3 Problem Description and Proposed Solution

In this section we describe the problem with pests in the vineyard and the placement of pheromone traps using a real-time camera for automatic recognition of the number of pests in the image. The system in this solution includes pheromone traps, a camera for monitoring the trap and a script for detecting the number of insects caught in the trap.

The traps, described in greather detail below, serve as an indicator of the approximate number of pests in the vineyard. The goal is to optimize pesticide spraying both in quantity in single spraying and in the number of times the vineyards are sprayed. There are several different types of grape pests. However, only two of them are of interest in this vineyard:

1. Lobesia botrana [9] - a pest also known as European grape moth. The symptoms of Lobesia botrana infestation depend on the period of infestation.
2. Eupoecilia ambiguella [6] - this pest can be found in Europe and some parts of Asia, most commonly in the same locations as Lobesia botrana, and infests the vineyards in a very similar fashion. The biggest problem is the punctures these larvae make in the grapes. Once damaged, the grape can easily rot.
3. Gray mold (Botrytis Cinerea) – is the most common grape disease. Grapes that have gray mold are very hard to use in wine production, and they definitely cannot be used as table grapes.

This family of pests is considered a serious worldwide problem. Great efforts are being made to find a way to prevent and destroy grape moths in the most organic way possible, without the use of chemical pesticides. So far, the greatest success has been achieved with pheromone traps.

Several pheromone traps are placed in the vineyard. A pheromone trap is an insect trap that utilizes pheromones to attract a specific type of insect. Pheromones are chemical substances that animals use when communicating with each other [5]. Constant monitoring of this number can help in the prevention of pest infestations, thereby directly minimizing the damage done in agriculture.

These traps have to be strategically placed throughout the fields. However, the number of pests caught on the traps does not necessarily reflect the damage already done to the field [8]. Improper handling and placing of the traps can reduce the number of insects caught. By using pheromone traps, farmers can effectively plan and perform pesticide spraying. Moreover, they have realistic and localized data about the insect population in a given time of the year (first appearance, greatest numbers, expectance of the highest population numbers etc.).

The traps used in this solution are from CSalmon. These are called “delta” traps with a sticky tape and a pheromone bait. This type of pest trap is used when the end goal is to detect the appearance of the insects as early as possible. On smaller plantations, it can be used as the only way of pest control. On larger plantations, as this case, the trap is not powerful enough to serve as pest control, but it is rather used as an indicator when planning optimal pesticide spraying times.

The detection of trapped insects is done on images sent in real-time by a camera placed on the grape plant. The camera used is Dahua Consumer G26, with 2Mpx, it is waterproof and reaches a maximum of 30fps @1080p. The camera should be immobile, such that weather conditions would affect its position as little as possible. Protection against adverse weather conditions, should be

taken into account as well. The code for detection, its functions, structure and algorithms are described in the next section.
The ultimate goal is to eliminate manual count of insects caught on each of these pheromone traps with automatic recognition of the number of pests caught in the image. Please see Fig. 1 for a visual representation of the data set.

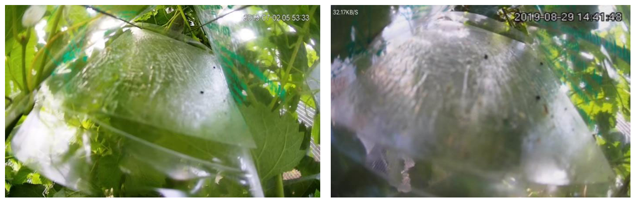
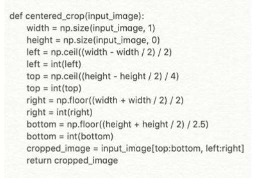


Figure 1: Image taken in the morning and afternoon

4 Implementation of the prototype

The data set is consisted of 50 images. Each image is a screenshot of what the camera was recording at a given moment. When the data set was created, special attention was drawn on introducing as much as possible diversity: images from different parts of the day, recorded from different angles, different weather conditions etc. The data was collected via real-time camera recording transmission, using the "Imou" application. The entire object recognition code was written in Python. The following text describes the main code blocks from the script.

There is a function for applying a centered crop to the source image serves to cut the unnecessary image parts. Namely, only the central part of the image should be taken into account, and all noise should be removed. The function has one input parameter - the image to pre-process. The width and height of the original image are extracted, and the right, left, top and bottom points of the cropped image are calculated as shown on Fig. 2. Then, the new dimensions of the pictures are set accordingly. The output image has the area of interest placed in its center.



```
def centered_crop(input_image):
    width = np.size(input_image, 1)
    height = np.size(input_image, 0)
    left = np.ceil((width - width / 2) / 2)
    left = int(left)
    top = np.ceil((height - height / 2) / 4)
    top = int(top)
    right = np.floor((width + width / 2) / 2)
    right = int(right)
    bottom = np.floor((height + height / 2) / 2.5)
    bottom = int(bottom)
    cropped_image = input_image[top:bottom, left:right]
    return cropped_image
```

Figure 2: Screenshot of the crop image function

Next, setting the parameters and initializing a Blob Detector: Blob is a group of linked pixels in some image that share a common feature, which separates that blob of pixels from the surrounding pixels. Blob Detection is a common choice when extracting a region that clearly differs from the rest of the image. OpenCV has a class for detecting such areas. It is called SimpleBlobDetector [1]. The basic algorithm of the detector is quite simple and has only several parameters.

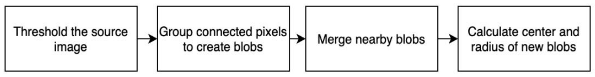
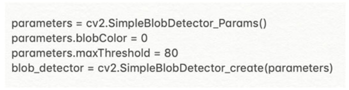

Threshold the source image
Group connected pixels to create blobs
Merge nearby blobs
Calculate center and radius of new blobs

Figure 3: A Block Diagram illustrating the steps of blob detection.

SimpleBlobDetector (as shown on Fig. 3) performs the following steps when extracting a blob [10]:

1. **Thresholding** - converts the original image to several binary images with a limit value set through the minThreshold parameter. Parameter values are incremented until the maxThreshold value is reached.
2. **Grouping** - in each binary image, the connected pixels are grouped, i.e. the blobs are assembled.
3. **Merging** - the blob centers are calculated and all the groups that have centers located at a distance, less than or equal to a preset minimum distance, are merged.
4. **Calculate** the center and radius of the new blobs. The parameters have been set as shown on Fig. 4:



```
parameters = cv2.SimpleBlobDetector_Params()
parameters.blobColor = 0
parameters.maxThreshold = 80
blob_detector = cv2.SimpleBlobDetector_create(parameters)
```

Figure 4: A Screenshot of the parameters set for calculation

The resulting blobs can be filtered by [10]: color, size, circularity, convexity and inertia ratio.
Iterating through the images in the data set: the image data set is located in a directory, which is iterated with a loop function.

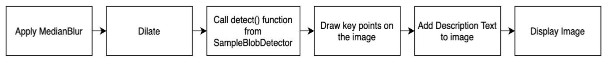

Apply MedianBlur
Dilate
Call detect() function from SampleBlobDetector
Draw key points on the image
Add Description Text to image
Display Image

Figure 5: A Block Diagram illustrating what happens during one iteration of the image data set.

One iteration follows the steps (please refer to Fig. 5 for the block diagram) as presented below:

1. The image is loaded in its original color space.
2. The function for making a centered crop (described above) is called.
3. Pre-processing of the image:

a) **Apply Median Blur** [11] - smoothing the image by using a 5px medial filter:
img = cv2.medianBlur(img, 5)
Median filtering is a non-linear technique, used to get rid of the unnecessary noise in images.

b) **Dilatation** [4] - a morphological operator, very common in image processing:
img = cv2.dilate (img, kernel, iterations=1)
Morphological operations are a group of functions that process an input image based on the shapes in it. This work uses a kernel of size 1, 1 in order to minimize the black spots on the images that are noise (dust, small raindrops).

c) This detect function **SimpleBlobDetector** returns the key-points in the image.
keypoints = blob_detector.detect(img)

d) The **key-points** are drawn on the image:
image_with_keypoints=cv2.drawKeypoints(img, keypoints, image_with_keypoints, [0, 0, 255], cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)

e) **Description text** is added to the key-point image to indicate how many points of interest (insects) are found in the image.

f) Both the original and key-point images are displayed.

The following Python libraries are used within this solution: **OpenCV** (Open Source Computer Vision Library) [13] - an open source library designed for machine vision and machine learning, mostly used for real-time applications.

Numpy (version 1.15.4) [12] is an open source library, which provides support for mathematical calculations and large n-dimensional matrices.

OS [15] is a module for using functionalities that are dependent on the operating system, such as loading, listing and modifying files and directories.

The script output consists of all the files obtained by the processing explained above, i.e. the images on which points of interest (insects) are marked. The output images have a description text indicating the number of detected insects printed on them. Please see Fig. 6 for an example output image.

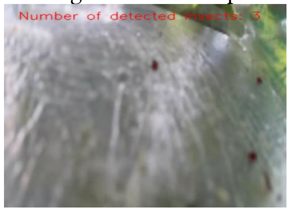


Figure 6: Example output of the script, number of detected insects is shown, and the insects are clearly marked.

The accuracy on the photos is 100% when the camera is set at a proper angle and the light is satisfactory, meaning that the number of insects detected by the algorithm is the same as the number of insects detected by a human. These photos comprise 90% of the dataset. On the subpar images (10% of the images in the dataset), the algorithm miscounts with 30% error, meaning it has accuracy of 70%. Greater attention should be paid to the quality of the images, because the algorithm's output is only as reliable as the input data set.

5 Conclusion

This work should be considered as a proof of concept. It has great potential, provided the appropriate equipment. Tracking the number of moths can be in real-time. The script can be called at user request or at predefined times of the day. The user will be able to view the results manually, or a notification trigger can be set if the critical number is exceeded. Considering future work, if the data set allows it and the image quality is good enough, machine learning algorithms can be implemented. The main disadvantage in this solution is that there is no proper training set from which the algorithm can learn how to classify the insects by type - there is still a need of human experts to classify insects into subspecies. There are some data sets for moth recognition [14]. These moths have similar physiognomy as the wine moths of interest here, but a thought should be put into how this would affect the accuracy of the classification [18]. Moreover, we can experiment with different types of pheromone traps, where the camera is placed directly above.

REFERENCES

[1] OpenCV Blob Detection Documentation Reference, updated on Jun 01, 2020.

[2] Y. Shi, Z. Wang, X. Wang, S. Zhang. Internet of Things Application to Monitoring Plant Disease and Insect Pests. *International Conference on Applied Science and Engineering Innovation* (ASEI 2015), doi: https://doi.org/10.2991/asei-15.2015.7 (2015)

[3] S. Dasiopoulou, Knowledge-assisted semantic video object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(10):1210–1224 (2015).

[4] Dilatation, https://docs.opencv.org/2.4/doc/tutorials/imgproc/erosiondilatation/erosiondilatation.html, last update 2019/12/31.

[5] A. Douglas. Pheromones - exploiting an insect's sense of 'smell'. NY FOREST OWNER 35: 4, JUL/AUG 1997 20.

[6] M.C. Epstein, T.M. Gilligan and S.C. Passoa, Screening aid: European grape berry moth, eupoecilia ambiguella (Hubner). *Identification Technology Program (ITP), USDA-APHIS-PPQ-ST, Fort Collins, CO. 80526 USA* (2014).

[7] F. J. Pierce and P. Nowak. Aspects of precision agriculture (1999).

[8] D. Johnson. Using pheromone traps in field crops (a practical cheat sheet)., U. S.Department of Agriculture, KENTUCKY COUNTIES COOPERATING (1994).

[9] A. Lucchi and P. L. Scaramozzino. Invasive species compendium: Lobesia botrana - European grapevine moth (2018).

[10] S. Mallick. Blob Detection Using OpenCV (Python, C++), (2015).

[11] Medianblur: https://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html?highlight=medianblur, last update 2019/12/31.

[12] Numpy: https://numpy.org/, 2019-2020 NumPy. All rights reserved.

[13] Open CV: https://opencv.org/, last update 2019/12/31.

[14] Open Images Dataset V6: https://storage.googleapis.com/openimages/web/visualizer/index.html, last update 26th February 2020.

[15] OS: https://docs.python.org/3/library/os.html, last update on Jun 01, 2020.

[16] P. P. Ray. Internet of things for smart agriculture: Technologies, practices and future direction. *Journal of Ambient Intelligence and Smart Environments 9* (2017) 395–420, DOI 10.3233/AIS-170440.

[17] W. Ding, G. Taylor. Automatic moth detection from trap images for pest management. Computers and Electronics in Agriculture. arXiv: 1602.07383v1 [cs.CV] 24 Feb **2016**.

[18] M. Cardim, F. Lima. Automatic Detection and Monitoring of Insects Pests, Agriculture **2020**, 10, 161; doi:10.3390/agriculture10050161.

[19] A. Sciarretta*, P. Calabrese. Development of Automated Devices for the Monitoring of Insect Pests. Agriculture Research Journal, ISSN: 2347-4688, Vol. 7, No. (1) **2019**, pp. 19-25.

[20] S. Chouali, A. Mostefaoui, M. Fayad, S. Benbernou. Fall detection application for the elderly in the Family Heroes System. *MobiWac '19*: Proceedings of the 17th ACM International Symposium on Mobility Management and Wireless Access, isbn:9781450369053, doi:10.1145/3345770, **2019**, pp17-23.