*MINI PROJECT FINAL REPORT*

*ON*

**“RECONSTRUCTION OF BROKEN CONTOUR LINES** **EXTRACTED FROM** **TOPOGRAPHIC MAPS”**

****

**SUBMITTED BY:**

**Wangzer Namgyal Bhutia Akshata Chettri Sarina Chettri**

**(201600312) (201600316) (201600322)**

**Under the guidance of**

**Mr. Ashis Pradhan : Assistant Professor I**

**Department of Computer Science and Engineering**

**SIKKIM MANIPAL INSTITUTE OF TECHNOLOGY**

MAJHITAR, RANGPO , EAST SIKKIM

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **SL. NO.** | **CONTENT** | **PAGE NO.** |
| **1** | 1. **ABSTRACT** | **1** |
| **2** | 1. **INTRODUCTION** | **2** |
| **3** | 1. **LITERATURE SURVEY** | **4** |
| **4** | 1. **PROBLEM DEFINITION** | **7** |
| **5** | **SOLUTION STRATEGY** | **8** |
| **6** | **IMPLEMENTATION** | **12** |
| **7** | **EXPERIMENTAL RESULTS** | **18** |
| **8** | **TESTING AND VALIDATION** | **19** |
| **9** | **CONCLUSION** | **20** |
| **10** | **LIMITATIONS AND FUTURE SCOPE** | **21** |
| **11** | **GANTT CHART** | **22** |
| **12** | **REFERENCES** | **23** |

## ABSTRACT

The topographic sheet represents various morphological features like cultural, hydrography, relief and vegetation present on Earth’s surface. Contour Lines on the topological maps are known for determining the elevation. Contour lines are imaginary lines that joins different points located at the same elevation, it is usually the lowest layer in the map. The reconstruction of broken contour lines is to form the complete contour line by identifying the degree of matching among broken lines known as breakpoint connection. The problem here is if any external features (like elevation number written on the contour lines and many more similar features) lying on the line are removed then there exists breaks in between contour lines. Therefore, in this paper, the reconstruction of the broken contour lines is obtained after removal of the overlying features.

The first approach was to execute the color image segmentation to obtain the grayscale image. Secondly, the thinning algorithm was applied to refine the text appearing in the image. Thirdly, the nearest terminal points were obtained using the concept of the Euclidean distance method. Lastly, using the concept of various line and curve drawing algorithms the required result was attained.The first approach was to execute the color image segmentation to obtain the grayscale image. Secondly, the thinning algorithm was applied to refine the text appearing in the image. Thirdly, the nearest terminal points were obtained using the concept of the Euclidean distance method. Lastly, using the concept of various line and curve drawing algorithms the required result was attained.

# INTRODUCTION

A topographic map is a kind of map, which is described by a broad extent of details and significant representation of relief commonly via contour lines but also using varieties of other approaches. Contour lines are lines on the topographic maps are acquired from connecting equal elevations points. The map includes not only the contour lines but also other features like water bodies (like rivers, lakes, etc.), roads, name of the areas, bridges, temples, and many more features.The reconstruction of broken contour lines is to form the complete contour line by identifying the degree of matching among broken lines known as breakpoint connection. After segmentation and morphological operations on topographical maps, gaps occurs in contour lines. It is well known that filling these gaps and reconstruction of contour lines with high accuracy is difficult. Therefore, non-trivial semi-automatic approach is proposed to solve the problem. It is difficult to achieve accurate contour lines and clean up all other features from a scanned topographic map.

There are a lot of techniques that can be used or have been used to reconstruct the broken contour lines extracted from the sampled image. The general methodology includes the steps involved to solve the given problem i.e. to reconstruct the broken contour lines. The steps that are followed to do the same are mentioned below:

1. A sample from the topographic map is taken and scanning of the colored image is done.
2. After scanning, extraction of the brown color from the image is done to extract the contour lines (as contour lines are brown).
3. Since the extracted image contains noise, the noise is removed.
4. After removal of noise, thinning is done to get a clear idea about the contour lines.
5. To proceed with the reconstruction, the terminal points are obtained using the techniques mentioned below.
6. The optimal points are selected from the terminal points.
7. The Euclidean distance, gradients, and terminals are calculated and the optimal match is selected.

The match is then joined.

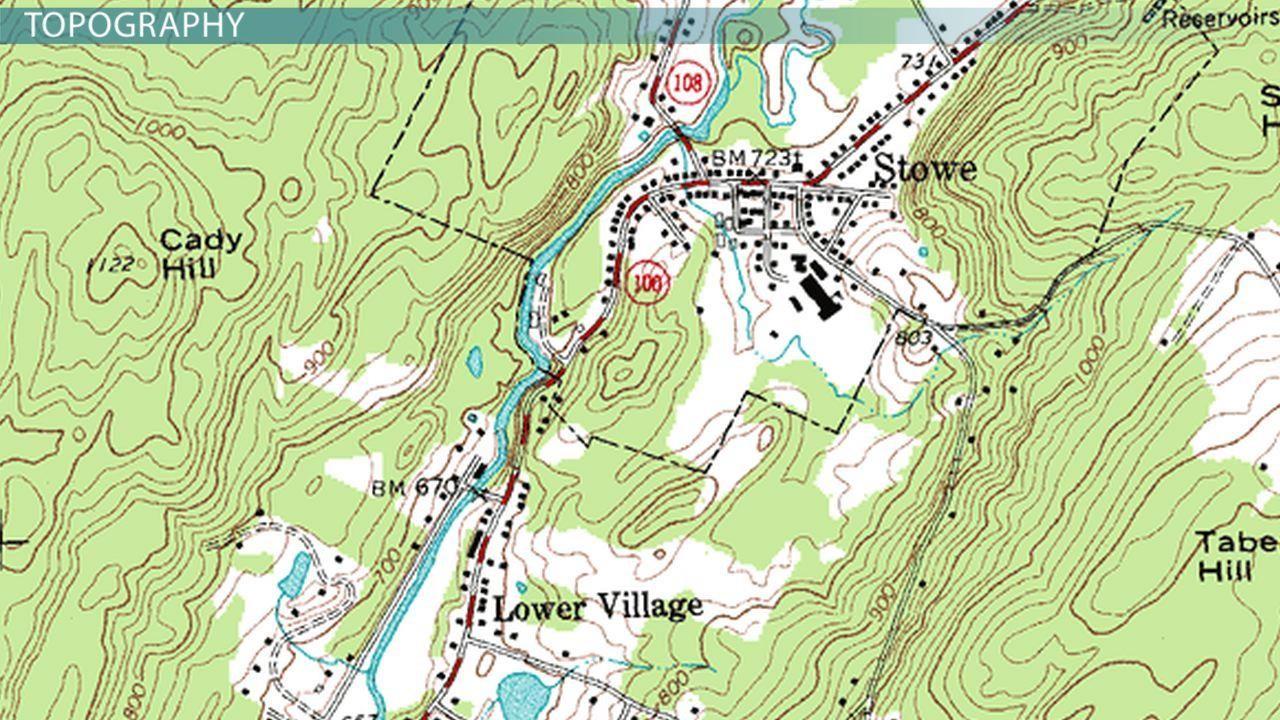


Fig 1 : Representation of Contour Lines in a Topological map

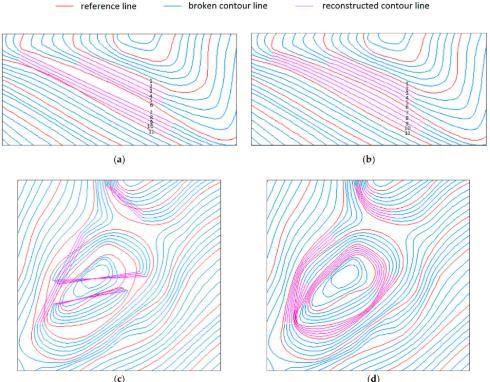


Fig 2: Representation of broken contour lines

# LITERATURE SURVEY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL. NO** | **AUTHORS** | **PAPER AND PUBLICATION DETAILS** | **FINDINGS** | **RELEVANCE** |
| 1. | Tudor G. and Remus B | “CONTOUR LINES EXTRACTION AND RECONSTRUCTION FROM TOPOGRAPHIC MAPS” | Quality Contour lines are extracted.  After extraction, binary images are processed and reconstruction of contour are implemented. | 1. Euclidean Distance. 2. The concept of Zhang Suen Thinning Algorithm. |
| 2. | R. Samet1, I.N. Askerzade Askerbeyli1, C. Varol1 | “An Implementation Of Automatic Contour Line Extraction From Scanned Digital Topographic Maps” | Proposed method contributes to the field of automated extraction of contour lines by achieving accurate contour lines and cleaning up all other features from a scanned topographic map. | 1. Model for extracting accurate contour lines. |
| 3 | Sadia Gul1 Muhammad Faisal Khan2 | “Automatic Extraction of Contour Lines from Topographic Maps” | 1. used thresholding for segmentation of the image. 2. thinning algorithm is applied. 3. Euclidean distance for the reconstruction. | 1. Thresholding 2. Thinning Algorithm 3. Euclidean distance. |
| 4 | Kenneth I. Joy | CHAIKIN’S ALGORITHMS FOR CURVES | Corner cutting or refinement algorithms specified to generate a curve from a set of control points or control polygons. | Algorithm to connect contours if they are curved. |

Other Findings:

Tudor G. and Remus B uses the concept of raster to vector conversion of lines using binary image conversion. i) In color processing: vector median filter for image smoothing. The algorithm is used for filtering capabilities and fast approximation of the Euclidean norm. Finding the distance between each pixel with all the other pixels using fixed-sized window in iteration for image smoothing.

ii) Used Gaussian Filter for image smoothing.

iii) Used Heckbert's Median-Cut Algorithm for Color Quantization of Filtered image where each the output colors should represent the same number of pixels from the input image.

iv) For Color Clustering: K-Means Algorithm is used here to segment interesting areas from the background, i.e. segmenting brown colored lines from the remaining different colored lines, resulting in Binary Image.

v) Using the concept of Zhang Suen Thinning Algorithm for reducing the contour line thickness.

vi) for the reconstruction of contour lines, they used the concept of the medial axis of curve proposed by N. Amenta using Delaunay triangulation on original set and sets of Voronoi vertices - problem: numbers on contour lines are of the same color as that of contour lines.

vii) Contour Line Vectorization and Interpolation: to save curve in vector form Freeman chain-code representation is used. Their suggestion to improve the work is to use of global information and introducing OCR (Optical Character Recognition) technique for extracting elevation values.

R. Samet1, I.N. Askerzade Askerbeyli1, C. Varol1 proposed method that contributes to the field of automated extraction of contour lines by achieving accurate contour lines and cleaning up all other features from a scanned topographic map.

L. Eikvil, K. Aas, H. Koren in their paper uses line tracing techniques for resolving the interruption by searching from the endpoints of the line.

S. Shimada, K. Maruyama, A. Matsumoto and K. Hiraki uses multiple tracing agents which follows definite contour to solve the reconstruction problem where the supervisor agent will decide which route to follow if an interrupt occurs.

S. Spinello, P. Guitton uses two computation techniques such as Voronoi diagrams and Delaunay triangulations for extraction and reconstruction of contour lines.

Jianfeng Song, Panfeng Wang, Qiguang Miao, Ruyi Liu, and Bormin Huang used GPU (Graphical Processing Unit) Implementation and CUDA ( Compute Unified Device Architecture ), developed by Nvidia for the reconstruction of contour lines. In CUDA, the basic idea is to use GPU for parallel programming which provides better performance for solving complex problems. In GPU, the hardware manages the use of lightweight threads which can switch at a low cost easily. Other findings are;

i) With the help of the linear feature separation method, denoising and removal of the background were performed.

ii) Using the Gustafson- Kessel clustering algorithm, contour lines were extracted.

iii) For reconstruction of contour lines; a) they calculated the distance by finding the central breakpoint which is obtained by backtracking; b) Euclidean distance: finding two positions of points a , b and applying the formula sqrt root of (xa-xb)^2 + (ya-yb)^2; c) CUDA & Divide and conquer patterns based GPU implementation;

iv) further they moved forward with loop-based patterns.

BinXu,1,2 JianpingChen,1,2 andMeijuanYao1,2 mentioned that for the reconstruction of broken contour lines they have used the concept of connecting pixels having identical colors (runs). They first calculated by collecting start, end, and width of connected pixels of the identical color. They found predecessor and successor runs, and with the help of its value nodes are extracted. They have taken the help of Chen et al. and Samet et al.'s algorithm for joining broken lines.

Dongjun Xin 1+, Xianzhong Zhou 2, Huali Zheng 3 used C-means Clustering for tracking the line and GGVF for extracting and reconnection of broken lines.

Sadia Gul1 Muhammad Faisal Khan2 used thresholding for segmentation of the image, a thinning algorithm is applied after thresholding. For removal of holes, hit or miss transform and other structuring elements are used to solve it. With the help of Euclidean distance, they have applied their algorithm for the reconstruction of broken contour lines.

Kenneth I. Joy explains in his paper that Chaikin’s algorithm works with control polygon directly unlike Bezier. Here, it requires a fixed ratio so that the corners are cut equally. This algorithm is equivalent to a quadratic B-spline curve algorithm and provide an effective curve drawing mechanism.

Kenneth I. Joy explains that Bresenham’s algorithm is used to draw lines on raster graphics devices. It is easy to implement and give an efficient and accurate result. Its drawback is that it does not take floating points.

T. Y. ZHANG and C. Y. SUEN develops an efficient thinning algorithm wherein each iteration removes its boundaries and corner points of digital patterns and this iteration is done in a loop until it achieves a skeleton of the pattern.

# PROBLEM DEFINITION

Incorrect and missing contour lines are most likely to occur using Traditional Methods. Also, when using Frechet distance algorithm, if the distance between two contour lines is too large, the points are sparse and the difference comparison during calculation is also large, which makes it difficult to accurately measure the similarity. Segmentation of contour lines from other limited layers of the map is difficult which leads to incorrect segmentation process. The process of finding the resulting binary image and reconstruction of contours , extracted lines that is fragmented, are affected by noise. Contour line extraction is a time consuming process by using manual techniques and procedures. DDA Algorithm and Bresenham’s Algorithm are not feasible for reconstruction of broken contour lines if the line is a curve as both the algorithms are for straight lines. Therefore, to construct a line between two nonlinear broken contour points is difficult. With increase in complex breakdown of contour line , cross dislocation and the number of missing connections in a contour gradually increases which leads to large workload including artificial checks and corrections, and the practicality of the method decreases rapidly. Another problem is that the segmented images might be incorrect and sometimes the samples might be improper.

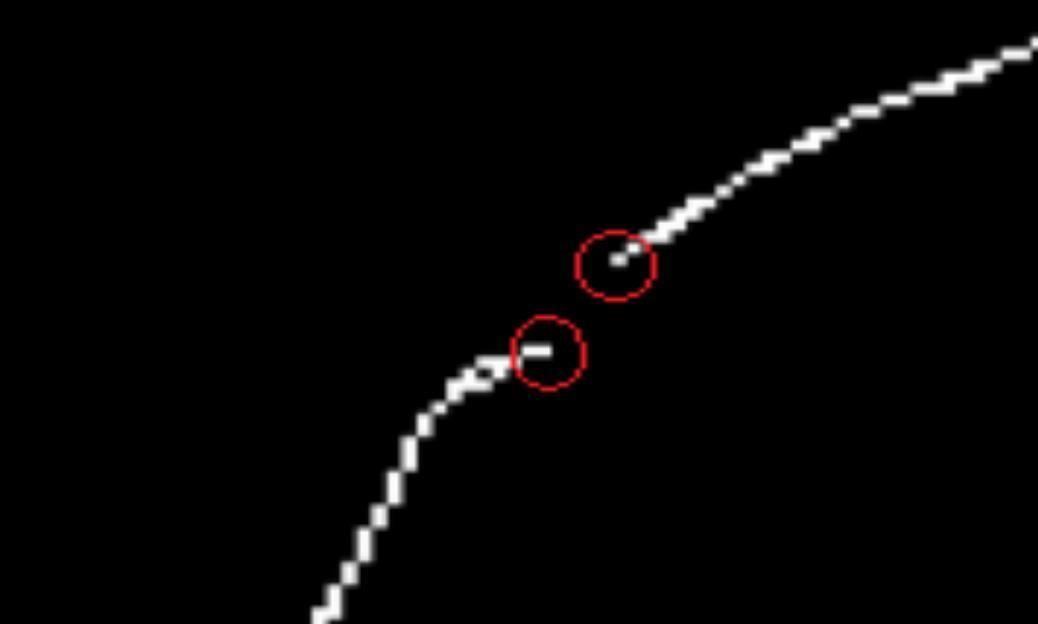


Fig 3 : Generation of broken contour lines

**SOLUTION STRATEGY**

**Table 1 : Comparative study of algorithms**

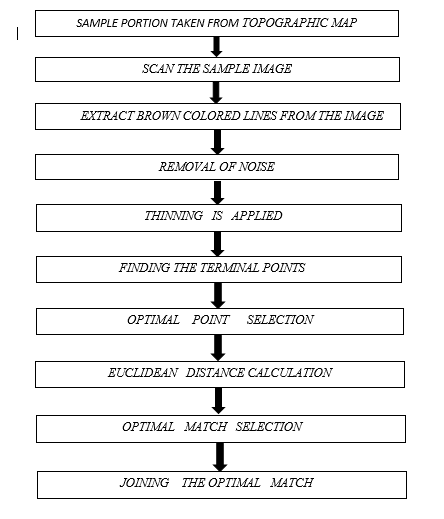
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl. No.** | **METHOD** | **ALGORITHM USED IN THE PROJECT** | **ADVANTAGE** | **DISADVANTAGE** |
| 1. | Line Drawing | Bresenham's Line Drawing Algorithm | 1. Easy to implement. 2. Uses fixed points. 3. Its gives more efficient and accurate result than that of DDA Algorithm | 1. The resultant line is not smooth even if it improves the accuracy. 2. It does not function with float values. |
| 2. | Corner Cutting | Chaikin’s Corner Cutting Algorithm | 1. Stepwise representation of a solution to a problem. 2. Can be understood even without the knowledge of programming language. | 1. Time consuming. 2. Big tasks are difficult to put into algorithms. 3. Difficult to show branching and grouping. |
| 3 | Curve Drawing | Bezier Curve | 1. Easy to draw curve as control points attract curve like a rubberband. | 1. Look of the curve depends on the control points, if any point is changed, the curve changes. 2. Computation time is directly proportional to the set of control points. |
| 4 | Line Drawing | DDA - Digital Differential Analyzer | 1. Faster technique for calculating position of the pixels. 2. Implementation is easy. 3. It can functions with float values. | 1. Time consuming when floating points are taken. 2. It is orientation dependent, hence the endpoint accuracy is poor. 3. The incrementation of accumulation of round-off errors when adding the floating-point can affect the calculation of the position of the pixels to drift away from the actual line. |
| 5 | Curve Drawing | B-Spline Algorithm | 1. It includes all necessary properties of Bezier curve. 2. It provides many control flexibility. 3. It can have a large number of control points even by using a lower degree. 4. B spline can generate better polished curves in comparison with the other interpolation techniques without a skip. | 1. Every minute change in a control point changes the entire view of the curve with the significance of the curve. 2. Huge computation time is needed for a large set of control of control points. |
| 6 | Curve Drawing | Cubic Spline Algorithm | 1. Cubic Spline technique is more suitable because it is the base extent that permits discrete control over two end points and two end derivatives. 2. Cubic Spline is also popular since it is the lowest degree that permits inflected points. 3. One of the basic pros of this method is that it can be used as an interpolating function, fitting function,adjustable smooth curve. | 1. A huge set of parameters to be estimated and also the effect of those parameters will be small due to the damages over complex fits. |
| 7 |  | Newton’s Interpolation | 1. Exact fitness of huge datasets can be done by higher order polynomials. 2. Estimation is easy in Newton's interpolation rather than non-polynomial similarity. | 1. Overfitting of the data is liable due to the consideration of the rigidity(because of smoothness) in this method. 2. Overfitting is a major problem here,hence it is more feasible to introduce another method called spline which is an assembly of polynomials in sync. |

**GENERAL METHODOLOGY :**

There are a lot of techniques that can be used or have been used to reconstruct the broken contour lines extracted from the sampled image. The general methodology includes the steps involved to solve the given problem i.e. to reconstruct the broken contour lines. The steps that are followed to do the same are mentioned below:

1. A sample from the topographic map is taken and scanning of the colored image is done.
2. After scanning, extraction of the brown color from the image is done to extract the contour lines (as contour lines are brown).
3. Since the extracted image contains noise, the noise is removed.
4. After removal of noise, thinning is done to get a clear idea about the contour lines.
5. To proceed with the reconstruction, the terminal points are obtained using the techniques mentioned in the next section.
6. The optimal points are selected from the terminal points.
7. The Euclidean distance, gradients, and terminals are calculated and the optimal match is selected. The match is then joined.

Flowchart for the general methodology is shown below:



**Fig 4: GENERAL METHODOLOGY**

**IMPLEMENTATION**

From the different methods mentioned above, the following methods were used in the project for obtaining the final result. The concepts of backtracking, gradient and euclidean distance are used to find the terminal points and to calculate the distance between the list of terminal points. The terminal points having the closest distance from the rest was matched and joined using the concept of Bezier curve drawing algorithm.

For lines with same gradients (segments of the same line) and with the shortest euclidean distance between them , the arctangent (i.e. tan inverse of (Δy / Δx) ,slope) of the terminals are compared.

If the arctangents match , the terminals are connected.

1. Finding the terminal points:

To obtain the terminal points from the input image, a 3X3 mask is used to iterate throughout the image. The center of the 3X3 mask is the terminal point in question. If the center is a pixel (white pixel value) with value 255, then the sum of the entire mask (except the center) is calculated. If the calculated sum is equal to 255, i.e. only one white pixel exists other than the center white pixel, proceed by adding the coordinates of the center pixel as a terminal to the terminals list.

Pseudo code:

1 loop till height of image - 1:

2 loop till width of image - 1:

2.1 if center of mask == 255:

2.1.1 sum = sum of the entire mask except center

2.1.2 if sum == 255

2.3.2.1 append appropriate points to a list (say terminals[] for later use)

2.3.3 end if

2.4 end if

3 end for

4 end for

1. Finding optimal matches between terminal points:

i. Finding the gradients of the terminals:

Here, the gradients of each terminal points is found that is obtained from the first step. Backtracking by one step is done for simplicity reasons i.e. going from the current terminal point to another with the same pixel value. Then the gradient is found out using equation 1:

*m =*  **- equation 1**

The gradients are stored in a list (gradientList) for computation in later stages.

Tan inverse value of every terminal is stored.

ii. Finding the euclidean distance:

For a euclidean plane (2 dimensions), the distance between terminals are calculated using eq. 2:

*d(p,q)* **= - equation 2**

iii. Optimal matches:

The entire terminal list is iterated to create optimal matches. A MatchedPoints set is maintained to prevent duplicate results.

If the current terminal coordinate is not present in the MatchedPoints set i.e. it has not yet found an optimal match, proceed with the computations, else iterate the terminal list to find a terminal coordinate that has not yet been matched.

Then a second terminal coordinate is extracted from the terminal list where the first coordinate != second coordinate and the second coordinate is not present in the MatchedPoints list.

Equation 2 is used to find the Euclidean distance between first and second points. If the distance calculated is less than the value is distance variable and the gradients of the two coordinates are complementary (i.e. one is +ve and the other -ve), store the coordinates into list “matched”. This step continues until the entire terminal list has been calculated and the result yields the combination of the first coordinate and a second coordinate with the least Euclidean distance and a complementary gradient.

Pseudo code-

1 for i in terminals:

1.1 distance = 100000( any large value)

1.2 currentTerminal = terminals[i]

1.3 if currentTerminal not in MatchedPoints(set):

1.3.1 for k in terminals:

1.3.1.1 workingTerminal = terminals[k]

1.3.1.2 if i!=k and workingTerminal not in MatchedPoints:

1.3.1.2.1 currDistance = eq. II for currentTerminal and workingTerminal.

1.3.1.2.2 if (currDistance<distance)and(product of both gradients of terminals in question= ‘-’) :

1.3.1.2.2.1 distance = currDistance

1.3.1.2.2.2 store terminals in a list (matched).

1.3.1.2.3 end if

1.3.1.2.4 if (currDistance < distance) and (gradients of both terminals are equal) and

(TanInverseValues of both terminals are equal):

1.3.1.2.4.1 distance = currDistance

1.3.1.2.4.2 store terminals in a list(matched)

1.3.1.2.4.3 end if

1.3.1.3 end if

1.3.2 end for

1.4 end if

1.5 add terminals from matched to set MatchedPoints and append to a list (MatchedTerminals).

2 end for

1. Joining the MatchedTerminals:

Here the idea of Bezier curves was used to join the optimal points. Bezier curve is one of the algorithms used in computer graphics uses to draw shapes, for CSS animation and in many other places. It is defined by control points. There may be 2 ,3, 4 or more control points.

De Casteljau's algorithm is a recursive method for the evaluation of polynomials in Bernstein form or Bézier curves.

Bezier curves can be defined for any degree of n.

A **recursive definition** for the Bézier curve of degree *n* can beexpressed as a point-to-point linear combination (linear interpolation) of two corresponding points of Bezier curves of degree *n* − 1.

Let denote the Bézier curve determined by any selection of points p0,p1,p2..pn. Then ,

and B(t) = (t) = (1-t)(t) + t(t)

**Explicit definition:**

The formula can be expressed explicitly as follows:

= 0<=t <= 1

Algorithm:

1 drawTheCurve():

1.1 numSteps = 10000 , tobeDrawn = set()

1.2 for k in range(numSteps):

1.4 t = float(k) / (numSteps - 1)

1.5 x =(deCasteljau(coorArrX, 0, n - 1, t))

1.6 y =(deCasteljau(coorArrY, 0, n - 1, t))

1.7 tobeDrawn.add((x,y))

2 def deCasteljau(coorArr, i, j, t):

2.1 if j == 0:

2.2 return coorArr[i]

2.3 return deCasteljau(coorArr, i, j - 1, t) \* (1 - t) + deCasteljau(coorArr, i + 1, j - 1, t) \* t

Output: The set tobeDrawn will contain all the points for joining the terminal points.

NOTE:

coorArrX contains x values of the two terminals and one control point.

coorArrY contains y values of the two terminals and one control point.

**Control Points:**

4 control points have been chosen for drawing the curve.

The first and last control points are the actual terminal points to be joined.

The second and third are points found after extending the terminal points with the help of the slope values.

i.e. newControlPointsX = firstTerminalsXValue + d \* firstTerminalSlopeX

newControlPointsY = firstTerminalsYValue + d\* firstTerminalSlopeY ,d = constant value.

Same procedure for the 3rd control point.

**Explanation of BEZIER CURVE - De Casteljau's algorithm**

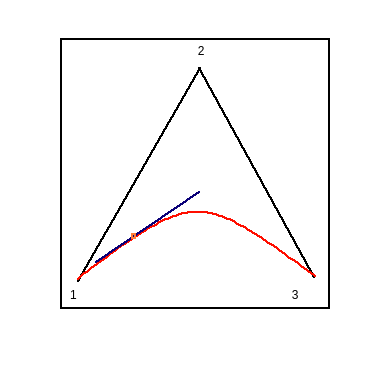
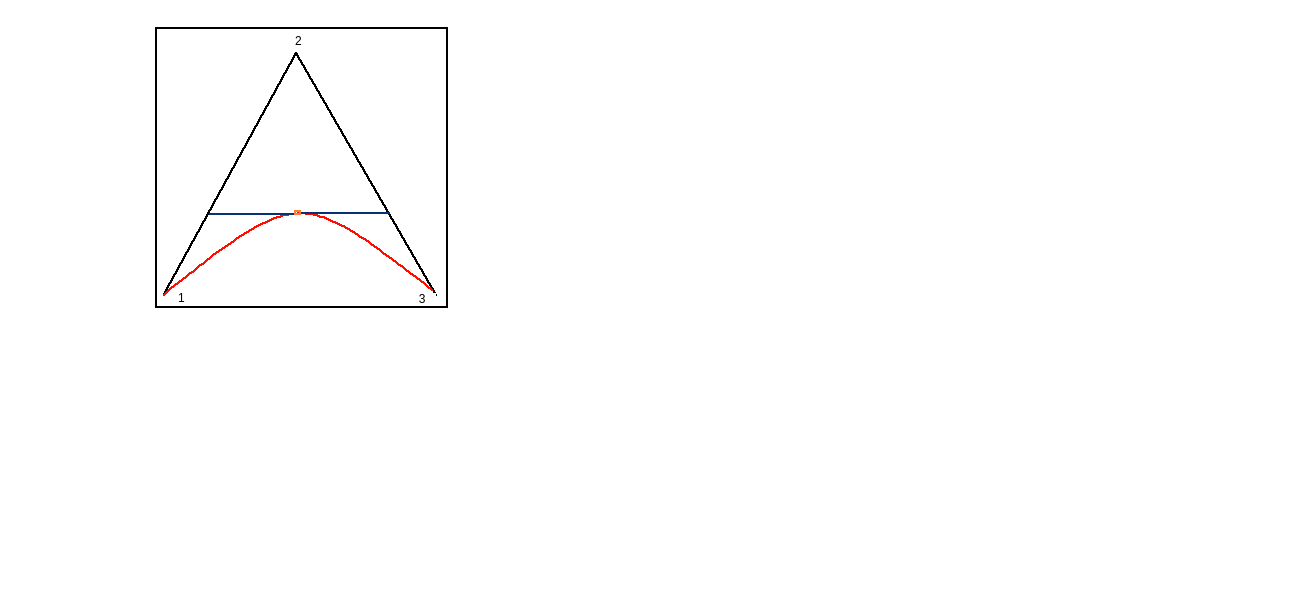
A bezier curve is generally characterized by control points, specifying as two point curve, three point curve and four point curve.

The above algorithm explains about the control points which controls the formation of a curve. Here in this technique use of bernstein polynomial is used as a basis function.

Here in this case there are three control points so the degree will be two(n-1) and the basis function will be

Q(n) = summation of P(i) from i=0 to n, B of i,n(u) where 0 will be less than or equal to u less than or equal to 1 **or** can also be written as

Q(n) = ∑P(i) from i=0 to n \* B(u) from i,n where 0<=u<=1. Here B(u) is the basis function and P(i) from i=0 to n is the control point like P1,P2,P 3 and u is called the parametric value lying in between 0 and 1 where 0 is the starting point and 1 is the ending point.

**Fig 5 Fig 6**

Fig 5 showsThree control points are drawn and marked as P1,P2,P3. Fig 6 shows

Frame portions amidst the control points 1->2->3 as shown above in the colour black. The parameter u moves from 0 to 1 and for every value of u :

1.Consider on every black portion, a dot is positioned on the length proportionate to u from its start. Since there are two segments therefore there will be two points.

2. Join the two dots(points). The blue coloured line is the connecting segment.

3. Now in that blue coloured segment, consider a point on a length proportionate to the similar value as u. Hence two points are generated one at the left part and another one in the middle and is coloured orange. Since the value u bounds from 0 to 1, each value of u sums up a point on the curve. And hence the collection of such points leads to the formation of bezier curve which is marked as red.

**The mentioned figure is the figure formed via De Casteljau’s Algorithm.**

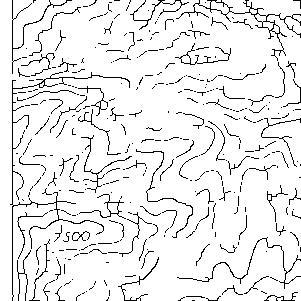
**EXPERIMENTAL RESULTS**

The reconstruction of broken contour lines were performed on various samples of topological map. First, extraction and refining of the image was done by thresholding, thinning and smoothing concepts. After the refinement is completed, the main work is to join the optimal points. For this purpose, first the terminal points are found,then the gradients of the terminal points are calculated. After computing the gradient, the nearest terminal points are matched.The concept of Bezier curve is applied here to join the points. However, the above mentioned methods do have some drawbacks like, the algorithm for finding the optimum matches does not work if the slope value is undefined i.e., for x2==x1.

Similarly, the following images shows stepwise explanation of the project.

1. First a small sample is taken from the topographic map as shown in Fig 7 and after converting the sampled image into a grayscale image thresholding (image segmentation) was performed to convert the sample into binary image shown in Fig 8.
2. Thinning was applied to the binary image where findings of terminal points were performed.

**(Gradient value of both the points = infinity , arctan(same for both terminals) is used to match the two terminals with same gradient value.)**

****

**Fig 7 : Sample topographic map Fig 8: Binarized Map Fig 9 : Thinned Map**

**TESTING AND VALIDATION**

Testing was done on various different test cases. Few of which are shown below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl** | **Input** | **Processed Output** | **Post-processed Output** |
| **1** | **C:\Users\ashis\Downloads\2a.png** |  |  |
| **2** |  |  |  |

The validation was done manually.

**CONCLUSION**

The reconstruction of the broken contour lines for a topographical map was successful for the test cases generated from the samples that were provided. At the start the removal of features was done, then extraction, dilation was also done. In the extracted image, Zhang Suen thinning algorithm was applied. Then on that thinned image, gradient of the terminal points was calculated. Using the euclidean distance concept, the optimal distance between the terminal points were calculated. The optimal points were used to find the optimal match that is joined to reconstruct the lines.

For the reconstruction, a lot of studies on different methods were done and their comparative study was prepared. With the help of the comparative study, an appropriate algorithm was prepared using the concept of Bezier curve and Chaikin's Curve drawing algorithm. Mostly the application of the bezier curve algorithm was applied. The application of the deCasteljau’s Algorithm was done for the reconstruction of the contour.

The techniques and procedures used are sufficient for the reconstruction of contour lines under specific conditions like :

A) gradients should be optimal.

B) input should be clear.

But, even after a lot of research work and preparation of new algorithms, there are some more problems yet to be solved. These include incorrect segmentation and improper samples due to the same color of the elevation number and the contour lines which was not correctly eroded. Future work includes correct segmentation of the images with proper samples, removal of the elevation numbers having the same color that of contour lines in the topographical maps. The accuracy of the code is yet to be calculated due to improper samples that lead to less test cases for the project.

**LIMITATION AND FUTURE SCOPE**

***Limitations***:

1. The use of gradients to identify the optimal matches is affected when the line has an infinite slope value. Arctan is used along with the gradient to identify the matches but this procedure doesn’t work for all cases as it is merely an assumption.
2. To obtain the gradient values,backtracking of one or two pixels is done as the input is imperfect.(Should move further back for the line to influence the results.)
3. If two line segments of different lines have the same gradient,they will be matched unless some other pixel is closer to them.

***Future Scope:***

Creating a database of contours so that it can be transformed into 3D space for DEM (Digital Elevation Model) creation.

**GANTT CHART**

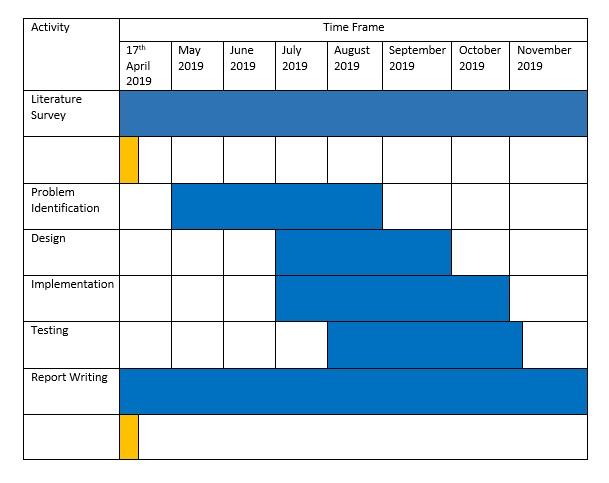




Fig 10 : Gantt Chart of Project Work Detail

**REFERENCES**

*[1] Tudor G. and Remus B., Contour Lines Extraction and Reconstruction From Topographic Maps, UbiCC Journal, 2010, pp. 681-692, DOI: 10.1.1.471.2341.*

*[2] R. Samet1, I.N. Askerzade Askerbeyli1, C. Varol1“, An Implementation Of Automatic Contour Line Extraction From Scanned Digital Topographic Maps” Appl. Comput. Math., V.9, N.1, 2010, pp.116-127.*

*[3] Kenneth I. Joy , CHAIKIN’S ALGORITHMS FOR CURVES , - Visualization and Graphics Research Group - Department of Computer Science - University of California, Davis.*

*[4] Sadia Gul1 Muhammad Faisal Khan2 , Automatic Extraction of Contour Lines from Topographic Maps, - College of Telecommunication Engineering, National University of Sciences and Technology Humayun Road, Rawalpindi, Pakistan*

*[5] L. Eikvil, K. Aas, H. Koren: Tools for interactive map conversion and vectorization, International Conference on Document Analysis and Recognition, Third International Conference on Document Analysis and Recognition, Vol. 2, pp. 927-930 (1995)*

*[6] S. Shimada, K. Maruyama, A. Matsumoto and K. Hiraki: Agent-based parallel recognition method of contour lines, In Proceedings of the Third International Conference on Document Analysis and Recognition, Vol. 1, IEEE Computer Society, Washington, DC, pp. 154 (1995)*

*[7] S. Spinello, P. Guitton:. Contour Lines Recognition from Scanned Topographic Maps, Journal of Winter School of Computer Graphics, Vol.12, No.1-3 (2004)*

*[8] Jianfeng Song, Panfeng Wang, Qiguang Miao, Ruyi Liu, and Bormin Huang, The Reconnection of Contour Lines from Scanned Color Images of Topographical Maps Based on GPU Implementation, IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING*

*[9] BinXu,1,2 JianpingChen,1,2 andMeijuanYao1,2, ResearchArticle Identification of Contour Lines from Average-Quality Scanned Topographic Maps, 1 School of Earth Sciences and Resources, China University of Geosciences (Beijing), 2 Beijing Key Laboratory of Development and Research for Land Resources Information, Beijing, Received 1 September 2015; Revised 11 December 2015; Accepted 24 December 2015*

*[10] CHAIKIN, G. An algorithm for high speed curve generation. Computer Graphics and Image Processing 3 (1974), 346–349.*

*[11] RIESENFELD, R. On Chaikin’s algorithm. IEEE Computer Graphics and Applications 4, 3 (1975), 304–310*

*[12] T. Y. ZHANG and C. Y. SUEN , A Fast Parallel Algorithm for Thinning Digital Patterns , RESEARCH CONTRIBUTIONS Image Processing and Computer Vision Robert M. Haralick Editor*

*[13] Chengming Li 1, Xiaoli Liu 1,\*, Wei Wu 2 and Zhiwei Hao 1, Article A Reconstruction Method for Broken Contour Lines Based on Similar Contours, 1 Chinese Academy of Surveying and Mapping, 2 College of Geomatics, Shandong University of Science and Technology, Received : 30 October 2018 ; Accepted : 22 December 2018 ; Published : 26 December 2018*

*[14] Kenneth I. Joy, BRESHENHAM’S ALGORITHM , Visualization and Graphics Research Group Department of Computer Science University of California, Davis*

*[15] Dongjun Xin 1+, Xianzhong Zhou 2 , Huali Zheng 3, Contour Line Extraction from Paper-based Topographic Maps , ISSN 1746-7659, England, UK Journal of Information and Computing Science Vol. 1, No. 5, 2006, pp. 275-283*

*[16] Z. Y. Yu, C. Bajaj. Vector Normalized Gradient Vector Diffusion and Image Segmentation. Proc. of 7th European Conference on Computer Vision, 2002, 517-530.*

*[17] Pavlidis, T. A Flexible Parallel Thinning Algorithm. Proc. IEEE Comput. Soc. Conf. on Pattern Recognition and Image Processing. Aug. 1981, pp. 162-167.*

*[18] J. X. Sun. Image processing, Beijing science press, 2004.*

*[19] L. M. San, S. M. Yatim, N. A. M. Sheriff, and N. I. bin Nik Ismail. Extracting contour lines from scanned topographic maps. 2004.*

*[20] E. R. Davies and A. P. N. Plummer, “Thinning algorithms: a critique and a new methodology,” Pattern Recognition, vol. 14, no.1–6,pp.53–63,1981.*

*[21] Jonghoon Seo 1, Seungho Chae 2, Jinwook Shim 2, Dongchul Kim 2, Cheolho Cheong 2 and Tack-Don Han 2,\*,Fast Contour-Tracing Algorithm Based on a Pixel-Following Method for Image Sensors, Academic Editor: Vittorio M. N. Passaro Received: 20 December 2015; Accepted: 29 February 2016; Published: 9 March 2016*

*[22] A. Soycan, M. Soycan: Digital Elevation Model Production from Scanned Topographic Contour Maps Via Thin Plate Spline Interpolation, The Arabian Journal for Science and Engineering Science, 34(1A), pp. 121-134 (2009)*

*[23] D. Xin, X. Zhou, Y. Shi(2006). Extraction algorithm of contour lines based on generalized gradient vector flow snakes model[J]. Journal of Computer-aided Design & Computer Graphics, 18(7), 982–986.*

*[24] Mohan P Pradhan, M K Ghose, Pooja S Rai, Nilanjan Mukherjee( 2013). Knowledge based contour line reconnection techniques[J]. International Journal of Computer Applications ,65(9),37–4*

*[25] H. Lei, L. Li, P. Zhang, M. Wang, and X. Li, “A binarization algorithm for digital meter image based on gray-scale morphology,”inProc.3rdInt. Congr. Image Signal Process., 2010, vol. 5, pp. 2427–2429.*

*[26] R. Samet, E. Hancer (2012). A new approach to the reconstruction of contour lines extracted from topographic maps[J]. J. Vis. Commun. Image R. 23, 642–647.*

*[27] L. Mun San, S. Mat Yatim, N. Azam MD Sheriff, N. Isrozaidi bin Nik Ismail(2004). Extracting contour lines from scanned topo-graphic maps, in: International Conference on CGIV’2004.*

*[28] D. Xin, X. Zhou, H. Zheng(2006). Contour line extraction from paper-based topographic maps, J. Inform. Comput. Sci. 1 (5), 275–283.*

*[29] F. Dupont, M. Deseilligny, M. Gondran:Terrain Reconstruction from ScannedTopographic Maps, Proceeding Third International Workshop Graphics Recognition,pp. 53-60 (1999).*

*[30] J. Du, Y. Zhang: Automatic extraction of contour lines from scanned topographic map,.Geoscience and Remote Sensing Symposium,IGARSS '04, Proceedings, IEEE International,Vol. 5, pp. 2886-2888 (2004).*