Approximating the Medial Axis from the Voronoi Diagram And Its Application in Shape Matching

CS 6319 Computational Geometry- Final Project

Project Repo: https://github.com/0xrutvij/medial-axis-transform.git

Rhythm Patel (RGP190001)

Rutvij Shah (RDS190000)

The University of Texas at Dallas

Abstract

Medial Axis Transform is a method to represent a geometry with the same expressiveness but in a more compact way by storing the geometry in a lesser dimension. An approximation to the medial axis is a subset of the Voronoi diagram of the given sample points. An algorithm is outlined in this project to extract the Medial Axis from a Delaunay triangulationVoronoi diagram of a 2D shape. Further to demonstrate the application of Medial Axis, a Path Similarity-based algorithm is used to match two shapes. The results indicate success in matching shapes.

Introduction

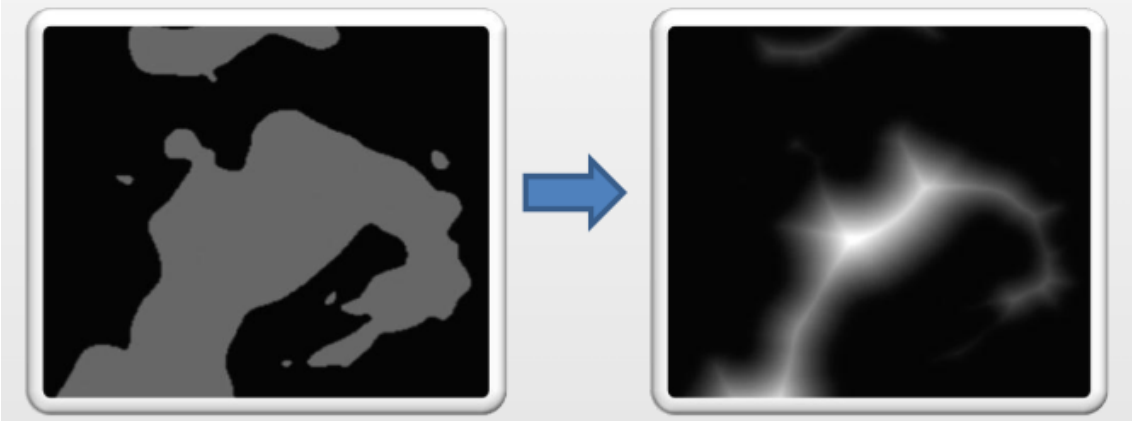
There are two major schemes to describe a geometry: explicit (surface meshes) and implicit (distance fields). These methods are extremely good at capturing all topological features of the geometry, however, a more compact representation with the same expressiveness is always desirable. A more compact representation means storing the entire geometry in a lesser dimension than the original. This in turn means that from a computational budget standpoint, the entire geometry can be stored in the most space-efficient way possible, and from a user standpoint, an interaction with a simpler abstraction (compact representation) is possible for visualizing and editing the same complex geometry.

This compact representation can be achieved using two major schemas: Medial Axis Transform and Skeletonization. Medial Axis Transform is the focus of this project. Medial Axis can be defined as the locus of all points in the interior of the geometry that has more than one closest point on the shape boundary. With this definition, the Medial Axis of any 3D geometry is a 2D surface and that of any 2D geometry is a 1D curve. This satisfies our requirement of a compact representation expressed in a lesser dimension than the original geometry's dimensions. Since each point on a Medial Axis has at least 2 closest points on the surface, each point can be thought of as encoding the local thickness of the body. This means that for a 2D geometry, each point on the Medial Axis can be thought of as associated with a disk of radius equal to the local thickness value of the body. Similarly, for a 3D geometry, each point on the Medial Axis can be thought of as associated with a corresponding sphere of radius equal to the local thickness of the body. Assembling all the disks/spheres for the infinite points on the Medial Axis will reconstruct

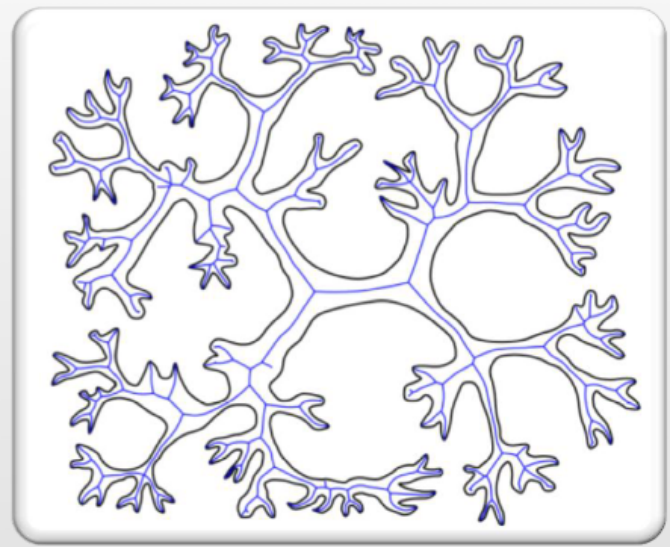
the original geometry with all of the topological features intact. This is one of the motives for us in this project to be able to demonstrate its working.

The algorithms to compute the Medial Axis of various geometries can be thought of as classified into the following categories:

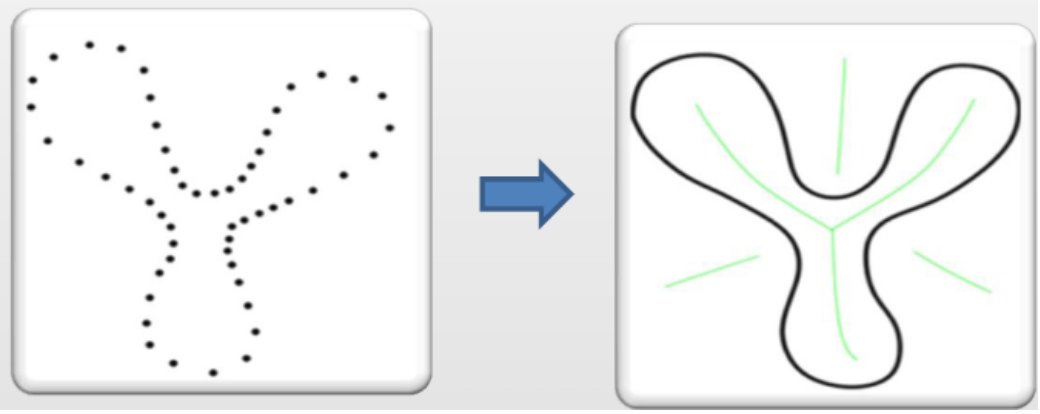
1. Discrete: Medial Axis is computed from the binary representation of an image where each pixel's value is the representation of the body.



2. Continuous: Medial Axis is computed from the analytical surface representation of the body.



3. Semi-continuous: Medial Axis is computed using the points sampled on the analytical boundary of the body. This is sensitive to sampling density. Sampling density is inversely proportional to the local thickness of the feature.



In this project, we are using the Voronoi-based method (Semi-continuous) to calculate the Medial Axis of a 2D geometry. We have used the developed algorithm to compute the Medial Axis of many different shapes as shown in the subsequent section. As an application to this and use the Path Similarity algorithm for shape matching. We demonstrate the successful working of this algorithm in the matching of different shapes.

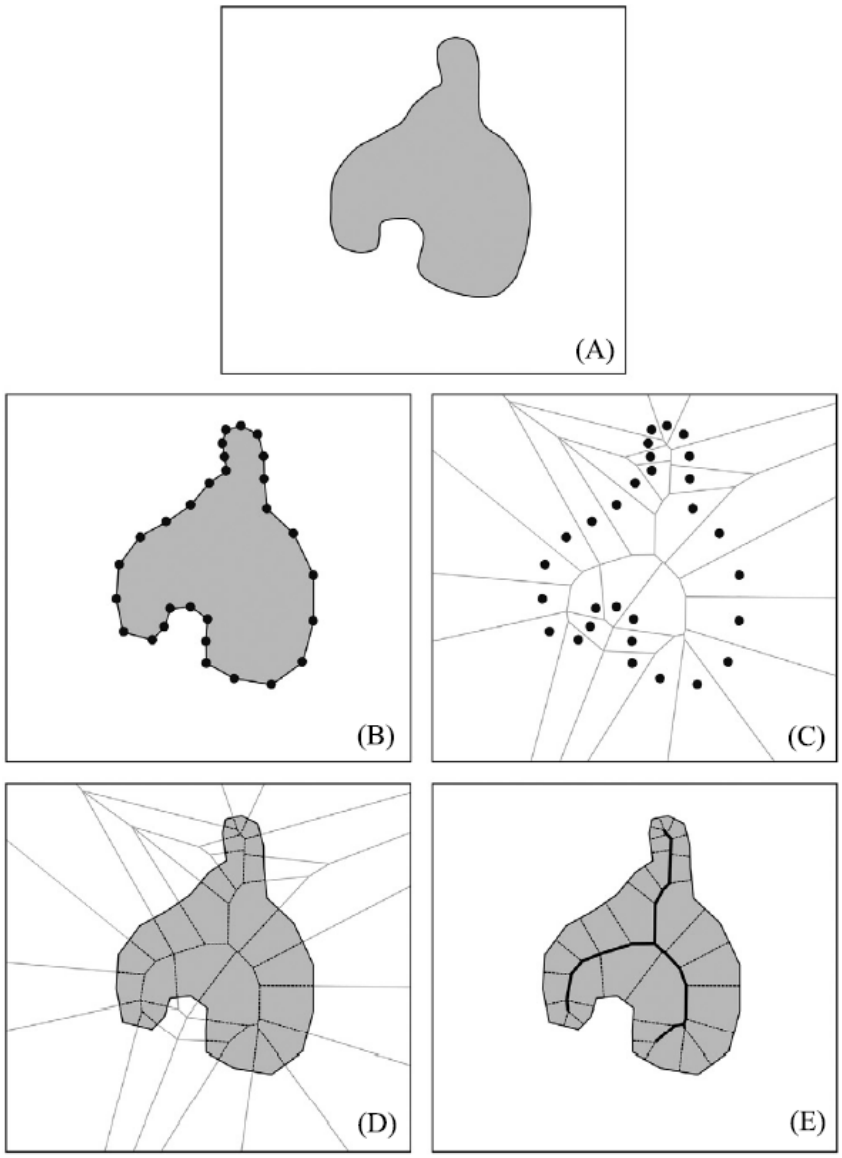
Method

Voronoi-based Medial Axis Computation

A Voronoi diagram assigns to each point in the discrete point set the set of points in the plane that are nearer to it than to any other boundary sample point. This results in the division of the plane into Voronoi regions. Delaunay triangulation can be defined as a dual of a given Voronoi diagram. The vertices of each Delaunay triangle are the Voronoi sites.

Given a sufficiently dense sampling of points on the boundary of a shape, the Voronoi diagram edges in the interior of the body are a good approximation to the Medial Axis by the

virtue of the definition of Voronoi cells as defined above. After the calculation of the Voronoi diagram, the next step is to extract those edges of the diagram which together approximate the Medial Axis of the body. It is worth noting that extraction of a Medial Axis through such a scheme is only an approximation and not the exact. However, as the sampling density approached infinity, the approximation of the Medial Axis approaches the exact one. The method can be visualized using the following diagram[1]:



An outline of the algorithm used in the computation of the Medial Axis is mentioned below[2]:

Input: Set of sampled points on the shape

Output: Approximation to the Medial Axis of the shape

1. Compute the Delaunay Triangulation of the given sample points.
2. Classify all the Infinite Delaunay faces as OUTSIDE and push them to a queue Q.
3. While Q is not empty:
    a. Pop Delaunay face f from Q
    b. For every unclassified neighbour triangle t of f:
        i. If the Delaunay edge shared by t and f is marked, classify t as the opposite of f
        ii. Else classify t the same like f
        iii. Push t to Q
4. For every Delaunay edge e:
    a. If the two neighbors Delaunay triangles d1, and d2 are both classified as INTERIOR, output the segment connecting the circumcenter of d1 and d2.

Comparing Shapes and Object Matching using Medial Axes

We aim to reproduce the approach used by *Bai et al.* in Path Similarity Skeleton Graph Matching [3]. The approach involves comparing geodesic paths between endpoints of skeleton graphs.

1. We compute the shortest paths along with the skeleton for each pair of endpoints and sample the paths at equidistant points using linear interpolation to get a radii vector of maximal inscribed balls. [Source Code 1]

2. Using the normalized radii vectors of two paths and the length of the paths, we estimate the distance between two curves $p(u,\ v)$ and $p(u',\ v')$ using the path distance function.

$$pd(p(u,\ v),\ p(u',\ v')) \ = \ \sum_{i=1}^{m} \frac{(r_i - r'_i)^2}{(r_i + r'_i)} \ + \ \alpha \frac{(l - l')^2}{l + l'}$$

3. Suppose two skeletons have $k \ + \ 1$ and $n \ + \ 1$ end points respectively, such that $k \ \leq n$. We find the path distance between the cartesian product of all paths emanating from some end point $a_i \ for \ i \ \in \ \{1 \ldots \ k \ + \ 1\}$ and $b_j \ for \ j \ \in \ \{1 \ldots \ n \ + \ 1\}$. This yields an $k \ \times \ n$ matrix. [Source Code 2]

4. To find the shortest path-distance between the two points in consideration, we obtain the optimal subsequence bijection of the matrix, which finds the least cost path between a_i and b_j . [Source Code 3]

5. This process is repeated for all pairs of end points between the two skeletons. To obtain $C(G,\ G')$, the matrix of least cost paths between a pair of end points a_i and b_j.

6. Now, applying the Hungarian algorithm on $C(G,\ G')$ we obtain the optimal linear sum between end points with a few points in G' skipped if $k \ < \ n$. This gives the total distance between two skeletons being compared, when comparing similar skeletons, the cost tends to 0. [Source Code 4]

The resultant end point correspondence finds similarity between skeletons with similar structures

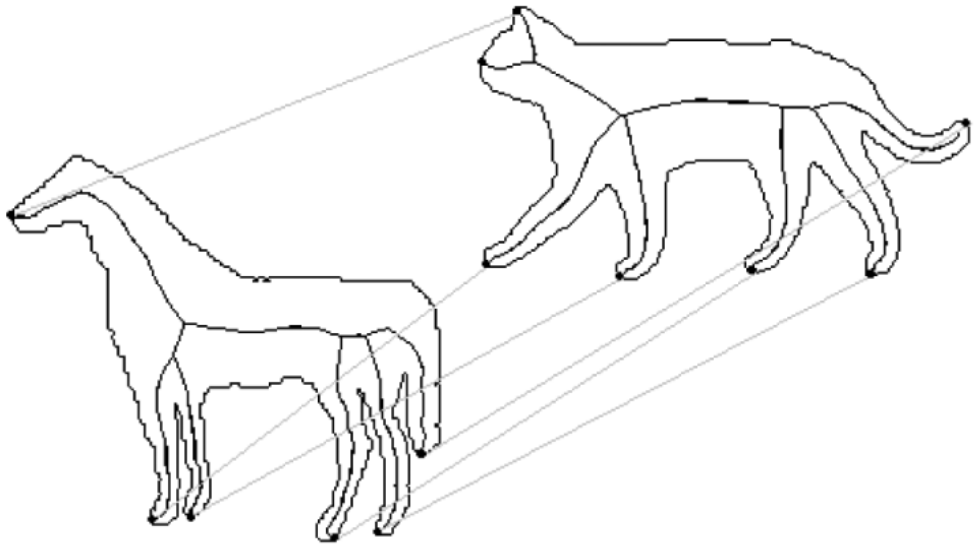
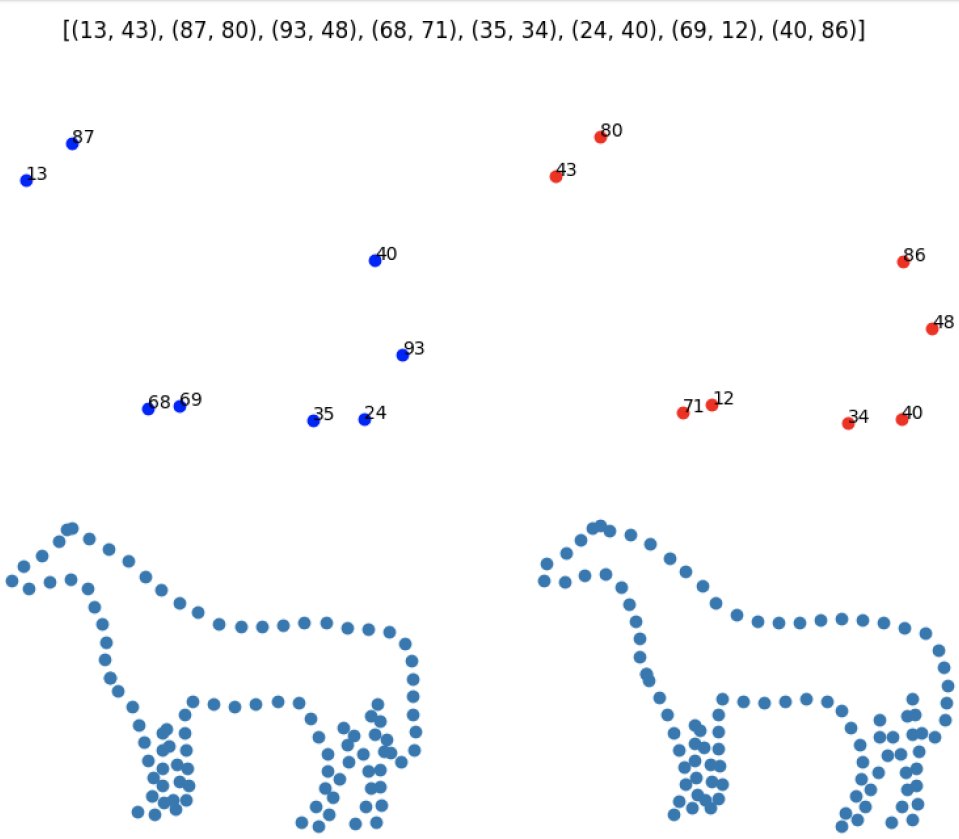


Fig. 7. The correspondence between a horse and a cat.

Following is an example of comparing two horse shapes



To calculate the cost above, the following algorithms and data structures were implemented (the ones which were taken from some libraries have been marked as such)

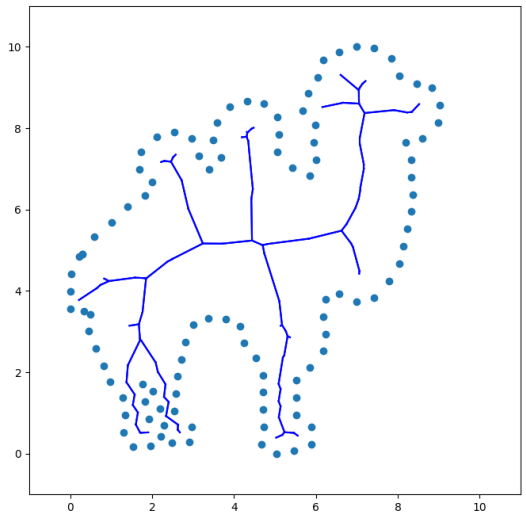
1. Path distance function
2. Optimal subsequence bijection based on the paper by *Latecki et al.* [4]
3. Finding shortest path in a DAG
4. Two way dictionary (a bijective function mapping data structure)
5. Dijkstra's shortest path algorithm
6. Linear Sum Assignment (scipy.optimize.linear_sum_assignment — SciPy v1.8.0 Manual)

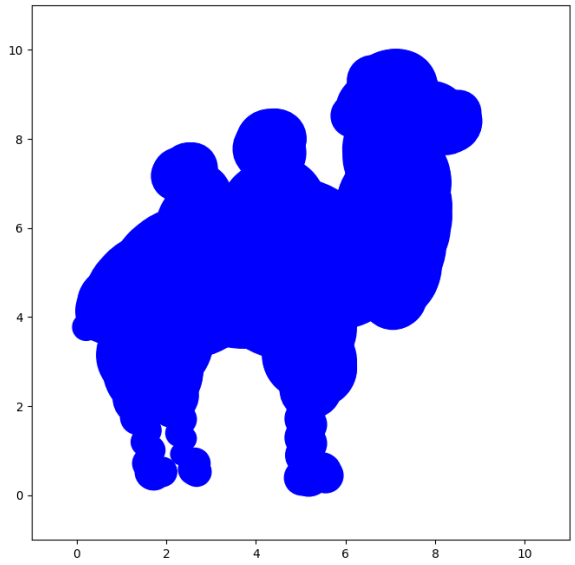
We further extended the concepts in the paper by using softmax function to get a probability of match whilst comparing some shape against a set of reference shapes. Since traditional softmax assigns higher probabilities to higher scored values, and we needed the inverse (minimum cost represents best match), instead of e, we use Euler's gamma as the base of our softmax function [Source Code 5].

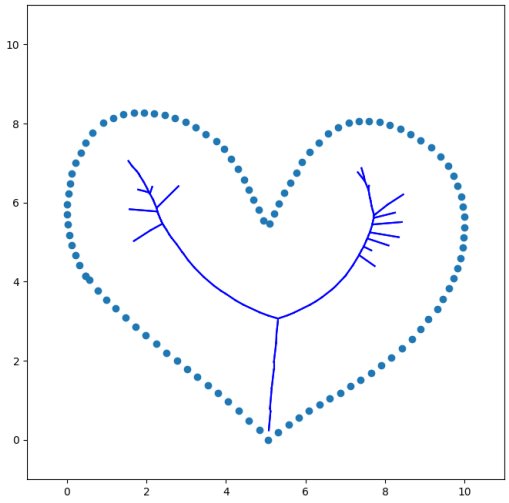
Results

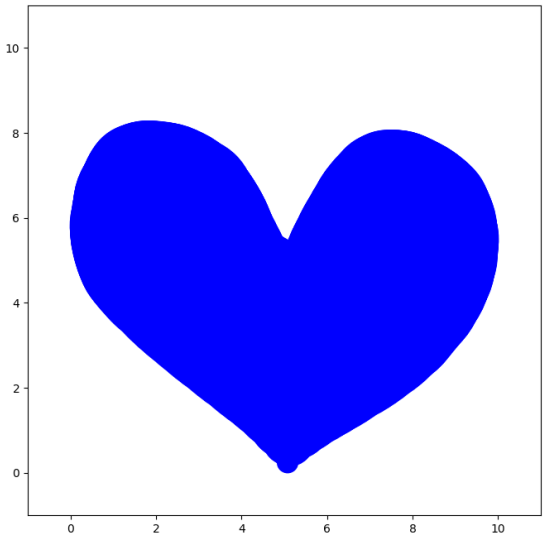
Medial Axis Transform:

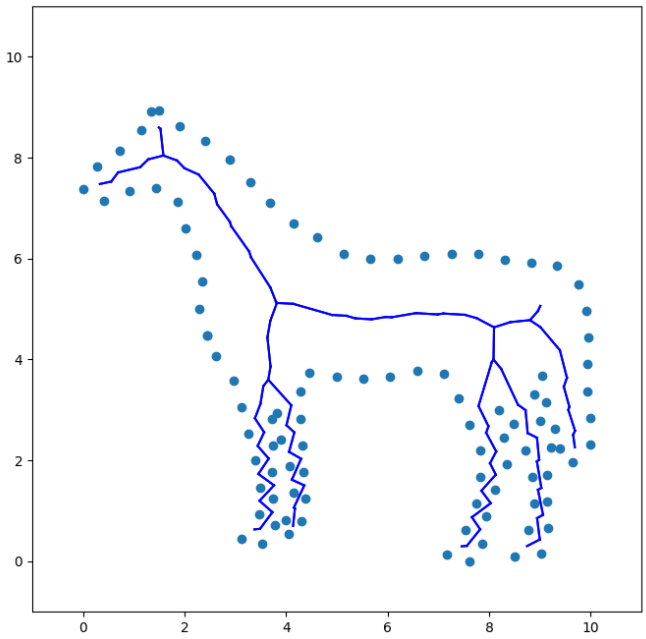
The first image is Medial Axis representation and the subsequent image is reconstruction of the original image from the medial axis computed.

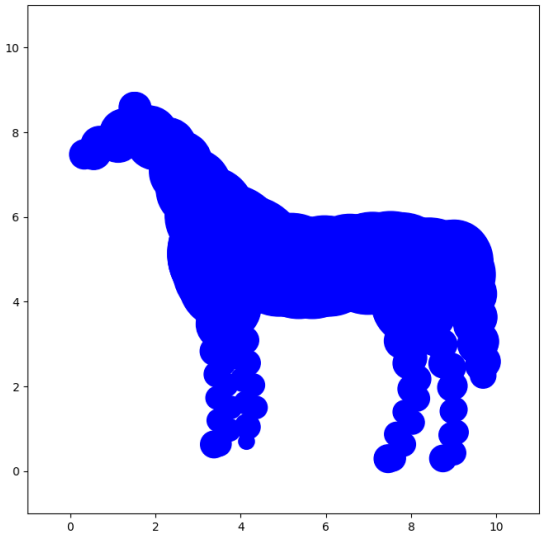



0
2
4
6
8
10


0
2
4
6
8
10


10
8
6
4
2
0
0
2
4
6
8
10


10
8
6
4
2
0
0
2
4
6
8
10



Shape Matching:

The reference shape set consisted of the following shapes ['device7', 'horseshoe', 'device5', 'misk', 'frog', 'octopus', 'key', 'hCircle', 'fork', 'device4', 'fish', 'personal_car', 'watch', 'classic', 'device2', 'beetle', 'bone', 'elephant', 'deer', 'heart', 'device0', 'chicken', 'horse', 'bell', 'children', 'cattle', 'shoe', 'cellular_phone', 'camel', 'comma', 'dog', 'pocket', 'guitar', 'spring', 'glass', 'ray', 'spoon', 'face', 'apple', 'device8', 'hat', 'pencil', 'hammer', 'device9', 'bird', 'flatfish', 'lizard', 'cup', 'crown', 'bat', 'jar']

Following are examples of the matching results

Best matches for horse_19

{ "deer_1": 17.86,

"elephant_1": 15.57,

"horse_1": 13.49,

"camel_1": 13.24,

"chicken_1": 10.29 }

Here we see that horse_19 matches best with various four legged animals, showing how similar shapes are identified by the skeleton matching algorithm.

Best matches for camel_2

{"camel_1": 14.79,

"chicken_1": 13.21,

"elephant_1": 10.78,

"frog_1": 10.52,

"beetle_1": 8.73}

In this instance, we are able to identify the shape correctly but other close matches consist of only one four legged animal.

To experiment with various matching scenarios, select an example image from the folder *shapes* and run the mat_example.py file using the following options

```
python3 mat_example.py -f shapes/horse/horse_19.txt --shape-matching
```

References

[1] Saha, Punam K., Gunilla Borgefors, and Gabriella Sanniti di Baja. "Skeletonization and its applications–a review." Skeletonization. Academic Press, 2017. 3-42.

[2] How to determine if a Delaunay triangle is internal or external? - Stack Overflow

[3] Xiang Bai and Longin Jan Latecki. 2008. Path Similarity Skeleton Graph Matching. *IEEE Trans. Pattern Anal. Mach. Intell.* 30, 7 (July 2008), 1282–1292. Path Similarity Skeleton Graph Matching | IEEE Journals & Magazine

[4] L. J. Latecki, Q. Wang, S. Koknar-Tezel and V. Megalooikonomou, "Optimal Subsequence Bijection," *Seventh IEEE International Conference on Data Mining* (ICDM 2007), 2007, pp. 565-570, doi: 10.1109/ICDM.2007.47. Path Similarity Skeleton Graph Matching | IEEE Journals & Magazine