PROJECT REPORT

Cartoonify An Image

Abstract:

This project presents the development of a dynamic and interac tive application that enables users to transform their ordinary

the powerful image processing capabilities of OpenCV and the user friendly interface of Tkinter, the application provides three distinct styles for image transformation: Watercolor, Sketch, and Cartoon. Users can select their desired style and upload an image, which is then processed to create a cartoon-like effect. The application supports full-screen mode for an immersive experience and includes functionalities for saving the processed images. This project not only demonstrates the practical application of advanced image processing techniques but also offers a creative and engaging tool for image manipulation and enhancement.

Introduction:

This project aims to develop a graphical user interface (GUI) application that allows users to cartoonify images. By utilizing various image processing techniques, the application can transform regular images to cartoon like-visuals, providing a fun way to view photographs.It includes options for different cartoon styles such as Watercolor,sketch,and a custom Cartoonify style.

Overview:

- Features:
  - Image upload
  - Cartoonification with 3 styles:
    - Water Color
    - Sketch
    - Animate
  - Image saving option
  - User friendly GUI

- Libraries Used:
  - cv2
  - easygui
  - numpy
  - matplotlib
  - tkinter
  - PIL
- Image Processing Techniques:
  - Water Color:
    - Stylization with cv2.stylization()
  - Sketch:
    - Grayscale conversion, median blur, adaptive threshold
  - Animate:
    - Edge detection, bilateral filtering, bitwise operations

METHOD:

First a user-friendly interface opens asking about the style requirement of the user. Then the user is asked to select the file and upload it. An image is displayed can be saved if wanted to . The core functionality relies on OpenCV for processing images. Depending on the selected style, the application implements different processing techniques like:

1. Water color style:

   Stylization: Utilizes cv2.stylization() to create a watercolor effect, which applies a non-local means filtering algorithm to enhance colors and create a painted look.

2. Sketch style:

   Grayscale Conversion: The image is converted to grayscale

   Smoothing: A median blur is applied to the grayscale to reduce noise

   Edge Detection: Uses cv2.adaptiveThreshold() to identify edges, creating a sketch-like effect.

   Color Filtering: A bilateral filter is applied to the original image to preserve edges while smoothing colors, followed by a bitwise AND operation to combine the edge mask with the color image.

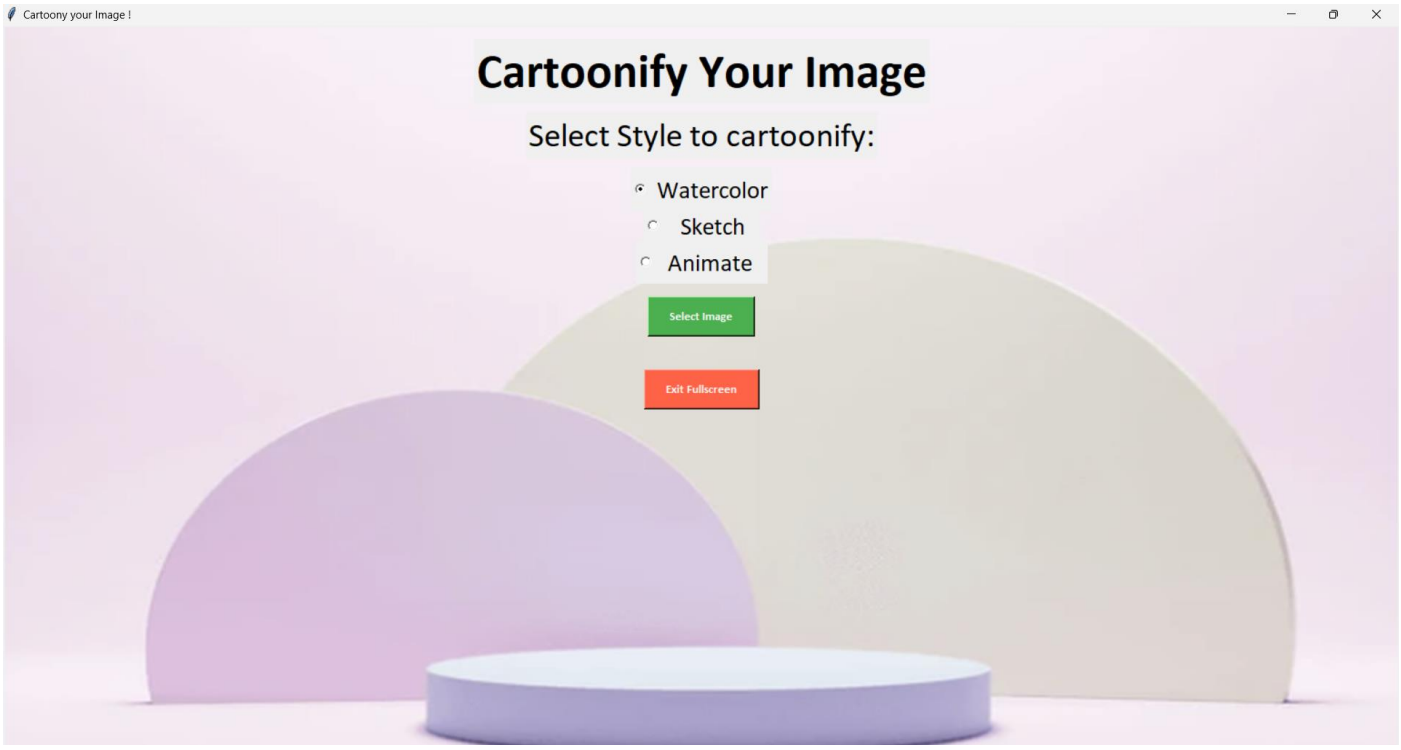
3. Animate:

   Edge Detection and Color Filtering: Similar to the sketch style, this method uses grayscale conversion, median blur, and adaptive thresholding.

Bilateral Filtering: Applies cv2.bilateralFilter() to smooth colors while keeping edges sharp.

Combination: The cartoon effect is achieved by combining edges and colored images with bitwise operations.

How the GUI window looks:



Uploaded image:

Water Color Style:

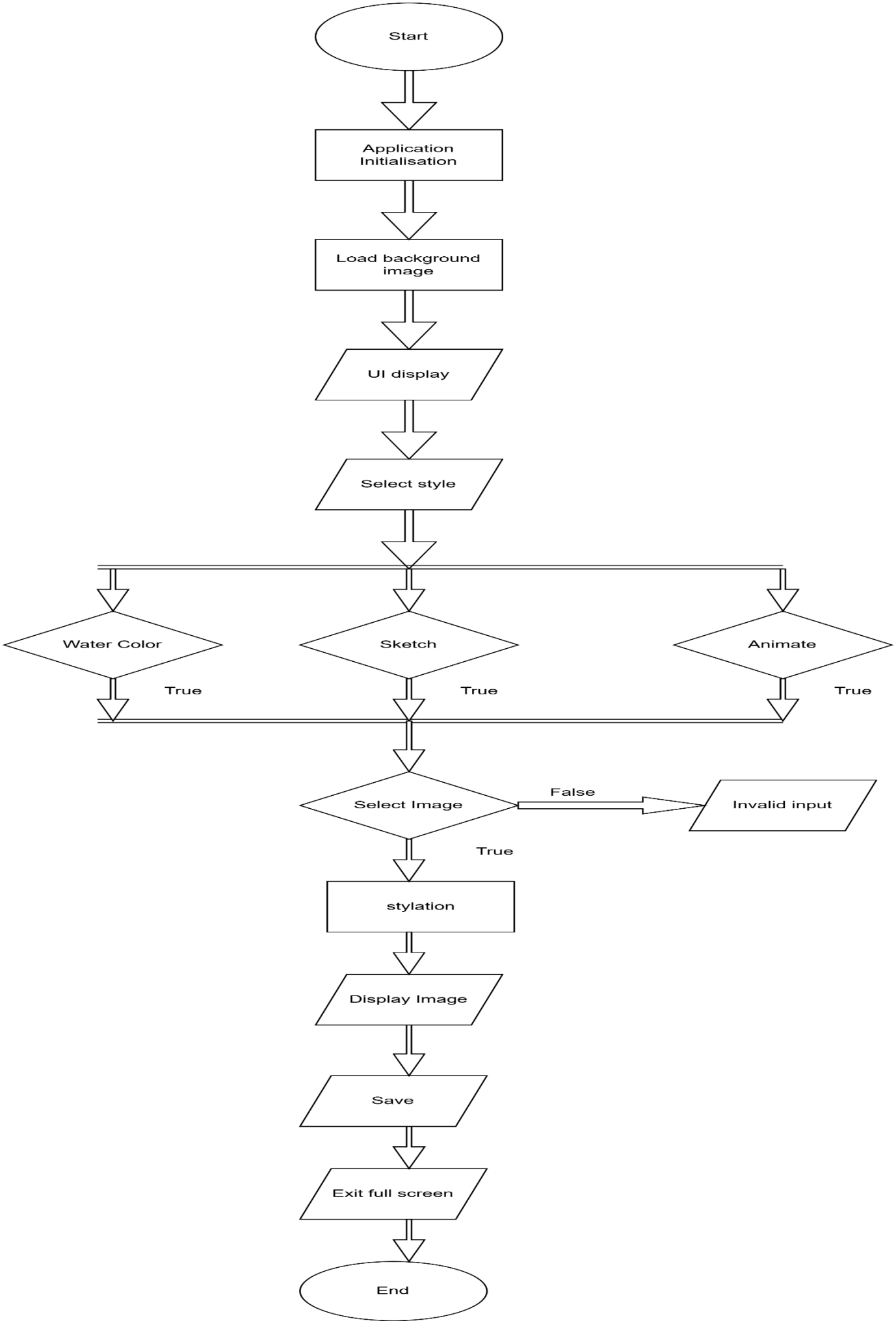
Sketch style:

Animate:

How to run:

Prior running, the libraries to be installed are opencv-python easygui, matplotlib, numpy, pillow. Then, as input an image is given or uploaded.

File name	Use
1.main.py	Contains the main script for the Tkinter GUI and cartoonification functions
2.background.png	Background image used in the GUI (ensure the path is correctly set in the script).
3.screenshot.png	Example input image file for cartoonification (this can be any image file selected by the user).


Start
Application Initialisation
Load background image
UI display
Select style
Water Color
Sketch
Animate
True
True
True
Select Image
False
Invalid input
True
stylation
Display Image
Save
Exit full screen
End

MILESTONES:

- Developed a functioning GUI using tkinter.
- Integrated easygui for easy files selection.
- Responsive style of GUI window.
- The best effect of animation style is not achieved.

The opened gui window has title on the top saying Cartoonify an Image! And after the cartooned image is displayed in a new window, the window has options to *align* the image to left or right top or bottom according to scale and one can *zoom* the image and *plot of image* is also shown.

Conclusion:

This application successfully combines image processing techniques with a user-friendly interface, making it easy for users to create artistic renditions of their images. Its modular design allows for future enhancements, such as adding more styles or additional editing features. Overall, this project demonstrates the power of combining Python libraries for creative applications, providing an engaging way for users to transform their photos into artwork.