International Journal of Computer Vision manuscript No.
(will be inserted by the editor)

A Bayesian Approach to Multi-view 4D Modeling

Chun-Hao Huang† · **Cedric Cagniart**† · **Edmond Boyer** · **Slobodan Ilic**

Received: date / Accepted: date

Abstract This paper considers the problem of automatically recovering temporally consistent animated 3D models of arbitrary shapes in multi-camera setups. An approach is presented that takes as input a sequence of frame-wise reconstructed surfaces and iteratively deforms a reference surface such that it fits the input observations. This approach addresses several issues in this field that include: large frame-to-frame deformations, noise, missing data, outliers and shapes composed of multiple components with arbitrary geometries. The problem is cast as a geometric registration with two major features. First, surface deformations are modeled using mesh decomposition into elements called patches. This strategy ensures robustness by enabling flexible regularization priors through inter-patch rigidity constraints. Second, registration is formulated as a Bayesian estimation that alternates between probabilistic datal-model association and deformation parameter estimation. This accounts for uncertainties in the acquisition process and allows for noise, outliers and missing geometries in the observed meshes. In the case of marker-less 3D human motion capture, this framework can be specialized further with additional articulated motion constraints. Extensive experiments on various 4D datasets show that complex scenes with multiple objects of arbitrary nature can be processed in a robust way. They also demonstrate that the framework can capture human motion and provides visually convincing as well as quantitatively reliable human poses.

Keywords Multi-view · Deformable Surface Tracking · Mesh Registration · Expectation-Maximization · 3D Human Motion Tracking · Bayesian Network

C.-H. Huang
Technische Universität Müchen
E-mail: huangc@in.tum.de

C. Cagniart
Technische Universität Müchen
E-mail: cagniart@in.tum.de

E. Boyer
LJK - INRIA Grenoble Rhône-Alpes
E-mail: edmond.boyer@inrialpes.fr

S. Ilic
Technische Universität Müchen
E-mail: slobodan.ilic@in.tum.de

† : The first two authors contributed equally to this paper.

1 Introduction

4D Modeling refers to the ability to produce animated shape sequences using videos of real dynamic scenes. This is, in particular, an alternative way to create realistic animated 3D contents which appears to be a tedious and expensive operation with graphical tools. Over the last decade, multi-view techniques have emerged as an effective solution to this aim. Multi-camera systems are inexpensive to deploy and allow an non-intrusive capture of 3D performances. Considerable efforts have been devoted to devise algorithms that automatically perform 3D reconstructions from the acquired visual data (see [40] for a survey). Yet when applied to temporal sequences of moving objects, most of these methods provide temporally topology-inconsistent models by treating each frame independently, ignoring the dynamic nature of the observed event.

A more complete 4D modeling of observed scenes is hence of crucial importance. Producing temporally consistent reconstructions enables artists to perform geometry or appearance edits on the content that automatically propagate through time. Animated meshes are also of interest for the purpose of compression, storage,

transmission and real-time rendering. Temporal information can be leveraged to discover latent object properties and to enrich a semantic description of the scene.

Recovering the temporal evolution of shapes from visual data remains challenging, due to perturbations in the acquisition process such as missing data or outlying geometries. In the case of arbitrary deformable surfaces, it becomes even more difficult since there is no easy way to identify erroneous correspondences. Remedies are in general two: a sophisticated data-model association (likelihood) or a constraining deformation model (prior). Most existing methods adopt the latter to limit the search space, e.g., *skeletons* in 3D human pose estimation [19, 52]. Some methods also leverage previously observed deformed surfaces by machine learning or modal analysis techniques, to effectively reduce the dimensionality of the optimization. These works lack generality and quickly lose precision when observations leave the explainable range of the model. Furthermore, they offer limited perspectives for the case of complex scenes involving multiple objects of unknown nature. Some approaches, on the other hand, choose to make fewer assumptions and simply compute the deformation of a reference mesh [12], which is more of our interest. It then boils down to an optimization problem over vertex positions, with prior knowledge regularizing vertex displacements. The generality, however, comes at the cost of an increased sensitivity to noise, and it becomes necessary to either strictly control the acquisition pipeline, or to look into the associated error model. A noise-resilient data-model association is therefore indispensable.

We present a generic surface-based approach that deforms a reference mesh according to the 3D observations, essentially casting the mesh deformation problem as a geometric registration problem. Our method builds on three complementary contributions [8, 9, 22]. The first contribution [8] consists of a mesh deformation and a numerical optimization framework that divides a surface into numerous patches. These patches are used to locally smooth data terms and to enforce inter-patch rigid constraints w.r.t. its rest state. Secondly, we formulate the registration problem in a probabilistic way, in order to account for the noise in the acquisition [9]. This increases the robustness and therefore relaxes the need for object-specific deformation priors such as skeletons, allowing for more complex scene modeling.

Nevertheless, recovering skeletal poses is still desired in many applications that requires human articulated motion information. As the last contribution, we propose to infer such information from surface deformations. This comes as a side product of our deformation framework and fits naturally to the aforementioned probabilistic formulation. Together, they form a *simultaneous optimization* framework, which can also be interpreted from a Bayesian network perspective. Unlike [48], who use this idea only in refinement, we directly use it for tracking [22]. To the best of our knowledge, in the context of 3D human tracking, this is the first work which recovers both human shapes and poses with a single optimization. In extension to the work in [22], this paper provides additional analysis on the simultaneous optimization strategy and presents an improved solution for the pose estimation from surface shapes.

The remainder of this paper is organized as follows: Sect. 2 contains an overview of the related work and discussions of our contributions. Our method is described in Sect. 3 (surface) and Sect. 4 (skeleton). In Sect. 5, we establish the probabilistic formulations. Thorough evaluations are presented in Sect. 6. Discussions are then provided in Sect. 7, followed by concluding remarks in Sect. 8.

2 Previous Work

Estimating motions from discrete observations is by nature ambiguous and amounts to solving the data-model correspondence problem. Motion cues have to be balanced with priors encoding the deformation range. We briefly review previous work around these two themes: first the commonly-used motion cues in visual data and then the existing paradigms of deformation models.

Motion cues in visual data. Several works extract sparse feature correspondences from images to infer the motion. Popular photometric keypoint descriptors such as SIFT [31] or SURF [3] can be complemented by edge features, geodesic-intensity histograms [45] or the extrema of the geodesic integral [51]. This sparse information is then propagated to the rest of the mesh using intersections of level sets of harmonic functions [1], or using sparse matches as handles in a mesh deformation framework [12]. However, these feature matches contain many outliers. Popa et al [36] recommend an aggressive filtering that checks the consistency of the forward and backward optical flow and of the 3D point correspondences. De Aguiar et al [12] find a maximally consistent (Euclidean distance preserving) subset of handles in the graph of correspondences via robust spectral matching. Starck and Hilton [45] cast the feature matching as a discrete labeling task, making use of Markov Random Field technique to regularize the results.

In the case of multi-camera systems, however, it is tempting to avoid such sophisticated processing on

sparse features, because deformation priors combined with geometric cues like silhouettes are usually sufficient to obtain a good approximated deformation. Some works drive the deformation from visual hulls exclusively: Vlasic et al [52] define an non-linear fitting function and optimize a skeletal pose while Corazza et al [11] use a articulated variant of the Iterative-Closest-Point (ICP) algorithm. For more generality, however, the surface tracking task can be cast as a non-rigid registration of point sets. Li et al [28] show that in the case of range scans, it can be driven from purely geometric information. To increase their robustness, a number of works model the uncertainty of point clouds. Many works suggest that Expectation-Maximization is an effective approach: Horaud et al [21] investigate its use for the registration of rigid and articulated point sets, while the Coherent Point Drift algorithm by Myronenko and Song [33] treats arbitrary deformations by regularizing the displacement field.

Deformation models. A number of approaches explicitly define acceptable intrinsic deformation space by manually creating a parametric model. For instance, in the case of human motion capture, Vlasic et al [52] or Gall et al [18] optimize the pose of skeletal models before fitting the surface to visual data. Similarly, for faces, parameterized models of human expressions can be used for tracking [27, 37]. Instead of manually defining it, other works seek to learn the possible deformations from the data beforehand and to regularize the results accordingly. This is usually achieved by exploiting previous observed deformations of the object via dimension reduction [15, 50] or machine learning techniques [20]. Salzmann et al [39] recover the 3D deformation of a piece of cloth using Principal Component Analysis on a randomly generated set of possible configurations. Chai et al [10] drive facial animations by tracking a low number of facial features and mapping them back to a linear combination of previously acquired laser scanned faces. All these methods dramatically increase the robustness by using severe constraints on the deformations. Yet, they are difficult to extend to more complex scenes, as they are often tailored for a specific object of interest and do not generalize well.

More generic methods use physics-inspired models to ensure *as rigid as possible* deformations, building on the knowledge in Computer Graphics for simulation and interactive modeling (see [34] for a survey). This is usually done by emulating elastic behavior and discouraging stretching and bending w.r.t. the rest state. A survey by Botsch and Sorkine [5] discusses how this can be approximated by penalizing changes in differential properties. A well-known example, Laplacian coordinate [44], is define as the edge displacement vector, and the regularization is exerted by preventing it from changes after local rotations. Recently, a number of Computer Vision works, e.g., [17, 29], have used this strategy to preserve local details during tracking. As demonstrated in [43], the Laplacian framework behaves well for its original purpose, that is, interpolation with manually set constraints. However, its use as in regularization terms in Vision applications requires more attention, since they usually involve large motions in noisy environments. In fact, as noted in the work by White et al [53], these methods do little to penalize strain and can cause many artifacts around the borders of open meshes. As a consequence, [18, 52] only use it to preserve small details when large motions have been already recovered. De Aguiar et al [12] use a tetrahedral variation of this principle, but the coarse-to-fine strategy remains.

Such a coarse-to-fine concept suggests the need to decouple the deformation parametrization from the intrinsic nature of the original geometry. This has obvious advantages in terms of computational complexity, and is therefore used for interactive editing [7, 49] as it allows to explicitly optimize local rotations and to embrace the 'inherently non-linear' [5] nature of the surface deformation. However, in the case of data-driven mesh deformation, this need mostly stems from the imperfection of noisy visual data. As such, several works increase the robustness by embedding the deformed geometry in a coarser control structure, benefiting from the averaging effects on data terms [28]. It should be noted that this decoupling from the intrinsic nature does not necessarily result in a loss of precision, since deformations of objects usually lies in low dimensional spaces, as indicated by the success of skeletal or dimension reduction techniques.

In this work, we recover the temporal evolution of a surface by fitting a reference mesh (usually the first topologically suitable reconstruction) to a sequence of independently reconstructed meshes. We do not assume object-specific deformation priors and yet still remain robust to noise and errors from the input. Our contributions in that aspect are two-fold: firstly we propose a coarse control structure made of surface patches where data terms are sampled and averaged [8]. Secondly, based on this structure, we propose a Bayesian formulation of the mesh registration problem [9]. It offers dense and soft patch-data correspondences and increases the robustness by explicitly modeling outliers. The registration process behaves like a probabilistic ICP algorithm. Besides surface tracking, we additionally develop two ways to estimate human articulated motion, inspired respectively by *beta coordinate* [48] (which shares a sim-

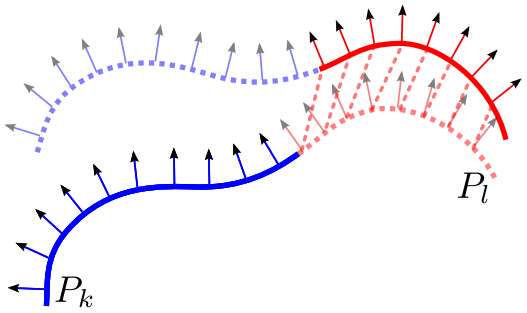


Fig. 1 P_k and P_l have their own rigid transform and predict positions for their own vertices (solid curves) and the vertices of the neighboring patches (dotted curves). The rigidity energy penalizes the discrepancies in these predictions (dotted lines).

ilar spirit with Laplacian coordinate), and by *linear blend skinning* [26]. Both approaches integrate easily into our registration framework and hence benefit from its robustness. The second way is further shown to yield more valid poses than the first way from our previous work [22]. To sum up, the presented method is a generic surface-based tracking framework that can track arbitrary object shapes robustly but also specializes to articulated human motion.

3 Patch-based Mesh Deformation Framework

As discussed in Sect. 2, the deformation of dense meshes can often be characterized by a set of low dimensional parameters, instead of raw vertex positions. A number of works, targeted at the compression of mesh animations [13, 24], show that a small set of rigid transformations and weighting functions can encode visually complex deformations, including those of cloth. In this section, we present our mesh deformation framework based on small surface elements called *patches*, each of which has a rigid body motion. The final vertex positions are computed by blending the transformations over different patches locally, in a way similar to *linear blend skinning* [26]. Patches and blending weights are computed on the template prior to the tracking process.

3.1 Patches

A rigid transformation w.r.t. the world coordinates is associated with each patch P_k. It is parameterized by the position of the patch center $\mathbf{c}_k$ and a rotation matrix $\mathbf{R}_k$ (or equivalently by a unit-length quaternion $\mathbf{q}_k$). This rigid transform yields for every vertex of the mesh a predicted position $\mathbf{x}_k$ (dotted lines in Fig. 1):

$$\mathbf{x}_k = \mathbf{R}_k(\mathbf{x}^0 - \mathbf{c}_k^0) + \mathbf{c}_k, \tag{1}$$

where the superscript 0 denotes the corresponding variable in the reference pose.

The mesh is deformed by linearly blending the predictions made by different neighboring patches for each vertex. The weighting functions α_k are Gaussians of the Euclidean distance to the center of the patch $\mathbf{c}_k$. As for the support for blending, we consider only the patch P_k itself and its direct neighbors, i.e., $k \cup \mathcal{N}_k$:

$$\mathbf{x} = \sum_{s \in k \cup \mathcal{N}_k} \alpha_s \mathbf{x}_s. \tag{2}$$

Blending weights α_s are normalized to add up to 1. Defining the shape parameter $\mathbf{\Theta}$ as $\{(\mathbf{R}_k, \mathbf{c}_k)\}_{k=1:N_p}$, where N_p is the total number of patches, the deformation of a mesh is a function of $\mathbf{\Theta}$, namely, $\mathbf{x}(\mathbf{\Theta})$.

Ideally, patches on the surface should follow the intrinsic nature of the shape, e.g. its rigid parts. However, in the absence of prior knowledge on this structure, they are preferably regularly distributed over the surface. To this purpose, our patching method considers geodesic distances, takes a maximum patch radius as parameter and seeds patches greedily. The idea is to randomly choose a vertex to be the center of the first patch and then to grow this patch until the maximum radius is reached. The subsequent patch centers are chosen among the unassigned vertices which lie on the most existing patch boundaries. The front of a new patch is propagated from the center until the maximum radius is reached or until the processed vertex is closer to the center of another patch. We assume that vertices distribute uniformly on the mesh and therefore we approximate the geodesic distance with the number of edges in the shortest path linking two vertices. The behavior of this greedy patch seeding algorithm are shown in Fig. 2.

3.2 Rigidity constraints

Given this patch-based control structure, we define an energy that penalizes non-rigid surface deformations w.r.t. its original status. This is inspired by the theory of elasticity in which deformable objects are defined by their material properties and a rest configuration. Many works simulate this behavior by constraining the displacement field [17, 29, 44, 43]. They operate on raw vertex positions and discourage the displacement fields from changing after local transformations, which are either expressed as a linear function of vertex positions [17, 29, 44], or iteratively estimated [43]. On the other hand, Botsch et al [7] and Sumner et al [49] optimize local transformations rather than vertex positions, and define elastic constraints between the transformed

Fig. 2 Greedy patching procedure evolution on the Stanford Armadillo model (170k vertices) with a maximum patch radius of 40. From left to right: patching after $1, 3, 4, 30, 143$ patches have been seeded.

vertices themselves. Although this is not as physically accurate as the proper computation of the strain energy, we pursue this path because it naturally integrates to our patch-based representation. Our rigidity energy, as defined in Eq. (3) and shown in Fig. 1, simply enforces the predictions $\mathbf{x}_k(v)$ and $\mathbf{x}_l(v)$ of a vertex v by two neighboring patches, k and $l \in \mathcal{N}_k$, to be consistent:

$$E_r(\boldsymbol{\Theta}) = \sum_{k=1}^{N_p} \sum_{l \in \mathcal{N}_k} \sum_{v \in P_k \cup P_l} w_{kl}(v) \left\| \mathbf{x}_k(v) - \mathbf{x}_l(v) \right\|^2 . \quad (3)$$

The choice of the weights $w_{kl}(v)$ is of importance as it allows to encode material properties. In all of our experiments, they are proportional to the sum of the blending weights: $\alpha_k(v) + \alpha_l(v)$ and is normalized such that all the $w_{kl}(v)$ depending on the same vertex v sum up to 1, simulating therefore uniform stiffness.

3.3 Optimization

Evolving a mesh can then be viewed as an optimization problem balancing the rigidity energy E_r (with a weight λ_r) and data terms which are squared functions f^2:

$$\underset{\boldsymbol{\Theta}}{\operatorname{argmin}} \; \lambda_r E_r(\boldsymbol{\Theta}) + \sum f^2 \left(\mathbf{x}(\boldsymbol{\Theta}) \right), \quad (4)$$

where f^2 can be manually specified constraints (Sect. 3.5), or more sophisticated probabilistic likelihoods (Sect. 5.3). Eq. 4 is a non-linear least-squares problem, since $\boldsymbol{\Theta}$ involves rotations. We employ an iterative Gauss-Newton method to find the minimizer. Instead of directly using elements in the rotation matrix as parameters with additional soft constrains for matrix-orthogonality, as in [49], we optimize the energy function w.r.t. small affine updates, $\boldsymbol{\theta}_k = [\mathbf{u}_k, \mathbf{v}_k] \in \mathbb{R}^6$. Specifically, the update in rotation, $\hat{\mathbf{R}}_k$, is approximated by the first-order expansion of the exponential mapping of $[\mathbf{u}_k]_\times$, namely, $\mathbf{I} + [\mathbf{u}_k]_\times$, and $\mathbf{v}_k$ is the displacement of $\mathbf{c}_k$. As shown in Eq. 5, this formulation allows to write the update of coordinates $\mathbf{x}_k$ linearly in $\boldsymbol{\theta}_k$, and thus of $\mathbf{x}$ linearly in $\{\boldsymbol{\theta}_k\}_{k=1:N_p}$.

$$\begin{aligned} \mathbf{x}_k \mapsto \mathbf{x}'_k &= \hat{\mathbf{R}}_k(\mathbf{x}_k - \mathbf{c}_k) + \mathbf{c}'_k \\ &= \mathbf{x}_k + [\mathbf{u}_k]_\times(\mathbf{x}_k - \mathbf{c}_k) + \mathbf{v}_k = \mathbf{x}_k + \mathbf{K}_k(\mathbf{x}_k)\, \boldsymbol{\theta}_k, \\ &\text{with } \mathbf{K}_k(\mathbf{x}_k) = \left[[\mathbf{c}_k - \mathbf{x}_k]_\times \; \mathbf{I} \right]. \end{aligned} \quad (5)$$

The first order approximation of Eq. 3 then yields a simple quadratic form in the update parameters.

$$\begin{aligned} E_r(\boldsymbol{\Theta}) \simeq \sum_{k=1}^{N_p} \sum_{l \in \mathcal{N}_k} \sum_{v \in P_k \cup P_l} w_{kl} \, \| \left(\mathbf{x}_k + \mathbf{K}_k(\mathbf{x}_k)\, \boldsymbol{\theta}_k \right) \\ - \left(\mathbf{x}_l + \mathbf{K}_l(\mathbf{x}_l)\, \boldsymbol{\theta}_l \right) \|^2. \end{aligned} \quad (6)$$

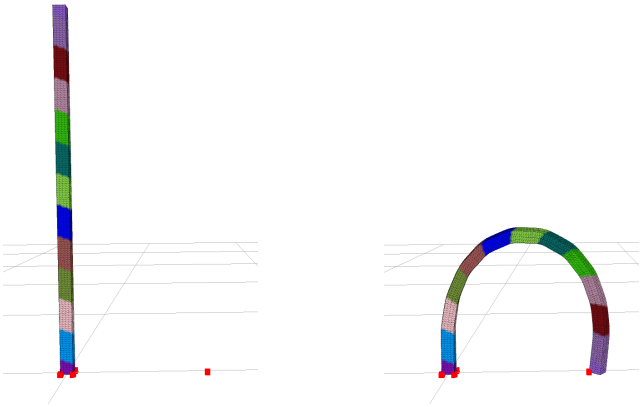
Knowing that $\mathbf{K}_k(\mathbf{x}_k)$ is actually the Jacobian of $\mathbf{x}_k$ w.r.t. $\boldsymbol{\theta}_k$, the gradient of Eq. 6 w.r.t. $\boldsymbol{\theta}_k$ can be expressed using the chain rule:

$$\left[\frac{\partial E_r}{\partial \boldsymbol{\theta}_k} \right] = \left[\frac{\partial E_r}{\partial \mathbf{x}_k} \right] \left[\frac{\partial \mathbf{x}_k}{\partial \boldsymbol{\theta}_k} \right] = \left[\frac{\partial E_r}{\partial \mathbf{x}_k} \right] \mathbf{K}_k(\mathbf{x}_k). \quad (7)$$

These first order approximations can be used to compute the gradient and the minimum of the quadratic approximation of the energy in the tangent space but the actual energy must be evaluated on the updated $\boldsymbol{\Theta}$. To recover the $\mathbf{R}_k$'s we actually perform the update of rotations in quaternion representation and normalize the result to limit the accumulation of numerical error. The cost function from Eq. 4 is in practice minimized by performing a line search in tangent space and making sure that the corresponding step taken on the parameter manifold actually decreases the energy.

3.4 Numerical Considerations

In the Gauss-Newton algorithm, the Hessian matrix is approximated by $\mathbf{G}^\top \mathbf{G}$ where $\mathbf{G}$ is the Jacobian matrix. In our case, $\mathbf{G}^\top \mathbf{G}$ is a $6N_p \times 6N_p$ sparse matrix, with only a few 6×6 non-zero blocks. This structure is fixed and reflect the connectivity in the graph of patches. It is good to look for a fill-reducing indexing of patches [25].



(a) Object in its rest pose (b) Deformed object

Fig. 3 Behavior of the patch deformation framework in an interactive application. The target positions for the constrained vertices of the original mesh are indicated by red boxes.

Since Eq. 7 offers an analytic formulation, it is more practical to compute directly the matrix, rather than computing first $\mathbf{G}$ and then $\mathbf{G}^\top\mathbf{G}$. This computation can be mostly parallelized thanks to the sparse blocky structure. For example, the off-diagonal 6×6 blocks come from the rigidity terms of Eq. 6 and require for block (k, l) to accumulate gradient terms over $P_k \cup P_l$. These operations can be distributed on multiple processors as they access different parts of the memory.

Finding a minimizer of the quadratic approximation at each step of the Gauss-Newton algorithm can then be tackled by any available sparse solver, either direct or iterative, as discussed in [6]. In our experiments, we have at most 400 patches and thus reasonably small matrices. We therefore use a sparse Cholesky factorization package [38] that pre-computes the symbolic part of the factorization for efficiency. We also confirmed that a simple Conjugate Gradient algorithm with a diagonal preconditioner is a functional alternative that requires much less involved implementation effort.

3.5 Example: Interactive Deformation

To evaluate the behavior of this patch-based mesh deformation framework, we implemented a simple interactive application where 3D constraints on the vertices positions could manually be set by the user. The results displayed in Fig. 3 illustrate two important facts on the method. Firstly, even though the number of patches is low, the resulting deformation of the mesh is reasonably smooth. Secondly, the constraints are set on vertices of the mesh and not on patch centers. This shows that the data terms can be sampled on the original geometry and not on the graph controlling the deformation.

4 Skeleton

As demonstrated above, our patch-based deformation framework evolves a surface with a low dimensional set of parameters. It is not limited to humanoid surfaces, holding the possibility to handle complex scenes. Nonetheless, in the context of human motion tracking, many authors use an intrinsic data structure to guide the deformation in an even lower dimensional space [19, 52], followed by a shape refinement stage. This structure often resembles the skeleton of human bodies, and each node actually corresponds to one human body joint. Due to this anatomical meanings, an articulated skeletal structure is usually preferred than the 3D surface shape in many applications. Thus, in this section, we extend our framework so that it retains the generality of surface-based methods, while providing skeletal poses as *side products*.

4.1 Initialization and Pose Parametrization

Our skeleton is a hierarchical tree structure consisting of N_j nodes (joints). It has to be placed properly inside the mesh such that the root is close to the pelvis of the body, and each vertex v is assigned to a branch-node joint, as in Fig. 4(a). This *rigging* process and vertex-joint associations are automatically accomplished offline once on the reference surface, using the software `Pinocchio` [2] prior to the tracking process.

After rigging, many authors attach a local coordinate frame on each joint and parameterize the pose as the rigid transformations of these coordinates [11, 19, 52]. Meshes are thereby controlled by these transformations based on blend skinning techniques. However, surface deformation is by nature high dimensional and difficult to be fully characterized by only few rigid transformations. For humanoid surfaces, typically $N_p \approx 150$ yields plausible surface shapes, while N_j is usually less than 20. We therefore advocate for the reverse strategy: guiding the skeleton by the surface deformation, i.e., *inverse skinning*. Formally, the pose is represented as a set of positions for each joint j: $\mathcal{J} = \{\mathbf{x}_j\}_{j=1:N_j}$ and is parameterized as a function of shapes, $\mathcal{J}(\mathbf{\Theta})$. As illustrated in Fig. 4(b), patch transformations can be regarded as intermediate controlling primitives that lie in between complex vertex positions and overly simplified skeletons. Both high dimensional surface shapes and low dimensional skeletal poses are controlled by the transformations of patches. To this end, we associate each patch P_k to a joint $j(k)$ by a taking majority vote on vertex-joint assignments. Given the patch-joint associations, there are two ways to devise $\mathcal{J}(\mathbf{\Theta})$: either through bones, or directly through joints.

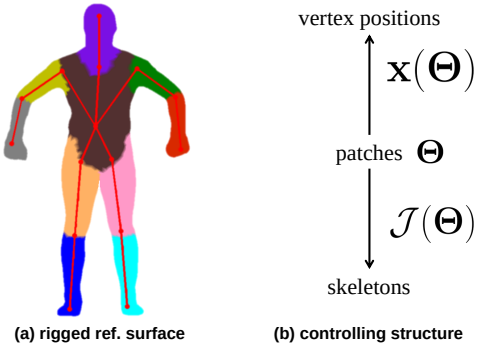


Fig. 4 (a) vertices associated to the same joint have the same color. (b) the control structure of our deformation framework.

4.2 Inverse Skinning through Bones

On the reference mesh, we compute the beta-coordinate proposed by Straka et al [48] in a per-patch manner. It represents the displacement between the bone and the patch center: $\boldsymbol{\beta}_k^0 = \boldsymbol{\delta}_k^0 - \mathbf{c}_k^0$, where $\boldsymbol{\delta}_k^0$ is the linear combination of the positions of joint $j(k)$ and its child:

$$\boldsymbol{\delta}_k^0 = \gamma_k \mathbf{x}_j^0 + (1-\gamma_k)\mathbf{x}_{\text{child}(j)}^0. \tag{8}$$

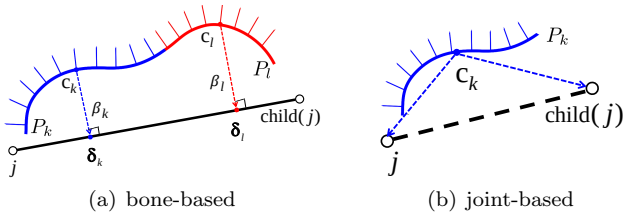
γ is chosen such that $\boldsymbol{\beta}$ is orthogonal to the bone, as shown in Fig. 5(a). We drop the dependency notation (k) in order to keep equations uncluttered.

By preventing $\boldsymbol{\beta}$ from varying after deformation, we encourage bones to follow their corresponding patches. More specifically, if a patch undergoes a rigid transformation $\mathbf{T}_k$ during tracking, it predicts the new $\boldsymbol{\delta}_k$ as $\mathbf{T}_k(\boldsymbol{\delta}_k^0) = \mathbf{c}_k + \mathbf{R}_k\boldsymbol{\beta}_k^0$. Substituting into Eq. 8, one can formulate the new position of a bone as:

$$\gamma_k \mathbf{x}_j + (1-\gamma_k)\mathbf{x}_{\text{child}(j)} = \mathbf{c}_k + \mathbf{R}_k\boldsymbol{\beta}_k^0. \tag{9}$$

Here γ stays fixed to prevent bones from sliding along the surface. Eq. 9 shows the linear relation between the transformation of a patch and the location of its associated joint. Since there are two unknown variables, $\mathbf{x}_j$ and $\mathbf{x}_{\text{child}(j)}$, we need at least two equations (i.e., two patches) to determine the new position of one bone. In practice, we stack this linear relation for every patch and form a linear system:

$$\underbrace{\begin{bmatrix} \vdots & \vdots \\ \dots\ \gamma_k\mathbf{I}\ \dots & (1-\gamma_k)\mathbf{I}\ \dots \\ \vdots & \vdots \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} \vdots \\ \mathbf{x}_j \\ \vdots \\ \mathbf{x}_{\text{child}(j)} \\ \vdots \end{bmatrix}}_{\mathbf{J}} = \underbrace{\begin{bmatrix} \vdots \\ \mathbf{c}_k + \mathbf{R}_k\boldsymbol{\beta}_k^0 \\ \vdots \end{bmatrix}}_{\boldsymbol{\Delta}}, \tag{10}$$



(a) bone-based (b) joint-based

Fig. 5 Two inverse skinning strategies. In (b), the bone is shown in dashed line because there is no explicit concept of bone.

where $\mathbf{B}$ is a $3N_p \times 3N_j$ matrix, $\mathbf{J}$ is a $3N_j \times 1$ vector containing the positions of all joints, and $\boldsymbol{\Delta}$ is $3N_p \times 1$ vector containing all $\mathbf{T}_k(\boldsymbol{\delta}_k^0)$. In general, N_p is larger than N_j, so Eq. 10 is an over-determined system whose optimal solution can be obtained via pseudoinverse:

$$\mathbf{J} = (\mathbf{B}^\top\mathbf{B})^{-1}\mathbf{B}^\top\boldsymbol{\Delta}. \tag{11}$$

Eq. 11 demonstrates that the pose of the subject can be parameterized as the function of the shape parameter (encoded in $\boldsymbol{\Delta}$), namely, $\mathcal{J}(\boldsymbol{\Theta})$. Thus, each time a when humanoid surface is deformed, the pose of the skeleton can be computed via Eq. 11 accordingly. Since pseudo-inverse solves a linear system in the least-square sense, one can also formulate an equivalent energy term:

$$E_{bone}(\boldsymbol{\Theta}, \mathcal{J}) = \sum_{k=1}^{N_p} \kappa_k \left\| \mathbf{T}_k(\boldsymbol{\delta}_k^0) - \boldsymbol{\delta}_k' \right\|^2, \tag{12}$$

where $\boldsymbol{\delta}_k' = \gamma_k \mathbf{x}_j + (1-\gamma_k)\mathbf{x}_{\text{child}(j)}$. It simply enforces the right-hand and the left-hand side of Eq. 9 to be consistent. This way we can also weight the contribution of each patch differently with κ_k.

4.3 Inverse Skinning through Joints

An alternative way to guide the skeleton is more straightforward: predicting the positions of joints by every associated patch, as illustrated in Fig. 5(b). When a patch k moves to a new position $\mathbf{c}_k$ with rotation $\mathbf{R}_k$, it assumes the associated joint moves together:

$$\mathbf{x}_k(j) = \mathbf{R}_k(\mathbf{x}_j^0 - \mathbf{c}_k^0) + \mathbf{c}_k. \tag{13}$$

The position of each joint is then recovered by linearly blending the predictions from different patches:

$$\mathbf{x}_j = \sum_k \tau_k \mathbf{x}_k(j), \tag{14}$$

where the weights τ_k are determined similarly as α in Eq. 2. This approach fits naturally into the patch-based deformation framework, and it requires at least only one patch to predict two joint positions.

It is informative to combine Eqs. 13 and 14 together in one formulation, as it emphasizes the difference of our approach from the others:

$$\mathbf{x}_j = \sum_k \tau_k \mathbf{T}_k (\mathbf{T}_k^0)^{-1} \mathbf{x}_j^0. \tag{15}$$

Here $(\mathbf{T}_k^0)^{-1}$ means $(\mathbf{x}_j^0 - \mathbf{c}_k^0)$ in Eq. 13, and $\mathbf{T}_k$ means rotating with $\mathbf{R}_k$ and moving to the new center $\mathbf{c}_k$. Eq. 15 mirrors the linear-blend skinning formulation:

$$\mathbf{x}_v = \sum_j w_j \mathbf{T}_j (\mathbf{T}_j^0)^{-1} \mathbf{x}_v^0, \tag{16}$$

where one first represents the vertices in local bone coordinate frames, applies new transformations, and blend the predictions from relevant joints. We clearly see that Eqs. 15 and 16 share the same mathematical computations, only the opposite operands. We argue that inferring skeletons from surfaces because it is a logical way to determine low dimensional representations (skeletal poses) from high dimensional deformations (surface shapes), instead of doing the other way around with unrealistic rigid-body-part assumptions.

Eq. 14 provides the second way to parameterize the pose $\mathcal{J}$ as a linear function of the shape parameter $\mathbf{\Theta}$, encoded in $\mathbf{x}_k(j)$. Let $\mathcal{N}_j$ denotes the patches connected to joint j or the parent of j. Similar to the first approach, we also formulate an equivalent energy term:

$$E_{joint}(\mathbf{\Theta}, \mathcal{J}) = \sum_{j=1}^{N_j} \sum_{k \in \mathcal{N}_j} \tau_{jk} \left\| \mathbf{x}_j - \mathbf{x}_k(j) \right\|^2. \tag{17}$$

4.4 Optimization

Given the two different ways to obtain the pose $\mathcal{J}$ from the shape $\mathbf{\Theta}$ (Eqs. 11 and 14), one can, during tracking, first get the optimal $\mathbf{\Theta}$ via Eq. 4 and then compute the pose $\mathcal{J}$ as a post-processing step. The alternative, as suggested by Straka et al [48], is to simultaneously estimate the two, with Eq. 4 is augmented into:

$$\underset{\mathbf{\Theta}, \mathcal{J}}{\operatorname{argmin}} \; \lambda_r E_r(\mathbf{\Theta}) + \lambda_s E_{skl}(\mathbf{\Theta}, \mathcal{J}) + \sum f^2 \left(\mathbf{x}(\mathbf{\Theta}) \right). \tag{18}$$

E_{skl} is either E_{bone} or E_{joint}, and λ_s is the balancing weight. This is again a non-linear least-squares problem. The quadratic approximation in Sect.. 3.4 applies here as well to both E_{bone} or E_{joint}, so the numerical considerations basically remain unchanged; only the approximated Hessian matrix $\mathbf{G}^\top \mathbf{G}$ becomes a $(6N_p + 3N_j) \times (6N_p + 3N_j)$ matrix. Although solving Eq. 18 recovers poses and shapes at once, we anyway stress that, due to the difference in the numeric scales, poses $\mathcal{J}$ behave like side products of shapes $\mathbf{\Theta}$. As discussed later in Sect. 7.2, it has negligible effects on shape deformations. In the next section we explain more on our data term and the optimization framework.

5 Simultaneous Shape and Pose Tracking

As discussed in Sects. 1 and 2, we deal with data-driven mesh deformation and cast the problem as a geometric registration of 3D point sets. The data term in Eq. 18 has to be designed carefully. In this section, we first consider our problem from a Bayesian network prospect, which leads us to a data term in the spirit of Gaussian Mixture Model (GMM), and then we explain our optimization framework which corresponds to the well-known Expectation-Maximization algorithm.

We aim at estimating the shape $\mathbf{\Theta}$ of the surface and the pose $\mathcal{J}$ of the skeleton simultaneously. In a Bayesian context, this means that given a set of observed 3D points, the estimation of shape and pose is achieved by maximizing the a posteriori (MAP) probability:

$$\max_{\mathbf{\Theta}, \mathcal{J}} P(\mathbf{\Theta}, \mathcal{J} | \mathcal{Y}), \tag{19}$$

where $\mathcal{Y} = \{y_i\}_{i=1:m}$ is the set of 6D vectors containing the observed 3D coordinates $\{\mathbf{y}_i\}_{i=1:m}$ and normals. Considering $P(\mathcal{Y})$ as a constant, maximizing Eq. 19 is equivalent to maximize the joint distribution, which can be decomposed as follows:

$$\max_{\mathbf{\Theta}, \mathcal{J}} P(\mathcal{Y}, \mathbf{\Theta}, \mathcal{J}) = P(\mathcal{Y} | \mathbf{\Theta}, \mathcal{J}) \cdot P(\mathcal{J}, \mathbf{\Theta}). \tag{20}$$

5.1 Bayesian Network Model

We employ two assumptions which further simplify Eq. 20:

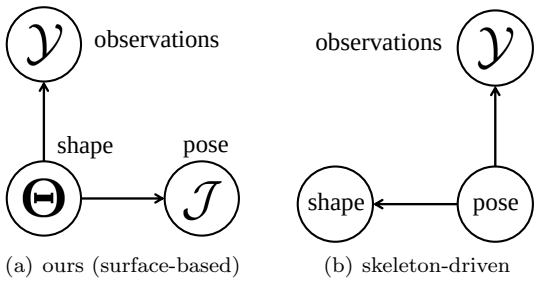
1. surface-based approach:

$$P(\mathcal{J}, \mathbf{\Theta}) = P(\mathcal{J} | \mathbf{\Theta}) \cdot P(\mathbf{\Theta}). \tag{21}$$

2. conditional independence between $\mathcal{J}$ and $\mathcal{Y}$:

$$P(\mathcal{Y} | \mathbf{\Theta}, \mathcal{J}) = P(\mathcal{Y} | \mathbf{\Theta}). \tag{22}$$

The first assumption comes from the fact that we rely on the shape parameter $\mathbf{\Theta}$ to determine the pose parameter $\mathcal{J}$, as described in Sect. 4. $P(\mathcal{J}, \mathbf{\Theta})$ is thus factorized as in Eq. 21, not $P(\mathbf{\Theta}|\mathcal{J}) \cdot P(\mathcal{J})$. Here, $P(\mathcal{J}|\mathbf{\Theta})$ represents the probability of the skeleton pose given the shape, and $P(\mathbf{\Theta})$ is the prior knowledge on possible shape deformations. Secondly, we assume that $\mathcal{J}$ is conditionally independent of $\mathcal{Y}$ given $\mathbf{\Theta}$, i.e., $\mathcal{Y} \perp\!\!\!\perp \mathcal{J} | \mathbf{\Theta}$. It is a reasonable assumption, since from the input data perspective, one usually observes only the surface of human bodies (shape) rather than the anatomical structure (skeleton). It makes sense that when conditioning



(a) ours (surface-based) (b) skeleton-driven

Fig. 6 Directed graphical models of two strategies.

on the shape, the observations and the skeleton can no longer influence each other. Eq. 22 holds as the property of conditional independence [4], and $P(\mathcal{Y}|\mathbf{\Theta})$ is considered as the likelihood between observations and the shape.

Based on Eqs. 21 and 22, the joint distribution is:

$$P(\mathcal{Y}, \mathbf{\Theta}, \mathcal{J}) = P(\mathcal{Y}|\mathbf{\Theta}) \cdot P(\mathcal{J}|\mathbf{\Theta}) \cdot P(\mathbf{\Theta}). \tag{23}$$

Alternatively, one can decompose the joint distribution as follows:

$$P(\mathcal{Y}, \mathbf{\Theta}, \mathcal{J}) = P(\mathcal{Y}, \mathcal{J}|\mathbf{\Theta}) \cdot P(\mathbf{\Theta}), \tag{24}$$

and see directly that the conditional independence between $\mathcal{J}$ and $\mathcal{Y}$ lead us to Eq. 23 by definition. Note that Eq. 23 is actually a Bayesian network model, whose directed graph is illustrated in Fig. 6(a). The figure also shows, in (b), the dependence graph corresponding to skeleton-based motion parameterizations. Next we proceed with each term in Eq. 23.

5.2 Shape Prior and Pose Posteriors

In the absence of knowledge on the nature of the shape, we model a probability distribution over the range of shape deformations by seeding patches on a reference surface and making the approximation:

$$P(\mathbf{\Theta}) \propto e^{-\lambda_r E_r(\mathbf{\Theta})}, \tag{25}$$

where $E_r(\mathbf{\Theta})$ is the rigidity energy defined in Eq. 3. This energy emulates elastic behavior with respect to the patched reference mesh. Because our patching approach infers the topology of the object from the vertex connectivity, this reference mesh has to be topologically suitable. See Sect. 7 for more discussion on this aspect.

The pose posteriors measure the probability of a pose given a certain shape, approximated as:

$$P(\mathcal{J}|\mathbf{\Theta}) \propto e^{-\lambda_s E_{skl}(\mathbf{\Theta}, \mathcal{J})}, \tag{26}$$

where $E_{skl}(\mathbf{\Theta}, \mathcal{J})$ is the skeleton energy in Eq. 18. This approximation assumes that the connectivity between the skeleton and the surface is stable, or say, the patch-joint association $j(k)$ does not change during tracking, which usually holds for human subjects.

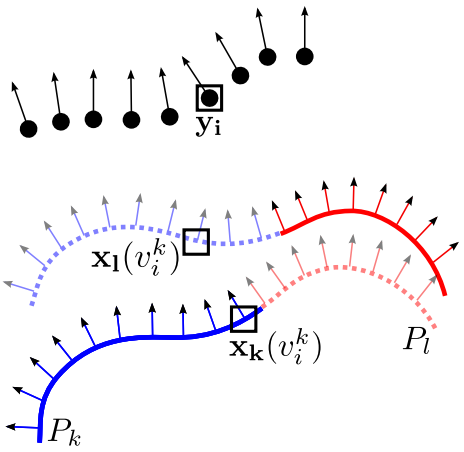


Fig. 7 A point/normal y_i with position $\mathbf{y_i}$ is associated to v_i^k, the closest vertex with a compatible normal among all the predictions for the patch P_k. In this case v_i^k is selected because of its position and normal predicted by the neighboring patch P_l.

5.3 Likelihood

The likelihood $P(\mathcal{Y}|\mathbf{\Theta})$ actually corresponds to the data term in Eq. 18 and remains to be approximated to complete the generative model. This is done with a mixture of distributions parameterized by a isotropic variance σ^2, where each component corresponds to a patch. This requires latent variables z_i for each observation $y_i \in \mathcal{Y}$, where $z_i = k$ means that y_i was generated by the mixture component associated with P_k. Similar to [33], we also increase the robustness to outliers by introducing a uniform component in the mixture to handle points in the input data that could not be explained by the patches. This uniform component is supported on the scene's bounding box and is indexed by $N_p + 1$.

$$P(y_i|\mathbf{\Theta}, \sigma) = \sum_{k=1}^{N_p+1} \mathit{\Pi}_k P(y_i|z_i = k, \mathbf{\Theta}, \sigma), \tag{27}$$

where the $\mathit{\Pi}_k = p(z_i = k|\mathbf{\Theta}, \sigma)$ represent probabilities on the latent variables marginalized over all possible values of y_i. In other words, they are prior probabilities on model-data assignments. We define them as constants $p(z_i = k)$ that add up to 1, using the expected proportion of outlier surface in the observations and the ratios of patch surfaces in the reference mesh.

The patch mixture component with index k must encode a distance between the position $\mathbf{y}_i$ and the patch P_k while accounting for the alignment of normals. For

computational cost reasons, we model this distance by looking for each patch P_k in its different predicted poses (i.e., the positions $\{\mathbf{x}_l(v)\}_{l\in\{k\}\cup N_k, v\in P_k}$ and corresponding normals as in Fig. 7) for the closest vertex v_i^k with a compatible normal. Two points are considered compatible when the normals form an angle smaller than a threshold, set to 45° in all of our experiments. This leads to the model for each component of the GMM:

$$\forall k \in [1, N_p],$$
$$P(y_i|z_i = k, \mathbf{\Theta}, \sigma) \propto \begin{cases} \mathcal{N}(\mathbf{y}_i|\mathbf{x}(v_i^k), \sigma^2) & \text{if } v_i^k \text{ exists} \\ \epsilon & \text{otherwise,} \end{cases} \tag{28}$$

where ϵ encodes a negligible uniform distribution defined on the scene's bounding box, and $\mathcal{N}(\cdot)$ denotes Gaussian density.

5.4 Expectation-Maximization

The variables z_i can not be observed, but we can use the posterior distributions of Eq. 29 in the Expectation Maximization algorithm [14].

$$P(z_i = k|y_i, \mathbf{\Theta}, \sigma) = \frac{\Pi_k P(y_i|z_i = k, \mathbf{\Theta}, \sigma)}{\sum_{l=1}^{N_p+1} \Pi_l P(y_i|z_i = l, \mathbf{\Theta}, \sigma)}. \tag{29}$$

The idea is to replace $P(\mathcal{Y}|\mathbf{\Theta}, \sigma)$ with the marginalization over the hidden variables of the joint probability.

$$\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) = \ln \sum_Z q(Z) \frac{P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)}{q(Z)}, \tag{30}$$

where $q(Z)$ is a positive real valued function that sums up to 1. The concavity of the log function allows to write a bound on the function of interest:

$$-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) \leq -\sum_Z q(Z) \ln \frac{P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)}{q(Z)}. \tag{31}$$

It can be shown that given a current estimate $(\mathbf{\Theta}^t, \sigma^t)$, it is optimal to choose $q(Z) = P(Z|\mathcal{Y}, \mathbf{\Theta}^t, \sigma^t)$ because the bounding function then touches the bounded function at $(\mathbf{\Theta}^t, \sigma^t)$. This means that the bounding function should be the expected complete-data log-likelihood conditioned by the observed data:

$$-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) \leq const - E_Z[\ln P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)|\mathcal{Y}, \mathbf{\Theta}^t, \sigma^t]. \tag{32}$$

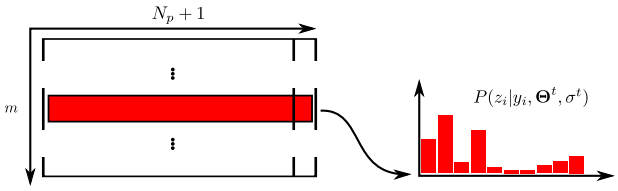


Fig. 8 The soft assignment matrix holds the posterior patch-assignment distributions for every target point. As such, the lines are normalized to add up to 1. The last column of the matrix corresponds to the outlier class.

We rewrite $P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)$ by making the assumption that each observation y_i is drawn from $\mathcal{Y}$ in an independent identically distributed way:

$$P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma) = \prod_{i=1}^{m} P(y_i, z_i|\mathbf{\Theta}, \sigma) \tag{33}$$
$$= \prod_{k=1}^{N_p+1} \prod_{i=1}^{m} \left[P(y_i, z_i = k|\mathbf{\Theta}, \sigma)\right]^{\delta_k(z_i)}. \tag{34}$$

The choice made for $q(z)$ then allows to write:

$$E_Z[\ln P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)|\mathcal{Y}, \mathbf{\Theta}^t, \sigma^t] = \sum_{k=1}^{N_p+1} \sum_{i=1}^{m} E_Z[\delta_k(z_i)|\mathcal{Y}, \mathbf{\Theta}^t, \sigma^t] \ln[\Pi_k p(y_i|z_i = k, \mathbf{\Theta}, \sigma)], \tag{35}$$

which leads to the bounding function to be minimized:

$$-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) \leq const - \sum_{k=1}^{N_p+1} \sum_{i=1}^{m} P(z_i = k|y_i, \mathbf{\Theta}^t, \sigma^t) \ln P(y_i|z_i = k, \mathbf{\Theta}, \sigma). \tag{36}$$

We use the Expectation-Maximization algorithm to iteratively re-evaluate the $(\mathbf{\Theta}, \sigma)$ and the posterior probability distributions on the latent variables $\{z_i\}$.

In the E - Step the posterior $P(z_i|y_i, \mathbf{\Theta}^t, \sigma^t)$ functions are evaluated using the current estimation $\mathbf{\Theta}^t, \sigma^t$ and the corresponding predicted local deformations of the mesh. As defined in Eq. 29, these functions require to find for each target vertex y_i and patch k the vertex index v_i^k of its nearest neighbor in the different predicted configurations of the patch. The complete E-Step amounts to the computation of a $m \times (N_p + 1)$ matrix whose lines add up to 1, as shown in Fig. 8. This is an very parallel operation as all the elements of this matrix can be evaluated independently, except for the normalization of each line that takes place afterwards. In theory, it is tempting to use space partitioning techniques to speed up the nearest neighbor

Table 1 Sequences used for evaluation. We apply different error measures, depending on the provided ground truth. A: silhouette overlap error. B: distances in $\mathbb{R}^3$ between markers and associated vertices. C: 3D error on joints positions. D: 2D pixel error on re-projected joints positions. A and B are metrics for shapes while C and D are for poses. As for input data, if photo-consistent meshes are not provided, we run a shape from silhouette algorithm [16] to obtain coarse reconstructed visual hulls.

Sequence	Views	Frames	Metric	Input	Sequence	Views	Frames	Metric	Input
Samba [52]	8	175	A	visual hull, silhouette	*Balloon* [9]	16	343	-	visual hull
Crane [52]	8	174	A		*Basketball* [9]	8	1330	-	
Handstand1 [52]	8	174	A		*Fighting* [30]	12	500	B	
Bouncing [52]	8	175	A		*S4_walking* [41]	4	350	C & D	
Dance [18]	8	573	A		*Flashkick* [46]	8	200	A	mesh, silhouette
Wheel [18]	8	280	A		*Head* [46]	8	250	A	
Handstand2 [18]	8	400	A		*Pop* [46]	8	250	A	
Skirt [18]	8	720	A		*Lock* [46]	8	250	A	

search. However, the dependency on the orientation of vertex normals makes this cumbersome. In practice, we run a brute-force search (see Sect. 7.5 for CPU/GPU timings).

The M - Step requires to minimize the bounding function defined by the the soft data - model assignment weights that were computed in the E-Step:

$$\boldsymbol{\Theta}^{t+1}, \mathcal{J}^{t+1}, \sigma^{t+1} = \operatorname{argmin} \Bigg[\lambda_r E_r(\boldsymbol{\Theta}) + \lambda_s E_{skl}(\boldsymbol{\Theta}, \mathcal{J}) - \sum_{k=1}^{N_p+1} \sum_{i=1}^{m} P(z_i = k | y_i, \boldsymbol{\Theta}^t, \sigma^t) \ln P(y_i | z_i = k, \boldsymbol{\Theta}, \sigma) \Bigg]. \tag{37}$$

The constant term is ignored due to the space limit. In this bounding function, all energy terms are weighted squared distances between 3D points. This fits exactly in the framework defined in Eq. 18. To prevent from degenerated mesh configurations, we however do not completely minimize the bounding function. Instead, we just run one iteration of Gauss-Newton algorithm, which amounts to minimizing the quadratic approximation of the objective function around ($\boldsymbol{\Theta}^t$, $\mathcal{J}^t$, σ^t).

It should also be noted that we do not solve Eq. 37 in one maximization step but instead follow the Expectation Conditional Maximization (ECM) approach [32] that shares the convergence properties of EM while being easier to implement. The idea is to replace the M-Step by a number of CM-steps in which variables are optimized alone while the others are fixed. Thus, in the M-step, we use the mesh deformation framework to first optimize for $\boldsymbol{\Theta}^{t+1}$, then $\mathcal{J}^{t+1}$ and finally update σ^{t+1}.

6 Results

In this section, we present the results of our algorithm on 16 publicly available multi-view sequences, ranging from rapid motions, e.g., *Flashkick* [46] and *Bouncing* [52], articulated motions, e.g., *Crane* [52], to non-rigid deformations, e.g., *Samba* [52]. We evaluate the shapes and the poses separately and analyze the results both qualitatively and quantitatively. Table 1 lists a overview of these sequences and the corresponding error measure. We first demonstrate the generality of tracking arbitrary objects, i.e., $\lambda_s = 0$, and then pay more attention to the application of human tracking.

6.1 Tracking Arbitrary Objects

The *Balloon* sequence shown in Fig. 9 from INRIA Rhône-Alpes involves two humans playing a ball. A topologically suitable reconstruction where three objects are separated is used as reference geometry. This sequence presents some outlying geometry resulting from the erroneous segmentation of shadows as foreground. The outlier class introduced in the Bayesian framework allows to limit the impact of such geometry in the inference of deformation by progressively reducing the weight of points that can't be explained by the model.

The *Basketball* sequence (1330 frames - about 55 sec.) in Fig. 10 was recorded in our own multi-camera studio. This scene is interesting for a few reasons. Firstly, the ball bounces between legs and is sometimes held close to the torso. The data involves two distinct objects with fast and complex interactions. Secondly, the visual hulls contain geometries exhibiting occlusions and numerous artifacts such as missing limbs. The results presented in Fig. 10 show that our algorithm can recover meaningful estimates of these difficult motions and deformations using a coarse model of the surface, even when confronted with numerous artifacts such as missing limbs, occlusions and self intersecting geometry. Experiments on these two sequences demonstrate that our framework is capable of handling generic deformations, not limited to human motions like [18, 52].

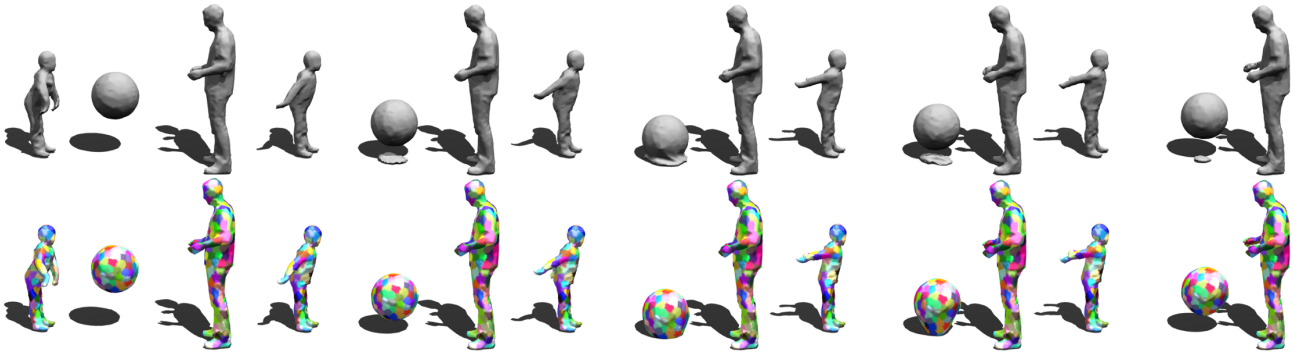


Fig. 9 Frames $292 - 295 - 296 - 297 - 298$ of *Ball* sequence show the effects of outlying geometry and the benefit of the outlier class in the Bayesian model. The shadows are wrongly segmented as foreground, which result in outlying geometry. One can observe that as the ball goes down, this outlying geometry is correctly handled by the EM framework and does not impact the estimation of the ball's deformation. As the ball bounces, the algorithm tries to find a compromise between rigidity and data while progressively reducing the weight of the erroneous points. It quickly converges to the proper estimate.

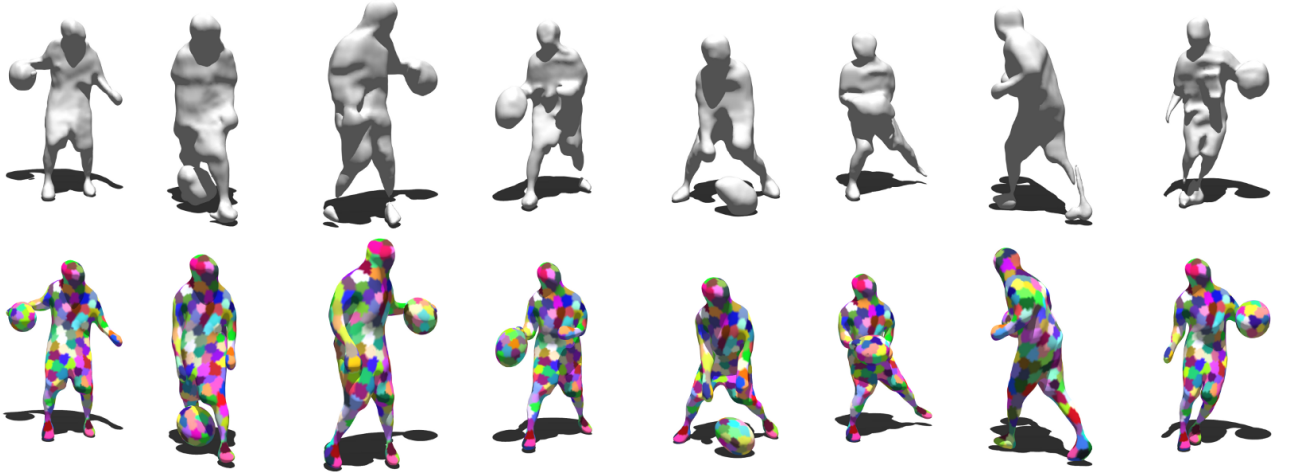


Fig. 10 Results on *Basketball* Sequence. The reference mesh is displayed on the top left. The hand and ball are manually separated for this initial mesh in a modeling software so that the deformation model would be topologically suitable. Note that despite a very coarse reference surface, wrong geometry, missing data and fast motion have a limited impact on our tracking algorithm.

6.2 Application: 3D Human Motion Tracking

6.2.1 Evaluation on human shapes

Now we turn our focus to human tracking and first evaluate our algorithm on shape estimation. As input, we used the results from either precise 3D reconstruction, or a rudimentary shape from silhouette. The metric is either silhouette overlap error in 2D, or distance to markers in 3D. For sequences evaluated with 2D error, we follow [18, 48, 52] who directly use silhouettes as input data and minimizes the re-projection error as a refinement step. This procedure relies on the same optimization framework defined in Sect. 3, uses considerably small patches and minimizes the residual error in silhouette overlap. The gradient of this energy is approximated by guessing from the current pose estimation which vertices are on occluding contours and pulls their projections towards the observed contours in the images. As shown in this section, compared with the state-of-the-art approaches, our algorithm provides satisfactory or even better results in both metrics.

Tracking with Visual Hulls as Input. We used the multi-view image data made public by MIT CSAIL group [52] and by MPI-Informatik [18] to run a simple shape from silhouette algorithm [16]. The resulting visual hulls, although only a coarse approximation of the true shape, are enough to drive the deformation of the provided template mesh through the sequences. In *Samba* sequence, skirts are difficult to handle for methods deforming a reference mesh as the interpolated surface between the bottom of the skirt and the legs has to undergo severe compression and stretching. We show in Fig. 11 that our approaches produces visually convincing results. We run our algorithm on eight sequences and compared the silhouette overlap error. Compared with [52], the results in Fig. 12 show that our approach

Table 2 Silhouette overlap errors (pix) for *Wheel*, *Dance*, and *Handstand2* sequences. Image resolution: 1004×1004.

	Wheel	*Dance*	*Skirt*	*Handstand2*
ours	**3961**	**3780**	**3413**	**4573**
Gall et al [18]	4168	5098	3678	5028
Straka et al [48]	4300	4100	4100	4900

yields a similar precision while using less complicated deformation model. Similarly, we also report better results than Gall et al [18] and Straka et al [48] in Table 2.

Tracking with Photo-consistent Meshes as Input. The `Surfcap Data` from University of Surrey consists of a series of temporally inconsistent meshes obtained by the photo-consistency driven graph-cut method [46]. Except for some rare reconstruction artifacts, these are overall very clean and smooth meshes. Because of their high resolution, meshes are down-sampled to roughly 10k vertices and fed to our algorithm. We present our results on four sequences. They show a hip-hop dancer whose moves are very challenging due to fast motions. The purpose of this experiments is to demonstrate the capacity of our method in capturing fast and large motions. In Fig. 13, our results on the *Flashkick* dataset show that we can cope with extremely fast deformations such as a backflip. In Fig. 14, we present our results on the *Pop* sequence in which the intricate and ambiguous motion of crossing arms is handled properly. Additionally, Fig. 15 shows the overlap error, which is given as the ratio of erroneous pixels and total number of pixels in the original silhouette. We attain approximately constant error at a value of 5%.

Temporal Consistency. In addition to silhouette overlap error which measures the discrepancies on 2D images, we also evaluate shapes in 3D. In *Fighting* sequence, the raw marker positions are provided for almost 500 frames. We follow Liu et al [30] who associated each marker to the closet vertex on the reference surface at frame zero and obtain $7.98mm$ initial distance. After tracking the whole sequence, the average distance between markers and the corresponding vertices becomes $38.49mm \pm 32.39mm$, which is reasonably small. Note that this includes measurement errors introduced by the marker-based system. It demonstrates the efficacy of our method on recovering temporal consistent meshes. The average time per frame for this sequence is 7 seconds. Compared to Liu et al [30] who attain more accurate results ($29.61mm \pm 25.50mm$), but require several minutes per frame for tracking, and to Stoll et al [47] which is fast (around 6 frames per second), but less accurate ($44.93mm \pm 27.16mm$), our approach certainly offers a good compromise between performance and accuracy. In Fig. 16, we overlay the estimated meshes and skeletons on images. We see that close interaction between subjects does not effect the results too much, which demonstrates that our method generalizes well to multiple humans.

6.2.2 Evaluation on human poses

It is also crucial to evaluate skeletal poses in the context of human tracking. The widely-used benchmark `HumanEva-II` [41] is challenging for 4D modeling because it contains too few cameras. We anyway test our algorithm on *S4* sequence *walking* section. Frames 298−335 are excluded due to the reported ground truth corruption. For the remaining frames, our method presents errors around $65mm$ in average (Table 3). According to Sigal et al [42], errors smaller than $80mm$ typically correspond to correct poses, which verifies the reliability of our method in terms of human pose estimation.

Compared with Corazza et al [11] who use visual hulls as well and report $80mm \pm 13mm$ errors, our approach is certainly more accurate and stable. Note that their approach is articulated ICP where deformations are guided by the underlying skeleton. This confirms the advantage of our inverse skinning strategy over conventional skeleton-based methods: when observations are noisy, a generic but robust surface-based approach offers better estimates on poses than approaches that constrain the search space with object-specific intrinsic deformation model. Still, we would like to stress that our goal is to track arbitrary objects and simultaneously provide a low-dimensional motion parametrization (which are skeletal poses) when the subjects are humans. We do not aim to estimate precise human joint locations since modeling a real human joint as a single 3D point is anyway an over-simplified assumption. The numerical error here is only a coarse measure of how well the pose is estimated. Further optimizing on this error does not necessarily improve the estimation.

7 Discussion

7.1 Influence of parameters

Our algorithm requires some parameters to be defined:

- the number of patch per subject, N_p
- the initial value for the standard deviation σ.
- the balancing between each energy term, i.e., λ_s and λ_r in Eq. 37.

It is worthwhile investigating how the patching affects tracking. We select 10 initial seeds to generate arbitrary segmentations, repeat this process for various

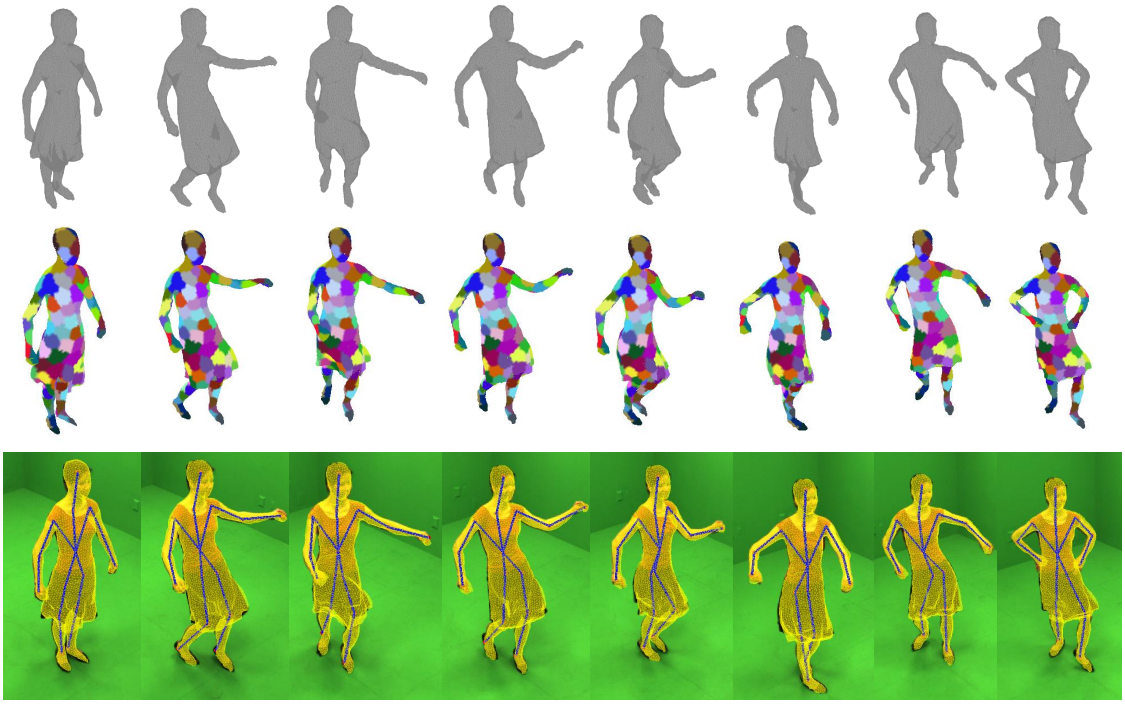
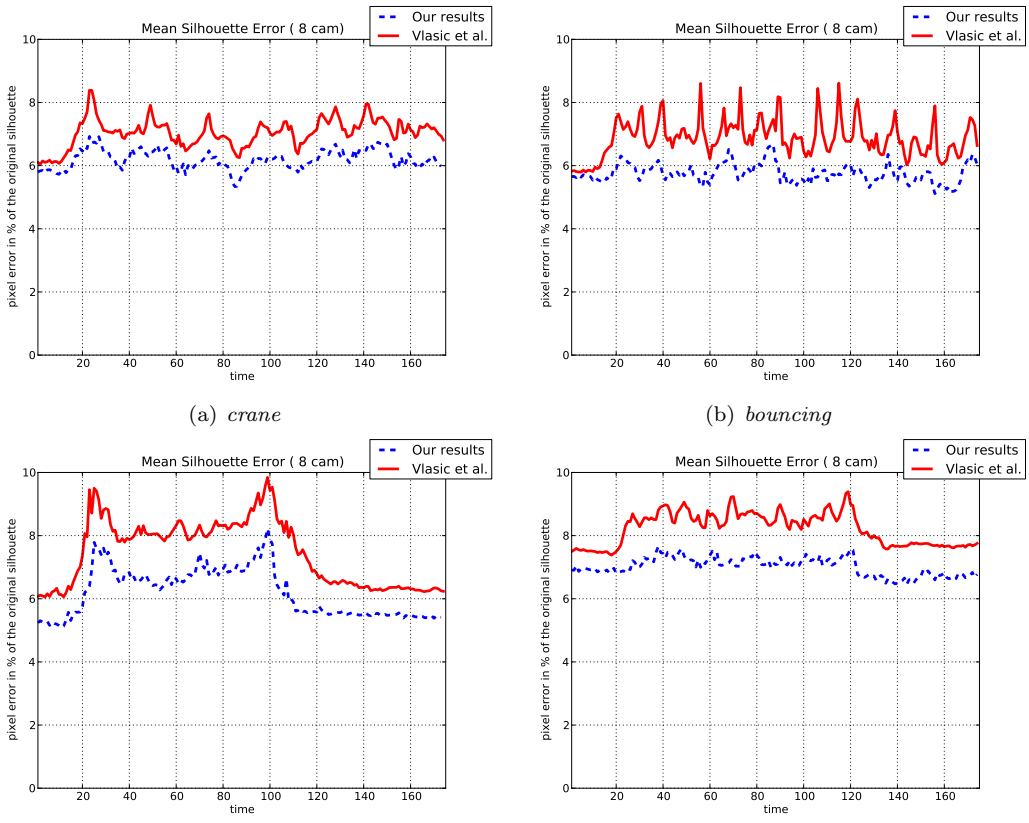


Fig. 11 Input visual hulls (1st row) and results (2nd and 3rd rows) on frame $10 - 38 - 68 - 78 - 84 - 100 - 118 - 124$ of *Samba* sequence. Here the skeletons are obtained via E_{joint}. Our approach yields visually convincing results on the tracking of a skirt.



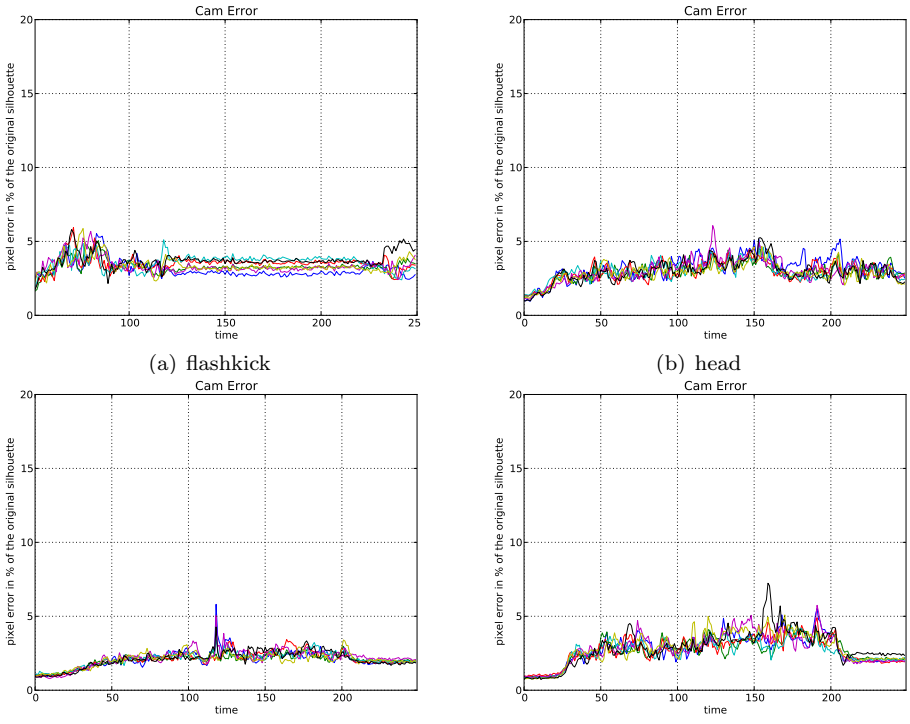
(a) *crane* (b) *bouncing*

(c) *handstand* (d) *samba*

Fig. 12 Numerical comparison of our silhouette overlap error to that of Vlasic et al [52]. Errors are shown in percentage. This graph indicates a consistent good behavior of our approach despite the much weaker underlying deformation model. Furthermore, our approach is fully automatic and requires no user intervention.

Fig. 13 Results on *Flashkick* sequence. The kickflip itself consists of extremely fast motion as it spans over 10 frames.

Fig. 14 Results on *Pop* sequence. Despite the geometrically ambiguous arm crossing, our approach still produces plausible results.



(a) flashkick (b) head

(c) pop (d) lock

Fig. 15 Silhouette overlap error of our deformed model in percentage of the original silhouette area. Each color represents one camera view. Image resolution: 1920 × 1080.

Table 3 3D error in millimeter and 2D errors in pixel number for *S4_walking*. Image resolution: 656 × 490.

Inverse skinning	3D (*mm*)	2D/Cam. 1 (pix)	2D/Cam. 2 (pix)	2D/Cam. 3 (pix)	2D/Cam. 4 (pix)
Bone (simultaneous)	65.48 ± 7.74	9.28 ± 1.31	8.32 ± 1.34	8.33 ± 1.36	9.99 ± 2.43
Bone (post processing)	65.78 ± 7.52	9.31 ± 1.31	8.38 ± 1.29	8.35 ± 1.36	10.08 ± 2.39
Joint (simultaneous)	$\mathbf{64.45 \pm 7.14}$	$\mathbf{9.14 \pm 1.19}$	$\mathbf{8.26 \pm 1.23}$	$\mathbf{8.21 \pm 1.31}$	$\mathbf{9.96 \pm 2.27}$
Joint (post processing)	65.06 ± 7.10	9.21 ± 1.23	8.36 ± 1.22	8.26 ± 1.34	10.07 ± 2.29

Fig. 16 Frames 190 and 250 of *Fighting* sequence in two views. Skeletons are obtained simultaneously via E_{joint}. Our approach applies well to multiple human subjects. Yellow circle: tracking is sometimes affected by close interaction, but is soon recovered.

patch size and report the raw tracking errors (without silhouette refinement) of *Crane* sequence in Fig. 17. It indicates that small patch sizes attain lower error, both in terms of bias (blue curve) and standard deviation (black error bar). Note anyway that smaller patches lead to larger approximated Hessian matrix $\mathbf{G}^\top\mathbf{G}$ and hence slow down the tracking. The number of patches is considered sufficient as long as it finely samples changes of curvature in the rest pose (see Fig. 2 for example). In practice, for articulated human motions, we find that $150 - 200$ patches offer good trade-off between speed and accuracy. Coarse-to-fine strategies that first estimate the rough shape with large patches quickly and refine it with small ones are also possible.

The initial σ determines how *greedy* our algorithm is. The starting value is always set to 2 times the average edge length. λ_r determines how *stiff* the reference surface is. It is empirically set on one of the sequences so that the residual energies have comparable magnitudes with the data term. All the other sequences use the same value. Since both E_{bone} and E_{joint} behave like auxiliary energy terms to compute poses from the given shapes, we set $\lambda_s = 1$ throughout all experiments.

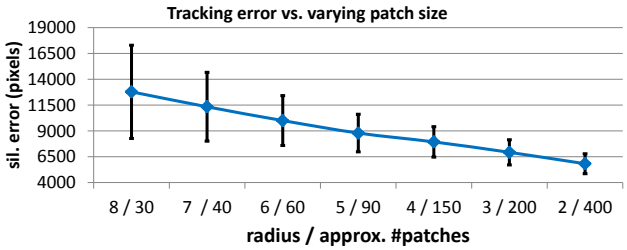


Fig. 17 Silhouette overlap error vs. different patch size / #patches. Image resolution: 1600 × 1200.

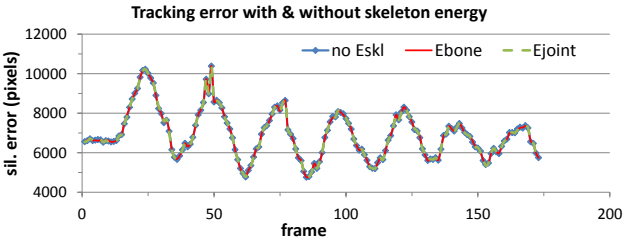
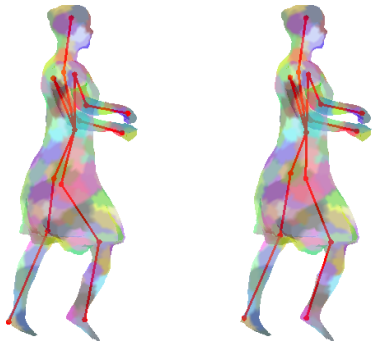


Fig. 18 Silhouette overlap error with different skeleton energies. Three strategies yield almost same errors in terms of shape. Image resolution: 1600 × 1200.

7.2 The role of E_{skl} in optimization

In Sect. 4.4 and 5.1, we emphasize that skeletal poses are side products of surface shapes. We verify this numerically by reporting again the raw silhouette overlap errors on *Crane* sequence in Fig. 18. Without the skeleton energy terms E_{skl} ($\lambda_s = 0$ in Eq. 37), the averaged error of surface shapes is 6938.73, whereas E_{bone} and E_{joint} get 6939.65 and 6938.69, respectively. Such a negligible difference confirms that in our inference framework, skeletal poses contribute little to surface shapes. E_{skl} plays the role as an augmentation term that recovers poses $\mathcal{J}$ in no time during optimizations. As shown previously in Table 3, $\lambda_s = 1$ already yields good numeric solutions for poses $\mathcal{J}$. In Eq. 37, the magnitude of E_{skl} is therefore relatively small compared to the data term and hence contribute little to the gradient of shapes $\mathbf{\Theta}$.



(a) E_{bone} (b) E_{joint}

Fig. 19 Visual comparison of results on frame 70 of *Samba* sequence from two different inverse skinning strategies.

7.3 E_{bone} Versus E_{joint}

To further select between E_{bone} and E_{joint}, we mark two observations in Table 3:

1. Simultaneous optimization yields equivalent or even slightly better results than post processing, which confirms our descriptions in Sect. 4.
2. Inverse skinning via joints attains consistently better and more stable results than via bones.

It is worth a closer look to contrast two approaches. We particularly choose *Samba* sequence which contains both rigid (arms) and non-rigid deformations (skirts). Results of frame 70 are shown in Fig. 19. Firstly, two surface shapes look visually the same, again confirming the above discussions that E_{skl} contribute little to the gradient of the shape $\mathbf{\Theta}$. To analyze how different deformations affect the poses, one can see that two strategies present similar poses on the arms but different behaviors on the legs. Since there is no ground truth for poses provided in this sequence, we check how the bone lengths of six body parts vary during tracking: the right hip bone (RHip), the right upper and lower legs and arms (RULeg, RLLeg, RUArm, and RLArm), and the torso spine. The results are shown in Table 4 and Fig. 20. In general, bone lengths in E_{joint} strategy are more correct (smaller bias) and more stable (smaller oscillations). For rigid body parts like arms and torso, variations are rather small and the differences are still negligible. For non-rigid body parts e.g., RULeg, the margins become significant (see the fluctuations of red and green solid curves in Fig. 20). We therefore conclude that E_{joint} is a more effective way to realize inverse skinning.

Table 4 Bone length variation on 6 body parts of *Samba* sequence. Bias means the absolute difference between average length during tracking and its original length.

Body part	Initial bone length (m)	Bias (mm)		Standard deviation (mm)	
		E_{bone}	E_{joint}	E_{bone}	E_{joint}
RHip	0.224	12.60	**1.75**	20.91	**7.01**
RULeg	0.377	7.13	**4.84**	27.92	**9.98**
RLLeg	0.454	19.16	**6.45**	21.97	**13.99**
RUArm	0.268	2.49	**1.74**	14.48	**9.18**
RLArm	0.249	5.90	**4.26**	9.33	**4.29**
Torso	0.322	**5.47**	7.22	6.40	**5.19**

7.4 Outlier consideration

In Eq. 27, we introduce an uniform component to model outliers. We are also aware of other alternatives, e.g., estimating outlier likelihood patch-wise and aggregating them [23], or removing the outlier class and replacing Gaussian distribution with heavy-tailed Student's-t distribution which is known to be robust to outliers [54]. However, with the first strategy we did not observe significant improvements, and yet the second strategy requires solving a differential equation at each iteration [35], which brings heavy computation overhead.

With Fig. 21, we study the influence of this expected outlier proportion. In this experiment, seven pairs of consecutive meshes (M_t, M_{t+1}) from the *Free* sequence were considered. Outlier points were distributed around M_{t+1} by duplicating a percentage of its vertices and perturbing them with a Gaussian noise of standard deviation 4 edge length. Then we ran the deformable registration of M_t to M_{t+1} with different values of this parameter. The figure shows the average residual registration error as a function of $e_{outlier}$ and the actual proportion of added outliers. We conclude from this experiment that this parameter doesn't require to be finely tuned and that it simply needs to be non zero to give the optimization enough slack to progressively ignore outliers and converge to a proper solution.

7.5 Computational cost

We list in Table 5 experimental timings on numerous sequences to give an idea of the complexity of the method. These measurements were obtained by looking at times when files were written to the hard-drive.They are only an indication on the computational load of our method and do not constitute a precise performance evaluation. The computational cost is largely dominated by the nearest neighbor search and the sparse linear system solver. The nearest neighbor search is straightforward to be parallelized with GPU. It is expected that with a smarter space partitioning approach or more

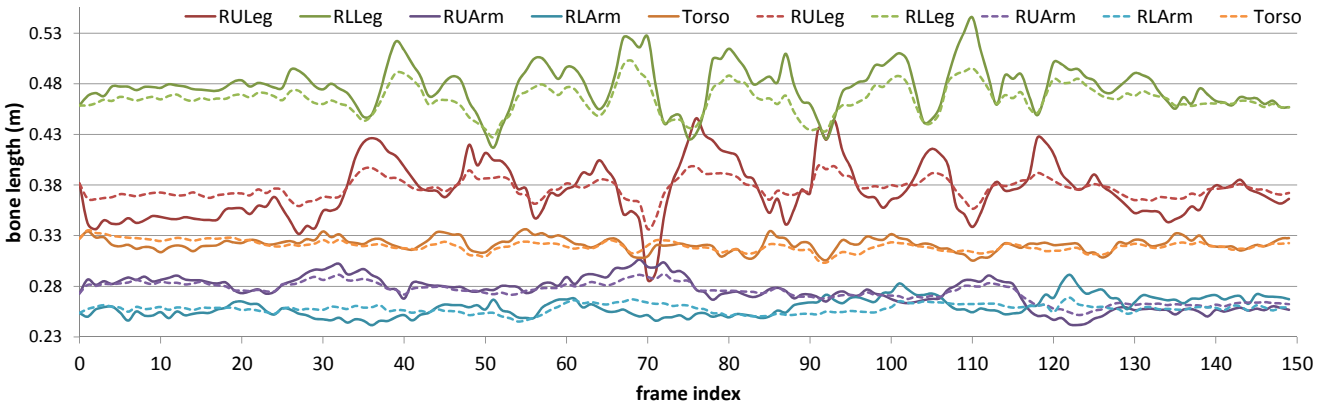


Fig. 20 Bone length variation on 6 body parts of *Samba* sequence. Solid curve: E_{bone}. Dotted curve: E_{joint}.

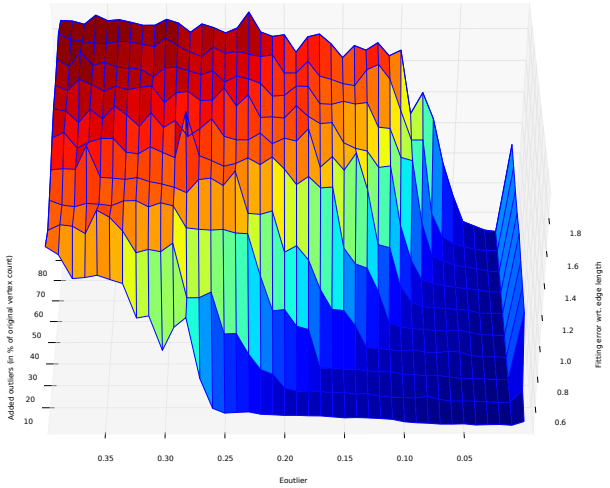


Fig. 21 Average fitting error on 7 random frames of the Free sequence. The x axis represents the expected outlier parameter. The y axis represents the actual proportion of added outliers to the target point cloud. The z axis shows the fitting error with respect to the mean edge length.

Table 5 Average timings on standard sequences for the EM procedure (max. 10 EM steps - without silhouette refinement) with target point clouds of roughly 10k vertices. The CPU implementation was run on a 2.5Ghz quad-core machine. The CUDA implementation was run on a NVIDIA Geforce GTX260.

Sequence	Frame#	Ref. surface vertex#	Average time per frame (sec.)	
			CPU	GPU
Flashkick	200	5445	24	3.60
Head	250	5548	29	3.79
Lock	250	5301	24	3.52
Pop	250	5596	16	3.44
Handstand	174	5939	29	4.11
Bouncing	174	3848	29	3.70
Crane	174	3407	11	2.72
Samba	150	5530	12	2.03

computational resources, this step should be negligible. The remaining bottleneck is therefore the sparse linear solver. Preliminary experiments on the CPU indicate that Conjugate Gradient is a viable alternative to the direct solver we currently use.

Failure cases. Although our framework assumes very little on the nature of the tracked objects, one fundamental premise is that no drastic variations are presented in the topology. The reference frame has to be topologically suitable, that is, it has to be split wherever the surface might split in the sequence. In other words, a moderate amount of disappearing geometry, e.g., self-intersections or close interactions of different subjects as in Fig. 16 can be handled, but it is highly likely to fail when any creation of new geometry occurs.

7.6 Comments on the approach

The prediction mechanism from neighboring patches in searching for associations described in subsection 5.3 is the key to our method, as it encodes multiple hypothesis on the position of the patch. More specifically, it gives a chance to the surface to locally quickly return to its rest pose by propagating the information from correctly registered patches to patches where the current approximated deformations are erroneous.

The i.i.d. assumption that leads to Eq. 33 needs more careful considerations when patches occlude each other. This clearly biases the drawing of samples in the distribution of 3D data. For example, in Fig. 10, when the arms and body are merged, the local density of points in the input data does not double, which clearly indicates that the data generation by two overlapping patches on the arm and the body is not independent. In that sense, our method and Eq. 33 are only approximations.

8 Conclusion

We develop a method to recover the free-form deformation of surfaces from data acquired in a multi-camera setup. Our approach casts the problem as the registration of meshes by iteratively fitting a reference surface to the rest of the sequence. Three contributions differentiate our method from previous work. Firstly, we introduce a generic mesh deformation and numerical optimization framework that enables to process complex scenes involving several deforming objects of unknown nature. This framework defines a coarse control structure for the deformation by splitting a reference surface into elementary elements called patches. These patches allow to express simple physics-inspired rigidity constraints on the surface and provide integration domains on which data terms are smoothed. The second contribution is a Bayesian formulation of mesh registration that builds on the control structure defined by patches, and models for the uncertainty in the input data. Last but not least, we demonstrate two approaches to estimate skeletal poses from surface deformations, such that the above framework is suitable for human motion tracking. We present numerous qualitative and quantitative analysis, confirming that we can recover meaningful deformations in spite of fast motions, large deformations and significant reconstruction artifacts.

Acknowledgements This work was partially funded by Deutsche Telekom Laboratories and partly conducted in their laboratory.

References

1. Ahmed N, Theobalt C, Rössl C, Thrun S, Seidel HP (2008) Dense correspondence finding for parametrization-free animation reconstruction from video. In: IEEE CVPR
2. Baran I, Popovic J (2007) Automatic rigging and animation of 3D characters. In: SIGGRAPH
3. Bay H, Ess A, Tuytelaars T, Gool LV (2008) Surf: Speeded up robust features. CVIU
4. Bishop CM, Nasrabadi NM (2006) Pattern recognition and machine learning, vol 1. Springer
5. Botsch M, Sorkine O (2008) On linear variational surface deformation methods. IEEE Transactions on Visualization and Computer Graphics
6. Botsch M, Bommes D, Kobbelt L (2005) Efficient linear system solvers for mesh processing. In: IMA Conference on the Mathematics of Surfaces
7. Botsch M, Pauly M, Wicke M, Gross MH (2007) Adaptive space deformations based on rigid cells. Comput Graph Forum
8. Cagniart C, Boyer E, Ilic S (2010) Free-from mesh tracking: a patch-based approach. In: IEEE CVPR
9. Cagniart C, Boyer E, Ilic S (2010) Probabilistic deformable surface tracking from multiple videos. In: ECCV
10. Chai J, Xiao J, Hodgins JK (2003) Vision-based control of 3d facial animation. In: In Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation
11. Corazza S, Mündermann L, Gambaretto E, Ferrigno G, Andriacchi TP (2010) Markerless motion capture through visual hull, articulated ICP and subject specific model generation. IJCV 87(1-2)
12. De Aguiar E, Stoll C, Theobalt C, Ahmed N, Seidel HP, Thrun S (2008) Performance capture from sparse multi-view video. In: ACM SIGGRAPH 2008
13. De Aguiar E, Sigal Leonid, Treuille A, Hodgins JK (2010) Stable Spaces for Real-time Clothing. In: SIGGRAPH
14. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the em algorithm. Journal of the royal statistical society, series B
15. Duveau E, Courtemanche S, Reveret L, Boyer E (2012) Cage-based motion recovery using manifold learning. In: 3DimPVT, IEEE
16. Franco JS, Boyer E (2003) Exact polyhedral visual hulls. In: BMVC
17. Furukawa Y, Ponce J (2008) Dense 3d motion capture from synchronized video streams. In: IEEE CVPR
18. Gall J, Stoll C, de Aguiar E, Theobalt C, Rosenhahn B, Seidel HP (2009) Motion capture using joint skeleton tracking and surface estimation. In: IEEE CVPR
19. Gall J, Rosenhahn B, Brox T, Seidel HP (2010) Optimization and filtering for human motion capture. IJCV
20. Guan P, Weiss A, Balan A, Black MJ (2009) Estimating human shape and pose from a single image. In: ICCV, pp 1381–1388
21. Horaud RP, Forbes F, Yguel M, Dewaele G, Zhang J (2010) Rigid and articulated point registration with expectation conditional maximization. IEEE PAMI
22. Huang CH, Boyer E, Ilic S (2013) Robust human body shape and pose tracking. In: 3D Vision
23. Huang CH, Boyer E, Navab N, Ilic S (2014) Human shape and pose tracking using keyframes. In: CVPR
24. James DL, Twigg CD (2005) Skinning Mesh Animations. SIGGRAPH
25. Karypis G, Kumar V (1998) A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput
26. Lewis JP, Cordner M, Fong N (2000) Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In: SIGGRAPH, ACM
27. Li H, Roivainen P, Forcheimer R (1993) 3-D Motion Estimation in Model-Based Facial Image Coding. PAMI
28. Li H, Sumner RW, Pauly M (2008) Global correspondence optimization for non-rigid registration of depth scans. Comput Graph Forum
29. Liao M, Zhang Q, Wang H, Yang R, Gong M (2009) Modeling deformable objects from a single depth camera. In: ICCV
30. Liu Y, Stoll C, Gall J, Seidel HP, Theobalt C (2011) Markerless motion capture of interacting characters using multi-view image segmentation. In: CVPR, IEEE
31. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. IJCV
32. Meng XL, Rubin DB (1993) Maximum likelihood estimation via the ECM algorithm: A general framework. Biometrika
33. Myronenko A, Song X (2010) Point-set registration: Coherent point drift. IEEE PAMI
34. Nealen A, Mueller M, Keiser R, Boxerman E, Carlson M (2006) Physically Based Deformable Models in Computer Graphics. Computer Graphics Forum
35. Peel D, McLachlan GJ (2000) Robust mixture modelling using the t distribution. Statistics and computing 10(4):339–348
36. Popa T, South-Dickinson I, Bradley D, Sheffer A, Heidrich W (2010) Globally Consistent Space-Time Reconstruction.

In: Computer Graphics Forum
37. Rydfalk M (1987) CANDIDE, a parameterized face. Tech. rep.
38. S Toledo (2003) Taucs: A Library of Sparse Linear Solvers, Version 2.2. Tech. rep.
39. Salzmann M, Pilet J, Ilic S, Fua P (2007) Surface deformation models for nonrigid 3d shape recovery. IEEE PAMI
40. Seitz SM, Curless B, Diebel J, Scharstein D, Szeliski R (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms
41. Sigal L, Balan AO, Black MJ (2010) HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. IJCV 87(1):4–27
42. Sigal L, Isard M, Haussecker H, Black MJ (2012) Loose-limbed people: Estimating 3d human pose and motion using non-parametric belief propagation. IJCV 98(1):15–48
43. Sorkine O, Alexa M (2007) As-rigid-as-possible surface modeling. In: Eurographics
44. Sorkine O, Or DC, Lipman Y, Alexa M, Rössl C, Seidel HP (2004) Laplacian surface editing. In: SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing
45. Starck J, Hilton A (2007) Correspondence labelling for wide-timeframe free-form surface matching. In: ICCV 2007
46. Starck J, Hilton A (2007) Surface capture for performance based animation. IEEE Computer Graphics and Applications
47. Stoll C, Hasler N, Gall J, Seidel HP, Theobalt C (2011) Fast articulated motion tracking using a sums of gaussians body model. In: IEEE ICCV
48. Straka M, Hauswiesner S, Rüther M, Bischof H (2012) Simultaneous shape and pose adaption of articulated models using linear optimization. In: ECCV, Springer
49. Sumner RW, Schmid J, Pauly M (2007) Embedded deformation for shape manipulation. In: ACM SIGGRAPH 2007
50. Urtasun R, Fua P (2004) 3d human body tracking using deterministic temporal motion models. In: ECCV, Springer
51. Varanasi K, Zaharescu A, Boyer E, Horaud RP (2008) Temporal surface tracking using mesh evolution. In: ECCV
52. Vlasic D, Baran I, Matusik W, Popović J (2008) Articulated mesh animation from multi-view silhouettes. In: SIGGRAPH
53. White R, Crane K, Forsyth D (2007) Capturing and Animating Occluded Cloth. In: SIGGRAPH
54. Zhou Z, Zheng J, Dai Y, Zhou Z, Chen S (2014) Robust non-rigid point set registration using student's-t mixture model. PloS one 9(3):e91,381