OPENSOFT SUBMISSION

AUTOMATED MOSAICING OF TORN PAPER DOCUMENTS

TEAM 3

INTRODUCTION

The automated restoration of torn documents is of much interest and utility in archiving and related applications. The problem is significant because of the potential for quick solutions, the on-screen visual verification, and the time-consuming and labour-intensive nature of the manual alternative. Image mosaicing is one of the most widely used techniques for the same. A number of constraints and hurdles, both at the local and global levels need to be taken care of while applying this technique.

Section I illustrates the User Interface, which is simple and visually appealing. Section II describes our overall software pipeline. Section III describes the details of our implementation. Section IV lists specific optimizations we have done, which greatly speed up our computation. Finally, Section V mentions the additional features that could be implemented for improved performance.

I. USER INTERFACE

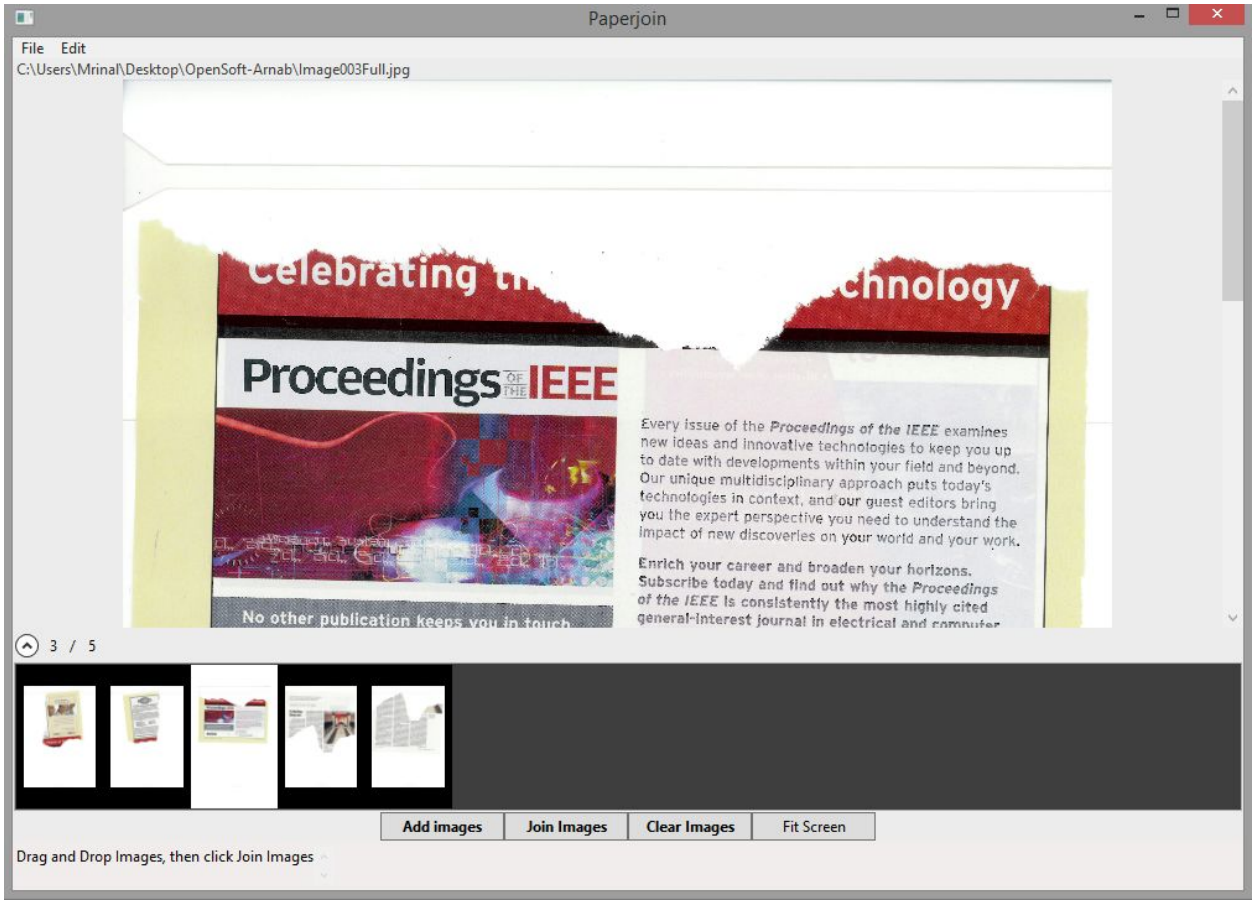


FIG 1: User Interface

Our user interface has been built in C# on Microsoft Visual Studio. There are buttons for adding images of torn pieces ('Add Images') and then for running the algorithm ('Join Images'). An option has also been given to clear the inputs and make the software ready to run again.
To take the advantage of the available screen space, one can also go to the full screen mode using the 'Fit Screen'.

II. PIPELINE

1. Extracting the foreground(torn piece) from the scanned image
2. Polygonal Approximation of the boundary of the torn piece
3. Geometrical and Content feature extraction
4. Feature matching
5. Final merging of matched images

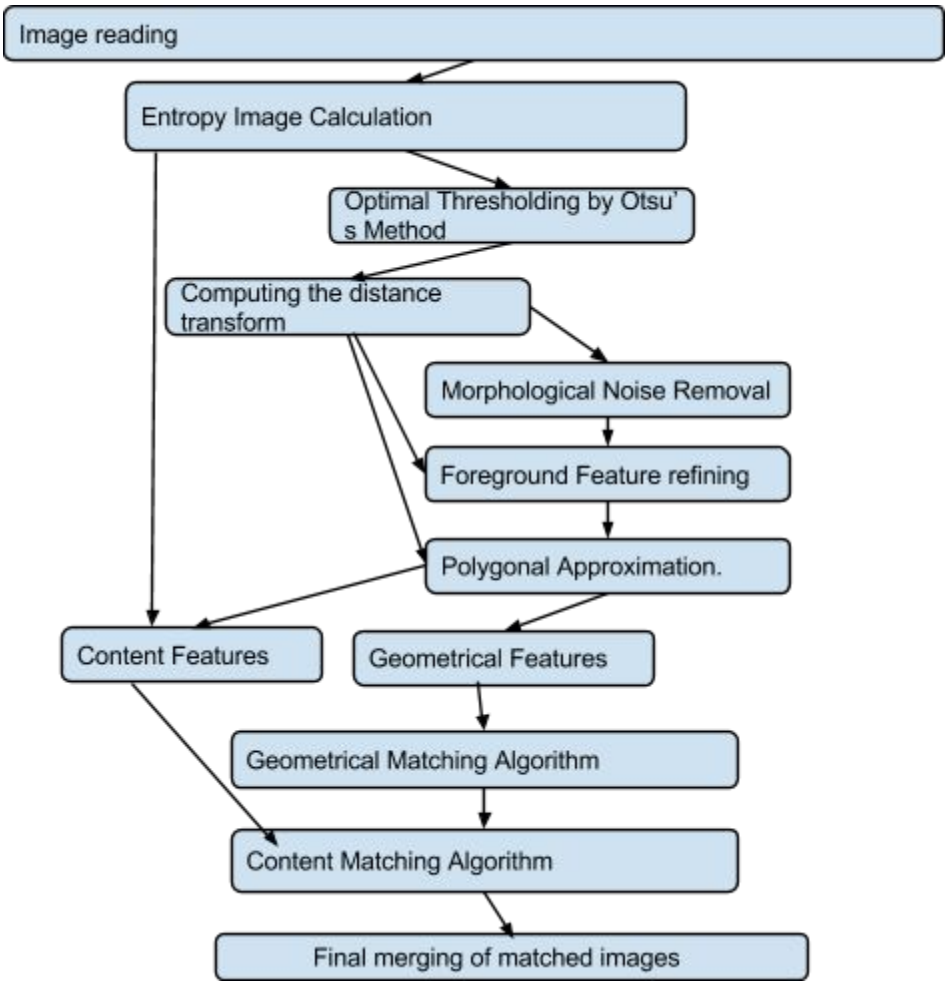


FIG 2: Flowchart of the Pipeline

III. IMPLEMENTATION

Here we have described in detail the various steps of our pipeline, and the methods used and approaches adopted for the same. Each of these steps are carried out on every image of a torn piece of the document, and so the process is described generally for a given piece.

Foreground Extraction

Each of the individual pieces of the torn document will be a coloured image over a uniform background. The first task therefore is to extract the portion of interest (the foreground) and disregard the background, which contains no useful information.
This process involves the following steps:

FIG 3: Input scanned image

1. Calculating image entropy

The entropy or average information of an image is a measure of the degree of randomness or uncertainty in the image. Low entropy images, have a lot of uniform areas. Paper texture adds unevenness in the image, raising it's entropy, whereas scanner beds are very smooth and have low entropy. It is independent of the gray levels in the image and it is only texture sensitive. The image entropy of the scanned image is calculated using Shannon's formula **[REF1]**

$$Entropy = \sum_{i}^{n} p_i log_2 p_i$$

A square window of size 9 is used for the calculation.

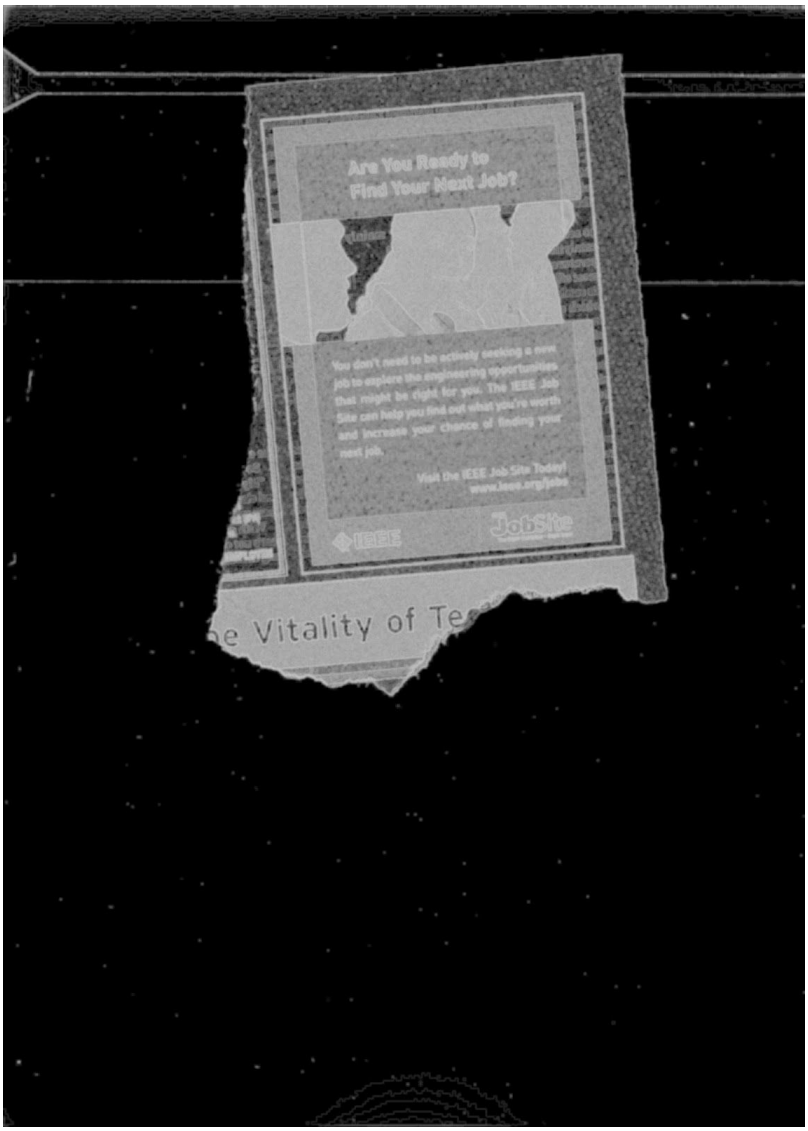


FIG 4: Entropy Image

2. Optimal thresholding by Otsu's Method

Otsu's Method is an automatic thresholding method, based on clustering. **[REF2]** It assumes there are two separate classes of pixels in the image, and thresholds the image as to minimize the inter-class variance. The method involves iterating through all the possible threshold values and calculating a measure of spread for the pixel levels each side of the threshold, i.e. the pixels that either fall in the foreground or the background. The aim is to find the threshold value where the sum of foreground and background spreads is at its minimum.
This effectively distinguishes between most of the background and foreground. But the thresholding by Otsu method is not entirely sufficient, as a lot of noise is left, which needs to be removed.

3. Computing the distance transform

The distance transform of an image **[REF3]** assigns a gray-scale value to each pixel such the intensity of the pixel is proportional to it's distance from the boundary.
Let G be a regular grid and $f: G \rightarrow R$ a function on the grid. We define the distance transform of f to be

$$D_f(p) = min_{q \in G}(d(p, q), f(q))$$

where $d(p, q)$ is some measure of the distance between p and q. Here we have used the Euclidean distance as the metric.
This step helps us in reducing the time complexity of the morphological operations of the next step from $O(n^3)$ to $O(n^2)$. It also in content-based feature extraction and polygonal approximation, which will be done in further steps. Distance transform helps in extracting and linearizing the the boundary. It is again used in content feature extraction part.

4. Morphological Noise Removal

In the context of computer vision, morphology refers to the description of the properties of shapes of areas on the image. Two fundamental morphological operations are image erosion and dilation.
The Image opening of an image f by a structuring element s is a compound morphological operation (denoted by $f \circ s$), it is an erosion $(\ominus)$ followed by a dilation $(\ominus)$:

$$f \circ s = (f \ominus s) \oplus s$$

We perform the gray-scale opening operation with a square window of size 20 on the distance transform obtained in the previous step. Morphological opening is particularly helpful in reducing the noise in the image.

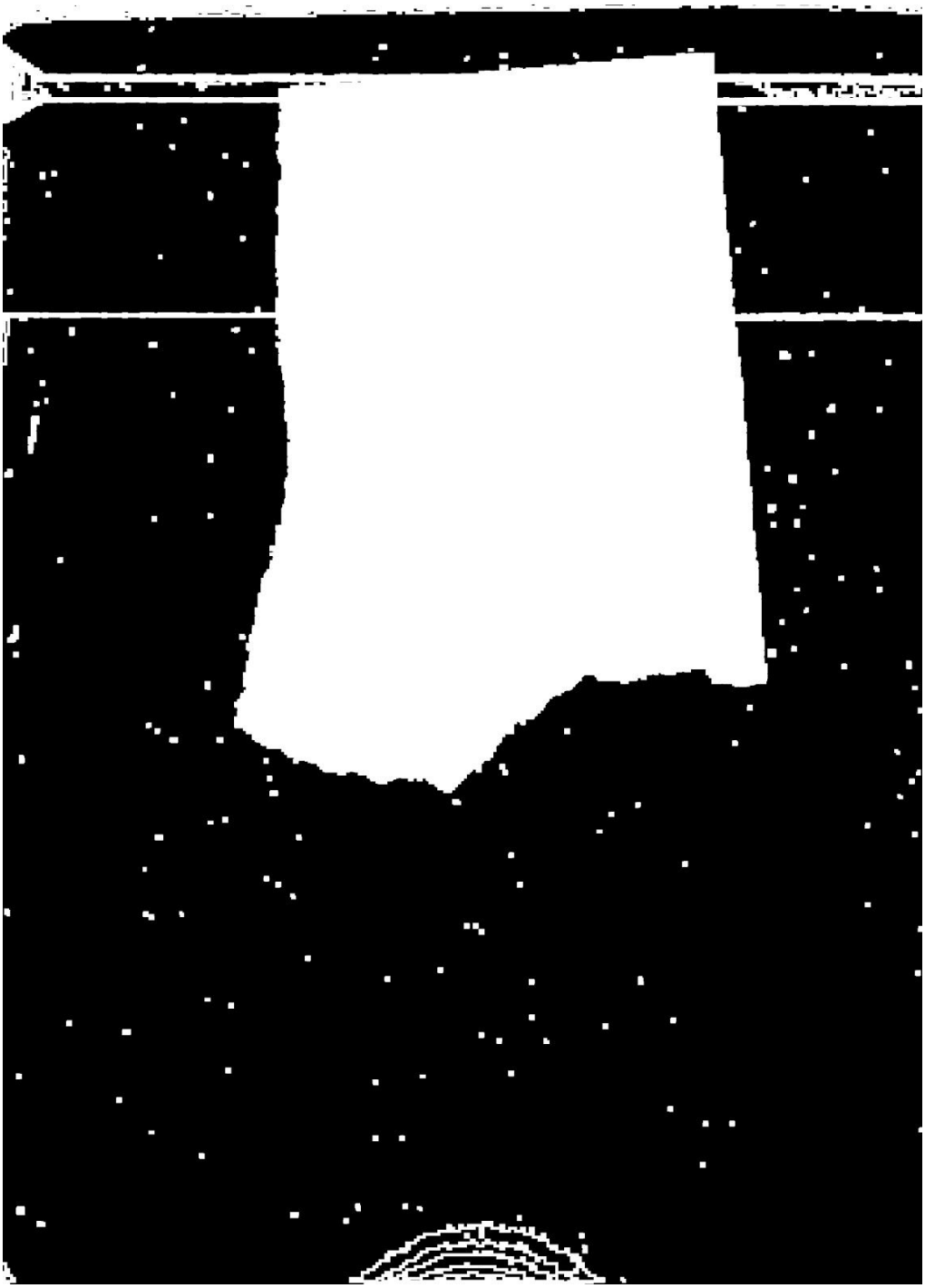


FIG 5: After Morphological Noise Removal

5. Foreground Refining via connected component extraction

Since we are given just one torn piece in the scanned image, the foreground should be a single connected piece. Capturing it as a single component is crucial for further operations. In this step we find the largest connected part of the image, label it and suppress everything else. We use breadth-first search to perform connected component labelling in one pass. This is done by reducing the image to a graph where the pixels are vertices and two vertices are connected by an edge if the corresponding pixels are 8-connected to each other (each pixel is one of the 8 neighbours of the other). Thereafter, this problem becomes akin to finding the largest connected component of the graph.The above step basically helps in excluding the random noisy patches that have survived the thresholding and noise reduction steps.

All the white regions which have distance transform of 1 from the boundary are removed along with the white components connected to it. After that other random noise and jaggedness of the boundary is removed by again applying morphological opening.

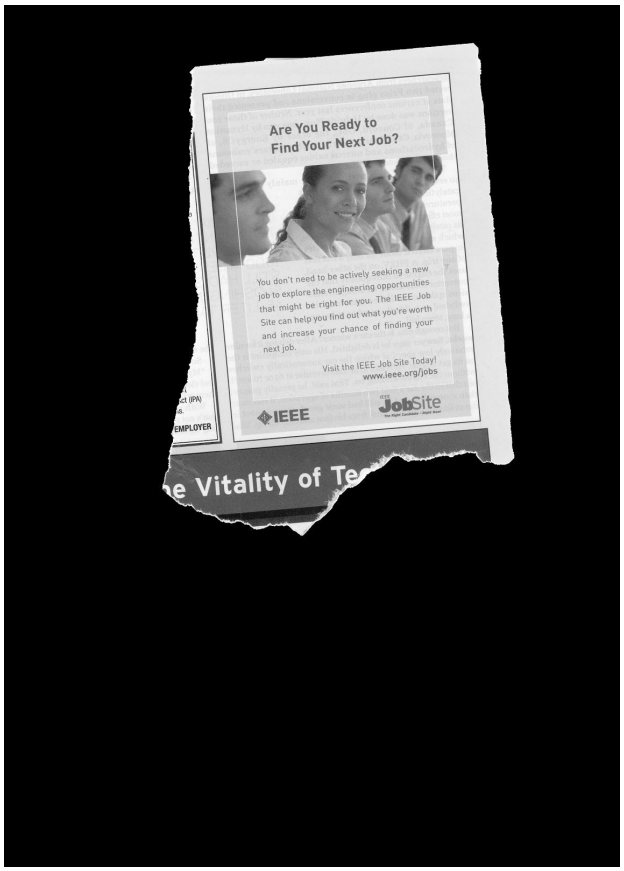


FIG 6: Final masked image

Polygonal approximation

The boundary of the mask that represents the foreground is a set of pixels, most likely in a very irregular manner, with many small edges and corners.. This would lead to a very large feature vector, upon which computations would not be possible. We approximate the extracted foreground as a polygon. Each polygon is represented as an ordered list of vertices, and the corresponding edges connecting them, and the angles at each vertex.

Our approach is based on the Ramer–Douglas–Peucker algorithm **[REF4][REF5]**. The purpose of the algorithm is, given a curve composed of line segments (in this case the foreground), to find a similar curve with fewer points. The algorithm defines 'dissimilar' based on the maximum distance between the original curve and the simplified curve. The simplified curve consists of a subset of the points that define the original curve. The algorithm works in a way such that there is no point in the original boundary that is farther than a variable parameter ε from the newly approximated polynomial. The expected runtime of the algorithm is $T(n) \in O(n \log(n))$ with worst case runtime as $O(n^2)$.

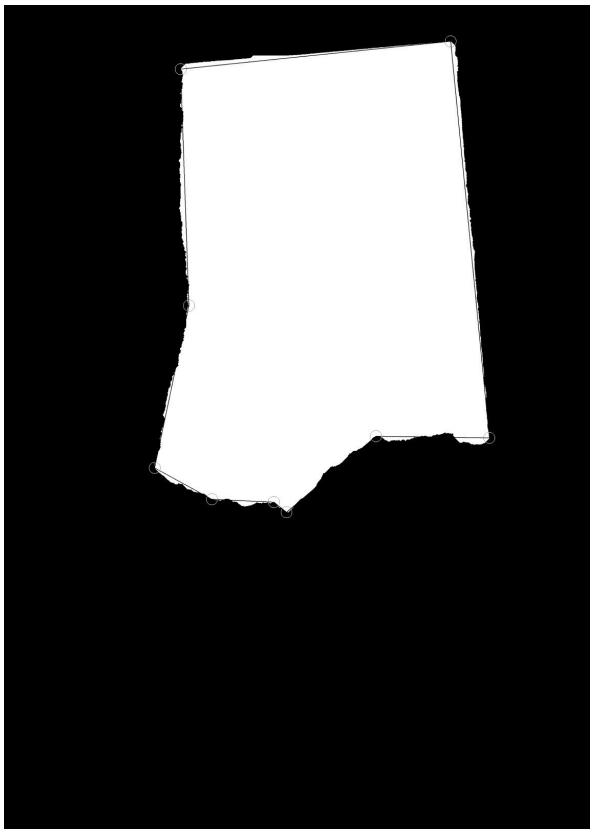


FIG 7: The boundary of the image approximated by a polygon.

Feature Extraction

Geometrical feature extraction

We first compute all the internal angles and edge lengths of the polynomial which approximates the boundary of the torn piece.

Content feature extraction

Along the polygonal edges on integral distances we draw square windows on the boundary of the torn piece. On those windows we calculate various statistical properties of the image which are mean of the red, blue and green channel, entropy and variance.

The next step, feature matching occurs in two broad sub-steps. The first is content-agnostic geometric matching based on shape characteristics. The second is a content-based feature matching.

Feature Matching

Content-agnostic geometric matching

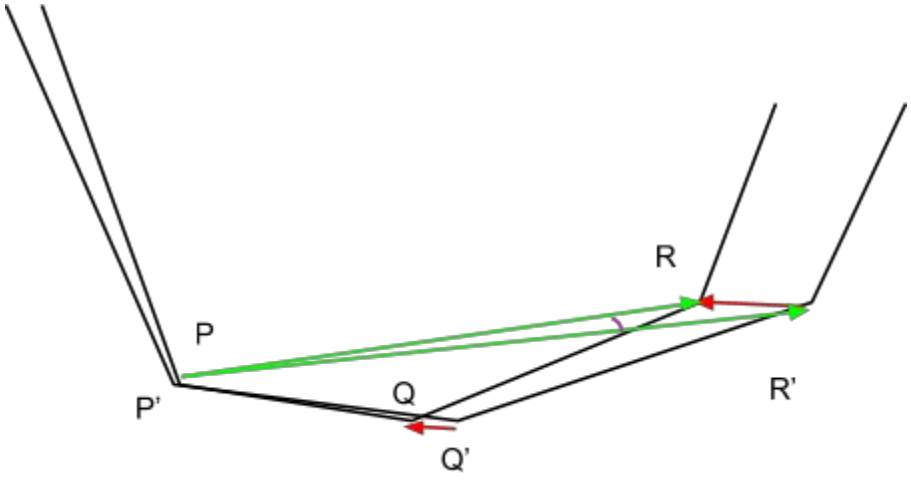
This method considers only the shape of the polygon boundaries, and tries to compute potential matches between them, without taking into account the content of the images at the corresponding borders. This assumption is made as it vastly reduces the number of possible match candidates, with a low probability of generating a false negative.
This sub-step has two aspects to it:

1. Cumulative angle sum

First we consider all possible vertex chains of all the polygons. For every pair of vertex chains for two diff polygons, we decide if these chains are potentially matching or not, using a cumulative angle sum. We compute the sum of the angles at each vertex along the two chains parallely, discarding the pair as a non-match if the difference ever becomes higher than a certain threshold. For one of the polygons we sum the internal angles, and for the other, the

external angles. This step prunes the number of potential matches significantly, while not being too computationally intensive.

2. Polygonal Matching



1. Here P and P' are the vertices, then we forcefully align the edges PQ and P'Q'.
2. If error(Euclidian distance) between the next vertices here(Q and Q`) is less than a certain threshold we keep moving to the next pair of vertices till then the error crosses the threshold.
3. The distance error in between the candidate polygons is calculated as the sum of the distance between the corresponding vertices. We then translate the polygon to minimise the mean distance error.
4. Since the matching edges also doesn't align perfectly therefore we have a rotation error. So we rotate the polygon to reduce the rotation error.
5. Go to Step 2 and repeat the process until we stop getting new vertices.

Content-aware Feature extraction

Cross-correlation between two data is given by

$$R(\tau) = \frac{E[(X_t - \mu_t)(X_s - \mu_s)]}{\sigma_t \sigma_s}$$

The cross-correlation is maximum when the data matches well, and low when it doesn't. We try to maximize the cross correlation between the boundaries of the two pieces we are trying to match, based on their content.

Final merging of matched images

The merging of polygons is via a simple step-by-step growth.

$mergepoly(P)$

Let S = $\cup P_i$ where P_i is the i^{th} polygon, $i >= 2$

Let R = P_1

$while\ |S|\ > 0$

$Select\ Q\ s.t\ Q\ \varepsilon\ S\ and\ Q\ =\ Best\ Match\ with\ R$

$R\ =\ R\ \cup Q$

$S\ =\ S \setminus Q$

$Return\ R$

IV. OPTIMIZATIONS

- We have used very few of OpenCV's inbuilt functions, as they are quite slower. Barring the reading/writing of images, all other steps have been custom implemented.
- We have made our own Image class instead of using OpenCV's Mat class, thereby speeding up our computation significantly as the latter has many error checks.
- We have optimized the calculation of image entropy from $O(n^4)$ to $O(n^3)$, and of distance transform from $O(n^4)$ to $O(n^2)$, where the former methods in each case are the more commonly used ones.
- The iterative error reduction is far more elegant than the more commonly adopted corner matching, without increasing the time complexity too much.

V. FUTURE WORK

There are certain additional features that could not be incorporated due to paucity of time. However, considering the problem from a research perspective, these methods would significantly add to the accuracy of the procedure.

- Region Classification - Distinguishing between visual, textual, non visual and non textual regions.
- Finetuning the feature-based matching with the mutual information of

VI. REFERENCES

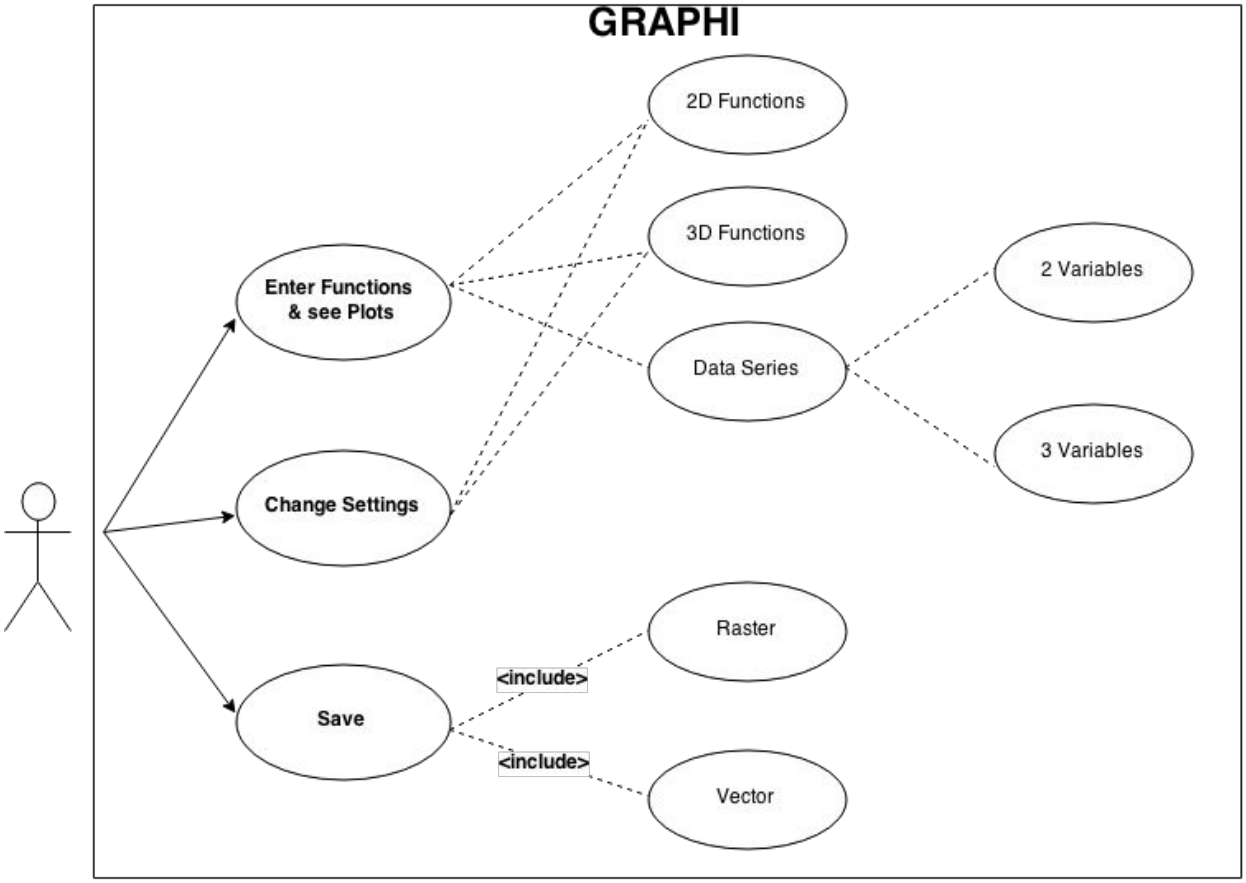
REF1 - Shannon, Claude E. (July/October 1948). "A Mathematical Theory of Communication". *Bell System Technical Journal* 27 (3): 379–423.

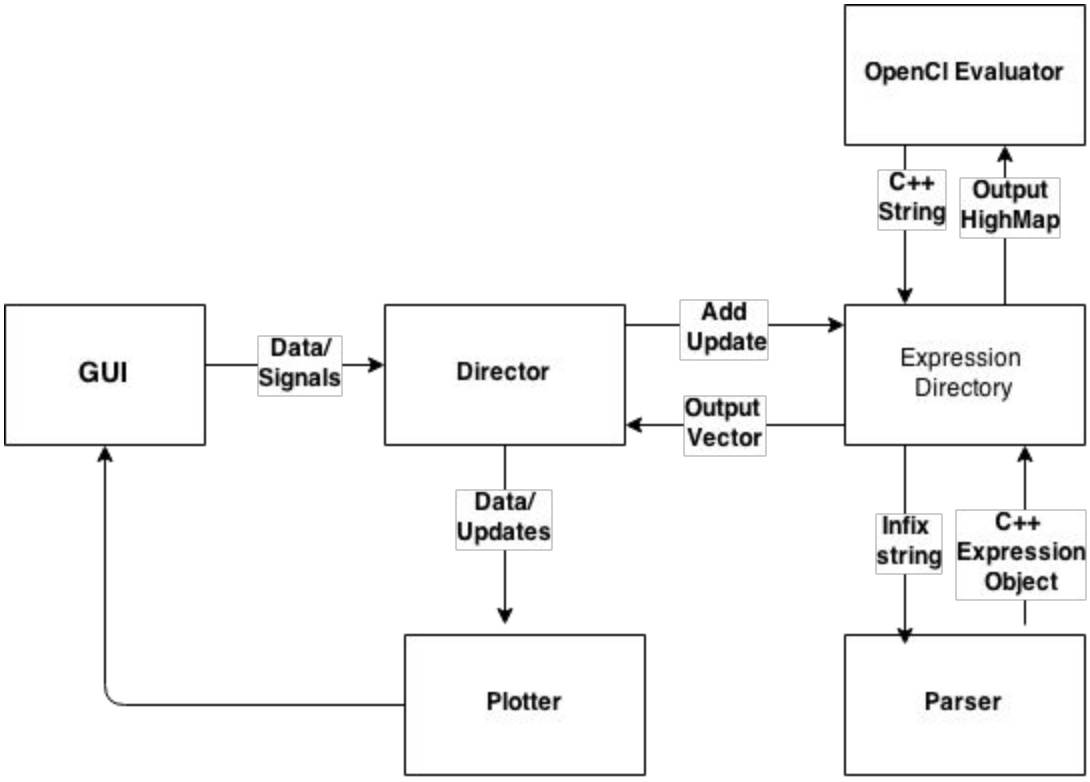
REF2 - Nobuyuki Otsu (1979). "A threshold selection method from gray-level histograms". *IEEE Trans. Sys., Man., Cyber.* 9 (1): 62–66.

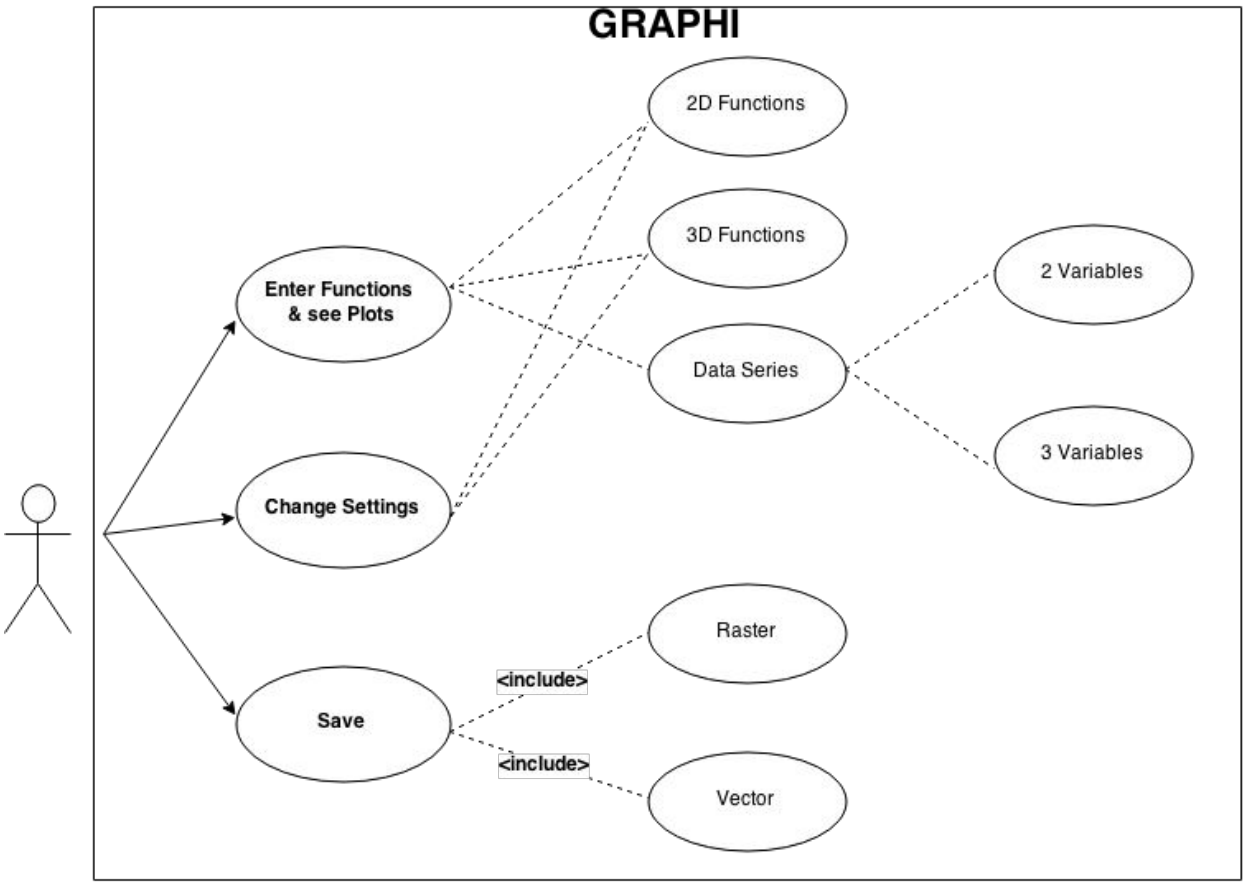
REF3 - G. Borgefors “Distance transformations in digital images”, Computer Vision, Graphics and Image Processing, 1986

REF4 - Urs Ramer, "An iterative procedure for the polygonal approximation of plane curves", Computer Graphics and Image Processing, 1(3), 244–256 (1972)

REF5 - David Douglas & Thomas Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature", The Canadian Cartographer 10(2), 112–122 (1973)


GRAPHI
2D Functions
3D Functions
Enter Functions & see Plots
Data Series
2 Variables
3 Variables
Change Settings
Raster
<include>
Save
<include>
Vector


OpenCl Evaluator
C++ String
Output HighMap
GUI
Data/ Signals
Director
Add Update
Expression Directory
Output Vector
Data/ Updates
Infix string
C++ Expression Object
Plotter
Parser


GRAPHI
2D Functions
3D Functions
Enter Functions & see Plots
Data Series
2 Variables
3 Variables
Change Settings
Raster
<include>
Save
<include>
Vector