Journal of the Korean Society of Remote Sensing, Vol. 9, No. 2, 1993

그래프이론적 알고리즘들을 이용한 2차원 지형도로 부터 DEM의 자동생성방법*

구 자 영
단국대학교 전산통계학과

Automatic Generation of Digital Elevation Model from 2D Terrain Map Using Graph-theoretic Algorithms

Ja Young Koo
Department of Computer Science and Statistics,
Dankook University, Seoul, 140-714

Abstract

Digitalized topographic information is necessary for many areas such as landscape analysis, civil engineering planning and design, and geographic information systems. It can also be used in flight simulator and automatic navigation of unmanned plane if it is stored in computer in relevant format. Topographic information is coded with various symbols including contour lines, and is analyzed by trained personnels. The information should be stored in computer for automatic analysis, but it requires a lot of time and manpower to enter the contours using manual input devices such as digitizing tablet. This paper deals with automatic extraction and reconstruction of 3D topographic information from 2D terrain map. Several algorithms were developed in this work including contour segment finding algorithm and contour segment linking algorithm. The algorithms were tested using real 2D terrain map.

* 본 연구는 1993년도 교내연구비의 일부 지원을 받아 수행된 것임

Journal of the Korean Society of Remote Sensing, Vol. 9, No. 2, 1993

요 약

디지탈화된 지형 정보는 조망분석, 토목계획이나 설계, 또는 지리정보시스템등 여러 분야에 필요한 요소가 된다. 또한 컴퓨터에 적절한 형식으로 입력된 지형정보는 모의 비행훈련이나 무인비행장치에서도 사용될 수 있다. 지형정보는 지도위에 등고선을 비롯한 여러가지 기호들로 표시되어 있어서 훈련된 요원에 의해 분석된다. 자동분석을 위해서는 컴퓨터에 입력되어야 하는데 디지타이징 타블렛과 같은 장비를 사용하여 수동으로 입력하는 것은 많은 시간과 인력을 필요로 한다. 본 논문에서는 이차원 지형도로부터 삼차원 지형정보를 자동적으로 추출하는 방법을 다루고 있다. 등고선 조각 추출 및 등고선조각연결 알고리즘을 포함하는 몇 가지 알고리즘들이 제안되었고 실제 지형도를 사용하여 실험되었다.

1. Introduction

Digital description of topographical information is essential to many applications such as landscape analysis, civil engineering planning, flight simulation, and automatic navigation of unmanned plane, etc. The topographic information is coded in the contour map and can be digitized using digitizing tablet, but the manual operation over wide area requires a lot of time and manpower. Several researches deal with automatic extraction of contours in the map(Viseshin and Murai,1990; Park et al.,1990). However, none of these deals with automatic method of linking broken contours. In this paper a new contour extraction method is proposed, which includes automatic finding and linking of the contour segments.

A graph is constructed to find the contour segments. Contour pixels are treated as the vertices of the graph, and closely located vertices are connected by edges. The minimum spanning tree(MST)(Horowitz and Sahni,1978; Zahn,1971) of each connected component of the graph is constructed, and the longest path of the tree is chosen to be the contour segment.

Another graph is constructed to link the contour segments. The end points of the contour segments are treated as the vertices of the graph, and closely located vertices are connected by edges. A cost function is defined for subgraphs of the graph, and the optimal link is chosen among the subgraphs which minimizes the cost function. Figure 1. shows the overall procedure.

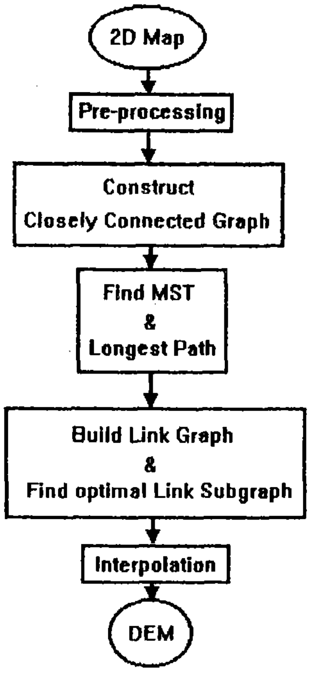
2D Map
Pre-processing
Construct Closely Connected Graph
Find MST & Longest Path
Build Link Graph & Find optimal Link Subgraph
Interpolation
DEM

Figure 1. Overall Processing

2. Preprocessing

The map contains a lot of items such as letters, numbers, and symbols as well as contour lines, so a method of discriminating contours from the other symbols is needed. The scanned map image consists of pixels with red, green, and blue components, each of which is one of the possible 256 values. Because the color of the contour lines is distinguishable from the colors of the other symbols, it clusters in the three dimensional RGB color space. In this research k-means clustering algorithm(Tou and Gonzalez,1974) is used for the color separation and the result is shown in figure 3. The color separation cannot extract the contours completely but there remains some noise. The remaining noise was reduced by the spatial filtering. Furthermore, thinning algorithm(Zhang and Suen,1984) was used to extract curve segments.

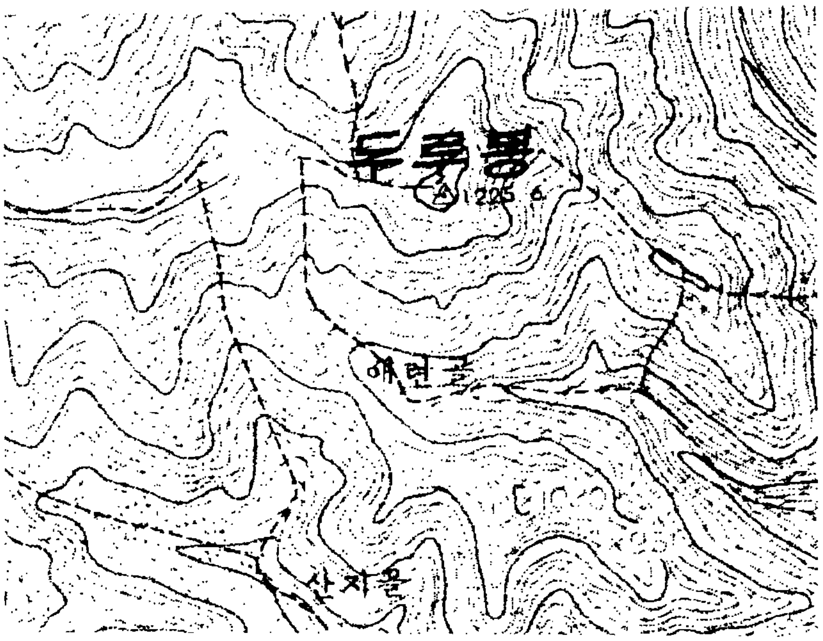

Figure 2. Scanned Image of Map

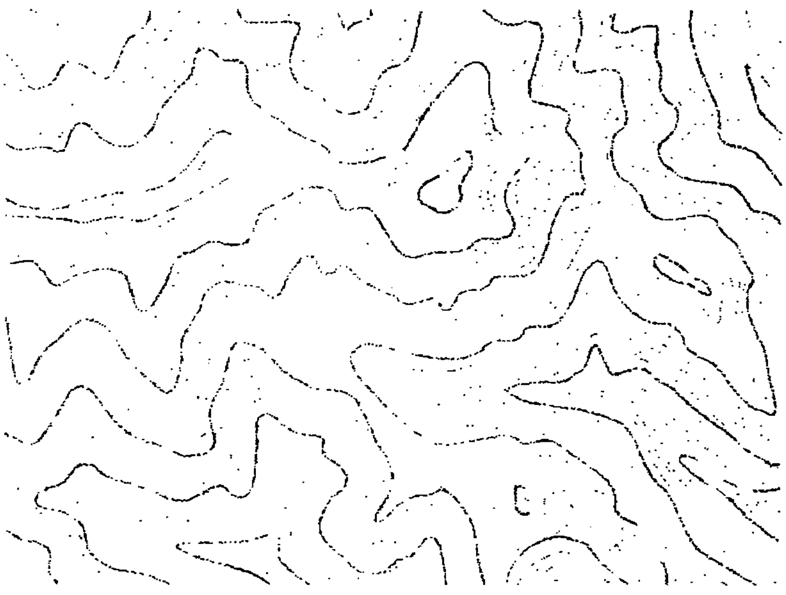

Figure 3. Result of Color Separation

3. Contour reconstruction

The extracted contour pixels can be broken by other symbols such as characters, by imperfect color separation, or by spatial filtering and thinning. If the contour pixel is connected to the neighbour pixel on the basis of distance and angle, significant error will occur. This paper proposes two step contour extraction method. In the first step, contour segments are constructed by detecting curves in the clusters of closely located contour pixels. In the second step, the contour segments are connected in such a way that minimizes a cost function.

3. 1 Finding the contour segments

A contour pixel is considered as a vertex in a graph. Two vertices are connected by an edge if the Euclidean distance between the two vertices is less than Ds, a prescribed value, and a graph of closely located pixels is constructed. The graph is represented as a set of connected components, and a contour segment is extracted from a connected component. The connected component containing the vertex vi is constructed by the following algorithm.

```
V={vi} : set of vertices
E={}   : set of edges

Find_component(vi)
begin
    for (each vertex vj such that |eij| ≤ Ds) do
    begin
        V = V ∪ {vj}
        E = E ∪ {eij}
        if (vj is first visited) Find_component(vj)
    end
end
```

The minimum spanning tree of a connected component is constructed and the contour segment is the longest path in the tree. An example is shown in figure 4.

3. 2 Linking the contour segments

The result of the algorithm presented in the last section is shown in figure 5. The next thing is to link the segments to construct the complete contours. A segment from another

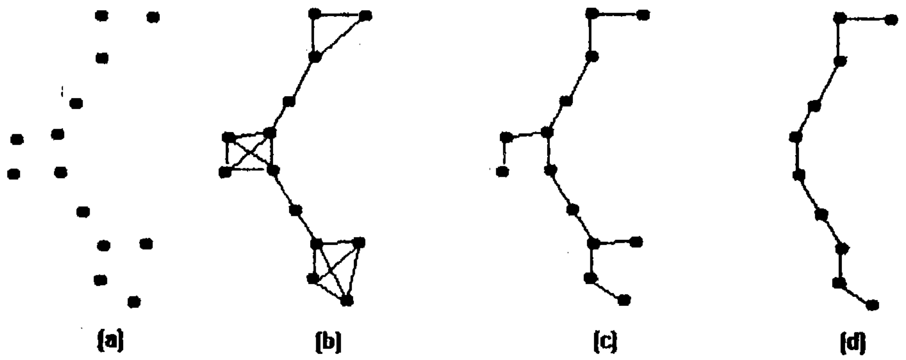

Figure 4. An Example of Finding Contour Segments
a) contour pixels, b) closely connected graph, c) MST of b), d) the longest path.

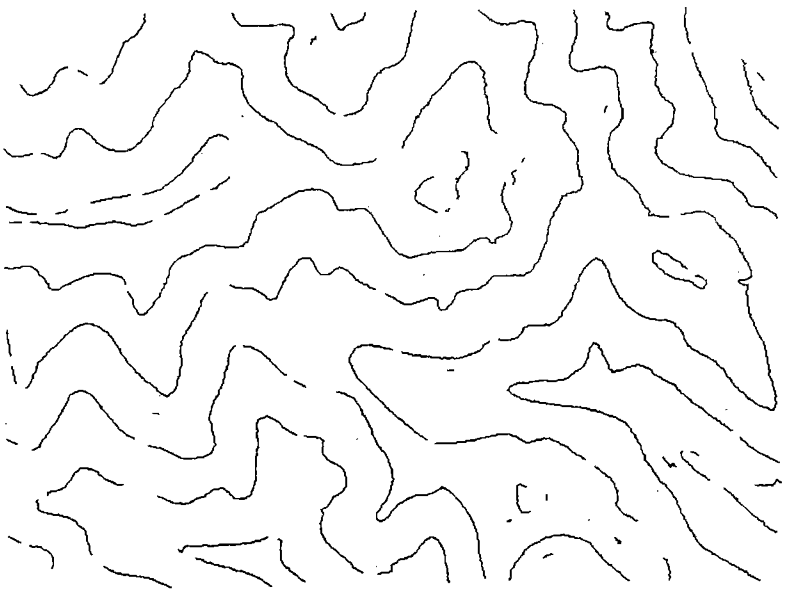

Figure 5. Extracted Contour Segments

contour line can be the nearest segment, so it is not reasonable to link the contour segments on the basis of local characteristics such as distances and proceeding angles. The algorithm proposed in this section operates robustly by considering the conflicts between the links.

The end points of each contour segment is considered as vertices in a graph. The link graph is constructed by linking the two vertices whose distance is less than Dl, a prescribed value, similar to the algorithm in section 3.1. The link graph is represented as a set of connected components, and the segment linking is done by finding a link subgraph, which is unambiguous and minimizes a cost function which will be described shortly. A subgraph of a connected component whose vertices do not have degree greater than 1 is defined to be unambiguous subgraph. An end point of a contour segment should not be connected to multiple segments, so correct link should satisfy the condition that it is an edge of the unambiguous link subgraph.

The cost function assigned to each vertex satisfies the following condition.

1. Degree 0 vertex has sufficiently large value.
2. Vertices connected to crossing edges has sufficiently large value.
3. The vertex is an end point of a contour segment, so the sharper the angle between the contour segment and the link edge the larger the cost.
4. Vertices connected to longer edges have larger cost.

An unambiguous link subgraph is constructed by the following algorithm.

```
C0 : cost of degree 0 vertex
V  : set of vertices of a component of link graph
E  : set of edges of a component of link graph

function Min_cost(V,E)
begin
    if (E = {}) return(|V| * C0)
            /* |V| : number of degree 0 vertices */
    for (eij ∈ E) do
    begin
        costij = ci + cj + Min_cost(V-{vi,vj},E-{ekl|k=i or l=j})
```

```
            /* ci, cj : cost of vertex vi and vj */
        end
        return min(costij)
    end
```

An example illustrating the above procedure is shown in figure 6. a) shows a connected component of the link graph and b)~e) are unambiguous link subgraphs and d) is the link with minimum cost function. The result for the real date is shown in figure 7.

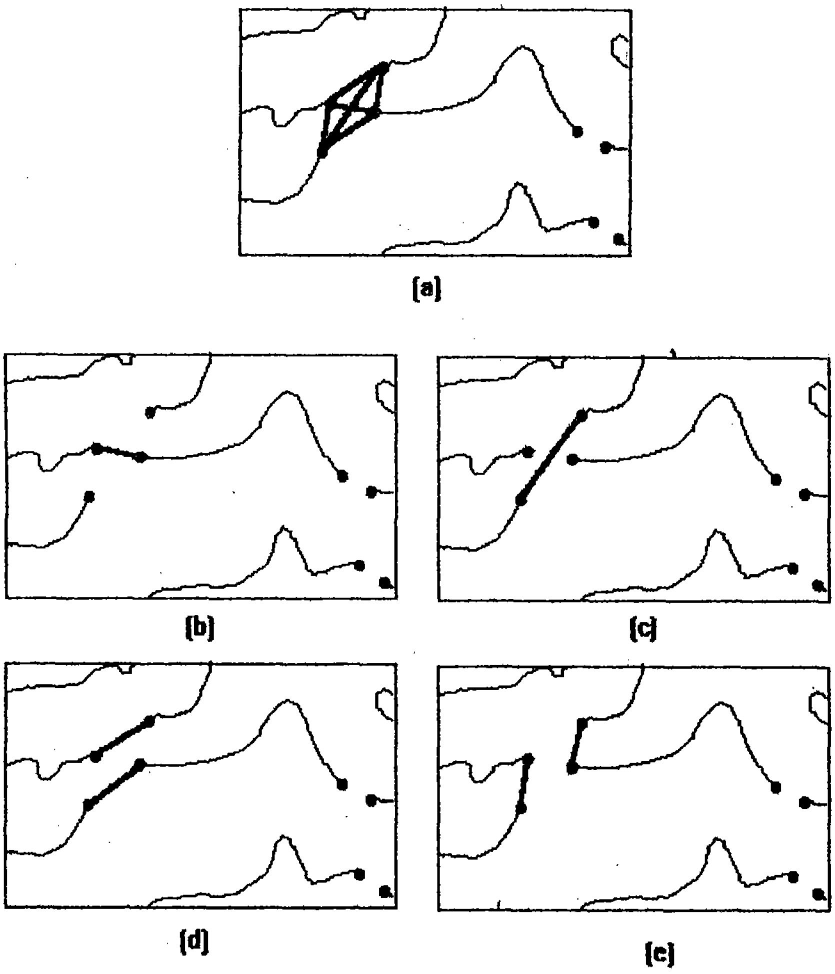

Figure 6. Illustration of the Contour Linking Algorithm

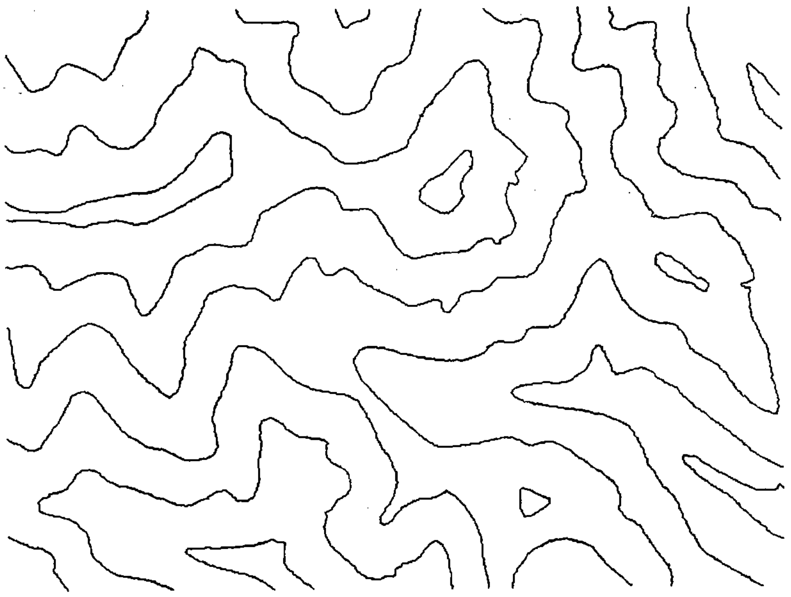

Figure 7. Reconstructed Contour Lines

4. Interpolation between the contours

4. 1 Constructing the containment graph

It is not trivial to assign heights to the contour lines. If two adjacent contours have heights h1 and h2, all three relations h1 > h2, h1 < h2, and h1 = h2 are possible, and automatic assignment is impossible. In this work, height information is given interactively.

A contour is considered as a vertex in the containment graph. If the relation h1 < h2 is entered for the contours c1 and c2, a directional edge from c1 to c2 is constructed. Contour numbers are shown in figure 8.a, and the above relations are shown in figure 8.b. The contours with the same height and adjacent each other are merged to a single node and the containment graph of 8.b is reduced to the containment tree of figure 8.c. If the height of the root is given, other heights are assigned automatically. This containment tree is also used in the interpolation.

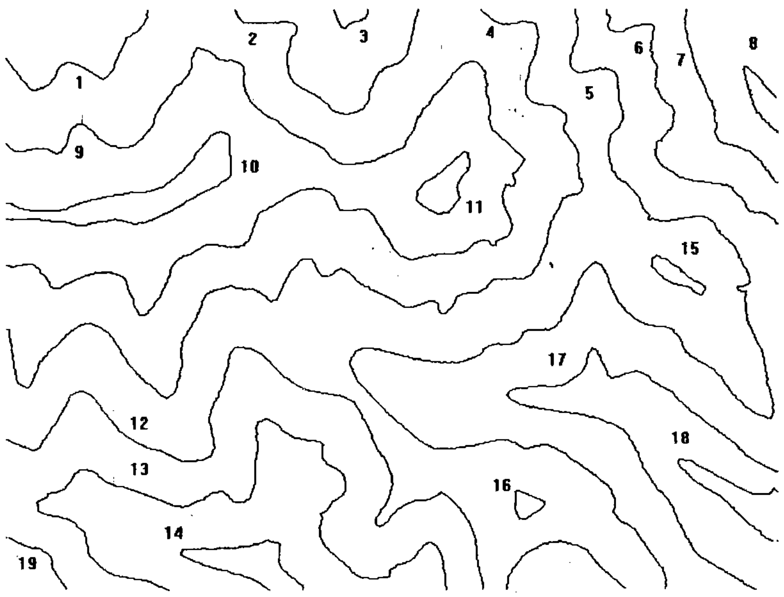

Figure 8.a Contour Numbers

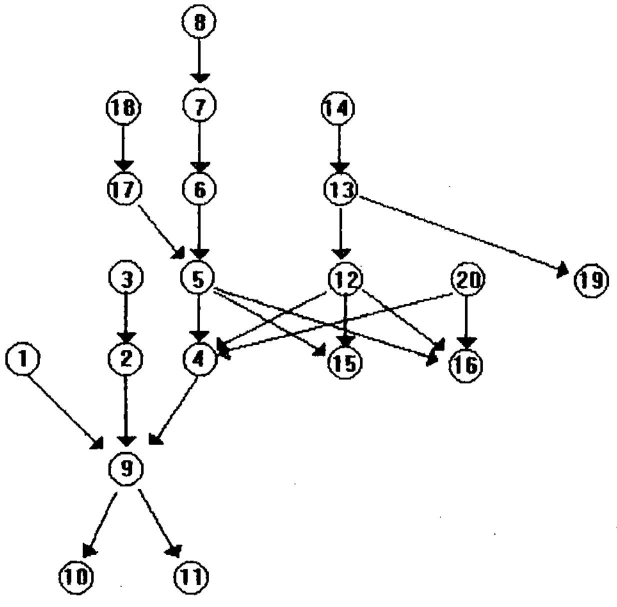

Figure 8.b Contour Containment Graph

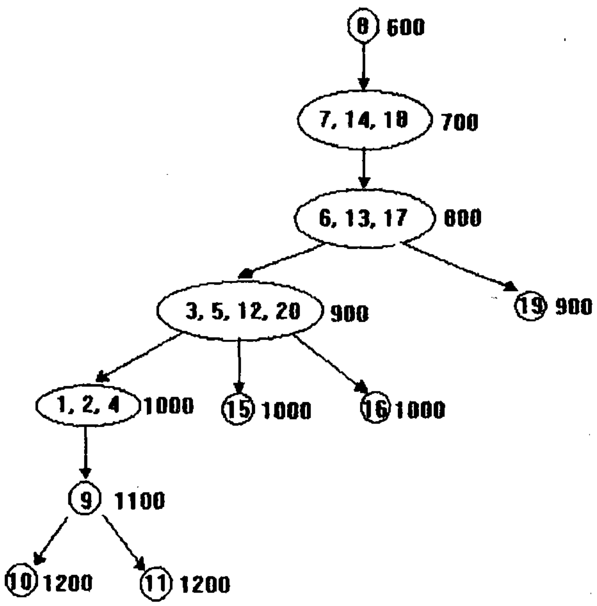

Figure 8.c Reduced Containment Tree

4. 2 Interpolation method

To complete the digital description of the terrain information, heights are to be defined at the lattice points. But the heights are defined only on the contour lines, so the heights at the lattice points should be calculated by interpolation. To find the height at the point X, we search two adjacent contours c_1 and c_2 with different heights h_1 and $h_2 (h_1 < h_2)$, and calculate the height according to the following equation:

$$h = h_1 + \Delta * d_1/(d_1+d_2),$$

where Δ is height difference per contour step, and d_1 and d_2 are distance from X to c_1 and c_2, respectively.

To find the two contours adjacent to point X, search the nearest contour. If it is contour 5 in figure 8.c, for example, decision is made whether the point X is contained in contour 5. The decision is made by the winding method(Harrington,1987). If the point X is contained in the contour 5, d_2 is the nearest distance to the contour 1,2,4,15,16, and $h_1=900$ and $h_2=1000$. If the point X is out of the contour 5, d_1 is the nearest distance to the contour 6,13,17, and $h_1=800$ and $h_2=900$.

If the point X is at the outside of the contour 8 or at the inside of the contour 11 or 12, the above equation cannot be used. In the first case, if we denote the distance from X to the contour 8 as d_1, and to the contour 7,14,18 as d_2, the height of the point X can be calculated by the following equation.

$$h = h_1 - \Delta * d_2/(d_2-d_1)$$

In the second case, the region in the innermost contour is interpolated using the cubic Bezier curve(Rogers and Adams,1989) as shown in figure 9.

The final result is shown in figure 10. Figure 10.a displays color coded DEM, and 10.b shows the relief map and 10.c shows the wireframe display of the DEM.

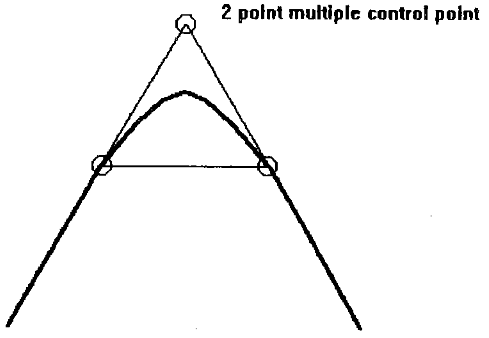

Figure 9 Interpolation by Cubic Bezier Curve

5. Conclusion

In this paper several useful algorithms were developed. Graph theory is used intensively in the course of constructing DEM from 2D terrain map. The contour segments were constructed from the longest path in the spanning tree of the closely located contour pixels.

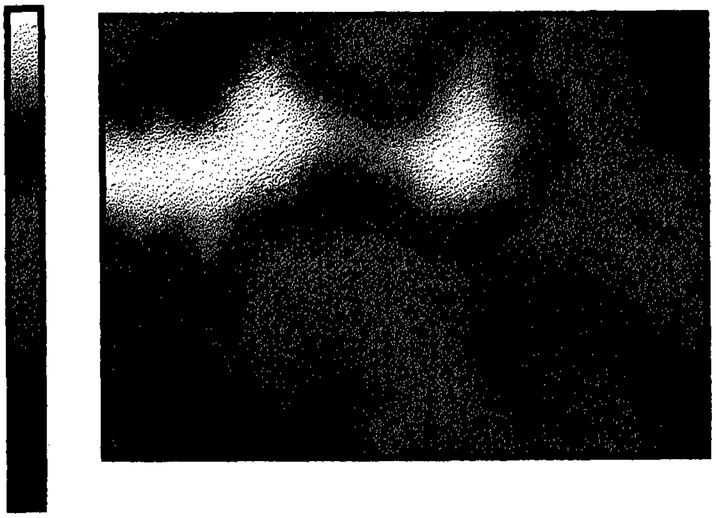

Figure 10.a Color coded DEM

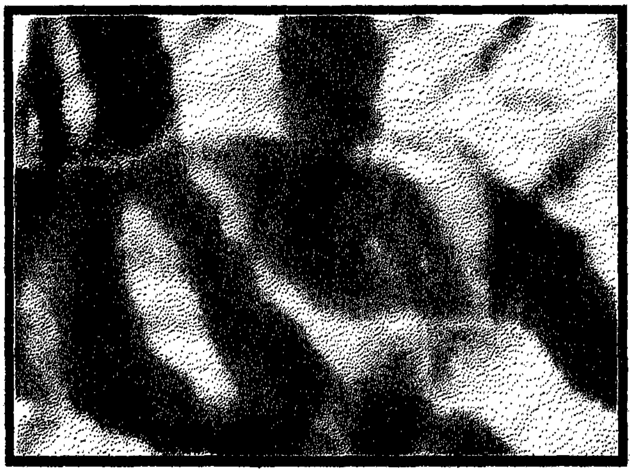

Figure 10.b Relief Map

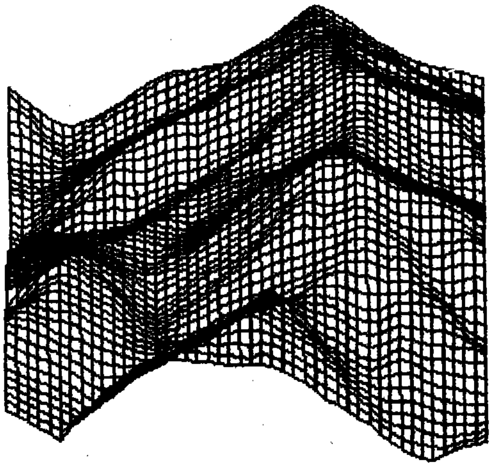

Figure 10.c Wireframe Display of DEM

An optimal segment linking algorithm which minimizes cost function were developed and tested using the real data, and were shown to be effective. The containment graph were constructed and used in height assignment to contours and interpolation.

References

C.T.Zahn, 1971, "Graph theoretical methods for detecting and describing gestalt clusters".

D.F.Rogers and J.A.Adams, 1989, Mathematical Elements for Computer Graphics, McGraw Hill.

E.Horowitz and S.Sahni, 1978, *Fundamentals of Computer algorithms*, Computer Science Press.

J.T. Tou and R.C. Gonzalez, 1974, *Pattern Recognition Principles*, Addison-Wesleys.

S.Harrington, 1987, *Computer Graphics*, McGraw-Hill.

S.Park et al, 1990, "A study on the construction technique of DEM using a commercial map", *J. of The Korean Society of Remote Sensing*, 6-1.

S.Viseshin and S.Murai, 1990, "Automated Height Information Acquisition From Totographic Map", Proc. of the IAPR Workshop on Machine Vision Applications.

T.Y.Zhang and C.Y.Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns", *Comm. ACM*, 27-3 : .236-239.