2-D Medial Axis Computation in Matlab
Version 1.0

Krishnan Suresh; suresh@engr.wisc.edu
Engineering Representations and Simulation Laboratory
University of Wisconsin, Madison
www.ersl.wisc.edu

1. Background & Caveats

The attached Matlab routines compute the **medial axis** (see Figure 1) of **2-D solids** whose boundary is made-up of line segments and circular arcs. For an introduction to medial axis and some of its applications, see [1-13] and references therein. The figure below illustrates two examples. Run "**test.m**" to get started.

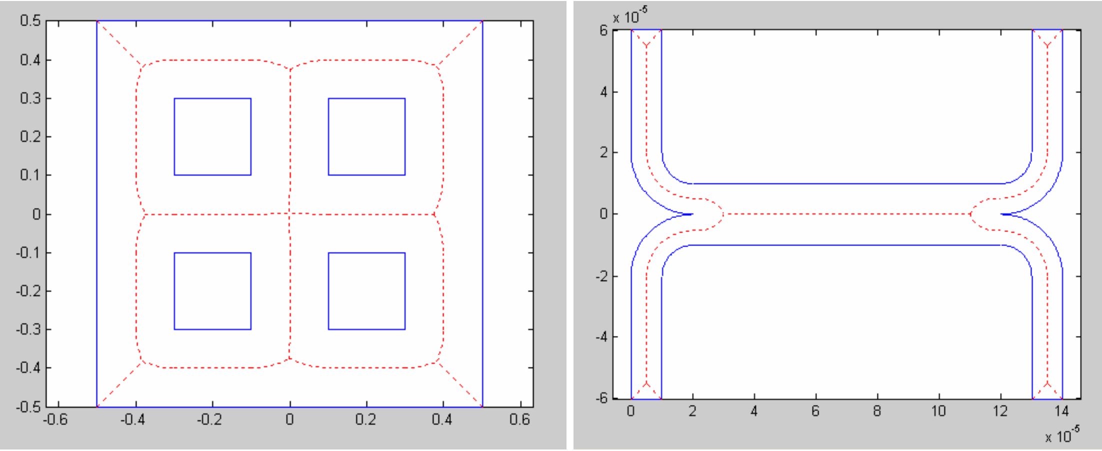


Figure 1: Medial axis examples.

Three main **caveats:**

1. The attached code is entirely written in MATLAB and not optimized, hence slow.
2. **There are also a few bugs that we are aware of**, and expect to address them in the near future. But, you can always send-in your bugs to Krishnan Suresh (suresh@engr.wisc.edu), together with the geometry that caused the problem. In any case, use the code with caution!
3. You will need the Matlab optimization toolbox to use the code.
4. Only the medial curves are computed; the radius function is not computed explicitly. However, you can compute the radius using the included routine `distToParents(pdeGeomPaddded,pt,parents)`

2. Input Description

The input to our code is a matrix 'pdeGeom' that describes the geometry. The matrix format is identical to the one used by the pde toolbox. You don't need the pde toolbox to use our code, but you can certainly use the toolbox to create complex geometry, export the geometry matrix, and use it as input to our code.

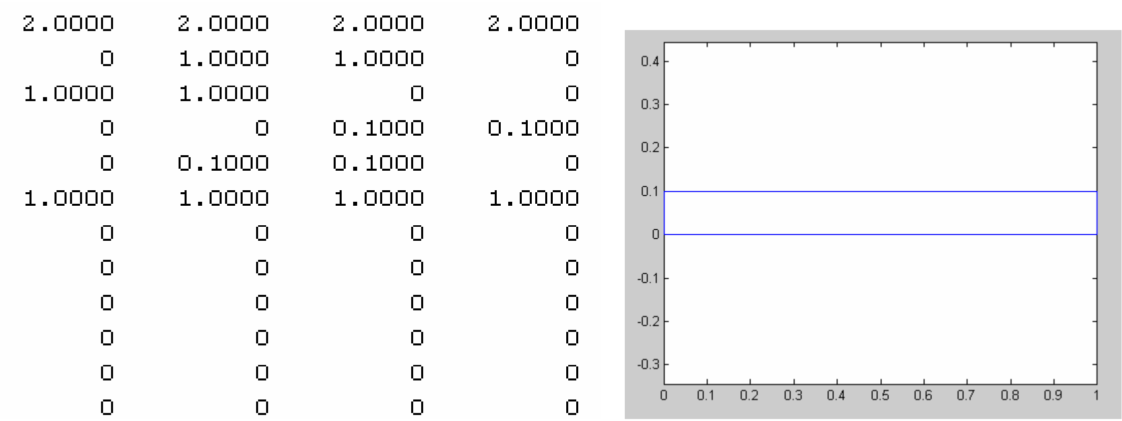
Briefly, the matrix 'pdeGeom' consists of 11 rows and as many columns as there are boundary segments. The first row (for each column is either 1 or 2); 1 means the segment is a circular arc, 2 means it is a line segment. The remaining rows are described below. Note that, per pde toolbox convention, the start and end segments of a circular arc segment are such that **the direction of the arc is always counter-clockwise**.

	Line segment (Row-1 = 2)	**Arc Segment (Row-1 = 1)**
Row-2	x_{start}	x_{start}
Row-3	x_{end}	x_{end}
Row-4	y_{start}	y_{start}
Row-5	y_{end}	y_{end}
Row-6	1 if the domain is to the left of segment, else 0	1 if the domain is to the left of segment, else 0
Row-7	0 if the domain is to the left of segment, else 1	0 if the domain is to the left of segment, else 1
Row-8	Not used	x_{center}
Row-9	Not used	y_{center}
Row-10	Not used	*radius*
Row-11	Not used	*radius*

Table 1: Input matrix format.

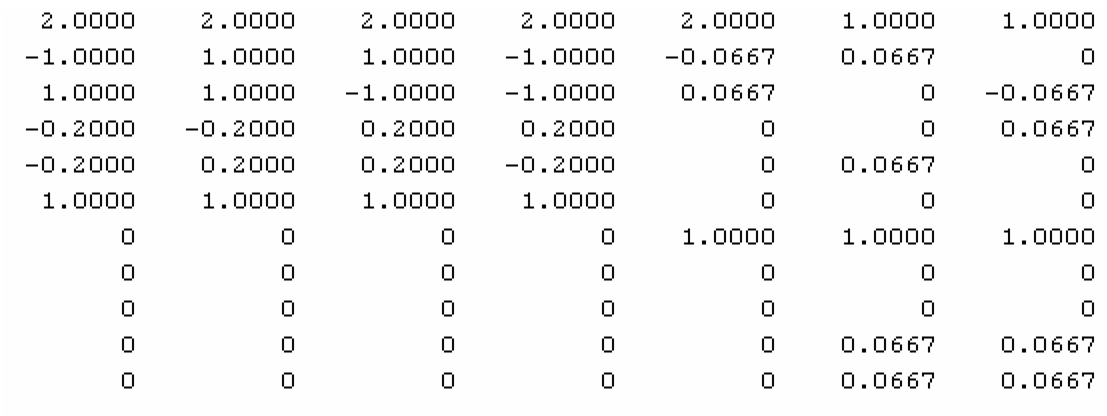
2.1 Input Examples

A few examples and corresponding geometries are provided below.



```
    2.0000    2.0000    2.0000    2.0000
         0    1.0000    1.0000         0
    1.0000    1.0000         0         0
         0         0    0.1000    0.1000
         0    0.1000    0.1000         0
    1.0000    1.0000    1.0000    1.0000
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0         0
```

Figure 2: Input matrix for a rectangle.



```
  2.0000    2.0000    2.0000    2.0000    2.0000    1.0000    1.0000
 -1.0000    1.0000    1.0000   -1.0000   -0.0667    0.0667         0
  1.0000    1.0000   -1.0000   -1.0000    0.0667         0   -0.0667
 -0.2000   -0.2000    0.2000    0.2000         0         0    0.0667
 -0.2000    0.2000    0.2000   -0.2000         0    0.0667         0
  1.0000    1.0000    1.0000    1.0000         0         0         0
       0         0         0         0    1.0000    1.0000    1.0000
       0         0         0         0         0         0         0
       0         0         0         0         0         0         0
       0         0         0         0         0    0.0667    0.0667
       0         0         0         0         0    0.0667    0.0667
```

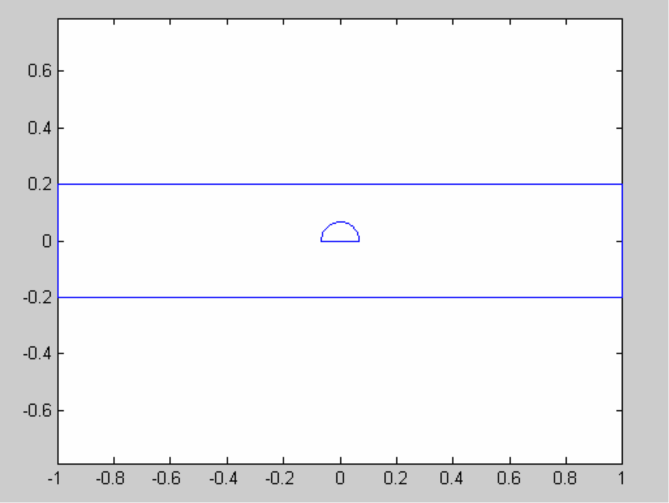


Figure 3: Input matrix for a rectangle.

There are a number of examples provided. **Please see the matlab file 'test.m'**.

2.2 Global Variables

There are four global variables used within the code with the following functionality.

```
1. global MMA_DEBUG; Default 'false'. Set to true if you want to see
   intermediate results for debugging
2. global MMA_REL_TOL; Default '1e-6'; The relative tolerance is
   used to detect identical and merge nearby medial points.
3. global MMA_ABS_TOL; Computed from MMA_REL_TOL. Do not assign.
4. global MMA_FLAT_VERTEX_ON; Default to 'false'. Set to true if one
   or more medial branches are missing (can happen in the presence
   of circular arcs). But the code will run even slower!!
```

3. Usage & Output Description

Once the matrix pdeGeom is created, the medial axis can be computed and displayed per:
>> [medialCurves, pdeGeomPadded, box] = computeMedialAxis(pdeGeom);
>> plotMedialCurves(medialCurves);

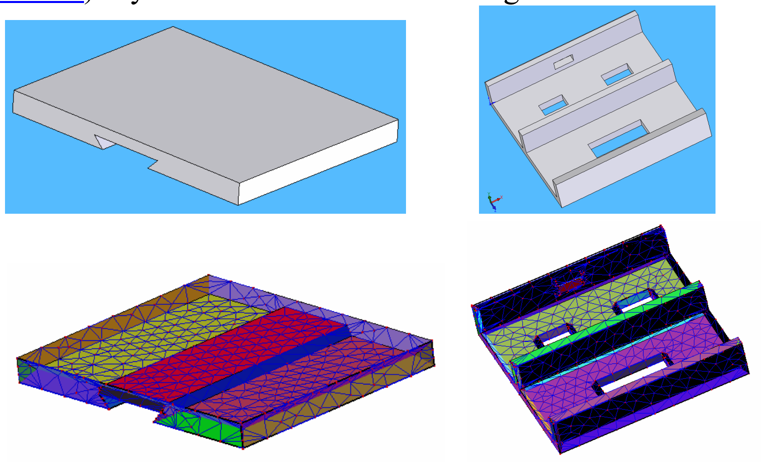
As seen above, the code generates three outputs:

- **medialCurves**: A vector of structures; the length of the vector corresponds to the number of medial branches. Each structure represents a Bezier curve consisting of
    - N control points (N is 2 is it is straight line bisector, else 3),
    - N weights,
    - type (neglect, used in one of our applications), and
    - 2 parents (the two boundary segments that the medial curve bisects, essentially the column numbers of pdeGeomPadded matrix described below).
- **pdeGeomPadded**: It is similar to the pdeGeom input, but with the columns shuffled and padded with additional columns; one columns is added for each concave point within the geometry. The first row for the padded columns has the value -1 to denote a concave point.
- **box**: The bounding box.

Please see the code plotMedialCurves for further details.

4. Medial Axis Computation in 3-D

We are currently working on a 3-D version, written in C++ and integrated within SolidWorks; see examples of the output below. Contact Krishnan Suresh (suresh@engr.wisc.edu) if you are interested in obtaining a license.



CPU time: 3.5 seconds **CPU time: 4.2 seconds**

Figure 4: Examples of 3-D Skeletal Surfaces and CPU time required to compute them (Code in development, and not included in the MATLAB version).

5. References

1. Blum, H., and Nagel, R. N., *Shape description using weighted symmetric axis features.* Pattern Recognition, 1978. **10**: p. 167-180.
2. Choi, H., Choi, S., Moon, H., *Mathematical theory of medial axis transform.* Pacific Journal of Mathematics, 1997. **181**(1): p. 57-88.
3. Amenta, N., Choi, S., Kolluri, R. *The Power Crust.* in *ACM Symposium on Solid Modeling and Applications*. 2001: ACM.

4. Culver, T., Keyser, J., and Manocha, D., *Exact computation of the medial axis of a polyhedron.* Computer Aided Geometric Design, 2004. **21**(1): p. 65-98.
5. Dey, T.K., Woo, H. Zhao, W. *Approximate medial axis for CAD models*. in *Proceedings of the eight ACM Symposium on Solid Modeling and Applications*. 2003. Seattle.
6. Donaghy, R.J., McCune, W., Bridgett, S. J., Armstrong, C. G., Robinson, D. J., McKeag, R. M.,. *Dimensional Reduction of Analysis Models*. in *5th International Meshing Roundtable*. 1996.
7. Goldak, J., Yu, X., Knight A., Dong. L., *Constructing discrete medial axis of 3-D objects.* International Journal of Computational Geometry and its Applications, 1991. **3**: p. 327-339.
8. Foskey, M., Lin, M. C., Manocha, D. *Efficient computation of a simplified medial axis*. in *Proceedings of the eighth ACM symposium on Solid Modeling and Applications*. 2003. Seattle.
9. Yang, Y., Brock, O., Moll, R. N. *Efficient and robust computation of an approximated medial axis*. in *ACM Symposium on Solid Modeling and Applications*. 2004. Genoa, Italy.
10. Suresh, K., *Generalization of the mid-element based dimensional reduction.* Journal of Computing and Information Science in Engineering, 2003. **3**(4): p. 308-314.
11. Sinha, M., Suresh, K. *Simplified Engineering Analysis via Medial Mesh Reduction*. in *ACM Symposium on Solid and Physical Modeling*. 2005. Cambridge, MA.
12. Suresh, K., *Skeletal Reduction of Boundary Value Problems.* International Journal of Numerical Methods in Engineering; Accepted for publication; pre-print available for download at www.ersl.wisc.edu, 2006.
13. Suresh, K. *Skeletal Reduction of Eigen-Value Problems over Thin Solids*. in *The International Conference on Computational Methods*. 2004. Singapore.