Spatio-temporal coupling with the 3D+t motion Laplacian

Thibaut Le Naour, Nicolas Courty, Sylvie Gibet

Spatio-temporal coupling with the 3D+t motion Laplacian

T. Le Naour
IRISA/UBS
thibaut.le-naour@univ-ubs.fr
www-irisa.univ-ubs.fr/Thibaut.Le-Naour

N. Courty
IRISA/UBS
Courty@univ-ubs.fr
www-irisa.univ-ubs.fr/Nicolas.Courty

S. Gibet
IRISA/UBS
sylvie.gibet@univ-ubs.fr
www-irisa.univ-ubs.fr/Sylvie.Gibet

Abstract

Motion editing requires the preservation of spatial and temporal information of the motion. During editing, this information should be preserved at best. We propose a new representation of the motion based on the Laplacian expression of a 3D+t graph: the set of connected graphs given by the skeleton over time. Through this Laplacian representation of the motion, we propose an application which allows an easy and interactive editing, correction or retargeting of a motion. The new created motion is the result of the combination of two minimizations, linear and non-linear: the first penalizes the difference of energy between the Laplacian coordinates from an animation to the desired one. The other one preserves the length of segments. Using several examples, we demonstrate the benefits of our method and in particularly the preservation of the spatiotemporal properties of the motion in an interactive context.

Keywords: Character Animation, Spatial Relationship, Motion Editing, Motion Retargeting

1 Introduction

Human motion editing is compelling in character animation, because manipulating directly motion data requires a detailed knowledge of the quality and essential features of these data. One key issue in the edition process is to create new animations driven by user-defined objectives and constraints, while preserving as much as possible spatiotemporal properties, expressivity, and consistency of the original motion.

Over the past decade, much effort has addressed the problem of adapting, deforming, and editing interactively movement from existing motion data. Among promising approaches, some have highlighted the effectiveness of Laplacian coordinates to encode implicitly the spatial relationships between the bodies of single or multiple characters. These spatial relationships are indeed intuitive and suitable to describe the semantics of the interaction. The technique has been successfully applied to editing or retargeting captured motion, by preserving deformation energy over a sequence of frames [1]. However, if these Laplacian editing methods satisfy smooth, continuous spacetime deformation, they do not implicitly incorporate the temporal dynamics of the motion within the Laplacian operator.

We propose a novel Laplacian-based motion editing method that couples spatial and temporal information through a unique optimization framework, thus preserving automatically the geometrical motion deformation and the temporal dimension of the motion.

We thus extend the classical expression of the Laplacian operator for skeleton animation by incorporating the time t into the 3D Laplacian coordinates x, y and z. Applied to a function f, the Laplacian operator becomes:

$$\Delta f = \frac{\partial^2 f}{\partial t^2} + \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} \ .$$

In this expression, the second partial derivative evaluates the motion acceleration in the temporal neighborhood of the current frame. The goal of the time component during the edition process is to locally preserve acceleration and deceleration, that is to say motion dynamics.

In this paper, the motion is represented by a global geometrical structure - a so-called 3D+t graph - encoding both spatial information through the connected links between the joints of the skeleton, and temporal information representing the trajectories of joints over time. The edition process is achieved through a Laplacian representation of this 3D+t graph. Thus, fixing some points of the graph and preserving the lengths of the body segments allow us through the Laplacian operator to significantly edit the motion while maintaining spatiotemporal features of the original motion. The new created movement results from the combination of two respectively linear and non-linear minimizations: the first one penalizes the difference of energy between the Laplacian coordinates of the original animation to the desired one. The second one, applied subsequently to the first preconditioning minimization, uses a gradient decent algorithm in order to preserve the length of the segments. Our method is similar to the approach developed by [2] for mesh animation processing. In contrast, our approach aims to implicitly incorporate i) spatial relationships, in terms of skeleton connections and distances to objects of the environment, and ii) temporal relationships in terms of joint connections along motion trajectories.

Our contributions can be stated as:

- We propose an unified way of handling both temporal and spatial variations through a 3D+t graph. Associated to the Laplacian operator, our system is able to preserve during the edition process spatial and temporal properties of the original motion;
- Instead of using uniform weights associated to edges of the graph, we introduce a parameterization of the Laplacian coordinates in terms of Gaussian kernel weights that enhance the influence of the temporal dynamics;
- A control of the rotations is directly extracted from the control points and used within the minimization function to efficiently modify the animation sequence of skeletons through an interpolation process;
- We propose an original numerical solving principle by decoupling in an iterative process the linear and non linear minimization parts.

In this paper, we develop a complete, robust and interactive system of motion capture editing, which is capable to edit all types of motion (walking, jumping, boxing, swimming,etc), while allowing an intuitive control by the user. Moreover, as our system preserves nearby contacts by maintaining the distance between 3D points, it is also appropriate for motion retargeting.

2 Background

Motion editing There exists a lot of work dedicated to the production of new animation form existing ones. Most of the times, the existing motion is edited to fit new kinematic constraints using inverse kinematics techniques (e.g. [3–5]) or build a statistical model of the motion to constraint the synthesis [6–8]. Those statistical methods need generally an important quantity of motions to learn a good motion model. This is not the case with inverse kinematics, which major drawbacks lie on the fact that the quality of the original motion is usually lost when the new task is different from the reference one. Physically based motion retargeting [9] is an interesting alternative in its possibility to modify an original motion submitted to physical constraints, but requires a filtering step (based on unscented Kalmann filtering) which may fail when the new constraints is too different from the reference motion. In our work, the characteristics of the original motions are preserved by changing the representational space of the motion to a differential representation based on Laplacian coordinates.

The Laplacian operator in computer graphics In last years, the use of Laplacian operator has been addressed in many studies. In the context of computer graphics, many authors use it to deform geometric structures (e..g image editing [10] or mesh editing [11, 12]). A good survey proposed by Sorkine [13] is specifically addressed to mesh editing. Generally, this geometric deformation operates in space and does not consider changes in time. Rencently Hetroy [2] used a discrete Laplacian operator applied to mesh sequence or 3D videos. Its approach is similar to our method since we characterize the motion with the Cartesian coordinates associated to the time dimension.

In the context of computer animation, the Laplacian operator has been used in the context of crowd simulation [14–16], where it is generally used to encode the spatial relationships among pedestrians, and eventually their relations with the environment. In the more specific case of articulated human figures, Kim et al. [17] used the Laplacian to edit the interaction among multiple characters. More precisely, they associate inverse kinematic on end effectors and ground contacts with the Laplacian representation of motion path (time series of body pelvis positions projected onto the ground). Note that in this work, the authors use also the Laplacian to combine temporal information with spatial trajectory of the root of the skeleton in 2D. Ho's approach [1] is particularly related to our work since the discrete Laplacian operator is used to edit poses and take into account the interaction between characters or with the environment. The constructed Laplacian makes reference to "an interaction mesh" which is built upon a convex hull of a pose. In this work, each pose is edited independently in time, while a temporal smoothing term is added to the optimized energy function. Our method is similar in that we also employ the Laplacian to edit motion but we use an expression of the Laplacian operator applied on a graph built upon the whole sequence, thus taking explicitly into account the motion dynamics. Also, our work considers the notion of rotations induced by control points, and encompasses a wider range of motions involving large translations and rotations of the whole body.

3 3D+t Laplacian motion representation

The main goal of this paper is to provide a motion editing system that preserves both temporal and spatial variations of the motion. For this purpose, we use the Laplacian operator which computes the differential coordinates of each vertex of the graph, discretized in space and time with its neighbors, thus minimizing locally both the spatial deformation and acceleration energy.

3.1 Discrete Laplace operator

In the spatial context, we consider the graph $G = (\boldsymbol{V}, \boldsymbol{E})$, where E is the connectivity and $\boldsymbol{V} = v_1, ..., v_n$ describes the geometry by Euclidian coordinates of each points of the graph in $\mathbb{R}^3$. From absolute coordinates of $\boldsymbol{V}$, we can determine the graph by the set of differential coordinates $\Delta = \delta_i$. More precisely, the coordinate i is represented by:

$$\delta_i = \mathcal{L}(\boldsymbol{v}_i) = \sum_{j \in \mathcal{N}(i)} \mathsf{w}_{ij}(\boldsymbol{v}_i - \boldsymbol{v}_j). \qquad (1)$$

where $\mathcal{L}$ represents the discrete Laplacian operator, and $\delta_i = \mathcal{L}(v_i)$ is the differential coordinate for the point i. $\mathcal{N}(i)$ is the set of points connected to point v_i, also called the one-ring neighbors, and w_{ij} is the weight associated to the edge e_{ij}. The Equation (1) can be written in the matrix form $\Delta = LV$, where L is determined by:

$$L_{ij} = \begin{cases} \sum_{j \in \mathcal{N}(i)} w_{ij} & \text{if } i = j \\ -w_{ij} & \forall (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

3.2 3D+t Discrete Laplace operator

Let a sequence of skeletons $S = (S_1, ..., S_m)$, each skeleton S_k ($1 \leq k \leq m$) being defined by $S_k = (\boldsymbol{V}_k, \boldsymbol{E}_k)$, with $\boldsymbol{V}_k$ a set of vertices, and E_k a set of edges. We note $v_{i,k}$, the vertex of index i contained into $\boldsymbol{V}_k$. As described in [2], we define the 3D+t Laplacian coordinates of the vertex $\boldsymbol{v}_{i,k}$, as the sum of the spatial coordinates corresponding to the skeleton k and the temporal coordinates corresponding to the "temporal" edges $(\boldsymbol{v}_{i,k-1}, \boldsymbol{v}_{i,k})$ and $(\boldsymbol{v}_{i,k}, \boldsymbol{v}_{i,k+1})$, all these Laplacian coordinates being associated to specific weights:

$$\mathcal{L}(\boldsymbol{v}_i) = (\mathsf{w}^-(\boldsymbol{v}_{i,k} - \boldsymbol{v}_{i,k-1}) + \mathsf{w}^+(\boldsymbol{v}_{i,k} - \boldsymbol{v}_{i,k+1})) + \sum_{j \in \mathcal{N}_k(i)} \mathsf{w}_{ij,k}(\boldsymbol{v}_{i,k} - \boldsymbol{v}_{j,k}), \tag{2}$$

where w^- and w^+ are the weights of each "temporal" edge linking $\boldsymbol{v}_{i,k}$, and $\mathcal{N}_k(i)$ is the set of neighbours of $\boldsymbol{v}_{i,k}$. To simplify notations, we won't distinguish in the remainder of the paper the weights characterizing temporal or spatial edges, as specified in section 4.1.

3.3 From the motion to the 3D+t graph

Our method initially converts the sequence of skeletons over time into a geometrical structure, the so-called 3D+t graph, encoding both spatial data (joints of skeletons) and temporal data (links between adjacent skeletons). As shown in Figure 1, we create a 3d+t dimensional graph by extracting from a given motion the chunk that we want to edit. It is not necessary to retrieve the entire motion, as the editing time computation is directly linked to the size of the motion. The geometrical information is given by the set of positions of the joints of all skeletons. As mentioned before, the topological information of the graph is determined by the connectivity of each skeleton, associated with links between skeletons. Thus, as illustrated in Figure 1, the vertex $\boldsymbol{v}_{i,k}$ of joint i of the skeleton at time k is linked to the vertices $\boldsymbol{v}_{i,k-1}$ and $\boldsymbol{v}_{i,k+1}$ corresponding to joint i of the skeletons at times $k-1$ and $k+1$. More precisely we define the 3D+t graph $G = (\boldsymbol{V}, \boldsymbol{E}) = (\boldsymbol{V}, \boldsymbol{E}_S \cup \boldsymbol{E}_T)$, with $\boldsymbol{V} = \{V_k\}$ the set of points, $\boldsymbol{E}$ the set of edges, $\boldsymbol{E}_S$ the set of edges within a skeleton, and $\boldsymbol{E}_T$ the set of temporal edges between adjacent skeletons in time. Note that $\boldsymbol{E}_T$ corresponds to the first term of Equation (2) and $\boldsymbol{E}_S$ to the second term. In addition, we will note in the remainder $e(i,j)$ the edge connecting the points with indexes i and j.

4 Application to motion editing

Assuming that the motion edition is performed by moving some control points of the graph both

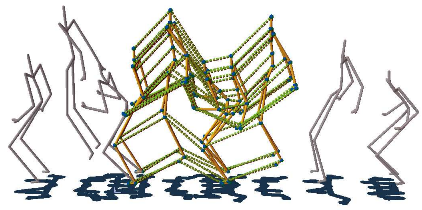

Figure 1: Illustration of the extraction and construction of G. Postures of reference motion are represented in grey. Blue points represent the geometrical information. The connectivity of the graph is represented in dashed and yellow. The yellow edges are extracted from the segments of skeletons and in dashed green the new edges created from joints (blue points) from a posture to another.

in space and time, our goal is to solve an optimization problem involving the concepts presented in the above Section. Recalling $\boldsymbol{V}$ being the set of vertices from the graph G, let $\mathbf{U} = \mathbf{u}_1, ..., \mathbf{u}_m$ be a set of m control points ($m << n$). Our objective is to compute a new spatial configuration of $\boldsymbol{V'} = \boldsymbol{v}'_1, ..., \boldsymbol{v}'_n$ that minimizes the classical quadratic function [11]:

$$E(\boldsymbol{V}) = E_l(\boldsymbol{V}, G) + E_p(\boldsymbol{V}, \boldsymbol{U}). \tag{3}$$

The first term $E_l(\boldsymbol{V}, G) = \sum_{i=1}^{n} \|\delta_i - \mathcal{L}(\boldsymbol{v}_i)\|^2$ penalizes the differences between the differential coordinates after reconstruction, and the second one $E_p(\boldsymbol{V}, \boldsymbol{U}) = \sum_{i=1}^{m} \|\boldsymbol{v}_i - \boldsymbol{u}_i\|^2$ penalizes the change of positions of control points. The new graph respects the constraints of the control points that maintains as much as possible Laplacian properties of the graph, thus resulting into a natural deformation of the graph. However, following this line has major drawbacks in our case:

- The weight w_{ij} defined introduced by Equations (1) and (2) strongly influences the optimization result and may lead to inconsistent dynamic behaviors,
- The deforming graph looks coherent as long as the rotations, induced by the control points, have low amplitudes,
- the bones length of the underlying skeleton are not preserved in this minimization.

We propose to consider this three problems in the following paragraphs.

4.1 Laplacian weights

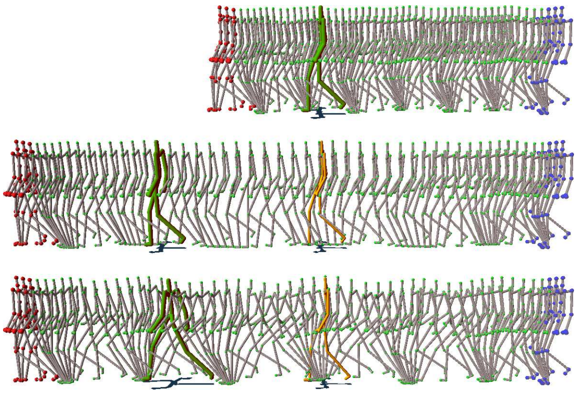

Figure 2: Example of weight in the construction of the Laplacian matrix for a walking motion: at the top the movement without deformation; in the middle with uniform weights, and at the bottom with an exponential parameterization. The blue and the red points represent the control points of the graph and the green ones represent the positions of the joints. In yellow, the skeleton of the original animation, and in green the resulting animation skeletons after deformation.

In the mesh editing problem, the Laplacian operator is generally tuned by uniform weights [18] or cotangent weights [19]. In our method, the dual space/time nature of the graph makes it mandatory to consider differently the edges related to time and space. Setting of the Laplacian operator with uniform weights is not suitable, since it results in smoothing the output graph. We therefore introduce an exponential parameterization of the Laplacian matrix in order to give more importance to some edges of the graph than others. Thus, w_{ij} are then defined by:

$$\mathsf{w}_{ij} = \begin{cases} 1 & (i,j) \in E_S \\ 1 + \alpha e^{(-\beta \frac{d_{ij}^2}{d_{max}^2})} & (i,j) \in E_T \\ 0 & \text{otherwise} \end{cases}$$

where α and β are the coefficients given by the user. α is used to define the importance of "time" edges by comparison with the edges of rest of the graph, and β is used to set the intensity of the exponential. d_{ij} is the Euclidian distance corresponding to the edge connecting vertex i and j, and d_{max} is the maximum length of edges in $\boldsymbol{E}_T$. Figure 2 shows the interest of introducing such a weight. By comparing the reference movement (at the top of the figure) and the one which is defined by the new weight w_{ij}, we observe a best preservation of the dynamics around the feet, and in particular a strong limitation of the sliding effect.

4.2 Rotational constraints

Laplacian coordinates are invariant to translations, but sensitive to linear transformations. The graph structure can be translated, but it can not be rotated. As shown in Figure 3, it is essential to introduce a control of the rotations, since the absence of rotations completely distorts the motion. We suggest that a rotation induced by control points directly impacts groups of points within the skeletons. We therefore propose to introduce a new rotation term T_k, which is applied to $\boldsymbol{\delta}_i$. Index k corresponds to the k^{th} posture of the animated sequence:

$$E_l(\boldsymbol{V}, G) = \sum_{i=1}^{n} \|T_k \boldsymbol{\delta}_i - \mathcal{L}(\boldsymbol{v}_i)\|^2. \quad (4)$$

Rotations are induced by control points that are directly applied on various skeletons of the graph. In order to calculate T_k, we use a linear spherical interpolation (with quaternion) between skeletons.

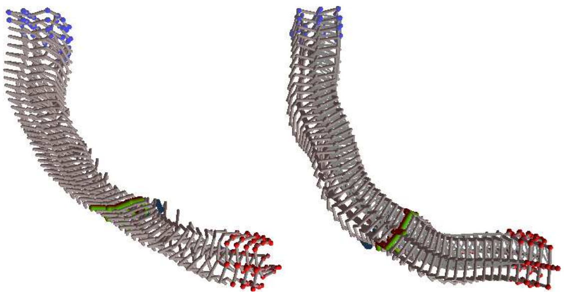

Figure 3: Example of graph editing on a walking motion with rotations (right), and without (left). In green the animation of the resulting skeleton, in grey the graph, and in blue and red the control points.

4.3 Bone lengths preserving constraints

The vector $\boldsymbol{V'}$ which minimizes E does not preserve distances between vertices of the graph. However, many distances between adjacent joints of skeletons are invariant over time and should be preserved through the edition of G.

The following energy term is therefore added:

$$E_b(\boldsymbol{V}, G) = \sum_{e(a,b) \in E_S} (\|\boldsymbol{v}_a - \boldsymbol{v}_b\|^2 - d_{ab}^2)^2, \quad (5)$$

with d_{ab} the desired distance of the bone between the vertices a and b.

Finally, we can rewrite equation 3 with the new added set of constraints:

$$E = w_l E_l + w_u E_p + w_d E_b. \quad (6)$$

where w_l , w_u and w_d are respectively the associated weights at Laplacian, positional and distance constraints. The choice of these weights is discussed in the experiments Section. Yet, it is noticeable that the last distance constraints adds severe non-linearities to the energy function E, which renders its minimization dependent to local minima, and eventually hard to achieve. We propose in the next Section an original and efficient way to solve this problem.

5 An efficient algorithm to minimize E

Most of the studies that associate the Laplacian properties with distance constraints [1, 10], combine in an iterative and unique system linear and non-linear constraints. Here we present an annealing-based alternation algorithm which decouples the resolution of the linear and non-linear part of E.

Relaxing the distance constraints We first relax the constraint distance by expressing Equation(5) with an equivalent linear problem. This problem considers the differential vector between vertex a and b $\boldsymbol{\gamma}_{ab} = -\boldsymbol{\gamma}_{ba} = \boldsymbol{v}_a - \boldsymbol{v}_b$. Indeed, let us note that its norm $\|\gamma_{ab}\| = \|\gamma_{ba}\| = d_{ab}$. From E_S the set of adjacent edges of G with fixed lengths, we define the constraints of invariant distances associated to these edges by the quadratic function E_d:

$$E_d(\boldsymbol{V'}, \boldsymbol{\Gamma}) = \sum_{e(i,j) \in E_S} \|\gamma_{ij} - \mathcal{D}_{ij}(\boldsymbol{V'})\|^2, \quad (7)$$

where $\mathcal{D}_{ij}(\boldsymbol{V'}) = \boldsymbol{v}_i - \boldsymbol{v}_j$, and $\boldsymbol{\Gamma}$ is a vector stacking all possible values of γ_{ij}. Minimizing this energy function amounts to match in terms of norm and alignment the vectors $\boldsymbol{v}_i - \boldsymbol{v}_j$ and γ_{ij}. It is equivalent to solving the following linear system in the least square sense:

$$D\boldsymbol{V'} = \boldsymbol{\Gamma} \quad (8)$$

where D is a matrix of size card(E_S)$\times n$ which role is to compute all the differential coordinates of each segment $\in E_S$. We see now how to determine the differential vector $\boldsymbol{\Gamma}$ which will be the target of our minimization.

Determining $\boldsymbol{\Gamma}$ $\boldsymbol{\Gamma}$ is in fact the solution of the minimization of the equation considered alone, but which initialization is provided by a first guess solution of the problem without considering the distance constraint. The solution is obtained through a gradient descent approach which simply tries to modify $\boldsymbol{V}$ to fit the distance constraints.

Putting all together The energy function E is now:

$$E = w_l E_l + w_u E_p + w_d E_d. \quad (9)$$

The last term E_d is now linear but depends on the $\boldsymbol{\Gamma}$ term which has to be found. After building the 3D+t graph of the motion, one can operate the minimization of $E_l + E_f$ efficiently. This first solution gives a good estimate for the resolution of $\boldsymbol{\Gamma}$, thanks to a simple gradient descent conducted on $E_b(\boldsymbol{V}, G)$ (equation 5). We then solve for the entire system

$$\begin{pmatrix} L \\ w_u U \\ w_d D \end{pmatrix} V' = \begin{pmatrix} \boldsymbol{\Delta_R} \\ w_u \boldsymbol{u} \\ w_d \boldsymbol{\Gamma} \end{pmatrix} \quad (10)$$

for which we can not assure that the bone length constraints will be preserved, but thanks to the term $E_d(\boldsymbol{V'}, \boldsymbol{\Gamma})$ will not be orthogonal to the previous solution. We can now solve again for a new $\boldsymbol{\Gamma}$, and for a new solution, until convergence (no significant changes in the solution $\boldsymbol{V'}$). The addition of constraints implies that our linear system is over-determined, so there is no exact solution. These are the weights w_u and w_d defined by the user which allow us to give more or less importance to the constraints. In our case, distance constraints are more important than the others so to guarantee it we use $w_d > w_u$. The system is full rank and therefore has a unique solution in the least squares sense. The solution vector is calculated through a Cholesky factorization. In addition, the matrix of the first member of the Equation (10) being strongly sparse, efficient sparse Cholesky matrix decomposition can be readily used for optimized performances. In practice, only a few iterations of this loop is required to solve for the system. The algorithm is given in the following synopsis:

Algorithm 1: MOTION EDITING PROCESS

Data: $\boldsymbol{V}$: Vector of vertices of original motion
$\boldsymbol{V'}$: Vector of new position after editing
$\boldsymbol{U}$: A set of position constraints
G: the original motion graph

$\boldsymbol{V'} = \text{argmin}_{\boldsymbol{V}} E_l(\boldsymbol{V}) + E_p(\boldsymbol{V})$
//Laplacian and positionnal constraints;
while *not convergence* **do**
  form $\hat{\boldsymbol{\Gamma}}$ from $\text{argmin}_{\boldsymbol{V}} E_b(\boldsymbol{V}, G)$ (equation 5), preconditionned with$\boldsymbol{V'}$;
  $\boldsymbol{V'} = \text{argmin}_{\boldsymbol{V}} \begin{pmatrix} w_l L \\ w_u U \\ w_d D \end{pmatrix} \boldsymbol{V} = \begin{pmatrix} \boldsymbol{\Delta_R} \\ w_u \boldsymbol{u} \\ w_d \hat{\boldsymbol{\Gamma}} \end{pmatrix}$

Motion	frame count	vertex count	Time to build the Cholesky syst. (sec.)	Time to update syst.	RDE for first frame	RDE after 10 frames
Fig 2	254	2910	12.48	0.15	0.031	0
Fig 3	254	2910	12.06	0.16	0.01	0
Fig 4(a)	115	1323	3.2	0.064	0.031	0
Fig 4(b)	115	1323	3.17	0.068	0.04	0
Fig 4(c)	115	1323	3.2	0.063	0.01	0

Table 1: Table of results for different motions.

6 Results

Our goal here is to provide a complete and flexible system, easy to use, while being able to respond to the issues raised by motion editing or retargeting. Our system is interactive, and therefore in real time. The only time consuming process is the pre-computing of the Cholesky system, performed off line. In the different experiments, the user can choose and edit one or several control points of the graph, and specify distance constraints between several points of the skeleton or environmental objects. The code has been written in C++ and our application has been tested on a computer with 2.7 Ghz for 3GB of RAM.

6.1 Motion editing

The first application tackles the general problem of motion editing. From the specification of fixed points of the graph and other moving control points, our Laplacian-based solver allows us to significantly edit the movement while maintaining spatio-temporal features. Our method is adapted to any type of motion involving high amplitude rotation, such as walking, running, boxing, etc., and several kinds of editing such as translations and rotations. One of our experiments consists of rotating or translating a walking motion (see attached video).

In another experiment 4, we have edited a motion provided by the CMU database which contains many information including body movement, accelerations, decelerations and highly dynamic movements of the feet. The objective of this demonstration is to spatially edit the position of the right foot during contacts.

Figure 5 shows the convergence of our method, both for the distance energy and the global system energy, after 10 iterations of the minimization process. This shows that the global energy E of the system decreases, while enforcing the bone length constraints in a few iterations. This illustrates the success of our alternating minimization scheme.

Our editing method is then compared to another method based on conditional stochastic simulation proposed by [8] for similar goals.

Visually our system meets our objectives: the right foot is able to reach the targets while the rest of the animation is preserved. The extracted curve illustrated in Figure 4 shows that the acceleration (revealing the dynamics of the movement) is properly respected. The curve of the retargeted sequence has indeed the same characteristics than the one of the ground truth motion and the reference motion. We can also observe that our method keeps the contact of the left foot with the ground when compared to other meth-

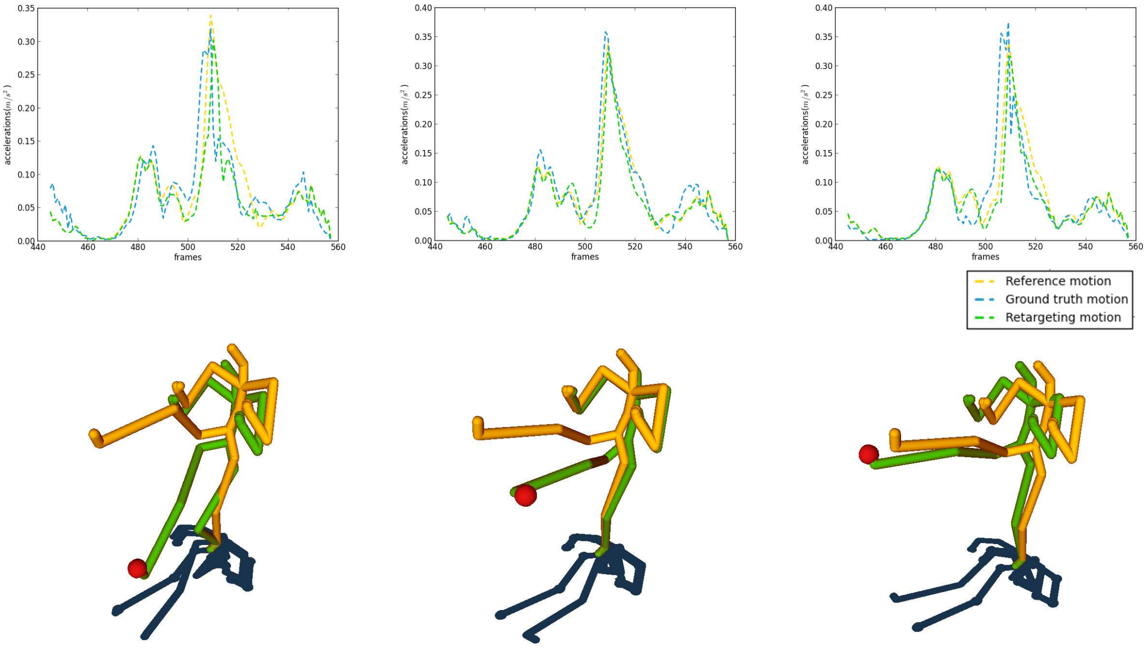

Figure 4: Comparison of three cases of right feet target with reference motion and ground truth motion.

ods where there is a sliding foot effect. The resulting animations can be visualized in the associated movie.

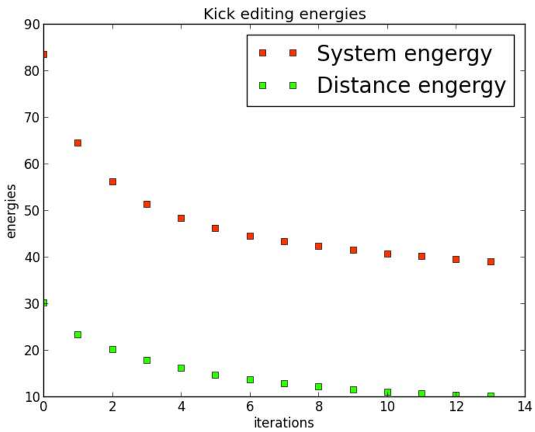

Figure 5: Convergence of the system and distance energies for example 4.

We have compared various sequences of motion editing whose results concerning computing time or constrainst errors are presented in table 1. Our framework presents a reasonable display, considering the frame-rate per second, for a graph up to 3000 vertices. As for the constraints, we have used the coefficient $w_u = 1$ and $w_d = 2$. These weights have been chosen in order to preserve, in priority, the lengths of the skeleton segments. We define RDE by:

$$RDE = \frac{\sum_{e(i,j)\in E_S} \frac{\|v_i^l - v_j^l\|}{d_{ij}}}{|E_S|} \tag{11}$$

with v_i^l and v_j^l being the vertices to determine, from the edge $e(i,j)$ and the original length d_{ij}. We observe that the results are satisfactory since the error RDE after 10 iterations is null.

6.2 Motion retargeting

Our method is also appropriate for motion retargeting, when the goal consists in adapting an animated motion from one character to another. As Gleicher [3], we focus here on morphological adaptation, the articulated figures having an identical structure but different segment lengths. It is easy indeed with our method to add in the reference graph new constraints of distances extracted from another skeleton. Before launching the minimization, specific motion features are identified as constraints that have to be maintained during the animation. The nature of these constraints are naturally positions or distances. During the minimization, the Laplacian operator preserves the space-time properties of the motion. In this section, we demonstrate the capacity of our approach to apply motion capture extracted from a same reference motion on various mesh animations with different morphologies.

In order to test our method, we firstly build two target skeletons manually. We then align the two skeletons at the same scale factor and select some vertices as position constraints, before performing the minimization. This step is done to preserve some close contacts. The results are illustrated in Figure 6. We can observe visually that our system is robust to strong morphological differences, and this visual interpretation is confirmed by measuring RDE (difference of rate between the lengths of the segments). Here also, the Laplacian operator associated with distance constraints maintains the dynamical nature of the movement. The method is also applied successfully to the interaction of retargeted characters, as illustrated in the accompanying video. In this experiment, some edges have been added in the graph in order to preserve the interaction. Moreover the applications on the mesh deformation show that the results are coherent. Finally, this experiment shows the possibility to combine motion editing with motion retargeting.

7 Discussion and perspectives

We have proposed a new representation of the motion by a so-called 3D+t graph, gathering the geometrical structure of skeletons over time. With this new motion representation, we presented a motion editing method which is able to preserve the essential features of the movement and in particular its **dynamics** as well as its **spatiotemporal** properties. Our method runs at interactive frame rates and can handle a broad range of motions, with the benefits of an easy and intuitive control for the user. Yet, one major drawback our system is that it only produces the positions of the skeletons' joints over time (and not rotations). That being said, and as stated in [1], it is quite possible to use inverse kinematic to find the rotations of joints, or we can use recent techniques [20, 21] to deform the mesh using only the positions.

Perspectives work will consider the preservation of dynamic properties of the motions (like zero moment point or projection of the center of mass) by tuning the weights of the graph. Also, the 3D+t Laplacian, as it encodes the structural information of the motion, seems adapted to build low dimensional spaces for motions that could be used efficiently in a motion retrieval contest, or in more complicated retargeting operations, involving interactions with the environment or other characters.

References

[1] Edmond S. L. Ho, Taku Komura, and Chiew-Lan Tai. Spatial relationship preserving character motion adaptation. *ACM Trans. Graph.*, 29(4):33:1–33:8, July 2010.

[2] Franck Hétroy. A discrete 3D+t Laplacian framework for mesh animation processing. Research Report RR-8003, INRIA, June 2012.

[3] Michael Gleicher. Retargetting motion to new characters. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 33–42. ACM, 1998.

[4] Benoît Le Callennec and Ronan Boulic. Interactive motion deformation with prioritized constraints. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, pages 163–171, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[5] C. Hecker, B. Raabe, R. W Enslow, J. DeWeese, J. Maynard, and K. van Prooijen. Real-time motion retargeting to highly varied user-created morphologies. *ACM Trans. on Graphics (TOG)*, 27(3):1–11, 2008.

[6] J. Chai and J.K. Hodgins. Constraint-based motion optimization using a statistical dynamic model. *ACM Trans. on Graphics*, 26(3):686–696, July 2007.

[7] L. Ikemoto, O. Arikan, and D. Forsyth. Generalizing motion edits with gaussian processes. *ACM Trans. Graph.*, 28(1):1–12, 2009.

Figure 6: Comparison of three cases of right feet target with reference motion and ground truth motion.

[8] Nicolas Courty and Anne Cuzol. Conditional Stochastic Simulation for Character Animation. *Computer Animation and Virtual Worlds*, pages 1–10, 2010.

[9] S. Tak and H.-S. Ko. A physically-based motion retargeting filter. *ACM Trans. On Graphics (TOG)*, 24(1):98–117, 2005.

[10] Yanlin Weng, Weiwei Xu, Yanchen Wu, Kun Zhou, and Baining Guo. 2d shape deformation using nonlinear least squares optimization. *Vis. Comput.*, 22(9):653–660, September 2006.

[11] Marc Alexa. Differential coordinates for local mesh morphing and deformation. *The Visual Computer*, 19:105–114, 2003.

[12] Yaron Lipman, Olga Sorkine, Daniel Cohen-or, David Levin, Christian Rssl, and Hans peter Seidel. Differential coordinates for interactive mesh editing. In *In Proceedings of Shape Modeling International*, pages 181–190. Society Press, 2004.

[13] Olga Sorkine. Differential representations for mesh processing. *Comput. Graph. Forum*, 25(4):789–807, 2006.

[14] T. Kwon, K. H. Lee, J. Lee, and S. Takahashi. Group motion editing. *ACM Trans. on Graphics*, 27:80:1–80:8, August 2008.

[15] S. Takahashi, K. Yoshida, T. Kwon, K. H. Lee, J. Lee, and S. Y. Shin. Spectral-based group formation control. *Computer Graphics Forum (Proc. Eurographics 2009)*, 28(2):639–648, 2009.

[16] J. Henry, H. Shum, and T. Komura. Environment-aware real-time crowd control. In *Symposium on Computer Animation SCA*, pages 193–200, July 2012.

[17] Manmyung Kim, Kyunglyul Hyun, Jongmin Kim, and Jehee Lee. Synchronized multi-character motion editing. *ACM Trans. Graph.*, 28(3):79:1–79:9, July 2009.

[18] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '04, pages 175–184. ACM, 2004.

[19] Mark Meyer, Mathieu Desbrun, Peter Schrder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds, 2002.

[20] Alexander Belyaev, Hans-Peter Seidel, and Shin Yoshizawa. Skeleton-driven laplacian mesh deformations. Research Report MPI-I-2006-4-005, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany, November 2006.

[21] C. Budd and A. Hilton. Skeleton driven laplacian volumetric deformation. In *Proceedings of the 2009 Conference for Visual Media Production*, CVMP '09, pages 61–68. IEEE Computer Society, 2009.