# ME/IE/CS 558

## Final Project: Shape Processing

*Due May 4, 2018*

**For all assignments:** *Unless specifically indicated, you are free to use any publicly available sources: papers, books, programs, online material, etc. – as long as you clearly indicate and attribute the origin of the information.*

The goal of this assignment is to gain experience with tasks in polygonal shape processing: (1) construction, (2) validation, (3) querying, (4) triangulation, and (5) computation of its medial axis. Specific requirements for each task are described below.

1. *Construction.* Your first challenge is to construct the polygonal approximation of a shape $S$ defined by an inequality $f(x, y) \geq 0$. More specifically, given an expression $f(x, y)$ and a size parameter $h$, your program needs to construct a piecewise linear approximation (a connected sequence of line segments) of the boundary $\partial S$ defined by $f(x, y) = 0$, such that the length of every segment is $\leq h$.[1] Discuss how this is achieved by your construction in the analysis report.

2. *Validation.* Generally speaking, there is no guarantee that $f(x, y) = 0$ is not empty or that it corresponds to the boundary of a valid shape $S_p$. For the purpose of this assignment, we shall assume that a shape $S_p$ is valid if it is a simple polygon. Describe and implement all tests necessary to determine if the collection of line segments bounds a simple polygon $S_p$.

3. *Querying.* In case the computed shape $S_p$ is a simple polygon, design and implement an efficient point membership test and distance computation for an arbitrary point $p$. Point membership test should return in/on/out result; distance query should return the shortest distance from point $p$ to the polygon's boundary.

4. *Triangulation* Design and implement an algorithm of your choice to compute triangulation of the polygon $S_p$. The running time of the algorithm should not be worse than $O(n^2)$.

5. *Medial Axis* Design and implement an algorithm to compute (either exactly or approximately) the medial axis of the polygon $S_p$. The running time of the algorithm should not be worse than $O(n^2)$.

**Analysis (50 points)** The analysis portion of the project should discuss all mathematical assumptions, data structures, and algorithms in the implementation of the above tasks. Provide running time analysis of your algorithms using the usual $O()$ notation. In the construction task, discuss any limitations or guarantees provided by your algorithm. (For example, can you miss some important features? what happens to sharp corners of $S$? etc.) If any of your algorithms are approximate (for example, medial axis computation), clearly discuss the quality of approximation.

**Implementation & Testing (50 points)**

1. Implement the program to produce graphic outputs showing the computed piecewise linear approximation of $f \geq 0$ (including its interior), triangulation and medial axis respectively.

[1] The set $f(x, y) = 0$ is sometimes called a level set of $f$, and its computation is sometimes called contouring.

2. The input to the program is string specifying the function as a Python expression, such as "5*x**3 - 17.3 * y**2 + sin(x*y)" , etc. and a real number specifying the value of the size parameter $h$. The expressions may be evaluated in Python using eval() function.

3. Test your program on variety of functions. Wolfram is a good source for variety of such functions http://mathworld.wolfram.com

4. During grading, there are two input files. The expression file specifies the implicit expression, axis limits and $h$. The points file defines several points for point membership check.

   The format of **expression file** is

   ```
   expression
   xmin, xmax, ymin, ymax
   h
   ```

   You can assume the expression defines a function $f(x, y)$. And the level set ($f(x, y) = 0$) is within the domain $[x_{min}, x_{max}] \times [y_{min}, y_{max}]$.

   The format of **points file** (you can assume the delimiter is tab) is

   ```
   x1      y1
   x2      y2
   ...
   xn      yn
   ```

   your program will be called from command line as

   ```
   >> python shape_processing.py <expression_file> <points_file>
   ```

   For the below expression file

   ```
   -x**2/4 - y**2 + 9
   -7, 7, -4, 4
   1
   ```

   and points file

   ```
   0       0
   100     0
   ```

   A valid solution includes plotting (see Figure 1 - 3) and the below command line output.

   ```
   The constructed boundary is a simple polygon.
   Point membership check and minimum distance are listed below
   (0, 0) : in, 2.961293
   (100, 0): out, 94.062987
   ```

   If the constructed boundary is not a simple polygon, you program should output "the constructed boundary is not a simple polygon" and exit.

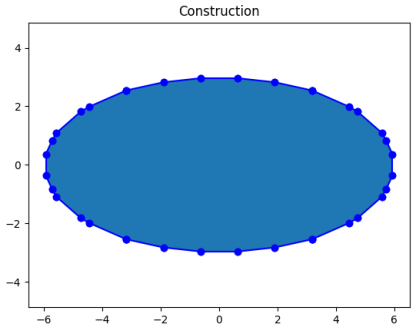


Figure 1: Construction (boundary and interior defined by $f > 0$)

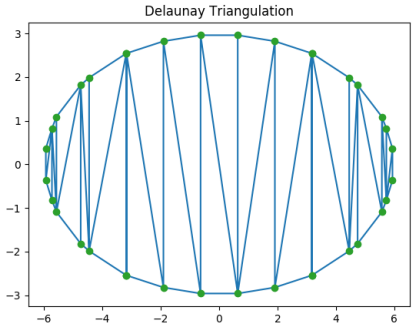


Figure 2: Delaunay triangulation

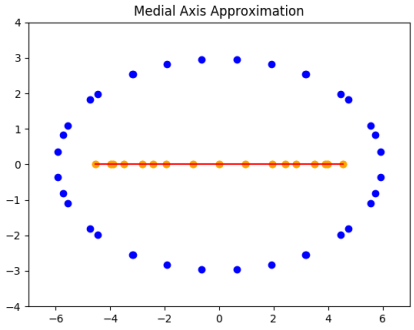


Figure 3: Medial Axis Approximation

5. Please also provide four other python files which are used for partial credit grading.

   The first python file is called “reconstruction.py” which is for step 1 and step 2. It will be called from command line as below.

```
>> python reconstruction.py <expression_file>
```

It produces graphic outputs showing the computed piecewise linear approximation of $f \geq 0$ (including its interior) and prints whether the boundary is a simple polygon or not to command line.

The second python file is called "pmc.py" which is for step 3. It will be called from command line as below.

```
>> python pmc.py <polygon_file> <points_file>
```

Polygon file is represented by its vertices in ccw orientation. For example a unit square is represented by

```
0    0
1    0
1    1
0    1
```

Its output is the point membership check and minimum distance results.

The third python file is called "triangulation.py" which is for step 4. It will be called from command line as below.

```
>> python triangulation.py <polygon_file>
```

It plots a triangulation of the polygon.

The third python file is called "medial_axis.py" which is for step 5. It will be called from command line as below.

```
>> python medial_axis.py <polygon_file>
```

It plots the approximated medial axis of the polygon.

**Extra Credit** (up to 15 points)

1. Extend your program to work on more general types of polygons defined by $f(x, y) \geq 0$ (e.g., with holes, disconnected, etc.)
2. Improve your point membership test and distance computation algorithm to perform better than $O(n)$ time (with pre-processing).
3. Modify your triangulation program to compute Delaunay triangulation of the polygon.
4. Design and implement $O(n)$ algorithm to compute convex hull of the polygon.

## Deliverables

Please use the course website to submit a single zip named FirstName_LastName_Project.zip The zip archive should contain: (1) the analysis portion of the assignment, (2) the documented python source file, and (3) a PDF readme file specfying the instructions for running the code. It should also include at least 1 sample run with input and output, and specify any specific dependencies or requirements of your code.