Project Report: AI-Enhanced Image Stitching and Edge Detection

Gaza Sky Geeks

Computer Vision Bootcamp 2024

Prof. Izzeddin Teeti

Rama Hasiba

Rasheed Alqobbaj

Contents

Introduction

The purpose of this project is to develop an application that implements image stitching and different edge detection techniques, including Canny Edge Detection, as well as Difference of Gaussians (DoG), and an AI-based human segmentation model. The application provides users with an interactive graphical interface to adjust parameters and compare the results of different edge detection methods. This report outlines the design decisions, algorithms employed, challenges faced, and solutions implemented throughout the project.

Design Decisions

Image Stitching

- Utilized OpenCV library for image stitching.

- Allowed the user to select multiple images to stitch through the HTML input tag.

Edge Detection

- Employed OpenCV for Canny Edge Detection and NumPy to calculate the lower and upper thresholds.

- Implemented DoG edge detection followed by morphological close operation for noise reduction and used a plus shaped kernel (+) since it had better results in my tests.

- Provided a slider for adjusting the kernel size.

AI-based Human Edge Detection

- Integrated the pre-trained YOLO (You Only Look Once) model for human segmentation.

- Filtered and displayed human figure detections with confidence levels above 50%.

User Interface

- Designed a simple user interface using HTML and CSS then connected it to my python code through the Flask library.

- Implemented features for uploading images, viewing processed results, and dynamically adjusting kernel size.

Challenges and Solutions

User Interface

- **Challenge:** Making a decent looking GUI and connecting it to a python code.

- **Solution:** Long painful hours of trial & error and debugging. This was my first real experience with Flask, so I had to learn the basics.

Performance Optimization

- **Challenge:** Stitching images and edge detection took longer than expected when using the GUI.

- **Solution:** Cleaned the code by removing redundant code. In the future I will experiment with multi-threading and parallel processing to make it faster.

Time Schedule

- **Challenge:** Finding time and energy to work on the project with university, work, and being the head of the technical team for my university's GDSC.

- **Solution:** No real solutions were found other than an embarrassing amount of energy drinks. I just toughed it out.

Code Quality and Documentation

Readability

- Ensured code clarity through meaningful variable names and well-commented sections.

- Maintained a consistent coding style throughout the project.

Documentation

- Provided comprehensive documentation within the source code using docstrings.

- Maintained the project through git on GitHub and used issues to structure my progress.

Future Enhancements

- Incorporate additional edge detection algorithms for comparison and analysis.

- Extend AI-based detection capabilities to recognize other objects or specific patterns within images.

- Enhance user interface features for better visualization and interaction with processed results.

Conclusion

In conclusion, the developed application successfully meets the project requirements by providing efficient image stitching and accurate edge detection capabilities. The integration of an AI-based human detection model enhances the application's functionality, making it a valuable tool for various image processing tasks. The user-friendly interface ensures ease of use, while the code quality and documentation ensure maintainability for future enhancements.

Acknowledgements