Home Assignment #3
Fast marching method / Parametric Surface Evolution / Level sets

Submission date: midnight 8/1/2016

Instructions

1. The report should be written in a **printed PDF file** (not handwritten), and HW should be electronically submitted in a **ZIP file**.

2. The name format of the submission file should be HW3_ID_name.zip. For example: **HW3_012203443_Gil_Shamai.zip**. The name of the PDF file should be HW3_name.PDF. For example: **HW3_Gil_Shamai.pdf**.

3. Submit through the webcourse system, or by mail to cs236861@gmail.com with subject **"HW3 submission"**.

4. For implementation exercises, hand in all relevant and documented .m files, and relevant images as .png/.gif files. Each coding exercise **must include a running script** whose name is given in the dry part, and be reasonably internally documented.

5. Solitary hand in - no couples. In case of doubt - it is always best to ask. **HW copies will not be accepted**.

Problem 1: C/ C++ assignment

Note that a *running* MATLAB code will be checked. In this exercise you will implement the Fast Marching in C / C++ and link it with MATLAB as a matlab executable (mex). In any case, your implementation is required to solve the maze in at most 1 minute.

Implement the Fast Marching algorithm for weighted Euclidean domains. Write a MATLAB (mex) function with the following matlab interface

```
function [T] = fmm (F, src, t0, display)
```

accepting as the input an array of weights `F` (propagation "slowness", inverse proportionally to propagation velocity), a vector of source point indices `src` and the corresponding vector of initial values `t0`. The function solves the Eikonal equation

$$\|\nabla T\| \;=\; F \quad , \quad T(x_i) = T_i \quad \forall x_i \in X_0$$

and returns the distance map `T`.

Use an additional `display` argument to allow two display modes:

`'silent'` produces no output.

`'iter'` draws the current distance map `T` and the point attributes (Black, Green, Red) at each iteration.

Use `drawnow` to redraw the figure in a running process. Use `mexCallMATLAB` inside the C++ code to run matlab functions as part of your `mexFunction`. You can use STL data algorithms or downloaded modules to keep the list (priority queue) of alive (Red) points.

The following tasks should be done in Matlab.

1. Calculate the distances inside the maze given as 'maze.mat', using FMM and a source point at $(x, y) = (815, 384)$. Submit the result in 'maze_dist.mat'.

2. Find the shortest path from inside the maze, at $(815, 384)$, to the point $(9, 234)$, outside of it. Plot the shortest path on the top of the maze image.

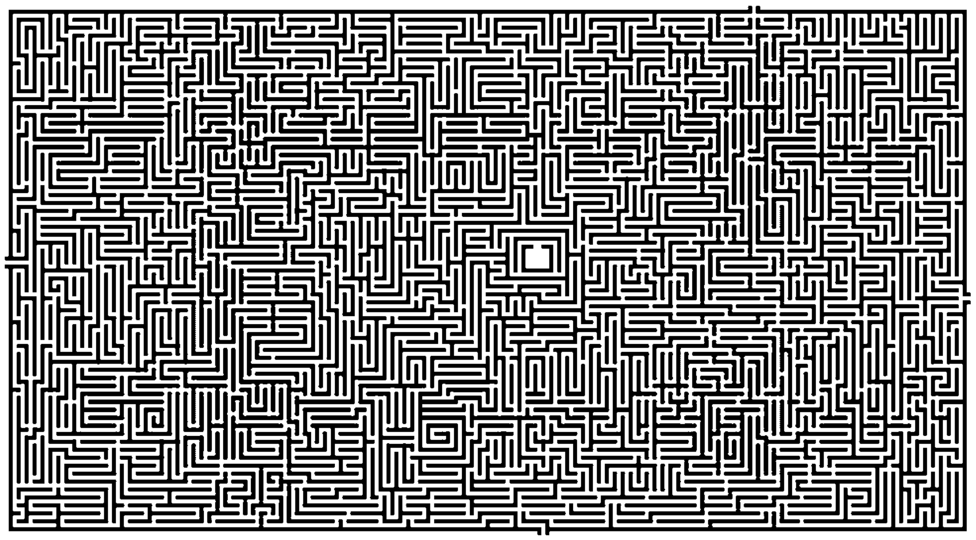
Figure 1: Maze.

3. Now, rotate the maze by 45°, and re-calculate the distances from the same (transformed) point inside the rotated maze. Compare the distances calculated for the two versions of the maze. Submit the result in 'maze45_dist.mat'.

4. The optical path length (OPL) is defined as the product of the Euclidean length of the path of light, and the index of refraction of the medium through which it propagates (denoted by n)

$$OPL = \int_C n(s)ds$$

Use FMM to calculate the OPL between two points, one inside a pool with water, with coordinate $(x, y) = (400, 500)$ and one outside, with coordinate $(x, y) = (1, 1)$, using the indices of refraction given in 'pool.mat'. Plot the shortest path on the top of the image of the indices of refraction.

Solve the maze given as 'maze.mat', using FMM, finding the shortest path from inside the maze, at $(815, 384)$, to the point $(9, 234)$, outside of it.

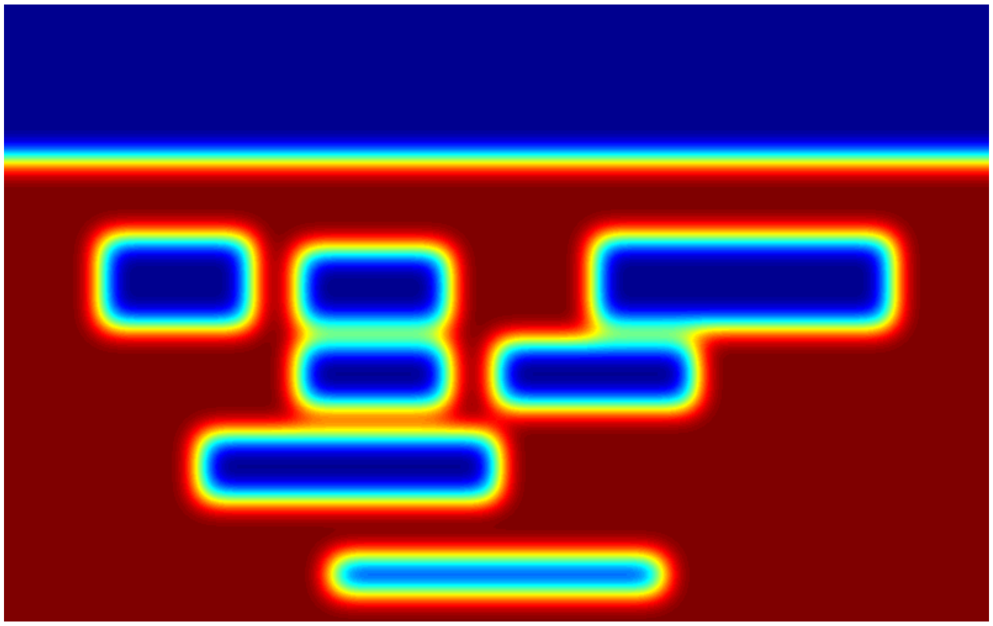

Figure 2: Indices of refraction given in the 'pool.mat'.

Problem 2

Consider the curved material in Figure 3, which consists of edges of a $1 \times 1 \times 1$ cube. Suppose we put this material into soup water and then extract it, so that a thin layer of soup appears, where its boundaries are constrained to the curve. It is known that this soap surface have minimal area. In this question, we will attempt to visualize this surface.

1. Write the explicit Mean curvature flow scheme from Tutorial 6, for updating Z_t .
2. Create an initial surface $S(x, y) = (x, y, z(x, y))$ with boundaries defined by the given curved material, and show it. Note that you should rotate the curve so that the function $z(x, y)$ is valid.
3. Compute the evolution of the surface using the explicit scheme, and show the evolution of the surface at several different time steps. Show

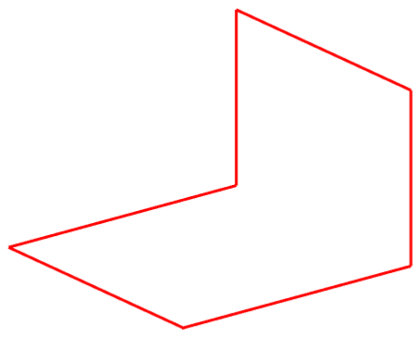

Figure 3:

the result minimal soap surface after it converges.

4. If we change the curve to the one in Figure 4, how would the soap surface look?

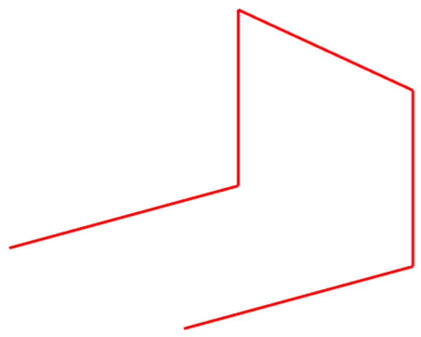

Figure 4:

5. **Bonus (7 points)**. Suppose we change the curved material so that it looks like the one in Figure 5. Still, it consists of edges of a $1 \times 1 \times 1$ cube.

   Visualize the mean curvature flow of the surface constrained to the new curve, and show the surface with the minimal area.

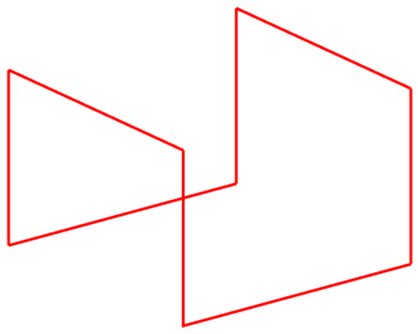

Figure 5:

Problem 3

1. Create a curve that looks like the one in Figure 6.

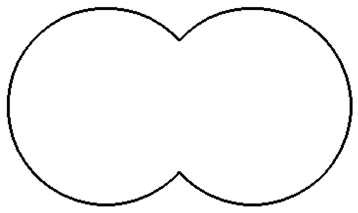

Figure 6: A 2D curve.

2. Implement the flow $C_t = \overrightarrow{N}$ using level sets evolution, so that the curve shrinks. Show the evolution of the curve at several different time steps.