SOFTWARE REQUIREMENTS SPECIFICATION

For

Pairwise Image Visualizer

Version 1.0

Group 18

- Yatharth Gupta
- Rohit Dhanotia
- Abhinav Kumar
- Pilla Venkata Sekhar
- Khushi Agarwal

Submitted to:

Dr. Puneet Gupta

15 March 2023

INDEX

1 Introduction

We decided to use the Incremental Development Model. The design is broken down into smaller, more manageable pieces in an incremental development model, and each iteration produces a working, usable portion of the finished product. This enables the incorporation of comments and adjustments throughout the development process, ensuring that the final product meets the customers' expectations.

This paradigm works incredibly effectively for projects involving particular features since these systems frequently have complex requirements or requirements that change quickly. Since we might attempt to add more features to our project in the future, It will make the development process more adaptable and iterative, allowing the team to make changes and advancements in response to comments and suggestions.

The Waterfall model follows a sequential procedure, where each step must be finished before moving on to the next, in contrast to the incremental development. For systems with rapidly changing requirements, this method might not be the best. Since time is not a restriction for us, timeboxing splits work into predetermined time frames, which may not be appropriate. RUP (Rational Unified Process) is a heavier-weight, more organised paradigm that might be too much for a small to mid-sized design. Instead of controlling the entire software development process, integration and configuration management concentrate on managing the specific components of software design.

1.1 Purpose

The purpose of this software is to open images from multiple folders and visualise them. The images are of the same name and exist in different folders, maybe due to a series of image processing like segmentation, landmark or object detection, or upscaling. This tool would allow the user to visualise such images.

1.2 Intended Audience

The target audience for our image visualisation software is professionals and researchers who work with images and need to evaluate the results of image processing algorithms. This could include:

- Computer vision researchers: Computer vision researchers who develop and evaluate image processing algorithms would find your software tool useful for visualising and comparing the results of their work.

- Medical imaging professionals: Medical imaging professionals who work with MRI, CT, or ultrasound images would benefit from your software to evaluate the quality of image processing algorithms and compare different images of the same patient.
- Geospatial analysts: Geospatial analysts who work with satellite or aerial images would find your software useful to compare different versions of the same image after processing, such as segmentation or object detection.
- Graphic designers: Graphic designers who work with images could use your software to compare different versions of an image at different stages of the design process.
- Hobbyists and enthusiasts: Even hobbyists or enthusiasts who work with images, such as photographers or artists, could use your software to compare and evaluate different versions of their work.

1.3 Project Scope

The system will consist of a software application that can run on Windows, macOS, and Linux. The application will be responsible for processing directories and images.

1.4 User Requirements

- The application shall be able to take input of directories from your machine.
- The application shall be able to detect filenames in the directory.
- The application shall be able to display images.
- The application shall be able to process images.

2 System Features

2.1 System Architecture

Modules: Tkinter, customTkinter, Python

Python- Python is a high-level, general-purpose programming language. Its design philosophy emphasises code readability with the use of significant indentation. Python has good file-handling options which make it suitable for our purpose.

Tkinter - Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit and is Python's de facto standard GUI. It will help display the images on the screen and build a good GUI.

customTkinter - CustomTkinter is a python UI library based on Tkinter, which provides new, modern, and fully customizable widgets. They are created and used like regular Tkinter widgets and can also be used in combination with normal Tkinter elements. The widgets and the window colours either adapt to the system appearance or the manually set mode ('light',

'dark'), and all CustomTkinter widgets and windows support HighDPI scaling (Windows, macOS). With CustomTkinter, you'll get a consistent and modern look across all desktop platforms (Windows, macOS, Linux).

Github repo: https://github.com/TomSchimansky/CustomTkinter

2.2 System Requirements Specification

2.2.1 Functional Requirements:

Input

The application should be able take in

- Paths of Folders
- Number of Folders
- Filename(Optional)

Open Folders

The application should be able to open folders and provide the user a UI to navigate the folder and select an image.

Change Number of Folders

The application should be able to change the number of folders to more than two so that multiple images can be visualised.

Image Display

The application should be able to open and display multiple images simultaneously on the screen from different folders.

Image Processing

The application should be able to process images such as zoom, crop etc.

2.2.2 Non-Functional Requirements

- The system shall have a user-friendly interface that is easy to use and understand.
- The application should load quickly and the image visualisation should be smooth and seamless.
- The application should be able to handle images of different formats simultaneously.
- The application should be able to integrate with other software systems and API’s, allowing for seamless data exchange.

- The system shall be able to run on a variety of devices and operating systems, including desktop computers and laptops.
- The application should be easy to maintain and update with well clear documentation and well organised code.

2.3 System Evolution

There are several potential features or improvements that could be added to our image visualisation software in the future, such as:

- Batch processing: Adding a batch processing feature that allows users to apply the same image processing algorithm to multiple images at once, saving time and effort.
- Multi-platform compatibility: Expanding the software to be compatible with multiple operating systems, such as MacOS or Linux.
- Cloud storage integration: Allowing users to store and access their images on cloud-based storage platforms like Google Drive or Dropbox.
- Machine learning integration: Integrating machine learning algorithms to automate or assist with image analysis and processing, allowing for more advanced feature extraction or pattern recognition.
- Image editing tools: Adding more advanced image editing tools, such as cropping or colour correction, to allow users to fine-tune their images before analysis.
- Image comparison metrics: Providing metrics or quantitative measures for image comparison, such as the mean squared error or structural similarity index, to help users objectively evaluate different image processing techniques.

Ultimately, the direction of the project's development will depend on user needs and feedback, as well as the availability of resources and development expertise.

3 Appendices

3.1 Hardware Requirements:

- Computer device: The application will require sufficient processing power and memory to handle our software.
- Storage: The application may require additional storage space on the device to store the names of images and arrange the name list as needed for the software.

3.2 Software Requirements

- Compatibility: The software should be compatible with various operating systems such as Windows, Mac, and Linux.

- User interface: The user interface should be intuitive, user-friendly, and easy to navigate. It should allow users to select folders, find image pairs and view them side by side.
- Image format support: The software should support various image formats, including popular formats such as JPEG, PNG, and BMP.
- Performance: The software should be optimized for performance and capable of handling large image files without any lag or delay.
- Permissions: The user may need to grant the application permission to access the storage.
- Documentation and Support: The software should have proper documentation and customer support to assist users in case of any issues or queries.
- Security: The software should have robust security features to ensure the confidentiality and privacy of the user's data.

Overall, the software requirements for the application's end user are relatively simple and familiar to most computer applications. The main requirement is that the user has a compatible computer device and access to the storage to read the names of images present in folders.