UNIVERSITY OF THE PHILIPPINES MINDANAO

Bachelor of Science in Computer Science

Juliene Airen C. Bontilao

Interactive Computer Vision Toolkit with Real-Time Processing Capabilities

Adviser

Armacheska R. Mesa-Satina, PhD

Department of Math, Physics, and Computer Science

Date of Submission

May 2025

ABSTRACT

This project presents an interactive computer vision application built with Streamlit that integrates four major techniques: image manipulation, object tracking, contour and shape detection, and dense optical flow. The goal is to create an accessible, real-time visualization tool for demonstrating foundational concepts in computer vision. Each module is implemented with OpenCV and accessible via a user-friendly graphical interface, allowing users to explore the power of image and video processing interactively.

Objectives

1. To build a functional and interactive computer vision application with a GUI.
2. To apply at least three computer vision techniques learned from the course.
3. To demonstrate understanding and creativity in solving real-world or simulated problems using computer vision.
4. To enable real-time processing and visualization of image/video transformations.

Project Features

1. User-Friendly Interface: Built with Streamlit for simplicity and accessibility.
2. Image Manipulation Tools: Allows sharpening, dilation, erosion, edged, and combined effects.
3. Object Tracking: Tracks yellow-colored objects in real-time via webcam and draws their path.
4. Shape Detection: Detects geometric shapes (triangles, squares, rectangles, circles, stars) using contour approximation.
5. Dense Optical Flow: Visualizes motion in a video stream using color-coded flow vectors.
6. Modular Structure: Each feature can be accessed and tested independently via the sidebar.

Techniques Used and Their Relevance to Class

1. **Image Manipulation (Sharpening, Dilation, Erosion, Combined, Edges)**

   Relevance - Demonstrates low-level image processing techniques using convolution and morphological operations.

   Application - This helps in preprocessing steps like enhancing features and removing noise.

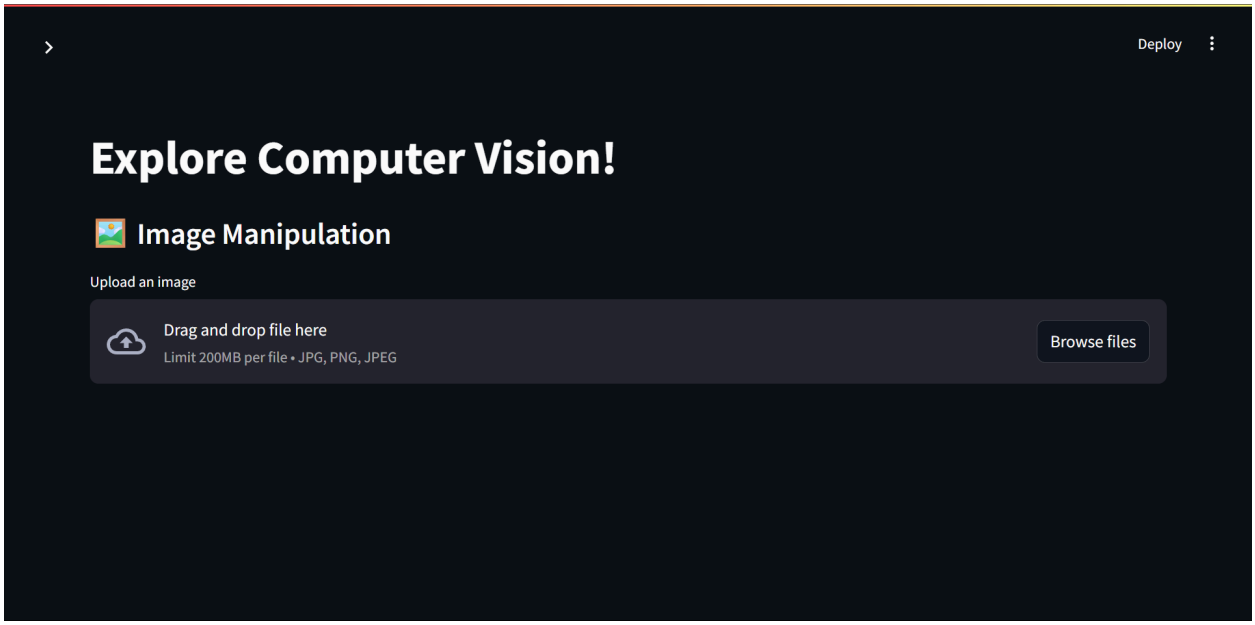
2. **Object Tracking**

   - **Relevance:** Uses HSV color space and contour tracking, which are fundamental in motion detection and gesture tracking.
   - **Application:** Can be used in surveillance systems and interactive gesture-based applications.

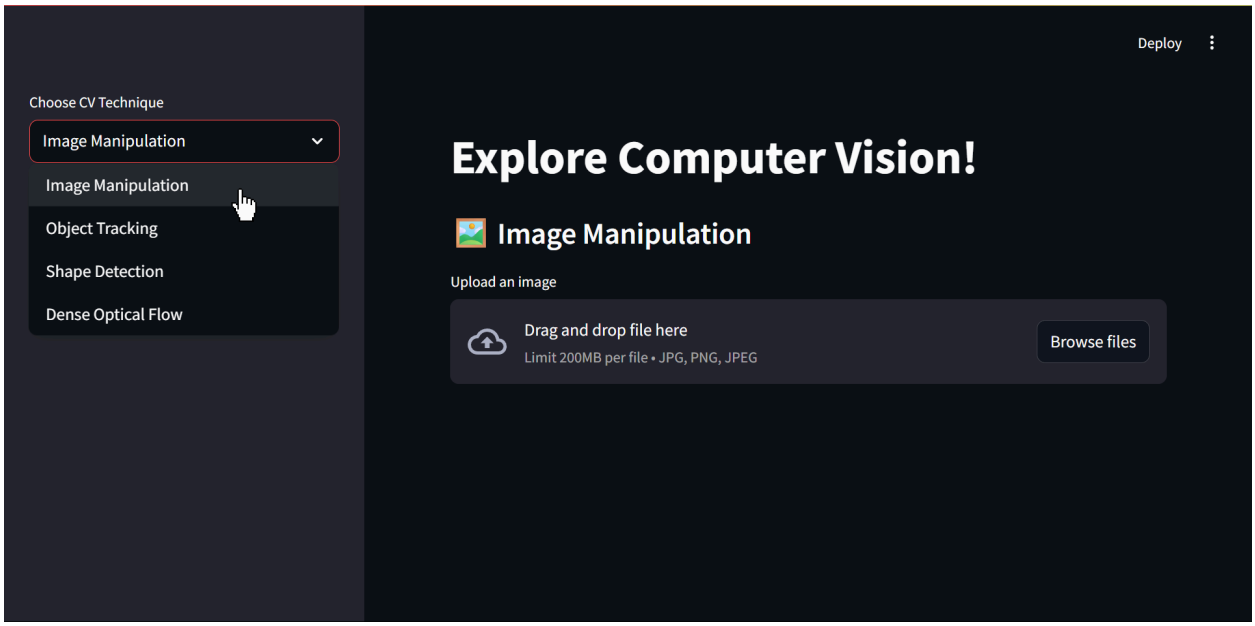
3. **Contour and Shape Detection**
    - Relevance: Illustrates the importance of contours, moments, and shape approximation.
    - Application: Useful in pattern recognition and object classification.
4. **Dense Optical Flow**
    - Relevance: Shows how motion estimation can be visualized using Farneback's algorithm.
    - Application: Applied in traffic monitoring, motion analysis, and activity recognition.

Screenshots and Workflow

- Landing Page



- The application starts by launching a user interface built with Streamlit, where users can interact with the different computer vision features through a sidebar menu. The user begins by selecting one of four modules: Image Manipulation, Object Tracking, Contour and Shape Detection, or Dense Optical Flow through the sidebar



- Once an option is selected—such as Image Manipulation—users can upload an image file to apply various transformations, including sharpening, dilation, erosion, edge detection, and combined effects. The processed image is then displayed immediately on the screen.
- In the Object Tracking option, the application accesses the webcam feed. It tracks the movement of a yellow-colored object in real-time, drawing a trail of its path on the video. This helps visualize how the object moves across the screen over time.
- In the Contour and Shape Detection option, the application also uses the webcam feed. It processes each frame to detect and label geometric shapes like triangles, rectangles, squares, circles, and stars based on their contours. Detected shapes are highlighted and labeled directly on the video stream.

- In the Dense Optical Flow option, a preloaded video file is analyzed frame by frame. The application calculates and visualizes the motion within the video using color-coded vectors, showing how movement occurs between frames.

After the user finishes exploring a feature, they can return to the sidebar to switch to another module. All outputs are displayed directly in the Streamlit interface, making it easy to interact with and understand each computer vision technique in real time.

Challenges and Limitations

- **Webcam Access Issues.** Some browsers or systems restrict webcam access when used through Streamlit, causing permission or compatibility errors.
- **Streamlit Limitations.** Streamlit is not optimized for high-frame-rate video processing, making optical flow slightly laggy.
- **Color Dependency in Object Tracking**. Object tracking depends on fixed color thresholds (yellow), which limits flexibility in different lighting conditions.
- **Performance Constraints.** Dense optical flow is computationally expensive and may lag on low-end systems.

Future Enhancements

1. Add support for uploading custom videos for optical flow analysis.
2. Allow dynamic color selection for object tracking instead of hardcoded HSV values.
3. Improve the UI layout using advanced components.
4. Add shape counting and bounding box labels in contour detection.
5. Convert the project into a standalone executable or deploy as a web service.

Reflection

This project was a meaningful opportunity to explore how theory from CMSC 191 translates into practical applications. Implementing and visualizing multiple CV techniques helped deepen my understanding of how low-level image operations and real-time algorithms work together. I particularly enjoyed creating the interactive GUI, which made the project more user-friendly and engaging. While there were technical challenges, they provided valuable learning experiences, especially around debugging video processing and integrating multiple modules cohesively. This project really sparked my interest in computer vision and motivated me to pursue more advanced topics in the field.

References

All the code implementations used in this project were based on the example codes and materials provided in our CMSC 191 course.

Modifications or edits were made to combine or change different functions and to adapt the code for integration into a unified graphical user interface (GUI) using Streamlit to enhance functionality, visual clarity, and user interactivity.

This project is intended for educational purposes only, to demonstrate my understanding and practical application of concepts learned throughout the course.

Demo Video Link Here.

Github Link Here.