M. Tech. (Computer Science) Dissertation Series

# Skeletonizing Binary Images using Neural Networks

**a dissertation submitted in the partial fulfilment of the requirement for the M. Tech. ( Computer Science ) degree of the Indian Statistical Institute**

*By*

Manish Bhaumik

**under the supervision of**

Amitabha Dutta
CSSC Unit , I.S.I.

INDIAN STATISTICAL INSTITUTE
203, Barrackpore Trunk Road
Calcutta - 700 035

STATISTICAL INSTITUTE LIBRARY
CALCUTTA

# Certificate Of Approval

This is to certify that the dissertation work entitled "Skeletonizing Binary Images using Neural Networks" submitted by Manish Bhaumik, in partial fulfilment of the requirements for M.Tech. in *Computer Science* degree of the ***Indian Statistical Institute*** is an acceptable work for the award of the degree.

Date :

( Supervisor )

S. Chattopadhyay
( External Examiner )

## Acknowledgements

I would like to acknowledge my gratitude to Prof. Amitabha Dutta ( Computer Science and Statistical Service Unit ) for providing me invaluable support through out this project. During the course of this project he has always given me helpful suggestions whenever I was in trouble . Finally I would like to thank all my classmates , specially Bani Kumar Maity , K.Rajeshbabu and Subhasis Bhattacharjee, for their support throughout this project.

Manish Bhaumik .

# Contents

## *ABSTRACT*

The project aims at developing an incremental neural network model which is able to learn a *'stick figure'* of a given image . Some of the ideas are similar to those in Growing Neural Gas model by Bernd Fritzke which learns the topologies of a data distribution P(ς) . At first , it has been shown that a different criterion for growth of the network ( which we have called Vote Counting Method) has the same performance as Growing Neural Gas model. Then we have introduced a different approach to growth which learns a *'stick figure'* of a given image .

keywords: Neural net ; Stick figures ; Binary Image ; Skeleton ; Delauny triangulation

# 1.Introduction

Artificial neural network models have been of great interest for several years in various areas like Pattern Recognition , Computer Vision , Image Processing . Extracting the skeleton of a binary image is an important topic of Image Processing . In this work , we have proposed a neural network model that is able to extract our desired form of skeleton ( *'Stick figures'*) of a binary image .

Topology learning is a special field of interest in neurocomputing . Given a high-dimensional data distribution P(ς) , topological learning means finding a topological structure which closely reflects the topology of the data distribution . Various methods have been used for topology learning .

Two remarkable methods are

a. *Competitive Hebbian learning*(CHL) proposed by Martinez[3] in 1993
b. *Neural Gas method* proposed by Martinez and Schulten[4] in 1991

In CHL a number of neurons are chosen in $R^n$ first . Then for each input signal s the two neurons which are the closest and second closest to the signal are connected by an edge , thus producing a subgraph of the *Delauny triangulation* . This subgraph is called *induced Delauny triangulation* . The *induced Delauny triangulation* is restricted to those areas of the input space $R^n$ where $P(\varsigma) > 0$ and thus reflects the topology of the data distribution P(ς) .

The neurons which are distant from the region of the input space where $P(\varsigma) > 0$ are not useful in topology learning and are often called *dead units* . They can be used in topology learning if they can be gradually moved towards the areas of the input space where the signals come from . For this purpose Martinez and Schulten introduced the *Neural Gas method* which is as follows:

For each input signal s move the k nearest neurons towards the signal where k decreases from a large initial to a small final value .

In the discussion we have repeatedly used the term *'stick figure'* . By *'stick figure'* we mean a kind of skeleton giving an idea about the shape of an image e.g. the *'stick figure'* of an image of a one dimensional object is the image itself , for two dimensional object a *'stick figure'* is the one dimensional skeleton (obtained by neglecting the insignificant dimension) and the two dimensional skeleton for an image of a three dimensional object .

The Growing Neural Gas method[5] which is very efficient in topology learning is discussed in a brief next .

## 2. Growing Neural Gas Method

The network consists of :

- a set A of units(or nodes) . Each unit $c \in A$ has a *reference vector* $w_c \in R^n$ associated with it .
- a set of edges among pairs of units . A counter *age* is associated with each link .

The parameters of the network are MaxAge ,$\in_b$, $\in_n$, $\lambda$ , $\alpha$ ,d .

The algorithm is

1. Start with two neurons at random positions $w_a$ and $w_b$ in $R^n$.
2. Generate an input signal $\varsigma$ according to $P(\varsigma)$ .
3. Find the nearest unit s1 and second nearest unit s2 .

4. Increment the age of all the edges emanating from s1 .

5. Add the squared distance between s1 and $\varsigma$ to the error counter of s1 .

6. Move s1 towards the signal s by $\in_b$ times the distance between s1 and $\varsigma$ and all

direct neighbours k of n by $\in_n$ times the distance between $\varsigma$ and k .

7. If s1 and s2 are connected by an edge , set the age of this edge to zero . If such

an edge does not exist, create it .

8. Remove edges with an age greater than *MaxAge* . If this results in points having no

emanating edges, remove them as well .

9. If the number of input signals generated so far is an integer multiple of $\lambda$,

   a. Find the unit M with the largest accumulated error and its neighbour m with largest error variable .

   b. Insert a neuron q midway between M and m .

   c. Remove the link between M and m .

   d. Insert a link between M and q , as well as , between q and m .

   i. Decrease the error variable of both of M and m , multiplying them by a constant $\alpha$ .

   f. Set the error variable of q equal to that of M .

10. Decrease all error counters by multiplying them with a constant d .

11. If a stopping criterion (e.g. net size or some performance measure) is not yet fulfilled, go to Step 2 .

In this method, it is assumed that input signal $\varsigma$ is generated obeying some unknown probability density function $P(\varsigma)$ . Step 1 is basically creating the initial network which grows dynamically depending on the signals generated . In step 6 the network is adapted by moving the neurons towards the region of the input space where the signals come from (i.e., the region where $P(\varsigma) > 0$ ) . After adapting the network $\lambda$ times , a new unit is added to the network midway between the unit with the largest error and its neighbour with the largest error (Step 10 ). The unit with the

the largest error is chosen since it is , most probably , the unit which was either nearest to the signal maximum number of times or adaptation is maximum for this unit . Thus this midway position is likely to be very close to the input space and hence is the ideal position for the insertion of a new unit . An *aging* scheme has been used to remove the edges and the units ( Steps 4 and 8 ) which will have no further role in learning the topology . The fact that those units become second nearest but never nearest to the signal (see Step 4 and 7) over a number of previous iterations ( at least *MaxAge* times ) confirms that they will have no further role in proper growth of the network . Step 7 also creates a single connection of the '*induced Delauny triangulation*' . Step 11 is basically introduced to avoid floating point overflow .

## 3.Vote Counting Method

Here a local counter *vote* is attached with every edge , instead of the local error counter used in Growing Neural Gas Method . Whenever a signal is generated , the vote associated with the edge nearest to the signal is incremented . At the time of insertion , the edge with the maximum vote is selected a neuron is inserted in at the midpoint of the edge . Thus the algorithm is almost same as Growing Neural Gas method except the necessary modification to step 4 and step 10 of Growing Neural Gas algorithm .

## 4. δ_insertion method

So far we were concerned with incremental neural networks which learn topology of a data distribution $P(\varsigma)$ . In this method , a totally different approach to growth is adopted with an aim to get a ***'stick figure'*** of a given image wherein the signals are simulated by randomly choosing a point on the image .

The network consists of :

- a set of units (representing neurons) . Each unit has an associated *reference vector* which may be regarded as position in input space of the corresponding unit .
- a set of links among pairs of units . A counter *age* is associated with each link .

The parameters of the network are MaxAge ,$\in_b$, $\in_n$,$\in$, $\delta$.

The proposed algorithm is :

1. Generate two signals which are basically two random points $w_a$ and $w_b$ on the image . Add two neurons with reference vectors $w_a$ and $w_b$ to the network .
2. Let prevNet be the current network and maxLimit = 20 $\times$ size of the network , count = 0 .
3. Increment count .
4. Generate a signal which is a randomly chosen position s on the image .
5. Find the unit n nearest to the signal and the unit N second nearest to the signal .
6. Increment the age of all the links outgoing from n .
7. Move n towards the signal by $\in_b$ times the distance between s and n and all direct neighbours k of n by $\in_n$ times distance between s and k .
8. If there exists an edge between N and n , set the age counter of that edge to zero .

   Otherwise insert an edge between N and n .
9. Remove links with an age greater than *MaxAge.*
10. Remove isolated neurons .
11. If count is less than maxLimit go back to Step 3 .
12. If the sum of the distances between the positions of the individual neurons in prevNet and their current positions is greater than $\in$ go back to Step 2 .

13. Find the unit M and its neighbour m such that the length of the link between them is maximum and greater than $\delta$. If such units are not found , go to Step 2 .
14. Insert a neuron q midway between M and m .
15. Remove the link between M and m .
16. Insert a link between M and q , as well as , between q and m .
17. If the network didn't grow further in last (200 * size of the network iterations) , stop .
18. Go back to step 2.

In this method , Step 1 creates the initial network . Then a signal is generated (Step 4) and the network is adapted (Step 7) maxLimit times . Then the sum of the shifts in the positions of the individual neurons over the adaptation period is measured (Step 12) and if it is less than $\in$ (i.e. the network is stable w.r.t. adaptation and therefore more neurons are required for further growth) we go for adding another neuron to the network (Step 13-16) . In case the sum is greater than $\in$ , we hope that the network may improve by adaptation only and so we go back to Step 2 and start another adaptation phase . We use the same aging scheme as in Growing Neural Gas method for removing undesirable links . Since the network is designed to learn a *'stick figure'* of the image , the position midway between the neighbouring neurons with spacing greater than or equal to $\sigma$ and minimum age of the connecting edge is chosen for adding a new neuron . The minimum age edge is chosen to avoid insertion on an edge which is likely to be removed in near future .

# 5. Simulation Results

We have implemented the *Growing Neural Gas method* , *Counting vote method* and *$\delta$-Insertion method* and applied them on various images to extract their shapes. The outputs we obtained are shown below . We shall mainly concentrate on results obtained using $\delta$-Insertion method , since our primary interest is there .

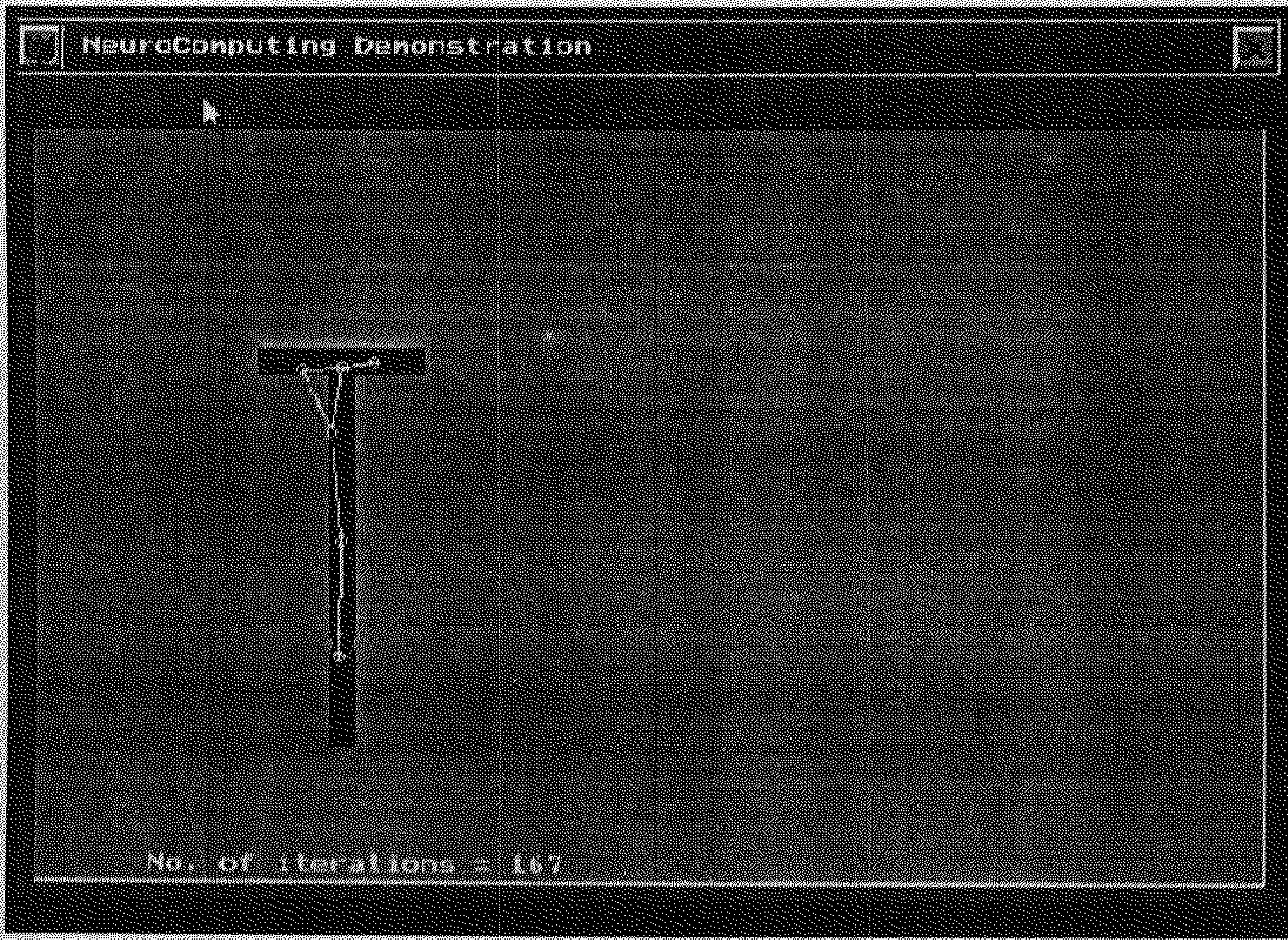
### 5.1. Platform used

The Software developed for testing the δ-Insertion method is developed on MS_DOS using TC++ and portable to MS_WINDOWS also .

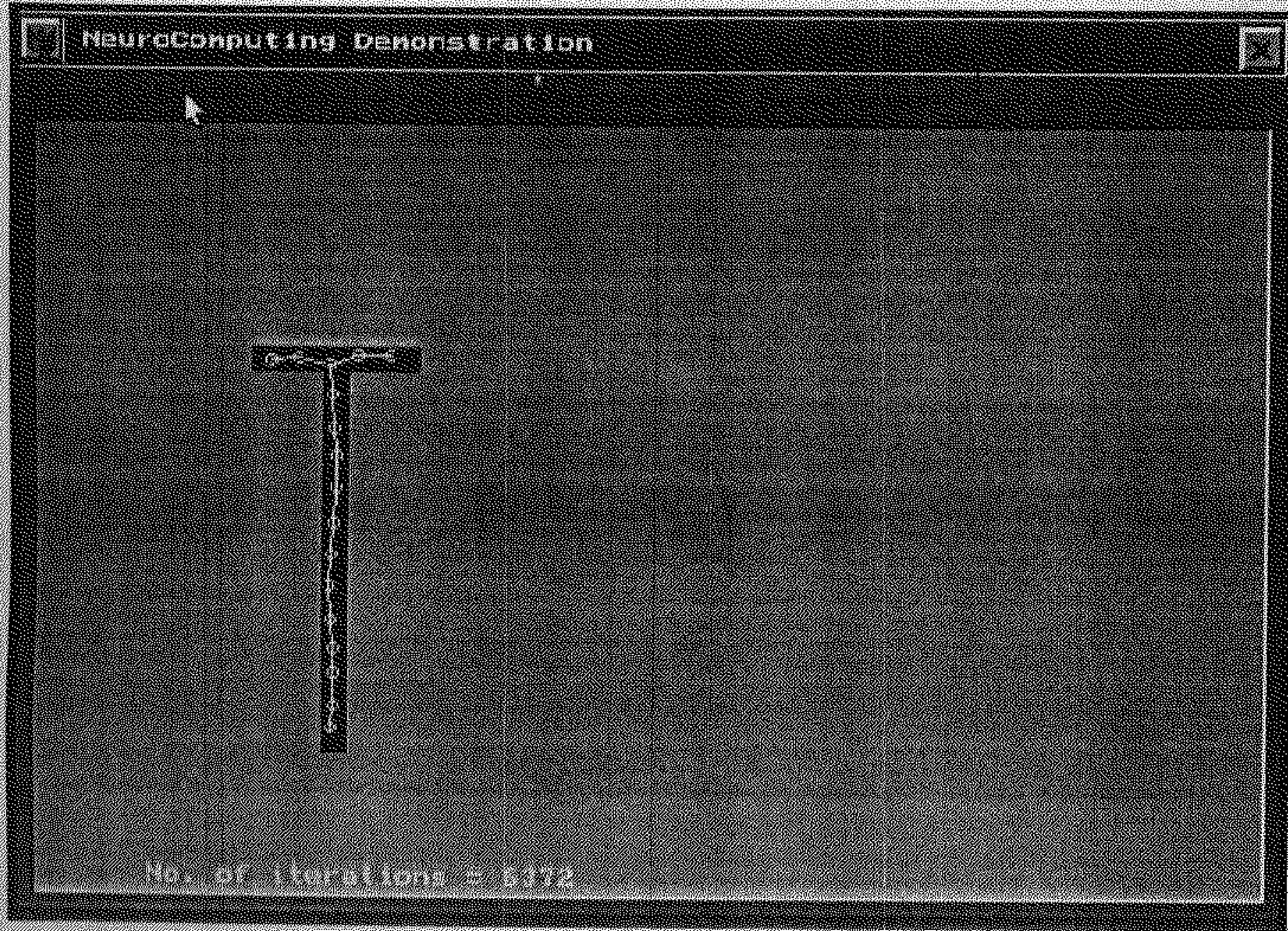
### 5.2. Results Obtained

In the outputs shown below , the image consisting of circles and straight lines joining them represents the network . The circles represent the *neurons* and the straight lines joining them represent the *links* . The background image on which the network rests is the actual image whose *'stick figure'* has to be generated by the neural network . The neural network has been shown on the given image whose shape has to be extracted to compare the net with the actual skeleton .

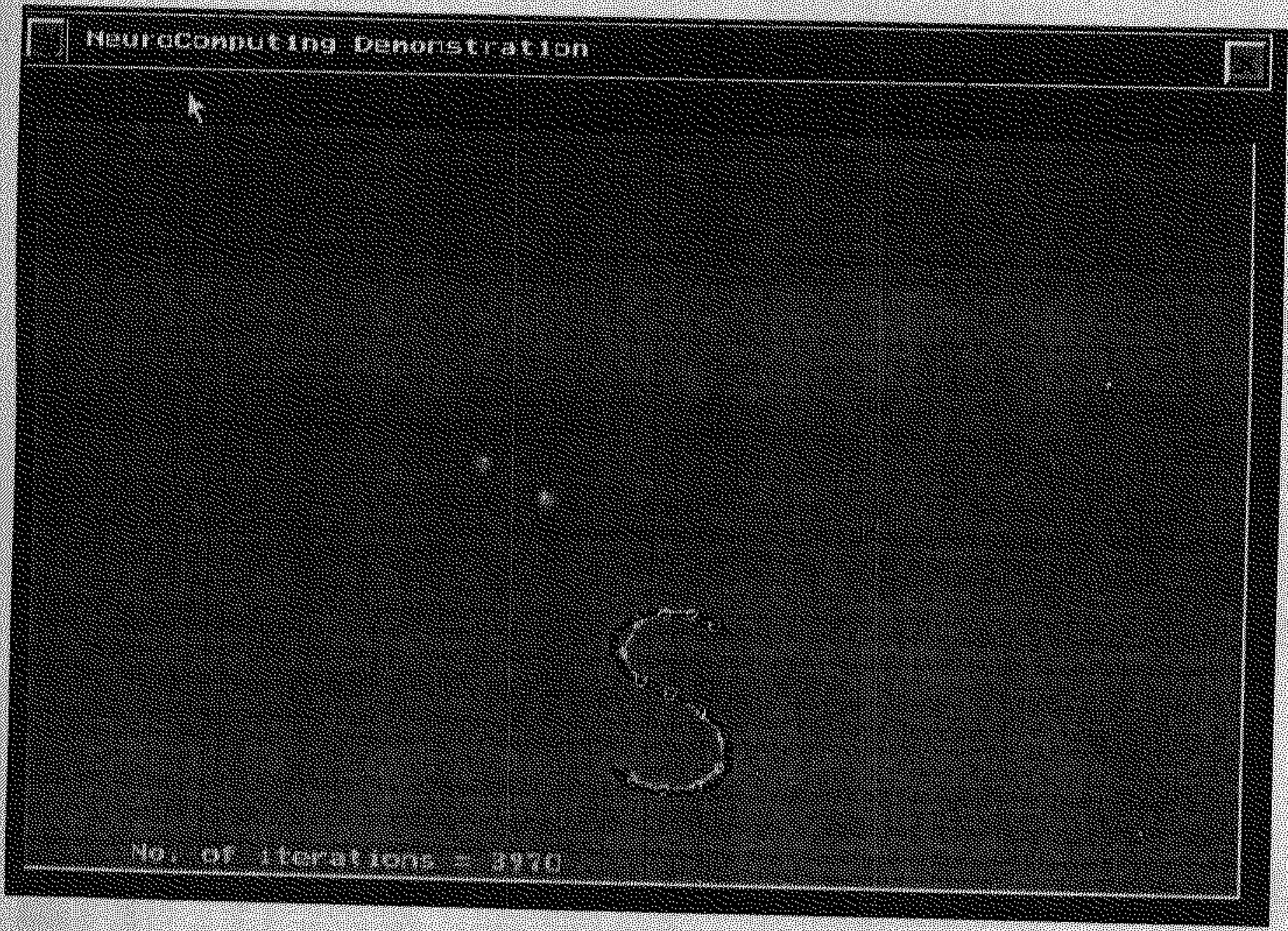
The following figures show the outputs obtained by applying the δ-Insertion method with the parameter values $\in = 100$ , MaxAge = 20 , $\in_b = .1$ , $\in_n = .01$ , $\delta = 20$ on an image which looks like a thick T .




The final network is shown below .

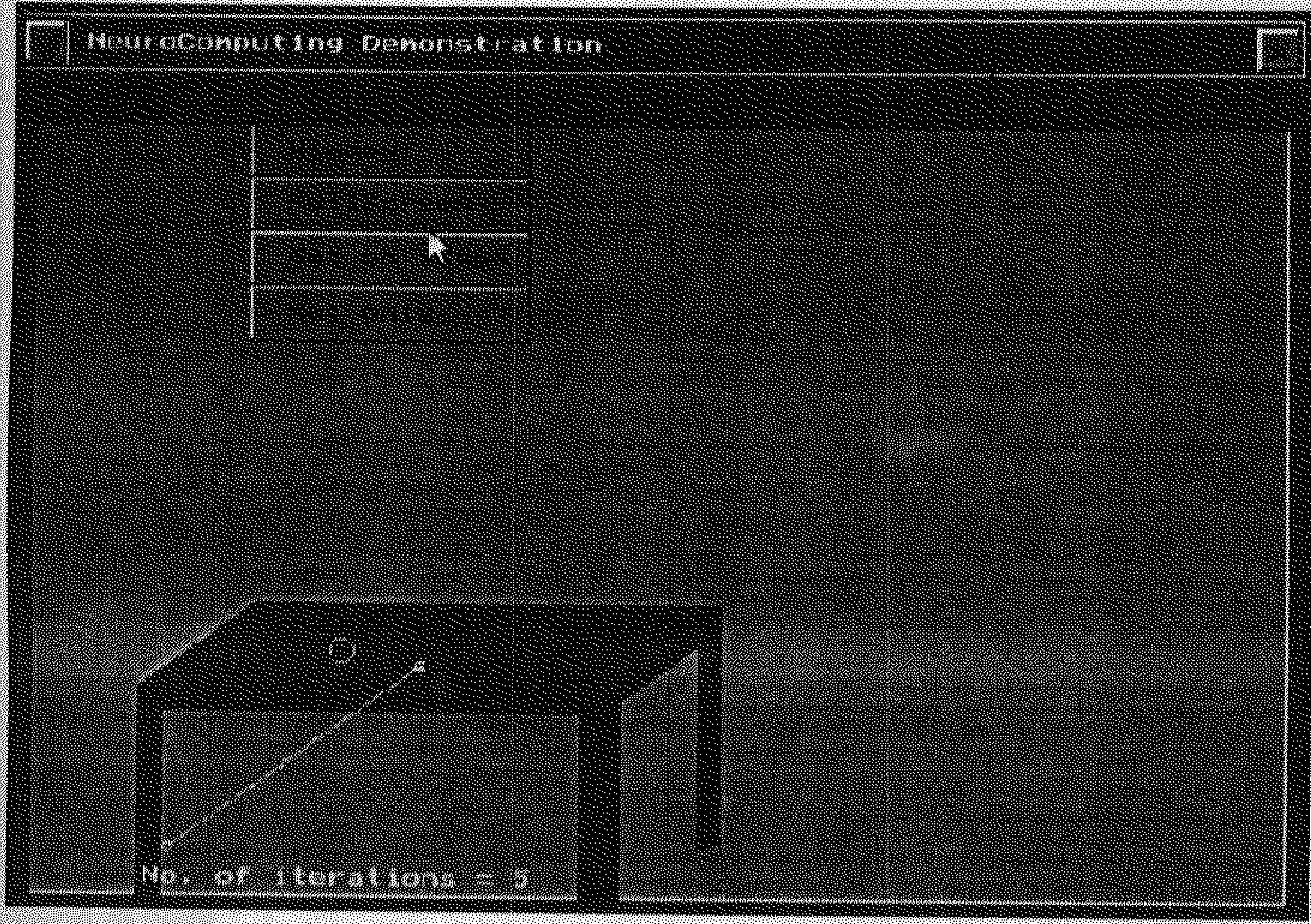


The following figures show the outputs obtained by applying the δ-Insertion method with the parameter values $\in = 100$ , MaxAge = 20 , $\in_b = .1$ , $\in_n = .01$ , $\delta = 20$ on an image which looks like a thick *S* .

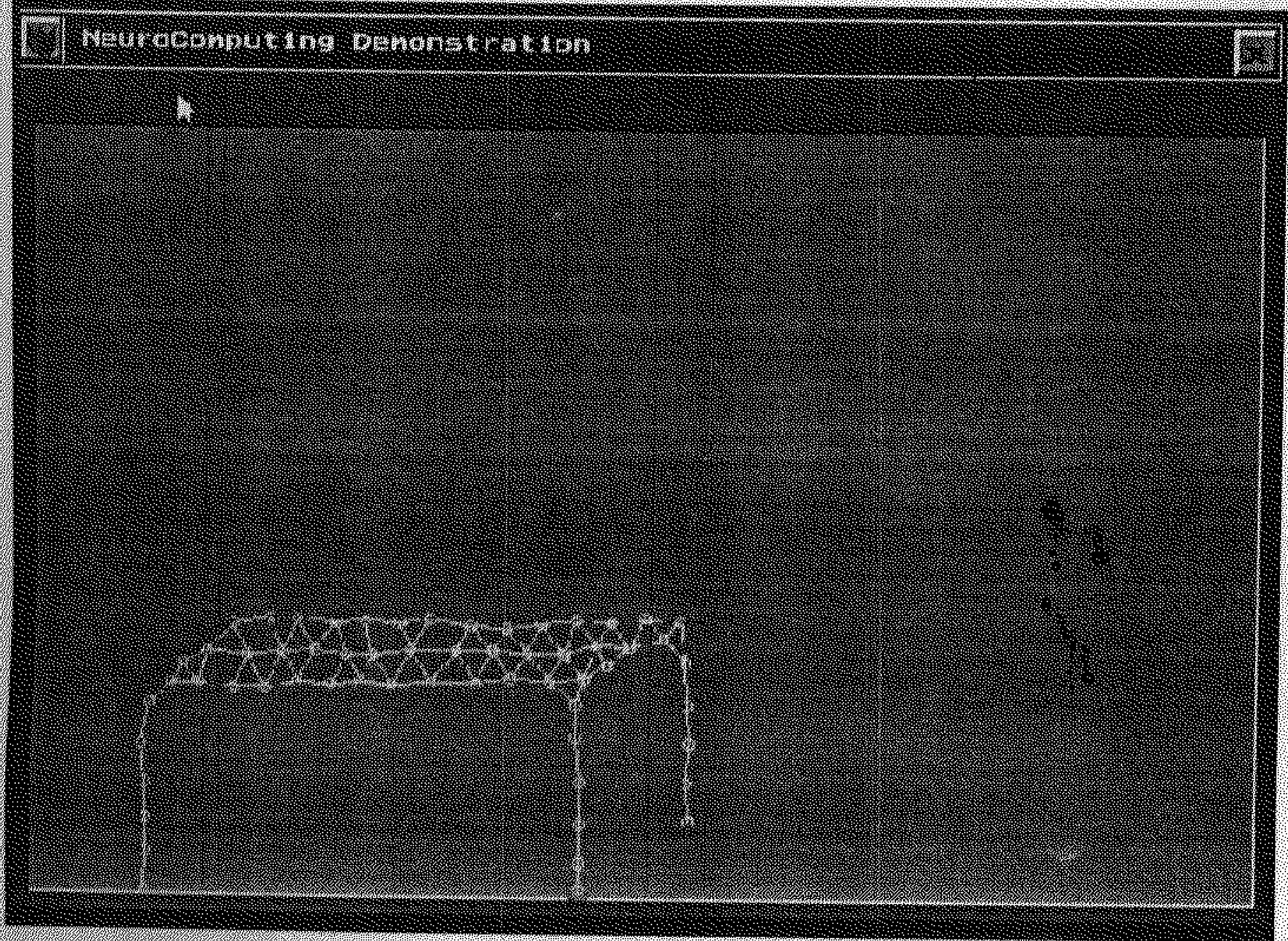



The following figures show the outputs obtained by applying the δ-Insertion method with the parameter values $\in = 200$ , MaxAge = 20 , $\in_b = .1$ , $\in_n = .01$ , $\delta = 20$ on an image of a 3D table .

The following is the network after 5 iterations only .



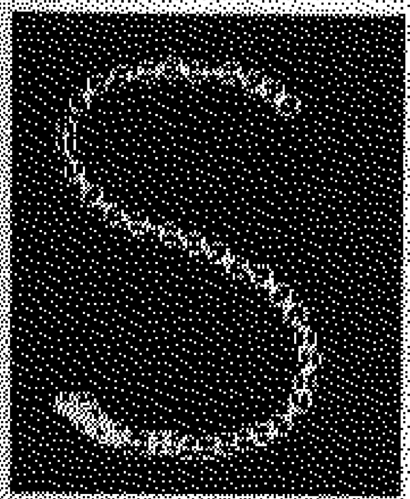

The final network generated is shown below .



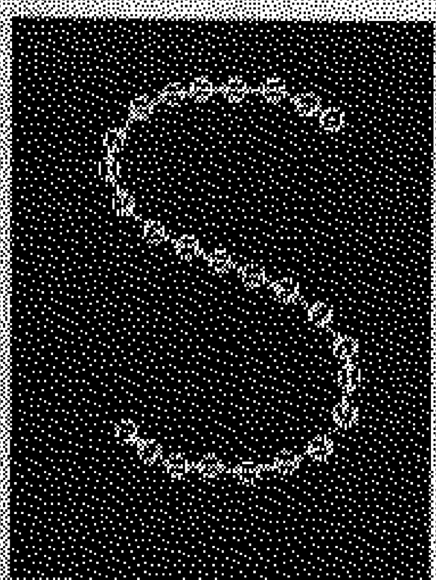
## 6. Discussions

The experimental results shown in section confirms the fact that the δ-Insertion model is efficient enough to extract the *'stick figure'* of a binary image . We show below the *'stick figure'* s of a image generated by δ-Insertion method for various values of δ . The other parameters are $\epsilon = 200$ , MaxAge = 20 , $\epsilon_b = .1$ , $\epsilon_n = .01$ in all the cases .

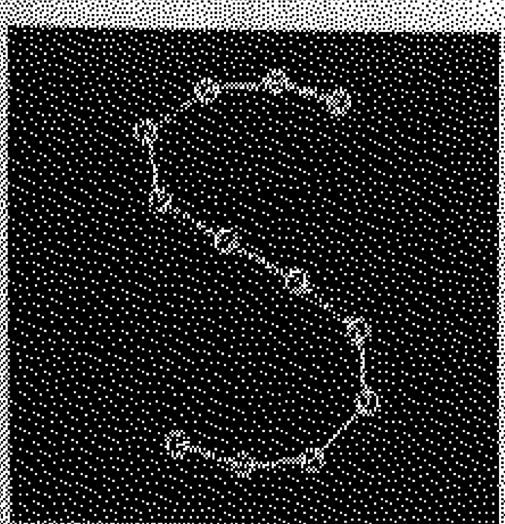
Case 1: $\delta = 0$



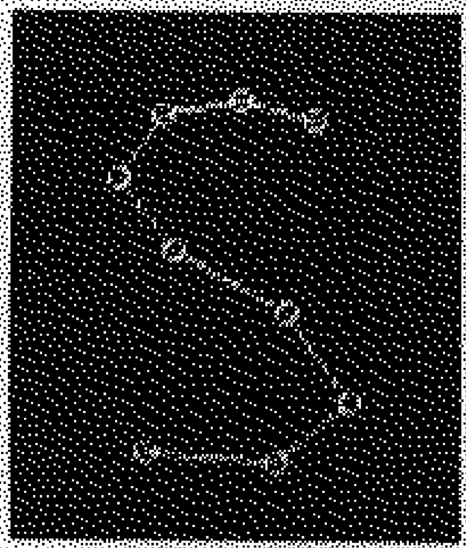
Case 2: $\delta = 10$



Case 3: $\delta = 20$

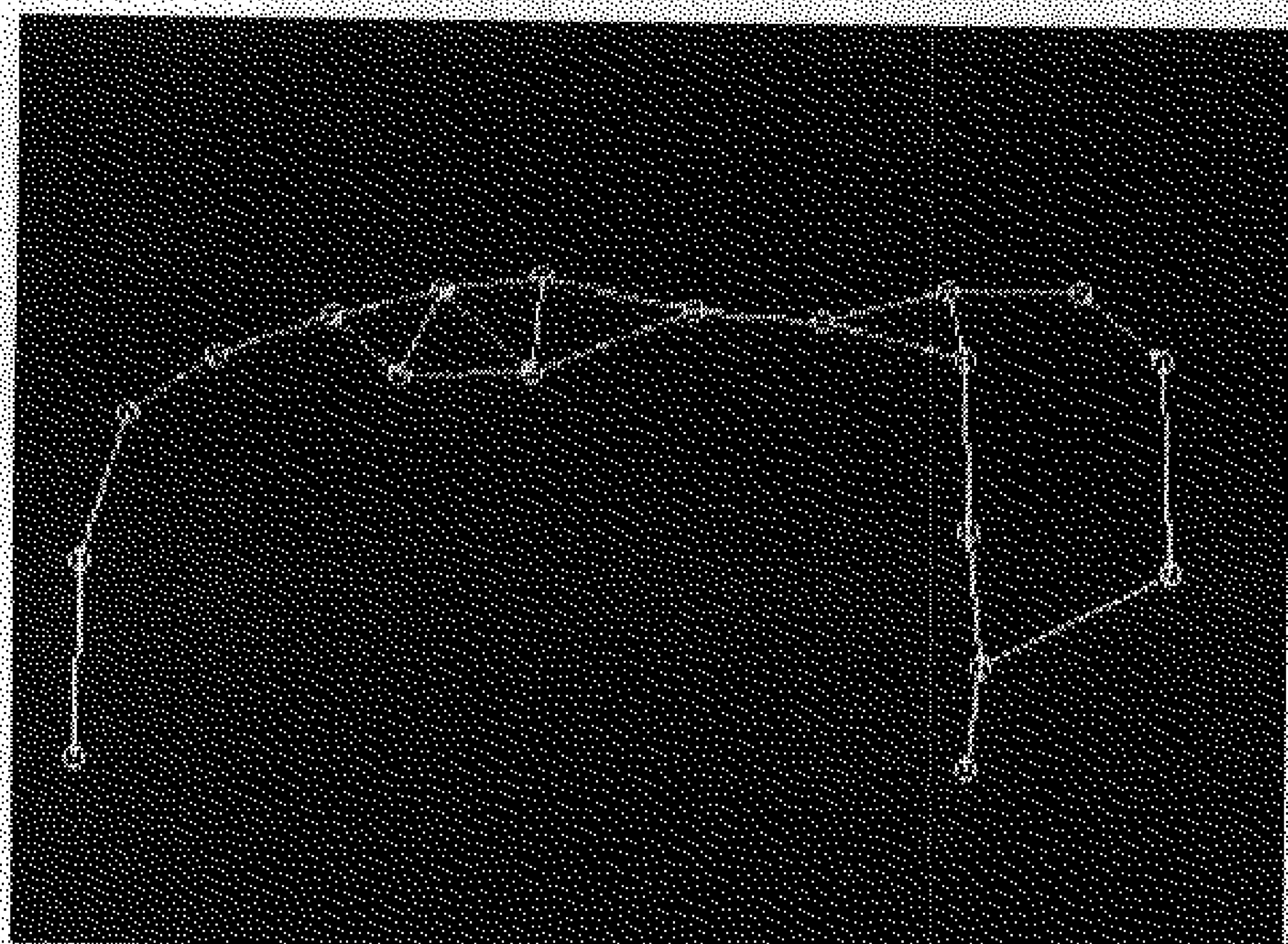


Case 4: $\delta = 30$



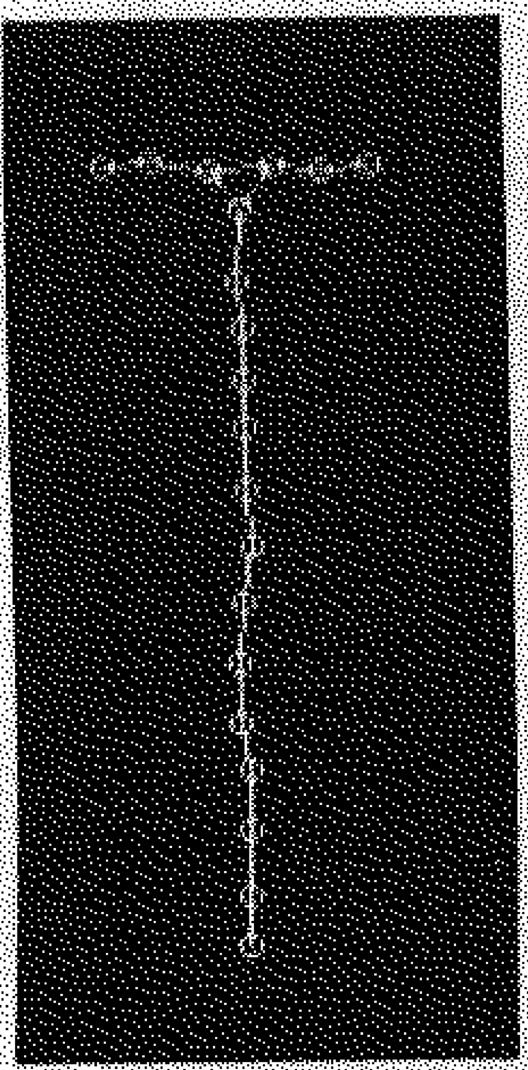
Almost the exact topology is generated in Case-1 and the net becomes closer to the skeleton as $\delta$ increases and the best skeleton is obtained in Case-2. As $\delta$ increases further , the net starts deviating from the skeleton . To obtain a good skeleton , we have to choose $\delta$ properly .

In cases where the net has to spread enough as in case of the image of a table , we have to choose a large value for $\in$ . Otherwise the net may not be able to grow e.g. the net learning the *'stick figure'* of the image of a table is



As discussed in the first paragraph of this section , the exact topology is generated for $\delta = 0$ . Thus the $\delta$-Insertion model generates the same output as the Growing Neural gas model for $\delta = 0$ . Therefore $\delta$-Insertion model may be termed as the ***Generalised Growing Neural Gas Model*** .

Although the method is efficient in most of the cases , it fails to generate a good skeleton in some cases due to wrong choice of parameters e.g.



Several heuristics may be introduced to avoid such situations . For example , the heuristic

*If there exists three neurons x , y , z such that*

1. *x is neighbour to y and z .*
2. *y and z are neighbours .*
3. *There exists a neighbour of y such that the angle at y made by the neighbour and z is almost equal to zero and there exists a similar neighbour of z .*

*then*

a. *insert a neuron w midway between y and z .*
b. *connect w and x by an edge .*
c. *remove the links between x and z and between y and z and between x and y .*
d. *connect w and y , w and z by edges .*

will be able to avoid the undesirable situation shown above . Thus further improvements can be made to make the σ-Insertion model more efficient to in extracting the shape of an binary image .However a heuristic may improve the efficiency for a particular type of images , as well as , decrease for others . So one must be careful in adding heuristic to the model .

# 7.Conclusion

Extracting skeleton has usefulness in Character Recognition and Data Reduction. So various methods has been devised to extract the skeleton [6-8] . In this work , a have introduced a neural network technique to do that .

# References


**1.**Y.H.Pao , Adaptive Pattern Recognition and Neural Networks (Addison-Wesley , 1989) .

**2.**Amitabha Dutta , T.Pal , S.K.Parui , A modified self-organising neural net for shape , Neurocomputing 14(1997) 3-14 .

**3.**Martinetz T.M , Competitive Hebbian learning rule forms perfectly topology preserving maps .In ICANN '93 : Pages 427-434 .

**4.** Martinetz T.M , "Neural Gas" network for vector Quantization and its application to Time_Series Prediction , IEEE Trans Neural Network , Vol 4 , 558-569.

**5.**Bernd Fritzke , A Growing Neural gas Network Learns Topologies , IEEE Trans Neural Network , 1994 .

**6.**R.W.Smith , Computer processing of line images : a survey , Pattern Recognition 20(1987) 7-15 .

**7.**L.Lam , S.W.Lee and C.Y.Suen , Thinning methodology - a comprehensive survey . PAMi 14(1992) 869-888 .

**8.**A.Dutta and S.K.Parui , A robust parallel thinning algorithm for binary images. Pattern Recognition 27(1994) 1181-1192 .