**UNIVERSITI PUTRA MALAYSIA**

**FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY**

**DEPARTMENT OF MULTIMEDIA**



**SKM 4213 - 1 : DIGITAL IMAGE PROCESSING (REPORT)**

**GROUP : 1**

**COURSE CODE : SKM 4213 - 1**

**COURSE NAME : DIGITAL IMAGE PROCESSING**

**LECTURER NAME : DR. HIZMAWATI BINTI MADZIN**

| **STUDENT NAME** | **MATRIC NUMBER** |
| --- | --- |
| MUHAMMAD AMIRUL AFIQ BIN JUMA’AT | 205911 |
| NIK MUHAMMAD ASYRAF BIN NIK ISMAIL | 206630 |

**Pest Detection Techniques: Report**

*Muhammad Amirul Afiq Bin Juma’at*

*Student*

*Faculty of Computer Science and Information Technology,*

*University Putra Malaysia (UPM)*

*Selangor, Malaysia.*

205911@student.upm.edu.my

*Nik Muhammad Asyraf Bin Nik Ismail*

*Student*

*Faculty of Computer Science and Information Technology,*

*University Putra Malaysia (UPM)*

*Selangor, Malaysia.*

206630@student.upm.edu.my

**Abstract-Malaysia is blessed with a land that is suitable for planting plants such as palm trees, paddies, rubber trees and others. In fact, Malaysia is the second largest palm oil producer and exporter in the world. Therefore, researchers have conducted various solutions with the help of deep learning techniques that follow the advancement of technology nowadays such as image processing solely to detect pests damaging our plant before it gets out of hand. For example, we can use image acquisition to capture an image then run on through phases of segmentation, classification and others. In this literature review,we aim to differentiate all the methods by discussing the research background of our scope first and then analyzing the methods with listing out their advantages and disadvantages.**

**Keywords- Pest detection, image processing, acquisition,segmentation, classification.**

1. **INTRODUCTION**

The art and science of cultivating the land, growing crops, and raising livestock is known as agriculture. It entails preparing plant and animal goods for human consumption as well as their delivery to markets. Agriculture was the beginning of civilization, and agriculture is still highly essential today, despite the fact that mankind has evolved greatly. Its importance is more visible in certain nations, but the fact is that agriculture is important to every country in the globe for one reason or another. Agriculture is crucial since it is the country's primary source of raw resources, a component of international trade, and a contributor to its growth.

However, one of the most deadly organisms capable of destroying the entire agricultural process will inevitably find its way into the field. Pest species are a major source of worry, not only because of the potential loss of money due to crop damage, but also because, if left untreated, they can severely destroy machinery, equipment, and property. Insects, birds, and rodents are examples of pest species.

Image processing technologies can help in this situation. The goal of this literature review is to look at the methods utilized by researchers and computer scientists to design an algorithm for detecting insect presence in the agriculture sector. The purpose of object detection in computer vision is to determine not only whether items from generic categories are present in a picture, but also to return the spatial location and extent of each object using a bounding box [1].

1. **RESEARCH BACKGROUND**
2. *“GrabCut”—Interactive Foreground Extraction using Iterated Graph Cuts*

Carsten Rother, Vladimir Kolmogorov and Andrew Blake who wrote about using GrabCut to accomplish a great quality of output like a precise segmentation to differentiate the foreground and background with only a tiny effort from the user. There was a method found which is similar to GrabCut but it has a flaw which cannot determine the background and foreground if both colors are identical.

First, they use iterative graph cutting to create a "hard" segmentation. The next step is border matting, which involves computing alpha values in a small area surrounding the hard segmentation boundary. Finally, GrabCut does not address complete transparency, other than at the border. The matting brush, on the other hand, might be used to do it, and in our experience, this works effectively in places that are sufficiently free of camouflage.

Their method's originality initially manifests itself in how segmentation is handled. Iterative estimates and incomplete labeling, two improvements we added to the graph cuts process, allow for significantly less user engagement for a given level of result quality. As a result, the user's engagement with GrabCut is minimal and only requires dragging a rectangle around the target item. By doing this, the user is identifying a background area without having to also designate a foreground area. In order to eliminate visual artifacts, we have created a new approach for regularizing alpha values, which is utilized for border matting. Figure 1 depicts the examples of GrabCut.

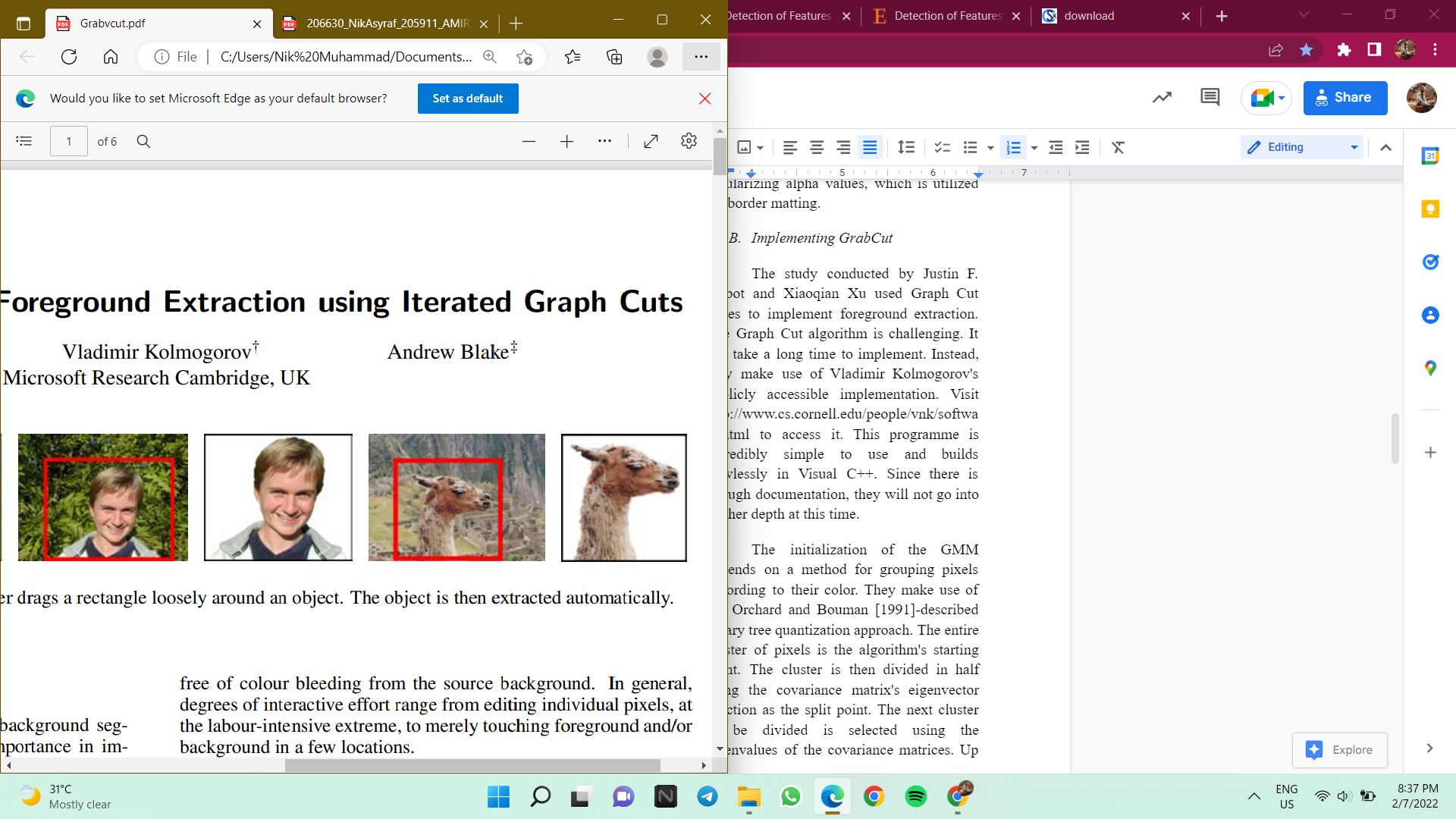


Figure 1: Three examples of GrabCut . The user drags a rectangle loosely around an object. The object is then extracted automatically.

1. *Implementing GrabCut*

The study conducted by Justin F. Talbot and Xiaoqian Xu used Graph Cut codes to implement foreground extraction. The Graph Cut algorithm is challenging. It can take a long time to implement. Instead, they make use of Vladimir Kolmogorov's publicly accessible implementation. Visit http://www.cs.cornell.edu/people/vnk/software.html to access it. This programme is incredibly simple to use and builds flawlessly in Visual C++. Since there is enough documentation, they will not go into further depth at this time.

The initialization of the GMM depends on a method for grouping pixels according to their color. They make use of the Orchard and Bouman [1991]-described binary tree quantization approach. The entire cluster of pixels is the algorithm's starting point. The cluster is then divided in half using the covariance matrix's eigenvector function as the split point. The next cluster to be divided is selected using the eigenvalues of the covariance matrices. Up until the appropriate number of clusters is reached, this process is repeated. For sizable clusters with Gaussian distributions, it is the ideal answer.

They must calculate the mean, the inverse of the covariance matrix, the determinant of the covariance matrix, and the weighting value for each of the Gaussian components. It is simple to calculate the mean and covariance matrix from the pixel colors. The covariance matrix makes it simple to calculate the determinant. By locating the cofactor matrix and dividing it by the determinant, the inverse of the covariance matrix may be calculated. Simply put, the weight,, is the percentage of pixels in the foreground (or background) that were given to this Gaussian component. As shown in Figure 2, the results of the iterative segmentation algorithm on two natural images.

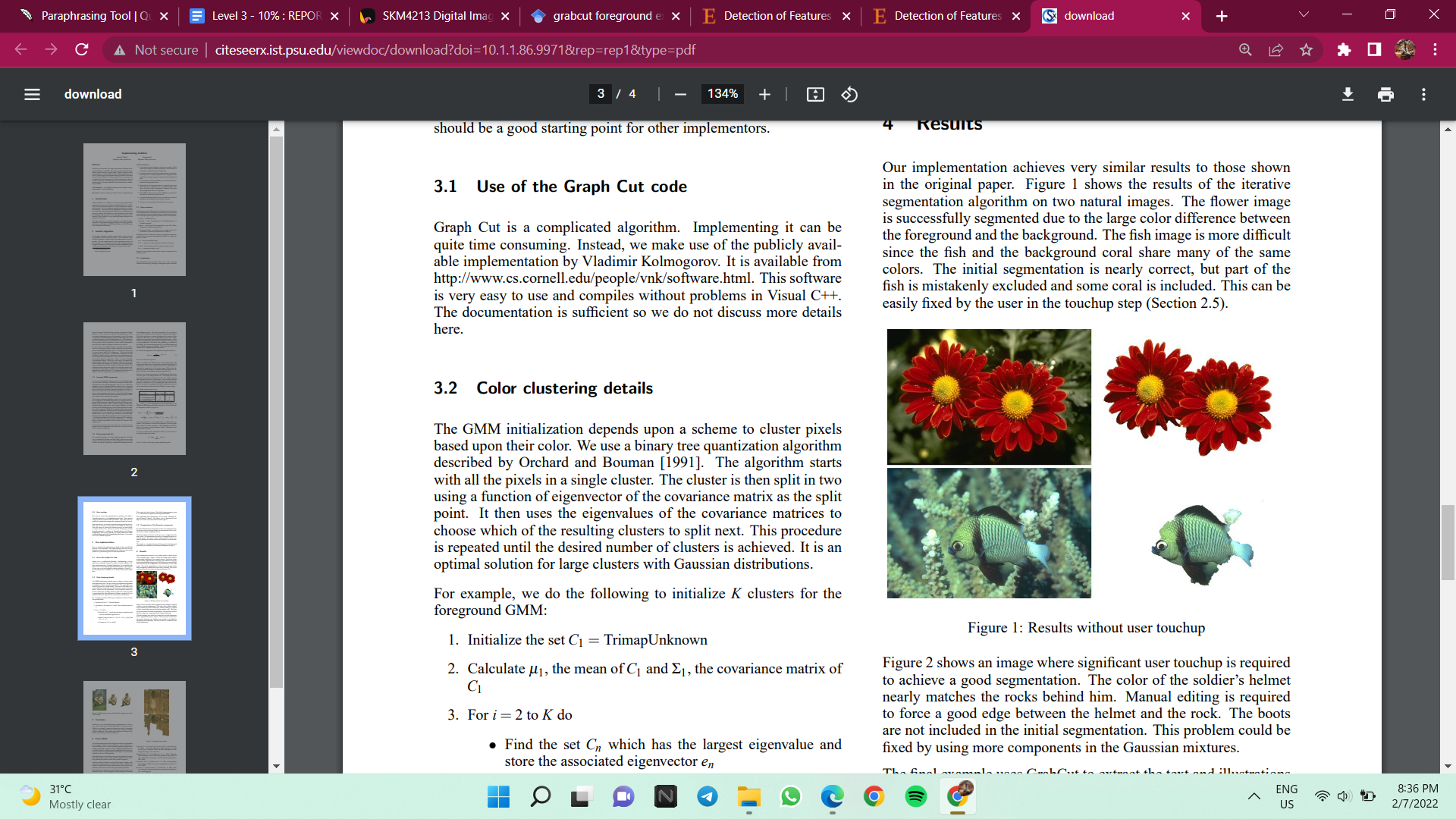


Figure 2: Results without user touchup

1. *Foreground algorithms for detection and extraction of an object in multimedia*

Rekha V. and Natarajan K suggested a few methodologies for extraction of an object by using the foreground algorithm. Figure 3 depicts the grab cut method, which first applies a mask to the inputted picture to extract the foreground and background items. Here, if the mask value is 2, it is regarded as the background, and if it is 1, it is regarded as the foreground. Finally, a foreground mask is created by multiplying this mask value by the frame.

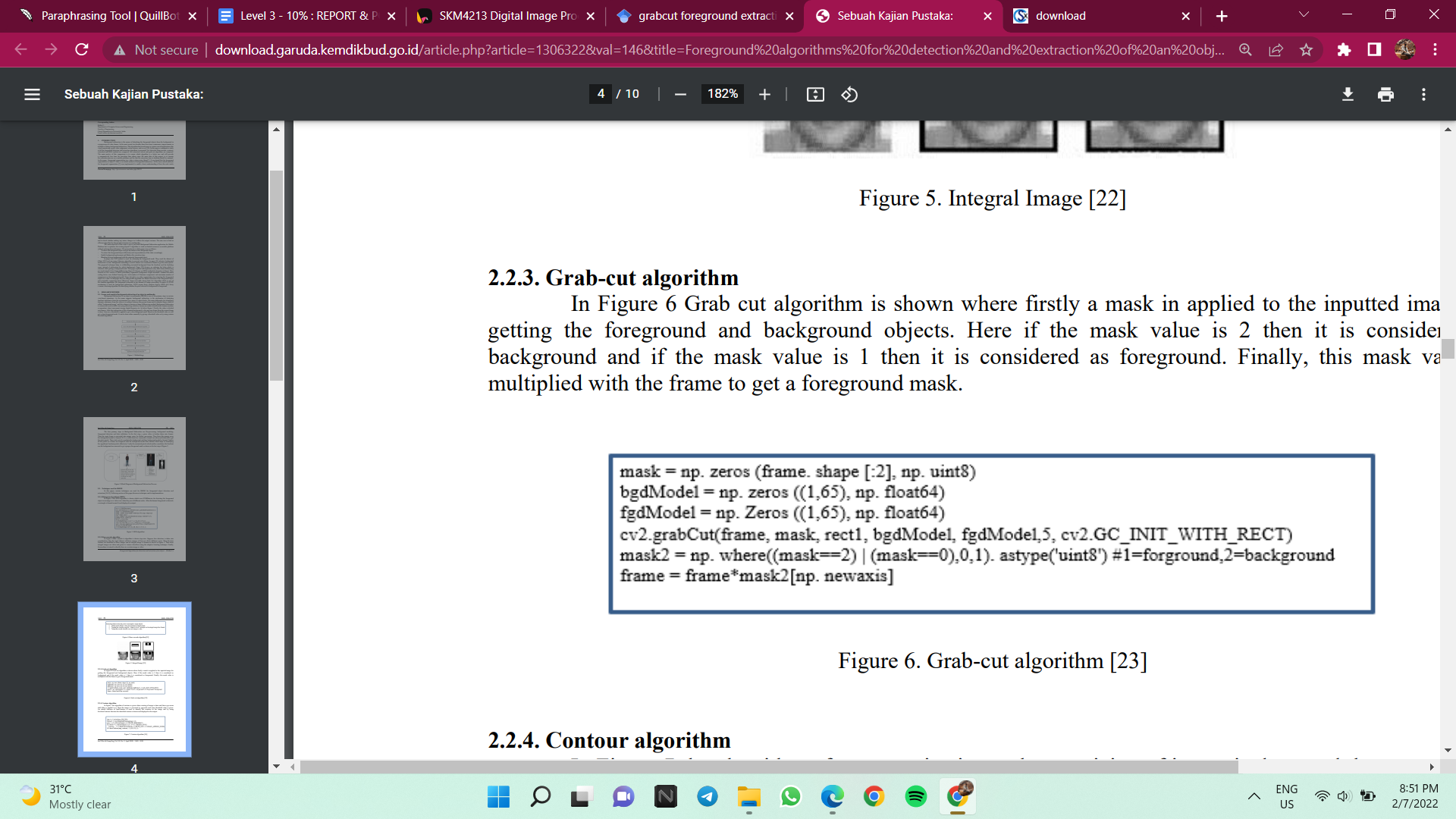


Figure 3: Grab-cut algorithm

The procedure for drawing outlines is shown in Figure 4, which involves scaling the image before applying a pyr mean shift filter. The image is then made grayscale, and a threshold value is specified. The contours in the image are recognised using the built-in findContours function, and using the drawContours function, the found contour is drawn and presented in the output.

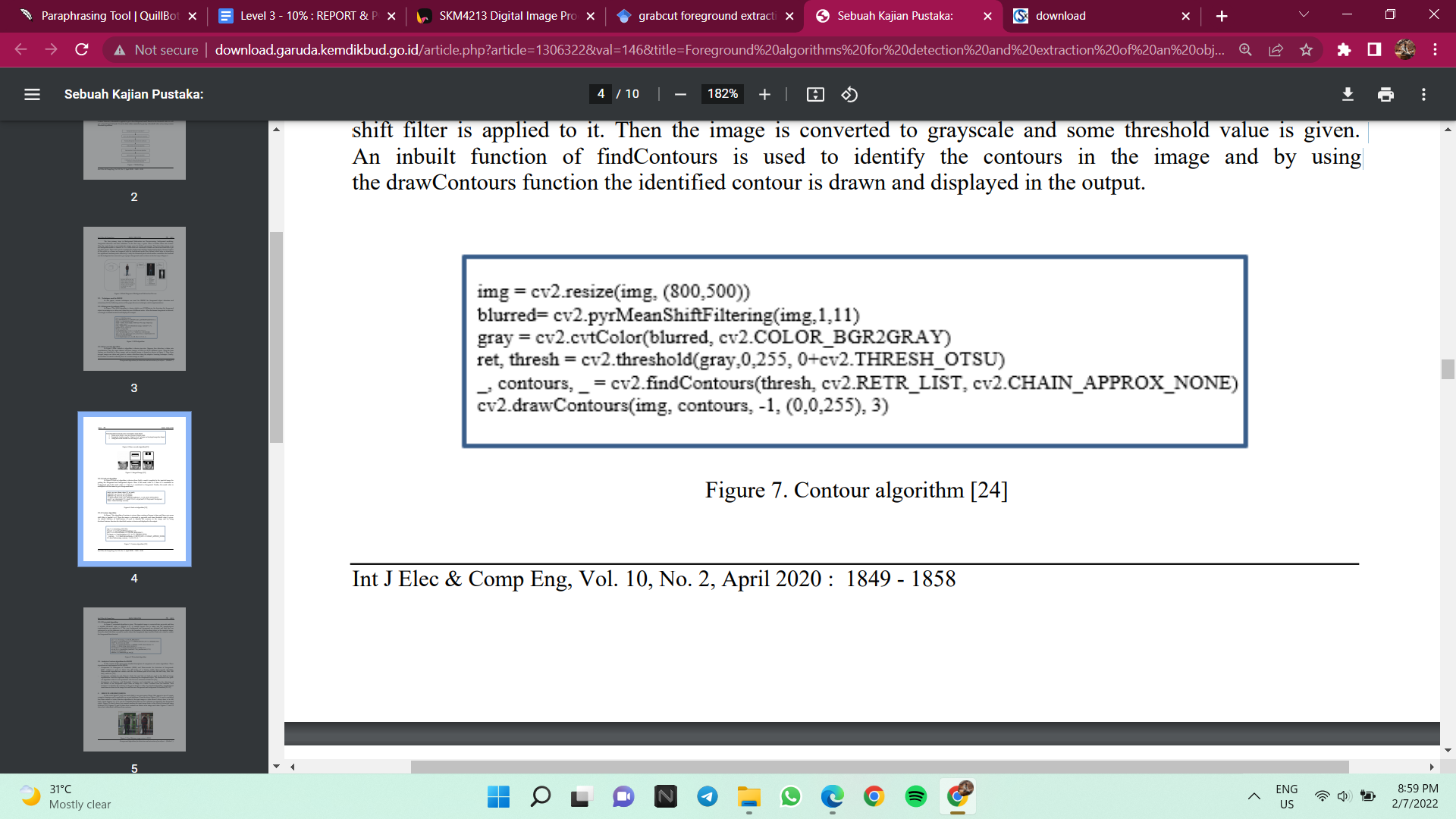


Figure 4: Contour algorithm

Figure 5 depicts the resulting image after applying the algorithms above.

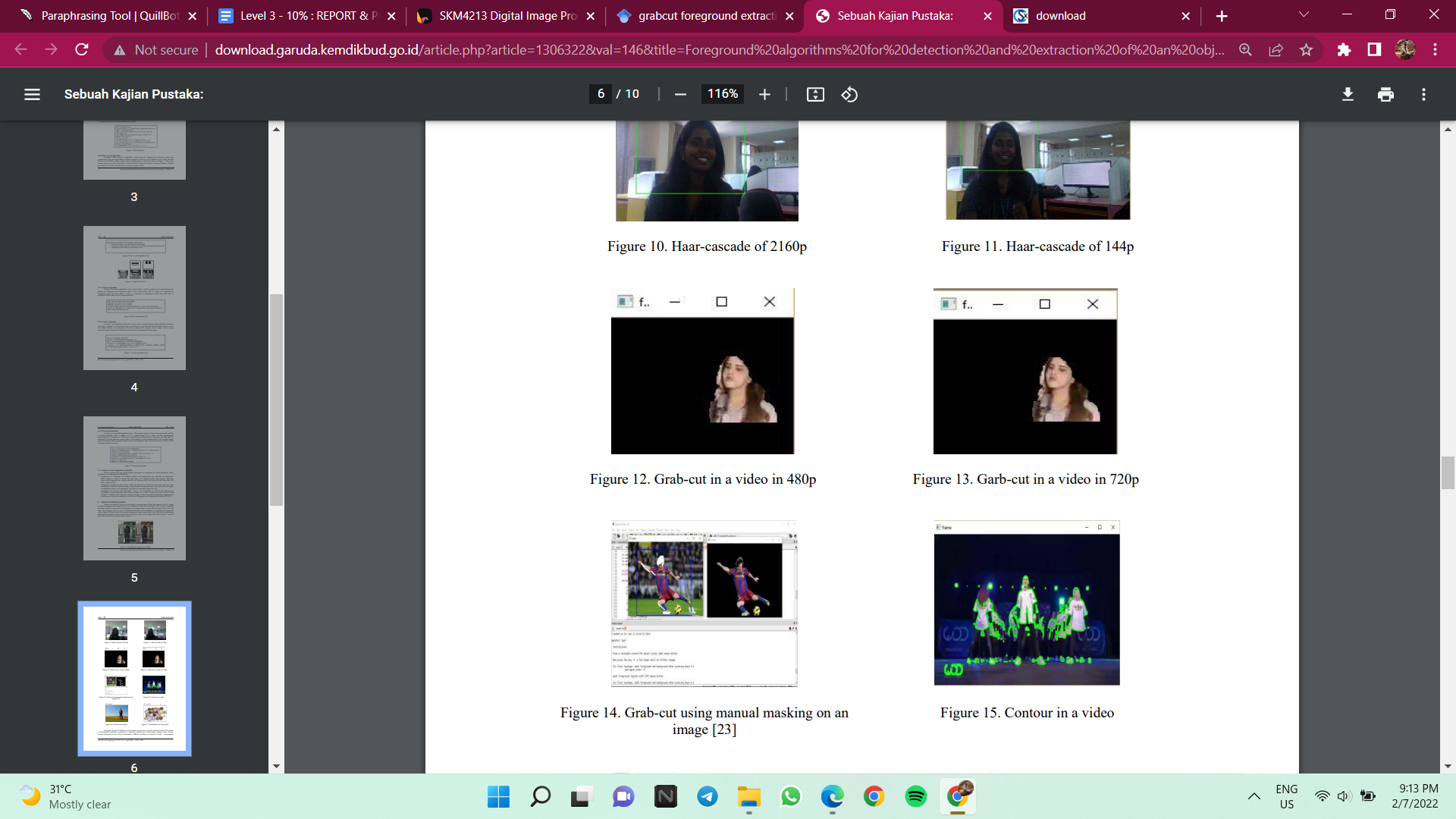


Figure 5: Grab-cut in a video in 480p

1. **METHODOLOGY**
2. *Technique: Cropping*

Cropping is the process of removing unnecessary exterior parts from an image. Many of the images above included black pixels, which you can simply remove by cropping. Images in OpenCV are essentially Numpy arrays. As a result, we can use Numpy array slicing to crop an image and delete the parts we don't want. The first dimension of a Numpy array represents the array's rows (which are the image's height or y-coordinates), while the second dimension represents the array's columns (which is the width of the image or the x-coordinates). First, we'll read the "pest.jpg" input image and convert it from BGR to RGB so that we can plot it using matplotlib. Then we deactivate the x and y axes, resulting in 1700 pixels on the x-axis and 2390 pixels on the y-axis, which we save as "pest cropped.jpg" in the same directory as the input file.

1. *Technique: Foreground Subtraction*

In our application, we applied the GrabCut Foreground Detection algorithm to remove the background of the input image and detect the subject of that image. Carsten Rother, Vladimir Kolmogorov, and Andrew Blake of Microsoft Research Cambridge, UK, developed the GrabCut method in their article "GrabCut": interactive foreground extraction with iterated graph cuts. GrabCut was created as a consequence of the necessity for a foreground algorithm that required minimum user intervention. The GrabCut algorithm works by accepting an input image with *either* a bounding box that specified the location of the object in the image we wanted to segment or a mask that *approximated* the segmentation Iteratively performing the following steps:

* Estimating the color distribution of the foreground and background via a Gaussian Mixture Model (GMM)
* Constructing a Markov random field over the pixels labels (i.e., foreground vs. background)
* Applying a graph cut optimization to arrive at the final segmentation

OpenCV has an implementation of GrabCut via the cv2.grabCut function that makes applying GrabCut a breeze (once you know the parameters to the function and how to tweak them, of course).

1. **RESULT**

Figure 6 depicts the difference of the input image after using the cropping technique.



Before After

Figure 6: Result after cropping

Figure 7 shows the final image after applying the Foreground Subtraction Technique.



Before After

Figure:7 Result after Foreground Subtraction

We then conducted a test on what happens if we change the value of the mask.

**mask2 = np.where((mask == 2) | (mask == 1), 0,1).astype('uint8')**

If we change the value of the mask in the first parameter from 2 to 3, the foreground object will disappear as shown in Figure 8 below. Meanwhile, if we decrease the value of the mask in the left parameter from 2 to 1, the foreground extraction method will not work as shown in Figure 9. Next, when we increase the value of the mask in the right parameter from 0 to 1, there will be a small portion of the background left in the output as shown in Figure 10.



Figure 8: Value of mask in the left parameter = 3



Figure 9: Value of mask in the left parameter = 1



Figure 10: Value of mask in the right parameter = 1

1. **CONCLUSION**

In conclusion, we succeed in detecting a pest in a plant using our application. The selected methods are also able to help us convey the result that we desire. By doing this, we notice how digital image processing is truly helpful for society, especially farmers. However, this application can be improved by changing the value of the mask in the coding so that the damaged leaves disappear, leaving only the pests. Although, our result is also compatible as we wanted to observe the damage done by these pests so that we are able to locate their trails hence getting rid of it.

1. **REFERENCES**

| [1] | C. Rother, V. Kolmogorov and A. Blake, "“GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts," *ACM Trans. Graph.,* pp. 309 - 314, 2004. |
| --- | --- |
| [2] | J. F. Talbot and X. Xu, "Implementing GrabCut," *Brigham Young University,* vol. 3, 2006. |
| [3] | V. Rekha, K. Natarajan and J. Innila Rose, "Foreground algorithms for detection and extraction of an object," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 10, pp. 1849 - 1858, 2020. |

[4] H. Kuzuhara, H. Takimoto, Y. Sato and A. Kanagawa, "Insect Pest Detection and Identification Method Based on Deep Learning for Realizing a Pest Control System," In 2020 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), pp. 709-714, 2020.

[5] S. K. Bhoi, K. K. Jena, S. K. Panda, H. V. Long, R. Kumar, P. Subbulakshmi and H. Jebreen, "An Internet of Things assisted Unmanned Aerial Vehicle based artificial intelligence model for rice pest detection," Microprocessors and Microsystems 80, 2021.

[6] M. Türkoğlu and D. Hanbay, "Plant disease and pest detection using deep learning-based features," Turkish Journal of Electrical Engineering & Computer Sciences, vol. 27(3), pp. 1636-1651, 2019.