Real-time mesh animation from low dimensional positional constraints

Thibaut Le Naour, Nicolas Courty, Sylvie Gibet

Real-time mesh animation from low dimensional positional constraints

1st Thibaut Le Naour
IRISA-UBS
University of Bretagne Sud
56000 Vannes, France
tlenaour@gmail.com

2nd Nicolas Courty
IRISA-UBS
University of Bretagne Sud
56000 Vannes, France
courty@univ-ubs.fr

3rd Sylvie Gibet
IRISA-UBS
University of Bretagne Sud
56000 Vannes, France
City, Country
sylvie.gibet@univ-ubs.fr

***Abstract*—The challenge proposed by this study is to re-consider the entire animation pipeline for data-driven character animation. By observing that significant loss of information and precision occur in the traditional animation pipeline (skeleton reconstruction from markers, rigging and retargeting), our goal is to directly control at interactive framerates articulated meshes from a low number of positional constraints. Our method builds on top of efficient deformation techniques and proceeds as follows: an original mesh is embedded into a coarse volumetric control lattice which contains simplified information from the initial reference mesh, skeleton elements and marker locations. An iterative method is applied on this structure which preserves the geometry details, the bones lengths, and the associated joint limits. We show the ability of our approach to animate and interactively deform high resolution models from a low-number of markers while retaining the subtleties of the motion. It notably allows to entirely skip the tedious rigging phase.**

***Index Terms*—Computer Graphics, Character Animation, Computational Geometry and Modeling**

I. Introduction

Interactive and natural looking shape animation and deformation of articulated bodies is one of the key topics in computer graphics. Many techniques have been proposed with different approaches from the perspective of data-driven animation, *e.g.* skeleton-based, or positional-constraints-based animation. The traditional pipeline of character animation is classically composed of two stages: first, the skeleton-driven animation, *i.e.* joints rotations over time, is computed from positions of motion capture markers. This process can be rather approximate, in particular for determining articulatory centers and axis of rotation, especially for complex articulatory systems such as hands. It sometimes requires the intervention of designers or automatic processing involving inverse kinematics. Secondly, the mesh vertices of an existing 3D mesh are bound to one or several joints of the skeleton through a rigging operation. This last step is a tedious task which is often conducted manually by infographists. Several drawbacks characterize this animation pipeline: (i) difficulties to adapt the approximate skeleton with the mesh , generally implying retargeting techniques that tend to alter the original data; (ii) difficulties to deal with too few markers , which most often amplifies the manual post-processing and correction steps; (iii) limited control over the mesh through the skeleton; the inconsistency between the skeleton and mesh models leading consequently to some incoherencies (mesh collisions, unnatural foldings or deformations, etc.) along the animation.

The challenge proposed by this study is to re-consider the entire animation pipeline for data-driven character animation. We focus especially on the animation of complex systems driven by low dimensional positional constraints, and in particular by few trajectories of markers. Our approach is strongly inspired by shape deformation techniques which use markers as control points to animate the character's mesh [Krayevoy and Sheffer(2005)], [Stoll et al.(2007)Stoll, de Aguiar, Theobalt and Seidel]. However our method differs from these works as it implicitly integrates all the properties induced by an underlying skeleton (the segment lengths and the joint limits) while preserving the details of the mesh geometry. The introduction of skeleton specific constraints compensates for the use of a small number of input data. As a consequence, our approach enables to control the mesh structure by using few trajectories of captured marker positions. It also gives the opportunity to drive the animation by editing manually some mesh vertices, or by using solely positions of the skeleton joints as control inputs. It is demonstrated over several examples of articulated bodies animation. We focus in particular on a difficult case which is the animation of high quality hand models. Indeed these complex articulated models include many degrees of freedom and constraints that make the operation of rigging tedious and pore to errors.

Methodology. Our novel interactive mesh animation method relies on a volumetric control mesh, deformed by control points. The different steps of the approach are illustrated by the Figure 1. More specifically our method follows the following processes: (i) the control mesh is built as a tetrahedral mesh which has the property of two-manifold, implying no intersection and no hole, thus leading to consistent deformation properties. It includes the original mesh and takes into account internal constraints derived from the skeleton joints, and external control points representing the markers; (ii) Then, we apply a volumetric deformation preserving the surface details and skeleton properties; consequently the deformation process respects the constrained positions induced by the markers; (iii) Simultaneously, we use a skeleton described by

a hierarchy of joints updated at each step of the deformation, and satisfying joint limits. These properties are then backward integrated into the previous step; (iv) As suggested by Borosan [Borosán et al.(2010)Borosán, Howard, Zhang and Nealen], we optionally improve this last step by using an intermediate and simplified coarse mesh including the skeleton and markers properties. More precisely, the method consists in applying the deformation on this coarse mesh that drives the original mesh by using a cage-based method called *Green Coordinates* (*GC*) [Lipman et al.(2008)Lipman, Levin and Cohen-Or].

Fig. 1. Our animation pipeline: with a short number of markers' trajectories (left image), our method first start by building a volumetric control mesh (middle image) which embeds the rigidity information of the mesh expressed as a skeleton endowed with joint limits; this control mesh is then used to animate the full resolution mesh (right image)

Contributions. Our method builds on top of recent deformation and marker-based animation techniques. Its originality stems from a strong coupling of an internal skeleton structure with the deformation process. This coupling is operated inside an optimization procedure which alternates an efficient volumetric shape deformation and the enforcing of skeleton specific constraints. As a result, it allows to bypass the skeleton reconstruction of the original animation pipeline and control directly the mesh from the motion capture markers. Contrary to previous works, one of the main advantage is to allow the introduction of non-linear constraints such as the enforcing of joint limits. The resulting mesh animation is characterized by an enhanced rigidity and volume preservation of the mesh structure. Our method also implicitly embeds a retargeting step, which allows to conveniently transfer animations to multiple target meshes. Finally, it is robust and particularly well suited for interactive character animation.

After reviewing previous work in Section 2, we will describe in Section 3 the method for constructing the different mesh structures and explain how the deformation method can be applied to these structures for generating realistic animation. We will then show in Section 4 the benefits of our method in several practical applications with multiple sets of data, before concluding and giving perspectives of this work.

II. Background

Many different approaches have been proposed to tackle the problem of generating realistic animation of characters while maintaining plausible shape deformation. Most of them concern the use of a skeleton as a guide to deform and animate the mesh, for example using linear skinning [Lewis et al.(2000)Lewis, Cordner and Fong] or using dual quaternions interpolation [Kavan et al.(2008)Kavan, Collins, Žára and O'Sullivan], and more recently a geometric technique to make realistic folding around joints [Vaillant et al.(2013)Vaillant, Barthe, Guennebaud, Cani, Rohmer, Wyvill et al.]. However, these methods animate meshes only through the control of a skeleton. With our goal to drive the mesh with marker trajectories, we focus on the two following thematic: animation of models driven by markers, and mesh deformation using skeleton constraints.

Animation driven by markers

Several studies have investigated marker-driven animation techniques. In this line of research, Park et al. [Park and Hodgins(2006)] propose a technique to animate mesh models from 3D captured motion. The binding between the mesh and the markers is achieved by establishing a skinning relationship obtained through the segmentation of rigid areas of the mesh. The animation results, which integrate the geometry variations induced by muscles' deformations, are very realistic. However, their method requires a large number of markers (several hundred). On the contrary, the method proposed by Sumner et al. [Sumner et al.(2005)Sumner, Zwicker, Gotsman and Popović] proposes the animation of a mesh from a low number of control points, thanks to a learning space determined from mesh samples. Recently [Wheatland et al.(2013)Wheatland, Jörg and Zordan] used also a learning method for producing high quality hand motion using a small number of markers by establishing the correspondence between markers and joint angle trajectories with the help of a reference database. Zordan et al. [Zordan and Van Der Horst(2003)] also propose a dynamical system driven by 3D captured motion, which guarantees several physical properties: the global rigidity of the system, the preservation of segment lengths and the consideration of joint limits. This approach is closely linked to ours, but is applied to physical bodies instead of geometrical shapes, and therefore it does not take into account all the geometrical details during the animation.

Closer to our approach, Krayevoy [Krayevoy and Sheffer(2005)] and Stoll [Stoll et al.(2007)Stoll, de Aguiar, Theobalt and Seidel] use a deformation technique called *Mean-value Encoding* for the former one, and the Laplacian operator for the latter one to directly drive the mesh by markers. Their objectives are the same as ours, but the first approach does not take into account the properties of the skeleton and requires a morphology consistent with the cloud of markers as input. The second study proposes to use a coarse volumetric mesh that enables interactive shape editing. The method alternates a Laplacian deformation technique and a gradient-based differential update process, making it

stable even for large deformations. It produces high-quality mesh animations from motion capture data. Although very promising, the method does not incorporate same skeleton properties such as the length of bones or joint limits, which are essential to guarantee natural deformations.

Mesh deformation

Other approaches directly focus on mesh deformation and edition techniques, without skeletal information. In order to produce high-quality deformations, several recent works have been interested in preserving surface details of the geometry. Some of them, based on Laplacian differential coordinates [Sorkine et al.(2004)Sorkine, Cohen-Or, Lipman, Alexa, Rössl and Seidel], [Alexa(2003)], [Lipman et al.(2004)Lipman, Sorkine, Cohen-or, Levin, Rssl and peter Seidel], facilitate interactive mesh editing while producing natural-looking deformations, but they fail to handle large rotations. In order to solve the problem of large deformations, several methods derived from this approach have been developed. In particular, the $ARAP$ deformation paradigm, which stipulates that the mesh is directly linked to the local rigidity of the shape, has been proposed to edit surface meshes subject to large amplitude rotations [Sorkine and Alexa(2007a)].

These surface differential methods have been extended to volumetric graph Laplacian representations, with the idea to preserve the volume during the deformation, and to enable large deformations (i.e [Zhou et al.(2005)Zhou, Huang, Snyder, Liu, Bao, Guo et al.], [Huang et al.(2006)Huang, Shi, Liu, Zhou, Wei, Teng et al.], [Stoll et al.(2007)Stoll, de Aguiar, Theobalt and Seidel], [Zhao et al.(2009)Zhao, Liu, Peng and Bao], [Jacobson et al.(2011)Jacobson, Baran, Popović and Sorkine], [Zhao and Liu(2012)]). Among them, several include skeleton properties. For example, Huang et al. [Huang et al.(2006)Huang, Shi, Liu, Zhou, Wei, Teng et al.] present a subspace iterative solver for mesh deformation constrained by skeleton and volume properties. Their system enables to maintain the segment lengths during the deformation, but fails to preserve joint limits. This method has been recently extended by Zhao [Zhao and Liu(2012)] to reach the volume preservation during the deformation. Closer to our objectives, Shi et al. [Shi et al.(2007)Shi, Zhou, Tong, Desbrun, Bao and Guo] present a technique that preserves the same previous properties with the approximate respect of joint limits. Indeed, their system enables to satisfy an angle constraint along a plane, but does not exactly preserve the joint limits on the three degrees of freedom. Recently Jacobson et al. [Jacobson et al.(2011)Jacobson, Baran, Popović and Sorkine] use bounded biharmonic weights to intuitively deform geometries associated with a skeleton structure.

Concerning the methods using *ARAP*, Zhang et al. [Zhang et al.(2010)Zhang, Nealen and Metaxas] have demonstrated the benefit to introduce the concept of rigidity in the interior of the surface mesh to improve the preservation of the volume. Zhao [Zhao et al.(2009)Zhao, Liu, Peng and Bao] propose to apply ARAP with a volumetric graph associated with rigidity constraints in particularly to avoid unnatural volumetric distortions. Latter Borosan [Borosán et al.(2010)Borosán, Howard, Zhang and Nealen] combine ARAP with the mean coordinate values method to deform faster high-resolution meshes. Faraj and all [Faraj et al.(2012)Faraj, Thiery, Bloch, Varsier, Wiart and Boubekeur] also propose in a problem of volumetric data editing to use ARAP on cage structures before applying the deformation with the Green Coordinate method. The other benefit to use a lattice structure is to deform all kinds of meshes since the mesh to deform needs to be without self intersection and holes.

A more general method than the ARAP principle uses the concept of elastic energy. Introduced by Chao [Chao et al.(2010)Chao, Pinkall, Sanan and Schröder], it allows to produce high quality deformation on tetrahedral models and presents several advantages as the preservation of the rotation and the robustness of the method for large transformations. This concept has been extended first in the work of McAdams et al. [McAdams et al.(2011)McAdams, Zhu, Selle, Empey, Tamstorf, Teran et al.], and then in Kavan et al. work [Kavan and Sorkine(2012)] to solve the general problem of skinning around character articulation. However, as the classical skinning methods, these techniques need a good determination of the weights applied to each vertex of the mesh.

Our approach inherits the advantages of recent differential techniques, applied on volumetric meshes for animation of characters driven by captured motion points. We use a tetrahedral mesh representation of the articulated system, thus ensuring the preservation of the internal mesh volume and avoiding joint artifacts. In addition, we integrate within this mesh skeleton the internal skeleton structure with its rigidity properties: length of bones, joint limits, etc.

III. System Overview

Our approach aims at animating and deforming articulated bodies. This deformation can be possibly driven by motion capture markers, and the following description does not make any assumption on the source of animation data. In Figure 2 an illustration of the pipeline is given through an example of a hand animated by motion capture markers. Those markers are located at specific places over the hand's skin, with no assumption on their positioning with respect to joints. From a highly detailed reference mesh of a hand, that can be morphologically different from the captured hand, a volumetric control mesh is automatically constructed. This volumetric control mesh is built by directly taking the reference mesh and by adding the skeleton structure and marker points (note that the binding between the markers and the reference mesh is established by the volumetric control mesh). In the remainder of this paper we will call *original mesh* the reference mesh, and *control mesh*, the volumetric mesh. The control mesh is deformed by minimizing over time the distances between the marker positions and the corresponding vertices of the control mesh. This deformation tends to preserve the shape of the original mesh while taking into account the rigidity constraints induced by the skeleton. Note that the original mesh is included into the control mesh, thus the deformation

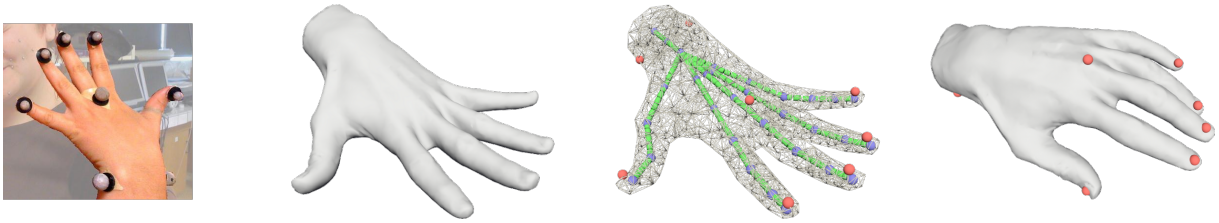

Fig. 2. Different steps of our method. 1: hand with reflexive markers. 2: the original mesh. 3: volumetric mesh integrating skeleton data and marker positions according to the original hand markers. 4: one posture produced by our system.

of the control mesh induces the deformation of the original mesh.

IV. Volumetric mesh deformation with skeleton constraints

In this Section, we detail the different operations of our animation pipeline. We first begin by explaining how the control mesh is built. We then proceed with the deformation pipeline, with a focus on the preservation of the skeleton properties. The deformation of the original mesh with Green coordinates is then detailed, before giving an overall view of the complete algorithm.

A. Pre-processing and mesh-skeleton coupling

Regarding the skeleton, several methods are available to automatically create it with respect to a reference mesh (*e.g* [Au et al.(2008)Au, Tai, Chu, Cohen-Or and Lee]), but it is also possible to construct it manually with a commercial software (this step takes a few minutes and does not require specific knowledges). At this step, no explicit relations exist between the mesh and the skeleton apart from the relative positioning in the 3D space.

Then we create the volumetric mesh by using a Delaunay tetrahedrization applied on the original surface mesh. This tetrahedrization also includes:

- the set of markers positions. They are associated to vertices of the control mesh. They are manually positioned at the right place according to the original hand markers (see Figure 2, step 3) (possibly outside of the original mesh),
- the vertices and edges corresponding to a discretization of the skeleton (blue points and green edges in Figure 2). This discretization is performed by a spatially regular sampling of the skeleton. The length of the sampling step allows to control the number of points associated to the skeleton in the tetrahedrization process, and as such can act as a tradeoff parameter for the overall computational complexity.

The Delaunay tetrahedrization is a complex process and several works have been proposed on the subject [Bridson et al.(2005)Bridson, Teran, Molino and Fedkiw], [Cutler et al.(2004)Cutler, Dorsey and McMillan], [Shewchuk(1998)]. We chose the so-called *piecewise linear complex* (PLC) method designed by Miller et al. [Miller et al.(1996)Miller, Talmor, Teng, Walkington and Wang], and implemented by Hang Si et al. [Si(2006)], which proved to be robust and flexible. In the end, the resulting mesh is closed and composed of triangles, which are then ready to be deformed by the following method.

B. Volumetric ARAP mesh deformation

We present here our volumetric mesh deformation process. We first begin by recalling the basic principles of Laplacian surface editing [Sorkine et al.(2004)Sorkine, Cohen-Or, Lipman, Alexa, Rössl and Seidel], before presenting an adaptation of the ARAP algorithm to consider volumetric cells.

Laplacian Surface Editing (LSE) Let the closed triangularized structure, that is the control mesh characterized by $\mathcal{V} = (\boldsymbol{V}, E)$, where E describes the connectivity, and $\boldsymbol{V} = \boldsymbol{v_1}, ..., \boldsymbol{v_n}$ are the Euclidian coordinates of the mesh vertices. From $\boldsymbol{V}$ and a set $\boldsymbol{U} = \boldsymbol{u_1}, ..., \boldsymbol{u_m}$ of control vertices, we can compute the new positions $\boldsymbol{V'} = \boldsymbol{v'_1}, ..., \boldsymbol{v'_n}$ minimizing the quadratic function given by

$$E(\boldsymbol{V'}) = E_l(\boldsymbol{V'}) + E_p(\boldsymbol{V'}). \tag{1}$$

The first term E_l that penalizes the difference between the differential coordinates after reconstruction can be written as

$$E_l(\boldsymbol{V'}) = \sum_{i=1}^{n} \|\boldsymbol{\delta}_i - \mathcal{L}(\boldsymbol{v'_i})\|^2, \tag{2}$$

where $\mathcal{L}$ is the Laplace-Beltrami discrete operator with cotangent weights (for details, see [Pinkall et al.(1993)Pinkall, Juni and Polthier], [Meyer et al.(2002a)Meyer, Desbrun, Schröder and Barr]) and $\boldsymbol{\delta}_i = \mathcal{L}(\boldsymbol{v_i})$ is the differential coordinate for the vertex i. The second term E_p penalizes the changes of position of control points and is defined by

$$E_p(\boldsymbol{V'}) = \sum_{i=1}^{m} \|\boldsymbol{v'_i} - \boldsymbol{u_i}\|^2. \tag{3}$$

Equation 1 can then be minimized as a $(n+m) \times n$ overdetermined linear system

$$\begin{pmatrix} \boldsymbol{L} \\ \boldsymbol{I}_u \end{pmatrix} \boldsymbol{V'} = \begin{pmatrix} \boldsymbol{\Delta} \\ \boldsymbol{U} \end{pmatrix} \tag{4}$$

where $\boldsymbol{I}_u$ is the index matrix of $\boldsymbol{U}$. The reconstructed shape looks generally natural when low amplitude rotations are applied. For large transformations, Sorkine [Sorkine and Alexa(2007b)] proposes a non linear solution called ARAP to preserve the local rigidity of each cell of the mesh.

ARAP modeling From a first estimation of $\boldsymbol{V'}$ given by Equation 1, the goal of ARAP is to find iteratively the new values of differential coordinates preserving the local rigidity. Thus, if the cell C_i of the mesh corresponding to the vertex i is deformed into a cell C'_i, the approximate rigid transformation between C_i and C'_i can be expressed as a rotation matrix which minimizes

$$E(C_i, C'_i) = \sum_{j \in \mathcal{N}(i)} w_{ij} \|(\boldsymbol{v'_i} - \boldsymbol{v'_j}) - \boldsymbol{R}_i(\boldsymbol{v_i} - \boldsymbol{v_j})\|^2 \quad (5)$$

where $\mathcal{N}$ is the set of vertices connected to the vertex i and w_{ij} is the cotangent weight of vertices i et j defined by

$$w_{ij} = \frac{1}{2}\,(cot\alpha_{ij} + cot\beta_{ij}) \quad (6)$$

(see [Sorkine et al.(2004)Sorkine, Cohen-Or, Lipman, Alexa, Rössl and Seidel], [Meyer et al.(2003)Meyer, Desbrun, Schröder and Barr] for details). Finally, solving this problem can be rewritten as an iterative problem, each iteration being expressed as $E(\boldsymbol{V'}) = E_a(\boldsymbol{V'}) + E_p(\boldsymbol{V'})$, with E_a defined by:

$$E_a(\boldsymbol{V'}) = \sum_{i=1}^{n} \|\boldsymbol{\delta}_i^A - \mathcal{L}(\boldsymbol{v'_i})\|^2 \quad (7)$$

where $\boldsymbol{\delta}_i^A$ is a term evaluated at each iteration. In the remainder of the paper we use the shorthand notation $\boldsymbol{\Delta}^A = \boldsymbol{\delta}_i^A$.

Volumetric ARAP This algorithm is efficient on surface meshes but can be improved by extending it to a volumetric setting. The cell can now be considered as an element of a 3-manifold, and as such its energy is thus linked not only to its volume but also to its global shape. The Laplace-Beltrami operator is also virtually defined anywhere on this manifold provided that it is differentiable everywhere, which is guaranteed by the tetrahedrization process.

As described by the work of Meyer [Meyer et al.(2003)Meyer, Desbrun, Schröder and Barr], the surfacic Laplacian Beltrami operator can be readily generalized to 3D-manifold. We therefore propose to extend the ARAP surfacic method by using this operator applied on 3D-manifold. The weight w_{ij} is now defined by:

$$w_{ij} = \frac{1}{6} \sum_k l_{i,j}^k \; cot\alpha_{ij}^k \quad (8)$$

where k is the number of tetrahedrons sharing the edge (v_i, v_j), $l_{i,j}^k$ is the length of the edge opposite to (v_i, v_j), and α_{ij}^k is the dihedral angle opposite to the edge (v_i, v_j).

Yet, while this method provides successful results, it does not implicitly allow a good preservation of the skeleton properties: rigidity around segments, length of segments and joints limits (see Figure 3). In the next section we will see how to consider those implicit constraints in the deformation process.

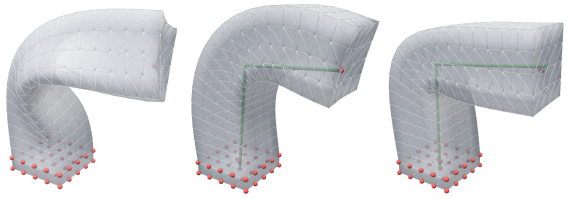

Fig. 3. Example of large deformation with surface ARAP (left) and volumetric (middle and right) methods (100 iterations). Volumetric representation preserves better the volume and is more robust to preserve the shape of the original mesh. The right model is the result of our method incorporating skeleton constraints.

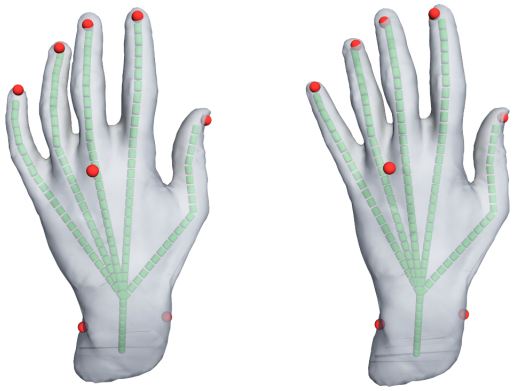

Fig. 4. Example of respect of joint limits for the same control mesh and markers target: left) without joint limits and right) with

C. Preservation of skeleton properties

As illustrated by the Figure 2 (step 3), the skeleton structure is embedded into the tetrahedrization. The positions of joints are represented by a set of vertices and the segments between joints are decomposed into a set of edges that are included into the control mesh. Two steps are considered. First, we introduce a constraint that preserves simultaneously the collinearity between the edges of a same segment and the lengths of segment edges. This constraint produces a deformation induced by a novel rigidity constraint that follows the skeleton structure, as shown in Figure 3. Second, we propose to add joint limits to produce realistic postures. As these new constraints are not linear, and joint limits are hardly expressed with our formulation (no angular representation), we solve this problem in two steps through an alternating descent method:

1) From the current configuration of the skeleton provided by the deformation of the control mesh, we compute a new skeleton posture close to the current one but respecting the skeleton properties,
2) Then, the new posture is introduced into the previous problem in terms of differential constraints.

These two steps are then repeated until convergence. This formulation allows **at the same time** to deform the control mesh by minimizing the previously described volumetric ARAP constraint and maintain the correct properties of the skeleton. A formalization of this method is proposed below.

Skeleton constraints as a linear problem: Each segment of the skeleton is decomposed into several edges. Given e_{ab} the edge between the vertices a and b, we consider the differential vector $\boldsymbol{\gamma}_{ab} = -\boldsymbol{\gamma}_{ba} = \boldsymbol{v}_a - \boldsymbol{v}_b$, and the length defined by $\|\boldsymbol{\gamma}_{ab}\| = \|\boldsymbol{\gamma}_{ba}\| = d_{ab}$. Let us remark that those differential coordinates both encode the direction and the length of the edge. In the following we denote E_S, the set of edges which belong to the skeleton. This can be formalized in the following quadratic energy function E_d:

$$E_d(\boldsymbol{V}') = \sum_{e(i,j)\in E_S} \|\boldsymbol{\gamma}_{ij} - \mathcal{D}_{ij}(\boldsymbol{V}')\|^2, \tag{9}$$

where $\mathcal{D}_{ij}(V') = \boldsymbol{v}_i - \boldsymbol{v}_j$.

Denoting by $\boldsymbol{\Gamma}$ the vector containing all possible values of $\boldsymbol{\gamma}_{ij}$ that match the distances and colinearity criteria, Equation 9 is therefore equivalent to solve the following linear system in the least square sense

$$\boldsymbol{D}\boldsymbol{V}' = \boldsymbol{\Gamma} \tag{10}$$

where $\boldsymbol{D}$, matrix of size card$(E_S)\times n$, is a discrete differential operator which role is to compute all the differential coordinates of every segments of E_S. More precisely, a row of $\boldsymbol{D}$ is given by the values 1 and -1 at the $i-th$ and $j-th$ indices, 0 elsewhere.

Now we turn on to the determination of the $\boldsymbol{\Gamma}$ differential vector.

Determining $\boldsymbol{\Gamma}$***:*** The differential coordinates stored in $\boldsymbol{\Gamma}$ are extracted from a correct skeleton respecting the rigidity constraints as well as its joint limits. This is achieved through a two-steps procedure: *(i)* First, the current approximate posture of the skeleton S_c is extracted from all the vertices that belong to the skeleton. The skeleton is traversed and for each segment, the edges orientations are given by averaging all of the edges of the segment. The lengths of the edges are also restored to their initial, constant values. This intermediate skeleton respects the rigidity constraints but may fail in respecting the joint limits. *(ii)* We then project S_c into the space of admissible postures by simply thresholding the joint angular values to the closest acceptable value. This final step produces a new skeleton S_a.

$\boldsymbol{\Gamma}$ is then directly computed from S_a, through the direct computation of the vectors $\boldsymbol{\gamma}_{ij}$. Figure 4 illustrates the deformation of a hand before and after adding constraints. The posture of the hand looks more natural in the latter case, as the rigidity of all the fingers degrees of freedom is better preserve, and as unnatural rotations are forbidden along the fingers.

Projecting S_c ***into the space of admissible postures:*** We present below: (i) first how we calculate the rotations from the displacement of the joints' positions, and (ii) second how we correct the postures of the skeleton during the deformation.

In a first step, we calculate the orthonormal basis for each joints as illustrated by the figure 5. For example, considering the positions $\boldsymbol{p}_0$, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ of the joints j_0, j_1 and j_2, we

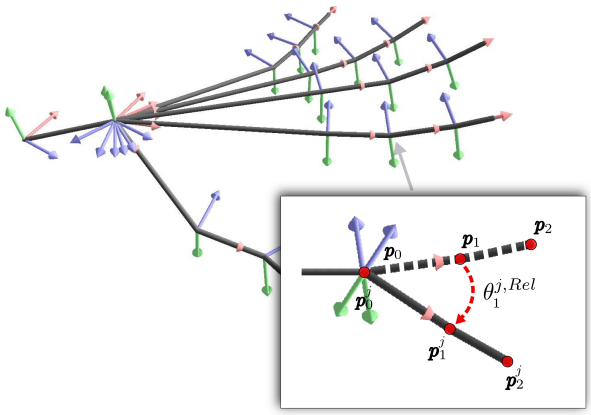

Fig. 5. At the top: example of coordinate systems for each joint of the hand. At the bottom: illustration of a rotation $\boldsymbol{\theta}_1^{j,Rel}$ applied on the joint J_1.

define the Orthonormal basis $\boldsymbol{B}_1$ of j_1 with the axis $\boldsymbol{x}_1$, $\boldsymbol{y}_1$ et $\boldsymbol{z}_1$ by:

$$\begin{cases} \boldsymbol{x}_1 = \boldsymbol{p}_{12}/\|\boldsymbol{p}_{12}\| \\ \boldsymbol{z}_1 = \boldsymbol{x}_1 \times \boldsymbol{p}_{01}/\|\boldsymbol{p}_{01}\| \\ \boldsymbol{y}_1 = \boldsymbol{x}_1 \times \boldsymbol{z}_1 \end{cases}$$

$\boldsymbol{x}_1$ is the *orientational* term and $\boldsymbol{z}_1$ et $\boldsymbol{y}_1$ compose the *rotational* of the joint rotation (see [Aristidou and Lasenby(2011)] for more details). Next we define manually for each joint the joint limits around each axis $\boldsymbol{x}_i$, $\boldsymbol{y}_i$ and $\boldsymbol{z}_i$.

Moreover, the skeleton is classically defined as an hierarchy of joints. Let the frame j and the joint i defined by J_i, the absolute transformation of J_i is defined by $\boldsymbol{B}_i^j = \boldsymbol{B}_{i-1}^j * \boldsymbol{B}_i^{j,Rel}$ where the term Rel means that the transformation $\boldsymbol{B}_i^{Rel}$ is expressed in the coordinate system $\boldsymbol{B}_{i-1}^{j,Rel}$. Finally, $\boldsymbol{B}_i^j = \prod_{k=0}^{k=i} \boldsymbol{B}_k^{j,Rel}$ where $k=0$ corresponds to the root node of the skeleton.

Considering a skeleton with n joints and a vector of relative rotation $\boldsymbol{\theta}_i^{j,Rel} = \{\boldsymbol{\theta}_0^{j,Rel}, ..., \boldsymbol{\theta}_n^{j,Rel}\}$ corresponding to the variation of rotation in each basis $\boldsymbol{B}_i$, the absolute transformation of J_i is then $\boldsymbol{T}_i^\theta = \prod_{k=0}^{k=i} \boldsymbol{B}_k^{Rel} * \boldsymbol{\theta}_i^{Rel}$. Note that $\boldsymbol{T}$ is composed by a rotation and a translation.

We follow then the method of Blow [Blow(2002)] to calculate the angle $\boldsymbol{\theta}_i^{j,Rel}$, by using a quaternion from the absolute posture of the skeleton. As illustrated by the figure 5, we consider for example the vector $\boldsymbol{p}_{01}$ and the vector $\boldsymbol{p}_{01}^j = \boldsymbol{B}_i^{-1} * \boldsymbol{p}_{01}^{j,Abs}$, where $\boldsymbol{p}_{01}^j$ is the vector expressed in the coordinate system of j_i. We determinate the quaternion $\boldsymbol{q}_0 = (\boldsymbol{a_0}, \alpha_0)$ by:

$$\begin{cases} \boldsymbol{a_0} = (\boldsymbol{p}_{01}/\|\boldsymbol{p}_{01}\|) \times (\boldsymbol{p}_{01}^j/\|\boldsymbol{p}_{01}^j\|) \\ \alpha_0 = \arccos\left((\boldsymbol{p}_{01}.\boldsymbol{p}_{01}^j)/(\|\boldsymbol{p}_{01}\| \times \|\boldsymbol{p}_{01}^j\|)\right) \end{cases}$$

As described in the work of Blow [Blow(2002)], it is now easy to correct each quaternions $\boldsymbol{q}_i$ by respecting the joint limits given manually before. Finally, we calculate the final posture of the skeleton by: $\boldsymbol{T}_i^\theta = \prod_{k=0}^{k=i} \boldsymbol{B}_k^{Rel} * \boldsymbol{\theta}_i^{Rel\_corrected}$.

Control mesh deformation: Finally the system computes a new configuration of the control mesh by minimizing the quadratic function:

$$E(\boldsymbol{V}') = w_l E_a(\boldsymbol{V}') + w_u E_p(\boldsymbol{V}') + w_d E_d(\boldsymbol{V}'), \qquad (11)$$

where the weights w_l, w_u and w_d are used to balance between the three energies: Laplacian, positional constraints and distance constraints.

D. Optimization

Fig. 6. Examples of our mesh editing with user input deformations

Our system enables to produce realistic animations, but as suggested by Borosan [Borosán et al.(2010)Borosán, Howard, Zhang and Nealen], the computation time can be easily improved by applying our method to a simplified control mesh which drives then the original mesh.

In this way, a coarse mesh usually known as the *cage mesh* [Meyer et al.(2002b)Meyer, Barr, Lee and Desbrun] is created, which covers the original mesh. Note that for complex structures such as hands, this cage mesh should preserve at best the morphology of the system; especially, it should take into account more details around the joints to best preserve the geometric features for large deformations. Some works have been proposed to create automatically this kind of structure (i.e [Xian et al.(2009)Xian, Lin and Gao], [Xian et al.(2012)Xian, Lin and Gao]). In our application, we simply combined a mesh simplification by edge collapsing and a global upscaling so that the cage mesh covers the original mesh. This resulting cage mesh is characterized by the absence of holes or self intersections. Then, the control mesh is built from this cage mesh and as presented in the section IV-A, we include into it the skeleton and markers information. Finally, as presented in the next paragraph, we use the Green Coordinates method (GC) [Lipman et al.(2008)Lipman, Levin and Cohen-Or] to deform the original mesh.

Let the original mesh denoted by $\mathcal{R} = (\boldsymbol{X}, F)$, where F describes the set of faces of $\mathcal{R}$ and $\boldsymbol{X}$ the set of Euclidan coordinates of the vertices. The control mesh $\mathcal{V}$ is composed of a set of tetrahedrons, also called cages, produced by the Delaunay tetrahedrization. Each cage C is defined by $C = (V_c, F_c)$ where V_c is the set of vertices of the cage (4 in our case), and F_c the set of faces (4 also). The GC method determines an application A which computes the transformation of the vertex x belonging to the cage C into the vertex x' as follows

$$\boldsymbol{x'} = A(\boldsymbol{x}, C') = \sum_{i \in |V|} \phi_i(\boldsymbol{x}) \boldsymbol{v_i'} + \sum_{j \in |F|} \psi_j(\boldsymbol{x}) s_j \boldsymbol{n}(f_j') \quad (12)$$

where C' is the deformed cage of C. The "coordinates" ϕ and ψ, as well as the coefficients s are calculated off line. Their determination is described in [Lipman et al.(2008)Lipman, Levin and Cohen-Or]. As shown in Figure 6 and Table I, this technique allows us to reduce the preprocessing time as well as the computation time during the animation.

E. Algorithm

Finally, we propose to use an iterative algorithm to solve simultaneously the volumetric ARAP deformation and the preservation of skeleton constraints. We evaluate the general error of the algorithm by $0.5 * E_{RRMS} + 0.5 * E_{ED}$ where E_{RRMS} is the relative root mean squared error given by Equation 5 and $E_{ED} = \sum_{(i,j) \in E_S} |d'_{ij} - d_{ij}|/d_{ij}$ with d_{ij} and d'_{ij} are the lengths of the edge e_{ij} respectively before and after the deformation. We test the convergence criteria of an iteration by evaluating the change of the general error compared to the previous iteration. The algorithm converges rapidly because of a good starting point is given by the previous step in the motion. The overall algorithm is given in algorithm 1.

Algorithm 1: Control mesh deformation with implicit skeleton constraints

Data:
$\boldsymbol{X}$: Vector of original vertices of the original mesh
$\boldsymbol{X}'$: Vector of positions of the original mesh after editing
$\boldsymbol{V}$: Vector of original vertices of the control mesh
$\boldsymbol{V}'$: Vector of positions of the control mesh after editing
$\boldsymbol{U}$: A set of position constraints
$\mathcal{V}$: the original control mesh

$ComputeGreenCoordinates(\boldsymbol{X}, \mathcal{V})$
//Laplacian and positionnal constraints (offline):
$\boldsymbol{V}' = \text{argmin}_{\boldsymbol{V}} E_l(\boldsymbol{V}) + E_p(\boldsymbol{V})$
while *playing animation* **do**
  while *not convergence* **do**
    form Γ from $\text{argmin}_{\boldsymbol{V}} E_s(\boldsymbol{V}, \mathcal{V})$ (equation 9), preconditionned with $\boldsymbol{V}'$;
    $\boldsymbol{V}' = \text{argmin}_{\boldsymbol{V}} \begin{pmatrix} w_l \boldsymbol{L} \\ w_u \boldsymbol{I}_u \\ w_d \boldsymbol{D} \end{pmatrix} \boldsymbol{V} = \begin{pmatrix} \boldsymbol{\Delta}^A \\ w_u \boldsymbol{U} \\ w_d \boldsymbol{\Gamma} \end{pmatrix}$
  $\boldsymbol{X}' = GreenCoordinates(\boldsymbol{V}', \mathcal{V})$

V. Implementation and results

This Section shows the practical advantages of using our method to animate an articulated mesh driven by different sets of low dimensional input. Our C++ implementation uses the Cholmod sparse Cholesky solver [Chen et al.(2008)Chen, Davis, Hager and Rajamanickam] on a Core2 Duo with 2.7GHz and 3GB RAM on Windows XP Pro. The Cholmod

library is able to solve large sparse linear system using a Cholesky factorization. This resolution method is fast enough for real time animation or interactive mesh editing.

Model (percentage of mesh control information form original mesh)	Figure 6 (b) (0.24%)	Figure 6 (b) (15.81%)	Figure 7 (b) (23.57%)	Figure 9 (a) (7.32%)
Original mesh vertex and triangle count	172914 345944	172914 345944	8256 15198	5023 10042
Control mesh vertex and triangle count	423 5092	27350 36814	1946 26364	368 4728
Time to build the Cholesky syst. (offline)(sec.)	0.101	264.432	1.836	0.102
Time to update control mesh system (sec.)	0.012	0.986	0.068	0.012
Time to update GC system (sec.)	0.063	0.236	0.014	0.002
RRMS_E	0.04	0.027	0.101	0.087
Original mesh RE_V	0.042	0.029	0.012	0.061
Skeleton edges length error: ED_E	0.19	0.011	0.019	0.016

TABLE I

TABLE OF TIMES AND ERRORS FOR DIFFERENT ORIGINAL AND CONTROL MESH SIZES AFTER 50 ITERATIONS. RRMS_E IS THE RELATIVE ROOT MEAN SQUARED ERRORS (EQUATION 5), RE_V MEANS THE RELATIVE ERROR OF VOLUME MAGNITUDES($|Volume_{Original} - Volume_{current}|/Volume_{Original}$).

We show through various examples that our method is efficient to produce high quality animation in real time. These experiments require a preliminary step of creating a skeleton and a cage geometry corresponding to the mesh. These two elements are achieved rather fast (about 10-15 minutes) by a student.

In the first example, we apply the method to the animation of a mesh composed of 8256 vertex and 15198 triangles, driven by only 36 markers: eight markers for each hand, two for each foot, four for the head and twelve for the rest of the body). The markers are located inside and outside the target mesh according to the difference of morphology between the captured model and the target geometry. As shown in figure 7 and in the accompanying video, the movement of the markers are successfully tracked and the resulting deformations look very natural. For a quantitative evaluation of our algorithm, table I shows that the constraints are well preserved and figure 8 shows the good convergence of our alternating descent approach which minimizes simultaneously the deformation energies while respecting the skeleton properties.

In the second example, we animate two-hand meshes from a sign language motion. Sign language animation of hands is very critical: it requires high precision, especially for hand configurations. Small changes in some postures or trajectories might indeed modify the meaning of the produced signs [Gibet et al.(2011)Gibet, Courty, Duarte and Naour]. Each hand is represented by a geometry of a high resolution (5023 vertices for 10042 triangles) and driven by few control points (eight markers for each hand). The mesh model was acquired from the Aim@shape repository and symmetrized for the demonstration purpose. Our original hand motion is represented by some low dimensional and low quality signals extracted from markers trajectories. This poorness of the captured motion is due to several factors. First, occlusions occur frequently in sign language statements, and the low number of markers does not allow us to reconstruct complete trajectories. This takes the form of missing samples or markers inversion in some markers trajectories. Such motion data requires heavy post-processing which also introduces some errors. When observing the animations in the accompanying video, we thus notice at some points jerky motion, or inter-penetration effects introduced by motion retargeting. Skeleton errors or rigging artifacts due to approximate weights are also visible for some specific hand configurations.

Instead, as can be seen in figure 9 and in the accompanying video, our method animates successfully both hands. The meshes are subject to large deformations but as presented in table I, they preserve both geometric details and skeleton properties. Furthermore, the animations look very realistic. Finally, the accompanying video shows a comparison between the original motion played after a traditional motion capture post-processing/mesh rigging by a skilled CG artist and the motion produced by our method. The results are very convincing and demonstrate the interest of such an animation pipeline for this kind of animation task.

VI. CONCLUSION

The challenge proposed by this paper was to reconsider the traditional animation pipeline of character driven by the control of a skeleton. Our main objective was both to improve the quality of the animation by reducing a number of post-processing tasks and control the mesh with a low signal dimension. In this context, we presented an animation pipeline driven by a small number of constraints based on an iterative system involving a volumetric mesh with a skeleton structure. The originality of the method lies in this strong coupling that allows us to incorporate implicit skeleton constraints within the deformation process. The concept of rigidity is improved by several aspects of the approach. By extending the *as-rigid-as-possible* paradigm to volumetric mesh, we prevent unintuitive shape deformations and preserve the global volume of the object. When building the volumetric mesh, we include an internal structure resulting from a spatial discretization of the skeleton, which plays a central role in the tetrahedrization process. Furthermore, the rigidity properties of the system are enhanced by taking into account skeleton properties such as bone lengths and joint limits. As the skeleton constraints are not linear, our descent method alternates between a linear Laplacian deformation of the volumetric mesh and the computation of postures enforcing these constraints. In order to produce natural postures, we believe that the inclusion of such constraints is of paramount importance for the animation of articulated bodies, as illustrated for hand animations. Finally, our animation pipeline is stable and robust, and allows us

Fig. 7. Animation of an arbitrary mesh. At left, first figure: the bind pose geometry and the manual positioning of markers. Next: five frame of our output system.

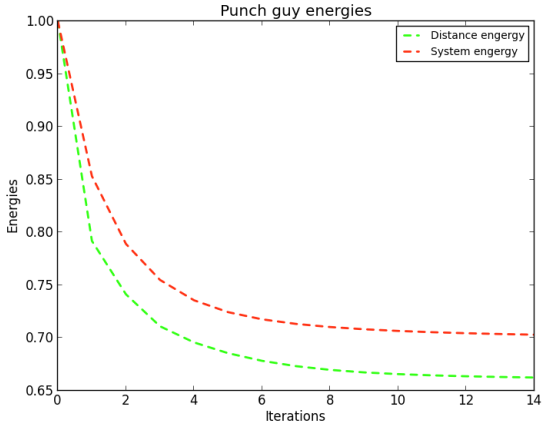

Fig. 8. Example of convergence of the alternating descent algorithm for an interval of ten frames during the animation. The distance and system energies are the normalized values described in the table I

to produce in real time high-quality mesh animations from a wide variety of motion capture inputs, even when the set of input points is strongly reduced. As a major advantage, it allows to bypass the tedious rigging phase, but also some of the traditional post-processing of motion capture process.

The thorough study of the coupling mesh / skeleton opens up promising perspectives for the animation of articulated characters. One possible extension might be to consider other types of constraints related to biomechanically plausible modeling of the articulations. The proposed alternate descent method might be challenged when strongly non-linear constraints are involved, which might require more sophisticated optimization schemes. Also, other types of low dimensional input constraints can be tested in our framework, such as using accelerometers or vision inputs from mono or multi-camera, which would alleviate the need of costly infrared motion capture equipments.

REFERENCES

[Krayevoy and Sheffer(2005)] Krayevoy, V., Sheffer, A.. Boneless motion reconstruction. In: ACM SIGGRAPH 2005 Sketches. SIGGRAPH '05; New York, NY, USA: ACM; 2005, p. 0–0.

[Stoll et al.(2007)Stoll, de Aguiar, Theobalt and Seidel] Stoll, C., de Aguiar, E., Theobalt, C., Seidel, H.P.. A volumetric approach to interactive shape editing. Research Report MPI-I-2007-4-004; Max-Planck-Institut für Informatik; Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany; 2007.

[Borosán et al.(2010)Borosán, Howard, Zhang and Nealen] Borosán, P., Howard, R., Zhang, S., Nealen, A.. Hybrid mesh editing. In: Proceedings of Eurographics; vol. 2010. 2010, p. 0–0.

[Lipman et al.(2008)Lipman, Levin and Cohen-Or] Lipman, Y., Levin, D., Cohen-Or, D.. Green coordinates. ACM Transactions on Graphics (TOG) 2008;27(3):78:1–78:10.

[Lewis et al.(2000)Lewis, Cordner and Fong] Lewis, J.P., Cordner, M., Fong, N.. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques. SIGGRAPH '00; New York, NY, USA: ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5; 2000, p. 165–172.

[Kavan et al.(2008)Kavan, Collins, Žára and O'Sullivan] Kavan, L., Collins, S., Žára, J., O'Sullivan, C.. Geometric skinning with approximate dual quaternion blending. ACM Transactions on Graphics (TOG) 2008;27(4):105:1–105:23.

[Vaillant et al.(2013)Vaillant, Barthe, Guennebaud, Cani, Rohmer, Wyvill et al.] Vaillant, R., Barthe, L., Guennebaud, G., Cani, M.P., Rohmer, D., Wyvill, B., et al. Implicit skinning: Real-time skin deformation with contact modeling. ACM Transactions on Graphics (TOG) 2013;32(4):125:1–125:12.

[Park and Hodgins(2006)] Park, S.I., Hodgins, J.K.. Capturing and animating skin deformation in human motion. In: ACM Transactions on Graphics (TOG); vol. 25. ACM; 2006, p. 881–889.

[Sumner et al.(2005)Sumner, Zwicker, Gotsman and Popović] Sumner, R.W., Zwicker, M., Gotsman, C., Popović, J.. Mesh-based inverse kinematics. In: ACM Transactions on Graphics (TOG); vol. 24. ACM; 2005, p. 488–495.

[Wheatland et al.(2013)Wheatland, Jörg and Zordan] Wheatland, N., Jörg, S., Zordan, V.. Automatic hand-over animation using principle component analysis. In: Proceedings of Motion on Games. MIG '13; New York, NY, USA: ACM. ISBN 978-1-4503-2546-2; 2013, p. 175–202.

[Zordan and Van Der Horst(2003)] Zordan, V.B., Van Der Horst, N.C.. Mapping optical motion capture data to skeletal motion using a physical model. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation. SCA '03; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 1-58113-659-5; 2003, p. 245–250.

[Sorkine et al.(2004)Sorkine, Cohen-Or, Lipman, Alexa, Rössl and Seidel] Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., Seidel, H.P.. Laplacian surface editing. In: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing. SGP '04; New York, NY, USA: ACM. ISBN 3-905673-13-4; 2004, p. 175–184.

[Alexa(2003)] Alexa, M.. Differential coordinates for local mesh mor-

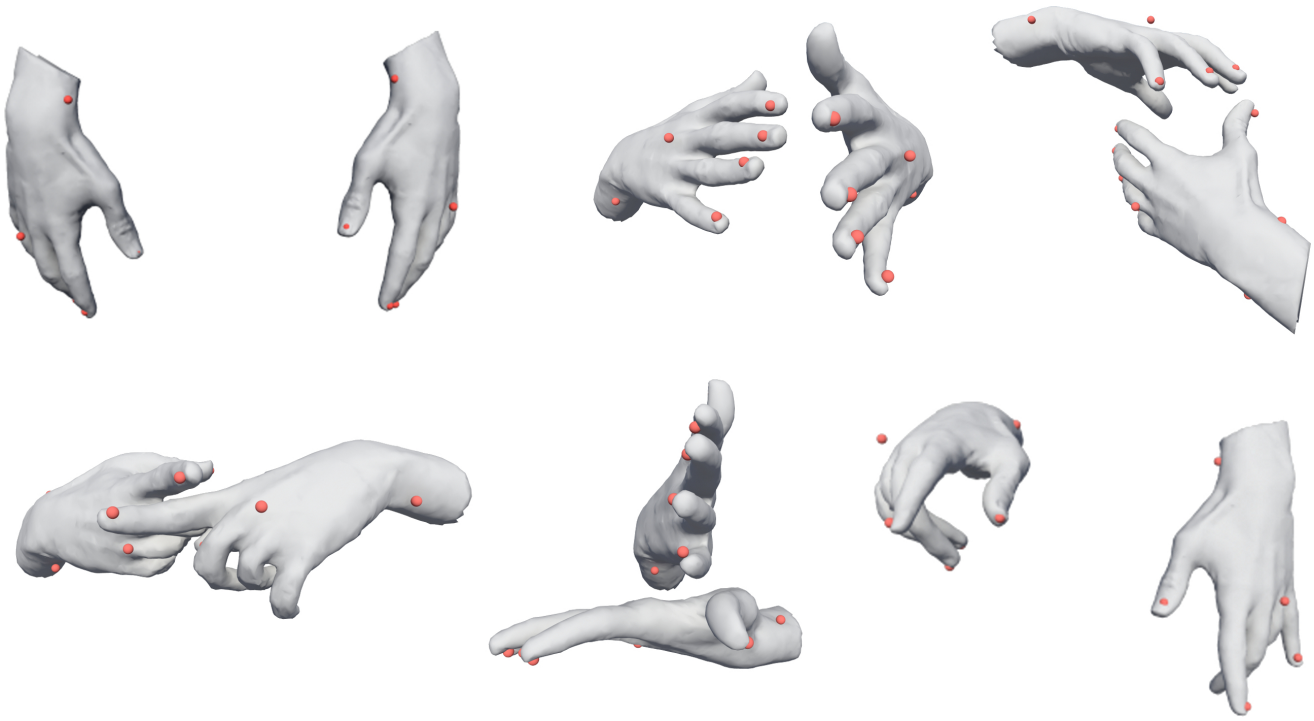

Fig. 9. Sign language animation of hands. Six excerpts from the original animation. The low dimensional markers set used to animate the model are depicted in red.

phing and deformation. The Visual Computer 2003;19:105–114. 10.1007/s00371-002-0180-0106.

[Lipman et al.(2004)Lipman, Sorkine, Cohen-or, Levin, Rssl and peter Seidel] Lipman, Y., Sorkine, O., Cohen-or, D., Levin, D., Rssl, C., peter Seidel, H.. Differential coordinates for interactive mesh editing. In: In Proceedings of Shape Modeling International. Society Press; 2004, p. 181–190.

[Sorkine and Alexa(2007a)] Sorkine, O., Alexa, M.. As-rigid-as-possible surface modeling. In: Proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing. 2007a, p. 109–116.

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

[Huang et al.(2006)Huang, Shi, Liu, Zhou, Wei, Teng et al.] Huang, J., Shi, X., Liu, X., Zhou, K., Wei, L.Y., Teng, S.H., et al. Subspace gradient domain mesh deformation. ACM SIGGRAPH 2006 2006;:11–26.

[Zhao et al.(2009)Zhao, Liu, Peng and Bao] Zhao, Y., Liu, X.G., Peng, Q.S., Bao, H.J.. Rigidity constraints for large mesh deformation. Journal of Computer Science and Technology 2009;24(1):47–55.

[Jacobson et al.(2011)Jacobson, Baran, Popović and Sorkine] Jacobson, A., Baran, I., Popović, J., Sorkine, O.. Bounded biharmonic weights for real-time deformation. ACM Transactions on Graphics (TOG) 2011;30(4):78–78.

[Zhao and Liu(2012)] Zhao, Y., Liu, J.. Volumetric subspace mesh deformation with structure preservation. Computer Animation and Virtual Worlds 2012;23(5):519–532.

[Shi et al.(2007)Shi, Zhou, Tong, Desbrun, Bao and Guo] Shi, X., Zhou, K., Tong, Y., Desbrun, M., Bao, H., Guo, B.. Mesh puppetry: Cascading optimization of mesh deformation with inverse kinematics. ACM Trans Graph 2007;26(3). URL: http://doi.acm.org/10.1145/1276377.1276479. doi:10.1145/1276377.1276479.

[Zhang et al.(2010)Zhang, Nealen and Metaxas] Zhang, S., Nealen, A., Metaxas, D.. Skeleton based as-rigid-as-possible volume modeling. Eurographics Association, Norrkoping, Sweden 2010;:21–24.

[Faraj et al.(2012)Faraj, Thiery, Bloch, Varsier, Wiart and Boubekeur] Faraj, N., Thiery, J.M., Bloch, I., Varsier, N., Wiart, J., Boubekeur, T.. Robust and scalable interactive freeform modeling of high definition medical images. Mesh Processing in Medical Image Analysis 2012 2012;:1–11.

[Chao et al.(2010)Chao, Pinkall, Sanan and Schröder] Chao, I., Pinkall, U., Sanan, P., Schröder, P.. A simple geometric model for elastic deformations. ACM Transactions on Graphics (TOG) 2010;29(4):38–38.

[McAdams et al.(2011)McAdams, Zhu, Selle, Empey, Tamstorf, Teran et al.] McAdams, A., Zhu, Y., Selle, A., Empey, M., Tamstorf, R., Teran, J., et al. Efficient elasticity for character skinning with contact and collisions. ACM Transactions on Graphics (TOG) 2011;30(4):1–9.

[Kavan and Sorkine(2012)] Kavan, L., Sorkine, O.. Elasticity-inspired deformers for character articulation. ACM Transactions on Graphics (TOG) 2012;31(6):196–196.

[Au et al.(2008)Au, Tai, Chu, Cohen-Or and Lee] Au, O.K.C., Tai, C.L., Chu, H.K., Cohen-Or, D., Lee, T.Y.. Skeleton extraction by mesh contraction. ACM Transactions on Graphics (TOG) 2008;27(3).

[Bridson et al.(2005)Bridson, Teran, Molino and Fedkiw] Bridson, R., Teran, J., Molino, N., Fedkiw, R.. Adaptive physics based tetrahedral mesh generation using level sets. Eng Comput (Lond 2005;21:2–18.

[Cutler et al.(2004)Cutler, Dorsey and McMillan] Cutler, B., Dorsey, J., McMillan, L.. Simplification and improvement of tetrahedral models for simulation. In: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing. SGP '04; New York, NY, USA: ACM. ISBN 3-905673-13-4; 2004, p. 93–102.

[Shewchuk(1998)] Shewchuk, J.R.. Tetrahedral mesh generation by delaunay refinement. In: Proceedings of the fourteenth annual symposium on Computational geometry. SCG '98; New York, NY, USA: ACM. ISBN 0-89791-973-4; 1998, p. 86–95.

[Miller et al.(1996)Miller, Talmor, Teng, Walkington and Wang] Miller, G.L., Talmor, D., Teng, S.H., Walkington, N., Wang, H.. Control volume meshes using sphere packing: Generation, refinement and coarsening. In: In Fifth International Meshing Roundtable. 1996, p. 47–61.

[Si(2006)] Si, H.. TetGen User Manual; 2006.

[Pinkall et al.(1993)Pinkall, Juni and Polthier] Pinkall, U., Juni, S.D., Polthier, K.. Computing discrete minimal surfaces and their conjugates. Experimental Mathematics 1993;2:15–36.

[Meyer et al.(2002a)Meyer, Desbrun, Schröder and Barr] Meyer, M., Des-

brun, M., Schröder, P., Barr, A.H.. Discrete differential-geometry operators for triangulated 2-manifolds. 2002a.

[Sorkine and Alexa(2007b)] Sorkine, O., Alexa, M.. As-rigid-as-possible surface modeling. In: Proceedings of the fifth Eurographics symposium on Geometry processing. SGP '07; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 978-3-905673-46-3; 2007b, p. 109–116.

[Meyer et al.(2003)Meyer, Desbrun, Schröder and Barr] Meyer, M., Desbrun, M., Schröder, P., Barr, A.. Discrete differential-geometry operators for triangulated 2-manifolds. In: Hege, H.C., Polthier, K., editors. Visualization and Mathematics III. Mathematics and Visualization; Springer Berlin Heidelberg. ISBN 978-3-642-05682-6; 2003, p. 35–57. doi:10.1007/978-3-662-05105-4_2.

[Aristidou and Lasenby(2011)] Aristidou, A., Lasenby, J.. Fabrik: A fast, iterative solver for the inverse kinematics problem. Graph Models 2011;73(5):243–260.

[Blow(2002)] Blow, J.. Inverse kinematics with quaternion joint limits. Tech. Rep.; 2002. [Online: http://number-none.com/product/IK with Quaternion Joint Limits/].

[Meyer et al.(2002b)Meyer, Barr, Lee and Desbrun] Meyer, M., Barr, A., Lee, H., Desbrun, M.. Generalized barycentric coordinates on irregular polygons. J Graph Tools 2002b;7(1):13–22.

[Xian et al.(2009)Xian, Lin and Gao] Xian, C., Lin, H., Gao, S.. Automatic generation of coarse bounding cages from dense meshes. In: Shape Modeling and Applications, 2009. SMI 2009. IEEE International Conference on. IEEE; 2009, p. 21–27.

[Xian et al.(2012)Xian, Lin and Gao] Xian, C., Lin, H., Gao, S.. Automatic cage generation by improved obbs for mesh deformation. The Visual Computer 2012;28(1):21–33. doi:10.1007/s00371-011-0595-6.

[Chen et al.(2008)Chen, Davis, Hager and Rajamanickam] Chen, Y., Davis, T.A., Hager, W.W., Rajamanickam, S.. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. ACM Transactions on Mathematical Software 2008;35(3):22:1–22:14.

[Gibet et al.(2011)Gibet, Courty, Duarte and Naour] Gibet, S., Courty, N., Duarte, K., Naour, T.L.. The signcom system for data-driven animation of interactive virtual signers: Methodology and evaluation. ACM Transactions on Interactive Intelligent Systems 2011;1(1):6:1–6:23.