A Study On Shock Graphs
Graph representation of objects

by

Desai Abhi Hitesh
201111047

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

MASTER OF TECHNOLOGY
in
INFORMATION AND COMMUNICATION TECHNOLOGY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

July, 2013

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Master of Technology in Information and Communication Technology at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

Desai Abhi Hitesh

Certificate

This is to certify that the thesis work entitled SHOCK GRAPHS: GRAPH REPRESENTATION OF OBJECTS has been carried out by DESAI ABHI HITESH for the degree of Master of Technology in Information and Communication Technology at *Dhirubhai Ambani Institute of Information and Communication Technology* under my/our supervision.

Prof. Aditya Tatu
Thesis Supervisor

Prof. V. Sunitha
Thesis Co-Supervisor

Acknowledgments

I would like to express my deepest gratitude to all those who provided me with the opportunity for this thesis work. First of all, I am grateful to my thesis guide Prof. Aditya Tatu for giving full support, guidance and encouragement, and bearing with me for one whole year with patience. Second of all, I would like to thank my thesis co-guide Prof. V. Sunitha for her support and guidance.

I would like to thank my thesis examiner Prof. Rahul Muthu for his support. His quick, witty oneliners made the thesis presentations more fun-occupied, than terrifying.

I express special gratitude towards DA-IICT and all the faculty members, for creating such a nice environment to study, work and grow intellectually. The past two years helped me in number of ways to develop myself as an individual.

I am thankful to all the authors and publications, which helped me in enhancing my knowledge and understanding about my work.

I thank my parents for their love, trust and support they have shown. I would also like to thank all of my friends, for their continuous support and encouragement.

Contents

Abstract

A shock graph is an abstraction of a two-dimensional shape of an object. It represents the shape as a graph, using its boundary information. In this thesis, we study the shock graph representation of two-dimensional shapes. We discuss a method to compute an approximation of shock graph. We discuss different thresholds we need to set for shock graph computation, and some solutions on how we can get rid of them in some cases. Next, we see the results showing some unique properties of shock graph. And finally, we discuss how we can use shock graphs in different applications, and for further research.

List of Tables

List of Figures

CHAPTER 1

Introduction

Object recognition in computer vision is the task of finding a given object in an image or video sequence. We, humans recognize a multitude of objects in images with little effort, despite the fact that the image of the objects may vary somewhat with view points, different sizes/scale or other transformations like translation and rotation. They can even be recognized when they are partially obstructed from view. This task is still a challenge for computer vision systems, in general.

The backbone of any recognition system is its model object representation, for it governs what kinds of features are extracted, how features are grouped, and how features are matched to models. The first and foremost thing is to select the method for representing the objects. There are intensity-based approaches, which take into account the intensity values and their variations. There are shape-based approaches where boundary features are taken in to account.

In this work we are interested in object recognition of 2D objects based on their shape. This information can be extracted from the object boundary. As we will see, shock graphs extract a good amount of information from the two-dimensional boundary of objects. And due to some interesting features of shock graphs, it makes them helpful in object recognition problem for some special cases.

Let us first discuss some boundary representation methods. Here, by object, we mean two-dimensional representation, projection or image of a three-dimensional thing which can be seen, felt or perceived.

1.1 Two$-$Dimensional Object Representation

The three most common ways of representing an object boundary are:

1.1.1 Boundary Represented as Curves

A 2D curve C in $\mathbb{R}^2$ is a map $C : (\alpha, \beta) \mapsto \mathbb{R}^2$, for some α,β with $-\infty \leq \alpha < \beta \leq \infty$. The symbol (α, β) denotes the open interval

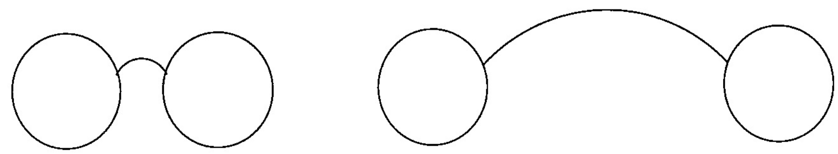


Figure 1.1: A pair of spectacles(left) and an object with same parts but does not look like spectacles(right).

$$(\alpha, \beta) = \{t \in \mathbb{R} | \alpha < t < \beta\}.$$

Object recognition is done by computing the similarities of part structure of two objects, using the approaches that compare local properties of the contours of the shapes(Figure 1.1). The approach is known as elastic matching[1].

Much work has been done by Michor et. al. in [2, 3] where they find similarity between shapes represented by curves by finding the easiest deformation between and by Younes et. al. in [4] where they construct a metric space of all anatomical images from the geodesic connecting one anatomical structure to another in an orbit.

1.1.2 Boundary Represented as Level Sets

Boundary embedded in the zero-level set of the distance function $C = F^{-1}(\{0\})$, $F : \mathbb{R}^2 \mapsto \mathbb{R}$ defined as:

$$F(x, y) = \begin{cases} +d(x, y) & \text{if } (x, y) \in interior(C), \\ -d(x, y) & \text{if } (x, y) \in exterior(C) \end{cases}$$

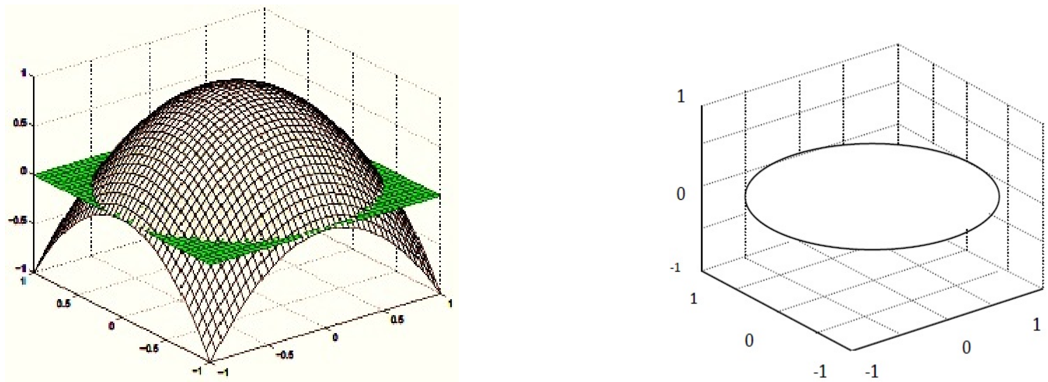
where $d(x, y)$ is the shortest distance between (x, y) and C. With this approach, it is easy to determine the shape of those objects which change their topology(splitting in two, developing holes, or the reverse of these operations)[5](Figure 1.2).

Charpiat et. al. in [6] have used Hausdorff distance to define the metric on different set of shapes, and warping a shape into another by infinitesimal gradient descent, minimizing the corresponding distance (figure 1.3).

1.1.3 Boundary Represented as a Graph

We create a graph, using the boundary of the object. We can compute the skeleton, medial axis and shock graph, using the boundary information of the object:

- A **skeleton** is obtained through a process of "skinning" a polygon such that all edges move with the same speed in a "self−parallel" manner, and each vertex moves along the bisector of its incident edges. Object recognition using skeleton-based graph matching approach has been discussed in [7].



(a) The level set function $(x, y, F(x, y))$ and $(x, y, 0)$.

(b) C: F = 0

Figure 1.2: Level set

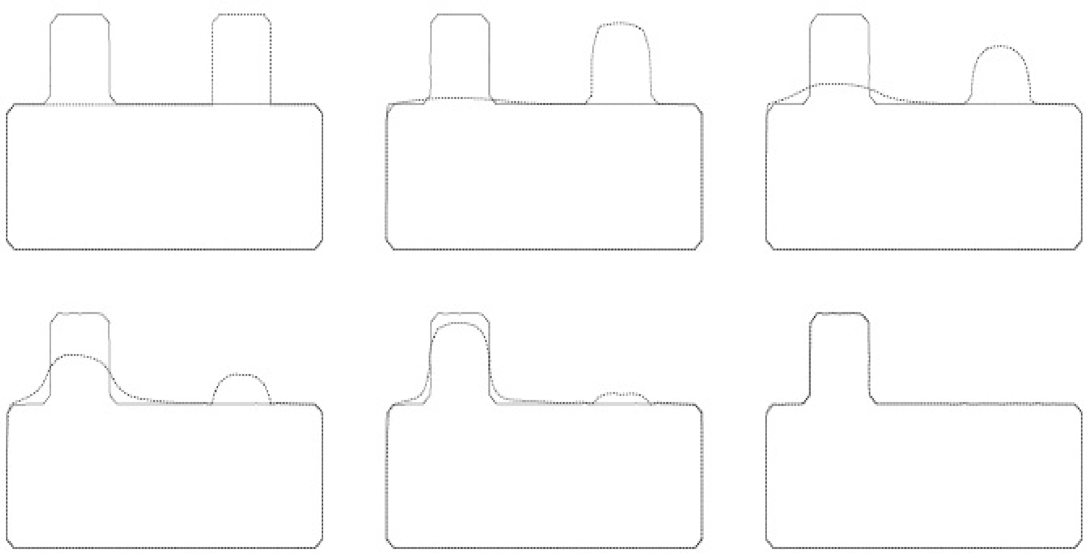


Figure 1.3: Hausdorff warping boxes: a detail disappears while another one appears.

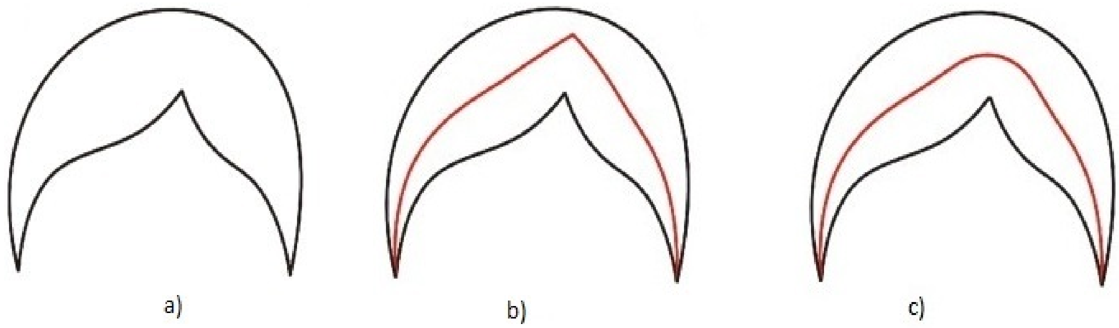


Figure 1.4: Comparison between skeleton and medial axis. a) Original object. b) Skeleton. c) Medial axis.

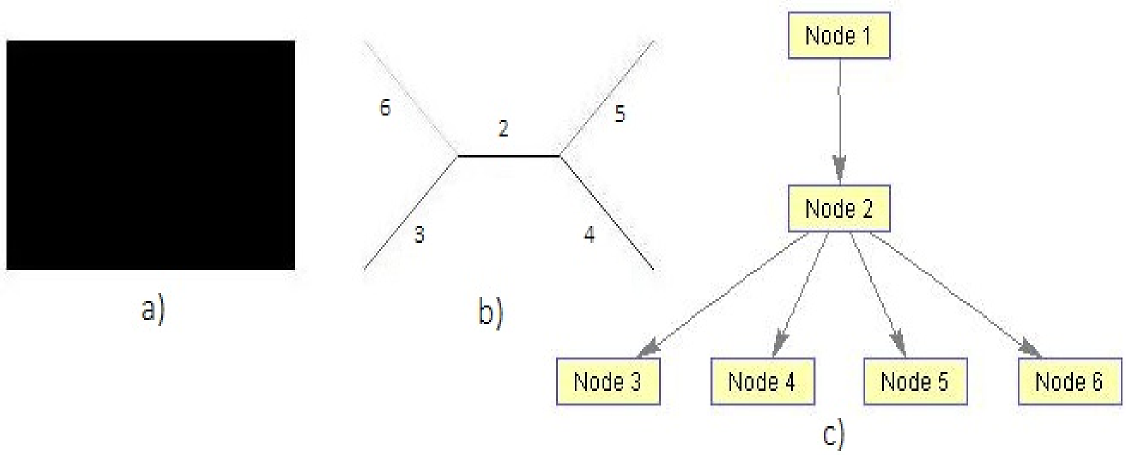


Figure 1.5: Shock Graph of rectangle object. a) Original object. b) Medial axis(numbers denote the nodes of the shock graph). c) Shock Graph.

- The concept of **medial axis** can be described with the help of the fire grass concept [8]. If a fire starts from all points of a planar curve at the same time and moves with constant velocities in all directions in the same plane, then the Medial Axis is the locus of points where the fire meets itself. In other words, medial axis is the locus of center points of all the maximal circles contained within the shape boundary. The work involving medial axis computation can be found in [9, 10, 11, 12, 13, 14, 15, 16, 17]

  Figure 1.4 illustrates how the skeleton and medial axis are different from each other. Skeleton just gives the thinned version of the object boundary. It doesn't take care of the concavity of the object. You cannot find which part of the object boundary is concave. Medial axis gives some information about concavity present in the boundary. Wherever there is a curve in the medial axis, there is a possibility that the boundary is concave along the radius of that curve. So medial axis representation contains more information than the skeletal representation of the object[18]

- A **shock graph** is obtained from the medial axis of an object boundary. It is an abstraction of the medial axis of a shape onto a directed acyclic graph. The time of formation of each of the medial axis point is defined by the distance of that point to the closest boundary point, which corresponds to the radius of the inscribed circle with maximal radius. The medial axis points are first labelled according to the local variation of the radius function(mapping of each medial axis point to the radius of the inscribed circle with maximal radius, centred at that point) at each point, and according to that, shock graph is constructed, with a node being a parent of other one if it forms first in the object evolution, and if both of them are connected[19].

Figure 1.5 shows the shock graph representation of a rectangle object. Node 2 is parent of nodes 3,4,5 and 6 as it forms first in evolution, and all the nodes 3,4,5 and 6 are connected to node 2. Node 1 is the initial or root node(it would be more clear afterwards that if we don't include it, in some cases, we may have two root nodes of the tree).

Shock graphs possess much more information about the shape of an object, than medial axis does. This helps us to compare the similarities between different views of the same object, occluded view of an object, or even tranformations like scaling and rotation. One of the interesting feature of shock graphs is they are bend invariant. This helps us in finding the similarities between different views of the objects which have a tendency to bend. This can be extremely helpful in many cases, as will be seen shortly. Significant amount of work done on shape matching and shape recognition using shock graphs can be found in [20, 21, 22, 23] and recently by Feragen et. al. in [24, 25, 26]. A comparison between object represented by curves and graph has been discussed in [27].

We focus on shock graph representation of shapes and explain the details in the following chapter.

CHAPTER 2

Computing Shock Graphs

2.1 Introduction

In this chapter we focus on computational algorithms for shock graphs. Following is the procedure of computing an approximation of shock graph of a two-dimensional object:

- Compute the medial axis of the object.
  - Use divergence based approach[17] and set a threshold on divergence to find the probable medial axis.
  - Apply thinning to get one pixel thick medial axis.
  - Remove noise branches.
- Assign labels to all the medial axis points in a branch, according to their distance from the object boundary.
- Construct a shock graph for the given shape.

A branch, in medial axis, is a set of all the connected medial axis points between two endpoints, or an endpoint and a junction point, or two junction points of the medial axis. End point ia a terminal point of the medial axis. Junction point is a point which is connected to more than two connected set of points. The resulting graph is a hierarchical structure where each node represents one branch. So each connected set of same-labelled points will be stored in one node. Now we discuss each step in detail in the following sections. We require to set certain thresholds at different stages, which will also be discussed in this chapter.

2.2 Computing Medial Axis

Blum has given a very simple and popular definition of medial axis in terms of grass fire flow[8]. Consider the shape of the object to be uniformly covered with grass and simultaneously, fire is set to all the points on the boundary of the shape. The flow of the fire will be propagating towards the centre of the object, and will get extinguished when it will meet the fire flow coming from opposite direction. The meeting points of these fire flows are the medial axis points of the shape. They together form the medial axis. Mathematically,

$$\frac{\partial \mathcal{C}}{\partial t} = \mathcal{N}$$

acting on a 2D closed curve $\mathcal{C}$, such that each point on its boundary is moving with unit speed in the direction of the inward normal $\mathcal{N}$. This leads to divergence computation on the gradient vector field of Euclidean distance function[28]. Such equation can be solved by looking at the evolution of the phase space of the equivalent Hamiltonian system. Since Hamiltonian systems are conservative, the locus of skeletal points coincides with locations where a conservation of energy principle is violated.

Let D be the Euclidean distance function to the initial curve C_0. The magnitude of its gradient, $||\nabla D||$, is equal to 1 in its smooth regime. with $\mathbf{q} = (x, y)$, $\mathbf{p} = (D_x, D_y)$, the Hamiltonian system is given by

$$\dot{\mathbf{p}} = -\frac{\partial H}{\partial(q)} = (0, 0), \qquad \dot{\mathbf{q}} = -\frac{\partial H}{\partial(p)} = (D_x, D_y),$$

with an associated Hamiltonian function $H = 1 - (D_x^2 + D_y^2)$[28]. Compute the divergence of gradient vector field $\mathbf{q}$ and detect locations where it is not zero. The divergence is defined as the net outward flux per unit area, as the area about the point shrinks to zero:

$$div(\dot{\mathbf{q}}) \equiv lim_{\Delta a \to 0} \frac{\int_L < \dot{\mathbf{q}}, \mathcal{N} > dl}{\Delta a}$$

where Δa is the area, L is its bounding contour and $\mathcal{N}$ is the outward normal at each point on the contour. Or

$$\int_a div(\dot{\mathbf{q}}) da \equiv \int_L < \dot{\mathbf{q}}, \mathcal{N} > dl$$

So, the integral of the divergence of the vector field within a finite area gives the net outward flux through the contour which bounds it. Negative flux indicates that the energy is lost. These locations are sinks or medial axis points. Similarly, positive flux indicates that those locations are sources. But we neglect these points as these are exterios to the object contour and our goal is to find the medial axis.

Figure 2.1 illustrates the divergence-based computation on a panther silhouette. Figure 2.2 shows the calculation of medial axis of an object. We apply thinning algorithm on the obtained axis points to have the thickness of the medial axis as one pixel.

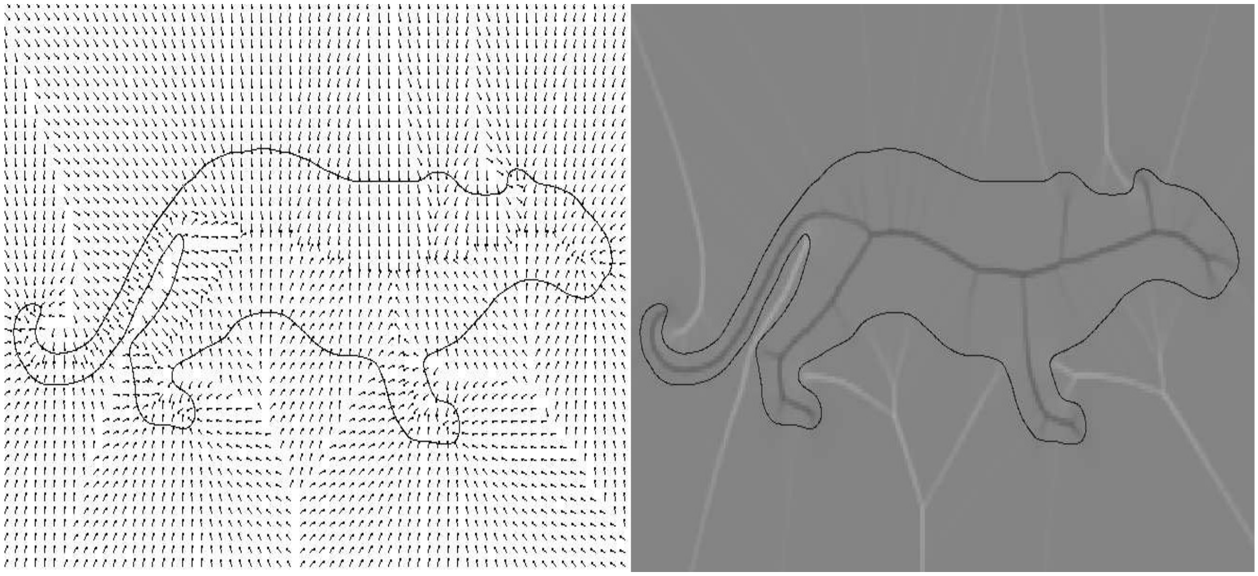


Figure 2.1: The gradient vector field of a signed distance function to the boundary of a panther shape(left), with the associated total outward flux(right).

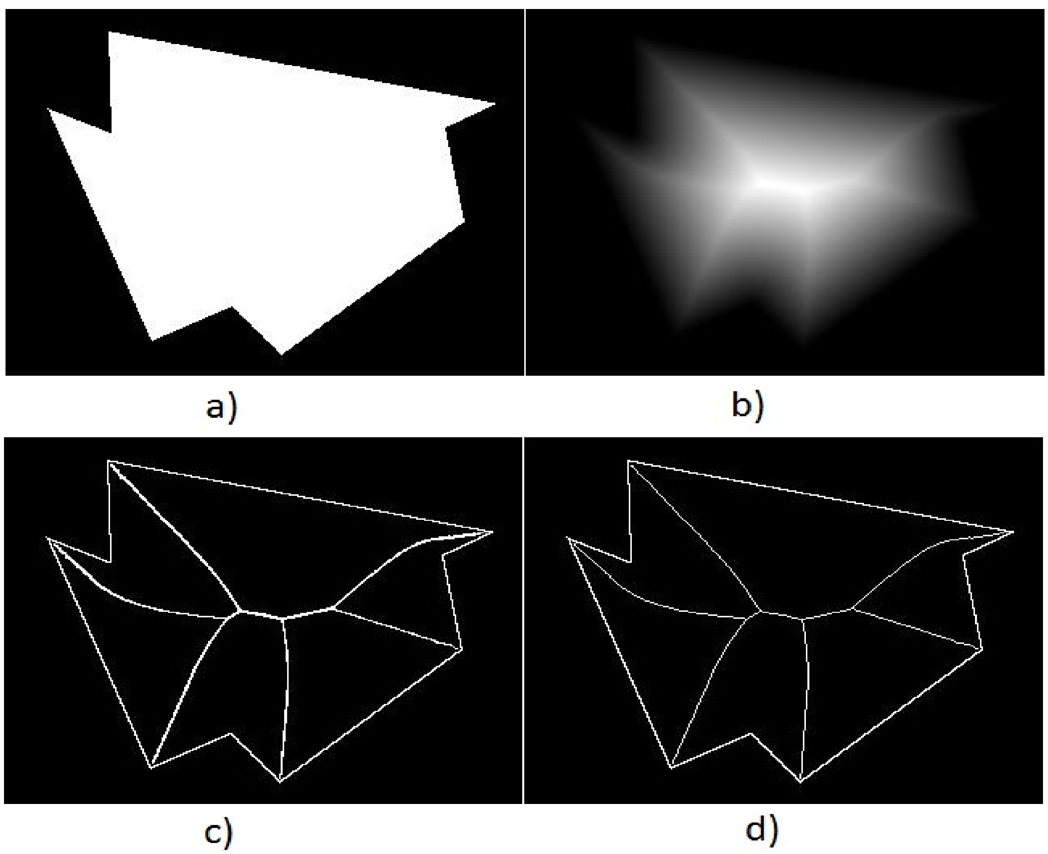


Figure 2.2: Medial axis computation. a) Original Object. b) The Euclidean distance function from the boundary. c) Calculating the total outward flux on the gradient of the gradient vector field of the Euclidean distance function. The points having the function value greater than a threshold value give rough medial axis. The boundary points are also shown for reference. d) Thinned medial axis.

We can get some branches in the medial axis which are actually very small in length, and are generally formed just because of digitization. We consider them as noise. So, we set a threshold value, and the branches which are of length less than this value, are considered noise, and removed. We discuss this threshold on divergence value in detail in section 2.5.1. Figure 2.3 shows the medial axis after removing the branches considered as noise.

To remove the branches, we need to know which points are the end-points and which are the branch-points, because we have to trace the branch from the end-point till a branch-point is encountered and find the length of the branch. If it is less than the set threshold value, we remove the traced points. We have matlab in-built functions to find end-points as well as branch-points(bwmorph(BW,’branchpoints) and bwmorph(BW,’endpoints’)). But these functions do not give results as we require them to be. For example, if we have a 4x8 matrix:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

If this matrix is given as input to the function to find end-point, the resulting matrix is:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

which is not actually the desired output, in our case. It gives all the end-points we require, but also some points on the medial axis which are not end-points.

End-points have a property, that after removing them along with their 8 neighborhood points, the medial axis remains one connected component. But then this also applies to the neighbors of the end-points. If the neighbor of an end-point is removed along with its 8 neighborhood points, then also the medial axis remains one connected component only. So to find the actual end-points, we take the intersection of this property and the result of the function to find the end-points. We get the desired result as:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

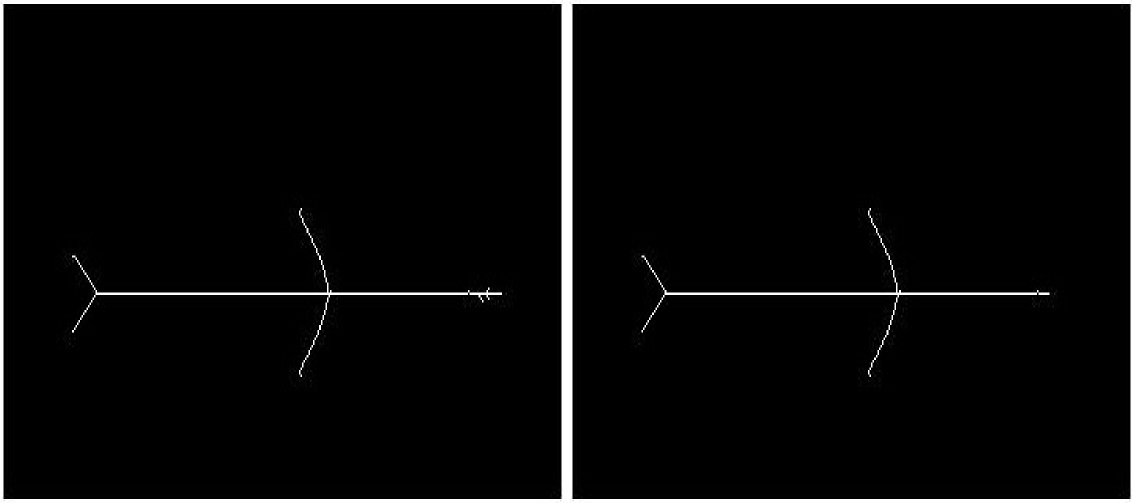


Figure 2.3: (Left) Medial axis with small branches present. (Right) Result after removing noise branches (threshold chosen = 10 pixels).

If we have the following 4x8 matrix as an input to the function to find branch-points:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

we get the resulting matrix as:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

which again is not the desired output. The desired one would be a matrix with all values zero.

Branch-points have a property, that after removing them with their 8 neighboring points, we get atleast three connected components. But here also, sometimes this applies to the neighbors of the branch-point. So we take the intersection of this property and the result of the function to find the branch-points and also add a constraint that no two branchpoints can be 8-neighbors. We get the desired result as:

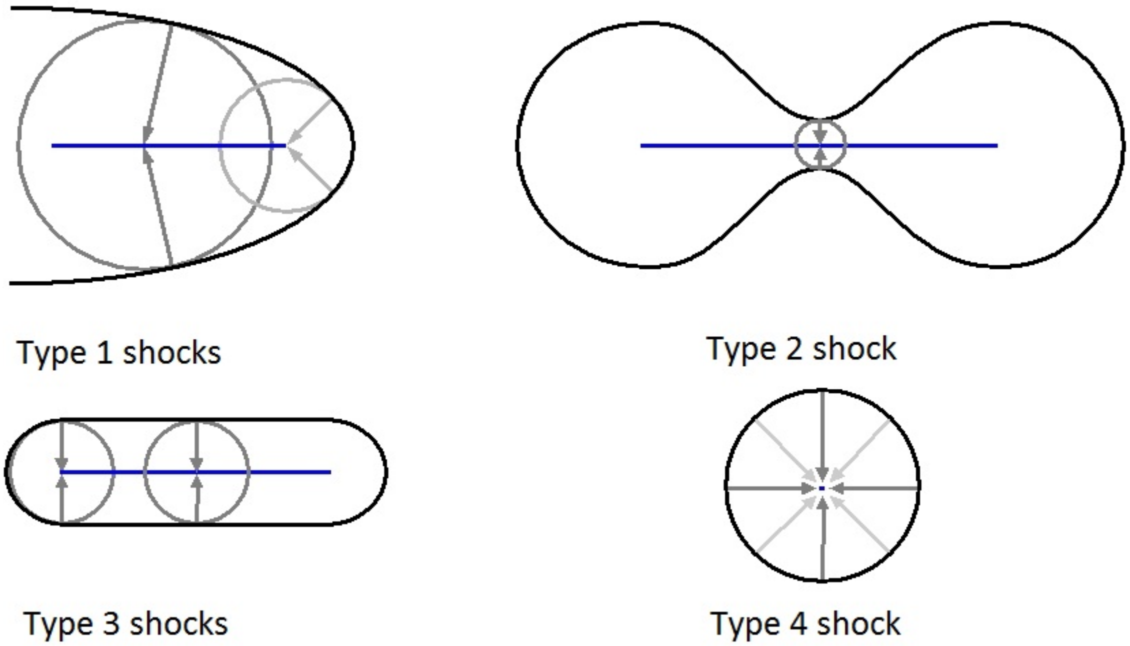


Figure 2.4: Medial axis points labelled according to their distance function.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Generally, after applying a threshold to the divergence value, the medial axis we get is not one pixel thick. So we need to apply further thinning.

We use divergence based thinning technique again, so that we don't loose the important information about the medial axis points. For each border point P, we remove the point P if it is 'removable', and if its divergence value is not the minimum in its 8-neighborhood. A point is potentially 'removable' if it has at least one background point in its neighborhood and on removing it, the 8-connected set of medial axis points does not get divided into two different 8-connected set of points.

2.3 Labelling of the points

Each point on the medial axis is labelled, taking into account its minimum distance from the boundary with respect to its neighbors' minimum distance from the boundary. We will consider the set of points S to be a continuous and connected set of the medial axis points(which, after labelling, are referred to as shock points, or simply shocks) of a closed curve[21]. Two points A and B in a set are continuous and connected if from A to B, there is a continuous path of points and all of these points lie in the same set. Four types of labelling can be done on the shocks(figure 2.4)[23]:

- Type 1: Assigned to a set of points whose minimum distance from the boundary varies monotonically. Either it is increasing or decreasing continuously.
- Type 2: Assigned to a point which is immediately followed by two type-1 branches flowing away from it in opposite direction.
- Type 3: Assigned to a set of points whose minimum distance from the boundary is constant. So the boundary on either side is parallel.
- Type 4: Assigned to a point which is immediately followed by two type-1 branched flowing towards it.

So type 2 shocks represent a local minima of the object boundary, like a neck, and type 4 shocks represent local maxima.

Now let us define them mathematically.Let $N(s, \epsilon)$ be the set of shocks in $S/\ \{s\}$ within distance ϵ from s, and let $N(s)$ be the set of the largest connected sets of points in $S/\ \{s\}$. Let $R(s)$ be the radius function $R : S \mapsto \mathbb{R}^+$, and $l(s)$ be a shock labelling function $l : S \mapsto \{1, 2, 3, 4\}$, where S is the set of continuous medial axis points (shocks) of shape X. Then $l(s)$ for $s \in S$ is defined as:

$$l(s) = \begin{cases} 4 & \text{if } \exists\epsilon > 0 \text{ s.t. } R(s) > R(s')\ \forall s' \in N(s, \epsilon), \\ 3 & \text{if } \exists\epsilon > 0 \text{ s.t. } R(s) = R(s')\ \forall s' \in N(s, \epsilon) \neq \emptyset, \\ 2 & \text{if } \exists\epsilon > 0 \text{ s.t. } R(s) < R(s')\ \forall s' \in N(s, \epsilon) \text{ and } N(s, \epsilon) \neq \emptyset \text{ and } N(s, \epsilon) \text{ is not} \\ & \text{connected,} \\ 1 & \text{otherwise.} \end{cases}$$

Again in discrete world we have to choose some threshold value ρ such that if the difference between the minimum distance of two points from the boundary is less than ρ, then we consider them to be at same distance from the boundary, other wise, different. Accordingly the shock types are assigned to the points. We discuss this threshold on radius function in detail in section 2.5.3. Figures 2.5, 2.6, 2.7 and 2.8 show the objects and their corresponding medial axis with assigned shocktypes.

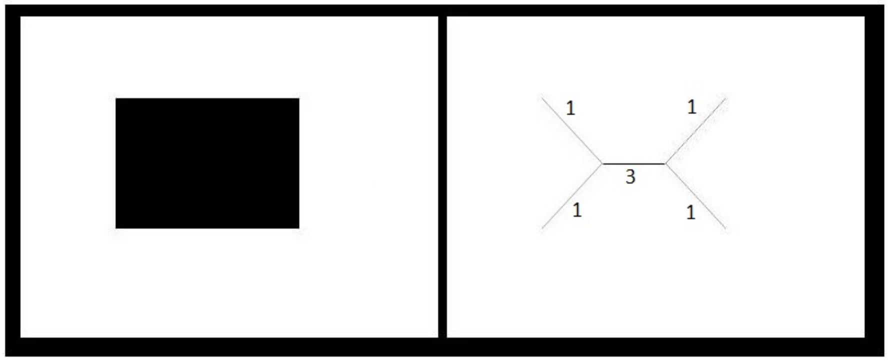


Figure 2.5: (Left) Object: rectangle. (Right) Medial axis, after assigning shocktypes.

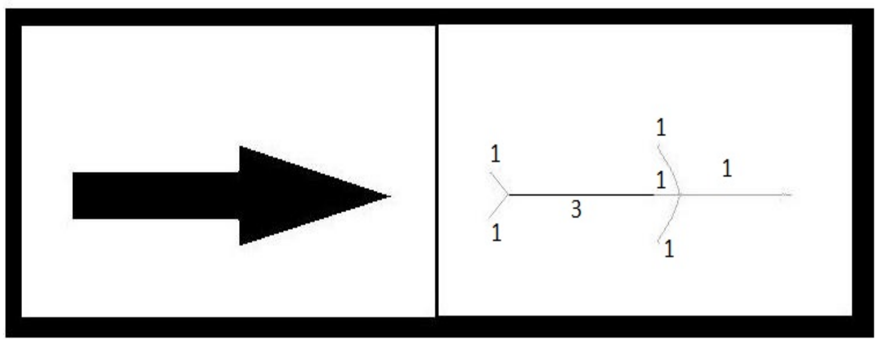


Figure 2.6: (Left) Object: arrow . (Right) Medial axis, after assigning shocktypes.

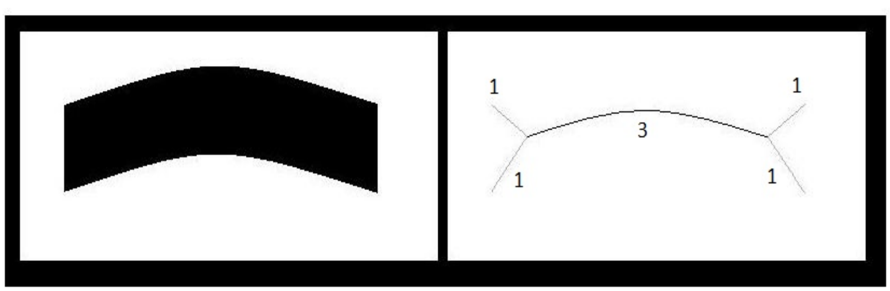


Figure 2.7: (Left) Object: Bent rectangle . (Right) Medial axis, after assigning shocktypes.

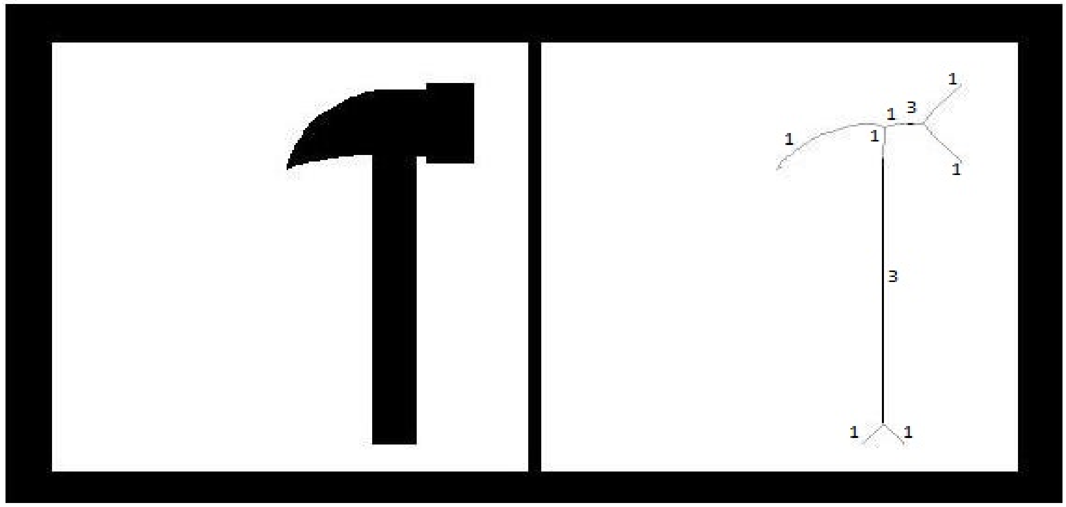


Figure 2.8: (Left) Object: hammer. (Right) Medial axis, after assigning shocktypes.

2.4 Constructing Shock Graphs

We group the set of points which are assigned same label and belong to the same branch(i.e. same set of connected points). Let $B_1 ... B_n$ be the largest groups of connected shocks in S s.t. $\forall s, s' \in B_i$, $l(s) = l(s')$ and $\forall s \in B_i$, either $|N(s)| \leq 2$ or if $|N(s)| > 2$, then s must be a terminal point of B_i(i.e. $B_i / \{s\}$ is connected), for $1 \leq i \leq n$. Let the group's label, $l(B_i)$, be the label of the shocks in B_i, and let the time of formation of the group, $t(B_i)$, be the interval $[\min_s(R(s)), \max_s(R(s))]$ defined by the time of formation $R(s)$ of all $s \in B_i$[29][18]. The shock graph of a 2-D shape is a labelled graph $G = (V, E, \gamma)$:

- Vertices $V = \{1, ..., n+1\}$, where node 1 denotes the root node, and node 2 to $n+1$ correspond to the groups $B_1 ... B_n$

- Edges $(i, j) \in E \subseteq V \times V$ directed from vertex i to vertex j if and only if $i \neq j$, and $i \neq 1 \wedge t(B_{i-1}) \geq t(B_{j-1}) \wedge B_{i-1} \cup B_{j-1}$ is a connected set of shocks, or $i = 1 \wedge \forall k \neq 0 (k, j) \notin E$.

- Labels $\gamma : V \mapsto \{\#, 1, 2, 3, 4\}$ such that $\gamma(i) = \#$ if $i = 1$ and $\gamma = l(B_{i-1})$ otherwise.

From the above definition, it can be seen that shock branches are represented as nodes of the shock graph. For an object with n number of medial axis groups, we have a shock

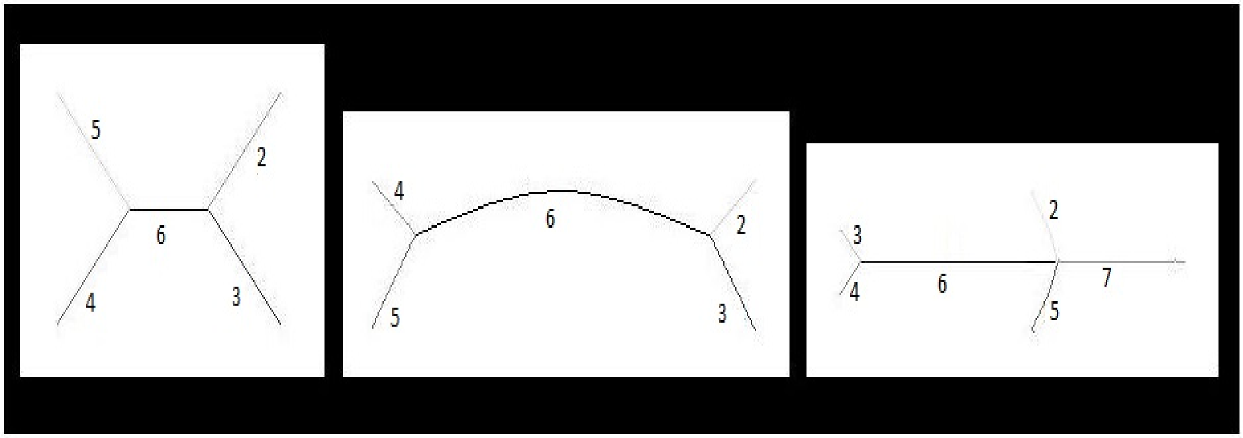


Figure 2.9: Grouped branches of the medial axis of Rectancle object(left), Bent rectangle object(middle) and arrow object(right).

graph with $n + 1$ number of nodes with the node 1 denoting the root node of the shock graph. The direction of edges is given according to the relative time of formation of each node. So there is an edge directed from node i to node j if group $i - 1$ is formed before group $j - 1$ and both the groups form a connected and continuous set of points. In other words, node j is child of node i. Also, a node is a child of the root node(node 1), only if it is not the child of any other node. Finaly we get a hierarchical or tree like structure for an object. We get the central parts of the object at the top of it, while the smaller shape extremities lie below them in the hierarchy.

2.4.1 Grouping of the Branches

Once we have a labelled medial axis, we assign a distinct group number to each branch on the medial axis. The result after assigning group numbers to the different branches has been shown in figures.

Figure 2.9 shows the grouping of the branches of the medial axis. The different intensities of different branches indicates their group number. For example, for rectangle object, group number is ranging from 1 to 5 (represented by the intensity values).

2.4.2 Creating Shock Graph

Next, we calculate the time of formation of each group on the medial axis. If we see this from curve evolution point of view, then we are finding the time at which a particular group starts evolving, and the time at which it has fully evolved. Here time is nothing but the distance of a point to the closest boundary. So, larger the value, older the point is, in the evolution process. For example, the start time and end time of all the groups of the

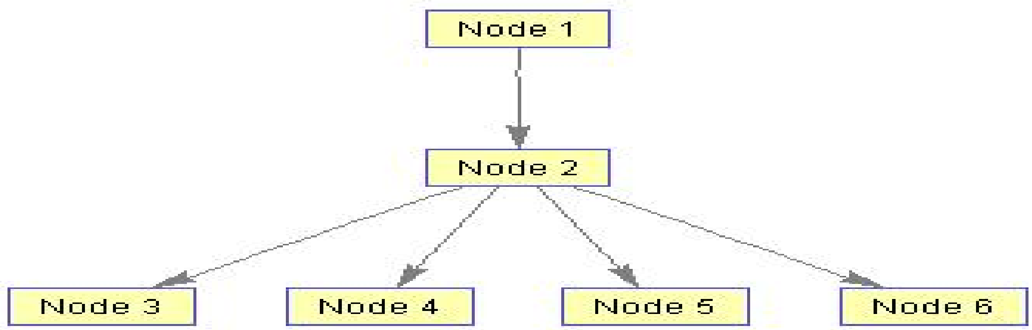


Figure 2.10: Shock Graph for object rectangle.

Group no.	Start time	End time
2	1.0000	34.7131
3	1.0000	33.0000
4	1.0000	32.8938
5	1.0000	33.9559
6	33.2415	36.0000

Table 2.1: Start and end time of the groups of the medial axis of rectangle

medial axis of bent rectangle are as shown in table 2.1.

After we have calculated the time, we proceed further to create an adjacency matrix, which along with the information of shock types will represent our shockgraph. It is directed graph. A directed edge from node A to node B conveys that the group which is represented by node A was formed before the group represented by node B, and A and B are connected in the medial axis. So, for rectangle, the adjacency matrix would be like:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 \\ +1 & +1 & +1 & +1 & 0 \end{pmatrix}$$

Figure 2.10 shows the shock graph constructed for rectangle object. We have deliberately started numbering the groups from number 2(which is subsequently denoted by node 2), as node 1 is considered as the root of this graph. The reason for considering one extra node as a root node is, as there can be some objects where there are two nodes being created at time 1, and if there is no extra node, then their will be two root nodes for their

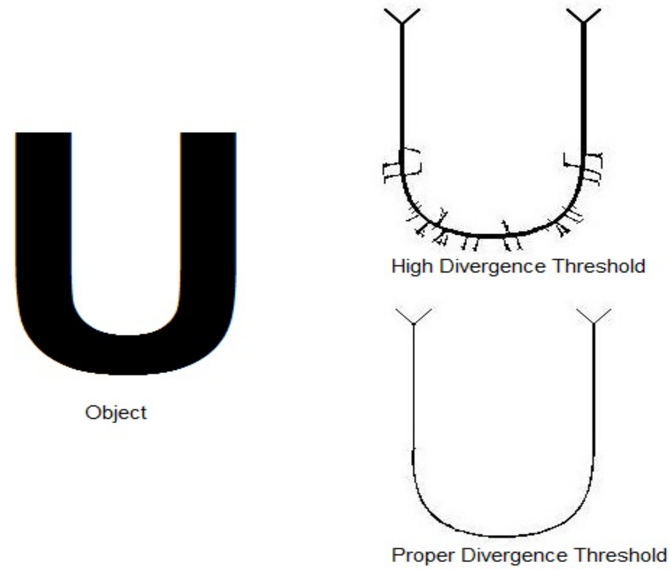


Figure 2.11: Variation in medial axis depending on the threshold on divergence value.

shock graphs.

2.5 Various thresholds and their effects on Shock Graphs

In this section we discuss effect of various thresholds we required to compute the approximation of the shock graphs, in detail. We will also see the problems faced due to digitization and solutions regarding these thresholds.

2.5.1 Threshold on divergence value

Different objects have different thresholds for divergence value, depending on their size and shape. Only the points which have divergence value less than the set threshold, are taken as probable medial axis points. Others are set to zero. If we put higher threshold value, then we get extra branches which shouldn't be there in ideal case. If we put a lower threshold then we may miss some of the branches which actually should be there in ideal case. Figure 2.11 shows the effect of different threshold values on the formation of medial axis. As can be seen, the threshold value has to be chosen very carefully(and unfortunately, manually).

Though, there is a solution to find the proper threshold value for a subset of the objects. When we compute medial axis by our divergence based approach, medial axis is curvy along the concave corners. But, it always touches the convex corners of an object. We exploit this property of medial axis. First we find the convex corners in the given shape, and then find the value of threshold, such that these corners get included in the medial axis(as medial axis touches the convex boundary). The procedure is given in Algorithm 1.

Algorithm 1 Algorithm to find number of convex corners in an object

---

for all the rotations of the object **do**
    Find the probability of all the boundary points for being a corner by Harris corner detector[30]. Set a threshold value such that the points having the probability above it, will be considered as Probable Corner Points
    for each of the Probable Corner Points **do**
        Find the neighborhood for a particular set radius. A convex corner will have more than half of the neighborhood points as background points. If the point satisfies this condition, take it as a convex Corner Point
        Remove all other points which are Probable Corner Points, and are in this point's neighborhood(as we have already chosen one point from that area)
    end for
end for
Find the mode of the number of convex Corner Points obtained from each of the rotations(we do this as this gives more accurate number of convex corners in the shape). This give the Actual Number of convex Corners of the object
Take one of the rotated shape where we found the Actual Number of convex Corners, and find its divergence based medial axis, initially with a very low set threshold
while the Actual Number of convex Corners are not included in the medial axis **do**
    Increase the threshold by a small set value
end while

We can find the convex corners for the rotated shapes of the object for every 10 degree rotation. In this way, we will have 36 different rotations of the object, and thus 36 results for the number of convex corners of that object. We find the maximum occurring result out of these results(i.e. their mode) which gives the actual number of convex corners. Next, we choose one of the rotated shape which gave the correct number of convex corners, and find its medial axis. We start with a very low threshold for divergence and keep on increasing it with a small value until all the convex corners of the shape are not included in the medial axis.

This algorithm works good only for the shapes having the convex corners. The algorithm works sufficiently well for the corners having the internal angle upto 110 degrees. So it has limited application.

2.5.2 Threshold on the length of noise(branch) to be removed

Some branches in medial axis get formed due to digitization, and are considered to be noise. Generally noise branches are shorter than the significant branches. So we can apply threshold on the length of the branches. Branches having length less than that threshold

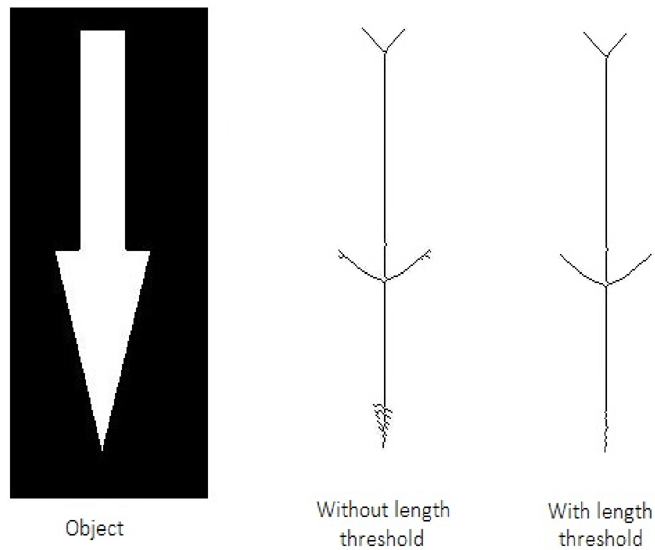


Figure 2.12: Variation in medial axis depending on the threshold on length of noisy branches.

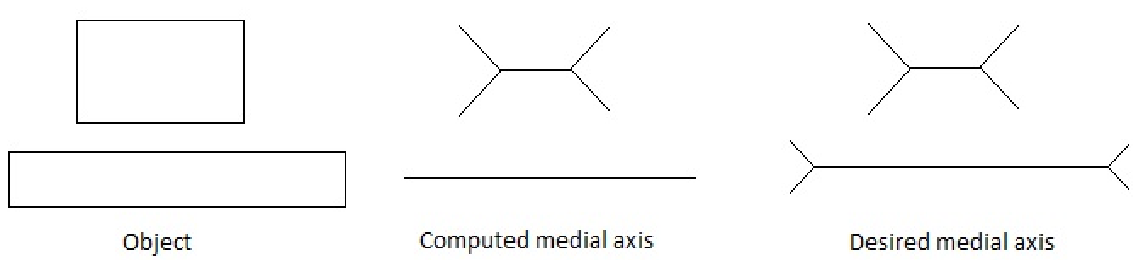


Figure 2.13: Sometimes the significant branches may also get lost due the threshold.

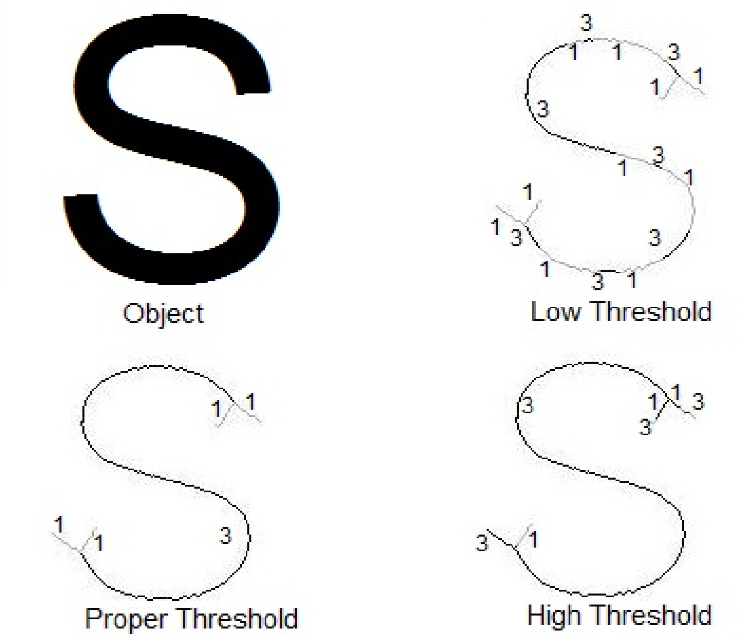


Figure 2.14: Variation in the assignment of shock types, depending on the threshold value.

value are set to zero. But the value of threshold varies from object to object, and has to be chosen carefully, because there is possiblility that a significant branch may get removed. To find the threshold on length, we use the fact that these unwanted branches are much shorter than the largest branch in the medial axis. So we can take a preset percentage value of that largest branch's length as the threshold value. For example, any branch whose length is less than 10 percentage of length of the largest branch, will be termed as insignificant and be deleted. Figure 2.12 shows the medial axis with the noisy branches and without noisy branches after applying the threshold on their length.

This brings its own disadvantage as well. Sometimes, due to the kind of the shape of the object, we may miss out on some significant branches from the medial axis. If there is even a single branch which is too long in length, then all the branches of length less than its 10 percent length, even if some of them were intended to be stored, will be lost. This will lead to the formation of different shock graph of the object, than the desired one(figure 2.13).

2.5.3 Threshold on radius function to assign shocktype

As discussed in section 2.2, we have to take this threshold into account. We have to use the equality condition for type 3 shock points as $R(s) = R(s) \pm \rho$. The choice of ρ depends on the scale. This threshold value also has to be chosen very carefully. The effect of the different values of this threshold are shown in figure 2.14.

Object	Threshold on divergence	Threshold on radius function
Alphabet ’L’	65	0.35
Rectangle	105	0.35
Alphabet ’C’	75	0.5
Alphabet ’S’	75	0.5
Alphabet ’U’	75	0.5
Hammer	75	0.9

Table 2.2: Values of thresholds for different objects

Also, the shock types 2 and 4 are single points. But generally they get assigned to type 3 or type 1 because of discrete image or small noise. This problem can be solved by labelling type 3 and type 1 shocks as type 2 or type 4 according to their local context.

The thresholds have to be set to different values for different objects, to get the desired output. For example, for the alphabet ’L’, the threshold value for divergence has to be set to 65, to compute the medial axis and the threshold value for radius function has to be set to 0.35. This means that two adjacent points in medial axis will be labelled as type 3 if the difference between their radius function is less than 0.35 units. Table 2.2 shows the threshold values used for different objects whose shock graphs are shown here. The divergence values are scaled in the range of 0 to 255.

CHAPTER 3

Effects of tranformations on shock graphs

In this chapter, we discuss effects of some transformations on our shock graph computation algorithm. Theoretically, shock graphs are translation, rotation, scale and bend invariant. It is obvious that translation operation will give the same shock graph, as practically, nothing changes in the object's boundary or shape. We see the results obtained when other trasformations are applied. Figure 3.1 shows the invariance of the shock graph for the mirror image of the object.

As the labelling of point depends on local boundary variations, it is also invariant of rotation and scaling(figure 3.3). But since we are dealing with a finite grid of pixels, there are some noticeable limitations when computing medial axis. Scaling may originate new branches in the medial axis. This happens as small details on the boundary which get neglected at a lower scale, become larger in high scale. New branches may affect the structure of the shock graph.

The most interesting property of shock graph is that it is bend invariant. Figure 3.2 shows the shock graph of a rectangular object, which remains the same even when the object is bent in different ways. This property of shock graphs can make them useful for some specific applications. For example, tracking an object movement. If we have a video of earthworm moving, and we want to track its movement. By providing proper threshold values initially, we can do that. Because of the bend invariancy of shock graphs, despite the movement of the earthworm, its shock graph remains same.

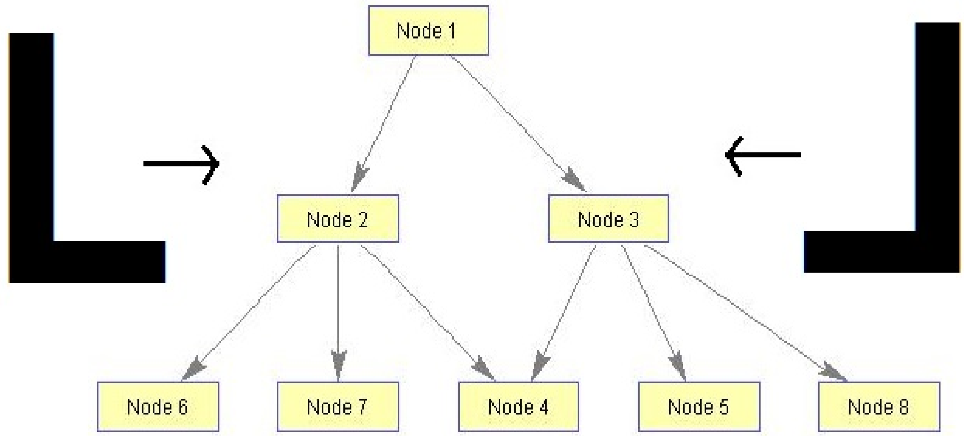


Figure 3.1: Shock Graph for English alphabet L and its mirror image.

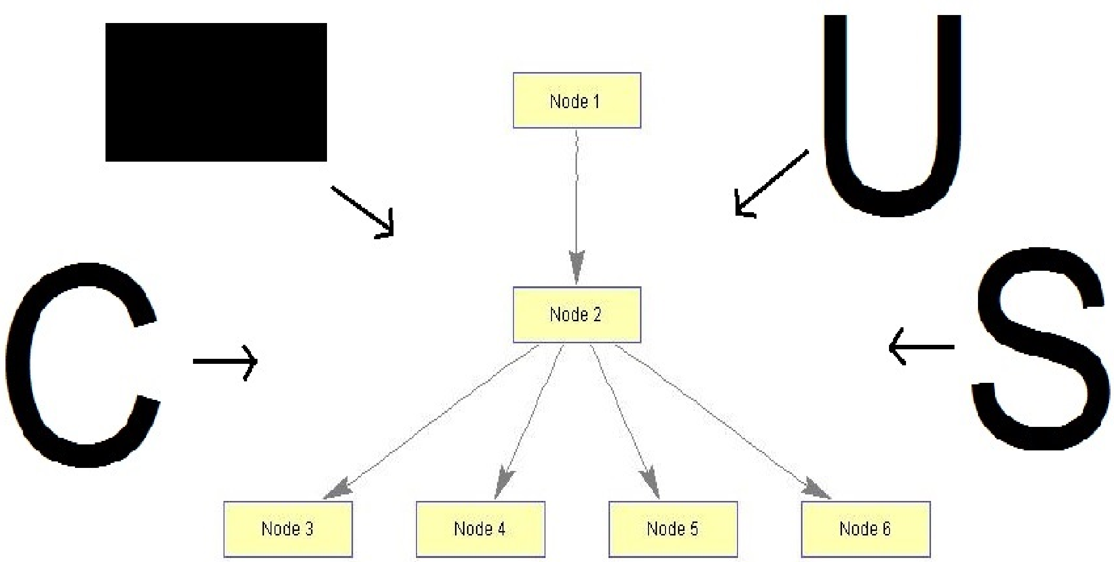


Figure 3.2: Bend invariant: Shock Graph remains same for rectangle and the letters S, U and C.

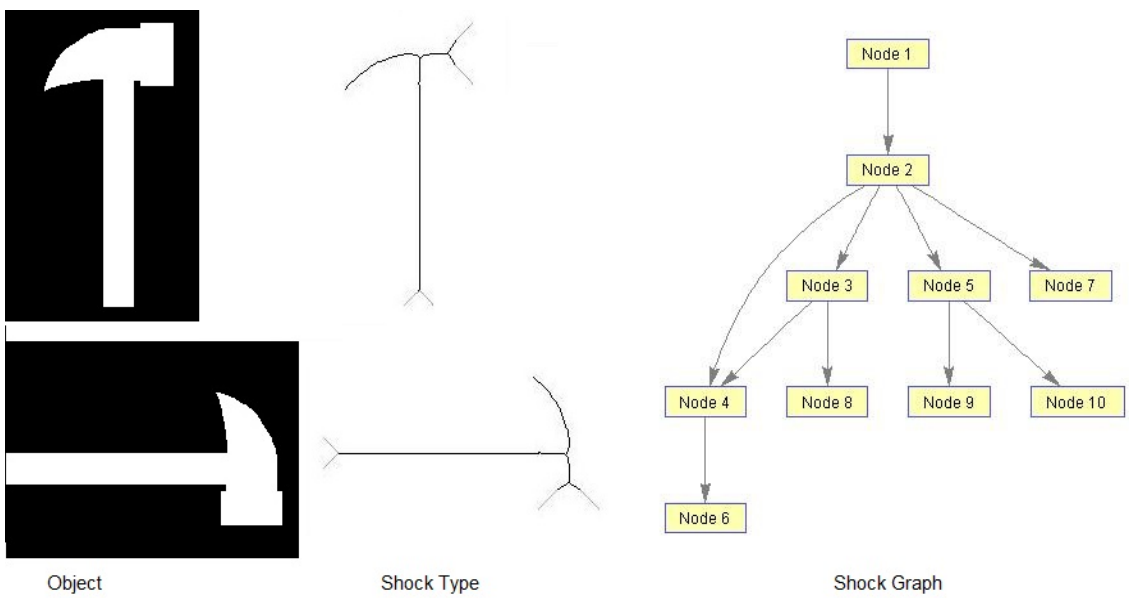


Figure 3.3: Shock Graph for Object hammer after rotating 90 degrees

CHAPTER 4

Conclusion and Future Work

4.1 Conclusion

Ideally shock graphs are rotation, translation, scale and bend invariant, but in digital world, we have to set some thresholds to compute a shock graph of an object. These thresholds may vary for different objects. Once these thresholds are set carefully, we can get desired results and work on them. This means shock graphs are not fully robust, but they can be used for specific purposes, where we can provide some a priori information(that means we set the threshold beforehand, according to our requirements). For example, it cannot distinguish between the animals like cat, dog, tiger or cheetah, but it can recognise all of them as four legged animal.

4.2 Future Work

In this thesis, we worked on the method of computing the shock graph and discussed thresholds involved in the process and how we can reduce the dependency on them. In future, besides working on reducing more number of thresholds from the process of finding shock graphs, much work can be done in object recognition. Statistical analysis of different shapes can be done. Compute the mean shape, given many shapes, feed them in a system and use them as a priori information for real-time object recognition. As discussed earlier, the system may not recognize the correct shape, but it can give the class of shapes to which the object belongs.

Graph matching algorithms can be used to find distance between two shapes[21, 31]. Thus comparison between two shapes can be done.

Given a shock graph, reconstruction of the probable shape of the object can be done. The labelling of the nodes and direction between two nodes can give the information about which node was formed after which one, and which two nodes are conneected to each other. This will lead to the formation of one of the simplest shape of its class.

References

[1] R. Basri, L. Costa, D. Geiger, and D. Jacobs, “Determining the similarity of deformable shapes,” *Vision Research*, vol. 38, no. 15, pp. 2365–2385, 1998.

[2] P. W. Michor and D. Mumford, “An overview of the riemannian metrics on spaces of curves using the hamiltonian approach,” *Applied and Computational Harmonic Analysis*, vol. 23, no. 1, pp. 74–113, 2007.

[3] ——, “Riemannian geometries on spaces of plane curves,” *arXiv preprint math/0312384*, 2003.

[4] M. I. Miller, A. Trouvé, and L. Younes, “On the metrics and euler-lagrange equations of computational anatomy,” *Annual review of biomedical engineering*, vol. 4, no. 1, pp. 375–405, 2002.

[5] V. N. Pawar and S. N. Talbar, “Article: Object recognition technique based on level set method and neural network,” *International Journal of Computer Applications*, vol. 40, no. 2, pp. 8–12, February 2012, published by Foundation of Computer Science, New York, USA.

[6] G. Charpiat, O. Faugeras, and R. Keriven, “Approximations of shape metrics and application to shape warping and empirical shape statistics,” *Foundations of Computational Mathematics*, vol. 5, no. 1, pp. 1–58, 2005.

[7] L. He, C. Y. Han, and W. G. Wee, “Object recognition and recovery by skeleton graph matching,” in *Multimedia and Expo, 2006 IEEE International Conference on*. IEEE, 2006, pp. 993–996.

[8] H. Blum *et al.*, “A transformation for extracting new descriptors of shape,” *Models for the perception of speech and visual form*, vol. 19, no. 5, pp. 362–380, 1967.

[9] H. Blum and R. N. Nagel, “Shape description using weighted symmetric axis features,” *Pattern recognition*, vol. 10, no. 3, pp. 167–180, 1978.

[10] F. L. Bookstein, “The line-skeleton,” *Computer Graphics and Image Processing*, vol. 11, no. 2, pp. 123–137, 1979.

[11] L. Calabi and W. E. Hartnett, "Shape recognition, prairie fires, convex deficiencies and skeletons," *The American Mathematical Monthly*, vol. 75, no. 4, pp. 335–342, 1968.

[12] D. G. Kirkpatrick, "Efficient computation of continuous skeletons," in *Foundations of Computer Science, 1979., 20th Annual Symposium on*. IEEE, 1979, pp. 18–27.

[13] U. Montanari, "A method for obtaining skeletons using a quasi-euclidean distance," *Journal of the ACM (JACM)*, vol. 15, no. 4, pp. 600–624, 1968.

[14] J. L. Pfaltz and A. Rosenfeld, "Computer representation of planar regions by their skeletons," *Communications of the ACM*, vol. 10, no. 2, pp. 119–122, 1967.

[15] F. P. Preparata, "The medial axis of a simple polygon," in *Mathematical Foundations of Computer Science 1977*. Springer, 1977, pp. 443–450.

[16] D.-T. Lee, "Medial axis transformation of a planar shape," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 4, pp. 363–369, 1982.

[17] D. Brunner and G. Brunnett, "Divergence analysis of discrete 2-d shapes," 2010.

[18] A. A. Eftekharian and H. T. Ilieş, "Distance functions and skeletal representations of rigid and non-rigid planar shapes," *Computer-Aided Design*, vol. 41, no. 12, pp. 865–876, 2009.

[19] P. Dimitrov, C. Phillips, and K. Siddiqi, "Robust and efficient skeletal graphs," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 1. IEEE, 2000, pp. 417–423.

[20] T. B. Sebastian, P. N. Klein, and B. B. Kimia, "Recognition of shapes by editing shock graphs." in *ICCV*, vol. 1, 2001, pp. 755–762.

[21] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker, "Shock graphs and shape matching," *International Journal of Computer Vision*, vol. 35, no. 1, pp. 13–32, 1999.

[22] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker, "Shapes, shocks, and deformations i: the components of two-dimensional shape and the reaction-diffusion space," *International journal of computer vision*, vol. 15, no. 3, pp. 189–224, 1995.

[23] K. Siddiqi and B. B. Kimia, "A shock grammar for recognition," in *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on*. IEEE, 1996, pp. 507–513.

[24] A. Feragen, F. Lauze, P. Lo, M. de Bruijne, and M. Nielsen, "Geometries on spaces of treelike shapes," in *Computer Vision–ACCV 2010*. Springer, 2011, pp. 160–173.

[25] A. Feragen, F. Lauze, and M. Nielsen, "Fundamental geodesic deformations in spaces of treelike shapes," in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 2089–2093.

[26] A. Feragen, "Complexity of computing distances between geometric trees," in *Structural, Syntactic, and Statistical Pattern Recognition*. Springer, 2012, pp. 89–97.

[27] T. B. Sebastian and B. B. Kimia, "Curves vs. skeletons in object recognition," *Signal Processing*, vol. 85, no. 2, pp. 247–263, 2005.

[28] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker, "Hamilton-jacobi skeletons," *International Journal of Computer Vision*, vol. 48, no. 3, pp. 215–231, 2002.

[29] D. A. Macrini, "Indexing and matching for view-based 3-d object recognition using shock graphs," Ph.D. dissertation, Citeseer, 2003.

[30] K. G. Derpanis, "The harris corner detector," *York University*, 2004.

[31] J. R. Ullmann, "An algorithm for subgraph isomorphism," *Journal of the ACM (JACM)*, vol. 23, no. 1, pp. 31–42, 1976.