**Task 2**

**Computer Vision**

**Team: 20**

Edge and boundary detection (Hough transform and SNAKE)

****

|  |  |  |
| --- | --- | --- |
| Name | Section | Bench Number |
| Abdullah Mohamed | 1 | 39 |
| Mohamed Gamal | 2 | 12 |
| Mohamed Ali | 2 | 22 |

**Introduction**

In the realm of image processing, the accurate detection of edges and shapes within images plays a pivotal role in numerous applications, ranging from computer vision to medical imaging. Our project focuses on developing a software application that implements various algorithms for edge detection, shape detection, and Active Contour Model (snake) evolution. The primary objectives of this software are to provide users with tools for analyzing and manipulating images, ultimately aiding in tasks such as object recognition, segmentation, and feature extraction.

The significance of edge detection lies in its ability to identify boundaries between objects or regions within an image, which serves as a fundamental step in higher-level image analysis tasks. Similarly, shape detection algorithms, such as the Hough transform, enable the identification of geometric shapes like lines, circles, and ellipses, facilitating object recognition and characterization. Moreover, the Active Contour Model, also known as the snake algorithm, offers a flexible framework for contour initialization and evolution, enabling precise delineation of object boundaries.

In this report, we present the design and implementation of our software application, which is built using the PyQt framework for the user interface and incorporates algorithms for edge detection, shape detection, and Active Contour Model evolution. We discuss the theoretical foundations of these algorithms, describe the user interface design, present the testing procedures and results, and provide a comprehensive analysis of the software's performance and capabilities.

**Background**

The success of our software application relies on the understanding and implementation of several key algorithms in the field of image processing. In this section, we provide a brief overview of the theoretical foundations of the algorithms employed in our software:

**Edge Detection:** Edge detection algorithms aim to identify significant changes in intensity or color within an image, which typically correspond to object boundaries or discontinuities. One widely used edge detection technique is the Canny edge detector, which operates by detecting local maxima in the gradient magnitude of the image.

**Shape Detection (Hough Transform):** The Hough transform is a powerful technique for detecting simple geometric shapes, such as lines, circles, and ellipses, within an image. By representing image features in a parameter space, the Hough transform allows for robust detection of shapes even in the presence of noise or partial occlusion.

**Active Contour Model (Snake):** The Active Contour Model, also known as the snake algorithm, is a method for iteratively deforming a contour or curve to align with object boundaries in an image. By minimizing an energy function that combines both image-based and contour-based terms, the snake algorithm can accurately delineate object boundaries, making it valuable for tasks such as segmentation and object tracking.

In the subsequent sections of this report, we delve into the implementation details of these algorithms within our software application, discussing their roles and contributions to the overall functionality of the system.

**User Interface**

**1. Active Control Tab:**

The Active Control tab provides functionalities related to the Active Contour Model (snake) processing. This tab is designed to allow users to initialize and evolve contours to accurately delineate object boundaries within images. Here's a more detailed breakdown of the components:

• Open Button: Allows users to open an image file for processing.

• Parameter Inputs:

• Alpha, Beta, Gamma: Parameters controlling the behavior of the active contour model, such as elasticity, rigidity, and smoothness.

• Iterations: Specifies the number of iterations for the contour evolution process.

• Center and Radius: Parameters for defining the initial contour position and size.

• Initial Control Button: Initiates the placement of the initial contour based on the specified parameters.

• Start Button: Begins the evolution of the contour using the Active Contour Model algorithm.

• Graphics View: Displays the image being processed along with the evolving contour. This provides visual feedback to the user during the contour evolution process.

**Functionality:**

• Users can load an image and specify parameters for the Active Contour Model.

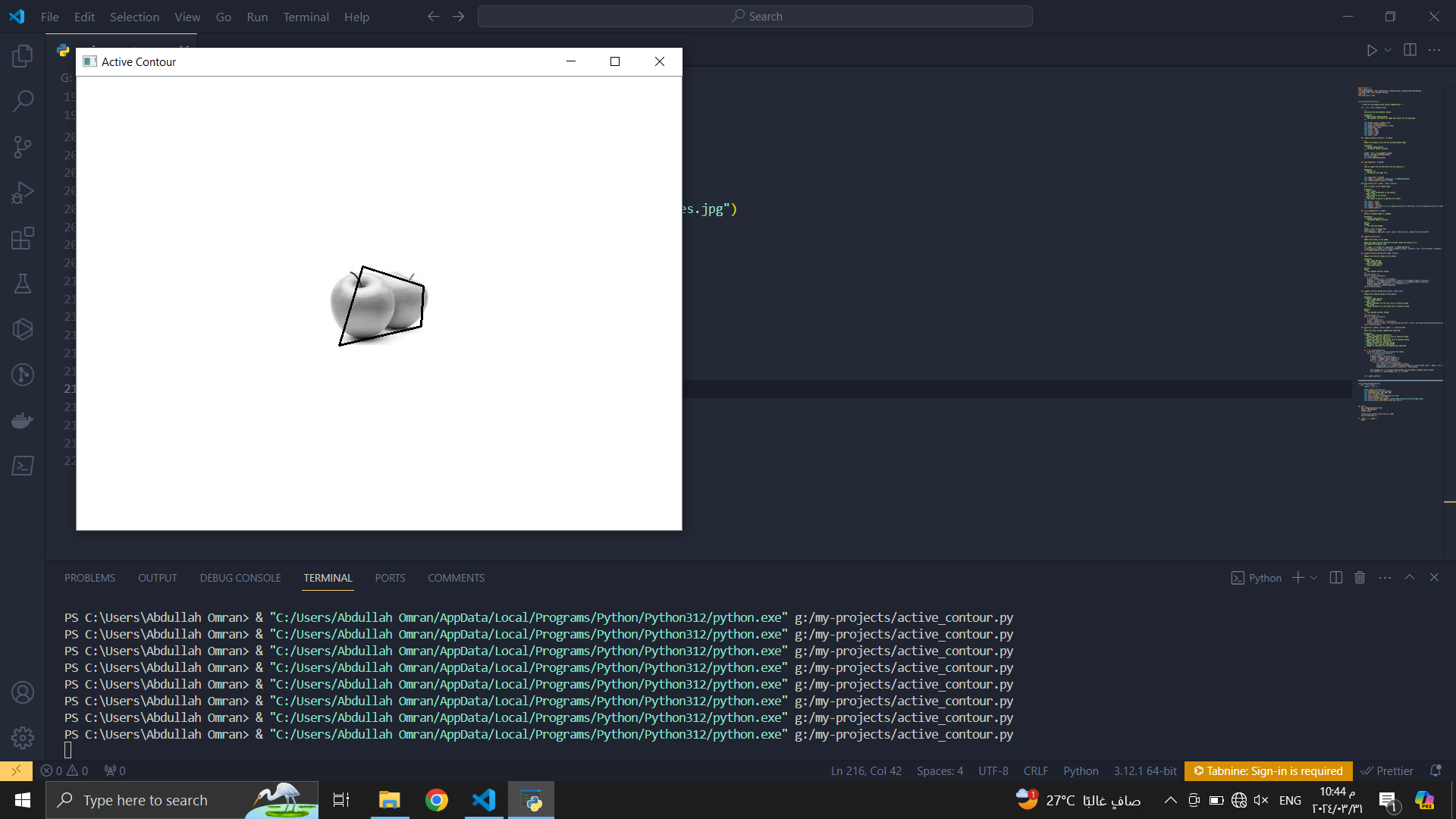
• The initial contour is placed based on the specified parameters.

• The contour evolves over iterations to accurately capture object boundaries.

• Visual feedback is provided through the Graphics View, showing the original image and the evolving contour.

**Experimental Results:**

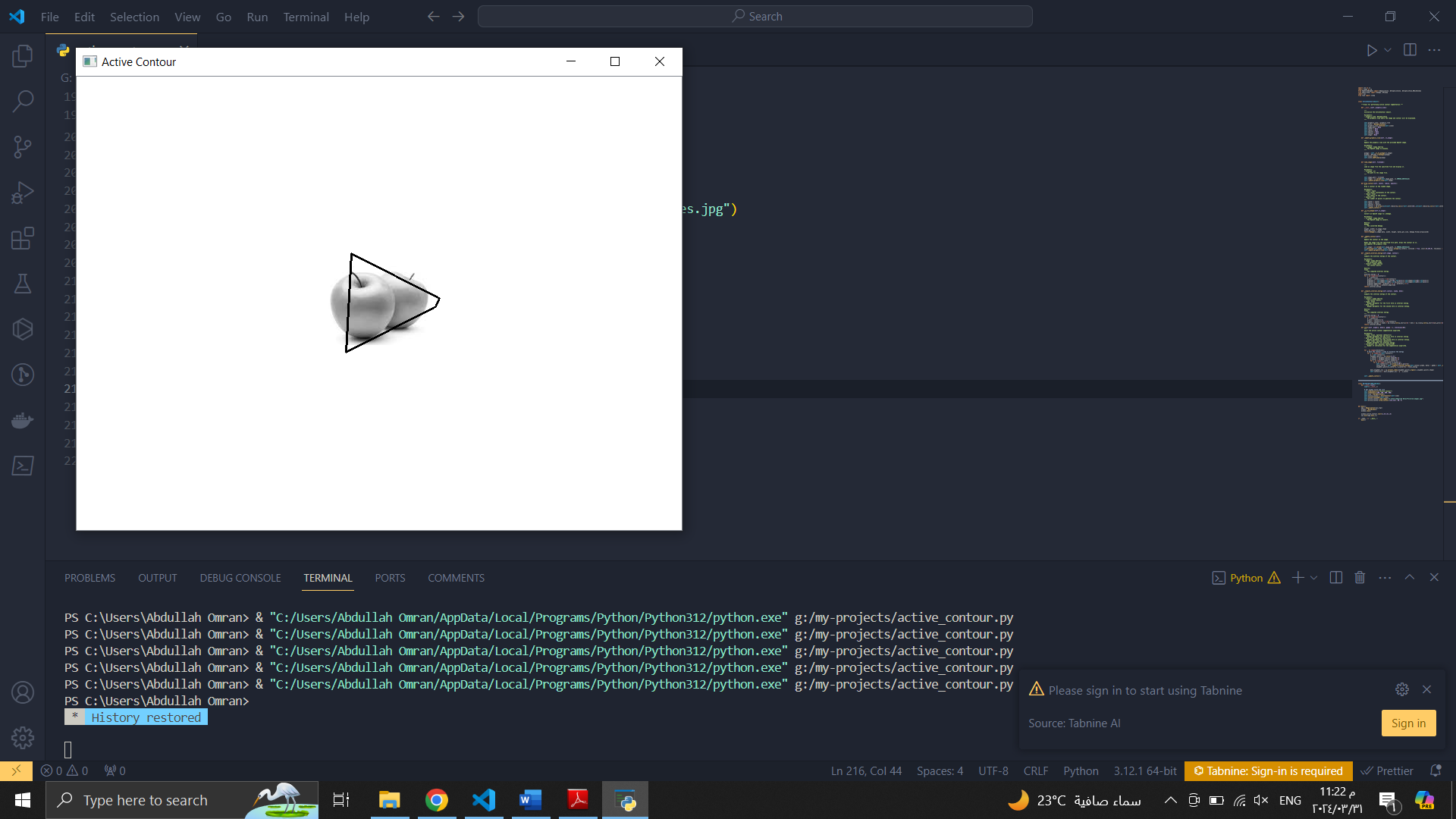
In our first experiment, we test our active contour on six synthetic images. We can discuss all of them as follow:



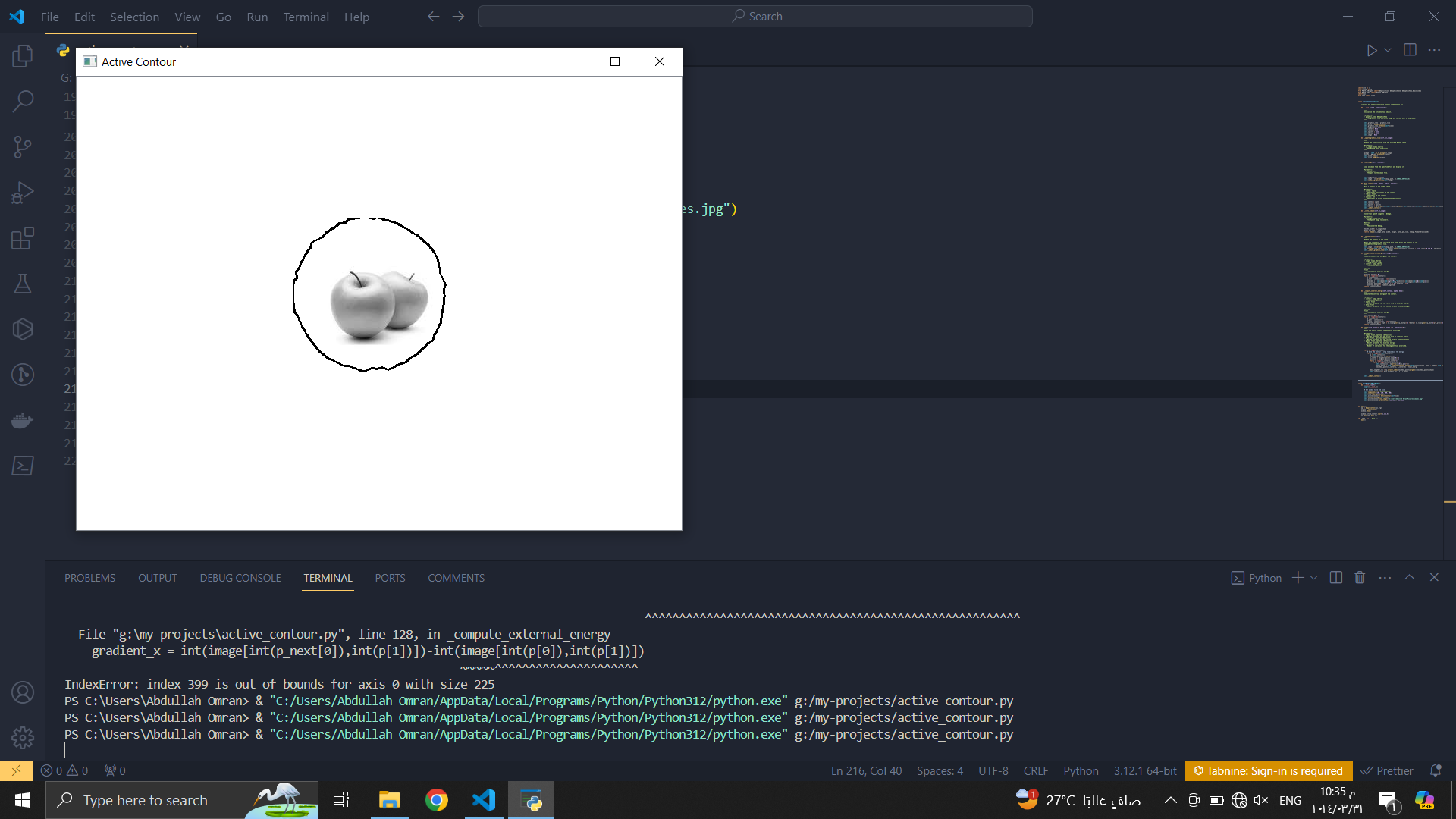
We created a contour that consists of four points due to limitations of the computation time so we can test the behavior of the contour and choose alpha =1 , beta = 0.5, and gamma = 0.1 .

We selected that parameters such that we need to minimize the effect of the external energy and maximize the effect of the internal energy so the contour deforms perfectly to fit the feature by increasing the number of iterations

Also we selected the iterations to be 1500.



We can see the initial contour above , the four points are located apart from the apple and we run the algorithm so the four points become nearst to the apple without penetration.



As we can see from the figure above, we have increased the number of points that construct the contour. The contour deforms slightly based on the selected parameters above. So we need to increase the number of iterations to make the contour fit the apple. As we increase the number of iterations, we need more computational power to achieve that.

**2. Detection Tab:**

The Detection tab is dedicated to various image detection functionalities, including edge detection and shape detection using the Hough transform. Here's a detailed breakdown of the components:

• Open Button: Allows users to open an image file for processing.

• Edge Detection Button: Initiates edge detection on the loaded image using algorithms like the Canny edge detector.

• Shape Detection Buttons: Perform shape detection on the image to identify lines, circles, and ellipses.

• Graphics Views: Display the original image and the detected shapes (lines, circles, ellipses) for visual inspection and analysis.

Functionality:

• Users can load an image and apply edge detection to highlight object boundaries.

• Shape detection algorithms identify geometric shapes such as lines, circles, and ellipses within the image.

• The detected shapes are superimposed on the original image for visual inspection and analysis.

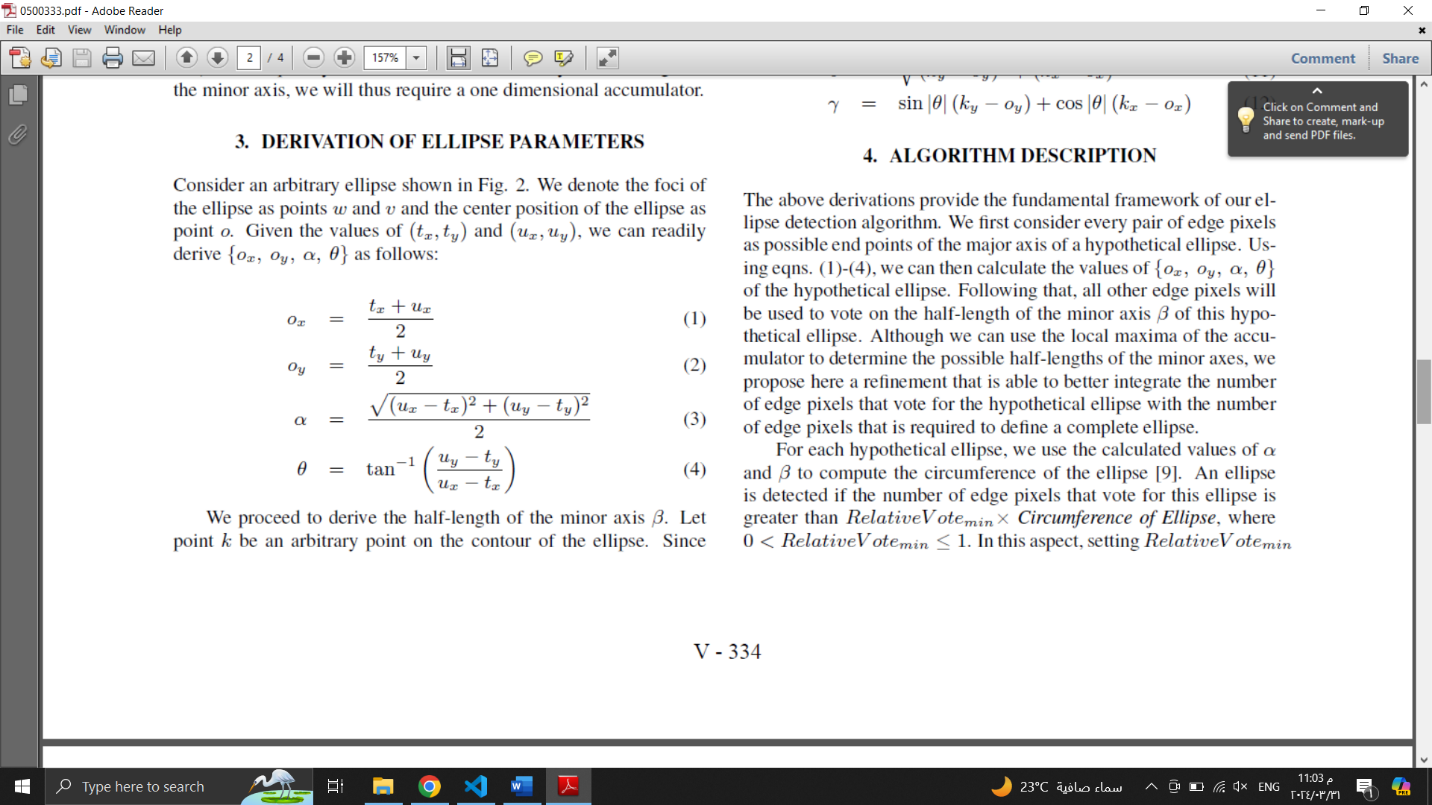
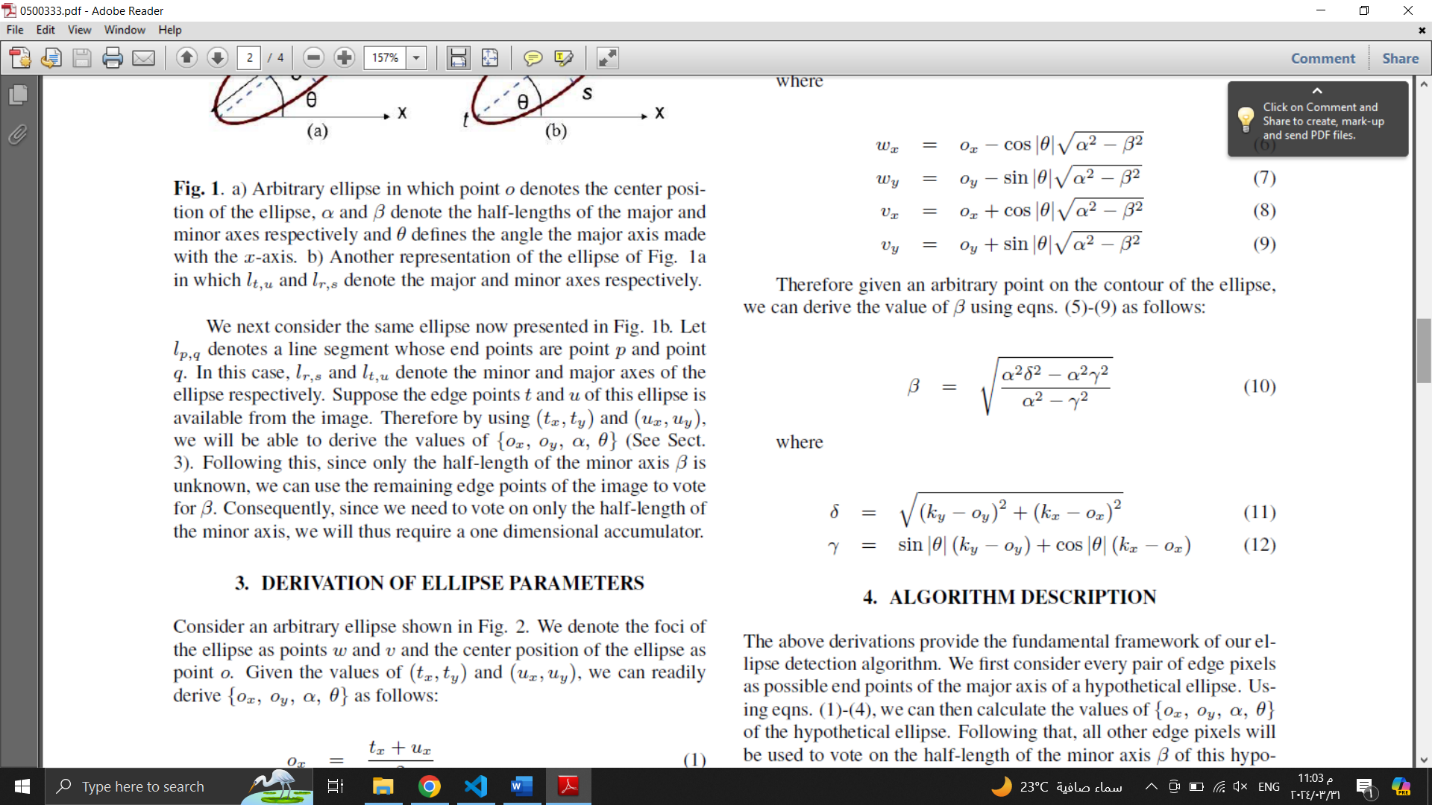
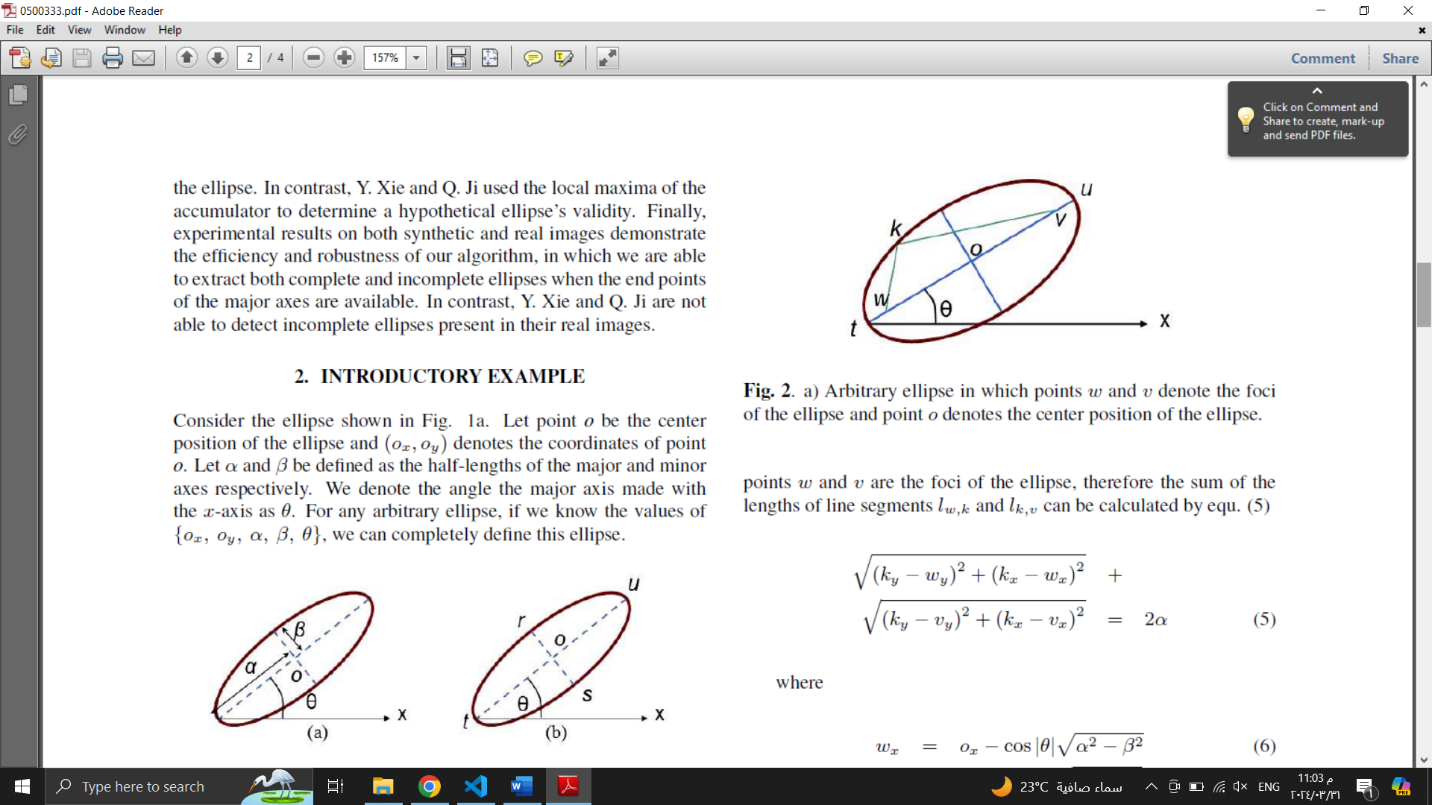
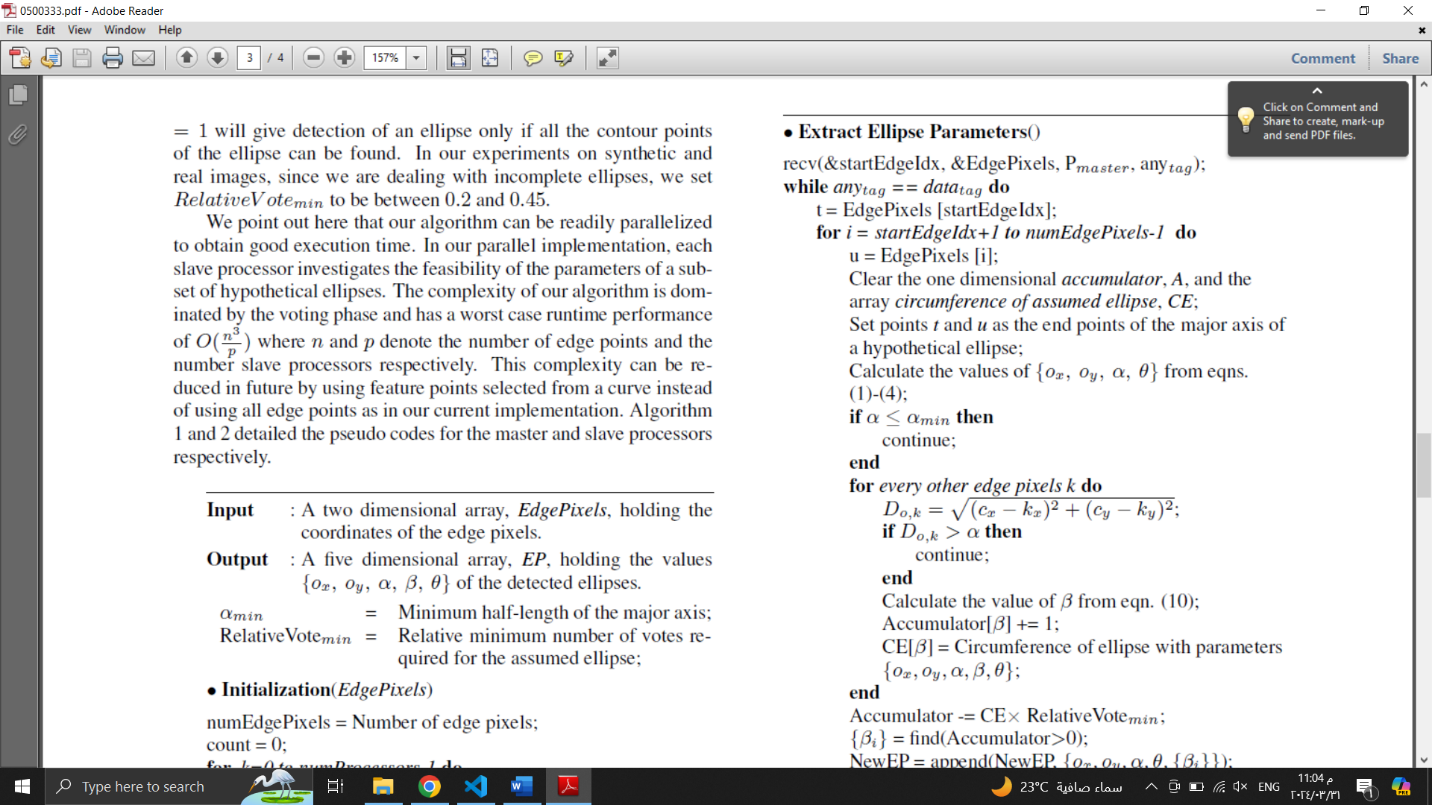
Use Case:

• Object detection: Identifying objects of interest in surveillance or satellite imagery.

• Feature extraction: Extracting key features such as road lanes or facial features from images.

• Pattern recognition: Recognizing specific patterns or structures within images for classification or analysis purposes.

**2. Ellipse:**

We have tried to implement a research paper. We were about to finish the algorithm but we faced a problem related to the accumulator array that consists of about 6 millions of elements and such things are hard to debug due to looping over it. So we need extra time to finish it. The paper is shown below.

**Testing, Results, and Discussion**

**Conclusion**

In conclusion, the development of our software application represents a significant step forward in the field of image processing, particularly in the domains of edge detection, shape detection, and Active Contour Model evolution. Through the integration of PyQt for the user interface and implementation of various algorithms, including the Canny edge detector, Hough transform, and Active Contour Model, our software provides users with powerful tools for analyzing and manipulating images.

The testing and validation of our software have demonstrated its effectiveness and robustness in detecting edges, identifying shapes, and accurately delineating object boundaries. Furthermore, the comprehensive analysis presented in this report highlights the strengths and limitations of our software, paving the way for future enhancements and refinements.

Overall, our software application holds great potential for a wide range of applications, including object recognition, medical imaging, and computer vision. By continuing to improve and expand upon its capabilities, we aim to contribute to advancements in image processing and empower users with versatile tools for image analysis.

**References**

[1] Canny, John. "A Computational Approach to Edge Detection." IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, no. 6, 1986, pp. 679-698.

[2] Duda, Richard O., and Peter E. Hart. "Use of the Hough Transform to Detect Lines and Curves in Pictures." Communications of the ACM, vol. 15, no. 1, 1972, pp. 11-15.

[3] Kass, Michael, et al. "Snakes: Active Contour Models." International Journal of Computer Vision, vol. 1, no. 4, 1988, pp. 321-331.

[4] PyQt Documentation. Riverbank Computing, 2024. Available at: https://www.riverbankcomputing.com/software/pyqt/

[5] OpenCV Documentation. OpenCV Foundation, 2024. Available at: https://docs.opencv.org/

[6] NumPy Documentation. NumPy Community, 2024. Available at: <https://numpy.org/doc/>

**Appendices**