Element-Free Elastic Models for Volume Fitting and Capture

Jaeil Choi Andrzej Szymczak Greg Turk Irfan Essa
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
{jerry,andrzej,turk,irfan}@cc.gatech.edu

Abstract

We present a new method of fitting an element-free volumetric model to a sequence of deforming surfaces of a moving object. Given a sequence of visual hulls, we iteratively fit an element-free elastic model to the visual hull in order to extract the optimal pose of the captured volume. The fitting of the volumetric model is acheived by minimizing a combination of elastic potential energy, a surface distance measure, and a self-intersection penalty for each frame. A unique aspect of our work is that the model is mesh free – since the model is represented as a point cloud, it is easy to construct, manipulate and update the model as needed. Additionally, linear elasicity with rotation compensation makes it possible to handle local deformations and large rotations of body parts much more efficiently than other volume fitting approaches. Our experimental results for volume fitting and capture in a multi-view camera setting demonstrate the robustness of element-free elastic models against noise and self-occlusions.

1. Introduction

Silhouettes provide rich information about the shape of an objects. Silhouettes from multiple viewpoints can be combined to reconstruct a 3D shape and can also capture the motion of the shape over time. There is a rich body of work in this area, starting from the visual hull reconstruction from different viewpoints [16, 19]. Many extensions have since been proposed to aid in this reconstruction which include utilizing color information [17, 26], using sequence of silhouettes from rigid motion [7, 27], relying on a skeletal structure [4, 9] and reconstructing shape and motion from silhouettes from non-rigid motions of articulated body [6, 15]. However, accurate, efficient and robust reconstruction of moving 3D shapes still remains a difficult problem. One reason for this is that volume reconstruction methods are limited in their ability to deal with large motions, and most of the existing approaches require a pre-defined mesh representation.

We begin by calculating silhouettes from the videos from several calibrated camera. We present a new method of fitting a volumetric model to a sequence of deforming surfaces that are created from these silhouettes. Our approach relies on using an *element-free elastic model* with rotation compensation. Our iterative fitting approach makes it possible to capture local deformations on the surface and large rotations of body parts which are common in the motions of articulated body. This point-based model provides a general framework for volume fitting and capture that can be applied to any rigid, non-rigid, or articulated object without any prior knowledge. Our contributions include

- introduction of an element-free model for volume fitting, which provides a mesh-free and data-driven fitting model
- a new formulation of linear elasticity with rotation compensation for the element-free model
- a new iterative fitting approach that minimizes a combination of an elastic potential energy, a surface distance measure and a self-intersection penalty.

Our experimental results demonstrate the robustness of our element-free elastic model against noise and self-occlusions that are common in the visual hulls of articulated objects.

1.1. Related Work

Several researchers have extended the Shape-From-Silhouette (SFS) approach to temporal volume sequences of an articulated body such as the motions of a human. Color information has been successfully incorporated with SFS to build temporal correspondences and capture the motion.

Cheung *et al*. [6] uses visual hull alignment technique and kinetic information to reconstruct the shape and motion from the volume sequence of a human body. Their "temporal SFS" utilizes the consistency of color, assuming a Lambertian surface. The acquisition of kinematic information still remains a challenge since their process is more about *measuring* than automatic *estimation*, requiring prior knowledge and cooperation of the subject.

Chu *et al*. [9] and Brostow *et al*. [4] use a skeletal representation of each frame in the volume sequence to build an temporal correspondence and capture the motion of an articulated body. Even though 1D skeletons simplify the creation of a mapping from one frame to another, most detail geometric information of the silhouettes is lost in the skeletal representation. Furthermore, there are objects that cannot be described by a 1D skeleton because their 'principal components' are 2D manifolds.

Kehl *et al*. [15] use a template model of human skin and skeletons to track human motion. Their fitting approach is based on gradient descent for an objective function defined as the sum of distances between model and the observed surface. Stochastic sampling is used to speed up the computation. Since their goal is to find the optimal configuration of the skeleton of a generalized model, all the subject-specific features and geometric details on the surface are lost.

To the best of our knowledge, all the previous work on the reconstruction of 3D shape and motion from a volume sequence used segmentation and/or skeletal information of the model to deal with large rotations of body parts. This means that they assume prior knowledge of a skeletal structure or they have to estimate it somehow. Furthermore, purely skeleton-based motion estimation loses details such as the deformation of the skin and muscles. Our approach to attack this problem is to use a volumetric, deformable and energy minimizing model.

Energy minimizing models [5, 11, 14] and deformable models [20, 22, 23, 24] have a long history in computer vision. These models are based on the minimization of a combination of internal potential energy and external data error, with additional geometric/topological constraints. Our approach using an element-free elastic model is based on the same principle, but significantly differs in that it allows large deformations and rotational movements of body parts while preserving their volume as much as possible.

We have studied the issue of rotation compensation and iterative fitting based on the minimization of elastic potential energy and surface distance in our previous work [8]. However, in that effort, elasticity was simulated by the Finite Element Method (FEM). Use of FEM was restrictive as the generation and maintainance of the high quality mesh is a big challenge, especially for a highly deforming objects. The element-free elastic models proposed in this paper make the generation and modification of the model relatively easy. We also extend the iterative fitting approach to include a self-intersection penalty, which can be identified and evaluated efficiently using spatial hashing.

2. Overview

In this paper, we concentrate on element-free elastic models and their use for finding the optimal pose that minimizes a combination of elastic potential energy and geometric errors. As the input, we take segmented videos of a moving object captured using multi-view calibrated cameras. We construct a sequence of visual hulls from the segmented video frames, and one of the frames is selected to build the initial model that also serves as the rest-state. Then, the initial model is fit to the visual hull by minimizing a quadratic objective function frame by frame. The same fitting approach can be applied to a sequence of 2D profiles.

The basic representation of the element-free elastic model is a point cloud. Additionally, we need boundary representation of the model and local neighborhood information of the points in the model (section 3.1). The boundary information is needed for fitting (section 4), and the neighborhood information is used to compute the shape functions that lead to the displacement field inside the model (section 3.2) and elastic potential energy (section 3.3). We use linear elasticity for its simplicity, and the instability of linear elasticity for large rotational movement and deformation is resolved by compensating for local rotations (section 3.4).

Our iterative fitting procedure finds the optimal pose of the model by minimizing the weighted sum of the elastic potential energy, a surface distance measure (section 4.1) and the self-intersection penalty (section 4.2). Each iteration of the optimization is reduced to a linear system that is solved efficiently by the conjugate gradient method using a small number of iterations (section 4.3).

3. Element-Free Elastic Model

Element-free methods have been successfully used in engineering fields such as structural mechanics for the analysis of deformation and strain/stress tensors. A survey on mesh-free methods can be found in Fries and Matthies [13]. Even though element-free approaches are generally a constant factor slower than the Finite Element Methods, they have an advantage because they do not require high-quality finite element meshes whose construction and modification is difficult, especially for an animated sequence of a deforming object.

3.1. Representation of the Model

Each point in the model can be thought of as a sample of the object that represents the physical properties of the object in its Voronoi region. The points are organized in a data structure that allows us to find neighbors efficiently. In our implementation, we use the spatial hashing technique

[25] to accelerate such queries as well as to detect self-intersections. The domain of influence (the size of neighborhood) of a point may vary to allow adaptive sampling and to make sure that each point has enough neighbors to estimate the displacement field inside the model in order to compute the elastic potential energy. We assume the neighbors do not change during deformation, and keep the list of neighbors for each point in the rest-state model.

While the internal elastic force is computed from the displacements of interior points in the model, the external fitting force results from the input data acts on the boundary surface of the model. In our fitting approach, the representation of the boundary must allow: (a) quick tests for the points on the boundary and (b) efficient queries of the surface normal. We use a triangular mesh for this purpose, which can be generated automatically using alpha shapes [10] from all the sample points in the model.

3.2. MLS Approximation of Displacement Field

One of the driving forces in our fitting approach is the elastic potential energy, which is computed from the displacement field inside the model. The continuous displacement field is obtained from the displacement data available at the points of the model (let 'nodes' denote those discrete samples [1]) using the Moving Least Squares (MLS) approximation. In this section, we treat a displacement vector as a scalar value, and briefly summarize the form of the MLS approximation.

To obtain a continuous scalar field $u(\mathbf{x})$ from the discrete sample u_i at each node, we use the MLS method, which was originally developed for surface approximation [18] and applied later in Element-Free Galerkin (EFG) methods [3]. In this method, $u(\mathbf{x})$ is approximated by the result of a scalar product of $\mathbf{\Phi}(\mathbf{x})$, the vector of shape function, and $\mathbf{u}$, the vector of scalar values of all the nodes. The function $\mathbf{\Phi}(\mathbf{x})$ is computed by minimizing a discrete L_2 norm of sample values over the coefficients of a given basis.

Let $\mathbf{p}(\mathbf{x})$ be a linear basis for the approximation, $\mathbf{p}(\mathbf{x}) = [1, x, y, z]^T$, where $\mathbf{x} = [x, y, z]$. The scalar field $u(\mathbf{x})$ at an arbitrary 3D point $\mathbf{x}$ is approximated by

$$u(\mathbf{x}) \simeq u^h(\mathbf{x}) = \sum_k p_k(\mathbf{x}) a_k(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\,\mathbf{a}(\mathbf{x}). \quad (1)$$

The coefficients $\mathbf{a}(\mathbf{x})$ are functions of $\mathbf{x}$ and are obtained by minimizing a weighted, discrete L_2 norm given by

$$J = \sum_i w(\mathbf{x} - \mathbf{x}_i)[\mathbf{p}^T(\mathbf{x}_i)\mathbf{a}(\mathbf{x}) - u_i]^2, \quad (2)$$

[1]Each point in the model has one corresponding node, but the displacement at a point is not exactly equal to the displacement of the corresponding node because $\mathbf{\Phi}$ does not satisfy Kronecker delta property [3]. Nodes can be thought of as control points as in splines.

where i enumerates all the nodes in the neighborhood (the domain of influence) of $\mathbf{x}$, where the given weight function $w(\mathbf{x} - \mathbf{x}_i)$ is nonzero.

The minimization of J with respect to $\mathbf{a}(\mathbf{x})$ leads to

$$\mathbf{a}(\mathbf{x}) = \mathbf{A}^{-1}(\mathbf{x})\,\mathbf{B}(\mathbf{x})\ \mathbf{u}, \quad (3)$$

where

$$\begin{aligned} \mathbf{A}(\mathbf{x}) &= \Sigma_i [w(\mathbf{x} - \mathbf{x}_i)\,\mathbf{p}(\mathbf{x}_i)\,\mathbf{p}^T(\mathbf{x}_i)], \\ \mathbf{B}(\mathbf{x}) &= [w(\mathbf{x} - \mathbf{x}_1)\mathbf{p}(\mathbf{x}_1)\ \ldots\ w(\mathbf{x} - \mathbf{x}_n)\mathbf{p}(\mathbf{x}_n)], \\ \mathbf{u} &= [u_1\ u_2\ \ldots\ u_n]^T, \end{aligned}$$

and n is the number of nodes in the model.

Now we have

$$u^h(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\,\mathbf{A}^{-1}(\mathbf{x})\,\mathbf{B}(\mathbf{x})\ \mathbf{u} = \mathbf{\Phi}^T(\mathbf{x})\,\mathbf{u}, \quad (4)$$

where $\mathbf{\Phi}^T(\mathbf{x})$ is the shape function of the vector of nodal values $\mathbf{u}$.

For the weight function $w(\mathbf{x} - \mathbf{x}_i)$, we use the tensor product cubic spline weight [12]

$$w(\mathbf{x} - \mathbf{x}_i) = w(r_x)\ w(r_y)\ w(r_z), \quad (5)$$

where $r_x = ||x - x_i||/d_{dmi}$, with d_{dmi} denoting the size of the domain of influence, and

$$w(r) = \begin{cases} \frac{2}{3} - 4r^2 + 4r^3 & \text{for } r \le \frac{1}{2} \\ \frac{4}{3} - 4r + 4r^2 + \frac{4}{3}r^3 & \text{for } \frac{1}{2} < r \le 1 \\ 0 & \text{for } r > 1 \end{cases}. \quad (6)$$

For the computation of the shape function $\mathbf{\Phi}(\mathbf{x})$, u_i, the displacement value of node i, can be treated as a single scalar quantity, since the shape function $\mathbf{\Phi}(\mathbf{x})$ acts on all the axes equally. For the rest of the paper, however, u_i is actually a 3×1 displacement vector. Hence the dimension of the global displacement vector $\mathbf{u}$ is $3n \times 1$.

3.3. Linear Elasticity

The displacement field provided by the MLS approximation can be coupled with elasticity simulation of any order. In our approach, we use linear elasticity because of its simplicity for the optimization.

Given the displacement field as a function of nodal values $\mathbf{u}$ (Eq. 4), now we can calculate linear strain tensor ϵ and stress tensor σ. With partial differential operator $\mathbf{L}$ and shape function matrix $\mathbf{H}$ defined as

$$\mathbf{L}^T = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial z} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \text{ and}$$

$$\mathbf{H} = \begin{bmatrix} \Phi_1 & 0 & 0 & \ldots & \Phi_n & 0 & 0 \\ 0 & \Phi_1 & 0 & \ldots & 0 & \Phi_n & 0 \\ 0 & 0 & \Phi_1 & \ldots & 0 & 0 & \Phi_n \end{bmatrix},$$

we get the strain tensor

$$\epsilon = \mathbf{L}\, u^h = \mathbf{LHu} \tag{7}$$

and the stress tensor

$$\sigma = \mathbf{C}\epsilon = \mathbf{CLHu}, \tag{8}$$

where $\mathbf{u}$ is the nodal displacement vector of size $3n \times 1$ and $\mathbf{C}$ is the 6×6 strain-stress matrix defined by two scalar parameters - Young's modulus e and Poisson ratio ν - as

$$\mathbf{C} = \frac{e}{(1+\nu)(1-2\nu)} \begin{bmatrix} a & b & b & 0 & 0 & 0 \\ b & a & b & 0 & 0 & 0 \\ b & b & a & 0 & 0 & 0 \\ 0 & 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & 0 & c & 0 \\ 0 & 0 & 0 & 0 & 0 & c \end{bmatrix}, \tag{9}$$

where $a = 1 - \nu$, $b = \nu$ and $c = 0.5 - \nu$. Further details can be found in textbooks, for example, [1].

Now we can integrate over the entire domain of the model to get the elastic potential energy E with respect to nodal displacements $\mathbf{u}$ as

$$E = \frac{1}{2}\mathbf{u}^T \int_\Omega \mathbf{L}^T\mathbf{H}^T\mathbf{C}\,\mathbf{L}\,\mathbf{H}\, d\Omega\, \mathbf{u} = \frac{1}{2}\mathbf{u}^T\mathbf{K}\,\mathbf{u}, \tag{10}$$

where $\mathbf{K}$ is the global stiffness matrix. $\mathbf{K}$ is sparse and symmetric, and can be built by integrating over the local stiffness matrices. For the integration, we take the nodal integration approach which uses the weighted sum of the local stiffness matrices at the node, instead of gaussian quadrature over the domain Ω (the interior of the model) frequently used in EFG methods. A detailed analysis of the nodal integration approach can be found in Beissel and Belytschko [2].

3.4. Rotation Compensation

The drawback of linear elasticity is that linearized strain/stress tensors are not invariant under rotation, which is critical for large deformations and rotational movements. To address this problem, we estimate the rotational part of the transformation of the neighborhoods from the rest-state to the current configuration, apply the rotation to the rest-state neighborhood of the point, and then calculate linear elasticity based on the new rotated neighborhood. This process is illustrated in Figure 1.

To estimate the rotational part of the local transformation, let $\mathbf{a}_i = \mathbf{x}_{p_i} - \mathbf{x}_p$ for each point p_i in the neighborhood of p in the rest-state model and let $\mathbf{b}_i$ be the same for the current model. We need to find the optimal linear transformation T that minimizes $\sum_i m_i ||T\mathbf{a}_i - \mathbf{b}_i||^2$, where m_i is the weight for the neighbor i (a gaussian falloff). Setting

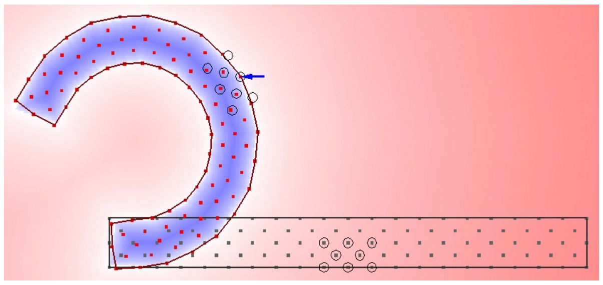


Figure 1. Displacement at a point (indicated by a blue arrow) in the deformed model (bent bar) is calculated based on the rotated version of the neighboring nodes (black circles) of the rest-state model (straight bar). The blue/red background illustrates the signed distance field.

its derivatives with respect to all coefficient of T to zero leads to the optimal transformaion [21]

$$T = (\sum_i m_i \mathbf{a}_i \mathbf{b}_i^T)(\sum_i m_i \mathbf{b}_i \mathbf{b}_i^T) = T_{\mathbf{ab}} T_{\mathbf{bb}}. \tag{11}$$

The second term is symmetric and contains only scaling but no rotation. Therefore, the pure rotation can be computed using singular value decomposition of $T_{\mathbf{ab}}$, $T_{\mathbf{ab}} = U\,S\,V^T$, as $U\,V^T$.

Inevitably, the new rotated rest-state neighborhoods of different points do not agree with each other – the point p' in the new rotated neighborhood N_p of point p is not at the same location in general as p' in N_q of the point q near p. Consequently, the displacement values of all the nodes $\mathbf{u} = \mathbf{x} - \mathbf{x}_r$ (with $\mathbf{x}$ and $\mathbf{x}_r$ denoting the nodal coordinate vectors of the current and the rest-state models, respectively) do not form a vector as a single variable, because values in $\mathbf{x}_r^{N_p}$ (rotated neighbor locations for node p) and $\mathbf{x}_r^{N_q}$ (rotated neighbor locations for node q) do not match. We resolve this problem by modifying the equation of elastic potential energy (Eq. 10) as

$$E = \frac{1}{2}\,(\mathbf{x}^T\mathbf{K}\,\mathbf{x} - 2\sum_{p \in M}(\mathbf{x}_r^{N_p\,T}\mathbf{k}^{N_p}\mathbf{x}^{N_p}) + \sum_{p \in M}(\mathbf{x}_r^{N_p\,T}\mathbf{k}^{N_p}\mathbf{x}_r^{N_p})), \tag{12}$$

where $\mathbf{k}^{N_p}$ is the local stiffness matrix at the neighborhood N_p. Strain/stress tensors are also calculated in the same manner.

4. Iterative Fitting

Our iterative fitting procedure starts by building the rest-state model $M_{\mathcal{R}}$ using one of the frames selected by user. Given a set of silhouettes of segmented videos from calibrated cameras, we use voxel carving to create the visual hull. Then the sample points are placed inside the visual hull with uniform density, and a triangular mesh for the surface is created. Alternatively, the surface mesh can be generated first and then the points can be selected from the volume bounded by the surface.

We now fit $M_{\mathcal{R}}$ to the consecutive frames, both forward and backward in time. For each frame, the fitting is performed iteratively. Let S_f and M_f denote the input surface and the model at the f-th frame, respectively. Assuming forward fitting, we would like to compute a deformation of M_f so that it approximates well the input shape of S_{f+1}. We use M_f as the initial approximation M^0_{f+1}. From the current approximation M^k_{f+1} after the k-th iteration, we compute M^{k+1}_{f+1} by minimizing the weighted sum of three cost functions: the elastic potential energy (described in the previous section), a surface distance measure (Section 4.1) and a self-intersection cost (Section 4.2). The optimization for each frame is iterated until convergence.

To speed up the queries for the closest points on S_f, we calculate a signed distance field d_f for each frame, using a Dijkstra-style marching front algorithm.

4.1. Fitting based on Surface Distance

With each point p on the surface ∂M^k_{f+1} of the model M^k_{f+1}, we associate a target position $\mathbf{t}_p$ that is the closest point on the target surface S_{f+1}. Given the signed distance field d_{f+1} of S_{f+1}, $\mathbf{t}_p$ can be easily found by following the gradient of d_{f+1} backward if $d_{f+1}(\mathbf{x}_p) > 0$ (like the point p in Figure 2 (a)), and forward otherwise (like the point r in the same figure). We use the sum of the squared distance between corresponding positions as the distance measure between the model M^k_{f+1} and S_{f+1}:

$$D = \sum_{p \in \partial M} w_p \, \mathrm{d}^2_D(\mathbf{x}_p, \mathbf{t}_p). \tag{13}$$

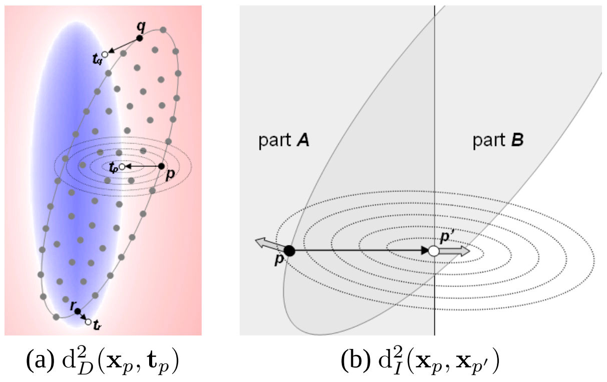
Since the naive approach of following the gradient of a distance field does not produce satisfactory results, we add three modifications to improve the match. First, the squared distances are weighted by a confidence value w_p for each association of points, which is defined by

$$w_p = \max(0, \cos\theta), \tag{14}$$

where θ is the angle between the surface normal $\mathbf{n}_p$ and the gradient of distance function d_{f+1} at p on M^k_{f+1}.

Second, we modify the distance function d^2_D to measure the distance from the *surface* S_{f+1} rather than just the Euclidean distance from the target position $\mathbf{t}_p$ on S_{f+1}. We use a squared anisotropic distance function that penalizes movement in a tangent direction less than movement along the normal direction:

$$\mathrm{d}^2_D(\mathbf{x}_p, \mathbf{t}_p) = (\mathbf{x}_p - \mathbf{t}_p)^T R^T \begin{bmatrix} s^2_D & 0 & 0 \\ 0 & s^2_D & 0 \\ 0 & 0 & 1 \end{bmatrix} R\,(\mathbf{x}_p - \mathbf{t}_p), \tag{15}$$



(a) $\mathrm{d}^2_D(\mathbf{x}_p, \mathbf{t}_p)$ (b) $\mathrm{d}^2_I(\mathbf{x}_p, \mathbf{x}_{p'})$

Figure 2. Our surface distance measure (a) and self-intersection penalty (b) in 2D : the point p is pushed toward $\mathbf{t}_p$, while q is not pushed at all because its weight w_q is zero. The concentric ellipses represent different isolevels of the distance measure (a) or the self-intersection penalty (b) for the location $\mathbf{x}_p$ of the point p. The amount of anisotrophy depends on s_D and s_I, respectively.

where R is a rotation that maps the gradient vector $\nabla d_{f+1}(\mathbf{x}_p)$ to a vector pointing along the z-axis, and $s_D \in [0, 1]$ defines the amount of anisotropy. Ideally, we could vary the anisotropy according to the differential properties of the input surface S_{f+1}. However, estimating differential properties reliably for noisy data is hard and therefore we use $s_D = d(\mathbf{x})/d_{dmi}$ in the maximum range of $[0.1, 1]$ in all experiments described in this paper. Intuitively speaking, this method pushes (or pulls) the model towards the target surface while allowing the model to slide along the surface.

Finally, in some cases where the motion is fast and the deformation is larger than the size of surface features across two consecutive frames, the association of target position $\mathbf{t}_p$ to surface point p may lose continuity and the surface can be pushed in the wrong direction. (For example, the tip of a limb may be pushed toward the surface of another limb.) To prevent this situation, we check every pair $(\mathbf{t}_p, \mathbf{t}_q)$ to see whether or not $\mathbf{t}_p$ and $\mathbf{t}_q$ are too close (within a third of their domain of influence) while p and q are not neighbors at all in the rest-state model. In that case, both associations of target positions for p and q are invalidated and excluded from the evaluaton of D (Eq. 13). In this way, the model is fit using good associations only, and the associations are improved over subsequent iterations.

4.2. Resolving Self-Intersections

Most of the motions of living creatures with limbs (including humans) involve collisions and contacts of different body parts. The distance measure described in the previous section (together with the simple invalidation rule) is not sufficient to capture the interaction between body parts around the contact point, and consequently self-intersections may occur in the fit models. To resolve this problem, we use another term in the cost function that is based on the squared anisotropic distance between the intersected surface points (Figure 2 (b)):

$$I = \sum_{p \in \partial M} \mathrm{d}_I^2(\mathbf{x}_p, \mathbf{x}_{p'}), \tag{16}$$

where p is a point on the surface in the intersecting region, p' is the closest point to p on the other surface, and

$$\mathrm{d}_I^2(\mathbf{x}_p, \mathbf{x}_{p'}) = (\mathbf{x}_p - \mathbf{x}_{p'})^T R^T \begin{bmatrix} s_I^2 & 0 & 0 \\ 0 & s_I^2 & 0 \\ 0 & 0 & 1 \end{bmatrix} R\,(\mathbf{x}_p - \mathbf{x}_{p'}), \tag{17}$$

where $s_I = 0.1$ and R is the rotation that maps the average vector of the surface normal at p and the negated surface normal at p' to a vector pointing along the z-axis. The spatial hashing of the points in the model is used to identify the points in the intersecting regions efficiently.

This cost function serves as a virtual force that pulls the surface points in the intersecting region out of the interior of the other body part. Note that both D (Eq. 13) and I (Eq. 16) are quadratic functions with respect to the global coordinate vector $\mathbf{x}$ of the model, without any linear or constant terms.

4.3. Iterative Optimization

With the equations of the elastic potential energy E (Eq. 12), the sum of squared distance D (Eq. 13) and the self-intersection cost I (Eq. 16) described in the previous sections, our goal is to find a global coordinate vector $\mathbf{x}$ of the model which minizes the weighted sum of the three cost functions:

$$\mathcal{E} = \alpha E + \beta D + \gamma I = \mathbf{x}^T \mathbf{A} \mathbf{x} - 2\mathbf{b}^T \mathbf{x} + c. \tag{18}$$

This objective function is quadratic with respect to $\mathbf{x}$, and its minimun can be found by solving the linear equation

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{19}$$

Note that $\mathbf{A}$ is sparse, symmetric and positive definite. In particular, the contribution from the surface distance term D is a block diagonal matrix, which, in all practical cases, resolves singularities inherent in the stiffness matrix $\mathbf{K}$ from the elastic potential energy term E in the objective function. Therefore, the final linear equation (Eq. 19) can be solved efficiently by the conjugate gradient method, allowing us to obtain the approximation M_{f+1}^{k+1} from M_{f+1}^{k}.

This process is iterated until convergence, updating the deformed model as $\{M_{f+1}^1, M_{f+1}^2, \cdots, M_{f+1}^{k_{max}}\}$. We set the maximum number of iterations to 10, and terminate early if the decrease of D is less than 1 percent of the total decrease from the previous iterations.

There are several parameters to be controlled in our approach: Young's modulus e and Poisson ratio ν (Eq. 9) of the model, anisotropy parameter s_D (Eq. 15) and s_I (Eq. 17), and objective function weights α, β and γ (Eq. 18). We assume uniform material properties (uniform flexibility) in the model and set $\nu = 0.3$, $s_D \in [0.1, 1]$, $s_I = 0.1$ and $\beta = \gamma$ in all the experiments in this paper. Since α, β, γ and e are all linear parameters and only the ratio of the weights in the objective function matters, setting all of them is equivalent to controlling two parameters – one to balance between the restoring force from internal elastic potential and the deforming force from external data conformity, and the other to control the self-intersection penalty. Our iterative fitting works well over a wide range of parameter values (Although fails when β is too small, due to the singularity), and the effects of all the forces change gradually with respect to the parameter values.

5. Experimental Results

We have implemented our fitting approach in both 2D and 3D. During the preparation of the input data, silhouettes were extracted from the segmented videos of multi-view calibrated cameras, and the visual hull for each frame was created using voxel carving. Then, the model was constructed by sampling the interior of the visual hull at a chosen frame, and a signed distance field was created for each frame. Finally, the model was fit to the visual hull of each frame by iterative optimization.

Our experiments focus on fitting elastic models to noisy and ambiguous surfaces of visual hulls. These examples demonstrate the robustness of our approach and the capability of handling large rotations and complex motions using silhouette information only. In practical situations, a template model can be deformed into an initial pose under user guidance and more information (color, texture and feature correspondences) can be incorporated into the fitting scheme.

The Dance dataset (Figure 3) was synthesized from motion capture data of a dancing person. We added a simple skin to each bone of the skeleton, and multi-view videos were created by putting six virtual cameras around it. The rest-state model was constructed from the skeletal model with the skin. It consists of 2465 points, and the domain of influence was selected so that each point has exactly 20 neighbors. The result shows that the element-free elastic model closely approximates the actual motion even though the motion is quite fast and involves large rotations of body parts. Also note that the visual hulls from six cameras have considerable amount of incorrectly labelled regions due to self-occlusion.

It should be noted that the fitted models are the result of global optimization, therefore some of them may still have local discrepancies with input surfaces and

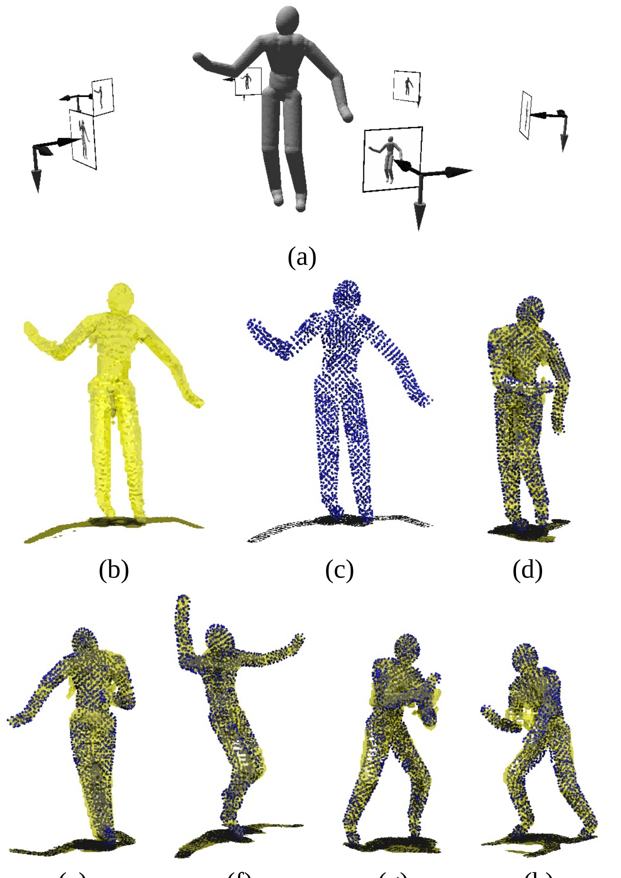


Figure 3. The Dance dataset: the simulated video capture setup (a), the initial input surface (b), the rest-state model (c) and models fitted to some frames in the volume sequence (d)-(h).

self-intersections of the limbs despite the self-intersection penalty in the objective function.

The Camel dataset provided by Brostow *et al*. [4] (Figure 4) is a volume sequence of a camel puppet captured from fourteen calibrated cameras. The videos were segmented by background subtraction and the resulting visual hull surfaces exhibit a significant amount of noise. The rest-state model was created from the first frame of the sequence. We observe from the results that the elastic model effectively smoothes out noise and finds the optimal pose which minimizes both internal deformation potentials and external data observation errors. The model consists of 3908 points with 20 neighbors per point.

By using the signed distance field, pre-determined neighborhoods and spatial hashing, the objective function can be evaluated efficiently using local information only, and the optimal solution is calculated by the conjugate gradient solver with a sparse matrix representation. The fitting requires less than 20 seconds for each iteration (a single optimization of setting up and solving Eq. 19) on a P4 2.8 GHz desktop PC, and 5 to 10 iterations (about 3 min.) were performed for each frame in all the experiments.

6. Conclusions

We have presented a new method for fitting an element-free elastic model to a volume sequence of complex motions. As a set of sample points with attributes that represent material properties of an object, an element-free elastic model is easy to generate and modify, and can describe a variety of objects including rigid, non-rigid or articulated objects. Combined with rotation compensation, our method also allows us to capture large rotations of body parts. Our iterative fitting procedure efficiently finds the optimal pose that minimizes a combination of elastic potential energy, a surface distance measure and a self-intersection penalty.

Our experimental results demonstrate that our element-free elastic models can be used to capture complex 3D motions from segmented videos in the presence of noise and self-occlusions. All the results were generated from the silhouette information only.

Simulation of elasticity for volume fitting induces a computational cost that is far from real-time for the computing power at present, but our approach is effcient enough for off-line applications such as markerless motion capture. The analysis of local deformations of fitted models and refinement of the initial model are among our future research directions toward these off-line applications in computer vision. Another interesting research direction is to use more information, such as color, texture and feature correspondences along with silhouettes to fit element-free elastic models.

References

[1] K.-J. Bathe. *Finite Element Procedures*. Prentice Hall Inc., Upper Saddle River, NJ, 1996.

[2] S. Beissel and T. Belytschko. Nodal integration of the element-free galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 139(1):49–74(26), 1996.

[3] T. Belytschko, Y. Y. Lu, and L. Gu. Element-free galerkin methods. *International Journal for Numerical Methods in Engineering*, 37:229–256, 1994.

[4] G. J. Brostow, I. Essa, D. Steedly, and V. Kwatra. Novel skeletal representation for articulated creatures. In *ECCV*, pages Vol III: 66–78, 2004.

[5] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. In *ICCV*, pages 694–699, 1995.

[6] K. M. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. In *CVPR'03*.

[7] K. M. Cheung, S. Baker, and T. Kanade. Visual hull alignment and refinement across time: A 3d reconstruction algorithm combining shape-from-silhouette with stereo. In *CVPR'03*.

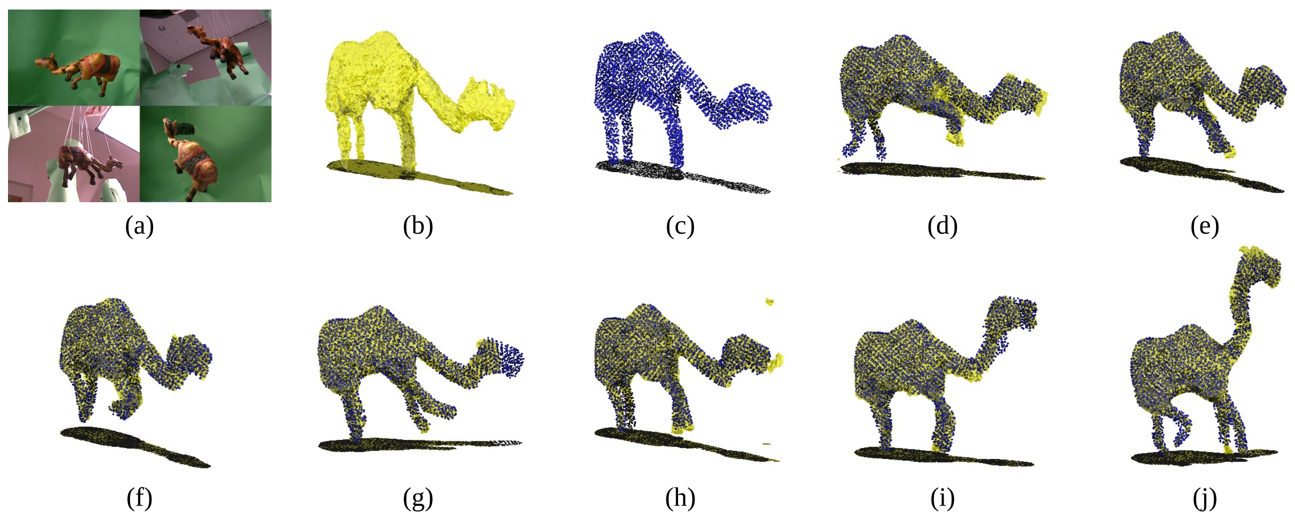


Figure 4. The Camel dataset: four stills from the fourteen videos of a camel puppet (a), the input surface at the first frame (b), the rest-state model generated from the first frame (c) and models fitted to some frame in the volume sequence (d)-(j).

[8] J. Choi and A. Szymczak. Fitting solid meshes to animated surfaces using linear elasticity. *ACM Transactions of Graphics - in review*.

[9] C.-W. Chu, O. C. Jenkins, and M. J. Mataric. Markerless kinematic model and motion capture from volume sequences. In *CVPR*, 2003.

[10] K. L. Clarkson. A program for convex hulls. http://cm.bell-labs.com/netlib/voronoi/hull.html.

[11] L. D. Cohen and I. Cohen. Finite element methods for active contour models and balloons for 2d and 3d images. *PAMI*, 15(11):1131–1147, 1993.

[12] J. Dolbow and T. Belytschko. An introduction to programming the meshless element free galerkin method. *Archives in Computational Mechanics*, 5(3):207–241, 1998.

[13] T.-P. Fries and H.-G. Matthies. Classification and overview of meshfree methods, 2004. Technical Report from TU Braunschweig, Germany.

[14] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, 1(4), 1998.

[15] R. Kehl, M. Bray, and L. V. Gool. Full body tracking from multiple views using stochastic sampling. In *CVPR'05*.

[16] J. J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13, 1984.

[17] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. In *ICCV*, 1999.

[18] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.

[19] A. Laurentini. The visual hull concept for silhouette-based image understanding. *PAMI*, 16(2):150–162, 1994.

[20] T. McInerney and D. Terzopoulos. A finite element model for 3d shape reconstruction and nonrigid motion tracking. In *ICCV*, Berlin, Germany, 1993.

[21] M. Mueller, B. Heidelberger, M. Teschner, and M. Gross. Meshless deformations based on shape matching. In *SIGGRAPH*, 2005.

[22] J. Park, D. Metaxas, and L. Axel. Volumetric deformable models with parameter functions: A new approach to the 3d motion analysis of the lv from mri-spamm. In *ICCV*, pages 700–705, 1995.

[23] A. Petland and B. Horowitz. Recovery of nonrigid motion and structure. *PAMI*, 13(7):730–742, 1991.

[24] D. Terzopoulos, A. Witkin, and M. Kass. Constraints on deformable models: Recovering 3d shape and nonrigid motion. *Artificial Intelligence*, 35, 1988.

[25] M. Teschner, B. Heidelberger, M. Müller, D. Pomeranets, and M. Gross. Optimized spatial hashing for collision detection of deformable objects. In *Proceedings of Vision, Modeling, Visualization*, pages 47–54, 2003.

[26] G. Vogiatzis, P. H. S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *CVPR*, 2005.

[27] K.-Y. K. Wong and R. Cipolla. Structure and motion from silhouettes. In *ICCV*, 2001.