Deriving Geodesics on a 3D Mesh from the Heat Equation

Matthieu Boyer

25 janvier 2025

Table des matières

Acknowledgement : I have reused a lot of the structure of [CWW12] as I find their article well thought and clear.

1 Introduction

Let us all remember our beloved heat equation :

$$\Delta H = \frac{\partial}{\partial t} H \tag{1}$$

This simple partial derivative equation encompasses the spreading of heat over time and can be described using the *heat kernel*, $k_{t,x}(y)$ which measures the heat transferred from

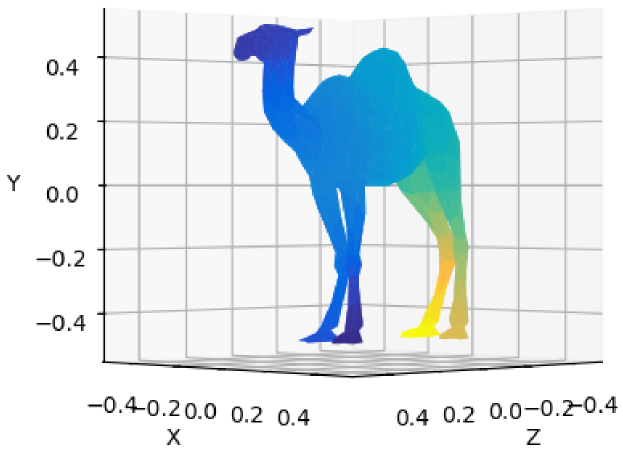

FIGURE 1 – An application of our method : a dromedary stepping on a hot rock

a source x to a target y after time t. We can explicitely derive the heat kernel on $\mathbb{R}^n$ to be :

$$k_{t,x}(y) = (4\pi t)^{-n/2} \exp\left(-\frac{1}{4t}\|x-y\|^2\right) \tag{2}$$

which leads to a particular setting of the Varadhan's formula, linking geodesics to the heat kernel. Indeed, in the euclidean space $\mathbb{R}^n$, the geodesic metric $d_{\mathbb{R}^n}(x, y)$ is defined by $\|x - y\|$ for the euclidean norm. The geodesic metric between x and y is the length of the shortest path (the geodesic) between x and y on our manifold. In 1967, Varadhan showed the following :

Théorème 1.1 — **[Var67]** If (M, g) is a complete Riemannian manifold, and $k_{t,x}(y)$ is the associated heat kernel, converging to $\delta_x(y)$ at small times, then :

$$-4t \log k_{t,x}(y) \xrightarrow[t\to 0]{} d(x, y)^2 \tag{VF}$$

where d is the distance function on the manifolds.

This equation comes, intuitively, from the fact that a solution of the heat equation spreads in a characteristic time t along a characteristic distance d will, numerically, verify $t \propto d^2$. The main idea we will present here, developed in [CWW12] is that looking at a manifold is, numerically, the same as looking at a triangulation of the manifold, and that we can thus approximate the geodesics on our object by solving the heat equation.

2 Related Work and Applications

2.1 Computing Distances

Other ways to approach distance computations have already been tried. The most widely used approach is solving the eikonal equation

$$\|\nabla\varphi\| = 1 \text{ subject to boundary conditions } \varphi_{|\gamma} = 0 \tag{3}$$

but this leads to a nonlinear and hyperbolic formulation, rendering it difficult to solve directly. Algorithms solving the equation on implicit surfaces such as the Gauss-Seidel scheme for solving linear systems, using fast marching and fast weeping work in $\mathcal{O}(n \log n)$ and $\mathcal{O}(n)$ but do not reuse information, as the distance from different subsets γ is computed from scratch each time, leading to greater amortized costs than what we will be able to achieve here.

It is important to note that a method developed in [GR14] using a time-independent version of the Schrödinger's equation resembles a lot our method, though their computation is limited to regular grids and R^n where solutions take a special form, and it needs an abitrary precision on Planck's constant to determine accurate.

2.2 Applications

Determining the geodesic metric has applications in multiple domains. For example, in [LB81], Lantuejoul and Beucher present multiple applications of the geodesic metric to image analysis. They suggest a method to reconstruct particles and create skeletons using the geodesic distance which provides an easier time dealing with connectivity problems. In [PC09], applications of the geodesic metric are proposed to the A^* algorithm and heuristically driven propagation as well as meshing applications. In more industrial applications, the computer-aided-design software Solidworks possesses a geodesic offset tool, to help designer add curves and decals to their models.

3 The Method

Let us first describe our method in terms of smooth manifolds, we will explore computation aspects in the next sections. This gives us algorithm 1.

Algorithme 1 The Heat Method

```
1. Integrate $\dot{u} = \Delta u$ for some fixed time $t$.
2. Evaluate the vector field $X = \frac{-\nabla u}{\|\nabla u\|}$.
3. Solve the Poisson equation $\Delta\varphi = \nabla \cdot X$.
```

φ approximates geodesic distance, approaching the true distance as t goes to zero. Since solutions to the Poisson equation are unique up to a constant, we simply make it so the smallest distance is 0. The initial conditions u_0 allow us to compute the distance to a single source point $x \in M$ by setting $u_0 = \delta(x)$ and to any piecewise submanifold γ by taking its generalized Dirac.

To prove the correction of the algorithm, remember the eikonal equation tells us that the gradient of the true geodesic metric is of unit norm. Then, if we know the gradient of our heat approximation ∇u_t points in the right direction (parallel to $\nabla\varphi$), we just need to solve $\Delta\varphi = \nabla \cdot \frac{-\nabla u}{\|\nabla u\|}$ to compute the distances.

However, our algorithm will not provide accurate results for now, if we do not add ways to discretize both time and space.

3.1 Discretizing Time

In practice, we cannot directly integrate the heat equation for step 1 of the heat method. Thus, to simplify the calculations for ourselves, we discretize the equation using a backward Euler step for a certain time step t. Thus, integrating the heat equation amounts to solving :

$$(\mathrm{id} - t\Delta)\, u_t = u_0 \tag{4}$$

over the entire domain M, where id is the identity function (spatial discretization will be discussed in the next subsection).

Then, looking back at the problem of computing distances from x, that is $u_0 = \delta(x)$ or :

$$\begin{aligned} (\mathrm{id} - t\Delta)\, v_t &= 0 \quad \text{on} \quad M \setminus \{x\} \\ v_t &= 1 \quad \text{on} \quad x \end{aligned} \tag{BP}$$

as established by Varadhan in is proof of VF, we get

$$-\frac{\sqrt{t}}{2} \log v_t \underset{t\to 0}{=} \varphi \tag{δt}$$

which proves the correction of the algorithm with the implicit backward Euler step.

3.2 Discretizing Space

The method can be applied to any domain with a discrete gradient ∇, divergence $\nabla\cdot$ and Laplacian Δ. As I've only considered simplicial meshes embedded in $\mathbb{R}^3$ in the implementation, since I wanted to focus on the graphics part, this is the only discretization we will look at in this theory part. Let M here be a simplicial complex represented by a triangulation $\left(\mathcal{V}, \mathcal{F} \subseteq \mathcal{V}^3\right)$. We will first remember the expressions of a few operators. For this purpose, let $u \in \mathbb{R}^{|V|}$ be a piecewise linear function. The gradient of u in a given triangle f is :

$$\nabla_f u = \frac{1}{2A_f} \sum_{i\in f} u_i \left(N \wedge e_{j_i}\right) \tag{5}$$

where A_f is the area of the face, N is its unit normal, e_{j_i} is the edge vector of the face opposite to vertex i, oriented counter clockwise, and u_i is the value of u at vertex i. To store the gradient, we can also see it as a $3\,|\mathcal{F}| \times |\mathcal{V}|$ sparse matrix (since it's actually 3, $|\mathcal{F}| \times |\mathcal{V}|$ matrices). If we denote $J\vec{e}_{j_i}$ the vector $N \wedge e_{j_i}$ (which is the vector e_{j_i} rotated to point inside the face) :

$$\begin{aligned} \left(\tilde{\nabla}\right)_{i,j} &= \left(J\vec{e}_{j_i}\right)_1 \\ \left(\tilde{\nabla}\right)_{i+|\mathcal{F}|,j} &= \left(J\vec{e}_{j_i}\right)_2 \\ \left(\tilde{\nabla}\right)_{i+2|\mathcal{F}|,j} &= \left(J\vec{e}_{j_i}\right)_3 \end{aligned} \tag{6}$$

Here we actually didn't compute the gradient but only the sum part of the operator, we then have

$$\nabla = \frac{1}{2A}\tilde{\nabla} \tag{7}$$

where :

$$\frac{1}{2A}_{f+k|\mathcal{F}|,f+k|\mathcal{F}|} = 1/2\,\|e_{f_1} \times e_{f_2}\| \tag{8}$$

is a diagonal square matrix of size $3\,|\mathcal{F}|$ whose coefficients are defined as above for $0 \leq k \leq 2$.

Then, we can compute the divergence operator's matrix as the transpose of the gradient for the face area dot product :

$$\nabla\cdot = \nabla^T A \tag{9}$$

where A is defined as previously as a $3\,|\mathcal{F}|$ diagonal matrix containing the areas of the faces. Then, we can easily compute the laplacian operator as the composition of the gradient and the divergence :

$$\Delta = \nabla \cdot \nabla = \nabla^T A \nabla \tag{10}$$

which is coherent with the expression we might derive directly :

$$(Lu)_i = \frac{1}{2VA_i} \sum_{j \in \mathcal{N}(i)} \left(\cot \alpha_{i,j} + \cot \beta_{i,j}\right)(u_j - u_i) \tag{11}$$

where VA_i is a third of the sum of all triangles incident on vertex i, the sum is taken over neighboring vertices and $\alpha_{i,j}, \beta_{i,j}$ are the angles opposing the corresponding edge. We will call L_C the sum part of this operator. Then, retrieving our distance function amounts to solving : $(VA - tL_C)\,u = u_0$ and then $L_C\varphi = \nabla \cdot \left(\frac{-\nabla u}{\|\nabla u\|}\right)$.

3.3 Time Step

The accuracy of our method relies on the choice of time step t. Equation δt tells us that having $t \to 0$ will lead us to better accuracy, when space is supposed continuous. However, when we discretize space, we see that the limit solution to the boundary problem BP will be a function of the combinatorial distance, independent on our Laplacian discretization :

Théorème 3.1 Let $G = (V, E)$ be the graph induced by the sparsity pattern of any real symmetric matrix A, and consider the linear system :

$$(\mathrm{Id} - tA)\,u_t = \delta$$

where δ is a Kronecker delta at a source vertex $u \in V$ and $t > 0$. Then, generically :

$$\boxed{\varphi = \lim_{t \to 0} \frac{\log u_t}{\log t}}$$

where $\varphi \in \mathbb{N}^{|V|}$ is the graph distance between each vertex $v \in V$ and the source vertex u.

Démonstration. Let σ be the operator norm of A. Then, for $t < \frac{1}{\sigma}$, $B = I - tA$ is invertible and we have :

$$u_t = \sum_{k=0}^{\infty} t^k A^k \delta$$

Let $v \in V$ be n edges away from u and consider $r_t = |R_v| / |s_v|$ where :

$$s_v = (t^n A^n \delta)_v \neq 0, \qquad R_v = \left(\sum_{k=n+1}^{\infty} t^k A^k \delta \right)_v$$

Since :

$$|s| \leq \sum_{k \geq n+1} t^k \left\| A^k \delta \right\| \leq \sum_{k \geq n+1} t^k \sigma^k$$

and thus :

$$r_t \leq \frac{t^{n+1} \sigma^{n+1} \sum_{k=0}^{+\infty} t^k \sigma^k}{t^n \left(a^n \delta \right)_v} = c \frac{t}{1 - t\sigma}$$

where $c = \sigma^{n+1} / \left(A^n \delta \right)_v$ does not depend on t. Therefore, $\lim_{t \to 0} r_t$, that is, only the first term s_0 is significant as t goes to zero. Moreover :

$$\log s_0 = n \log t + \log \left(A^n \delta \right)_v$$

is dominated by the first term as t goes to 0 hence $\log \left(u_t \right)_v / \log t$ approaches n, the combinatorial distance on the graph. ■

Therefore only edge proximity is influential on the accuracy of our mesh, when $t \to 0$. We will provide an estimation on how so much this is true in the next section. To find an optimal expression t^* is difficult, and we thus use an estimate $t = h^2$ where h is the expectation of the length of an edge.

4 Implementation and Results

In this section, we will look at the implementation and efficiency of the heat method on manifolds embedded in the 3D-space.

4.1 Difficulties in Complexity

Implementation for this project was done in python. We used `OFF` to represent our objects. The implementation follows the steps detailed in the proof of correction for the algorithm on triangulated meshes. When importing the object, we create its set of faces and vertices, and use the formulas derived earlier to compute its gradient, divergence and laplacian operators. We will denote by m the number of faces and n the number of vertices. Each operation needed to compute the gradient is done in $\mathcal{O}(m)$, as there are always three vertices per face, and we assume our mesh to be connected. Then, while the complexity of our matrix product has not been increased, the use of a sparse matrix representation allows for the operations of our $3\,|\mathcal{F}| \times |\mathcal{V}|$ matrices to be multiplied in $\mathcal{O}\left(|mF|^{\omega}\right)$ as $|\mathcal{F}| = \Theta\left(|\mathcal{V}|\right)$ and especially allows our space complexity to stay in $\mathcal{O}\left(|\mathcal{F}|\,|\mathcal{V}|\right)$. Indeed, some of our meshes have up to 50×10^3 vertices, leading to 150000×150000 matrices of floats using a good precision to occupy over 22.5 GB if not sparsified, which goes to 7.5 GB when considering three full matrices and about 2.5 GB when sparsified.

While the asymptotic complexity of our algorithm is simply in $\mathcal{O}\left(|\mathcal{F}|^{\omega}\right)$ using $\mathcal{O}\left(|\mathcal{F}|\,|\mathcal{V}|\right)$ space, the difficulties encountered are taken into account when factoring the precision needed on the mesh for accuracy : the distance obtained by the heat method is approximately the combinatorial weighted distance for smaller values of t. Thus, having a fine mesh is a necessity for our method to be precise, which leads to larger and larger computation times.

Moreover, testing the implementation proved far more difficult than predicted, as computing by hand gradients and laplacians on graphs is quite a stretch.

4.2 Results

4.2.1 Computation Time

In figure 2 we present a plot of the time needed for the computation of the distances from the origin on the mesh defined by ε :

$$\mathcal{V} = \left\{ (k\varepsilon, k'\varepsilon) \;\middle|\; 0 \leq k, k' \leq \frac{1}{\varepsilon} \right\}$$
$$\mathcal{F} = \left\{ (i, i+1, i+n), (i, i+n, i+n-1) \;\middle|\; i \leq \frac{1}{\varepsilon^2} \right\} \cap \mathcal{V}^3$$

Here, the average distance between two connected points is thus given by :

$$h = \frac{4}{6}\varepsilon + \frac{2}{6}\sqrt{2}\varepsilon = \frac{2+\sqrt{2}}{3}\varepsilon \simeq 1.14 \times \varepsilon$$

and thus $t = 1.3\varepsilon^2$.

In the above figure we can see that on iteration i, the number of vertices is in i^2 and the complexity seems to fit in i^6.

4.2.2 Accuracy

Let us take our plane mesh defined by ε. The geodesic distance to 0 can be directly computed :

$$d(x) = \|x\| \text{ and on indices } d_\varepsilon(i) = \varepsilon \sqrt{\left(i \mod \frac{1}{\varepsilon}\right)^2 + \left(i // \frac{1}{\varepsilon}\right)^2}$$

where // and $\mod$ denote the dividend and the rest respectively in the euclidean division. We then compute the mean of difference between the theoretical distance d which we just derived and the heat-method-computed geodesic metric $\tilde{d}$.

Section 3.3 and Theorem 3.1 tell us that our difference should be around the difference between d and the combinatorial distance φ multplied by the step size ε :

$$\begin{aligned} d(i) - \varepsilon\varphi(i) =& \varepsilon \times \sqrt{\left(i \mod \frac{1}{\varepsilon}\right)^2 + \left(i // \frac{1}{\varepsilon}\right)^2} - \varepsilon \times \left(i \mod \frac{1}{\varepsilon}\right) + \left(i // \frac{1}{\varepsilon}\right) \\ \leq& \frac{\varepsilon}{2}\left(\left(i \mod \frac{1}{\varepsilon}\right) + \left(i // \frac{1}{\varepsilon}\right)\right) \end{aligned}$$

From figure 3, we get that :

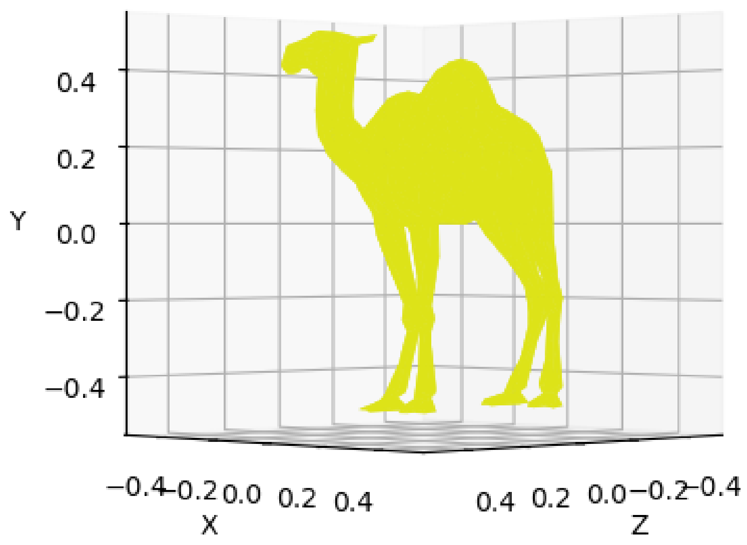

Figure 2 – Computation times for evenly spaced triangular mesh of step $1/i$ on the unit square.

Références

[CWW12] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. Geodesics in heat. *CoRR*, abs/1204.6216, 2012.

[GR14] Karthik S. Gurumoorthy and Anand Rangarajan. A fast eikonal equation solver using the schrödinger wave equation. *ArXiv*, abs/1403.1937, 2014.

[LB81] C. Lantuejoul and S. Beucher. On the use of the geodesic metric in image analysis. *Journal of Microscopy*, 121(1) :39–49, January 1981.

[PC09] Gabriel Peyré and Laurent D. Cohen. Geodesic methods for shape and surface processing. In João Manuel R. S. Tavares and R. M. Natal Jorge, editors, *Advances in Computational Vision and Medical Image Processing : Methods and Applications*, pages 29–56. Springer Netherlands, Dordrecht, 2009.

[Var67] S. R.S. Varadhan. On the behavior of the fundamental solution of the heat equation with variable coefficients. *Communications on Pure and Applied Mathematics*, 20(2) :431–455, May 1967.

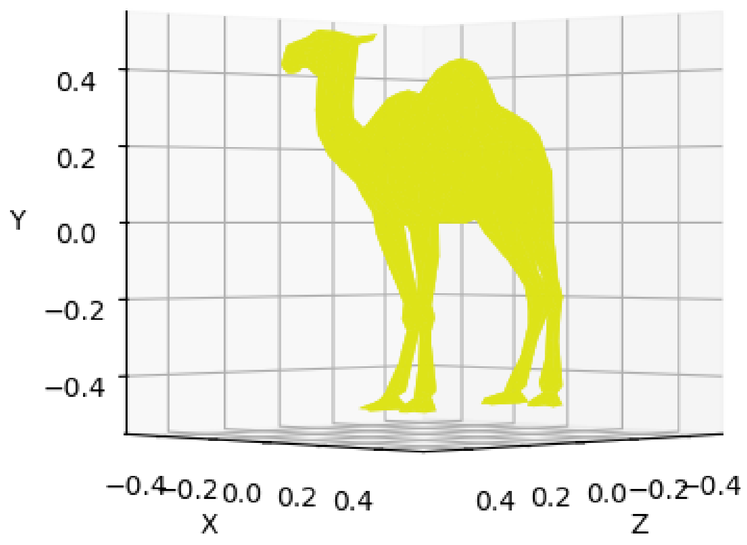

Figure 3 – Accuracy of the heat methods for regular meshes for the plane of step ε