



## ShapeDetection

This interactive command line application demonstrates the use of ITK components for performing object delineation using level sets and fast marching.

### ***What are Level Sets? What is Fast Marching?***

“Level sets” is a numerical technique for computing and analyzing front propagation. Instead of propagating the front directly, the idea is to embed the front as the zero level set of a higher order function call the “level set function”. The major advantages of this technique are:

- stable and robust algorithms
- supports recovery of arbitrarily complex shapes such as shapes with corners and protrusions
- topological changes such as merging and splitting are handled automatically and implicitly.

If the speed function governing the front propagation never changes sign, an efficient numerical technique called “fast marching” can be used to compute the position of the moving front. Starting with an initial position, this method systematically marches the front outwards one grid point at a time, producing a time crossing map.

Object delineation can be achieved using an image gradient dependent speed function. At high gradient areas (e.g. near edges), the speed is set close to zero, thus slowing down the front motion. At low gradient areas, the speed is set close to one allowing the propagation to move through freely.

### **Further Information**

J. A. Sethian’s website, <http://www.math.berkeley.edu/~sethian/> is a good starting for learning more about level sets and fast marching.

## ***Application overview:***

This application reads in a raw 3D image volume. An edge potential map is generated using image gradient information. Each 2D slice from the input volume and potential map is written out as PGM files – facilitating viewing with simple 2D image viewers. The user then specifies an initial seed point. A time crossing map is computed via fast marching with the edge potential map as the speed function. The user can then inspect the state of the front at various times by specifying a time value.

On startup the application prompts the user to type in:

1. the filename of the input volume
2. specify the endian-ness of the file
3. the size of the input volume
4. the directory where the PGM files are to be written (NB the directory must already exist otherwise no images are written out)
5. the initial seed index position

The application then goes into command mode. There are four valid commands:

Command	Description
t ww	Sets the time value to ww. A binary mask of the front at time ww is computed. The results are written to the specified directory.
s xx yy zz	Sets the seed to index (xx,yy,zz) in column-row-slice order. A new time crossing map is computed. A binary mask of the front at the existing time value is computed. The results are written to the specified directory.
d	Display the current seed and time values.
x	Exit the application.

## ***What components of ITK does this application use?***

This application makes use of itk: :FastMarchingImageFilter.

## ***Example run:***

This is an example using dataset in the InsightData repository. Start the application as follows:

```
F:\Insight\Insight-VC++\Examples\ShapeDetection> ShapeDetection
```

The application then prompts the users to provide input and output information:

```
Input file name: n:\upenn\tumor\4121\raw\tumor1.flair
Input image big endian? [y|n]: y
Input image size: 256 256 30
PGM output directory: pgms
Writing PGM files of the input volume.
Writing PGM files of the edge potential map.
```

Each 2D slice from the input volume and edge potential map is written out as PGM files into directory `pgms`. The input volume slices have prefix `input` while the potential map slices have prefix `map`. The prefix is followed by the slice number and extension `.pgm`. Fig. 1 and 2 respectively show slice 17 of the input volume and edge potential map for this example. The application then prompts for the initial seed index and a time crossing map is computed using fast marching.

```
Set initial seed index: 110 63 17
Generating time crossing map.
```

The user can then view the state of the propagating front at any time by specifying the time value.

```
Command [s|t|d|x]: t 1000
Re-thresholding time crossing map.
```

A binary mask of the state of the front is produced by thresholding the time crossing map. Each 2D slice from the mask is written out as PGM files into directory `pgms`. The output files have prefix `seg` followed by the slice number and then extension `.pgm`. Fig 3. shows slice 17 of the mask at time 1000 .

The user can then interactively change the seed point and/or time value.

```
Command [s|t|d|x]: t 1500
Re-thresholding time crossing map.
```

```
Command [s|t|d|x]: t 5000
Re-thresholding time crossing map.
```

Fig. 4 and 5 respectively show slice 17 of the mask at time 1500 and 5000. Command 'd' display the current seed and time values and 'x' exits the application.

```
Command [s|t|d|x]: d
Seed: [110, 63, 17]
Threshold: 5000
```

```
Command [s|t|d|x]: x
Goodbye.
```

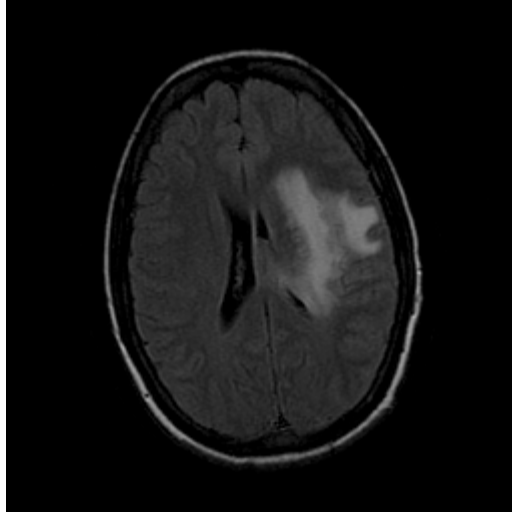


Fig 1: Slice 17 of input volume (pgms/i nput017. pgm)

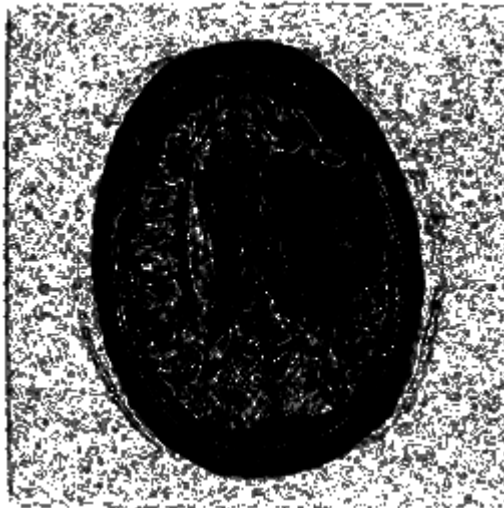


Fig 2: Slice 17 of the edge potential map (pgms/map017. pgm)

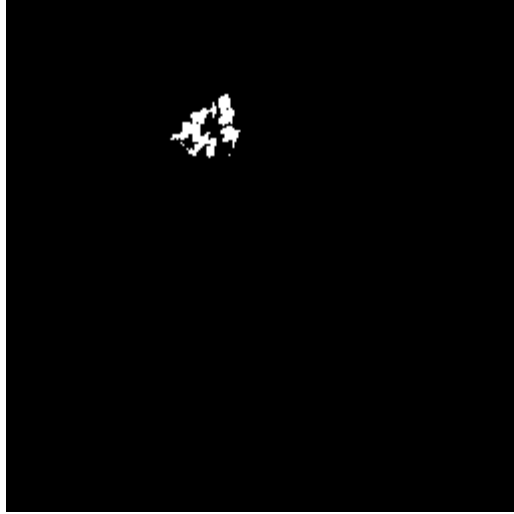


Fig 3: Slice 17 of binary mask with seed (110,63,17) at time 1000 (pgms/seg017. pgm)

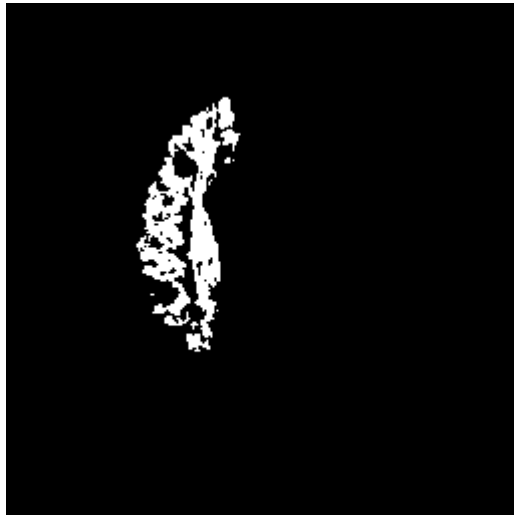


Fig 4: Slice 17 of binary mask with seed (110,63,17) at time 1500 (pgms/seg017. pgm)

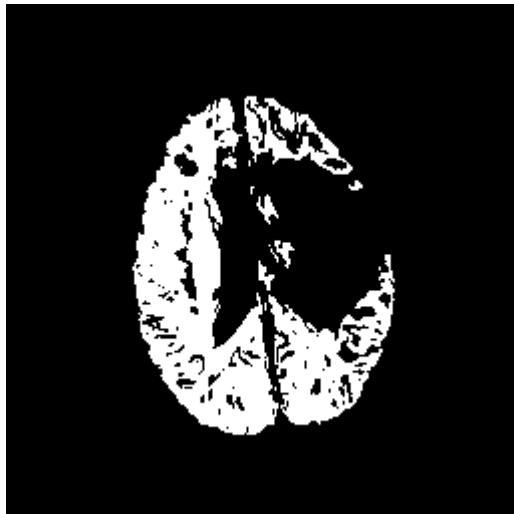


Fig 5: Slice 17 of binary mask with seed (110,63,17) at time 5000 (pgms/seg017. pgm)