A 3D+t Laplace operator for temporal mesh sequences

Victoria Fernández Abrevaya, Sandeep Manandhar, Franck Hétroy-Wheeler, Stefanie Wuhrer

A 3D+t Laplace operator for temporal mesh sequences

Victoria Fernández Abrevaya[a], Sandeep Manandhar[b,d], Franck Hétroy-Wheeler[c,d], Stefanie Wuhrer[c,d]

[a] *Universidad de Buenos Aires, Buenos Aires, Argentina*
[b] *Université de Bourgogne, Le Creusot, France*
[c] *Université Grenoble Alpes, Grenoble, France*
[d] *Inria, Grenoble, France*

Abstract

The Laplace operator plays a fundamental role in geometry processing. Several discrete versions have been proposed for 3D meshes and point clouds, among others. We define here a discrete Laplace operator for temporally coherent mesh sequences, which allows to process mesh animations in a simple yet efficient way. This operator is a discretization of the Laplace-Beltrami operator using Discrete Exterior Calculus on CW complexes embedded in a four-dimensional space. A parameter is introduced to tune the influence of the motion with respect to the geometry. This enables straightforward generalization of existing Laplacian static mesh processing works to mesh sequences. An application to spacetime editing is provided as example.

Keywords: Laplace operator, mesh animation, discrete exterior calculus

1. Introduction

The use of a discrete counterpart of the Laplace-Beltrami operator (the divergence of the gradient) for geometry processing has shown growing interests in the last decades. As explained by Sorkine in the case of 3D meshes [1], such an operator encodes the variation of a function around a given vertex, thus giving local information about the object under study. Possible applications include compression, watermarking, editing, segmentation, matching, retrieval, parameterization; multiple surveys review these applications in the case of 3D meshes [1, 2, 3]. Discrete Laplace operators are also widely used on graphs, for instance for applications in chemistry [4] and 2D images, mostly for edge detection and filtering [5].

In this paper we are interested in defining a discrete Laplace-Beltrami operator for temporal mesh sequences, that is to say sequences of surface meshes embedded in the Euclidean 3-dimensional space. Temporal mesh sequences, also called *mesh animations* or *3D videos*, are ubiquitous in various domains such as computer games, 3D movies or 3D television, to represent objects evolving through time [6]. Mesh sequences can be captured from real life scenes using multiple camera systems or generated using modelling software or physically-based simulation. In all cases, they may require time-consuming modifications, such as editing of some part of the geometry and/or the motion, to become usable in the production pipeline.

Although it may seem natural to process the geometry and the motion separately, we advocate here the use of a single operator, with a parameter to decouple time and space dimensions.We show that such an operator may lead to a variety of effects with a single formulation. Our discrete Laplace-Beltrami operator is defined by modelling mesh sequences as CW complexes embedded in a 4-dimensional space and using the Discrete Exterior Calculus (DEC) framework [7, 8]. Only one parameter α is to be chosen by the user: the one which balances the influence of geometry with respect to motion. We investigate the properties of this operator, in particular when α is big or small. We also explore a problem, as-rigid-as-possible mesh sequence editing, which can easily be expressed using this operator. Results show that a broad range of effects can be generated by tuning parameter α.

2. Related work

Following seminal works by Pinkall and Polthier [9] and Taubin [10], many discrete Laplace-Beltrami operators have been defined for static 3D triangle meshes, each with different properties. A popular choice is to use cotangent weights [10, 11]. This so-called *cotangent Laplacian* can be derived from the smooth Laplace-Beltrami operator on a 2-manifold shape using DEC [7, 8]. Our work can be considered as the extension of such operator with one more dimension.

Discrete Laplacians have also been defined on more general simplicial surfaces [12] and polygonal meshes [13], as well as on point clouds [14, 15, 16], volumetric models [17] and pseudomanifolds [18]. This last work also uses DEC. All these discrete Laplacians operate on static shapes. Other works have defined discrete Laplace-Beltrami operators on manifolds from a differential point of view, with a particular emphasis on convergence to the continuous operator [19, 20].

In the case of a moving shape, a couple of works have explicitly used a discrete Laplace operator. In the context of motion editing and retargeting, Le Naour et al. [21] propose to use a Laplace operator with Gaussian weights on the animation skeleton. Yang et al. [22] use a Laplacian with uniform weights to enhance details in a mesh sequence. We compare to these works in Section 5.5. To the best of our knowledge, our work

is the first to derive a discrete Laplace operator which is both geometry- and motion-dependent.

3. 3D+t DEC Laplacian

In this section the proposed definition of a Laplace operator for temporal mesh sequences is developed. Mesh sequences are defined as 3-dimensional CW complexes embedded in a 4D space. The Laplace operator is then derived from DEC on these complexes.

3.1. Embedding space

Let $\mathbb{E}$ be a 4-dimensional Riemannian manifold, equipped with a metric g such that the matrix of the metric tensor in any basis of vector fields on $\mathbb{E}$ is a diagonal matrix $G = Diag(\alpha, 1, 1, 1)$ with $\alpha > 0$.

In other words, if $X_1 = (t_1, x_1, y_1, z_1)$ and $X_2 = (t_2, x_2, y_2, z_2)$ are two vectors in $\mathbb{E}$, then the *inner product* of X_1 and X_2 is defined as $\langle X_1, X_2 \rangle = \alpha t_1 t_2 + x_1 x_2 + y_1 y_2 + z_1 z_2$. In particular, the *norm* of a vector $X = (t, x, y, z) \in \mathbb{E}$ is defined as $\|X\| = \sqrt{\alpha t^2 + x^2 + y^2 + z^2}$.

$\mathbb{E}$ represents the embedding space of our mesh sequences. The first coordinate t of a vector $X = (t, x, y, z) \in \mathbb{E}$ is called its *timelike coordinate*, while the three others coordinates x, y and z are called its *spacelike coordinates*. α is a user-defined parameter that describes the respective influence of space and time in the metric.

Note that α should be positive for $\mathbb{E}$ to be a Riemannian manifold. If $\alpha < 0$, G is no more positive definite.

3.2. Mesh sequence notations

A *temporal mesh sequence* is, in the most general case, a sequence $(M^1, \dots, M^K)$ of 2-manifold meshes. In this paper, we restrict to *temporally coherent mesh sequences*, that is to say sequences with a fixed connectivity, where there is a one-to-one correspondence between vertices (respectively, edges and faces) of successive meshes. Most temporal mesh sequences used in computer graphics are temporally coherent, since they are constructed from a single mesh that deforms over time.Various methods have been proposed to compute a temporally coherent mesh sequence which approximates a mesh sequence without explicit temporal coherence, see e.g. Allain et al. [23] and references therein.

In this paper, we use the following notations:

- $M^k = (V^k, E^k, F^k)$ is the k-th mesh of the input sequence and V^k, E^k and F^k are the sets of its vertices, edges and faces, respectively;
- v_i^k, v_j^k and v_l^k are vertices on M^k;
- V, E and F are the number of vertices, edges and faces of M^k, respectively;
- K is the number of meshes of the sequence.

Since we restrict to temporally coherent mesh sequences, V, E, and F are the same for all M^k.

3.3. Mesh sequence as a CW complex

We now model a temporally coherent mesh sequence as a CW complex embedded in $\mathbb{E}$.

Definition 3.1 (CW complex [24])**.** *A* CW complex *is a sequence (X^i) of i-dimensional spaces X^i inductively defined as:*

1. *a discrete set X^0, whose points are called* 0-cells*;*
2. *X^i is the disjoint union of X^{i-1} with a collection of i-dimensional disks, called i-*cells*. These disks are attached to X^{i-1} using continuous maps at their boundary.*

In case the sequence is finite, the dimension *of a CW complex is the greater dimension of its cells.*

The following property derives directly from the definition of a CW complex.

Property 3.2. *Let $MS = (M^1, \dots, M^K)$ be a temporally coherent sequence of 2-manifold triangular meshes. $\forall k \geq 1, k \leq K$, let $t^k \in \mathbb{R}$ be the timelike coordinate of all vertices of M^k, such that $t^1 < t^2 < \dots < t^K$. Then the union of all M^k, together with:*

- *$V(K-1)$ additional edges between all vertices v_i^k and $v_i^{k+1}, 1 \leq i \leq V, 1 \leq k \leq K-1$,*
- *$E(K-1)$ additional 2-cells between all edges $v_i^k v_j^k$ and $v_i^{k+1} v_j^{k+1}, 1 \leq i \leq E, 1 \leq k \leq K-1$,*
- *$F(K-1)$ additional 3-cells between all faces $v_i^k v_j^k v_l^k$ and $v_i^{k+1} v_j^{k+1} v_l^{k+1}, 1 \leq i \leq F, 1 \leq k \leq K-1$,*

forms a 3-dimensional CW complex embedded in $\mathbb{E}$.

In the following, we call such a CW complex a *temporally coherent mesh sequence embedded in* $\mathbb{E}$. Edges $v_i^k v_i^{k+1}$ of a temporally coherent mesh sequence embedded in $\mathbb{E}$ are subsequently called *temporal edges*. Other edges are called *spatial edges*. Figure 1 (a) depicts part of a temporally coherent mesh sequence embedded in $\mathbb{E}$.

3.4. Discrete Laplace operator

A discrete Laplace operator for 0-forms on temporally coherent mesh sequences is now constructed. This is done using the DEC framework [25, 7, 8]. Since in our modelling temporal 2-cells are not triangles but (skew) quadrilaterals and 3-cells are not tetrahedra, our CW complex is not a simplicial complex. DEC can nonetheless be applied since its structure is manifold-like by construction [18]: cutting each temporal 2-cell into two triangles would generate a 3-manifold tetrahedrisation. However, this cutting is non-canonical in the sense that several tetrahedrisations could be created from the same sequence with different cuttings, leading to possibly different Laplace operators. The DEC framework is thus applied directly to this CW complex rather than to some tetrahedrisation.

3.4.1. Discrete Exterior Calculus in a 4D space

We refer to Crane et al. [8] for an introduction to the discrete Laplace-Beltrami operator on triangular meshes, and its discretisation through DEC. In short, we start with a 0-form f, that is to say a function which associates a number to each

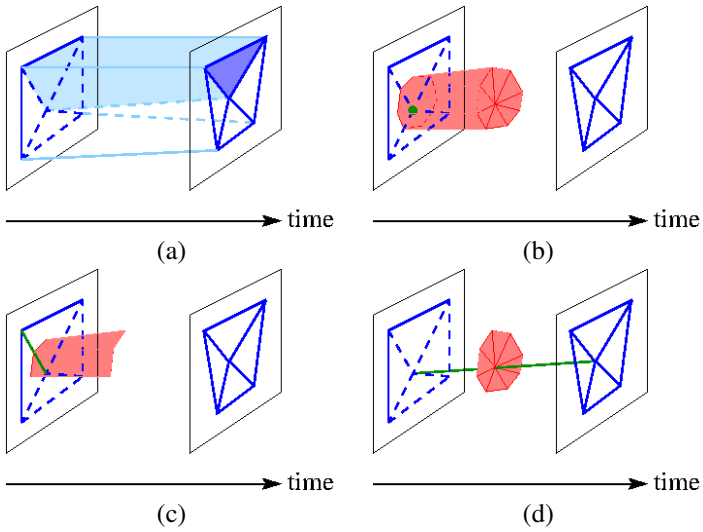


Figure 1: (a) Modelling of a temporally coherent mesh sequence embedded in $\mathbb{E}$. In dark blue are shown spatial edges at two successive time instants t^k and t^{k+1} and a face of M^{k+1}. In light blue are shown corresponding temporal edges and a 3-cell of the CW complex. (b,c,d) Barycentric dual cells (in red) of a (b) vertex, (c) spatial edge and (d) temporal edge, shown in green. Only parts of the cells with timelike coordinates between t^k and t^{k+1} are shown.

vertex v_i^k of the input temporally coherent mesh sequence MS embedded in $\mathbb{E}$.

The *discrete exterior derivative* $\mathbf{d}$ is the discrete counterpart of the gradient. It allows to integrate the derivative of f along the edges of MS. For an edge $v_i^k v_j^l$ with $|k-l| \leq 1$, it can be expressed as:

$$\mathbf{d}f(v_i^k v_j^l) = \int_{v_i^k v_j^l} df = \langle f, v_j^l \rangle - \langle f, v_i^k \rangle, \tag{1}$$

where $\langle f, v_i^k \rangle$ denotes the number associated to v_i^k by the 0-form f.

The *discrete codifferential* δ is the discrete counterpart of the divergence. This operator can be expressed as $\delta = *\mathbf{d}*$ with $*$ another operator called the *discrete Hodge star*. The discrete Hodge star is a map defined on 0-forms (in our case) such that, for any simplex σ:

$$\frac{1}{|\sigma|}\langle f, \sigma \rangle = \frac{1}{|\star\sigma|}\langle *f, \star\sigma \rangle, \tag{2}$$

where $\star\sigma$ denotes the *dual* of the simplex σ (see below) and $|\sigma|$ denotes its volume. In our case, the simplices are the vertices and the edges of the mesh sequence. By convention, the volume of a vertex is equal to 1.

The discrete Laplace-Beltrami operator $\triangle_u$ of f on the vertices v_i^k of MS is finally defined as the divergence of the gradient: $\triangle_u = \delta\mathbf{d} = *\mathbf{d}*\mathbf{d}$ [7, 8]. This leads to:

Definition 3.3 (Unsymmetrised discrete Laplace operator [7, 8])**.** *The* unsymmetrised discrete Laplace operator $\triangle_u$ *of a function f defined on the vertices v_i^k of a temporally coherent mesh sequence MS embedded in $\mathbb{E}$ is defined by:*

$$\frac{1}{|v_i^k|}\langle \triangle_u f, v_i^k \rangle = \frac{1}{|\star v_i^k|} \sum_{v_i^k v_j^l \in MS} \frac{|\star v_i^k v_j^l|}{|v_i^k v_j^l|}(f(v_i^k) - f(v_j^l)) \tag{3}$$

where $\star v_i^k$ denotes the dual of vertex v_i^k, $\star v_i^k v_j^l$ is the dual of edge $v_i^k v_j^l$ (thus $|k-l| \leq 1$), and $|c|$ denotes the oriented volume of any cell c, whatever its dimension.

By definition, the dual $\star v_i^k$ of a vertex v_i^k is a 3-cell whose vertices are the centres of incident (spatial and temporal) edges, (spatial) triangles, (temporal) quadrilaterals and 3-cells. The dual $\star v_i^k v_j^l$ of an edge $v_i^k v_j^l$ is a 2-cell whose vertices are the centres of incident triangles or quadrilaterals and 3-cells.

Figure 1 shows the dual cells of a vertex, a spatial edge and a temporal edge. Note that the dual of a spatial edge (Figure 1 (c)) is a set of four temporal quadrilaterals (two only for the first and the last meshes of the sequence). The dual of a temporal edge (Figure 1 (c)) is a set of triangles sharing the same timelike coordinate. The dual of a vertex (Figure 1 (b)) is a set of 3-cells with 6 vertices, defined by temporal quadrilaterals and spatial triangles.

The centre of a k-cell is chosen to be the isobarycentre (i.e., centroid) of the cell, as in prior works [26, 18]. Note that in general it is not possible to define circumcentres, thus circumcentric duals as in some previous works [25, 7], because of the quadrangular temporal 2-cells.

The area of a temporal quadrilateral is not properly defined since this quadrilateral is skew: its four points are not necessarily coplanar. In our case, we only consider quadrilaterals expressing the motion of an edge $v_i^t v_j^t$ from timelike coordinate $t = t^k$ to timelike coordinate $t = t^{k+1}$. As a consequence, we can define the area of the corresponding 2-cell as the integral of the length of this edge over time, from t^k to t^{k+1}:

Definition 3.4 (Area of a temporal 2-cell)**.** *Let v_i^k, v_j^k, v_j^{k+1} and v_i^{k+1} be the ordered vertices of a temporal quadrilateral $Q_{i,j}^k$. Let t^k be the timelike coordinate of v_i^k and v_j^k, and t^{k+1} be the timelike coordinate of v_j^{k+1} and v_i^{k+1}. Let all vertices be interpolated linearly $\forall t \in [t^k, t^{k+1}]$ as $v_i^t = \frac{t-t^k}{t^{k+1}-t^k}(v_i^{k+1} - v_i^k) + v_i^k$. Then,*

$$|Q_{i,j}^k| = \int_{v_i^k}^{v_i^{k+1}} \|v_i^t v_j^t\| dv_i^t \tag{4}$$

This integral can be expressed as $B\int_A^{1+A} \sqrt{u^2 + C^2} du$ with A and B two constants, $C = \|v_i^k v_j^k \times v_i^{k+1} v_j^{k+1}\|_2$ and $\|.\|_2$ the Euclidean distance in the 3D Euclidean space (not in the 4D space $\mathbb{E}$). This in turn can be expressed in a closed form. See Appendix (a) for the details.

Similarly, the dual $\star v_i^k$ of a vertex v_i^k can be expressed as the union of several 3-cells $F_{i,j,l}^k = v_i^k v_j^k v_l^k v_i^{k+1} v_j^{k+1} v_l^{k+1}$ expressing the displacement of triangles $v_i^t v_j^t v_l^t$ from timelike coordinate $t = t^k$ to timelike coordinate $t = t^{k+1}$. We can thus define the volume of such a 3-cell as the integral of the area of the triangle $v_i^t v_j^t v_l^t$ over time:

Definition 3.5 (Volume of a temporal 3-cell)**.** *Let $v_i^k v_j^k v_l^k$ and $v_i^{k+1} v_j^{k+1} v_l^{k+1}$ be the triangles defining a temporal 3-cell $F_{i,j,l}$. Let t^k be the timelike coordinate of v_i^k, v_j^k and v_l^k, and t^{k+1} be the timelike coordinate of v_i^{k+1}, v_j^{k+1} and v_l^{k+1}. Let all vertices be*

interpolated linearly $\forall t \in [t^k, t^{k+1}]$ *as* $v_i^t = \frac{t-t^k}{t^{k+1}-t^k}(v_i^{k+1} - v_i^k) + v_i^k$. *Then,*

$$|F_{i,j,l}^k| = \int_{v_i^k}^{v_i^{k+1}} Area(v_i^t v_j^t v_l^t) dv_i^t \tag{5}$$

We give a detailed expression of $|F_{i,j,l}^k|$ in Appendix (b).

3.4.2. Symmetrisation

Following Vallet and Lévy [27], it can be noticed that the operator $\triangle_u$ is not symmetric but can be symmetrised. The inner product on 0-forms is defined by the diagonal matrix $\star_0$ with elements $\frac{|\star v_i^k|}{|v_i^k|}$, that is to say the volumes of the vertex dual cells since for any vertex $v_i^k, |v_i^k| = 1$. The following symmetric Laplace operator can thus be defined.

Definition 3.6 (Discrete Laplace operator on mesh sequences)**.** *Let MS be a temporally coherent mesh sequence embedded in* $\mathbb{E}$. *The operator* $\triangle$ *on* 0*-forms on MS defined as*

$$\triangle = \star_0^{1/2} \triangle_u \star_0^{-1/2} \tag{6}$$

is called the Laplace operator *on MS*.

From Equation (3) and Equation (6) the following expression is derived.

Property 3.7. *Let* f *be a function defined on vertices* v_i^k *of a temporally coherent mesh sequence MS. Then:*

$$\langle \triangle f, v_i^k \rangle = \sum_{v_i^k v_j^l \in MS, |k-l| \leq 1} \frac{1}{\sqrt{|\star v_i^k||\star v_j^l|}} \frac{|\star v_i^k v_j^l|}{|v_i^k v_j^l|} (f(v_i^k) - f(v_j^l)) \tag{7}$$

4. Properties

We now investigate properties of the previously defined Laplace operator on mesh sequences. We first show that such operator can be expressed as a sparse and easy to handle matrix. We then study its behaviour for small and large values of α.

4.1. Matrix representation

Property 4.1. *Let MS be a temporally coherent mesh sequence. The operator* $\triangle$ *on* 0*-forms on MS as expressed in Equation (7) is local, and can be encoded by a sparse* $VK \times VK$ *symmetric block tridiagonal matrix L which can be written blockwise as:*

$$L = \begin{bmatrix} L^{(1)} & D^{(1)} & & & \\ D^{(1)} & L^{(2)} & D^{(2)} & & \\ & \ddots & \ddots & \ddots & \\ & & D^{(K-2)} & L^{(K-1)} & D^{(K-1)} \\ & & & D^{(K-1)} & L^{(K)} \end{bmatrix} \tag{8}$$

with $1 \leq k \leq K-1$, *the* $V \times V$ *matrices* $D^{(k)}$ *being diagonal and the* $V \times V$ *matrices* $L^{(k)}$ *being symmetric.*

Proof. Operator $\triangle$ is local since for any function f and any vertex v_i^k, $\langle \triangle f, v_i^k \rangle$ only depends on the values of f on v_i^k and neighbouring vertices v_j^l.

The matrix expression derives from Equation (7). Diagonal coefficients of matrices $D^{(k)}$ are given by:

$$D_{i,i}^{(k)} = -\frac{1}{\sqrt{|\star v_i^k||\star v_i^{k+1}|}} \frac{|\star v_i^k v_i^{k+1}|}{|v_i^k v_i^{k+1}|}. \tag{9}$$

The coefficients $L_{i,j}^{(k)}$ of matrices $L^{(k)}$ are equal to zero if there is no spatial edge in MS between vertices v_i^k and v_j^k. Otherwise,

$$L_{i,j}^{(k)} = -\frac{1}{\sqrt{|\star v_i^k||\star v_j^k|}} \frac{|\star v_i^k v_j^k|}{|v_i^k v_j^k|}. \tag{10}$$

Diagonal coefficients are given by:

$$L_{i,i}^{(k)} = -D_{i,i}^{(k)} - D_{i,i}^{(k-1)} - \sum_{j \neq i} L_{i,j}^{(k)}, \tag{11}$$

where the terms $D_{i,i}^{(k)}$ and $D_{i,i}^{(k-1)}$ are omitted when not defined (i.e. for the first and last frames). □

Note that this matrix is very sparse, since all sub-matrices $L^{(k)}$ are sparse and sub-matrices $D^{(k)}$ are diagonal. If we expect a vertex to have 6 spatial neighbours on average, the total number of non-zero coefficients is about $7VK + 2V(K-1)$ (each row of the matrix has the following non-zero entries: the diagonal coefficient, plus a coefficient for each spatial neighbour, and a coefficient for each temporal neighbour). The overhead with respect to using a purely static Laplacian matrix on each mesh of the sequence is thus only $2V(K-1)$, corresponding to each use of the K diagonal sub-matrices $D^{(k)}$. Moreover, the structure of matrix L allows efficient pre-allocation of memory. Since all meshes of the sequence MS share the same connectivity, the number N_{nz} of non-zero coefficients is the same for all sub-matrices $L^{(k)}$. We can thus compute N_{nz} for the first sub-matrix $L^{(1)}$ and then pre-allocate exactly a 1-by-$N_{nz}K+2V(K-1)$ block of memory. A simple algorithm enables then to map this block to the non-zero matrix coefficients. Finally, efficient solvers exist for block tridiagonal matrices, see e.g. Terekhov [28].

4.2. Matrix inversion

Block tridiagonal matrices (also known as block Jacobi matrices) have been extensively studied in the literature, see e.g. Meurant [29], Salkuyeh [30], Molinari [31]. In particular, it is easy to show that L admits a block LDLT decomposition, which is a variant of a Cholesky decomposition into a lower unitriangular matrix, a diagonal matrix and the transpose of the first.

Definition 4.2 ([29, 30])**.** *Let MS be a temporally coherent mesh sequence. Let L be its associated Laplacian matrix as defined in Property 4.1. Let* $\{\Lambda^{(k)}, 1 \leq k \leq K\}$ *be* $V \times V$ *matrices defined recursively as:*

- $\Lambda^{(1)} = L^{(1)}$;

- $\forall k \geq 2, \Lambda^{(k)} = L^{(k)} - D^{(k-1)} \left(\Lambda^{(k-1)}\right)^{-1} D^{(k-1)}$.

Let $\{\Sigma^{(k)}, 1 \leq k \leq K\}$ be $V \times V$ matrices defined recursively as:

- $\Sigma^{(K)} = L^{(K)}$;
- $\forall k \leq K-1, \Sigma^{(k)} = L^{(k)} - D^{(k)} \left(\Sigma^{(k+1)}\right)^{-1} D^{(k)}$.

Property 4.3 ([29])**.** *Let Λ be the $VK \times VK$ block diagonal matrix $\Lambda = diag(\Lambda^{(1)}, \ldots, \Lambda^{(K)})$ and Σ be the $VK \times VK$ block diagonal matrix $\Sigma = diag(\Sigma^{(1)}, \ldots, \Sigma^{(K)})$. Let L_o be the block lower part of L:*

$$L_o = \begin{bmatrix} 0 & & & & \\ D^{(1)} & 0 & & & \\ & \ddots & \ddots & \ddots & \\ & & D^{(K-2)} & 0 & \\ & & & D^{(K-1)} & 0 \end{bmatrix} \tag{12}$$

Then L can be decomposed as:

$$L = (\Lambda + L_o)\Lambda^{-1}(\Lambda + L_o^t) = (\Sigma + L_o^t)\Sigma^{-1}(\Sigma + L_o) \tag{13}$$

These LDLT decompositions of L lead to a simple way to invert this matrix.

Property 4.4 ([29])**.** *Let $\{U_k, 1 \leq k \leq f\}$ and $\{V_k, 1 \leq k \leq f\}$ be two sequences of $n \times n$ matrices such as:*

$$U_1 = I, V_1 = \left(\Sigma^{(1)}\right)^{-1}, \tag{14}$$

with I the $n \times n$ identity matrix and $\forall k \geq 2$,

$$U_k = (-1)^{k-1} \left(D^{(k-1)}\right)^{-1} \Lambda^{(k-1)} \ldots \left(D^{(1)}\right)^{-1} \Lambda^{(1)}, \tag{15}$$

$$V_k = (-1)^{k-1} \left(\Sigma^{(1)}\right)^{-1} D^{(1)} \left(\Sigma^{(2)}\right)^{-1} \ldots D^{(k-1)} \left(\Sigma^{(k)}\right)^{-1}. \tag{16}$$

Then $\forall j \geq i$, the (i, j) block of L^{-1} can be expressed as $U_i V_j$.

Note that in Meurant [29] it is requested that L is proper, that is to say that submatrices $D^{(k)}$ are nonsingular. This is our case since these matrices are diagonal with non zero diagonal elements.

The inverse of L can thus be computed only by computing the inverse of K $V \times V$ matrices, namely the inverse of the $\Sigma^{(k)}$ matrices.

4.3. Behaviour for large and small time steps

Let us now investigate the behaviour of our Laplace operator when α tends to infinity or to zero. Remember from Section 3.1 that α is the parameter which scales the temporal dimension of the embedding space $\mathbb{E}$ with respect to the spatial dimensions. A large α decreases the influence of the temporal neighbours v_i^{k-1} and v_i^{k+1} over a given vertex v_i^k, with respect to its spatial neighbours sharing the same timelike coordinates. Conversely, a small α increases their influence.

Property 4.5. *When α tends to infinity, L tends to a block diagonal matrix $Diag(L^{(1)}, \ldots, L^{(K)})$, where each matrix $L^{(k)}$ is the spatial DEC Laplacian matrix with cotangent coordinates.*

Proof. Let A_i^k denote the area of the spatial dual cell of v_i^k in the mesh M^k. If $\alpha \gg 1$, the difference between successive areas A_i^{k-1}, A_i^k and A_i^{k+1} is negligible with respect to α. As a consequence, the volume of any vertex dual cell $\star v_i^k$ can be approximated by $\sqrt{\alpha}(t^{k+1} - t^k)A_i^k$.

For the same reason, for any spatial edge $v_i^k v_j^k$, the area of the dual cell $\star v_i^k v_j^k$ is equivalent to $\sqrt{\alpha}(t^{k+1} - t^k)$ times the length $|\star_{\mathbf{s}} v_i^k v_j^k|$ of the *spatial* dual cell of $v_i^k v_j^k$ in mesh M^k. We then have $\frac{1}{\sqrt{|\star v_i^k||\star v_j^k|}} \frac{|\star v_i^k v_j^k|}{|v_i^k v_j^k|} \sim \frac{1}{\sqrt{A_i^k A_j^k}} \frac{|\star_{\mathbf{s}} v_i^k v_j^k|}{|v_i^k v_j^k|}$. Thus, the $V \times V$ matrix $L^{(k)}$ is equivalent to the spatial Laplacian matrix for mesh M^k.

The length of any temporal edge $v_i^k v_i^{k+1}$ of the mesh sequence is equivalent to $\sqrt{\alpha}(t^{k+1} - t^k)$, and the area of its dual is small with respect to α. Thus, any coefficient $D_{i,i}^{(k)}$ of the diagonal matrix $D^{(k)}$ is close to zero. □

This property proves that, if α is large, our 4D Laplacian acts as a standard static Laplacian on each frame.

Property 4.6. *Suppose the motion of each vertex is small with respect to the tessellation: $\forall k, \forall i, \forall j$ such that $v_i^k v_j^k \in E^k, \|v_i^k v_i^{k+1}\| \ll \|v_i^k v_j^k\|$. Then, when α tends to zero, the motion coefficients $D_{i,i}^{(k)}$ are dominating over the geometry coefficients $L_{i,j}^{(k)}$.*

Proof. If $\forall k, \forall i, \forall j$ such that $v_i^k v_j^k \in E^k, \|v_i^k v_i^{k+1}\| \ll \|v_i^k v_j^k\|$, then the difference between successive areas A_i^{k-1}, A_i^k and A_i^{k+1} is negligible with respect to $\|v_i^k v_j^k\|$. As a consequence, the volume of any vertex dual cell $\star v_i^k$ can be approximated by $\|v_i^k v_i^{k+1}\|_2 A_i^k$.

For the same reason, the area of the dual cell $\star v_i^k v_j^k$ is, for any spatial edge $v_i^k v_j^k$, equivalent to $\|v_i^k v_i^{k+1}\|_2$ times the length $|\star_{\mathbf{s}} v_i^k v_j^k|$ of the *spatial* dual cell of $v_i^k v_j^k$ in mesh M^k. As the motion is small with respect to the tesselation, $\frac{1}{\sqrt{|\star v_i^k||\star v_j^k|}} \frac{|\star v_i^k v_j^k|}{|v_i^k v_j^k|} \sim \frac{1}{\sqrt{A_i^k A_j^k}} \frac{|\star_{\mathbf{s}} v_i^k v_j^k|}{|v_i^k v_j^k|}$, which implies that the $V \times V$ matrix $L^{(k)}$ is equivalent to the spatial Laplacian matrix for mesh M^k.

The area of the dual of any temporal edge $v_i^k v_i^{k+1}$ of the mesh sequence is equivalent to A_i^k. As a consequence, the coefficient $D_{i,i}^{(k)}$ of the diagonal matrix $D^{(k)}$ is equivalent to $\frac{1}{\|v_i^k v_i^{k+1}\|_2^2}$. Since the motion is small with respect to the tessellation, $D_{i,i}^{(k)} \gg L_{i,j}^{(k)}$ for any spatial edge $v_i^k v_j^k$: the temporal coefficients are dominating. □

This property shows that, if α is small, and the motion is small with respect to the geometric discretisation, our 4D Laplacian enables to recover the motion of each vertex of the mesh independently.

5. Application to spacetime editing

We now present an application in which we can directly use the previously defined discrete 3D+t Laplace operator to generate a variety of effects. This application deals with mesh sequence editing.

5.1. Background

In a mesh sequence editing application the user interactively deforms one or several meshes, called the *key-frames*. Changes must then be propagated to other frames of the sequence such that the resulting motion remains smooth, and each intermediate frame remains visually plausible. Editing can be useful to reuse or correct captured data, preserving the acquired details and the nonrigid surface deformation involved.

The problem of adapting existing animations has been widely studied in the case of skeletal motion, see e.g. Ho et al. [32] and references therein. The work of Le Naour et al. [21] is particularly close to ours in that they also use a 3D+t Laplace operator for the propagation of changes, and we compare our framework to this work. Our results indicate that while skeleton-based approaches provide intuitive control over the animation and a simplified editing process, they are limited in scope and are not applicable to arbitrary surfaces.

Fewer works approach this problem in the context of mesh sequences. The work of Kircher and Garland [33] was one of the first to address it, introducing a multi-resolution approach that allows for automatic transfer of a static mesh edit onto the other frames. In a follow-up work [34] they develop a differential representation which is invariant under rotation and translation, and can be used both for static editing and motion processing. To the best of our knowledge, the work of Xu et al. [35] was the first to extend Laplacian mesh processing [1] to mesh sequences. Their method requires the construction of a coarse control mesh, called a *cage*, which must then be transferred to other frames. For the static deformation process a non-linear energy formulation closely related to Sorkine and Alexa [36], thus to our work, is used. Changes on handle vertices are propagated in order to get suitable positional and rotational constraints on each frame; these are used to obtain intermediate meshes by applying the static deformation scheme on each. Pushing further the idea of embedding the animation into a coarse representation while preserving the details, Sumner et al. [37] propose a framework to directly deform the space. In this work, space affine transformations are represented as a graph. Computing the final deformation is then formulated as an optimisation problem on this graph with positional and detail preservation constraints. More recent works are the ones of de Aguiar and Ukita [38] and Tejera et al. [39]. In de Aguiar and Ukita [38] a kinematic skeleton is constructed to represent the coarse deformation of the sequence. This skeleton can then be edited using previously mentioned methods. Tejera et al. [39] develop a spacetime editing framework based also on differential coordinates. Static deformation on the key-frame is handled by a new Laplacian editing approach which incorporates information of deformations seen throughout the sequence. As in our work, this implies that all meshes comprised in a temporal window must be simultaneously considered while editing. Their approach yields good results, but it does not perform well when the target deformation is far from the examples. To propagate changes they propose three methods, of which the most satisfying one requires the use of their new static editing approach on each frame. We compare the results using our framework to this work in Section 5.6.

All existing methods decouple motion editing from geometry editing. As a consequence, several parameters are required to tune the modification of the animation. On the contrary, our discrete Laplace operator can be used to edit a mesh sequence in space and time in a simple yet flexible manner.

To this aim, we build over the simple formulation presented by Sorkine and Alexa [36] for detail-preserving mesh deformation. Our framework results from a relatively direct extension of this work into four-dimensional space.

5.2. As-rigid-as-possible surface deformation

Since differential coordinates encode local information, many applications have been proposed that make use of this property for detail-preserving mesh editing [40, 41]. As an application example of our 3D+t discrete Laplace operator we have chosen to build over the work of Sorkine and Alexa [36] because its explicit energy formulation makes the extension to mesh animations straightforward. This work handles large rotations by asking for local rigidity to be maintained: the mesh is divided into overlapping *cells* whose transformation must be close to rigid. One cell $C(v_i)$ is defined for each vertex v_i, and includes v_i as well as its one-ring neighbourhood. Defining cells in this way allows later to find new vertex positions using a discrete static Laplace operator.

Let us first consider a *static* mesh M with vertex coordinates $\{v_i \in \mathbb{R}^3\}$, and its deformed version M' with new vertex coordinates $\{v'_i \in \mathbb{R}^3\}$. If the deformation is rigid, for each cell $C(v_i)$ there exists a rotation matrix $\mathbf{R}_i \in SO(3)$ such that:

$$v'_i - v'_j = \mathbf{R}_i(v_i - v_j), \forall v_i v_j \in M \tag{17}$$

If the deformation is not rigid, $\mathbf{R}_i$ can be approximated by a rotation matrix that minimises the as-rigid-as possible (ARAP) energy defined as follows:

$$\sum_{v_i v_j \in M} w_{ij} \| \left(v'_i - v'_j\right) - \mathbf{R}_i \left(v_i - v_j\right) \|^2, \tag{18}$$

where w_{ij} are per-edge weights. Both rotation matrices $\mathbf{R}_i$ and positions v'_i are unknown. This non-linear energy formulation is minimised using a two-step iterative approach, in which a first step searches for optimal rotations based on current vertex positions, and a second step finds optimal vertex positions based on previously found rotations. Finding new vertex positions amounts to solving a linear Poisson system $Lx = b$, with L being the Laplace-Beltrami operator based on cotangent weights [9, 11].

5.3. Space-time editing using the ARAP formulation

When a keyframe is modified we want it to deform in an as-rigid-as-possible manner. We also want other frames to closely follow this deformation, preserving the original structure as much as possible. Since we are treating a mesh sequence as a CW complex (see Section 3.3), it is natural to ask for the *entire animation* to be deformed in an as-rigid-as-possible way.

When deforming the CW complex –as it is in the static case– a neighbour v_j^l of an edited vertex v_i^k will only be subtly affected

if the deformation is small with respect to the length of the edge $v_i^k v_j^l$. Conversely, close neighbours to v_i^k will move accordingly to v_i^k in space-time, thus propagating user changes. With our definition of the embedded space, the length of temporal edges is effectively controlled by parameter α. Tuning α allows the user to control the extent of the propagation and achieve several effects like slow/subtle to abrupt transitions.

Now that we have cells in $4D$ with spatial and temporal edges, vertices are defined in the 4D space $\mathbb{E}$. The energy to minimise is extended as follows:

$$\sum_{v_i v_j \in MS} w_{ij} \parallel \left(v_i' - v_j'\right) - \mathbf{R}_i \left(v_i - v_j\right) \parallel^2 \tag{19}$$

where v_i, v_j, v_i' and $v_j' \in \mathbb{E}$, and $\mathbf{R}_i \in SO(4)$ denotes the optimal rotation of cell $C(v_i)$, now in 4D space. Optimal rotations can still be derived as in Sorkine and Alexa [36], because the derivation is not dependent on the dimension of the data. Given optimal rotations, differentiating the energy in Equation (19) w.r.t. v_i' and setting partial derivatives to zero leads to the following system of equations:

$$\sum_{v_i v_j \in MS} w_{ij}(v_i' - v_j') = \sum_{v_i v_j \in MS} \frac{w_{ij}}{2}(\mathbf{R}_i + \mathbf{R}_j)(v_i - v_j) \tag{20}$$

The left-hand side is the 3D+t Laplace operator, and we can get optimal positions by solving a sparse linear system of equations $\mathbf{L}\mathbf{v}' = \mathbf{b}$ in the least-squares sense. Note that, as in the static case, the matrix $\mathbf{L}$ remains unchanged during the iterative process, and thus once pre-factorized, it can be reused to solve using direct methods.

With enough positional constraints, and preserved edge lengths, in general each mesh will remain close to its original time frame. Nevertheless, some of the modified vertices will have time coordinates which are not placed exactly on the corresponding frame. In this case, simply correcting the time coordinate to its closest frame is sufficient.

5.3.1. Constraints

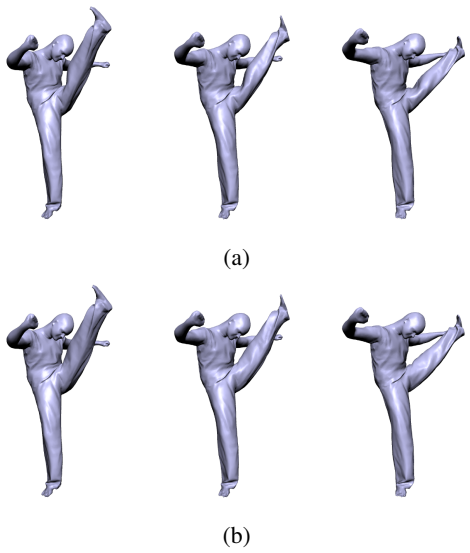
Deformation in 4D requires two types of constraints: (1) new positions of handle vertices, for the key-frame (2) constraints on the spatio-temporal boundary of the ROI, that is to say the geometric boundary of the ROI for each frame along with the first and last meshes of the temporal window. Anchoring the handle vertices to the user-specified positions allows the deformation to reach the desired target. Putting constraints on the boundary of the ROI creates a smooth transition between the edited submesh and the remaining mesh sequence. All of these constraints are specified as soft-constraints.

Anchoring the temporal boundary can be avoided if the deformation is expected to be propagated equally in all frames, see Figure 7 for an example. It is worth noting that constraints over the spatial boundary of the ROI can also be avoided, if α is sufficiently small. As opposed to static Laplacian editing on each frame, which requires at least one anchored vertex in order for the matrix to be non-singular, our 3D+t operator remains non-singular even if no constraints are set on intermediate frames, providing that the temporal influence is significant.

5.3.2. Initialisation

Our iterative framework requires proper initialisation for fast convergence. An initial guess can be obtained by solving Equation (19) using the identity as rotation matrix. Since editing of the key-frame must be done at interactive rates, we first allow for static editing to be performed on this frame. The result is used for the initial guess, along with Equation (19).

An advantage of this is that the full mesh can be anchored afterwards, thus forcing the user edits to be always fulfilled. A second advantage is that any method can be used for key-frame editing. The ARAP formulation has its limitations; among them failure to preserve the volume of the original mesh. The SR-ARAP method by Levi and Gotsman [42] is an extension to the ARAP framework that gives much better results in terms of volume preservation. Further, it does not require a volumetric discretisation of the interior of each mesh. Our experiments (Figure 2) show that using Levi and Gotsman [42] for key-frame editing yields better volume preservation for all the influenced frames than the original ARAP method of Sorkine and Alexa [36]. Note that we are only changing how we initialise the iterative process; our formulation for the entire sequence remains the same.



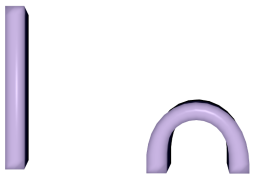
(a)

(b)

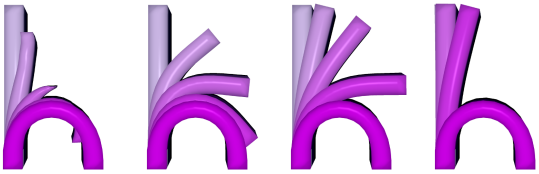
Figure 2: Spacetime editing using, for key-frame editing: (a) the original ARAP formulation [36] (b) the SR-ARAP method [42]. The second method avoids shrinkage of the upper leg not only on the key-frame but also on the other frames, even though the original ARAP formulation is used in 4D. Each row, from left to right: previous frame, key frame, next frame.

5.3.3. Influence of α

Figure 3 shows a simple example of how the α parameter can be tuned to change the editing effect. Deformation is done over a sequence with no motion consisting of 5 repetitions of a bar (Figure 3a). As α grows bigger, the effect is close to statically deforming the key-frame only. This is coherent with the theoretical analysis in Section 4.3, since a large value for α tends to modify only the key-frame. On the contrary, as α goes to zero, the temporal coefficients dominate, and vertices are independently, smoothly displaced along the temporal window.



(a) Original model (left), deformed target model (right).



(b) Editing a sequence of 5 repetitions of the original model with deformed target in last frame. From left to right: $\alpha = 0.1, 10, 100, 1000$.

Figure 3: Effect of parameter α when deforming a sequence with no motion.

Sequence	# vertices per frame	# frames	α	# iterations
Bar (Figure 3)	172	5	0.1	33
			1	74
Girl (Figure 7)	1215	7	0.01	12
			0.1	11
			1	5
Capoeira (Figures 9a, 9b)	6571	9	10	39
			1000	11
Capoeira (Figure 9c)	10788	19	0.5	10
Horse (Figure 5)	7135	9	0.1	157
			1	377
			10	400

Table 1: Number of iterations for the mesh sequences shown in the paper.

Note that the effect of a given value for α depends both on the geometric and the temporal discretisations. Similar effects on two mesh sequences require different values for α if their number of frames or number of vertices differ.

5.4. Implementation

The 3D+t Laplacian and the ARAP framework have been implemented in Python 2.7, with sparse Cholesky decomposition handled by a wrapper for the CHOLMOD library [43]. We stop the iterative process when the difference in mesh coordinates between consecutive iterations is below 10^{-3}, for all frames.

Computational times depend on the number of iterations, which itself depends not only on the initial guess but also on the value of α. Table 1 shows the number of iterations for the animations shown on this paper. The step with the highest computational complexity is the pre-factorisation of the Laplacian, and our running times for this range from $20ms$ in the case of the bar, to $75s$ for the capoeira sequence.

It is worth noting that the algorithm is highly parallelizable. The construction of our 3D+t Laplacian only requires *local* temporal information, i.e. a frame and one or two consecutive meshes. Because of the matrix structure, solving the sparse linear system can be done blockwise, and a parallel algorithm that solves for each frame on a different processor core could be devised.

Figure 6: Horse sequence [44]. Zoom on frames 5 to 7 showing that results using Tejera et al. [39] (top) and our method with $\alpha = 0.5$ (bottom) are visually similar.

5.5. Comparison to existing spacetime Laplace operators

We compare our method to the discrete spacetime Laplace operators proposed by Yang et al. Yang et al. [22] and Le Naour et al. LeNaour et al. [21] in Figure 4. Both methods use uniform weights for the spatial edges, which is not the case of our method. Consequently, they introduce severe artifacts in areas where the tessellation is non-uniform. Using Gaussian temporal weights as in LeNaour et al. [21] rather than uniform temporal weights only slightly improves the results. Our weights better preserve the relative motion of each vertex with respect to its neighbours. Comparison with a static Laplacian editing framework are shown in the next section.

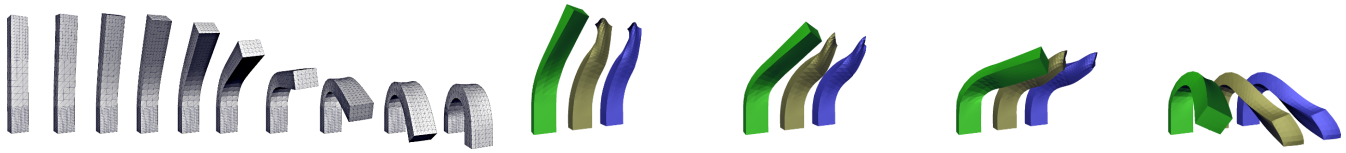
5.6. Results

Evaluation was performed using both synthetic (Figures 5 and 6) and captured animations (Figures 7, 8 and 9) obtained from publicly available databases [44, 45, 46]. For a better visualisation of the results, refer to the supplementary video.

Figures 7, 8, 5 and 6 show that our extension of as-rigid-as-possible editing to mesh sequences achieves similar effects to Tejera et al. [39], while allowing the user to have control over the propagation of changes.

Figures 7 and 8 provide a comparison with the method of Tejera et al. [39], while showing over a simple deformation the types effect that can be achieved by tuning α. Rows 3 to 6 show how different kinds of propagation can be achieved, ranging from an abrupt change on the sequence (using $\alpha = 1$) to a smoother interpolation using $\alpha = 0.01$. The last row was obtained without setting constraints on the temporal boundary, which as mentioned, is not necessary if the key-frame edits are to be propagated equally throughout the window. Notice that by using a very small value of α stretches all meshes in the window. These kinds of effects could be used for correcting reconstruction errors over a time window. Similar effects can be observed in Figure 5, this time performing a more complex deformation.

Examples of detail-preserving deformations can be seen in Figure 9. Note how the wrinkles in the cloth are preserved in



Original

Edited sequence showing frames 3, 4, 5 and 8.

Figure 4: Editing the bending bar sequence using different weighting schemes. Frame 6 was rotated and translated towards the right. The results using our method with $\alpha = 1$ are shown in green, using Yang et al. [22] in brown, and using Le Naour et al. [21] with $\alpha = 1$ in blue. In this last case several values for the β parameter were tested. All of them yield similar results.

Figure 5: From top to bottom: original sequence; results using Tejera et al. [39]; our results using $\alpha = 0.5$, $\alpha = 5$ and $\alpha = 0.1$, the latter without constraint on the temporal window. Key-frame in light blue (frame number 5).

the edited version, while achieving natural motions. This also holds for the case of a rotational edit shown in the last row.

6. Conclusion

In this paper we have introduced a discrete Laplace operator for temporally coherent mesh sequences. This operator is defined by modelling the sequences as CW complexes in a 4-dimensional Riemaniann space and using Discrete Exterior Calculus. A user-defined parameter α is associated to the 4D space to control the influence of motion with respect to the geometry. We have shown that this operator can be expressed by a sparse blockwise tridiagonal matrix, with a linear number of non zero coefficients with respect to the number of vertices in the sequence. The storage overhead with respect to frame-by-frame mesh processing is limited. We have also shown an application example, as-rigid-as-possible editing, for which it is relatively easy to extend the classical static Laplacian framework to mesh sequences with this matrix. Similar results to state-of-the-art methods can be reached with a simple, global formulation.

This opens the possibility of many other problems in animation processing to be tackled the same way by taking advantage of the existing literature on the Laplacian operator for 3D meshes [1, 2, 3]. In the future, we are in particular interested in studying the spectral properties of the defined discrete Laplace operator.

Acknowledgements

We thank Mohammed Azougarh, Mohamed El Bakkali, Lucas Razafindrainijama and Redouane Oubenal for preliminary work on this topic, and Bruno Lévy for helpful discussions. Furthermore, we thank Dan Casas for providing us figures for comparison, and Adrian Hilton and Christian Theobalt for sharing 3D motion sequences. This project has been partially supported by the ANR through the MORPHO project (ANR-10-BLAN-0206) and by Inria through the Inria Internship program.

References

[1] Sorkine O. Differential representations for mesh processing. Computer Graphics Forum 2006;25(4):789–807.

[2] Levy B, Zhang RH. Spectral geometry processing. SIGGRAPH Asia Course Notes; 2009.

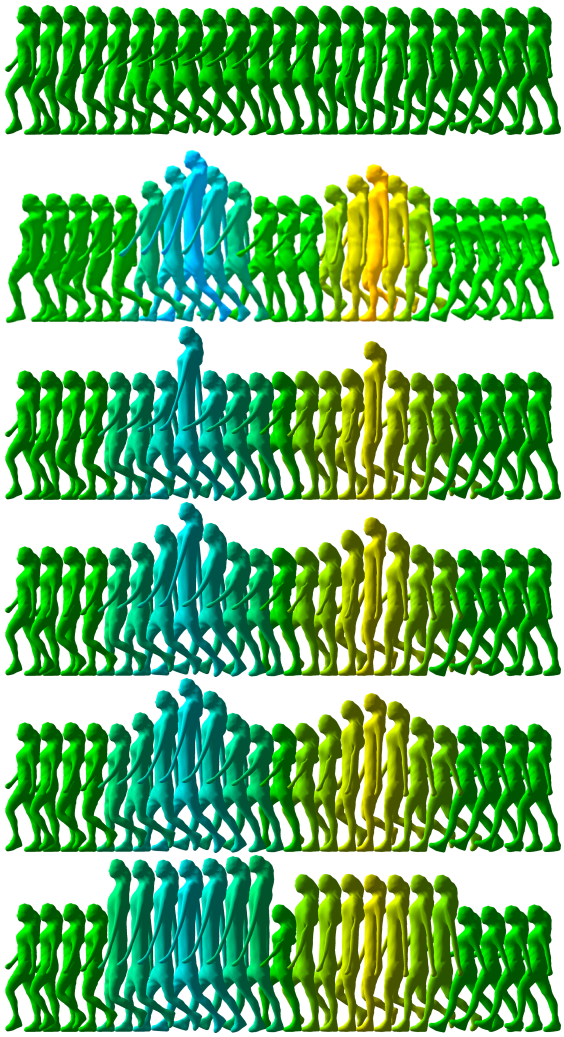


Figure 7: Editing a capture walk sequence [45]. Target deformations are in yellow and light blue. Frames not in green are the ones allowed to deform (frames 5 to 11 for the yellow deformation, 13 to 19 for the blue deformation). First row: original sequence. Second row: results using Tejera et al. [39]. Rows 3 to 5: $\alpha = 1, 0.1, 0.01$, with constraints on the temporal boundary. Row 6: $\alpha = 10^{-5}$ without constraints on the temporal boundary.

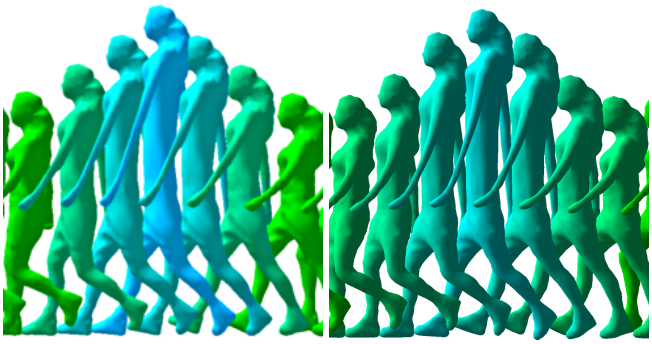
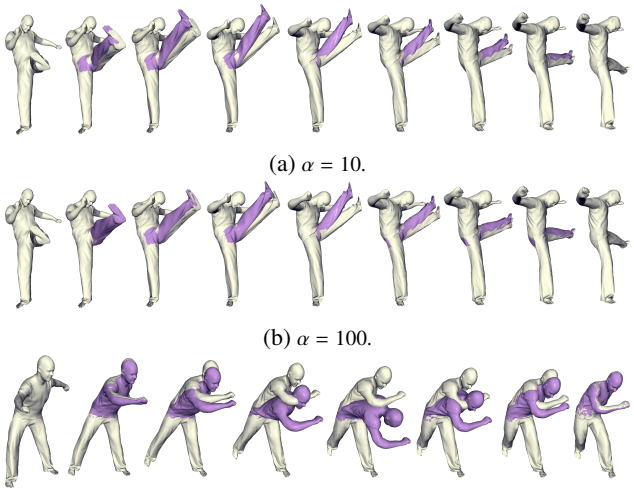


Figure 8: Editing a capture walk sequence [45]. Zoom on frames 5 to 11 showing that results using Tejera et al. [39] (left) and our method with $\alpha = 0.01$ (right) are visually similar.



(a) $\alpha = 10$.

(b) $\alpha = 100$.

(c) $\alpha = 0.5$.

Figure 9: Modifying Capoeira sequence [46]. First two rows: performing a higher kick. Last row: bending forward while dancing; showing frames $1, 4, 6, 8, 10, 11, 12, 14, 19$, key-frame is frame number 10. The original sequence is in white, while the modified one is shown in purple.

[3] Zhang RH, van Kaick O, Dyer R. Spectral mesh processing. Computer Graphics Forum 2010;29(6):1865–94.

[4] Merris R. Laplacian matrices of graphs: a survey. Linear Algebra and its Applications 1994;197–198:143–76.

[5] Young IT, Gerbrands JJ, Van Vliet LJ. Fundamentals of image processing. Delft University of Technology, The Netherlands; 1998.

[6] Collet A, Chuang M, Sweeney P, Gillett D, Evseev D, Calabrese D, et al. High-quality streamable free-viewpoint video. Transactions on Graphics 2015;34(4):69:1–13.

[7] Desbrun M, Hirani AN, Leok M, Marsden JE. Discrete exterior calculus. Tech. Rep.; arXiv:math/0508341; 2005.

[8] Crane K, de Goes F, Desbrun M, Schröder P. Digital geometry processing with discrete exterior calculus. SIGGRAPH Course Notes; 2013.

[9] Pinkall U, Polthier K. Computing discrete minimal surfaces and their conjugates. Experimental Mathematics 1993;2(1):15–36.

[10] Taubin G. A signal processing approach to fair surface design. In: SIGGRAPH. 1995, p. 351–8.

[11] Meyer M, Desbrun M, Schröder P, Barr AH. Discrete differential-geometry operators for triangulated 2-manifolds. In: Visualization and mathematics III. 2003, p. 35–57.

[12] Bobenko AI, Springborn BA. A discrete Laplace-Beltrami operator for simplicial surfaces. Discrete and Computational Geometry 2007;38(4):740–56.

[13] Alexa M, Wardetzky M. Discrete Laplacians on general polygonal meshes. Transactions on Graphics 2011;30(4):102:1–10.

[14] Belkin M, Sun J, Wan Y. Constructing laplace operator from point clouds in Rd. In: Symposium on Discrete Algorithms. 2009, p. 1031–40.

[15] Liu Y, Prabhakaran B, Guo X. Point-based manifold harmonics. Transactions on Visualization and Computer Graphics 2012;18(10):1693–703.

[16] Petronetto F, Paiva A, Helou ES, Stewart D, Nonato LG. Mesh-free discrete Laplace-Beltrami operator. Computer Graphics Forum 2013;32(6):214–26.

[17] Zhou K, Huang J, Snyder J, Liu X, Bao H, Guo B, et al. Large mesh deformation using the volumetric graph laplacian. Transactions on Graphics 2005;24(3):496–503.

[18] Calcagni G, Oriti D, Thürigen J. Laplacians on discrete and quantum geometries. Classical and Quantum Gravity 2013;30(12):125006:1–36.

[19] Wang R, Yang Z, Liu L, Chen Q. Discretizing LaplaceBeltrami operator from differential quantities. Communications in Mathematics and Statistics 2013;1(3):331–50.

[20] Xu G. Consistent approximations of several geometric differential operators and their convergence. Applied Numerical Mathematics 2013;69:1–12.

[21] LeNaour T, Courty N, Gibet S. Spatio-temporal coupling with the 3d+t motion Laplacian. Computer Animation and Virtual Worlds 2013;24(3-

4):419–28.
[22] Yang L, Xiao C, Fang J. Multi-scale geometric detail enhancement for time-varying surfaces. Graphical Models 2014;76(5):413–25.
[23] Allain B, Franco JS, Boyer E. An efficient volumetric framework for shape tracking. In: Conference on Computer Vision and Pattern Recognition. 2015, p. 268–76.
[24] Hatcher A. Algebraic topology. Cambridge University Press; 2002.
[25] Hirani A. Discrete exterior calculus. Ph.D. thesis; Caltech; 2003.
[26] Grady L, Polimeni JR. Discrete calculus: applied analysis on graphs for computational science. Springer; 2010.
[27] Vallet B, Lévy B. Spectral geometry processing with manifold harmonics. Computer Graphics Forum 2008;27(2):251–60.
[28] Terekhov AV. A fast parallel algorithm for solving block-tridiagonal systems of linear equations including the domain decomposition method. Parallel Computing 2013;39(6):245–58.
[29] Meurant G. A review on the inverse of symmetric tridiagonal and block tridiagonal matrices. Journal on Matrix Analysis and Applications 1992;13(3):707–28.
[30] Salkuyeh DK. Comments on "A note on a three-term recurrence for a triadiagonal matrix". Applied Mathematics and Computation 2006;176(2):442–4.
[31] Molinari LG. Determinants of block tridiagonal matrices. Linear Algebra and its Applications 2008;429(8–9):2221–6.
[32] Ho ES, Komura T, Tai CL. Spatial relationship preserving character motion adaptation. Transactions on Graphics 2010;29(4):33:1–8.
[33] Kircher S, Garland M. Editing arbitrarily deforming surface animations. Transactions on Graphics 2006;25(3):1098–107.
[34] Kircher S, Garland M. Free-form motion processing. Transactions on Graphics 2008;27(2):12:1–13.
[35] Xu W, Zhou K, Yu Y, Tan Q, Peng Q, Guo B. Gradient domain editing of deforming mesh sequences. Transactions on Graphics 2007;26(3):84:1–10.
[36] Sorkine O, Alexa M. As-rigid-as-possible surface modeling. In: Symposium on Geometry Processing. 2007, p. 109–16.
[37] Sumner RW, Schmid J, Pauly M. Embedded deformation for shape manipulation. Transactions on Graphics 2007;26(3):80:1–8.
[38] de Aguiar E, Ukita N. Representing and manipulating mesh-based character animations. In: SIBGRAPI Conference on Graphics, Patterns and Images. 2012, p. 198–204.
[39] Tejera M, Casas D, Hilton A. Animation control of surface motion capture. Transactions on Cybernetics 2013;43(6):1532–45.
[40] Sorkine O, Cohen-Or D, Lipman Y, Alexa M, Rössl C, Seidel HP. Laplacian surface editing. In: Symposium on Geometry Processing. 2004, p. 175–84.
[41] Au OKC, Tai CL, Liu L, Fu H. Dual Laplacian editing for meshes. Transaction on Visualization and Computer Graphics 2006;12(3):386–95.
[42] Levi Z, Gotsman C. Smooth rotation enhanced as-rigid-as-possible mesh animation. Transactions on Visualization and Computer Graphics 2015;21(2):264–77.
[43] Chen Y, Davis TA, Hager WW, Rajamanickam S. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. Transactions on Mathematical Software 2008;35(3):1–14.
[44] Sumner RW, Popović J. Deformation transfer for triangle meshes. Transactions on Graphics 2004;23(3):399–405.
[45] Starck J, Hilton A. Surface capture for performance-based animation. Computer Graphics and Applications 2007;27(3):21–31.
[46] De Aguiar E, Stoll C, Theobalt C, Ahmed N, Seidel HP, Thrun S. Performance capture from sparse multi-view video. Transactions on Graphics 2008;27(3):98:1–10.

Appendix: volumes of temporal cells

(a). Area of a temporal 2-cell

For all $t \in [t^k, t^{k+1}]$, $v_i^t v_j^t$ can be written as $\beta v_i^{k+1} v_j^{k+1} + (1-\beta) v_i^k v_j^k$, with $\beta = \frac{\|v_i^k v_i^t\|}{\|v_i^k v_i^{k+1}\|}$. $\forall k$, let us denote the coordinates of vector $v_i^k v_j^k$ by (t^k, x_k, y_k, z_k). Using these notations, it is

$$\begin{aligned}\|v_i^t v_j^t\| &= \Big((\beta x_{k+1}+(1-\beta)x_k)^2+(\beta y_{k+1}+(1-\beta)y_k)^2+(\beta z_{k+1}+(1-\beta)z_k)^2\Big)^{1/2}\\ &= \Big(\beta^2\big((x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2+(z_{k+1}-z_k)^2\big)\\ &\quad +2\beta\big(x_k(x_{k+1}-x_k)+y_k(y_{k+1}-y_k)+z_k(z_{k+1}-z_k)\big)+x_k^2+y_k^2+z_k^2\Big)^{1/2}.\end{aligned} \tag{21}$$

In case $v_i^k v_j^k = v_i^{k+1} v_j^{k+1}$, then $v_i^t v_j^t = v_i^k v_j^k$ and $|Q_{i,j}^k| = \|v_i^k v_j^k\| \cdot \|v_i^k v_i^{k+1}\|$.
Otherwise, let $A = \frac{x_k(x_{k+1}-x_k)+y_k(y_{k+1}-y_k)+z_k(z_{k+1}-z_k)}{(x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2+(z_{k+1}-z_k)^2}$ and $u = \beta + A$. Now $|Q_{i,j}^k|$ can be rewritten as

$$\begin{aligned}|Q_{i,j}^k| &= \int_{v_i^k}^{v_i^{k+1}} \|v_i^t v_j^t\| dv_i^t\\ &= \|v_i^k v_i^{k+1}\| \Big((x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2+(z_{k+1}-z_k)^2\Big)^{1/2}\\ &\quad \int_A^{1+A} \left(u^2 - A^2 + \frac{x_k^2+y_k^2+z_k^2}{(x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2+(z_{k+1}-z_k)^2}\right)^{1/2} du\\ &= \|v_i^k v_i^{k+1}\| \Big((x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2+(z_{k+1}-z_k)^2\Big)^{1/2}\\ &\quad \int_A^{1+A} \Big(u^2+(x_k y_{k+1}-x_{k+1}y_k)^2+(y_k z_{k+1}-y_{k+1}z_k)^2\\ &\quad +(z_k x_{k+1}-z_{k+1}x_k)^2\Big)^{1/2} du\\ &= \|v_i^k v_i^{k+1}\| \cdot \|v_i^{k+1} v_j^{k+1} - v_i^k v_j^k\| \int_A^{1+A} \Big(u^2 + \|v_i^k v_j^k \times v_i^{k+1} v_j^{k+1}\|_2^2\Big)^{1/2} du.\end{aligned} \tag{22}$$

Noting $B = \|v_i^k v_i^{k+1}\| \cdot \|v_i^{k+1} v_j^{k+1} - v_i^k v_j^k\|$ and $C = \|v_i^k v_j^k \times v_i^{k+1} v_j^{k+1}\|_2$, we have:

$$|Q_{i,j}^k| = B \int_A^{1+A} \Big(u^2 + C^2\Big)^{1/2} du. \tag{23}$$

This integral can be solved and gives

$$\begin{aligned}|Q_{i,j}^k| &= \frac{B}{2}\Big((1+A)\sqrt{(1+A)^2+C^2} - A\sqrt{A^2+C^2}\Big)\\ &\quad + \frac{B}{2}C^2 \log\left(\frac{\sqrt{(1+A)^2+C^2}+(1+A)}{\sqrt{A^2+C^2}+A}\right).\end{aligned} \tag{24}$$

(b). Volume of a temporal 3-cell

Using the same notations as above, it is

$$\begin{aligned}Area(v_i^t v_j^t v_l^t) &= \tfrac{1}{2}\|v_i^t v_j^t \times v_i^t v_l^t\|_2\\ &= \tfrac{1}{2}\|\big(\beta v_i^{k+1} v_j^{k+1} + (1-\beta) v_i^k v_j^k\big) \times \big(\beta v_i^{k+1} v_l^{k+1} + (1-\beta) v_i^k v_l^k\big)\|_2\\ &= \tfrac{1}{2}\|\beta^2 v_i^{k+1} v_j^{k+1} \times v_i^{k+1} v_l^{k+1}\\ &\quad +\beta(1-\beta)(v_i^{k+1} v_j^{k+1} \times v_i^k v_l^k + v_i^k v_j^k \times v_i^{k+1} v_l^{k+1})\\ &\quad +(1-\beta)^2 v_i^k v_j^k \times v_i^k v_l^k\|_2\\ &= \tfrac{1}{2}\|\beta^2 (v_i^{k+1} v_j^{k+1} - v_i^k v_j^k) \times (v_i^{k+1} v_l^{k+1} - v_i^k v_l^k)\\ &\quad +\beta\big((v_i^{k+1} v_j^{k+1} - v_i^k v_j^k) \times v_i^k v_l^k + v_i^k v_j^k \times (v_i^{k+1} v_l^{k+1} - v_i^k v_l^k)\big)\\ &\quad +v_i^k v_j^k \times v_i^k v_l^k\|_2,\end{aligned} \tag{25}$$

and hence

$$\begin{aligned}|F_{i,j,l}^k| &= \int_{t^k}^{t^{k+1}} Area(v_i^t v_j^t v_l^t) dt\\ &= \tfrac{1}{2}\|v_i^k v_i^{k+1}\| \int_0^1 \|\beta^2 A + \beta B + C\|_2 d\beta\end{aligned} \tag{26}$$

with

$$
\begin{aligned}
A &= (v_i^{k+1} v_j^{k+1} - v_i^k v_j^k) \times (v_i^{k+1} v_l^{k+1} - v_i^k v_l^k) \\
B &= \Big((v_i^{k+1} v_j^{k+1} - v_i^k v_j^k) \times v_i^k v_l^k + v_i^k v_j^k \times (v_i^{k+1} v_l^{k+1} - v_i^k v_l^k) \Big) \\
C &= v_i^k v_j^k \times v_i^k v_l^k .
\end{aligned}
\tag{27}
$$

Solving this integral in an exact manner would require finding the root of 4th degree polynomials. In practice, we choose to integrate numerically.