Problem Set 4 – Shortest Path Problems

Problem 4.1. Fast marching methods produce a first-order accurate approximation of a distance map on a mesh, which is itself a first-order accurate approximation of the underlying smooth surface. There exist algorithms that compute the exact distance map on meshes. The advantage of such algorithms is that exact geodesic distances on a mesh are second-order accurate approximation of the geodesic distances on the smooth surface. Demonstrate this claim on a simple one-dimensional example.

Hint: As the example use a circle and its approximation by a perfect polygon.

Definitions. Practical numerical algorithms use finite precision arithmetics to perform computations. As consequence, round-off and truncation errors can propagate from one computation to another, and possibly accumulate into large errors. The following definitions concern this phenomenon in the context of fast marching: Let $t$ be a triangle in a triangular mesh formed by the vertices $x_1$, $x_2$, and $x_3$. The fast marching update step can be thought of as an operator $d_3 = U_t(d_1, d_2)$ that given the values $d_1 = d(x_1)$ and $d_2 = d(x_2)$ of the distance map at $x_1$ and $x_2$, returns the value $d_3$ of the distance map at $x_3$. The error growth of $U$ is defined as

$$\sigma(U) = \max_t \sup_{d_1, d_2 \geq 0} \max \left\{ \left| \frac{\partial U_t(d_1, d_2)}{\partial d_1} \right|, \left| \frac{\partial U_t(d_1, d_2)}{\partial d_2} \right| \right\}.$$ 

A scheme with $\sigma(U) \leq 1$ damps errors and is therefore said to be numerically stable.

Problem 4.2. Show that the fast marching update scheme presented in the lecture is numerically stable.

Hint: Express the derivative of $d_3$ with respect to $d_1$ and $d_2$ and show that its magnitude is always bounded by 1.

Problem 4.3. In the lecture, we have derived the fast marching update based on the planar wavefront model expressed by the distance function $d(x) = n^T x + p$, where $p$ is the source offset and $n$ is the propagation direction. A different model is that of a spherical wavefront expressed by $d(x) = \|x - x_0\|$, where $x_0$ is a source point. Derive an update scheme for the spherical wavefront explicitly expressing

1. the update equation;
2. consistency conditions;
3. monotonicity conditions;
4. error growth.

Explain whether this scheme always guarantees to produce a consistent update. Is the scheme numerically stable?

In the lecture, we studied the problem of computing the metric induced by the Riemannian length structure

\[ L(\Gamma) = \int_{0}^{1} \sqrt{\langle \dot{\Gamma}(t), \dot{\Gamma}(t) \rangle_{\Gamma(t)}} \, dt \]

We have seen that a distance map in the sense of this metric can be modeled as a wave propagating in medium with constant unit velocity and described by the viscosity solution of the eikonal equation,

\[ \|\nabla_X d\| = 1. \]

In many applications, it is useful to remove the unit velocity assumption allowing non-homogenous medium. The following questions deal with such a generalization.

**Problem 4.4.** Let us be given a surface \( X \) with a Riemannian metric and let \( v : X \to (0, \infty) \) be a positive scalar velocity field attached to it. We define the length structure as

\[ L(\Gamma) = \int_{0}^{1} \frac{1}{v(\Gamma(t))} \sqrt{\langle \dot{\Gamma}(t), \dot{\Gamma}(t) \rangle_{\Gamma(t)}} \, dt, \]

which can be thought of as the time a wave takes to propagate along the trajectory \( \Gamma \). Show that the wavefront propagation time (i.e., the distance map associated with \( L(\Gamma) \)) is given by the viscosity solution of

\[ \|\nabla_X d\| = \frac{1}{v}. \]

**Hint:** Express the non-homogenous length structure \( L(\Gamma) \) as a homogenous length structure resulted from a point-wise scaled Riemannian metric.

**Bonus:** Show the eikonal equation corresponding to the length structure

\[ L(\Gamma) = \int_{0}^{1} \frac{1}{v(\Gamma(t)) + u(\dot{\Gamma}(t))} \sqrt{\langle \dot{\Gamma}(t), \dot{\Gamma}(t) \rangle_{\Gamma(t)}} \, dt. \]

What is the physical meaning of the velocity field \( u(\dot{\Gamma}) \)?

**Problem 4.5.** Derive a fast marching update scheme for the solution of the eikonal equation from the previous problem. Express the monotonicity and consistency conditions.
Problem 4.6. (Matlab) Implement a fast marching algorithm for the solution of the non-homogenous eikonal equation on a parametric surface. Assume the parametrization domain is the rectangle \([1, M] \times [1, N]\) sampled on an orthogonal Cartesian grid with unit step.

**Input:** \(X, Y\) and \(Z - M \times N\) matrices holding the \(x\)-, \(y\)-, and \(z\)-coordinates of the surface; \(V\) – an \(M \times N\) matrix holding the velocity field; \(D_0\) – an \(M \times N\) matrix holding the boundary conditions (for distance map from a point \((i, j)\), use \(D_0(i, j) = 0\) and \(D_0 = \infty\) elsewhere).

**Output:** \(D\) – an \(M \times N\) matrix holding the computed distance map.

Use 8-neighbor connectivity and ignore problems with obtuse triangles. Use either heap- or raster scan-based grid visiting order (in the latter case, implement a proper stopping condition).

**Test data:** Show the output of your code on the surface from

http://tosca.cs.technion.ac.il/data/surface.mat

Use velocity fields stored in the matrices \(V_1, \ldots, V_4\) with the corresponding boundary conditions from the matrices \(D_1, \ldots, D_4\). Show the distance maps both on the 3D surface and in the parametrization domain (as an image). Use Matlab

\[
surf(X,Y,Z,D); \quad \text{axis image; shading interp};
\]

calls to visualize the 3D results.

Problem 4.7. (Matlab) Implement a gradient descent algorithm for first-order approximation of geodesic paths under the assumptions of the previous problem.

**Input:** \(X, Y\) and \(Z - M \times N\) matrices holding the \(x\)-, \(y\)-, and \(z\)-coordinates of the surface; \(V\) – an \(M \times N\) matrix holding the velocity field; \((i_0, j_0)\) and \((i_1, j_1)\) – two \(2 \times 1\) vectors holding the parametrization coordinates of the starting and ending points of the geodesic.

**Output:** \(L\) – a \(3 \times L\) matrix each of which columns holds the extrinsic coordinates of linear segment of the computed geodesic.

Show the output of your code on the surface from the previous problem with several randomly selected starting and ending points spanning sufficiently long and interesting paths on the surface. Show the geodesics both on the 3D surface and in the parametrization domain. Use Matlab

\[
surface(X,Y,Z,V); \quad \text{axis image; shading interp};
\]

\[
hold on; \quad \text{plot3(L(), L(), L(), 'k'); hold off};
\]

calls to visualize the 3D results.