Informatics Institute of Technology

In Collaboration With

University of Westminster, UK

"TransformVector"

An automated raster to vector conversion platform for GIS (Geographical Information Systems)

A dissertation by

Damodaran Dillon Thisaru Lakshman

Supervised By

Mr. Pumudu Fernando

Submitted in partial fulfilment of the requirements for the

BSc (Hons) Computer Science

Department of Computing

Declaration

I hereby certify that this project report and all the artefacts associated with it is my own work and it has not been submitted before nor is currently being submitted for any degree programme.

Full Name of Student: Damodaran Dillon Thisaru Lakshman

Registration No: w1626703/ 2016086

…………………………………… ………………………………..

Signature Date

Abstract

Raster images and Vector images are two popular formats that can be used to store image type data. Raster images are created using an array of pixels with values in a set range to depict images when arranged to a grid. Vector images are images that are created employing mathematical functions to draw straight lines and quadratic lines between points on a coordinate system to create polygons that combine together to form an image. A Raster to Vector conversion tool, converts Raster images to this polygon based vector image format by using various algorithms such as Line Thinning, Gap Removal etc. It traces the paths identified using these algorithms; that process the pixels of images to convert the final output into a composition of polygons. Raster graphics and Vector graphics are used widely in Geographical Information System for their processing.

When considering the various approaches used for raster to vector conversion used currently there seems to be a trend where the conversion tool or software that is used, converts the image without any considerations to the properties of the image being converted. Therefore, a system has been proposed that identifies this gap and allows the parameters of the algorithms that are used for conversion to be studied and used in a process to identify the best said parameters when converting a specific image. This can be achieved through an image classification system. And by training a model and creating data regarding how a parameter affects a certain classification of image so that when a new image of similar classification is to be converted uses parameters more suited to its structure.

This project is name TransformVector. The research is carried out for the parameter identification component of the project and for pre-processing of images before conversion to optimize the process of conversion. An API base approach will be used to access the functionality of the project by the front end application that is to be developed. The project converts GIS imagery of selected classifications from Raster into Vector after pre-processing it and analysing to determine the best fit parameters.

Keywords:

Image Pre-Processing, Raster to Vector Conversion, Image Classification, Convoluted Neural Networks, Application Programming Interfaces

Acknowledgement

Throughout the final year project journey, I came across many challenging situations. Nonetheless, it was a delightful experience all thanks to the people who stood beside me and supported me throughout the project being pillars of strength.

First, I would like to express my sincere gratitude for my supervisor Mr. Pumudu Fernando for the excellent guidance and the supervision during the project.

I want to express my gratitude for the Informatics Institute of Technology, the lecturers, industry experts and my colleagues who supported me in many ways to achieve my project.

Most importantly I would like to thank my loving parents Lakshman Damodaran and Ruklanthi Damodaran for the tremendous support given to me throughout my life.

Table of Contents

List of Tables

List of Figures

1. Chapter 1: Introduction

1.1 Chapter Overview

This chapter covers the overview of the project, it's background, the justification of the problem being addressed and the research question relevant to this research. The motivation behind attempting this research is also explained. Furthermore, the Aim and the scope that will be attempted in this research will also be defined. The Research objectives that will be fulfilled and also personal objectives of conducting this research will be mentioned along with an overview of the solution. The requirements for such proposed solution will also be defined.

1.2 Project Overview

1.2.1 Project Background

Images

Human beings can be considered predominantly visual creatures as we use our sense of vision when available to understand our surroundings. Due to this reason we have created images to capture moments of this ever constantly moving world. Images have since then evolved to not only capturing real world moments but to digital drawings and other representations of visual still media. An image can be a single picture which represents a certain object, location or scenery. When image files are attempted to be classified, two main classifications can be identified. Vector images and Raster images. There are different pros and cons of using either a Vector image or a Raster image.

Raster Images

A raster image is built up of colour pixels which are arranged to form the necessary result image. Raster images are mostly suited for linear art images because they can better represent subtle chromatic gradients due to the fact that each pixel can change its value independently of other pixels to form the necessary image. These types of images are also called continuous tone images.

A raster image while being faster to process and display as there is no methodical processing or such involved will have a larger file size as information is electronically stored on a pixel-by-pixel basis (Winnemoeller *et al.*, 2018), the size of the image is directly proportional to the resolution of the image

Vector Images

Vector images on the other hand are created from points and lines and curves joining them. These are based off of mathematical formulas that create combinations of multiple true geometric primitives to create a final image (Seel-audom, Naiyapo and Chouvatut, 2017)..

Vector format based computer graphics tools have become very powerful tools allowing artists, designers etc. to mimic many artistic styles, exploit automated techniques, etc. and across different simulated physical media and digital media (SEVERENUK *et al.*, 2019). Similarly, in real world applications according to the factors that need to be considered, the image may be required in either Raster or in Vector format.

As stated above, as a vector file and raster file of the same image may have similar resulting image, when observed on a deeper level multiple differences can be identified between them.

1.2.2 Problem Domain

Geographic Information System (GIS) is the processes of managing, manipulating, analyzing, updating and presenting metadata according to its geographic location, to be effectively used in different aspects of life (Al-Bayari, 2018). The increasing popularity GIS technology has increased the usage of spatial data. Making maps is relatively easy even for those who do not have much cartographic training (Wong and Wu, 1996).

According to the analysis needed to be performed on a certain image obtained, the requirement for a Raster image or a Vector image may vary. There are several pros and cons when you consider each type of image. There is an old GIS adage stating that “Raster is Faster but Vector is Corrector”.(Berry, 1995).

Vector GIS results in the geometrization of the geographical world, and generalizing and reducing its theory into theories about relations between points, lines, polygons and areas. Such objects which are in a GIS can be counted, moved about, stacked, rotated labelled, cut etc. and be handled like a variety of other everyday solid objects that bear no particular relationship to geography (Couclelis, 1992).

Vector maps use simple geometric components such as points, lines and polygons in adjacencies, unions and inclusions to describe spatial information and Raster maps are based on pixel matrices and are richer and realistic than vector maps (Lin and Guo, 2011).

There are several advantages of using a Vector data format. These can be stated as the output being more aesthetically pleasing and zoom able to very close detail as it is made up of points and line segments connecting them and not using fixed number of pixels which might look pixelated and less clear when zoomed into more than its resolution allows. It also provides higher geographical accuracy due the same reason as it being not of a fixed resolution. There are other reasons as why vector images are used in GIS such as data integrity, and allowing network analysis and proximity operations as they both use vector data structures. As well as there are advantages are there are disadvantages to using vector images as well. As these images are a result of mathematical calculations it is often very processing intensive. Vector data structures are also poor performing when displaying continuous data, and needs to be generalized in some manner to display, which can result in loss of some information.

While vector data structures in GIS over determines the geographic world by forcing it into a geometric objects generalizing them, the raster data structure feigns maximal ignorance on the nature of things in the world. Yet Raster data structures provide an implicit view of the geographical world with measurable values discretized into pixel arrays (Couclelis, 1992).

Raster data can store unique values per each pixel without any generalization being required. Therefore, is a good option when continuous data is required to be displayed. Even though continuous data can be very accurately represented in a Raster image, because of the resolution. Raster graphics display devices are capable of reproducing very complex images (Sloan and Tanimoto, 1979). It struggles when representing linear features and can cause pixilation if the resolution of the image obtained is low or when zoomed to obtain a closer look. Raster datasets can also be very large file as when the resolution is increased to get a more accurate image with high detail the file size increases proportionally with it.

From the statements above, it can be identified that both these formats are equally important when considering the use of imagery in GI Systems. Therefore, there becomes necessary a method of conversion between these two data types.

1.2.3 Problem Justification

Automated conversion of engineering drawings and such similar content into Raster and vector data has been a very widely discussed topic. A critical step in this process can be considered as the conversion of these images into a vector format (Liu and Josep Lladós (Eds.), 2005). Many techniques for conversion of raster to vector have been proposed which has even led to development of commercial solutions to tackle this issue. The systems created all did provide quite acceptable results but each had their own drawbacks (Hilaire and Tombre, 2006). (Lacroix, 2009)

1.2.4 Research Question

How can the Raster to vector conversion process be improved to obtain better quality outputs by changing parameters that affect the conversion process?

1.2.5 Motivation

After researching on the basic concept of converting Raster images into Vector images for graphic design purposes, the author has come to find the importance of it but in a different domain which is in the field of GIS. Raster and Vector data structures are widely used in analysis in GIS and as both of these type of images are needed according to different situations. It has motivated the author to create this automated Raster to Vector conversion tool.

1.3 Project Aim and Scope

1.3.1 Project Aim

To investigate design and implement a Raster to Vector conversion platform that selects the best method of conversion using image processing techniques.

1.3.2 Project Scope

In-Scope

- Raster to Vector conversion tool is only developed geared towards GIS
- Training an image processing model to identify certain properties of images that affect conversion algorithms.
- Integrating Image processing for the identification special characteristics to identify the best conversion algorithm
- Considering of continuous tone images as well as line based images for the conversion process

Out-Scope

- Conversion of Raster to Vector for other domains such as graphic design.
- OCR functionality out of image text is not considered, and will be represented in the converted images as graphical data and not textual data.
-

1.4 Objectives

I have identified the following as the objectives to achieve to complete my Research successfully.

1.4.1 Research Objectives

- To conduct a thorough literature review on existing solutions and platforms
- To design an image processing model to identify classifications of a Raster image related to GIS.
- To implement functionality to determine the most favourable parameters for the conversion through the parameters identified from the image processing process.
- To evaluate the converted Vector image in terms of accuracy and speed of conversion.

1.5 Overview of the Solution

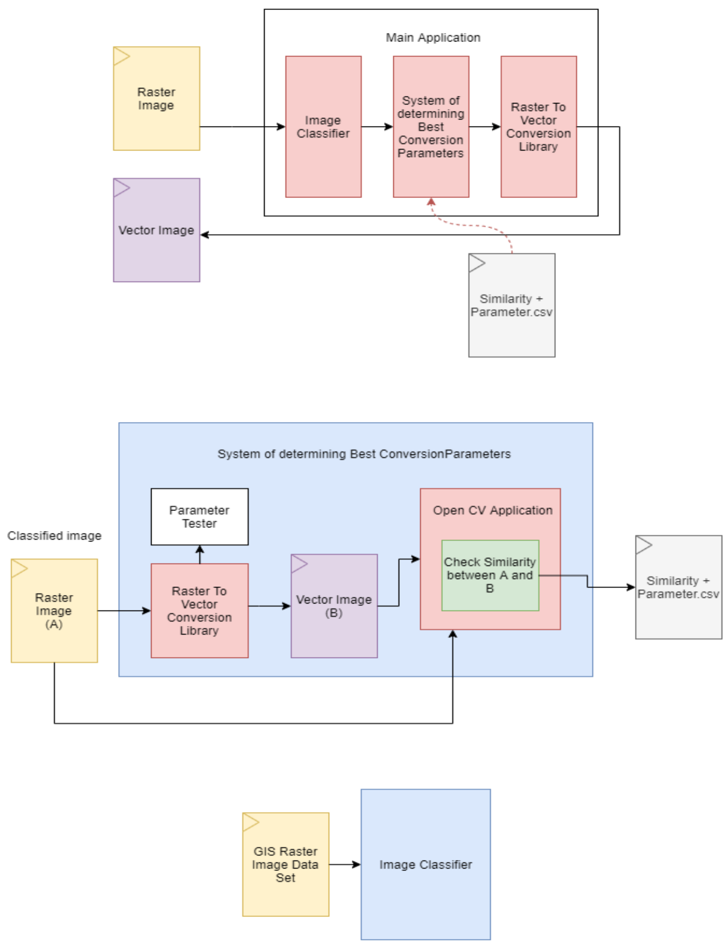


Figure 1.5:1 Rich picture Diagram of the system

As shown in the figure above the main system is made up of three components. The image classifier, system of determining best conversion parameters and the raster to vector conversion library.

When taking a closer look at each system; the system of determining best conversion parameters is a machine learning based system that finds the best conversion parameters by interpolating data from a set of previously identified conversion parameters for each image classification and the match rate of the image to its particular classification determined by the Image classifier.

The initial best fit parameters are identified by converting a single image into multiple vector images each time with different parameters within a certain range of each parameter and comparing its visual similarity with the original using an image processing library. This similarity is then recorded in a csv file for the system of determining the best conversion parameter to use.

A data set of several hundred labelled GIS image types are used to train the image classifier.

The raster to vector conversion library is used to convert the images from its raster to vector format in each step that its necessary to do so.

These three main components make up the main system and together convert any given GIS based image which can be classified into an accurate vector image.

1.6 Resource Requirements

1.6.1.1 Software Requirements

- **Windows 10 (64-bit version):** To accommodate and run the software
- **Java, Python:** For the conversion algorithms to function
- **NodeJS:** For backend related scripting
- **MSWord:** For documentation Requirements
- **GraalVM:** For cross platform application support

1.6.1.2 Hardware Requirements

- Core i7 Processor-High processing power required for algorithms to be executed
- Minimum 4GB RAM - application to run smoothly and not run out of system memory
- Disk Space: Up to 10GB - For storing of application and images and temporary files created while converting and running algorithms

1.7 Chapter Summary

The project overview was discussed which further explained the Project background, Problem domain, Justification and the Research question and the motivation behind this research. The project Aim and the scope of the project was also discussed. The research objectives of this project and the overview of the solution proposed was also discussed. Finally, an estimation of the software and hardware requirements of the system was discussed.

2. Literature Review

2.1 Overview

This chapter will provide a critical analysis on the domain of the problem based on literature. This chapter will analyze the current methods of raster to vector conversion why they are needed, the different variations of raster to vector conversion system. This chapter will also review similar work for the process of conversion of Raster to Vector.

2.2 Problem Domain

2.2.1 Geographic Information Systems

Geographic Information Systems (GIS) are computer based tool for process of manipulating, updating, managing, analyzing and presenting data of geographic locations and cartographic data which can be thereafter used in different fields of study or be applied in daily applications which require such data. (Al-Bayari, 2018) (Chrisman, 1999).

Due to proliferation of GI science and the usage of spatial data, map making has been simplified even for someone with little to no experience in cartographic training (Wong and Wu, 1996).

During the earlier stages of GI Science, the definition of the it simply putting maps into computers. The process of implementing this idea however was a tedious task and to complete satisfactorily was a major undertaking. Recently however the limitation of spatial data in 2 dimensional vertical view of the surface of the globe is not as sought after and scientist demand for 3 dimensional views and the ability to simulate various geographic processes though GIS data (Gold, 2006).

This GIS information can be stored and processed in image formats after being generated. There are two main types of image structures that can be used when storing this information. This can be classified as Raster data and vector data (Wade *et al.*, 2003).

2.2.2 Raster Image Data

Raster images, which are also known as bitmap images are categorized under digital images as being formed of tiny rectangular pixels which are arranged in a grid formation that combines together to represent an image. This format of image can support a wide range of colours and depict subtle gradients, it allows for a very accurate visual representation of continuous tone images such as shaded drawings, photographs and other highly detailed and complex imagery.

Raster graphics initially originated in television technology with images constructed much like the pictures on a television screen. Raster graphics are made of small uniform sized pixels which are arranged in a two dimensional grid which is made up of vertical and horizontal columns. A single pixel contains information of a single or multiple bits depending on the degree of information in the image. For example, black and white images contain only one bit per pixel in the image this can either be a true black bit or a true white bit, an image with shading or colour commonly contains 24 bits of information per pixel, this allow more than 16 million possible states of colour value for the pixel. Images with 24 bits of information per pixel are known as ‘truecolor’ images with 24-bit colour and can realistically depict colour images. The amount of detail stored into a single bit is represented by the colour depth and the number of pixels that form together to form the image can be represented as its resolution and affects how much detail is depicted in an image (Britannica, 2014).

2.2.3 Vector Image Data

Vector graphics are created using a sequence of commands or mathematical formulas that connect edges and nodes in a 2 or 3 dimensional space to render an image. In the field of physics, a vector is the representation of a quantity that contains both a direction and a quantity or value. For example, instead of storing the data as an array of pixels as when a raster images or bitmaps, a vector file creates its complex images by mathematically aligning and stacking multiple polygons created by connecting a series of point with lines segments. This results in the creation of typically smaller files with extremely high fidelity, lossless compress and scalability of any kind without the distortion or loss of fidelity of it (He, Xu and Zheng, 2009).

2.2.4 Need for both Raster and Vector based data in GI systems

According to the type of processing and analysis to be performed on a certain data set obtained the requirement for the format of the data type, either Raster or Vector may be different. Each data type has its advantages and disadvantages, strengths and weaknesses and to further solidify the need for both data types an old GUS adage states that “Raster is Faster but Vector is Corrector” (Berry, 1995).

A well-known logical consequence of the difference of the data structures vector and raster is that as while vector data can record position to and degree of accuracy, raster data have a built in level of positional accuracy. Therefore, raster positional data can be classified as integer and vector positional data that are real(Holroyd and Bell, 1992).

GIS data in vector format is produced by geometricizing the real geographical world, generalizing it and reducing it into theories about points, lines polygons and areas. Objects thus created using these theories can be counted translated, stacked, rotated, labeled, cut and etc. and can be handled like every day geometrical shapes that are not directly related to geographical data in any manner (Couclelis, 1992).

While vector maps use simple geometric structures as points, lines, polygons and relationships between them such as adjacencies, unions and inclusions to depict the geographical data, Raster based maps are stacked matrices of pixel based data which can be layered in a sandwich like structure and are more rich and realistic than vector data (Lin and Guo, 2011).

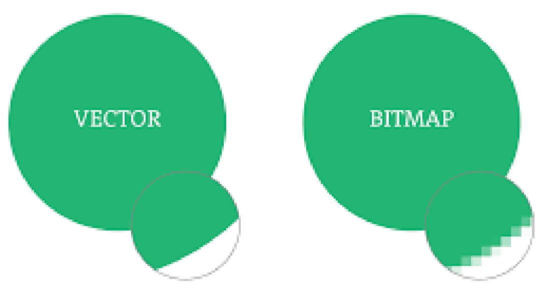


Figure 2.2:1 Visual comparison between raster and vector image fidelity

Advantages of using a vector data format can be stated as the output produced being more aesthetically pleasing, highly zoom able as it is made up points and line segments contented to each other to form multiple polygons and not a pixel array which when zoomed into more than its set resolution allowed may look pixelated. There are a number of other advantages of using a vector data structure rather than a raster data structure, and these can be listed down as follows.

1. Data integrity
2. Allowing network analysis
3. Proximity operations as they both use vector data structures

While there is strength in using vector data structures to represent and process GIS data, there are weakness of it as well. As vector data is produced by the result of geometrical mathematical calculations, this data produced can be very processor intensive when for example depicting continuous data regarding a point if stored in vector format is much more complex than a single pixel storing information which makes up a raster image. Therefore, due to the feasibility of the idea of storing data as a series of points made up of geometrical data being not practical these images must be generalized in some manner to ease storing and processing when needed to be displayed. This result in the loss of some portions of information.

Even though GIS data that is stored as vector data structures over determines the geographic world by generalizing them and forcing them into geometric objects, Raster data structures provides an implicit view of the world with measurable values discretized into pixel arrays but simulates ignorance on the nature of an actual object in a physical space (Couclelis, 1992).

A simple advantage of using raster data over vector is the ability to store continuous tone images as pixel array which stores unique values of data per pixel without performing any generalization which may lead to data loss in some manner. But while continuous data can be stored very accurately using raster images, as the resolution of a raster graphic is set and cannot be varied, it may limit the fidelity of final output if the process requires the image to be enlarged. Raster graphics display devices are capable of reproducing very complex images (Sloan and Tanimoto, 1979). Raster images also have the weakness of not being able to produce true linear features as pixels used to form as arranged horizontally and vertically, are limited when reproducing line data at any other angle. Raster data sets also have very big file sizes as the higher the resolution the more pixel data it contains. Therefore, the file size is directly proportional to the resolution of the image, and by resulting the file size of the image to be directly proportional to the visual fidelity of the image.

2.2.5 Raster to Vector Conversion in GIS

Spatial data obtained as aerial imageries, for example is very continuous in nature and must be represented as raster data. These types of maps are unable to be displayed by vector maps (Lin and Guo, 2011).

With development of high speed computing and its usage in a wide range of fields of study leading to improve process times and increasing the ability of highly computational tasks to be performed on smaller and more affordable system. Vector data can be used as a much more accurate data structure as the images even with complex processing requirements can be easily displayed and rendered faster than before (Winnemoeller *et al.*, 2018). Vector images are also known for its flexibility and compact nature etc.(Lin *et al.*, 2015). By going through the above use cases for each raster and vector images, the conclusion can be made that both of these data structure is widely used in the Geographical Information Sciences. Therefore, there arises a need for conversion between these data structures. Even though many commercial solutions and researches have been conducted in achieving a satisfactory result, they each have their drawbacks and hence arises a need for a common solution to this problem.

2.3 Image Processing

Digital Image Processing is the process of manipulation of images using computers. Image processing use cases have been increased exponentially during recent times. Its application ranges from entertainment, passing by geological processing and remote sensing. Multimedia systems which play a huge role in modern society also heavily rely on image processing.

When studying the various disciplines of image processing it can be identified that it is a vast topic with many techniques that are applied specific to the type and qualities of an image. An image can be regarded as a function $f(x, y)$ of two continuous variables x and y. *For images to be processed digitally the image has to be sampled and transformed into a matrix of number. And quantization is then required as a computer represents number using finite precision. Image processing then can be identified as the manipulation of those finite precision numbers.*

Image processing can be mainly categorized into (Eduardo A.B. da Silva, 2005).

- **image de-noising**
- **image restoration**
- **image analysis**
- **image compression**
- **colour quantization**

Some image pre-processing methods suitable to enhance the solution proposed will be discussed further below.

2.3.1 Image de-noising

An important step in image pre-processing can be identified as image de-noising. Image de-noising is the process of removing various kinds of noise from an image. Image noise can occur during transmission and acquisition, storing etc. of an image. This added noise can be reduced by estimators using prior information on a signals' properties to remove it (Li and Kong, 2009). Three main approaches to image de-noising can be stated as Spatial Filtering, Transform Domain Filtering and Wavelet Thresholding Method. These filtering approaches are used to Suppress noise, preserve edges and other image characteristics and finally, create a visually natural appearance. Out of the approaches mainly used for spatial filtering is a method of using spatial filter techniques to combat noise. There are several types of spatial filtering available (Su *et al.*, 2013). Types of spatial filtering Linear Filters, Mean filter, Wiener Filter, Median Filter Etc. A suitable image de-niosing filter approach will be selected out of the above mentioned approaches during the implementation phase of this project.

2.3.2 Image quantization

Image colour quantization is another common image pre-processing technique that can be favourable to use in this project. Colour quantization in images is the process of reducing the number of colours present in a colour image with less distortion. The main purpose of using colour quantization in this project is to make it easier for the vector conversion algorithm to group polygons when converting continuous tone images and to reduce processing time and power required (Su *et al.*, 2013). Colour image quantization mainly has two steps determine a colour-map with smaller number of colours and mapping each pixel to that colour map (Alamdar, Bahmani and Haratizadeh, 2010). Some popular image quantization algorithms are SaDE-CIQ, K-means and PSO.

2.4 Comparison of Similar Solutions

Name of Software	Supported Image Formats	Max Zoom	Adjust Image	OCR	Scripting Support	Ability to adjust conv. params
Easy Trace	ArcINFO, ArcView, AutoCAD, Credo, MapInfo, MicroStation	12800x	Yes	No	No	No
Free Hand	PNG, JPEG, TIFF, SVG, EPS	64x	No	No	Yes	No
Illustrator	PNG, JPEG, TIFF, SVG, EPS	640x	No	No	Yes	No
Potrace	PBM, PGM, PPM, or BMP (Bitmap only)	N/A	No	No	No	Yes
Scan2Cad	PDF (Scanned files)	64K	No	Yes	Yes	No
Image Tracer	PNG, JPEG, TIFF, SVG, BMP	N/A	No	No	Yes	Yes

Table 2.4:1 Feature comparison of Similar solutions

2.4.1 Easy Trace

The easy Trace pro software package is a convenient and powerful tool for vector map generation with editing capabilities. The program also is widely used in GI systems and supports most of the import and export formats such as.

- ArcINFO
- ArcView
- AutoCAD
- Credo
- MapInfo
- MicroStation

This software application is only functional on a windows operating system. It also contains raster to vector conversion tools and is create by the Easy Trace group (Russia). Old versions are free versions but all newer updated version are paid versions of this software. The drawback of this software comes when the fact that its conversion parameters cannot be modified and as it does not support scripting is considered.

2.4.2 FreeHand

Adobe Freehand (a.k.a Macromedia Freehand) is a commercial computer application for creating vector graphic content. Created by the Altsys Coporation in 1988 and licensed to Aldus corporation this software released version 1 through4 in 1994, and was discontinued since 2003. This application also allowed a user to convert a placed raster image into a vector. This application is now discontinued.

2.4.3 Illustrator

Adobe illustrator is an industry grade application for multimedia, online and print graphics creation. It provides tools to create technical illustrations or graphic related content for print publishing and also allows web related graphic content creation. It is a commercial tool. This software also allows a functionality known as image trace which allows the conversion of images from raster to vector. As this Raster to vector conversion is only a small part covered in this digital graphics creation software and it's lack of support for scripting and parametrization of conversion options, the author has moved to other solutions that can be used in the project.

2.4.4 Potrace

Portace(TM) is a bitmap tracing tool which allow the transforming of bitmaps into raster based scalable images. The input type of the bitmap can be of any of the following formats

- PBM
- PGM
- PPM
- BMP

And the output will be into a vector format file. A usecase of this software can be create SVG or PDF files from scanned raster images. The resulting vector image then can be scaled to any resolution without any distortion in the image quality.

Protrace uses the output formats

- SVG
- PDF
- EPS
- PostScript
- DXF
- GeoJson
- PGM

It is a software package that is currently in development and further image type support will be added in future times. Portrace does not preprocess the image before it is converted to a vector file format and this has to be performed beforehand by a different application. Though this application is a great software when it comes to doing what it is designed to do. The support for only BITMAP and grayscale line drawing output, really puts it in a bad spot when being considered to be used as a tool in this project.

2.4.5 Image Tracer

Image Tracer is a simple Raster to vector conversion tool. It is an open source software and has an active github that fixes and improves its code base. It allows the ability to modify its code as necessary for use and also provides the ability for its conversion parameters to be adjusted by the user for each conversion to get the best output. It supports the following Input formats.

- PNG
- JPEG
- TIFF
- SVG
- BMP

This application can be built and deployed as a Java application and can be executed using wrappers with other programming languages and also through the command line. As its output image quality is also of a high quality and as it also supports many features required on this project. Image Tracer is identified most suitable to be used for Raster to Vector Convesion in this project.

2.5 Algorithmic Analysis

2.5.1 General Raster to Vector Conversion Algorithms

The task of converting a raster based data structure in to line data or into vector based data structures can be divided into three basic operations. These are as follows

- Skeletonization/Line Thinning
    - This is process of reducing or thinning the thickness of the lines found in the image to a unit thickness relative to the given resolution
- Line Extraction/Vectorization
    - The process of identifying a series of data entities or coordinates that are used o form a single line segment in the input document

- Topology Reconstruction
    - The process of determining the adjacency relationship among the lines identified during line extraction.

The individual line segments are joined into whole lines if desired and can also be combined as polygons for continuous representation. Two other additional post processing tasks that can applied onto the basic raster to vector conversion process can be stated as follows.

- Line smoothening
- Gap Removal

These steps can be understood as the general steps of process when conversion of raster images to a vector format is considered. As for this project the following process of the Image tracer library has been employed. Justification as to why this method was selected as well as a process overview will be discussed in the next section

2.6 Image Tracer Raster to Vector Process Overview

Initially colour quantization for the image is carried out. This colour quantization is done using the indexed colour method. This method is a commonly used colour quantization technique to manage colours in a digital image to reduce its size. This helps in further computational tasks and reduces the number of colours therefore making polygonization of continuous colours easy moving forward. After colour quantization is complete layer separation and edge detection of the image is carried out. The layering function creates an array for every colour and calculates edge nodes based on the data obtained. These are at the center of every 4 dots as shown in the illustration below.

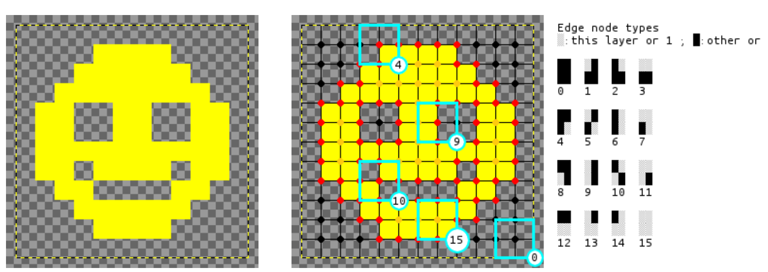


Figure 2.6:1 Edge Node Detection

After this step, a path scan is function finds and chains the edge nodes of the image. Once complete, the internode function interpolates the coordinates of every edge node path and finally the interpolated paths are traced to obtain the basic paths of the image.

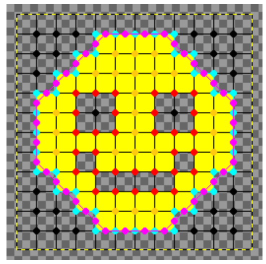


Figure 2.6:2 Interpolation

A function is then carried out to draw a connection between two lines to fit a straight line on the start - and endpoint of the sequence. If the distance error between the calculated points and actual sequence points is greater than the threshold, the point with the greatest error is selected.

Then a quadratic spline through the error point determined in the previous step is drawn and fitted appropriately. Finally, the coordinated are rendered into SVG paths.

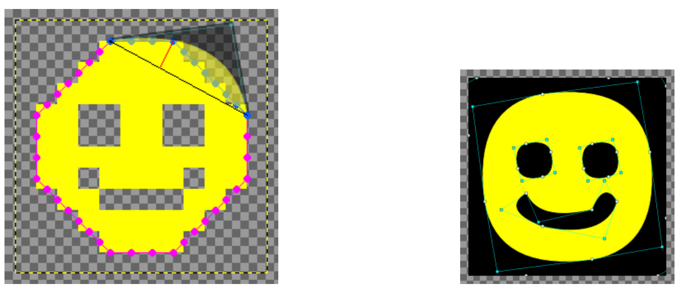


Figure 2.6:3 Fitting quadratic spline and generated view of SVG

2.7 Justification of the Selected Approach

As the literature review has been conducted, the various similar solutions available on a commercial level and on a research level are identified to not address identifying the qualities of the raster image before conducting the process of conversion of the image from raster to vector. Hence this project will be further looking into the usage of image processing to identify various image processing techniques for the pre-processing of the image before converting it from raster to vector and also identifying the most suitable parameters for the algorithm to be executed on for an accurate conversion also using more image processing techniques which will be selected and highlighted in the implementation chapter of this document.

From the raster to vector conversion products and libraries available in the market, potrace and image tracer were selected as main candidates to be used in this project due to its compatibility with the approach selected in this project as conversion parameters used when converting could be set by the usere. And as for the raster to vector conversion algorithm that has been selected and described in the process overview in section 2.6 of this chapter. It can be seen that Unlike the potrace algorithm which can only work with bitmap type images and can only perform line thinning in a grayscale environment. The selected image trace algorithm suits the GIS type image conversion better as it can convert continuous tone images and not only does line thinning to find vector paths but can also draw quadratic shapes which can be filled to obtain polygons.

2.8 Research Gap addressed

The research gap to be addressed in this project is the fact that Raster to Vector Image conversion systems found in the market and that are developed on a research-level mostly operate on fixed parameters on their conversion process and algorithms. And when application does have these parameters for access of the user to change to streamline an image output to be of a better quality and more suitable to the type of image which is to be converted. The user may not have enough knowledge on the factors that each of these parameters and their values might do the result output image obtained after conversion. Due to this, there arises a need for an application to analyze the image, classify it and identify the best parameters required for a better quality and accuracy to be obtained in the resulting vector image that is produced from the conversion process.

Image pre-processing is also packaged together with conversion in this conversion process so the user can customize as they see fit for image quality and other options such as de-noising and colours to be used in the resulting image through quantization.

2.9 Chapter Summary

As outlined in this chapter, an image processing solution to identifying the best conversion method and parameter identification when raster to vector conversion is considered is discussed. This is to combat the fact of there not being a standard when identifying the best method of conversion from raster to vector. A number of raster to vector conversion tools have been reviewed and researched on for this purpose. The image tracer conversion process was selected as the ideal conversion process to be used in the project. Several image pre-processing methods were also described.

3. Chapter 3: Methodology

3.1 Chapter Overview

In this chapter the methodology behind the research approach which has been selected for this research will be discussed. The research paradigm and the research methodology selected will be elaborated and justified. The design methodology and the development methodology of the project will also be discussed.

3.2 Research Methodology

This research is based on the conversion of the process of Raster to Vector in classified GIS imagery with the use of best-fit conversion parameters. The theory of the research is based on a hypothesis that has been established as follows. *"Images that have similar characteristics and can be classified as such using an image classifier can have a similar set of best-fit conversion parameters for Raster to Vector conversion to obtain optimal results of conversion."*

3.2.1 Research Paradigm

When considering the different approaches that can be considered when approaching this research topic. It can be identified that there are three clear approaches that have been established by researchers. These are the approaches of 1) Positivism 2) Interpretivism 3) Critical theory. This is essential because as consumers of research, researchers have to be able to look deeper into claims made by researchers who adhere to different research paradigms (Rehman and Alharthi, 2016).

As a hypothesis is established and the testing to obtain empirical data to form a relation between the phenomena is carried out. The research paradigm of this research can be concluded to be of a **positivism approach**.

3.2.2 Research Methodology

When considering research methodologies, there are two main types of methods that researches generally can be categorized under. 1) Deductive approach 2) Inductive approach. The deductive approach starts with specifying the objective. The inductive starts through an observation and arriving to a conclusions through the observation (Zalaghi and Khazaei, 2016).

As the hypothesis that is being established at the beginning of this section is clear, and the work of this research is to prove its validity. This research can be identified to be of the **deductive approach**.

3.3 Design Methodology

Two main design methodologies can be considered to be used to design the proposed system. These can be identified as Object-Oriented Analysis and Design (OOAD) and Structured Systems Analysis and Design Method (SSADM). Each of these methods have their pros and cons according to what kind of system is being designed. In this section factors that can be used to identify the most suitable approach for designing the proposed system will be discussed.

In the Object-Oriented analysis and design approach initially, requirements are identified and a software specification and documentation is developed in terms of an object model. These are objects that integrate both data and function ns and are modeled after real-world objects. After these objects are identified it is then mapped into classes and constraints and relationships are identified. This methodology gives access to certain OOD principles such as classes and objects, encapsulation, polymorphism and interfaces/abstract classes. OOD can be considered a good design methodology to manage complexity in applications and to enable the reusability of components.

In the structured system analysis and design methodology graphical tools are used in a systematic approach to refine objectives out of well-defined user requirements. In the case of these requirements not being clearly described initially, it can lead to problems in the process of the solution created. This approach also does not accommodate dynamic user requirements that might be subjected to change through the development life cycle of the solution.

When considering the above factors as the requirements of the proposed solution may be subject to change through the life cycle of this research. The **OOAD** methodology is better suited for the analysis and design methodology of this project.

3.4 Development Methodology

When starting a project that has as purpose the software development, it is very important to use a methodology that increases its success rate. The success rate of a Project can be improved by using a methodology that fits the characteristics of the project. Multiple software development methodologies that can be employed when determining the development life cycle of a software application. These can be identified as follows. Waterfall, Agile, Feature-Driven, Iterative, Spiral, Prototyping and RAD (Rapid Application Development). The following features that might be considered as factors that affect which mentioned methodology is most suitable for this project can be identified due to their repeated occurrence during this project.

- The initial requirements are identified and recorded in the Project initiation document.
- These requirements are subjected to change over the development of the project due to feedback and development constraints
- Development is done by breaking down the project into components
- Development will be done component by component.
- Each component will have the ability to be tested independently.

The Agile development methodology can be considered an iterative and incremental process that focuses on rapid delivery of working prototypes that finally create the working product in incremental builds. When considering the above stated factors, it is evident that an **Agile** Software development methodology is better suited for the development methodology of this project.

3.5 Chapter Summary

As to conclude this chapter, the projects Research methodology and Research paradigm has been selected as the deductive approach and the positivism approach respectively. The Design methodology of the project is also selected as to the OOAD (Object Oriented Analysis and Design) methodology. Further the chapter also mentions selecting Agile Development Methodology as the suitable Software development methodology of this project as the software as the initial requirements identified may be subjected to change during the course of the project and a dynamic development methodology is thus required to adapt and overcome the challenges that follow along with such a project lifecycle.

4. Chapter 4: Project Management

4.1 Chapter Overview

This chapter will discuss the Project Management Methodology of the project. Various project management mythologies will be evaluated and a suitable methodology will be selected as to fit the characteristics of this project. The potential risks and the mitigation plan in a case where these occur will also be discussed. Further the work breakdown structure and the project time line as a Gantt chart will be described. Finally, the Social, Legal, Ethical and Professional Aspect along with the chapter summary will be included in this chapter.

4.2 Project Management Methodology

There are many unique project management methodologies that have been tested and approved by many industry professionals. Some of these can be stated as Agile, Scrum, Kanban, PRINCE2 etc. An agile development methodology can be simply defined as iteratively trying to deliver whatever works at given equal stages of time. Scrum can be defined as using a small cross functional team to deliver result fast. Among these project management methodologies, a closer look at the PRINCE2 Project management framework is carried out.

PRINCE 2 is a project management framework that is scalable and can be adaptable to any type of project with ease. PRINCE 2 focuses on dividing the work process of a project into manageable and controllable stages. PRINCE 2 is a widely used standard among many professionals in the industry and is a flexible project management methodology that can accommodate with changing requirements of a project. PRINCE 2 is based on seven principles. These can be stated as follows, ***Continued Business Justification, learn from Experience, Defined Roles and Responsibilities, manage by Stages, manage by Exception, Focus on Products, Tailor to Suit Project Environment.***

When considering these principles, and as it is evident that the project being considered in this research does have tendency for requirement alteration as progress is being made and there is much to learn and discover through the life cycle of the research. PRINCE 2 can be considered a suitable project management framework to be employed for the management methodology of this project.

4.3 Potential Risks and Mitigation Plans

The following tables define potential risks and its respective mitigation plan.

Risk 01	Lack of Domain Knowledge				
Risk Level	High	**Phase**	All	**Frequency**	Medium
Mitigation	• Perform an in-depth review and analysis of currently available literature regarding domain and relevant topics				
Risk 02	Inability to meet set deadlines				
Risk Level	High	**Phase**	All	**Frequency**	High
Mitigation	• Meet with mentor and keep track of progress made on a regular basis. • Keep personal deadlines for each task by breaking it down further within main timeline				
Risk 03	Changes in Project Requirements				
Risk Level	High	**Phase**	All	**Frequency**	Medium
Mitigation	• Maintain priorities and make changes accordingly so as not to affect project timeline				
Risk 04	Technical limitations				
Risk Level	High	**Phase**	Implementation	**Frequency**	Medium
Mitigation	• Alter method of development to find work around said limitation				
Risk 04	Progress loss due to loss of data				
Risk Level	High	**Phase**	All	**Frequency**	Low
Mitigation	• Regular data back-ups to cloud storage • Maintain Github repository for code management • Have local backup of project if drastic change is being made in case roll-back is necessary.				

Table 4.3:1 Risk and mitigation plan

4.4 Work Breakdown Structure

Refer appendix

4.5 Gantt chart

Refer appendix

4.6 Compliance with BCS Code of Conduct

The BCS code of conduct is a professional set of standards to be followed to ensure ethical computing practices being followed in the United Kingdom. It is directed to all member of the British Computer Society. The principal duties of the BCS Code of conduct is as follows.

- The Public Interest
- Duty to the Profession
- Duty to Relevant Authority
- Professional Competence and Integrity

While implementing and conducting research on the project the author has followed the BCS code of conduct to ensure that Legal, Ethical and Social issues within the IT industry have been followed and carried out in a professional manner. The author also confirms that there have been no breaches of conduct of the BCS code of conduct during the implementation and research of this project.

4.7 Social, Legal, Ethical and Professional Aspects

<table>
<tr><th>Social</th><th>Ethical</th></tr>
<tr><td>The platform will be available to anyone who wishes to use the software under a Standard End User License Agreement (EULA)
Any individual willing to use this research for further development or use it as it is, will be allowed to do so.</td><td>Creating inaccurate result images -
In terms of an ethical standpoint, for data which might define county boundaries for example if produced inaccurately can cause a lot of issues if used for high influential statements.</td></tr>
<tr><th>Legal</th><th>Professional</th></tr>
<tr><td>Computer Miss use Act Violation
Data obtained for training of the system has been obtained through publicly available image training data sets with fair use.</td><td>BCS Code of conduct adhered to and not violated.
Having due regard for rights of Third Parties
Promote equal access to benefits to all sectors</td></tr>
</table>

Table 4.7:1 SLEP Analysis

4.8 Chapter Summary

In conclusion to this chapter, it has been identified the Project Management Methodology to be used in this project to be PRINCE2 after identifying the characteristics of the project and determining its suitability. The Potential risks that may occur through the lifespan of the project have also been identified and mitigation plans for each scenario has been discussed to avoid to factors that might lead to short coming in the project development. A work breakdown structure and Gantt chart for the time of the project have been illustrated. Further to conclude the compliance to the BCS code of conduct and Social, Legal Ethical and Professional aspect analysis of this project is discussed.

5. Chapter 5: System Requirements Specification

5.1 Chapter Overview

The goal of this chapter is obtain information on the requirements of the Raster to Vector Conversion system. Initially a stake holder analysis is carried out and then several requirement gathering techniques are used to identify functional and non-functional requirements that should be considered when implementing the system. Finally, user case descriptions are created to further strengthen the functional requirements identified and how they impact the operational flow of the system once implemented along with the chapter summary.

5.2 Stakeholder Analysis

The stakeholder analysis will be visually represented using an onion model diagram in this chapter and be further described using the roles and benefits table for each stakeholder.

5.2.1 Onion Model

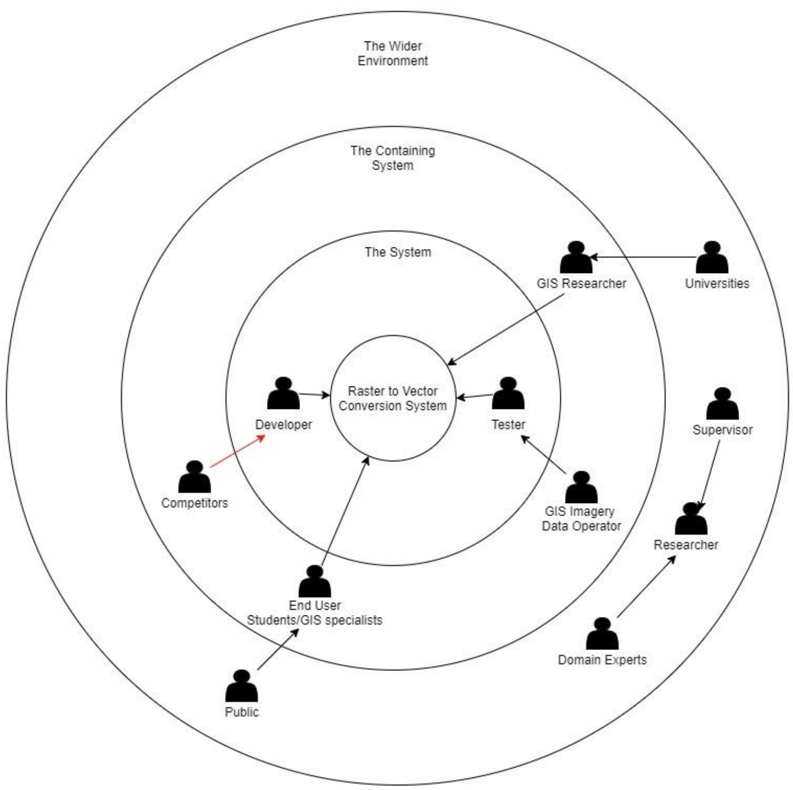


Figure 5.2:1 Onion model of stakeholder analysis

5.2.2 Stakeholders and Roles

The following table describes the stakeholders and their roles and benefits to the system.

Stakeholder	**Roles**	**Benefits**
Developer	Develop System	Develops the platform with less cost
Tester	Test System	Test and report the accuracy of the system
Domain Experts	Expert on the field of study (GIS)	Provide expertise on domain related matters to make system results accurate and to evaluate them
Universities	Functional Beneficiary	Is allowed to convert valuable geographical data from raster formats into vector formats and other educational uses
GIS Researches		Is allowed to convert valuable geographical data from raster formats into vector formats
End User		
Public		
GIS Imagery Data Operator	Gather Training and Testing Data	Make system more accurate if more data sets are found for training and testing
Supervisor	Assist in documentation and process of building system	Improve quality of system and its documentation by giving appropriate feedback
Researcher	Conduct Literature reviews and research to help implement system and identify requirements	Creates valuable documentation for ease of implementation of system
Competitor	Negative Stakeholder	Build similar solutions with better features

Table 5.2:1 Stakeholder roles and benifts

5.3 Requirement Elicitation Process

There are multiple techniques that can be used in order to validate and verify the requirements gathered. These can be states as Questionnaires, Observations, Literature Reviews etc. This section will briefly discuss the strengths and weaknesses of each such method and justify the method(s) of approach selected for this project.

<table>
<tr><td colspan="2">Method 1: Questionnaire</td></tr>
<tr><td colspan="2">Questionnaires are carried out to gain knowledge of the developers who have a similar experience in the industry. This form is sent out to the target audience to get their feedback regarding the research.</td></tr>
<tr><td>Advantages</td><td>Reach a large audience on different domains this platform is applicable on
As this research mainly focuses on a concept and an approach to raster to vector conversion, it is better to gather opinions of a large audience to prove the impact of it quantitatively</td></tr>
<tr><td>Disadvantages</td><td>Feedback could contain facts that are untrue</td></tr>
<tr><td colspan="2">Execution: A questionnaire is circulated among the target audience of the project which mostly comprises of developers working in the same field along with GIS Researchers who work on a daily basis with the cartographic and geographical image data that might require conversion between raster and vector. The questions were as follows.
To identify the percentage of users familiar with raster to vector conversion software.
To identify the level of the domain where these conversion methods are used.
To analyze if satisfactory result was obtained when using existing systems.
To understand the user reaction towards the system proposed.</td></tr>
</table>

Table 5.3:1 Questionnaire description

The information gathered from the questionnaire will be further discussed as this section continues.

Operationalization on Questionnaire

Objective	Question No.
• To identify the percentage of users familiar with raster to vector conversion software	**1**
• To identify the level of the domain where these conversion methods are used	**2, 3, 5, 6**
• To analyze if satisfactory result was obtained when using existing systems	**4, 7**
• To understand the user reaction towards the system proposed	**8, 9**

Table 5.3:2 Objectives of Questionnaire question

This questionnaire was sent out to individuals working in the IT and design industry who may or may not have experience with raster and vector graphics and platform that allows conversion between these two formats. This survey was helpful in determining if:

- Users were satisfied with the current industry options available for conversion.
- To gauge their reaction towards the proposed solution

Question 3

Aim: Determine purpose of using a raster to vector conversion software among users

Observation:

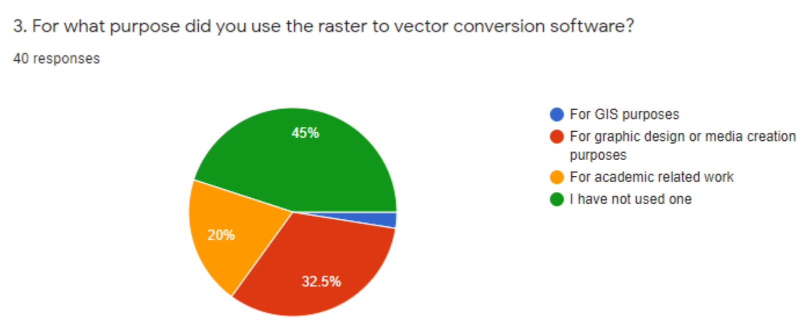


Figure 5.3:1 purpose of using a raster to vector conversion software among users

The purposes of this question was to identify for what purposes the target demographic as mentioned above of this questionnaire use Raster to Vector conversion platforms.

Conclusion:

It can be concluded that a majority (32.5%) of the users that use raster to vector conversion tools use it for graphic design and media creation purposes. It can also be observed that 20% of the users use these tools for academic purposes and Some users also use these platforms and tools for GIS Specific purposes.

Question 4

Aim: Identify if currently available solutions provide a satisfactory result

Observation:

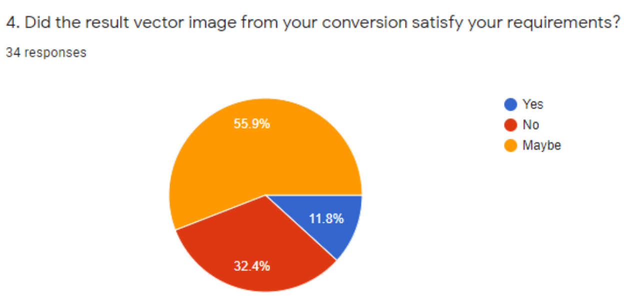


Figure 5.3:2 if currently available solutions provide a satisfactory result

The purpose of this question was to determine if the solutions currently available to users of raster to vector conversion software satisfy their needs of providing a satisfactory result of conversion between these two formats.

Conclusion:

A majority (55.9%) of the users were not sure if they were satisfied with their result. Almost a third (32.4%) of the users were clearly not satisfied with the result that the conversion software produced while only a 11.8% of the users were satisfied with their results. A conclusion can be made that there can be room for improvement in conversion methods and therefore is a good impact on the research being carried out.

Question 8

Aim: To get feedback on reaction to the research idea of this project.

Observation:

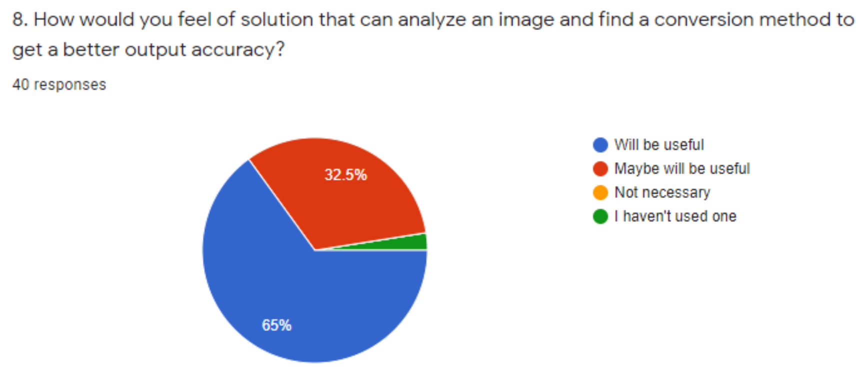


Figure 5.3:3 Feedback on reaction to the research idea of this project

The purpose of this question is to get the reaction of the users who provided feedback through this questionnaire to the solution proposed in this project.

Conclusion:

When observing the feedback obtained to this question it can be identified that majority of the feedback (65%) which is more than half the feedbacks obtained are in favour of such a software being developed and assume that it will be useful to them. 32.5% of the feedback also state that may find it useful while no feedback says that this solution will be not necessary contributing very well to the direction of this research and its progress.

Question 9

Aim: Understand the user reaction towards the system proposed

Observation:

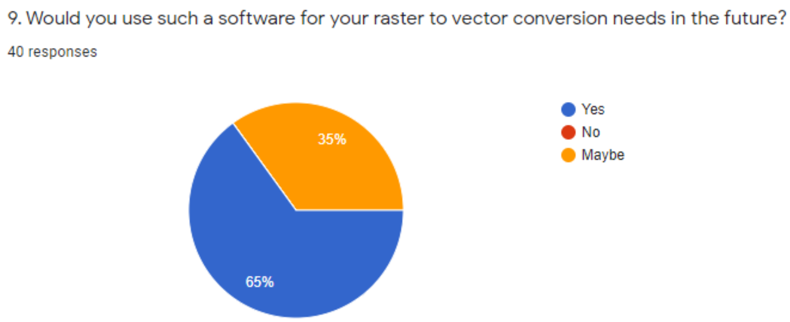


Figure 5.3:4 User reaction towards system

The purpose of this question is to understand if the users who provided feedback to this questionnaire would use the proposed solution in this research to perform their raster to vector conversion tasks, when necessary.

Conclusion:

When analyzing the observation of the results obtained through this questionnaire, 65% pf the feedback obtained are in favour of using such an implemented solution for their future raster to vector conversion needs and 35% of the feedback of the users are willing to try it as an option in their conversion necessities. Therefore, it is concluded that this greatly supports the research idea of this project and affects this project in a greatly favourable manner.

<table>
<tr><td colspan="2">Method 3: Literature Review</td></tr>
<tr><td colspan="2">A widely used method in the research community when it is necessary to gain knowledge on a domain or various techniques and technologies currently being used is by conducting a Literature review on existing material. Research repositories such as IEEE, Science Direct and Google Scholar can be used for this purpose. Hence by studying this material a research gap can be identified in current system to be addressed.</td></tr>
<tr><td>Advantages</td><td>Identify short comings of current solutions</td></tr>
<tr><td>Disadvantages</td><td>Observation varies according to observing individual</td></tr>
<tr><td colspan="2">Execution: A literature review has been conducted using the reference of research papers and documents found from the research libraries mentioned. The literature found has been categorized as Image Processing, GI systems and Raster to Vector Conversion related topics. This section is addressed in the second chapter of this thesis document.</td></tr>
</table>

5.4 Use Case Diagram and Description

5.4.1 Use Case Diagram

The following is a diagram that represents the use cases of the system visually.

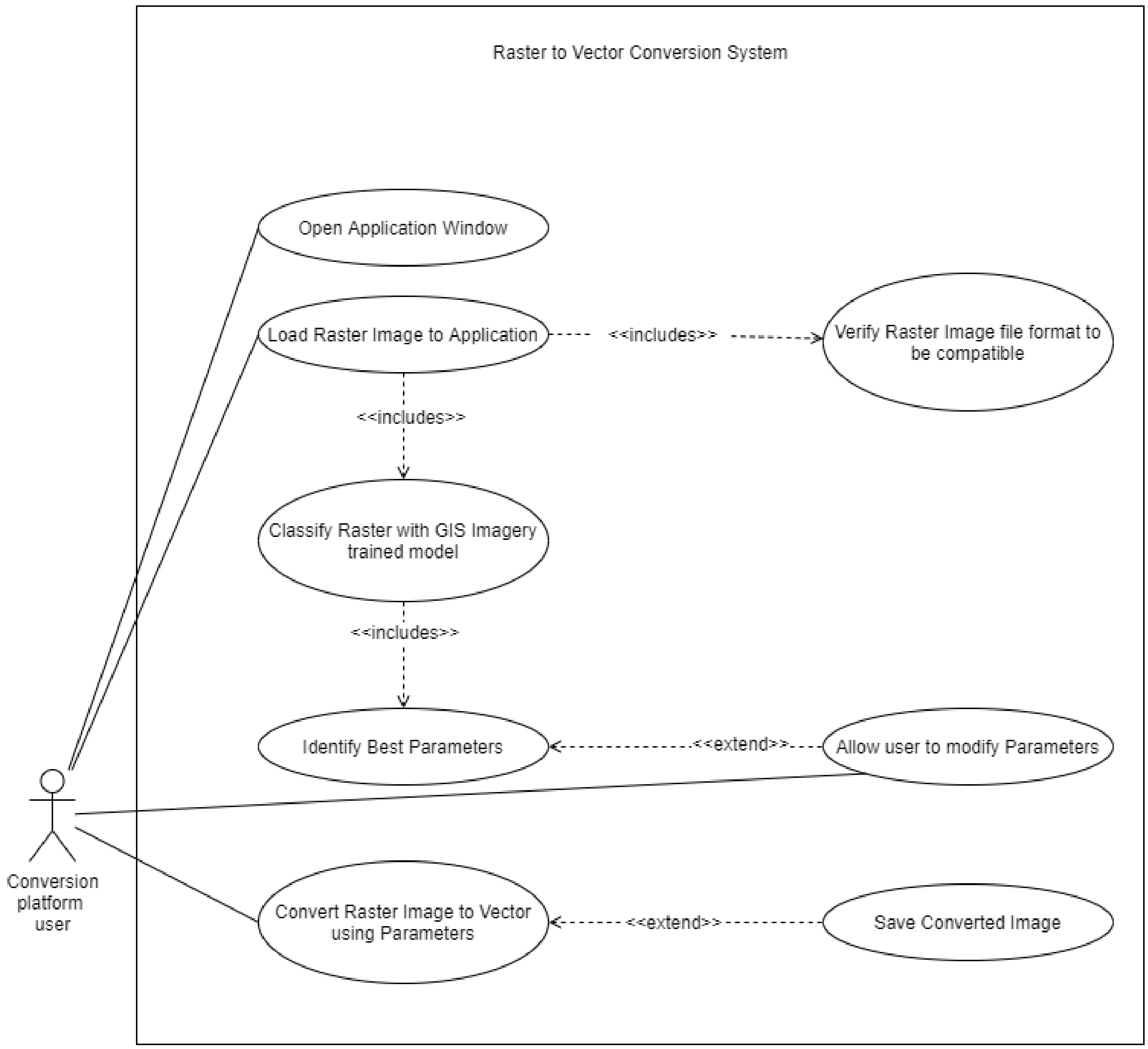


Figure 5.4:1 Use case diagram proposed

5.4.2 Use Case Description

The following table will further elaborate the use cases shown in the Figure 5.4:1.

Use Case ID	**UC-5**
Use Case Name	**Identify Best Parameters**
Priority	High
Actors	Conversion Platform User
Description	Use classification data to identify best fit conversion parameters for the image that was loaded
Pre-condition	Image classification must be performed on the loaded image
Extending Use Cases	Allow user to modify parameters
Including Use Cases	None
Triggering Event	System triggered after classification of image
Main Flow	1. Use parameter and classification data to identify best parameters for analyzed image 2. Return parameters for the conversion 3. Display parameters and allow editing to the user
Alternative Flow	none
Exceptional Flow	none

Table 5.4:1 Use case description for Use case 5

Use Case ID	**UC-6**
Use Case Name	**Convert Raster image to vector using parameters**
Priority	High
Actors	Conversion Platform User

Description	Convert Image using raster to vector conversion library using the parameters defined
Pre-condition	Conversion parameters for the particular image must be set
Extending Use Cases	none
Including Use Cases	Display converted image in new window
Triggering Event	Click on convert button on application
Main Flow	1. Initialize conversion package 2. Convert image using parameters set 3. Display success message for conversion
Alternative Flow	none
Exceptional Flow	3. If image conversion fails, due to not enough memory, catch exception and handle crashes that might occur

Table 5.4:2 Use case description for Use case 6

5.5 Functional and Non Functional Requirements

5.5.1 Functional Requirements

FR No.	Requirement	Inputs	Process	Outputs	Priority	Use case
1	Convert Image from Raster to vector	Raster Image	Convert Raster to vector from identified parameters	Vector Image	Critical	UC-6
2	Classify GIS image with trained model	Raster Image	Use image classification model trained to identify types of GIS imagery and classify image	Image classification score	Critical	UC-4

3	Identify conversion parameters	Classifi-cation score	Use classification score to find best fit conversion parameters	Best fit parameters	Critical	UC-5
4	Change auto selected parameters and redo conversion process		Convert Raster to vector from manual parameters		Critical	#
5	Save converted file		Save file to storage		Important	#

Table 5.5:1 Functional Requirements

5.5.2 Non-functional Requirements

#	Requirement title and description	Specification	Priority
1	Give user proper feedback on conversion process as it can be a long and time consuming process depending on the raster image and conversion parameters	Usability	Desirable
2	Result image should be visually similar and accurate to input raster image	Accuracy	Important
3	Develop API to allow conversions using online platforms	Usability	Desirable

Table 5.5:2 Non Functional Requirements

5.6 Chapter Summary

As chapter summary, first the stakeholders were identified and their roles were defined. After defining the roles, the requirement elicitation was carried out mainly with a questionnaire and a literature review. The outcomes of the questionnaire were discussed above with the statistics. Then the use case diagram of the system with the use case descriptions were discussed. After the use case diagram, the functional and non-functional requirements were identified

6. Chapter 6: Design

6.1 Chapter Overview

In this chapter, the class diagram, activity diagram and sequence diagram of the proposed solution will be elaborated and visually represented. Also an overview of all the diagrams will be elaborated at the end of this chapter.

6.2 Process Overview

The objective that is to be achieved in the research is to develop a software platform that can analyze an image, identify its properties and convert an image from raster to vector. Several separate components will have to be designed to achieve the final goal of this system, that is the conversion process. A process flow must initially be defined to design the system. This process flow will be discussed below and further elaborated in the diagrams found in this chapter. The three main components that are identified on a high level are the conversion component, the image classification component and the parameter trainer and selector component. The image classification component will be a sub system where an image is classified into types of GIS imagery types. A data set will be prepared and an image classification model using CNN (convolution neural networks) will be used to train a model that will be used in the classification process. The parameter trainer and identifier will be an application which converts images using the raster to vector conversion library that has been selected using each parameter combination possible and create a data set where the best parameter set and Structural similarity between the input and the output is calculated. Finally, this generated parameter and SSIM value data is used to convert the image after it is converted analyzed, and all functions of pre-processing deemed suitable are executed on the image. Figure represent a high-level diagram of the proposed system.

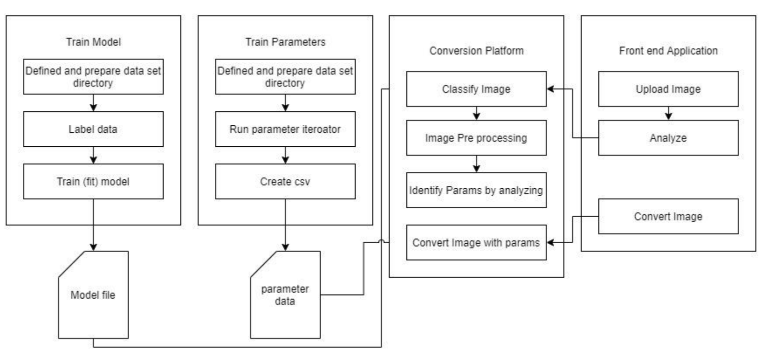


Figure 6.2:1 High level diagram of process

6.3 Class Diagram

Class diagram in figure 6.3:1 represents the attributes and methods of each class in the proposed raster to vector conversion platform. Each class's relationship with each other is also shown and also aggregation/composition relationships.

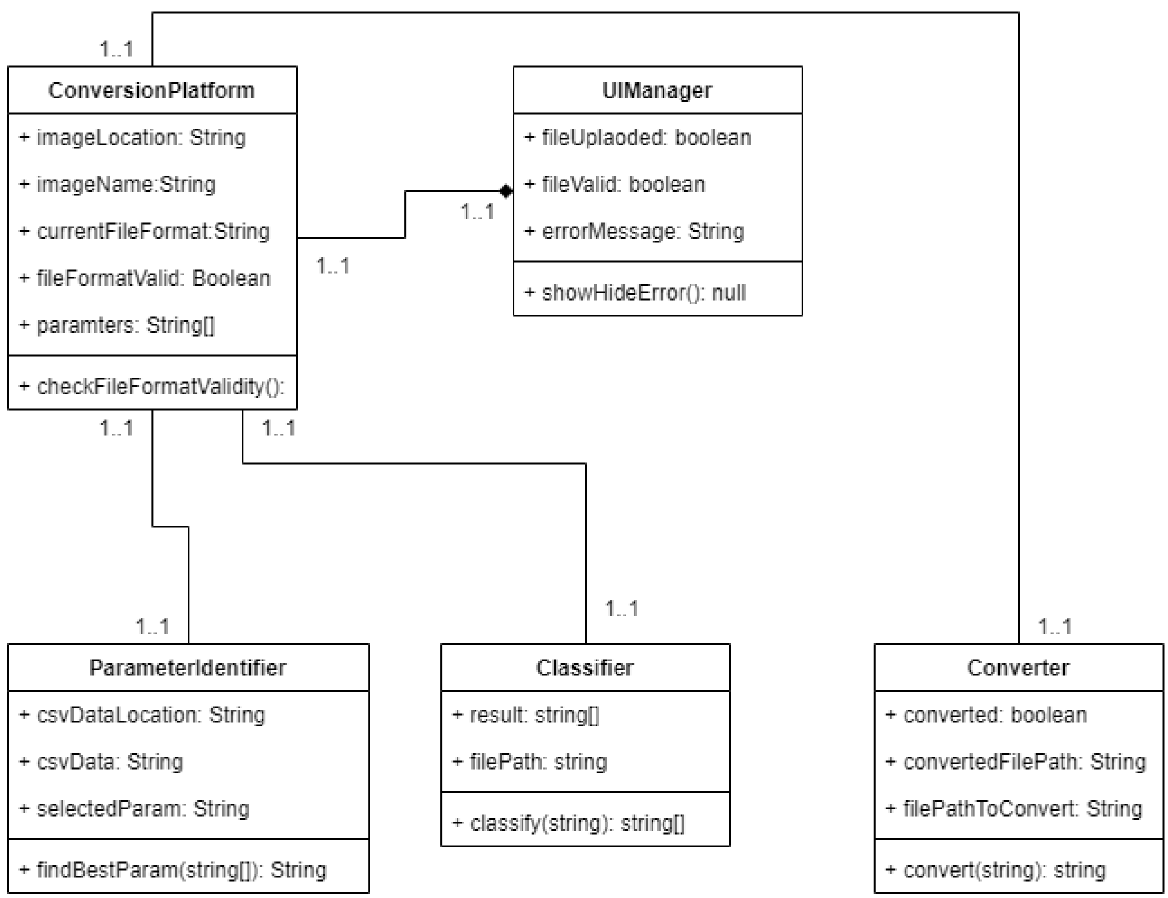


Figure 6.3:1 Class diagram proposed

6.4 Activity Diagram

The flow of activities in the Raster to Vector conversion platform is shown by the activity diagram shown below. This diagram is created according to the flows that were described in the use case diagrams and descriptions. This can be identified as the successful flow of activities for converting an image from raster to vector.

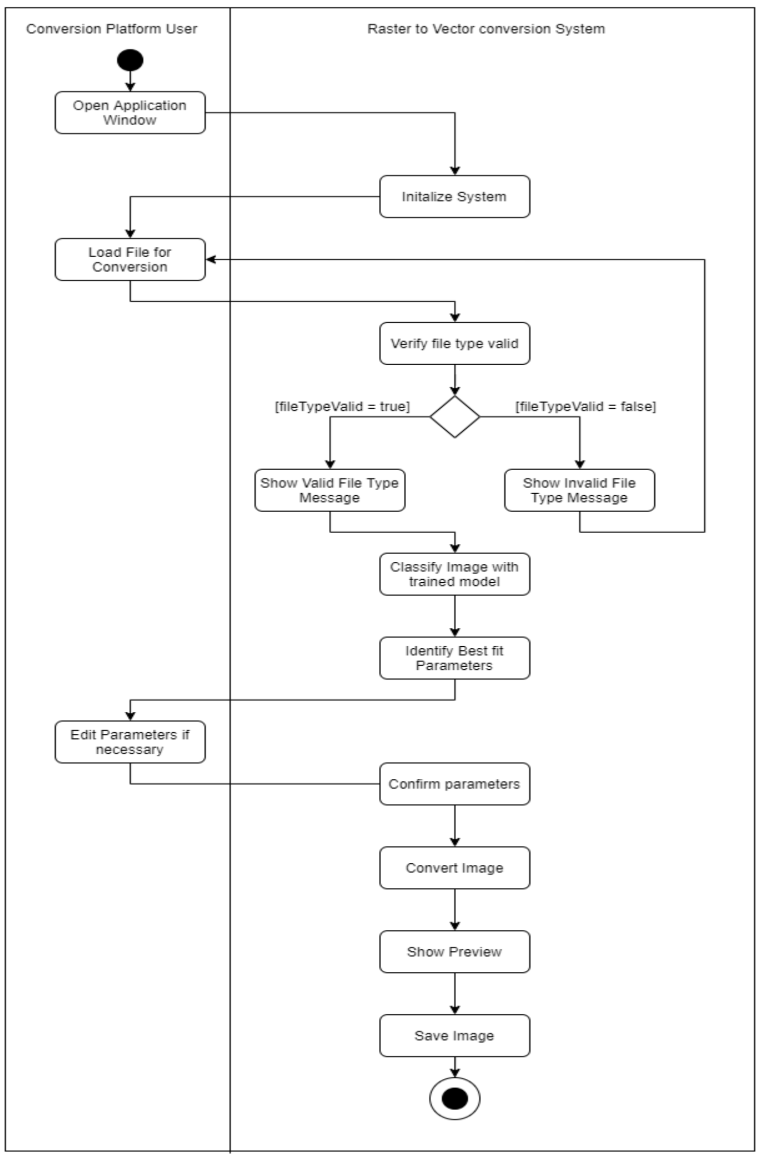


Figure 6.4:1 Activity Diagram

6.5 Sequence Diagram

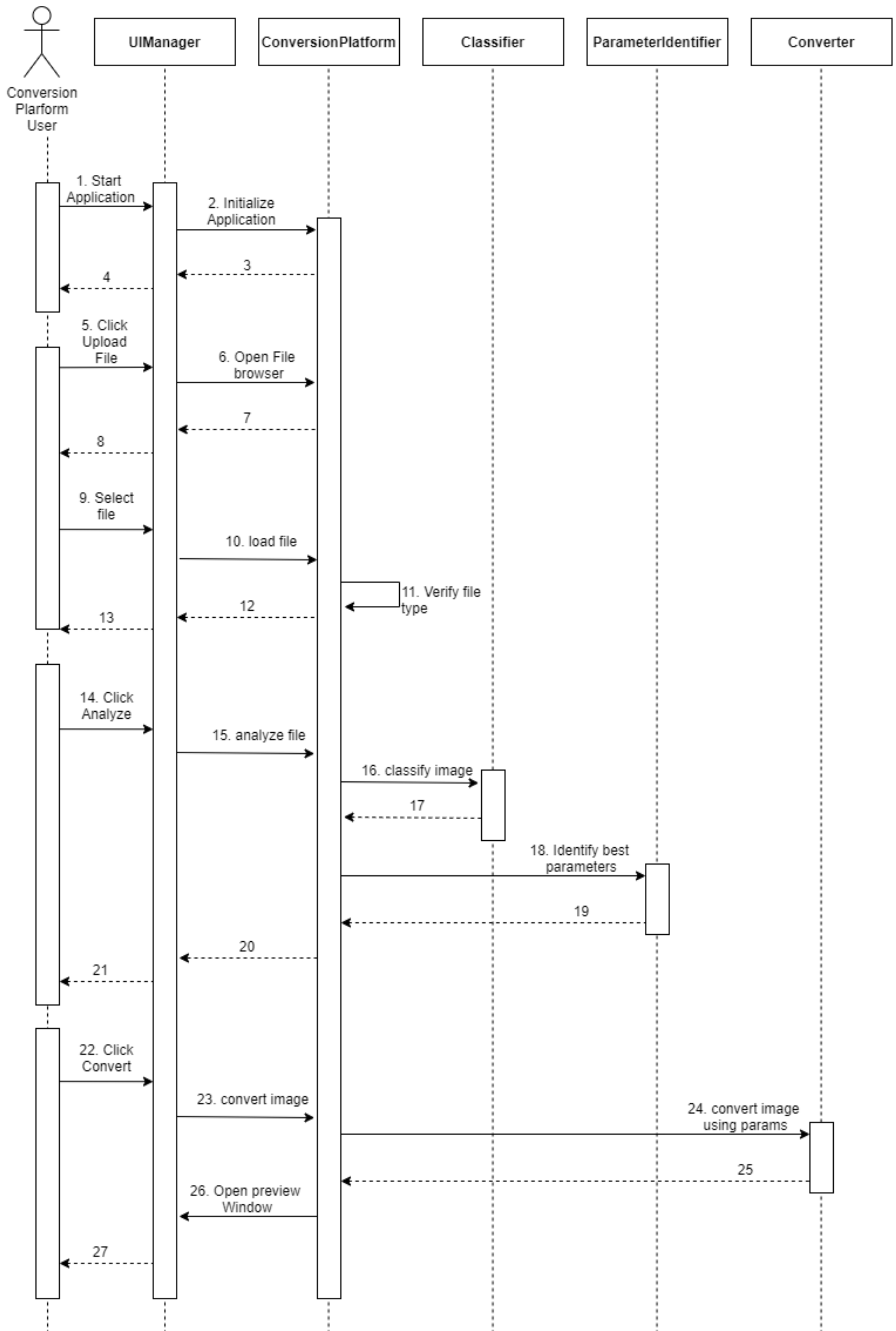


Figure 6.5:1 Sequence Diagram

The diagram above represents how the user interacts with the raster to vector conversion platform. The user starts by opening up the application, which then initializes the conversion platform. Then the user can locate an image file that they wish to convert. Once the file is loaded. It will be verified if it is of the valid file types accepted by the system. If this is the case. The user will be allowed to analyze the file. The conversion platform then uses the image classification trained model to classify the image and get its accuracy as to which classification it belongs to. Using this information then the best fit parameters are identified. Then the user is allowed to click the convert button which then converts the image into a temporary location. The user is also showed a preview of the converted image. This can be considered the sequence followed for a successful image conversion.

6.6 Chapter Summary

The Chapter depicted the design diagrams of the system. Initially the class diagram of the system along with the activity diagram for the main flow of activities in the system was described. A sequence diagram for the passing of messages between the components of the system and how they interact was also depicted.

7. Chapter 7: Implementation

7.1 Chapter Overview

This chapter elaborates on the technologies selected for the implementation of this project. This chapter also explains the reasons behind selecting them as the suitable technologies for this project. It also describes the libraries and tools used in this project. The chapter then depicts the High level architecture of the project according to the design specified in chapter 6 of this research document. Finally screenshots of the implemented system are given along with the chapter conclusion.

7.2 Basic Overview of the System

The basic component of the system can be divided into three main sub sections. These are the Main application which gets a raster image as an input, performs all acts of preprocessing on it and identifies the best fit conversion parameters for an output with high structural accuracy. The Image classifier that identifies what type of GIS image the main application is dealing with at the moment and the System that determines the best parameters for raster to vector conversion for a certain classification of images according to the structural accuracy comparison between input and output image.

The core functionality of all three of these components are implemented using the programming language Python. There are several libraries which are being employed to carry out specific tasks in each of these components and some are recurring libraries such as the python machine learning library that handles structural image comparison and the basic image processing libraries that are being employed for reading images and conversion of them into matrices for further operation to be carried out on them.

The User Interface of the Raster to Vector converter is implemented using Electron JS. A UI frame work that employs the power of HTML, CSS and JS to create cross platform desktop applications top of Node.js and Chromium. This will be further elaborated as this chapter continues. Parameters calculated are stored for future access for processing in csv format and are accessed and manipulated using the core language of the application which is Python. The Backend of this application is implemented using python Flask API and will be further elaborated and justified as to its use in the sections below.

7.3 Selection of Technologies

In the next sections the selection and justification for each of the technologies, libraries and tools used in this project will be discussed.

7.3.1 Selection of Programming Language

As most of the image processing, pre-processing libraries and machine learning libraries are libraries that are implemented on the language **Python (Python v3.6)**, it has been chosen as the language of implementation for the core features of the applications backend. There were also other reasons contributing to the selection of this language as the preferred language to be used in this project. Those are as follows.

- It is multi-paradigm programming language Therefore; it allows different programming approaches. Hence allowing it to follow the OOP design approach.
- Backed up with good documentation and an active and helpful community that is passionate about it.
- Support for wrappers to enable other languages to be integrated to the application. Allowing flexibility in coding certain features.
- Ability to extend functionality with libraries that support various machine learning functions.

7.3.2 Selection of UI Framework

The UI framework selected for implementing this project is Electron JS. Electron JS is a framework powered by Node.Js and Chormium that allows the implementation of cross platform desktop-suite applications using HTML, CSS and JS. The reasons for selecting electron JS as the preferred framework for implementing the UI component of this project are as follows.

- It is an efficient framework that takes care of tedious parts of developing a desktop application such as providing native menus and notification system, crash reporting system and tools for debugging etc.

- The author has prior experience and knowledge working on a similar framework. Therefore, allowing way adaptation to quickly implement and deploy the front end of the application.
- The framework allows for creative User interface to be created and to be developed in to a user friendly experience with ease.

7.3.3 Selection of Libraries

7.3.3.1 Scikit-learn (Image quantization)

Scikit-learn is a python library that provides unsupervised machine learning algorithms. It is a library that is built upon other common python libraries such as matplotlib, NumPy and Pandas. Out of the many algorithms that Scikit learn provides, this project will be utilizing the K-Means Algorithm which is a Clustering algorithm which will be used to quantize the colours found in an image to ease in raster to vector conversion and execute other functionalities which ease processing overheads and therefore, reduce processing time of the system.

7.3.3.2 Scikit-Image (Structural Similarity Index)

Scikit-Image is a python Image processing library that is built on other common python libraries such as numPy and sciPy. The methods that are created for comparing structural similarity and obtaining and index between two images of the same dimensions will be employed for the purpose of this project.

7.3.3.3 Tensorflow (Keras – Image classification CNN)

Tensorflow is an open source end to end machine learning platform. It supports multiple programming languages including python which is the core programming language used in this project. Its API is also available in multiple versions; this project will be using version r2.1 (stable) to implement the image classification training model that will be deployed using Tensorflow. Tensorflow allows multiple levels of abstraction. Due to this, the project will be utilizing the high level Keras API provided by Tensorflow. The Keras API will be utilized to implement a simple Image processing model using the layers provided to create a Convoluted Neural Network (CNN) to classify GIS related image according to the training data sets obtained.

7.3.3.4 OpenCV

Open Computer Vision Library (OpenCV) is another Image processing library that supports the programming language python among other programming languages that is being used for image processing in this project.

7.3.3.5 Python Flask API Framework

For the Application Programming Interface that is used to expose the backend functionality to the frontend desktop application, FLASK a popular python based API will be used. The advantages of using Flask as the selected API for the project are as follows.

When compared to popular web frameworks such as Django, Rails, Play and Laravel etc. Flask can be considered a significantly minimal web framework. This allow the programmer to implement flask applications in any design structure. In this case OOP based. As OOAD is selected as the preferred Software Design methodology as specified in the third chapter of this document. Flask has been chosen the Framework to be used to implement the backend exposing API in this project.

7.3.4 Image pre-processing methods

7.3.4.1 Smoothening Images

Smoothening images can also be known as blurring of images. This can be used to de-noise Images if they contain a lot of noise that might affect the quality of the raster to vector conversion process resulting in a final output with unnecessary polygons that were a result of noise on the image. As all images will not have noise in them. This should be selectively applied to images if only necessary.

The basic process of smoothing operations is done by applying a filter to the image. The most commonly used filter is the linear filter in which the output pixel value (i.e. $g(i,j)$) is a calculated as a weighted sum of input pixel values (i.e. $f(i+k,j+l)$) and $h(k,l)$ which is called the kernel, which is the coefficients of the filter.

$$g(i,j) = \sum_{k,l} f(i+k, j+l)h(k,l)$$

The commonly used filter types are mentioned below.

Normalized Box Filter.

The output pixels are calculated to be the mean of its kernel neighbors.

$$K = \frac{1}{K_{width} \cdot K_{height}} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \\ . & . & . & \dots & 1 \\ . & . & . & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \end{bmatrix}$$

Gaussian Filter

Convolving of each point of the input images array with a Gaussian kernel and adding them to produce the output array as a sum. A 2D Gausian can be represented as follows

$$G_0(x, y) = Ae^{\frac{-(x-\mu_x)^2}{2\sigma_x^2} + \frac{-(y-\mu_y)^2}{2\sigma_y^2}}$$

μ	Mean (the peak)
σ	Standard Deviation
x y	The values of the two axis

Median Filter

Replaces each pixel in the input images pixel array with the median values of its neighbouring pixels. The neighbouring pixels are of a square neighbourhood around the pixel (Corke, 2011). Median filter will be used as the image smoothening method used for this implementation as it provides low intensity edge removal that can be beneficial for de-noising images in this system.

7.3.4.2 **Image Colour Quantization using K-means technique**

Initially a fixed number of clusters and initial cluster centers in the colour space are identified and chosen. The reason for this is to change the position of cluster centers so long as the sum of distances between all points of clusters and their cluster centers will be minimal.

During these modifications all points are allocated to closest cluster centers using a predefined metric (Palus, 2004). Typically used metrics are,

- Euclidean distance
- City Block metric.

After each allocation a new positions of cluster centers are computed as arithmetical means of cluster points. The algorithm usually stops if the difference between new and old positions of cluster centers is too small. K-means converges to a locally optimal solution (Likas, Vlassis and J. Verbeek, 2003).

$$E(m_1, \ldots, m_M) = \sum_{i=1}^{N} \sum_{k=1}^{M} I(x_i \in C_k) \|x_i - m_k\|^2,$$

Therefore, in this project, K-means clustering has been identified as the preferred colour quantization method to be used.

7.3.5 Selection of Tools

7.3.5.1 Selection of IDEs

IntelliJ Pycharm is a phenomenal IDE developed by Jetbrains s.r.o. This is a very popular IDE used for Python Software Development. PyCharm also enables the project to be created in its own Virtual environment for which python Conda will be used.

- Existing experience of the author.
- Better intellisense for Python Development provided by the
- PEP8 checks for better code quality and alerts when violated with fix suggestions

For Frontend Development Visual Studio Code will be used, as it supports code suggestion and formatting support for the mainly used three languages used in the front end framework which are HTML, JS, CSS.

7.4 Implementation of Components

7.4.1 Parameter Trainer Component

This component handles the pre training of the system to determine the best conversion parameters for each classification of GIS image determined by the image classification model. It contains the methods for various image pre-processing functions that are used throughout the system as well.

7.4.1.1 **Image Colour Quantization**

```
# quantize image to reduce colour data for easy raster to vector conversion
def quantize_image(image: object, number_of_colors: int):
    (h, w) = image.shape[:2]
    image = cv2.cvtColor(image, cv2.COLOR_BGR2LAB)
    image = image.reshape((image.shape[0] * image.shape[1], 3))
    clt = MiniBatchKMeans(n_clusters=int(number_of_colors))
    labels = clt.fit_predict(image)
    quantized_image = clt.cluster_centers_.astype("uint8")[labels]
    quantized_image = quantized_image.reshape((h, w, 3))
    quantized_image = cv2.cvtColor(quantized_image, cv2.COLOR_LAB2BGR)
    # return resized image
    return quantized_image
```

Figure 7.4:1 Image Quantization code

Image quantization is also an important pre-processing functionality that is being employed to reduce the computational power required and time of running of the system when dealing with high quality images with high number of colours that also might make the conversion software run out of memory due there being really high polygon count when raster to vector conversion is to be done. Colour quantization in method is done by using the clustering algorithm found in k-means and passing the number of clusters to be used when quantization of the image. The image should be converted to the LAB colour format before being used to in the clustering algorithm the image after quantization is converted back into the BGR format the default colour space configuration for Open CV. The quantized image is then returned.

7.4.1.2 **Image Structural Similarity Index Calculator**

```
def check_similarity(path_1: str, path_2: str, image_size):
    img1 = cv2.imread(path_1)
    img2 = image_resize(cv2.imread(path_2), image_size)

    w1, h1 = img2.shape[:-1]
    img1 = cv2.resize(img1, (h1, w1))

    s = ssim(img1, img2, multichannel=True)
    return str(s * 100)
```

Figure 7.4:2 SSIM calculator

Structural similarity index measure (SSIM) calculates the mean squared error (MSE). Structural similarity takes texture of an image to account when comparing the original and to be compared to image. Two images are passed in as parameters where the later image is resized to be the same size as the original image, then using Scikit-images SSIM algorithm the similarity between these two images are calculated and returned as a percentage.

7.4.1.3 Variable Trainer Process

```python
def variable_full_process(category):
    image_size = 720
    end_file_type = ".svg"

    # Initialize directories for category
    try:
        os.mkdir("images/png/" + category)
        os.mkdir("images/tempimages/" + category)
        os.mkdir("images/converted/" + category)
        os.mkdir("csv/" + category)
    except OSError as e:
        print(e)

    # Load necessary paths for the reading and writing
    data_path = "images/data/" + category
    temp_path = "images/tempimages/" + category
    out_svg_base_path = "images/converted/" + category
    png_base_path = "images/png/" + category

    csv_path = 'csv/' + category

    for img in os.listdir(data_path):

        image = cv2.imread(os.path.join(data_path, img))
        image = image_resize(image, image_size)
        image = quantize_image(image, 16)

        image_name = img.split(".")[0]
        image_extension = img.split(".")[1]

        try:
            os.mkdir(os.path.join(temp_path, image_name))
            os.mkdir(os.path.join(out_svg_base_path, image_name))
            os.mkdir(os.path.join(png_base_path, image_name))
        except OSError as e:
            print(e)

        out_svg_specific_path = os.path.join(out_svg_base_path, image_name)

        sample_path = os.path.join(temp_path, image_name, "sample." + image_extension)
        save_image_to_path(sample_path, image)

        index = 0
```

Figure 7.4:3 Variable Trainer Process Part 1

```python

        with open(csv_path + "/" +image_name + '.csv', 'w', newline='') as f:
            thewriter = csv.writer(f)

            thewriter.writerow(["index", "ltres", "qtres", "pathomit", "file_path", "similarity"])

            for ltres in range(0, 9, 2):
                for qtres in range(0, 9, 2):
                    for pathomit in range(0, 101):
                        if pathomit == 1 or pathomit == 10 or pathomit == 100:
                            print("progress:" + str(int(index / 75 * 100)) + "%")

                            index = index + 1

                            subprocess.call([
                                'java', '-jar', 'libraries/imageTrace.jar', sample_path,
                                'outfilename', out_svg_specific_path + "/" + str(index) + end_file_type,
                                'pathomit', str(1 / pathomit),
                                'ltres', str(ltres),
                                'qtres', str(qtres),
                                'colorsampling', str(0),
                                'colorquantcycles', str(16)
                            ])

                            png_out_path = "images/png/" + category + "/" + image_name + "/" + str(index) + ".png"

                            convert_to_png("../../../" + out_svg_specific_path + "/" + str(index) + end_file_type,
                                           "../../../" + png_out_path)

                            similarity_val = check_similarity(sample_path, png_out_path, image_size)

                            thewriter.writerow(
                                [str(index), str(ltres), str(qtres), str(pathomit), str(index) + end_file_type,
                                 str(similarity_val)])

    print("completed!")
```

Figure 7.4:4 Variable Trainer Process Part 2

The full process of identifying best fit parameters of each classification of image as classified by the trained image classification model starts by creating and initializing the directories required by the function to store and read images to and from. Once all directories are initialized and the paths to the data set directories are all validated A for loop loops through all the data for the necessary category (images). The image is then resized and quantized using the methods described above. The processed image is then stored in a temporary location which gets overwritten every time the for loop moves into a new iteration. A csv writer then is opened and a csv of the name of the image is created to store the parameters, and similarity of each conversion. Then the code enters a series of nested for loops that iterate each through a single parameters possible values. The temporary image is then converted using these parameters each time. Converted to PNG using the method described above and stored and then the converted PNG and the temp image is compared to identify similarity between them using the method described above. The values are then stored in the csv file and after all the parameter values are iterated through the initial for loops moves on to the next image. After all images in the directory have been iterated through the function is ended.

7.4.1.4 Create Best Fit Parameters for category

```python
def read_csv(category: str):
    base_file_path = 'csv'

    with open('final_csv/' + category + '.csv', 'w', newline='') as f:
        thewriter = csv.writer(f)

        thewriter.writerow(["index", "ltres", "qtres", "pathomit", "file_path", "similarity"])

        for csv_path in os.listdir(os.path.join(base_file_path, category)):
            df = pd.read_csv(os.path.join(base_file_path, category, csv_path), engine='python')
            df = df.sort_values('similarity', ascending=False)
            df.to_csv(os.path.join(base_file_path, category, "converted_" + csv_path), index=False)

        for csv_path in os.listdir(os.path.join(base_file_path, category)):
            if (csv_path.startswith("converted_")):
                input_file = csv.DictReader(open(os.path.join(base_file_path, category, csv_path)))

                # print highest 10 accracy values
                index = 0

                for row in input_file:
                    if index < 1:
                        thewriter.writerow(
                            [str(row['index']), str(row['ltres']), str(row['qtres']), str(row['pathomit']),
                             str(row['index']) + row['file_path'],
                             str(row['similarity'])])

                    index += 1
```

Figure 7.4:5 Create best fit parameter code

The parameter and ssim csv files are read by a for loop where each is arranged in descending order of their ssim values and the highest ssim value of each parameter file is taken and noted down in a csv file. This csv file then contains the best parameters that were used to get the highest ssim value in all of the images that were used as the data set. In the previous method where the parameter iteration and ssim is carried out. This csv is then saved.

7.4.2 Image Classifier Component

This component handles the classification of various types of GIS data into three main categories. Satellite Imagery, Land classification Images and Scanned maps. This component has three main subcomponents. One that prepares the data and creates the training data set, the Model training component and the component that creates the neural network model with its layers and other parameters.

7.4.2.1 Create and Prepare Training Data

```
def create_training_data(categories, data_dir, img_size, training_data):
    for category in categories:
        print("Image Loading from " + category + "...")
        path = os.path.join(data_dir, category)
        class_num = categories.index(category)
        for img in os.listdir(path):
            try:
                img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
                resized_array = cv2.resize(img_array, (img_size, img_size))
                training_data.append([resized_array, class_num])
            except Exception as e:
                print("Image Failed to Load")
                pass
    random.shuffle(training_data)
```

Figure 7.4:6 Prepare training data code

The images from a directory are loaded onto the application and stored in an numpy image array. The images here are loaded in grayscale (2 channels) and are indexed with the category, which is their folder name in which they reside in the data directory. The data is then shuffled to get better test result when training the system as it gets trained to each category randomly without being trained one after the other. This can help with the performance of the trained models accuracy.

7.4.2.2 Training Image Classification Model

```
def train_save_model(training_data, img_size):
    print("starting to train data...")

    X = []  # images
    y = []  # labels

    for features, label in training_data:
        X.append(features)
        y.append(label)

    X = np.array(X).reshape(-1, img_size, img_size, 1)
    y = np.array(y)

    X = tf.keras.utils.normalize(X, axis=1)

    model = Sequential()

    model.add(Flatten(input_shape=X.shape[1:]))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(3, activation='softmax'))

    model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    model.fit(X, y, batch_size=32, epochs=20, validation_split=0.1)
    print(model.summary())
    return model
```

Figure 7.4:7 Model fitting code

Once the prepared training data and image sized that is being used is passed as arguments into this method. It creates two arrays, one with the features (the image) X and the other with their corresponding label y. The X array of images is converted to a numpy array and reshaped using the image size passed as an argument. The values in X are then normalized to obtain better accuracy when training. The model is then created using the **layers** library that is provided by Keras by Tensorflow. In this model, a flatten layers then two dense layers with 64 and 3 nodes respectively are added sequentially the activation function for the first dense layer is **relu** which is a linear activation function that ramps up for any value that is not negative. The second dense layer has a **softmax** activation function and as there are three categories of classification it has 3 nodes at the end layer. This model is then compiled using the *optimized* **adam**, *loss function* **sparse categorical crossentropy,** and to obtain metrics of accuracy. The model is then fitted with the data set. Which trains it to classify images between those three categories. And finally returns this model.

7.4.3 Backend API

This component exposes the backend functionalities to the front end application through an application programming interface. This API uses the python Flash API framework.

7.4.3.1 Flask API method signatures

```
converter = Converter()
analyzer = Analyzer()


@app.route('/convert')
def perform_conversion():
    return converter.convert()


@app.route('/analyze')
def perform_classification():
    return analyzer.analyze()


if __name__ == '__main__':
    app.run()

```

Figure 7.4:8 Flask API implementation code

These two methods are the methods of the API that are exposed through the routes defined and can be called through an http request from the front end. The methods which are being executed here will be further explained below.

7.4.3.2 Analyzer Class

```
class Analyzer:
    def prepare_image(path: str):
        img_size = 250
        img_array = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
```

Figure 7.4:9 Defining analyzer class

This is a python class that contains the method necessary for the image to be analysed and also for various pre-processing functions to be carried out before analysis, and finally identify the best fit parameters for the input image uploaded by the user. Methods that are used for quantization, resize and reshaping will not be discussed here as it has been discussed precious and the same concept is implemented in this class as well. Instead the unique functions of the analyzer class will be described.

7.4.3.3 Classify Image (Analyzer Class)

```
    def classify(path: str):
        my_model = load_model('classification_model/gis_classify.h5')
        prediction = my_model.predict(Analyzer.prepare_image(path))

        # ["LandClassificationMaps", "Satellite", "Scanned"]

        values = prediction[0]
        type = "unidentified"
        category= "unidentified"

        if values[0] == 1:
            type = "Land Classification Maps"
            category = "landclass"
        elif values[1] == 1:
            type = "Satellite Imagery"
            category = "sat"
        elif values[2] == 1:
            type = "Scanned Map"
            category = "scanned"

        return [type, category]
```

Figure 7.4:10 Classify image method

This method uses the h5 file of the model during the classification model training process and classifies an image that is passed into it. And return the type of the image and the category which is assigned by nested if statements that reads the prediction result and decides the classification of the image.

7.4.3.4 Get Best Parameters (Analyzer Class)

```python
def get_best_param_range(self, category):
    path = 'final_csv/'+category+'.csv'

    ltres_low = Analyzer.find_lowest(path, "ltres")
    ltres_high = Analyzer.find_highest(path, "ltres")

    qtres_low = Analyzer.find_lowest(path, "qtres")
    qtres_high = Analyzer.find_highest(path, "qtres")

    pathomit_low = Analyzer.find_lowest(path, "pathomit")
    pathomit_high = Analyzer.find_highest(path, "pathomit")

    return [('ltres_low', ltres_low), ('ltres_high', ltres_high), ('qtres_low', qtres_low),
            ('qtres_high', qtres_high),
            ('pathomit_low', pathomit_low), ('pathomit_high', pathomit_high)]

def find_lowest(path, prop_val):
    input_file = csv.DictReader(open(path))
    lowest = 0
    index = 0

    for row in input_file:
        if index == 0:
            lowest = row[prop_val]
        else:
            if row[prop_val] < lowest:
                lowest = row[prop_val]

        index += 1

    return lowest
```

Figure 7.4:11 Get best parameter code Part 1

```python
def find_highest(path, prop_val):
    input_file = csv.DictReader(open(path))
    highest = 0
    index = 0

    for row in input_file:
        if index == 0:
            highest = row[prop_val]
        else:
            if row[prop_val] > highest:
                highest = row[prop_val]

        index += 1

    return highest
```

Figure 7.4:12 Get best parameter code Part 2

This method is responsible for determining the best fit parameters range for the category of the image that has been uploaded by the user. This category is a parameter passed into the method and is the classification category determined when the image is classified using the model. The method then reads the best parameter csv regarding that category and determines the highest and lowest values for each of the parameters and determines a range for the best fit parameters to exist in and returns this as an array of tuples.

7.4.3.5 Determining Best parameters for user image (Analyzer Class)

```python
def get_params(image_path, category):
    try:
        os.rmdir('temp/png')
    except OSError as e:
        print("does not exist")

    try:
        os.rmdir('temp/svg')
    except OSError as e:
        print("does not exist")

    try:
        os.mkdir('temp/svg')
    except OSError as e:
        print("does not exist")

    try:
        os.mkdir('temp/png')
    except OSError as e:
        print("does not exist")

    end_file_type = ".svg"
    param_range = Analyzer().get_best_param_range(category)
    image_size = 720

    image = cv2.imread(image_path)
    image = Analyzer.image_resize(image, image_size)
    image = Analyzer.quantize_image(image, 16)
    image_extension = imghdr.what(image_path)
    temp_image_path = 'temp/sample.' + image_extension
    Analyzer.save_image_to_path(temp_image_path, image)

    index = 0
```

Figure 7.4:13 Determining Best parameters code Part 1

```python

    with open('temp/temp.csv', 'w', newline='') as f:
        thewriter = csv.writer(f)

        thewriter.writerow(["index", "ltres", "qtres", "pathomit", "file_path", "similarity"])

        for ltres in range(int(param_range[0][1]), int(param_range[1][1]) + 1, 1):
            for qtres in range(int(param_range[2][1]), int(param_range[3][1]) + 1, 1):
                for pathomit in range(int(param_range[4][1]), int(param_range[5][1])):
                    if pathomit == 1 or pathomit == 10 or pathomit == 100:
                        index = index + 1

                        svg_out_path = "temp/svg/" + str(index) + end_file_type

                        subprocess.call([
                            'java', '-jar', 'imageTrace.jar', temp_image_path,
                            'outfilename', svg_out_path,
                            'pathomit', str(1 / pathomit),
                            'ltres', str(ltres),
                            'qtres', str(qtres),
                            'colorsampling', str(0),
                            'colorquantcycles', str(16)
                        ])

                        png_out_path = "temp/png/" + str(index) + ".png"

                        Analyzer.convert_to_png("../../../" + svg_out_path, "../../../" + png_out_path)
```

Figure 7.4:14 Determining Best parameters code Part 2

```python
                    similarity_val = Analyzer.check_similarity(temp_image_path, png_out_path, image_size)

                    thewriter.writerow(
                        [str(index), str(ltres), str(qtres), str(pathomit), str(index) + end_file_type,
                         str(similarity_val)])

    df = pd.read_csv('temp/temp.csv', engine='python')
    df = df.sort_values('similarity', ascending=False)
    df.to_csv(os.path.join('temp/temp_sorted.csv'), index=False)

    input_file = csv.DictReader(open('temp/temp_sorted.csv'))

    # get highest accuracy values
    index = 0

    highest_row = []

    for row in input_file:
        if index < 1:
            highest_row = row

        index += 1

    return highest_row
```

Figure 7.4:15 Determining Best parameters code Part 3

This method is the main function of the Analyzer class which determines the best fit parameters for the image selected by the user. This method too gets the category as a parameter for its operations. It selects the csv file of best parameters for its category. And then after obtaining the best parameter range from the method previous defined it runs conversion on them similar to the parameter trainer component and finally returns the set of parameters with highest Structural similarity index.

7.4.3.6 **Analyze (Analyzer Class)**

```python
def analyze(self):
    file_path = request.args.get('file_path')

    classification = Analyzer.classify(file_path)
    color_count = Analyzer.count_colours(file_path)
    quant_val = Analyzer.identify_quntize_cluster(color_count)

    row = Analyzer.get_params(file_path, 'landclass')# category[1]

    return jsonify(file_path=file_path, classification=str(classification[0]), color_count=color_count
                   , quant_val=quant_val, row=row)
```

Figure 7.4:16 Analyze image method

This method combines all the methods defined above and creates the final JSON that will be passed to the front end with all of the parameters and some additional data that will be shown to the user before conversion.

7.4.3.7 **Converter Class**

```python
class Converter:
    def convert(self):
        file_path = request.args.get('file_path')
        file_name = request.args.get('file_name')
        qtres = request.args.get('qtres')
        ltres = request.args.get('ltres')
        pathomit = request.args.get('pathomit')
        k = request.args.get('k')
        quant_bool = request.args.get('quantBool')
        end_file_type = ".svg"

        if quant_bool:
            Analyzer.quantize_image(cv2.imread(file_path))

        subprocess.call([
            'java', '-jar', 'imageTrace.jar', file_path,
            'outfilename', file_name + end_file_type,
            'colorsampling', str(0),
            'colorquantcycles', str(k),
            'ltres', str(ltres),
            'qtres', str(qtres),
            'pathomit', str(pathomit)
        ])

        return jsonify(progress="completed", file_name=file_name, file_path=file_path, out_path=file_name + end_file_type)

```

Figure 7.4:17 Defining converter class

This component is the class that handles the conversion of the image from all the parameters that have been identified using the analyser phase of the applications flow. It obtains the parameters from the URL params of the request and converts the image using the imagetrace library and output the image to the specified location. Once the operation is complete returns a JSON with the success flag to update the UI.

7.4.4 Frontend

7.4.4.1 Initialization

```javascript
// Modules to control application life and create native brow
const {app, BrowserWindow} = require('electron')
const path = require('path')

function createWindow () {
  // Create the browser window.
  const mainWindow = new BrowserWindow({
    width: 1000,
    height: 600,
    frame: false,
    webPreferences: {
      preload: path.join(__dirname, 'preload.js'),
      nodeIntegration: true
    }
  })

  // and load the index.html of the app.
  mainWindow.loadFile('index.html')

  // Open the DevTools.
  // mainWindow.webContents.openDevTools()
}

// This method will be called when Electron has finished
// initialization and is ready to create browser windows.
// Some APIs can only be used after this event occurs.
app.whenReady().then(createWindow)
```

Figure 7.4:18 Electron initlization

Electron JS application initialization configuration. The configuration defined here will be used when creating the window for the frontend application which will be initialized.

```javascript
function analyze_file() {
    $.ajax({
        url: "http://127.0.0.1:5000/analyze?file_path=" + document.getElementById("convert-file").files[0].path,
        dataType: "json",
        success: function (data) {
            console.log(data);

            $('.step-1').hide();
            $('.step-2').show();

            $(".on-hover").css("pointer-events", "auto");

            preview_analysis_data(data);
        },
        error: function (data) {
            $('#analyze-error').show()

            $('#before-analyzing').show()
            $('#after-analyzing').hide()

            $(".on-hover").css("pointer-events", "auto");
            $('#analyze-btn').attr("disabled", false);
            $('#analyze-progress').hide();
        }

    })
}

function preview_analysis_data(data){
    document.getElementById("prediction-val").innerText = data.classification;
    document.getElementById("color-val").innerText = data.color_count;
    document.getElementById("quant-val").innerText = data.quant_val;

    document.getElementById("quant-range").value = data.quant_val;

    document.getElementById("ltres-val").value = data.row.ltres;
    document.getElementById("qtres-val").value = data.row.qtres;
    document.getElementById("pathomit-val").value = data.row.pathomit;
}
```

Figure 7.4:19 Analyze file ajax method

This function sends a request to the backend Flask API and gets the parameters for the file defined in the file path URL parameter. And after a successful response updates the UI with these values preparing the application for the conversion process.

```
$("#converter").click(function () {
    $.ajax({
        url: "http://127.0.0.1:5000/convert?" +
        "file_path="+ document.getElementById("convert-file").files[0].path+
        "ltres="+ document.getElementById("ltres-val").innerText+
        "qtres="+ document.getElementById("qtres-val").innerText+
        "pathomit="+ document.getElementById("pathomit-val").innerText+
        "k="+ document.getElementById("quant-range").value+
        "quantBool="+ true,
        dataType: "json",
        success: function (data) {
            console.log(data)
        }
    })
})
```

Figure 7.4:20 Convert file ajax method

And after converts the image from raster to vector using the parameters found. When triggered by the user.

7.5 Screenshots of the System

The following screenshots depict the UI developed for the raster to vector conversion application. This is custom UI created using HTML,JS and CSS on the Frontend framework Electron JS

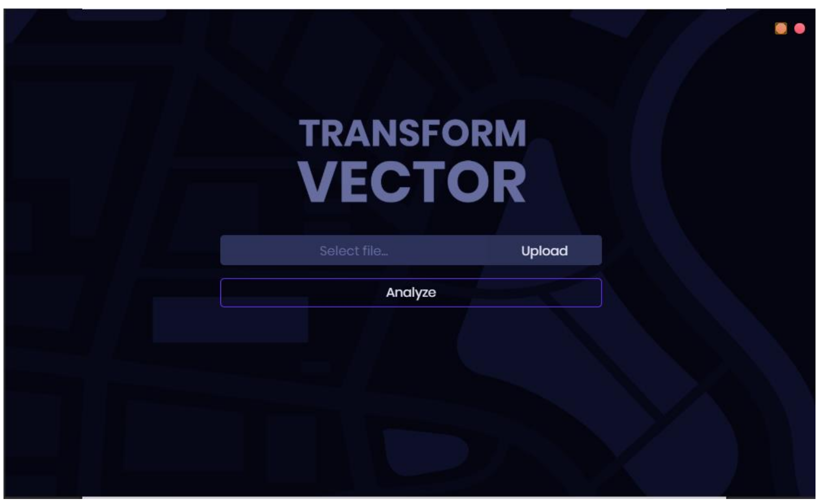


Figure 7.5:1 Screenshot 1

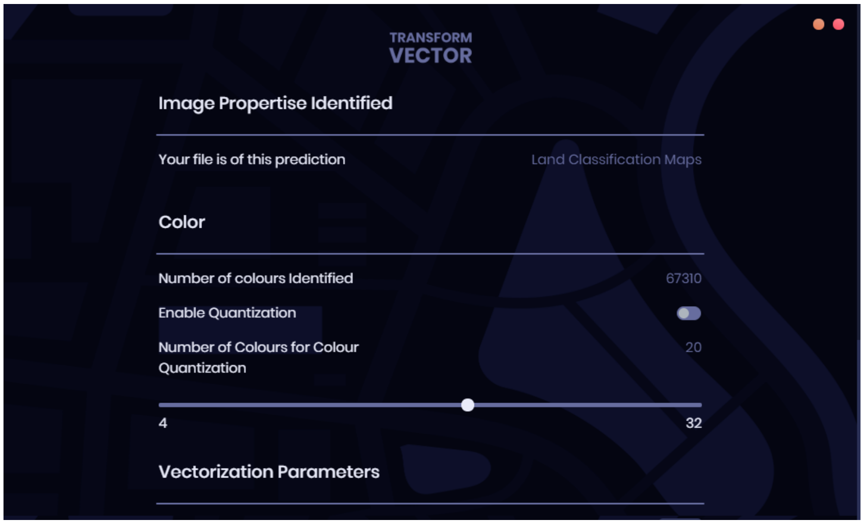


Figure 7.5:2 Screenshot 2

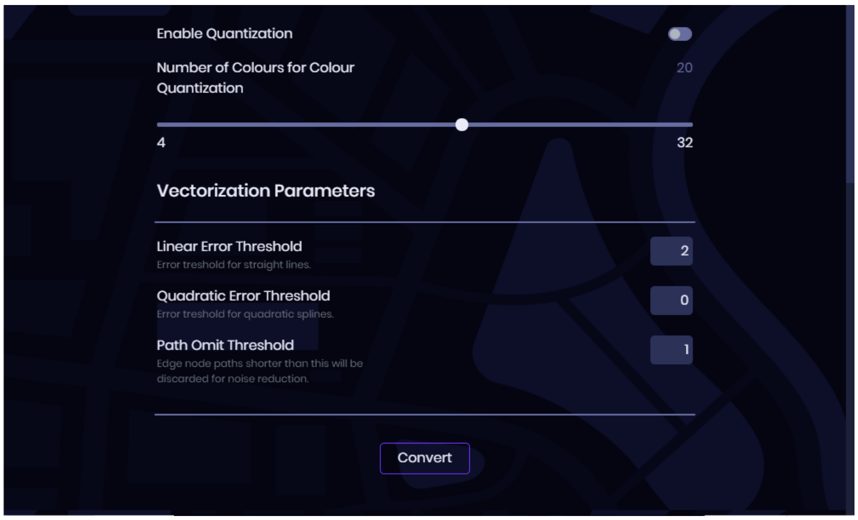


Figure 7.5:3 Screenshot 3

7.6 Chapter Summary

Initially the selection of technologies, programming language and tools required for the implementation of the project were discussed. A basic overview of the system was also described. Each component and the functionality that they implement in the system with screen shots of the code was then provided. Finally, to conclude the chapter screen shots of the systems UI and console outputs were depicted.

8. Chapter 8: Testing

8.1 Chapter Overview

This chapter discusses the testing process overview of the system and its results. The testing is carried out to test if functional and non-functional requirements are met and achieved by the implemented system. Initially, the testing goals and objectives are touched upon. The testing criteria is then mentioned. Black box texting is carried out on the main system and its results will be depicted. The image classification model will also be tested to get its error rate and accuracy. After functional testing is completely discussed. Non-functional testing according to the non-functional requirements mentioned in chapter 5 will also be elaborated on. The limitations of the testing will also be discussed as conclusion to the chapter.

8.2 Goals and Objectives of Testing

This testing phase is carried out to ensure that the system implemented as described in Chapter 7 of this document meet the functional and non-functional requirements that were descried during the requirement specification stage of this project. This testing phase makes sure the following goals are achieved.

- Identifying bugs and fixing them.
- Validate and verify if all the functional requirements have been met by the system.
- Validate and verify if all the non-functional requirements have been met by the system.

8.3 Testing Criteria

As the testing criteria of this testing phase, the following two criteria which are defined below will be used to identify and reduce the gap between the expectation and reality of the implementation.

Functional Testing – To test the functional requirements defined

Non-Functional Testing – To test non-functional requirements of the system. But elevate the overall experience of using the system.

8.4 Unit Testing

The system implemented in this research, comprises of many unit components that through integration complete the overall process of the system. As these unit functionalities have to perform and generate the expected outputs for the full system to work. The validity of each component may affect the next component that it is to interact with. If this is not validated and rectified where anomalies occur, a waterfall affect may occur and affect the accuracy and performance of the system as a whole. Unit testing is thus used to test the individual units of the system to validate that the component being tested performs as expected.

8.4.1 Unit Testing Tools

PyTest has been selected as the unit testing tool to be used in the testing phase of this projects' unit tests. It is a simple testing framework that is open source and is well documented. It allows for compact test scenarios. **PyTest** allows storing of values from test cases, the ones which were asserted successfully and the ones which weren't. Therefore, it is much easier than going through logs and using debuggers to identify test result.

The Following table describes the unit test cases and their results that were carried out during the unit testing phase. A more detailed version of this table can be found in the Appendix G of this document.

ID	Test case	FR ID	Priority	Result
UTC01	Load data set for variable and SSIM comparison csv creation	3	High	Passed
UTC02	Quantize loaded image	1	High	Passed
UTC03	Write parameters and SSIM to csv per each image	3	High	Passed
UTC04	Sub process call to Convert to Vector	3	High	Passed
UTC05	Bat file execution for SVG to PNG conversion	3	High	Passed
UTC06	Obtain Similarity Index for input image and output image comparison	3	High	Passed

UTC07	Read csv and order in ascending order of given column	3	High	Passed
UTC08	Find lower range of column in csv	3	High	Passed
UTC09	Find Higher range of column in csv	3	High	Passed
UTC10	Find best fit parameters from given csv	3	High	Passed
UTC11	Classify image using image classification model file which was fitted and saved.	2	High	Passed

Table 8.4:1 Unit test case results

8.5 Integration Testing

Integration testing is the testing of the unit modules after being logically integrated. This system comprises of multiple unit modules that in turn must communicate with each other to function as a complete system and produce the expected output. In this section integration test cases and their results will be depicted.

Test case	**Component Name**	**Actual Result**	**Expected Result**	**Test Result**
Train Parameters for Image category when data set provided	Python Parameter Trainer Module	(3*5*5) = 75 rows produced with ssim values in csv for each parameter iteration	75 rows created in csv for each parameter iteration with ssim value	Pass
Load Image for Analyzing	UI (Frontend)	Load image when selected from file browser, and reject all unsupported file formats	Load image file types only when selected from file browser	Pass

Analyze Image and return parameter JSON	Flask API	Return best row parameter obtained by reading csv	Return best row parameter	Pass
Send parameters and options defined by user to API through ajax call	UI (Frontend)	Send parameters through *url params* that were updated in the UI	Send parameters through *url params* to the backend	Pass
Convert Image with parameters obtained through request parameters	Flask API	Run conversion library and convert image	Run conversion library through subprocess and convert image	Pass

Table 8.5:1 Integration testing results

8.6 Testing Image Classification Model Accuracy

The following are the testing result and training result for the image classification model that was created to classify between the identified three types of common GIS imagery. The model was trained on 1418 samples and validated on 158 samples. 20 epoch were run to train the model and below are the results for the trainin accruacy and loss and validation accuracy and loss in each epoch during the training process.

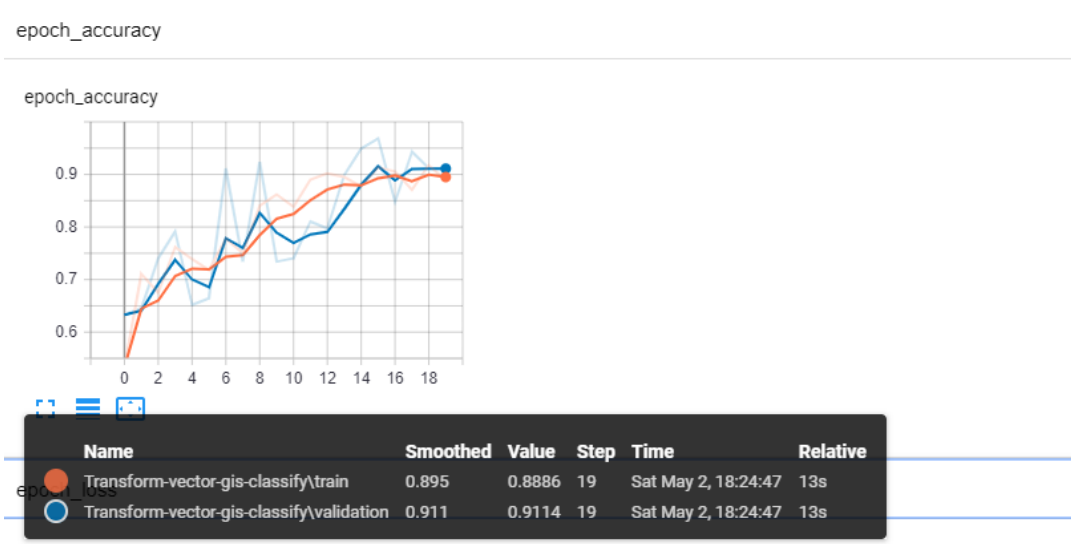


Figure 8.6:1 Accuracy of each epoch

Figure 8.6:2 Loss of each epoch

The above statistics have been logged using Tensor board. A tool for providing the measuring and visualizing machine learning workflows. It can be seen that the loss at the end of the training of the 20 epochs was 0.2669 and loss of validation was 0.2378, the accuracy for training was 0.8886 and validation was 0.9114 (all normalized between 1 and 0) And as this has high accuracy and low loss it can be considered this model has a good classification strength.

8.7 Testing Image analyzing process Performance of Flask API Backend

This testing method analyzes the time taken for an image to be analyzed according to its image size. The data is collected over several iteration of the process by checking duration taken to analyze images of various image sizes. The images were prepared for this task using an image editing tool. *Note: the images used were chosen at random*

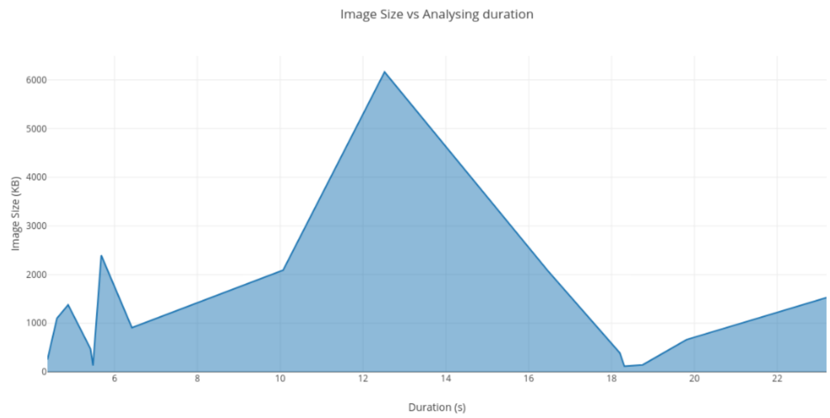


Figure 8.7:1 Image size vs analysis duration

Image 1: Land classification map

File size	Dimensions	Duration (s)
2080 kb	3720 x 2480	16.46
2092 kb	2500 x 1667	10.07
909 kb	1280 x 854	6.42
474 kb	300 x 534	5.42
129 kb	360 x 240	5.48

Table 8.7:1 Land classification map result for analysis duration testing

Image 2: Satellite Imagery

File size	Dimensions	Duration (s)
1378 kb	1024 x 1024	4.88
6165 kb	2500 x 2500	12.52
2395 kb	1280 x 1280	5.68
1012 kb	800 x 800	4.61
253 kb	360 x 360	4.38

Table 8.7:2 Satellite Image result for analysis duration testing

Image 1: Scanned Maps

File size	Dimensions	Duration (s)
116 kb	354 x 340	18.31
1527 kb	2500 x 2119	23.19
663 kb	1280 x 1085	19.81
389 kb	800 x 868	18.2
144 kb	360 x 305	18.75

Table 8.7:3 Scanned maps result for analysis duration testing

When going through the results obtained by analyzing the test image set, it can be observed that the image size and dimension plays a large role in the duration of analysis the longest duration of 23.19 s was recorded analyzing the scanned maps image of 1527 kb size. When Comparing the durations of analysis between classes of images, scanned maps clearly do take longer to analyze. The reason for this being that the suitable parameter range for scanned maps are much larger than the parameter ranges available in the other 2 categories therefor all parameters must be fitted to, to get the best parameters to convert the image.

Therefore, the conclusion can be made that, while image size and dimensions of an image matter when the analysis is being performed on it for its process duration. The suitable parameter range for that particular image class can also affect the image analysis time.

8.8 Testing Image Conversion Performance of Flask API Backend

This testing method analyzes the time taken for an image to be converted from raster to vector with the parameters auto selected by the platform according to its image size. The data is collected over several iteration of the process by checking duration taken to convert images of various image dimensions. The images were prepared for this task using an image editing tool.

Note: the images used were chosen at random

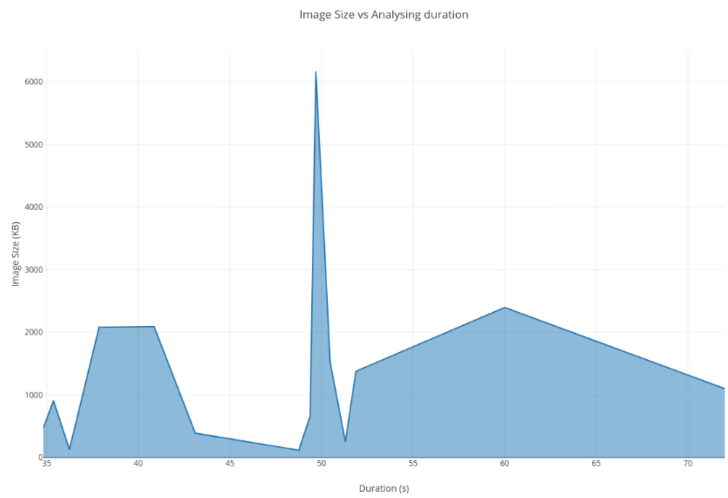


Figure 8.8:1 Image size vs conversion duration

Image 1: Land classification map

File size	Dimensions	Duration
2080 kb	3720 x 2480	37.86
2092 kb	2500 x 1667	40.86
909 kb	1280 x 854	35.37
474 kb	300 x 534	34.82
129 kb	360 x 240	36.24

Table 8.8:1 Land classification results for conversion duration

Image 2: Satellite Imagery

File size	Dimensions	Duration
1378 kb	1024 x 1024	51.88
6165 kb	2500 x 2500	49.69
2395 kb	1280 x 1280	60
1012 kb	800 x 800	72
253 kb	360 x 360	51.3

Table 8.8:2 Satellite image results for conversion duration

Image 1: Scanned Maps

File size	Dimensions	Duration
116 kb	354 x 340	48.79
1527 kb	2500 x 2119	50.47
663 kb	1280 x 1085	49.38
389 kb	800 x 868	43.11
144 kb	360 x 305	48.79

Table 8.8:3 Scanned map results for conversion duration

As it can be observed in the test results noted down above for each image and the time taken for it to be converted using the auto generated parameters of the system. It can be identified that roughly around 40-50 second averages can be seen in the duration of conversion of image of any dimension. This shows that the dimensions and file size of the image does not affect the conversion duration. There are several anomalies where there is60 and 72 seconds recorded as the time taken to convert. Therefore, further testing under different conditions will have to be carried out to understand why these occur.

8.9 Testing structural similarity between input image and converted vector image.

This phase will carry out a test to determine the structural similarity index between the original image that was uploaded by the user for conversion and the vectorized image after conversion. The image will be converted through the use of the platform using the auto selected best conversion parameters and no parameters will be changed. The testing will be done with de noising and without de-noising separately. An enlarged image of this result can be seen in Appendix – H.

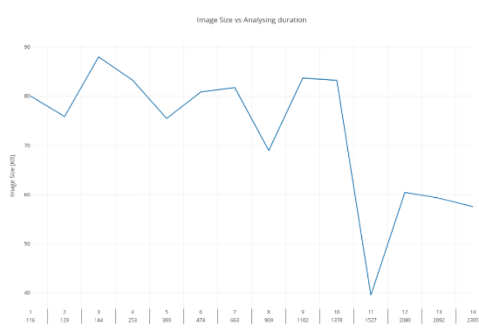


Table 8.9:1 Structural similarity between input image and converted vector image

When observing the results of the graph drawn against SSIM (similarity index) and the file size and index of conversion using the platform it can be seen that an average of 68% to 88% is maintained in the conversion process between the input image and the output. There were several anomalies that occurred during this process. This was due upscaling of the original image which was of very low quality to a higher resolution for testing purposes with the image editing tool. These upscaled images did not produce good results, while its original low resolution image produced 83% Similarity. Therefore, it can be said that the anomalies can occur in rare cases where an image is upscaled and converted when the initial image size is very low. But a better conversion can be obtained when just converting the original image resolution with its details.

8.10 Chapter Summary

This chapter initially defines the testing and goals and objectives that are to be determined through the test phase of this project. The unit test cases and their results then have been documented and further elaborated as well. Unique testing for the system has been carried out with custom test scenarios created by the author to test the performance of conversion and analyzing in this system. Finally a the Structural similarity index has been recorded over several images of different classification and file types to identify a reading on the average accuracy of the output produced when compared to the original image that was analyzed and converted by the system.

9. Chapter 9: Evaluation

9.1 Chapter Overview

The evaluation of the system that was described in the design chapter and implemented as described in the implementation chapter will be carried out in this chapter. Initially, the evaluation criteria are identified as well as the evaluation goals of this project. Feedback and evaluation from various experts in their respective fields are obtained and matched against the evaluation criteria. The evaluation methodology is touched upon and final the author's self-evaluation and reflection will be discussed.

9.2 Evaluation Methodology and Approach

To gain expert and user feedback on the Transform vector prototype developed and also about the contents of this research, several experts were contacted through emails, personal interviews could not be carried out due to a revealing state of lock down currently in the country. A questionnaire was sent out to the evaluators to obtain feedback through. The data received by the author has been analysed thematically. The evaluators feedback and a reflection by the author will be provided as this chapter proceeds.

9.3 Evaluation Criteria

There are two main methods that can be used to evaluate the solution developed. They are Qualitative and Quantitative evaluation. Quantitative analysis can be used for a solution that mainly focus on statistical and numerical research. While Quantitative analysis can be used to evaluate solutions that mainly focus on concepts, approaches etc. As this project primarily focuses on an approach of implementation a **Quantitative evaluation** has been deemed suitable.

9.4 Evaluation Goals

The main features of the system implemented will be evaluated according to the evaluation goals described. The table 9.3:1 outlines the evaluation goals identified for evaluation of the system. It is also mentioned the reason for selecting the mentioned goal thus helping to verify whether the solution developed can be acceptable as a valid POC (proof of concept) of the project.

<table>
<tr><td colspan="2">Evaluation of Concept and Scope of the Project</td></tr>
<tr><td>Goals</td><td>Reason for Evaluation</td></tr>
<tr><td>Need for a better raster to vector conversion approach for GI systems
Understand scope and depth of project</td><td>To get an insight into the concept and the overall scope covered by this project.</td></tr>
<tr><td colspan="2">Evaluation of Technicality of the Project</td></tr>
<tr><td>Goals</td><td>Reason for Evaluation</td></tr>
<tr><td>Evaluate architecture and proposed design of the system.
Evaluate use of tools and technologies used
Evaluate coding practices followed</td><td>As the end product of this research is a software solution, it must be evaluated if it meets the necessary standards of the industry and the technology stack used in the evaluation is accepted by the industry.</td></tr>
<tr><td colspan="2">Evaluation of the Usefulness and Impact to the domain of this Project</td></tr>
<tr><td>Goals</td><td>Reason for Evaluation</td></tr>
<tr><td>Evaluate raster to vector conversion platform features</td><td>Identify if the solution provided by this system impacts the domain which it address significantly.</td></tr>
<tr><td colspan="2">Limitation and Future Enhancements of the Project</td></tr>
<tr><td>Goals</td><td>Reason for Evaluation</td></tr>
<tr><td>Evaluate current limitations of the system developed
Reflect on future enhancements possible</td><td>Identify limitation of the system currently and future enhancements that can enhance the system further to provide a bigger impact to the domain.</td></tr>
</table>

Table 9.4:1 Evaluation Goals

9.5 Selection of Evaluators

As a qualitative evaluation approach has been taken in this project, to improve the evaluation quality, individuals from the following domains and fields have been requested to provide evaluation on the system.

Evaluator	**Description of Domain or field**	**Technical Proficiency**
Conversion Platform Users	End users of the conversion platform can be anyone using the conversion platform to convert an image from raster into vector	Not Technical
GIS experts	GI system users to convert their rasters and maps to vectors to be used in GI services and applications. They are also a type of end user of this system	Not Technical
Software Engineers and Architects	Individuals who are currently experienced members of the Software industry who can provide great feedback on the implementation architecture and evaluation in a development point of view	Technical
Graphic Design Experts	A type of end user who can use the system again for raster to vector conversion but with insight into intricacies of raster to vector conversion	Not Technical

Table 9.5:1 Selection criteria for evaluators

9.6 Evaluation Feedback

Still being compiled and waiting for some feedback.

9.7 Self-Evaluation

9.7.1 Evaluation of Concept and Scope of the Project

The hypothesis that was stated at the initial stages of the project to improve the quality of outputs of the raster to vector conversion process was achievable within the given project duration. The project was also streamlined towards GIS specifically, this was considered and evaluated in a suitable manner with the time frame. All objectives that were presented as within scope of the raster to conversion platform was also achieved during the project duration. And can be said that the project concept and scope were addressed in a suitable manner within the given time frame.

9.7.2 Evaluation of Technicality of the Project

This system is more of a frame work than a final fixed software product. And can be extensible to many more types of categories of conversion such as the GIS type of images that was trained during this project. This framework can be used with any raster to vector conversion library that supports parameters to be manually set by the user to obtain better conversions of the input images to vector with little modification and re-training. The whole framework is separated into three main components and run independently of each other. This allows more flexibility in coding each component and changes can be made without breaking the system easily. The API can also be deployed on a server and requests to convert and analyse images can be done using a client machine.

9.7.3 Evaluation of the Usefulness and Impact to the domain of this Project

All trainer modules, only requires a data set to train a particular category of images and to find the best fit range for them. The generated csv file with the best parameter range can be used for conversion. These parameters are then used in the main application backend to analyse the image and find the best parameters for that image. Therefore, it can be seen that this platform is extensible to any amount of classifications of images as long as the data set can be prepared for it. Therefore, this platform can be used not only for GIS image conversion but is extensible to many other categories of domain where necessary simply by training the models with the relevant data sets. It can also be seen that it converts the currently trained GIS images with better similarity to the original and thus achieves its purpose to be useful and impactful to its main target domain.

9.7.4 Limitation and Future Enhancements of the Project

Limitations and futures enhancements will be discussed in the conclusion chapter.

9.8 Benchmarking

In the benchmarking process the similarity between the same image converted using the Transform Vector platform and the unchanged library default conversion was compared to see if there were improvements made by the best fit parameter system that was implemented in this project.

Image index	Image Name	SSIM using default	SSIM using Transform vector platform
1	1280x857.png	58.33376	80.05545
2	2500x1667.png	57.93248	75.88013
3	360x240.png	55.1128	88.00611
4	800x534.png	56.56386	83.2759
5	original.png	0	75.48427
6	1280.png	50.23074	80.85716
7	2500.png	51.18976	81.78421
8	360.png	52.27106	68.96471
9	800.png	49.69352	83.71661
10	original.png	53.00599	83.25775
11	1280.jpg	39.54803	39.55269
12	2500.jpg	39.84814	60.46204
13	360.jpg	39.712	59.27562
14	800.jpg	39.0727	57.55102

Table 9.8:1 Benchmarking results

It can be observed that some images have 0 SSIM values, these images were images that were failed to be converted using the raw library as the application ran out of memory. However, using the Transform vector conversion platforms image preprocessing methods these images were able to successfully converted without any issues.

It can also be observed here that there is a significant increase in the Structural similarity index score between the vanilla conversion library and the Transform vector conversion process. 14 images have been used to test this of different image sizes and resolutions which have been picked on random. According to the above comparison it can be confirmed that a substantial increase in the output images similarity to its raster counterpart can be obtained by using the Transform Vector platform.

9.9 Reflection on Functional Requirements

The following table describes the functional requirements that were stated in the System Requirements Specification chapter of the document and if they have been implemented within the system.

FR No.	Requirement	Status	Priority
1	Convert Image from Raster to vector	Implemented	Critical
2	Classify GIS image with trained model	Implemented	Critical
3	Identify conversion parameters	Implemented	Critical
4	Change auto selected parameters and do conversion process	Implemented	Critical
5	Save converted file	Implemented	Important

Table 9.9:19.8 Reflection on Functional Requirements

The author is highly satisfied with the progress that has been achieved through this and as all the project deliverables have been met. The possible enhancements that can be done to this project for its further improvement are discussed in the next chapter of this document.

9.10 Chapter Summary

This chapter initially determines the evaluation criteria and goals set by the author of this project. Then the process of selection of evaluators are broken down and the evaluation of the selected evaluators are mentioned. The authors self-evaluation of the project is also discussed and benchmarking is carried out. Finally, the reflection on the functional requirements and its status is done.

10. Chapter 10: Conclusion

10.1 Chapter Overview

This chapter concludes the main section of the body of this document by discussing the achievement and thoughts of the author regarding the research carried out. The chapter initially discusses about the achievement of the aims and objectives, the functional and non-functional requirements. The milestones and deliverables completed during this project are also discussed. Problems faced while conducting this research and the current limitations will then be discussed. Next, the future enhancement will be stated, and finally the contribution and the concluding remarks will be elaborated on.

10.2 Achievement of Aim and Objectives

10.2.1 Achievement of the Project Aim

"To investigate design and implement a Raster to Vector conversion platform that selects the best method of conversion using image processing techniques."

The raster to vector conversion platform created as a solution in this research project was capable of selecting best fit conversion parameters to a given image using image classification and various image pre-processing techniques. Hence, it can be concluded that the aim of this project has been Satisfactorily achieved.

10.2.2 Achievement of Research Objectives

To conduct a thorough literature review on existing solutions and platforms

A through literature review has been conducted by going through the contents of over 30 publications related to GIS imagery, Raster and vector image processing, Raster to vector image conversions. Additionally, documentations of the libraries and packages to be used has also been conducted to gain more information when selecting the final technologies for implementation.

To design an image processing model to identify classifications of a Raster image related to GIS

An image classification model that uses a Convoluted Neural Network to classify image data into the classifications defined as Satellite imagery, land classification maps and Scanned maps (GIS Imagery) was implemented during this project.

<u>To implement functionality to determine the most favourable parameters for the conversion through the parameters identified from the image processing process</u>

A parameter trainer model that uses various image processing techniques have been developed through the course of this project. The parameter trainer method that uses Structural similarity index to identify best fit parameters and then compile files for each category with best parameters for the data set from which a best parameter range can be obtained. This is then used to convert the image.

<u>To evaluate the converted Vector image in terms of accuracy and speed of conversion.</u>

An evaluation has been conducted in the Testing chapter of this report where the accuracy of the conversion is evaluated with the raw platform which has not used best fir parameters, and the time of conversion has also been tested and documented in this chapter.

10.2.3 Achievement of Learning Outcomes

The author has developed the following mentioned skill and gained the following learning outcomes during the course of this final year project.

- As a considerable portion of the skills required for the implementation and preparation of the research were not covered during the modules. Self-studies had to be carried out to gain knowledge on the related technologies and so forth. Hence self-learning skills were improved during the course of this project.
- Technologies such as python, Tensorflow and frameworks such as Flash, Electron were learned to implement the project through online materials such as tutorials and by referring documentation of each technology.
- Research abilities were also improved in the task of studying and identifying a Research gap suitable.
- Was exposed to the Software Development Life Cycle and gained knowledge on testing and evaluation of a software project.
- Documentation skills were also improved in the process of creating the research document.

10.3 Achievement of Requirements

10.3.1 Achievement of Functional Requirements

Table 10.3:1 shows the achievement of functional requirements

FR1	**Convert Image from Raster to vector**
Remark	The system is able to convert any uploaded image from raster to vector. Therefore, it can be said that this requirement is successfully achieved.
FR2	**Classify GIS image with trained model**
Remark	The image classification model created using neural networks have the ability to classify GIS imagery to the trained classification. Therefore, this requirement is met in the system as well
FR3	**Identify conversion parameters**
Remark	The parameter files are analysed and then the classified image is analysed and the best fit parameters for that image can be identified by the system. Thus establishing that this requirement is also successfully achieved.
FR4	**Change auto selected parameters and do conversion process**
Remark	The auto generated parameters can be changed through the UI by the user and the conversion can be done to the new parameters, Therefore this requirement is also achieved in the system.
FR5	**Save converted file**
Remark	The converted file is saved to the file system of the device. Therefore this requirement is also achieved in the system

Table 10.3:1 Achievement of Non-functional Requirements

10.3.2 Achievement of Non-functional Requirements

Table 10.3:2 shows the achievement of non-functional requirements

NFR1	**Give user proper feedback on conversion process as it can be a long and time consuming process depending on the raster image and conversion parameters**
Remark	The loaders and spinners give the user feedback while the application is converting or analysing an image.
NFR2	**Result image should be visually similar and accurate to input raster image**
Remark	The result image has 70-80% accuracy; this is a good visual similarity percentage. Therefore, this non-functional requirement seems to be achieved in the system to a satisfactory level.
NFR3	**Develop API to allow conversions using online platforms**
Remark	The backend of the main code is developed as Flask API, even though this is not currently deployed as a web application it has the potential to be.

Table 10.3:2 Achievement of Non-functional Requirements

10.4 Problems and Challenges Faced

1. Lack of expertise

As the domain of GIS and Image processing technologies were a new domain of research and learning to the author, there was a lot of knowledge to be newly learned about the areas of research while working on this project.

2. **Working with python**

As the author had only been familiar with basics of this programming language and as it was a very convenient language to both GIS and image processing, classification functionality implementation. It had to be studied learned during the implementation of this project.

3. **Limited time**

As the project is to be complemented within the given time frame by the module. All the new technologies and languages that had to be learned to self-learning. It was highly challenging to learn and implement the solution within the given time period.

4. **Lack of datasets**

As the data sets used for the image classification part of this project was a unique approach to GIS image classification. The data sets for this had to be created by the author and was a time consuming and difficult process.

10.5 Limitations of the Project

1. **The parameter identification currently only supports GIS type imagery it has been trained on**

The system currently only has been trained on the three classification of GIS data that has been gathered and trained on. This is a current known limitation of the implementation, but can be fixed by gathering data se and training the system using them. As mentioned in this report this platform is extensible to any amount of classifications as long as the data sets are available.

2. **The raster to vector conversion library used is fixed and cannot be swapped out due to parameter naming conversions and uniqueness.**

The current Imagetracer library used cannot be swapped or changed with another conversion library as the parameter names are unique to this library. Therefore, limiting it to its current library only.

3. **The output image cannot be modified or altered.**

The output vector cannot be modified using the platform and has to be imported into third party applications to be edited.

10.6 Future Enhancements

Below mentioned are future enhancements that can be implemented to further improve this software application to develop it as more of a framework.

1. **Ability to plug different Conversion libraries into system for conversion.**

The system should be able to identify parameters common to the conversion libraries to be sued. Therefore, the conversion library used can be of the users’ choice, but the parameter identification works as a framework on top of the library.

2. **Provide the user with the ability to extend classification and parameter categorise to extend system**

Provide the user access to the training of data for parameters and classification. By developing a platform so the categories supported by the application can be extended by user preference.

3. **Provide user ability to modify vector before saving.**

Provide a tool set within the application itself for the user to modify and make changes to the generated vector file and save it as a new file.

10.7 Contribution

The system proposed in this research is a Raster to Vector conversion platform for GIS imagery. The contribution is that the possibility of creating a system where an image can be analysed and conversion parameters that can generate the highest structural similarity in the output being determined by the system in the conversion process. Another contribution is the development of the framework to train the parameters being fully automated which makes the platform extensible to any number of classifications as long as the data set can be created for it. The API that exposes the two endpoints for analysing and conversion of an image along with software integration is also one of the contributions of this project. The contribution areas of this project can be considered to be the following.

- Proposing an approach to identify best fit parameters to obtain an accurate conversion by analysing an image.
- Creating an already trained model for several popular GIS imagery types for classification and analysing purposes.
- Creating an API to utilize the analysing and converting functionalities developed.

10.8 Concluding Remarks

The research conducted for the Transform vector project has positive feedback from its end users, as well as domain experts. Even though Many obstacles and problems were faced during the life cycle of this project similar to any other project. With the amount of effort that was put into the project, it can be considered a successful project. The author would like to provide a self-reflection regarding the project as the final section of this document.

Transform vector is a new approach to Raster to Vector image conversion which is done by optimization of the conversion parameters to be used by the conversion library through a previously trained model on classifications of data in a GIS image. The project focuses on GIS imagery specially in this implementation. But it can be extended to other classifications using the training platform implemented as well. The project currently has limitations such as the conversion library used, and other such as only being trained to three classification of GIS imagery types, but has room for future enhancements using the training platform and making the conversion platform a framework which is common to all raster to vector conversion libraries that could support a common conversion parameter set. This project was also a good opportunity for the author to gain knowledge on the technologies used in this project such as image classification model, image processing and raster to vector conversion algorithms. The author also believes that this would open up a new approach to Raster to vector conversion enhancement.

References

Al-Bayari, O. (2018) 'GIS cloud computing methodology', *CITS 2018 - 2018 International Conference on Computer, Information and Telecommunication Systems*. IEEE, pp. 1–5. doi: 10.1109/CITS.2018.8440176.

Alamdar, F., Bahmani, Z. and Haratizadeh, S. (2010) 'Color quantization with clustering by F-PSO-GA', *Proceedings - 2010 IEEE International Conference on Intelligent Computing and Intelligent Systems, ICIS 2010*, 3(October), pp. 233–238. doi: 10.1109/ICICISYS.2010.5658548.

Berry, J. K. (1995) 'Raster is Faster, But Vector is Corrector', *GIS World*, 8(6), p. 35. Available at: https://www.esri.com.

Britannica, T. E. of E. (2014) 'No Title', *Encyclopædia Britannica*. Encyclopædia Britannica, inc. Available at: https://www.britannica.com/technology/raster-graphics.

Chrisman, N. R. (1999) 'Chrisman - What does GIS Mean', 3(2), pp. 175–186.

Corke, P. (2011) *Computer Vision: Algorithms and Applications, 1st ed.*, *Springer Tracts in Advanced Robotics*. doi: 10.1007/978-3-642-20144-8_11.

Couclelis, H. (1992) 'People Manipulate Objects but Cultivate Fields', *Proceedings from International Conference on GIS*, pp. 65–76. Available at: http://clamsitel.pbworks.com/w/file/fetch/49252349/Couclelis 1992 Objects Fields.pdf.

Eduardo A.B. da Silva, G. V. M. (2005) *The Electrical Engineering Handbook*. Edited by W.-K. CHEN. Academic Press. Available at: https://doi.org/10.1016/B978-012170960-0/50001-3.%0A(http://www.sciencedirect.com/science/article/pii/B9780121709600500013).

Gold, C. M. (2006) 'What is GIS and what is not?', *Transactions in GIS*, 10(4), pp. 505–519. doi: 10.1111/j.1467-9671.2006.01009.x.

He, B. S., Xu, X. L. and Zheng, T. (2009) 'Vector graphics rendering on mobile device', *Proceedings - 2009 WRI International Conference on Communications and Mobile Computing, CMC 2009*, 3, pp. 8–11. doi: 10.1109/CMC.2009.285.

Hilaire, X. and Tombre, K. (2006) 'Robust and accurate vectorization of line drawings', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6), pp. 890–904. doi:

10.1109/TPAMI.2006.127.

Holroyd, F. and Bell, S. B. M. (1992) 'Raster GIS: Models of raster encoding', *Computers and Geosciences*, 18(4), pp. 419–426. doi: 10.1016/0098-3004(92)90071-X.

Lacroix, V. (2009) 'Raster-to-vector conversion: problems and tools towards a solution a map segmentation application', *Proceedings of the 7th International Conference on Advances in Pattern Recognition, ICAPR 2009*, 1(c), pp. 318–321. doi: 10.1109/ICAPR.2009.96.

Li, L. and Kong, L. (2009) 'Image denoising base on non-local means with Wiener filtering in wavelet domain', *IIH-MSP 2009 - 2009 5th International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 471–474. doi: 10.1109/IIH-MSP.2009.76.

Likas, A., Vlassis, N. and J. Verbeek, J. (2003) 'The global k-means clustering algorithm', *Pattern Recognition*, 36(2), pp. 451–461. doi: 10.1016/S0031-3203(02)00060-2.

Lin, F. and Guo, C. (2011) 'Raster-vector integration based on SVG on mobile GIS platform', *Proceedings - 2011 6th International Conference on Pervasive Computing and Applications, ICPCA 2011*, pp. 378–383. doi: 10.1109/ICPCA.2011.6106534.

Lin, J. *et al.* (2015) 'Diffusion Based Vector Graphics on Mobile Devices', *Proceedings - 2015 9th International Conference on Frontier of Computer Science and Technology, FCST 2015*, pp. 153–156. doi: 10.1109/FCST.2015.12.

Liu, W. and Josep Lladós (Eds.) (2005) *Graphics Recognition: Ten Years Review and Future Perspectives, 6th International Workshop, GREC 2005 Hong Kong, China, August 25-26, 2005 Revised Selected Papers*. doi: 10.1007/11767978_5.

Palus, H. (2004) 'On Color Image Quantization by the K-Means Algorithm', (December). doi: 10.13140/2.1.5012.0649.

Rehman, A. A. and Alharthi, K. (2016) 'An introduction to research paradigms in distance education', *International Journal of Educational Investigations*, 3(October), pp. 51–59.

Seel-audom, C., Naiyapo, W. and Chouvatut, V. (2017) 'A search for Geometric shapes in a Vector Image', *9th International Conference on Knowledge and Smart Technology (KST)*, pp. 305–310.

SEVERENUK, T. *et al.* (2019) 'VECTOR GRAPHICS BASED LIVE SKETCHING METHODS

AND SYSTEMS'.

Sloan, K. R. and Tanimoto, S. L. (1979) 'Progressive Refinement of Raster Images', *IEEE Transactions on Computers*, C–28(11), pp. 871–874. doi: 10.1109/TC.1979.1675269.

Su, Q. *et al.* (2013) 'Color image quantization algorithm based on differential evolution', *Journal of Software*, 8(12), pp. 3035–3041. doi: 10.4304/jsw.8.12.3035-3041.

Wade, T. G. *et al.* (2003) 'Vector & Raster Methods', 69(12), pp. 1399–1405.

Winnemoeller, H. *et al.* (2018) 'ENHANCED VECTORIZATION OF RASTER IMAGES'.

Wong, D. W. S. and Wu, C. V. (1996) 'Spatial metadata and GIS for decision support', *Proceedings of the Annual Hawaii International Conference on System Sciences*, 3, pp. 557–566. doi: 10.1109/HICSS.1996.493251.

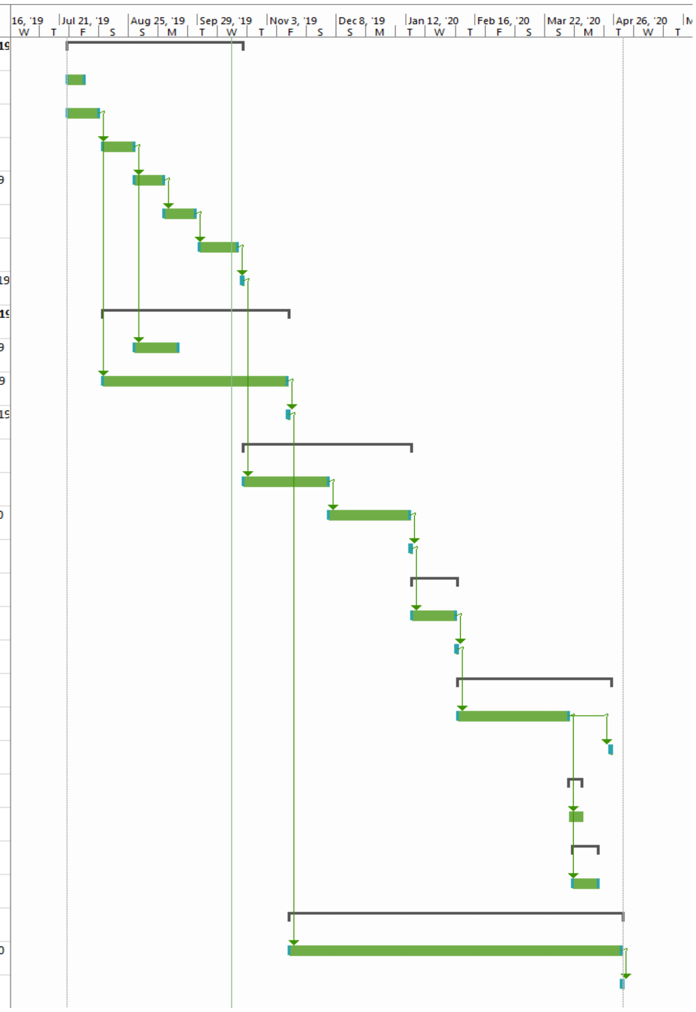
Zalaghi, H. and Khazaei, M. (2016) 'The Role of Deductive and Inductive Reasoning in Accounting Research and Standard Setting', *Asian Journal of Finance & Accounting*, 8(1), p. 23. doi: 10.5296/ajfa.v8i1.8148.

Appendix A – Work Breakdown Structure

Task Name	**Duration**	**Start**	**Finish**
1. Project Initiation	63 days	7/25/19	10/21/19
Research on the Project Background	7 days	7/25/19	8/2/19
Research on the Project Domain	12 days	7/25/19	8/9/19
Research on the existing systems and libraries	12 days	8/12/19	8/27/19
Specify Project Scope	11 days	8/28/19	9/11/19
Identify Resource Requirements	12 days	9/12/19	9/27/19
Finalizing PID	15 days	9/30/19	10/18/19
Submit PID	1 day	10/21/19	10/21/19
2. Literature Review	68 days	8/12/19	11/13/19
Research on Existing Systems	16 days	8/28/19	9/18/19
Prepare draft literature review	67 days	8/12/19	11/12/19
Submit draft literature review	1 day	11/11/19	11/11/19
3. Requirement Gathering	61 days	10/22/19	1/14/20
Requirement Specification	31 days	10/22/19	12/3/19
Create SRS document	29 days	12/4/19	1/13/20
Submit SRS document	1 day	1/14/20	1/14/20
4. Design Prototype	17 days	1/15/20	2/6/20
Create Design Specification	16 days	1/15/20	2/5/20
Submit Design Specification	1 day	2/6/20	2/6/20
5. Implementation of Prototype	56 days	2/7/20	4/24/20
Build Prototype	40 days	2/7/20	4/2/20
Prototype Report	1 day	4/24/20	4/24/20
6. Prototype Testing	5 days	4/3/20	4/9/20
System Testing	5 days	4/3/20	4/9/20
7. Prototype Evaluation	10 days	4/5/20	4/17/20
Critical evaluation of System	11 days	4/5/20	4/17/20
8. Project Finalization	121 days	11/14/19	4/30/20
Preparation of the finalized Document	120 days	11/14/19	4/29/20
Submission of the final Project Document	1 day	4/30/20	4/30/20

Appendix B – Gantt Chart

ID		Task Mode	Task Name	Duration	Start	Finish
1			**Project Initiation**	**63 days**	**Thu 7/25/19**	**Mon 10/21/19**
2			Research on the project background	7 days	Thu 7/25/19	Fri 8/2/19
3			Research on the project domain	12 days	Thu 7/25/19	Fri 8/9/19
4			Research on the exsisting systems and libraries	12 days	Mon 8/12/19	Tue 8/27/19
5			Specify Project scope	11 days	Wed 8/28/19	Wed 9/11/19
6			Identifying resource requirements	12 days	Thu 9/12/19	Fri 9/27/19
7			Finalizing the PID	15 days	Mon 9/30/19	Fri 10/18/19
8			Submit PID	1 day	Mon 10/21/19	Mon 10/21/19
9			**Literature Review**	**68 days**	**Mon 8/12/19**	**Wed 11/13/19**
10			Research on existing systems	16 days	Wed 8/28/19	Wed 9/18/19
11			Prepare draft literature review	67 days	Mon 8/12/19	Tue 11/12/19
12			Submit draft literature review	1 day	Wed 11/13/19	Wed 11/13/19
13			**Requirement Gathering**	**61 days**	**Tue 10/22/19**	**Tue 1/14/20**
14			Requirement Specification	31 days	Tue 10/22/19	Tue 12/3/19
15			Create SRS document	29 days	Wed 12/4/19	Mon 1/13/20
16			Submit SRS document	1 day	Tue 1/14/20	Tue 1/14/20
17			**Design Prototype**	**17 days**	**Wed 1/15/20**	**Thu 2/6/20**
18			Create design specification	16 days	Wed 1/15/20	Wed 2/5/20
19			Submit design specification	1 day	Thu 2/6/20	Thu 2/6/20
20			**Implementation of prototype**	**56 days**	**Fri 2/7/20**	**Fri 4/24/20**
21			Build prototype	40 days	Fri 2/7/20	Thu 4/2/20
22			Prototype report	1 day	Fri 4/24/20	Fri 4/24/20
23			**Prototype Testing**	**5 days**	**Fri 4/3/20**	**Thu 4/9/20**
24			System testing	5 days	Fri 4/3/20	Thu 4/9/20
25			**Prototype evaluation**	**10 days**	**Sun 4/5/20**	**Fri 4/17/20**
26			Critical evaluation of the system	11 days	Sun 4/5/20	Fri 4/17/20
27			**Project Finalized**	**121 days**	**Thu 11/14/19**	**Thu 4/30/20**
28			Preperation of the finalized document	120 days	Thu 11/14/19	Wed 4/29/20
29			Submission for the final project document	1 day	Thu 4/30/20	Thu 4/30/20

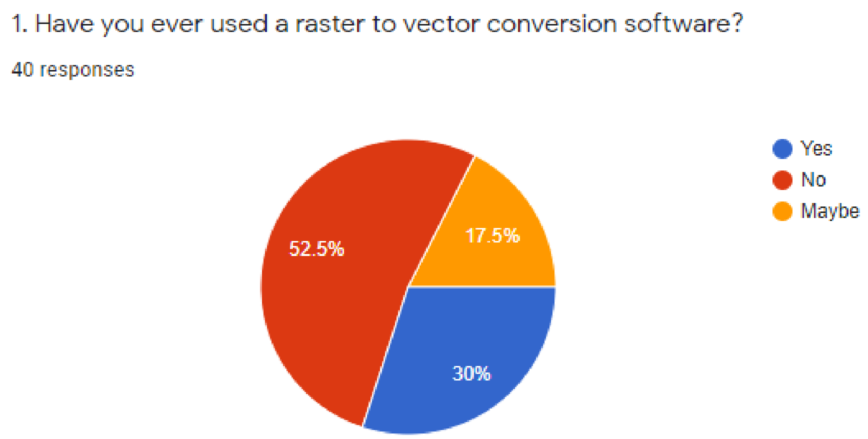


Appendix C - 1 Questionnaire for Requirement Gathering

Question 1

Aim: Identify the percentage of users familiar with raster to vector conversion software.

Observation:



The purpose of this question was to determine the percentage of the users that have used raster to vector conversion software previously. It seemed that 30% of the users had used a raster to vector conversion software and 52% of the user had no used any tool that would resemble raster to vector conversion.

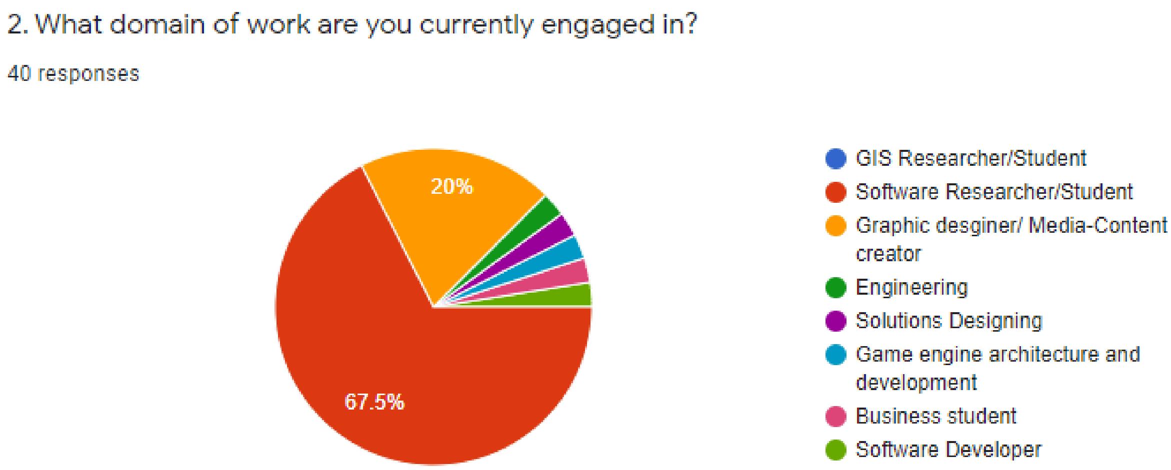
Conclusion:

The use of raster to vector conversion even in the technical and design domain is only just below a perfect third of the users. This shows that the requirement for the raster to vector conversion platform even though is niche amongst the user is necessary for some users.

Question 2

Aim: Identify domain of the users who have used raster to vector conversion software

Observation:



The purpose of this question was to determine the domain specification of the user base of the questionnaire so a better understanding as to who it needs to cater to can be identified.

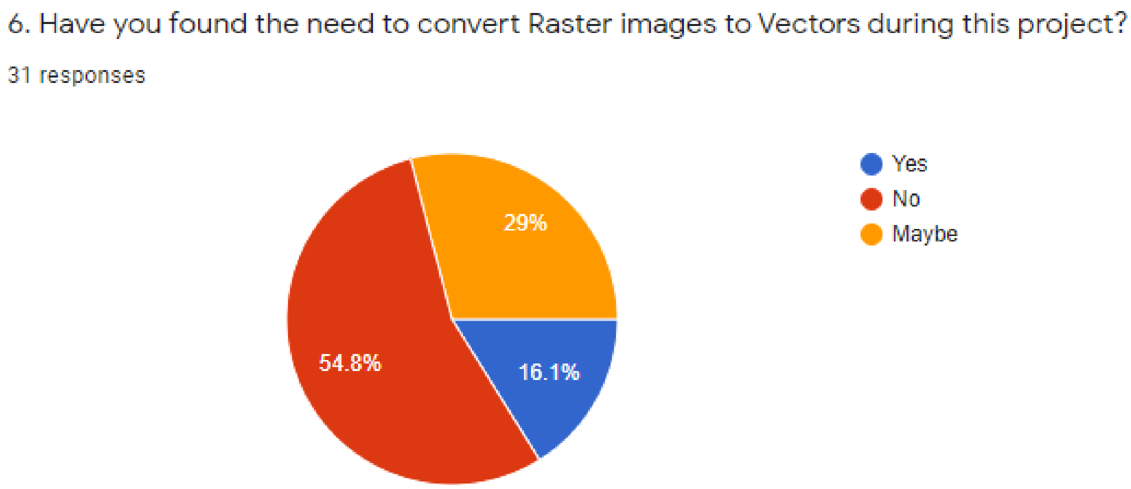
Conclusion:

It can be observed that a large base of the users that provided feedback for the questionnaire are Software Researchers/Students. While the second demographic being Graphic designers/media content creators. There is also a portion of GIS Researchers and Students. Therefore, it can be concluded that this questionnaire has reached the target demographic of users that are most likely to use a Raster to Vector conversion tool.

Question 6

Aim: Identifying the need for raster to vector conversion during GIS projects

Observation:



The purpose of this question is to identify if raster to vector conversion is necessary when conducting GIS related projects.

Conclusion:

Majority of users did not require a Raster to vector conversion software there were 16.1% results that said it was required during their GIS tasks. In conclusion it can be understood that even though raster to vector conversion is not a process that is always required in GIS related tasks, there are instances that require tools that carry out this function.

Question 7

Aim: Identify what commercial soft wares are used by the target demographic of this research questionnaire.

Observation:

7. If so what software did you use for this?

9 responses

imagemagik

Vector Magic

No

Adobe Illustrator

I didn't use a softwave

Adobe Ai

Rasterize

vectorizer.io/

I haven't yet experienced a scenario like this.

The purpose of this question was to get a better understanding on the currently used solutions for raster to vector conversion by the users who provided feedback to this questionnaire.

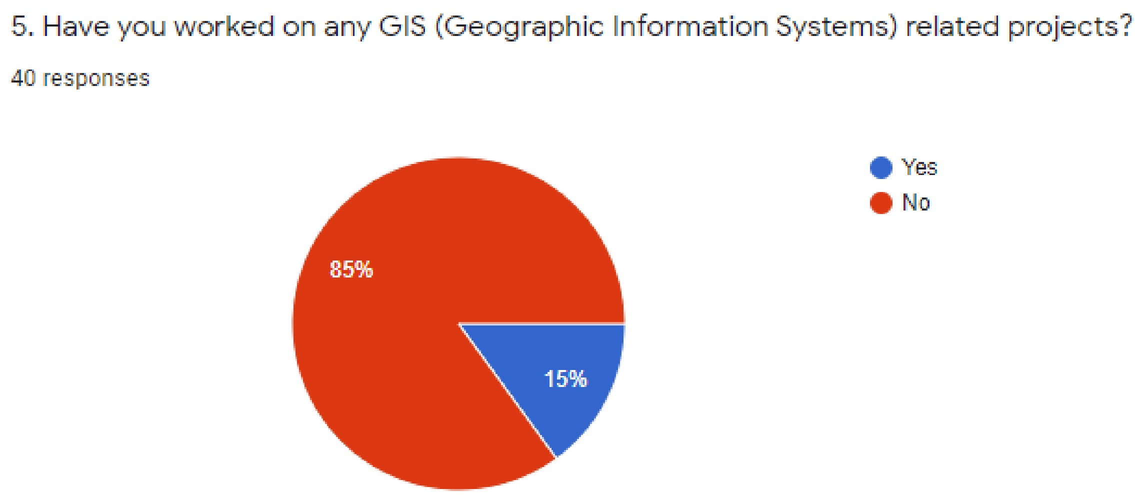
Conclusion:

It is clear that there are several soft wares that can be identified through the feedback obtained that perform raster to vector conversion as one of its functionalities or development entirely to do so. Almost all of these software has been analyzed and covered under the similar solutions analysis in the Literature review section. And can be concluded that their strengths and weakness have been taken into consideration when implementing this project.

Question 5

Aim: Identify percentage of users who have worked in GIS related project or activities

Observation:



The purpose of this question was to determine the percentage of the users that provided feedback on this questionnaire who had engaged or worked in any GIS related activities.

Conclusion:

It is clear that only 15% of the users who responded to this questionnaire have worked in any GIS related project or activities.

Appendix D – Use case diagrams

Use Case ID	**UC-1**
Use Case Name	**Open Application Window**
Priority	High
Actors	Conversion Platform User
Description	Open the application and initialize the system
Pre-condition	Application package must be downloaded and extracted onto machine running a windows OS
Extending Use Cases	none
Including Use Cases	none
Triggering Event	Click on Application icon from windows file explorer
Main Flow	1. Open Application 2. Perform Initialization functions to prepare application
Alternative Flow	none
Exceptional Flow	none

Use Case ID	**UC-3**
Use Case Name	**Verify Raster Image file format to be compatible**
Priority	High
Actors	
Description	Verify if the file format of the loaded file is one of the compatible file formats accepted by the system

Pre-condition	Application must be opened and initialization functions should have been executed and a file must be selected for the conversion process
Extending Use Cases	none
Including Use Cases	none
Triggering Event	Click on open file from file browser window
Main Flow	1. Load file onto the application 2. Check if the file belongs to one of the valid types 3. Display file loaded successfully message
Alternative Flow	3. If File is not one of the valid types show failed to load file message
Exceptional Flow	none

Use Case ID	**UC-2**
Use Case Name	**Load Raster Image to Application**
Priority	High
Actors	Conversion Platform User
Description	Load a valid raster type image file onto application for processing and conversion
Pre-condition	Application must be opened and initialization functions should have been executed
Extending Use Cases	none
Including Use Cases	• Verify Raster Image file format to be compatible • Classify Raster with GIS imagery trained model
Triggering Event	Click on open file from application window
Main Flow	1. Open file explorer window to browse and locate file

	2. Once file is selected load file into application as an array or matrices for processing and classification
Alternative Flow	none
Exceptional Flow	none

Use Case ID	**UC-4**
Use Case Name	**Classify Raster with GIS imagery trained model**
Priority	High
Actors	Conversion Platform User
Description	Classify image with trained classification model to aid automated identification of best fit conversion parameters for that particular loaded image
Pre-condition	Image must be loaded onto the application and be of a valid file format type
Extending Use Cases	none
Including Use Cases	Identify Best Parameters
Triggering Event	Click on Analyze button on application
Main Flow	1. Run classification model and classify image 2. Return classification accuracy data to find best match of GIS image classification
Alternative Flow	none
Exceptional Flow	none

Use Case ID	**UC-7**
Use Case Name	**Allow User to Modify Parameters**
Priority	High

Actors	Conversion Platform User
Description	Allow the user to adjust parameters obtained for best conversion within a certain range at the users own discretion
Pre-condition	Conversion parameters for the particular image must be set
Extending Use Cases	none
Including Use Cases	none
Triggering Event	Edit parameter button clicked
Main Flow	1. Display editable fields for user to change parameters found before conversion 2. Edit parameters 3. Save new edited parameters

Appendix E – Implementation Code snippets and explanation of sub processes

Image Resizing

```python
# resize Image
def image_resize(image_to_resize: object, set_width: int):
    width = int(image_to_resize.shape[1] * set_width / image_to_resize.shape[1])
    height = int(image_to_resize.shape[0] * set_width / image_to_resize.shape[1])
    dimensions = (width, height)
    return cv2.resize(image_to_resize, dimensions, interpolation=cv2.INTER_AREA)

```

The image pre-processing functions that ready the image for the training and ssim functions are a vital part of the whole system. Every time an image is to be imported and converted using open cv. After it is imported it has to be resized to certain dimension for operation such as quantization and structural analysis to be run on them. As some images are too large. They may cause the system to run out of memory. Hence for operations such as image classification model training and Structural similarity index to be calculated on the image data sets it should be scaled down to a constant size. This also helps with training of the image classification model as images with similar dimensions train better with less inconsistencies.

This function gets an image and a width and scales the image height, maintaining its aspect ratio to the newly defined width and returns it.

Converting SVG into PNG

```python
def convert_to_png(in_path: str, out_path: str):
    subprocess.call(["svg2png.bat", in_path, out_path])
```

As there is no python library to convert SVG images into PNG format, a library known as inkscape is being used to do this process. The requirement for an SVG image to be converted to PNG comes when the result conversion has to be compared to the original image to identify its Structural similarity as that is the index used to measure the quality of the output in this project. This code calls a sub process call and runs the svg2png batch file which accepts 2 arguments which are the path of the image to be converted and the output path where the converted image will be saved to.

Sub process is an inbuilt module in python that allows the spawning of new processes and obtain their return codes.

SVG2PNG.bat

```
cd libraries/inkscape/bin
set arg1=%1%
set arg2=%2%

set path1=%arg1%
set path2=%arg2%

inkscape  %path1% -o %path2%
```

Inkscape is an open source Vector graphics editor. It also has a command line version of its tool and also supports functionalities such as conversion vector data formats to PNG or jpeg formats. This batch file takes the power of the inksacpe command line tool and converts the image that it obtains as the first augment and saves it in the location provided in the second argument. As mentioned above this batch file is invoked though python using sub processes.

Cleanup

```python
def cleanup(base_file_path: object, category: object) -> object:
    for csv_path in os.listdir(os.path.join(base_file_path, category)):
        if csv_path.startswith("converted_"):
            os.unlink(os.path.join(base_file_path, category, csv_path))
```

Deletes all the temporary csv files created in the process of sorting and creating the final category best fit parameter csv file.

Full Training Process Executor

```python
def finalize_params():
    category = 'sat'

    variable_full_process(category)
    read_csv(category)
    cleanup('csv', category)

```

This method runs all of the methods described above in their proper sequence. This process is fully auto mated and the sample space for the best fir parameters can be increased simply by increasing the dataset of images per category. And as this is an automated process. There is no monitoring required by the user of the application to create the best fit parameter files.

Training and Saving Model

```python
def train():
    data_dir = "D:\FYP\DataSets\DataCategorised"
    categories = ["LandClassificationMaps", "Satellite", "Scanned"]

    img_size = 250
    training_data = []

    print("starting...")
    create_training_data(categories, data_dir, img_size, training_data)

    print("data loaded!")
    model = train_save_model(training_data, img_size)

    model.save('classification_model/gis_classify.h5')
```

This function calls the above two described methods of this component by passing in the necessary parameters for each function such as the data directory for the data sets, and categories array which is an array of the folder name of the categorized data set which will be used for labelling the images. The image size is also then defined. After the data set is created and the model is trained the model is saved locally to a file of file type h5. It is a Hierarchical Data Format that contain multi-dimensional array of scientific data.

Colour Counter (Analyzer Class)

```python
def count_colours(src: str):
    image = cv2.imread(src)
    unique, counts = np.unique(image.reshape(-1, image.shape[-1]), axis=0, return_counts=True)
    return counts.size
```

This method calculates the number of unique colours in the image by converting it into a numpy array and counting the number of unique values in each pixel. It is a rough value of the number of colours in the image but can be helpful to understand how to quantize the image without losing a lot of quality.

Determine Colour Clusters (Analyzer Class)

```
def identify_quntize_cluster(colors: int):
    if colors < (2 ** 4):
        return 4
    elif colors < (2 ** 8):
        return 8
    elif colors < (2 ** 12):
        return 12
    elif colors < (2 ** 16):
        return 16
    elif colors < (2 ** 20):
        return 20
    elif colors < (2 ** 24):
        return 24
    elif colors < (2 ** 28):
        return 28
    else:
        return 32
```

It is a simple nested If loop to determine the number of colour clusters required for quantization. It calculates this value using the colour counter value that is returned.

Flask API Initialization

```
from flask import Flask, jsonify, render_template, request
from flask_cors import CORS

import subprocess
import imghdr
import csv
import os

import numpy as np
import pandas as pd
import cv2

from tensorflow.keras.models import Sequential, load_model
from sklearn.cluster import MiniBatchKMeans
from skimage.metrics import structural_similarity as ssim

app = Flask(__name__)
CORS(app)
```

```
if __name__ == '__main__':
    app.run()
```

This segment of code initializes the Flask application, enables CORS for cross origin access as both these backend and frontend to communicate without the browser blocking it. And starts the application.

Appendix F – Application Console Screenshots

The Figure show the running of the Flask API Backend.

```
Run: FlaskAPI
FLASK_APP not set
FLASK_ENV = development
FLASK_DEBUG = 0
In folder D:/FYP/PythonProjects/FlaskAPI
C:\Anaconda3\envs\FlaskAPI\python.exe -m flask run
 * Environment: development
 * Debug mode: off
2020-04-30 15:51:25.338727: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library cudart64_101.dll
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

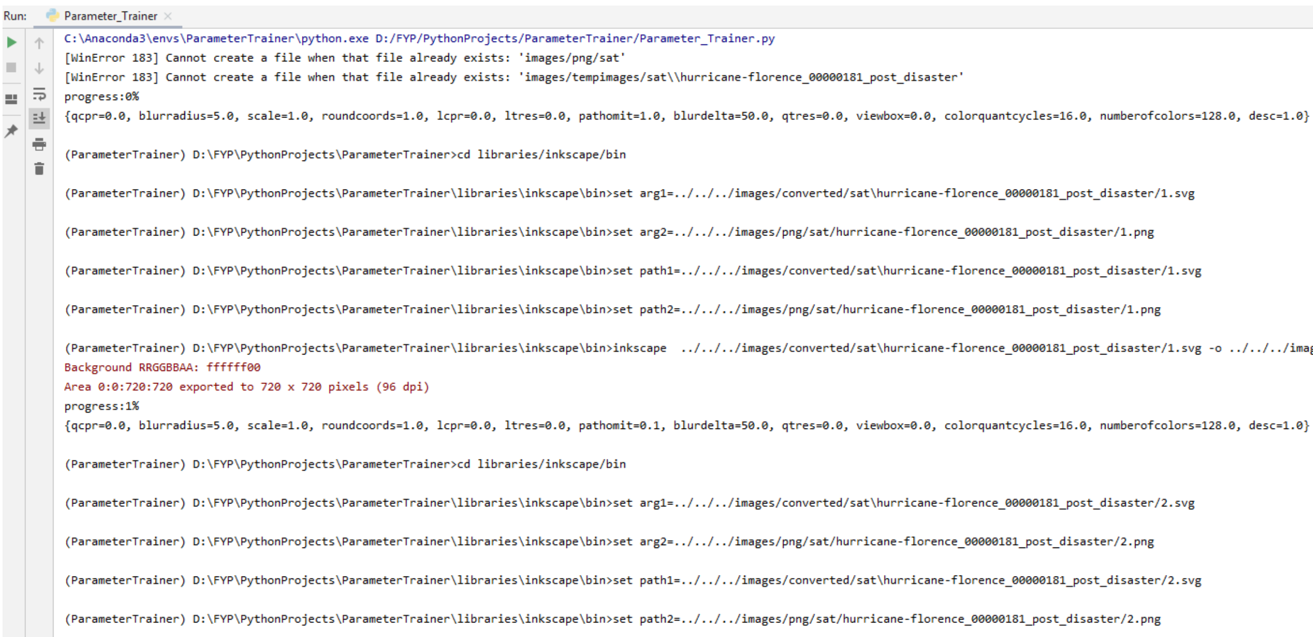
The Figure shows the response log created when responding to a request of the frontend

```
Run: FlaskAPI

(FlaskAPI) D:\FYP\PythonProjects\FlaskAPI\libraries\inkscape\bin>echo ../../../temp/png/2.png
../../../temp/png/2.png

(FlaskAPI) D:\FYP\PythonProjects\FlaskAPI\libraries\inkscape\bin>set path1=../../../temp/svg/2.svg

(FlaskAPI) D:\FYP\PythonProjects\FlaskAPI\libraries\inkscape\bin>set path2=../../../temp/png/2.png

(FlaskAPI) D:\FYP\PythonProjects\FlaskAPI\libraries\inkscape\bin>inkscape  ../../../temp/svg/2.svg -o ../../../temp/png/2.png
Background RRGGBBAA: ffffff00
Area 0:0:720:480 exported to 720 x 480 pixels (96 dpi)
127.0.0.1 - - [30/Apr/2020 15:52:39] "GET /analyze?file_path=D:\FYP\DataSets\DataCategorised\LandClassificationMaps\Aberdeenshire_Corine_2012_S.png HTTP/1.1" 200 -
```

The figure shows the training of the image classification model

```
Run: classification_trainer
1376/1418 [============================>.] - ETA: 0s - loss: 0.3603 - accuracy: 0.8510
1418/1418 [==============================] - 1s 427us/sample - loss: 0.3643 - accuracy: 0.8477 - val_loss: 0.4061 - val_accuracy: 0.8544
Epoch 20/20

  32/1418 [..............................] - ETA: 0s - loss: 0.4601 - accuracy: 0.8750
 192/1418 [===>..........................] - ETA: 0s - loss: 0.4434 - accuracy: 0.7917
 352/1418 [======>.......................] - ETA: 0s - loss: 0.4111 - accuracy: 0.8040
 512/1418 [=========>....................] - ETA: 0s - loss: 0.4028 - accuracy: 0.7910
 672/1418 [=============>................] - ETA: 0s - loss: 0.3970 - accuracy: 0.7946
 800/1418 [===============>..............] - ETA: 0s - loss: 0.4039 - accuracy: 0.7887
 960/1418 [==================>...........] - ETA: 0s - loss: 0.3989 - accuracy: 0.8021
1120/1418 [======================>.......] - ETA: 0s - loss: 0.3912 - accuracy: 0.8170
1280/1418 [==========================>...] - ETA: 0s - loss: 0.3868 - accuracy: 0.8227
1418/1418 [==============================] - 1s 409us/sample - loss: 0.3820 - accuracy: 0.8286 - val_loss: 0.3587 - val_accuracy: 0.8354
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 62500)             0
_________________________________________________________________
dense (Dense)                (None, 64)                4000064
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 195
=================================================================
Total params: 4,000,259
Trainable params: 4,000,259
Non-trainable params: 0
_________________________________________________________________
None
```

The figure shows the parameter trainer python file being executed to create the best fit parameter csv files from the data set of images per each classification.



```
Run:   Parameter_Trainer
C:\Anaconda3\envs\ParameterTrainer\python.exe D:/FYP/PythonProjects/ParameterTrainer/Parameter_Trainer.py
[WinError 183] Cannot create a file when that file already exists: 'images/png/sat'
[WinError 183] Cannot create a file when that file already exists: 'images/tempimages/sat\\hurricane-florence_00000181_post_disaster'
progress:0%
{qcpr=0.0, blurradius=5.0, scale=1.0, roundcoords=1.0, lcpr=0.0, ltres=0.0, pathomit=1.0, blurdelta=50.0, qtres=0.0, viewbox=0.0, colorquantcycles=16.0, numberofcolors=128.0, desc=1.0}

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer>cd libraries/inkscape/bin

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set arg1=../../../images/converted/sat\hurricane-florence_00000181_post_disaster/1.svg

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set arg2=../../../images/png/sat/hurricane-florence_00000181_post_disaster/1.png

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set path1=../../../images/converted/sat\hurricane-florence_00000181_post_disaster/1.svg

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set path2=../../../images/png/sat/hurricane-florence_00000181_post_disaster/1.png

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>inkscape  ../../../images/converted/sat\hurricane-florence_00000181_post_disaster/1.svg -o ../../../imag
Background RRGGBBAA: ffffff00
Area 0:0:720:720 exported to 720 x 720 pixels (96 dpi)
progress:1%
{qcpr=0.0, blurradius=5.0, scale=1.0, roundcoords=1.0, lcpr=0.0, ltres=0.0, pathomit=0.1, blurdelta=50.0, qtres=0.0, viewbox=0.0, colorquantcycles=16.0, numberofcolors=128.0, desc=1.0}

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer>cd libraries/inkscape/bin

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set arg1=../../../images/converted/sat\hurricane-florence_00000181_post_disaster/2.svg

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set arg2=../../../images/png/sat/hurricane-florence_00000181_post_disaster/2.png

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set path1=../../../images/converted/sat\hurricane-florence_00000181_post_disaster/2.svg

(ParameterTrainer) D:\FYP\PythonProjects\ParameterTrainer\libraries\inkscape\bin>set path2=../../../images/png/sat/hurricane-florence_00000181_post_disaster/2.png
```

Appendix G – Unit Test case decryption

ID	Test case	FR ID	Priority	Input	Actual Result	Expected Result	Test Result
UTC01	Load data set for variable and SSIM comparison csv creation	3	High	Data set of GIS labeled data created	All data is loaded onto system through iterations	All data is loaded onto system through iterations	Passed
UTC02	Quantize loaded image	1	High	Image, quantization cluster number	Image quantized to specified cluster value	Image to be quantized to specified cluster value	Passed
UTC03	Write parameters and SSIM to csv per each image	3	High	Images from system iterations through directories	Parameters and SSIM values calculated written to csv	Parameters and SSIM values calculated written to csv	Passed
UTC04	Sub process call to Convert to Vector	3	High	Image location, parameter details	Java lib converts image with parameters	Java lib converts image with parameters	Passed
UTC05	Bat file execution for SVG to PNG conversion	3	High	Input image (SVG) file path, file path to write out to	Convert SVG to PNG	Convert SVG to PNG	Passed
UTC06	Obtain Similarity Index for input image and output image comparison	3	High	Original image, Converted PNG from SVG, image size	Compare original and converted PNG for SSIM	Compare original and converted PNG for SSIM	Passed
UTC07	Read csv and order in ascending order of given column	3	High	Individual csv files per image with its params and SSIM values.	Order csv generated from all csv to generate final csv with best parameters	Order csv generated from all csv to generate final csv with best parameters	Passed
UTC08	Find lower range of column in csv	3	High	Final csv, field name	Find lowest value in csv row with given column name	Find lowest value in csv row with given column name	Passed
UTC09	Find Higher range of column in csv	3	High	Final csv, field name	Find highest value in csv row with given column name	Find highest value in csv row with given column name	Passed
UTC10	Find best fit parameters from given csv	3	High	Final CSV	Find best parameter range	Find best parameter range	Passed
UTC11	Classify image using image classification model file which was fitted and saved.	2	High	classification model, image	Classify passed image and return classification label	Classify passed image and return classification label	Passed

Appendix H – Conversion SSIM Test Result