Learning Temporally Consistent Rigidities

Jean-Sébastien Franco, Edmond Boyer

Learning Temporally Consistent Rigidities

Jean-Sébastien Franco Edmond Boyer
LJK, INRIA Grenoble Rhône-Alpes, France
firstname.lastname@inrialpes.fr

Abstract

We present a novel probabilistic framework for rigid tracking and segmentation of shapes observed from multiple cameras. Most existing methods have focused on solving each of these problems individually, segmenting the shape assuming surface registration is solved, or conversely performing surface registration assuming shape segmentation or kinematic structure is known. We assume no prior kinematic or registration knowledge except for an over-estimate k of the number of rigidities in the scene, instead proposing to simultaneously discover, adapt, and track its rigid structure on the fly. We simultaneously segment and infer poses of rigid subcomponents of a single chosen reference mesh acquired in the sequence. We show that this problem can be rigorously cast as a likelihood maximization over rigid component parameters. We solve this problem using an Expectation Maximization algorithm, with latent observation assignments to reference vertices and rigid parts. Our experiments on synthetic and real data show the validity of the method, robustness to noise, and its promising applicability to complex sequences.

1. Introduction

The problem of spatio-temporal modeling from multiple images has gained a lot of attention from researchers in recent years. The topic is becoming increasingly popular for applications related to shape acquisition, shape analysis, thanks to emerging technologies in the industry such as 3D television, entertainment based on 3D and full-body interaction. This is creating an ever increasing demand for robust space-time shape acquisition methods, for offline and real-time 3D content production.

The problem remains challenging for the vision community. Historically shape acquisition has first been addressed in a temporally independent manner, using sets of images of a single instant. This however leaves out an important source of redundancy and improvement in the case of temporal sequences. In recent years various surface deformation and tracking models have become very popular to leverage temporal information of multi-video sequences. A number of such approaches treat surface acquisition as the fitting of a deformable surface model based on image cues or 3D points obtained from stereo or visual hull techniques [10, 12]. This gives positive results with little prior knowledge of the surfaces, on the other hand it leaves out important cues about the scene such as rigidity, as it usually uses only surface smoothness and continuity criteria for space-time reconstruction, yielding generally underconstrained methods. On the other end of the spectrum, one can constrain reconstructions by using specific prior models such as a kinematic structure, through a reduced parameterization of surface movement. This however comes at the cost of genericity, often yielding specific, overconstrained methods that hardly handle the variability of real surface data. To lift this limitation and find a middle ground between genericity and efficient priors, researchers have sought to automatically identify rigid parts and kinematic structures in a pre-processing step. The main trend in the computer vision community has been to deal with the problem in 2D images, where occlusion discontinuity modeling is mandatory, or for the case of 3D point clouds, without a strong

Figure 1. Convergence of the method in fitting frame 30 of Lock sequence (courtesy U. Surrey [15]), using frame 20 as reference (left). Target observation points are shown in blue, color lines show probabilistic associations from observations to reference vertices. The method estimates rigid clusters coherent with both the spatial and motion characteristics of the model, in particular recovering the correct partition and motion of arm and forearm.

modeling of spatial coherence. Meanwhile the main trend in the graphics community has been to focus on kinematic segmentation of 3D surfaces from prominent spatial characteristics of the surface in a static pose [13], with the intrinsic limitations of not observing the actual rigidity in motion.

The method we propose has two contributions with respect to these existing research trends. First we show that segmentation and surface tracking stages need not be separated and can be efficiently performed simultaneously, through the online discovery, adaptation and alignment of rigidities. Second we propose a principled model to take advantage of both the spatial and temporal cues meaningfully for the segmentation (§3). We cast the problem as a Bayesian estimation of spatial and temporal rigid cluster parameters, over a given reference mesh of the observed scene. We assume the surface is subsequently observed through a set of unaligned sparse 3D points obtained from vision systems. We solve the estimation problem using an Expectation Maximization algorithm [7], while estimating distributions over latent variables modeling both the probabilistic segmentation of the surface into rigid parts, and the noisy matching of observed points to the reference mesh (§4). The model automatically estimates the position and temporal transforms of each cluster (§4.1), while dealing with noise and outliers among the 3D observations (§4.2). The main benefits sought with this approach is to simultaneously increase genericity and robustness of methods through common estimation of rigidity segmentation and shape tracking. Implementation details, experimental validation, and future work are discussed in §5, §6 and §7.

2. Related Work

Deformable Surface Tracking Deformable surface tracking is one of the most active trends to tackle the spatio-temporal modeling problem [6, 5, 17, 4, 8]. The topic has been addressed both in the computer vision and graphics communities, either with weakly constrained surfaces not taking advantage of rigidities [6, 8] or including a strong rigidity prior [5, 17], such as an underlying kinematic skeleton, which is not learned but imposed at initialization. The most recent successful approaches opt for a more flexible model of rigidity based on patches [4], in fact casting the problem as an EM, but still leaving aside the problem of rigidity learning. We propose to take this family of approaches one step further, in simultaneously addressing both mesh tracking and segmentation cast as a single, cooperative learning problem.

Shape Segmentation and Matching Static shape segmentation has received considerable attention in the computer graphics community [13], only considering the spatial aspect. They are fundamentally limited for the problem of rigidity detection, as rigidity itself is a dynamic notion. Some recent works in the graphics community have considered the rigid segmentation of dynamic meshes, but mainly with the limitation that the meshes are pre-aligned, e.g [11]. There is a growing interest in both vision and graphics communities for spectral clustering methods [9], sometimes applied to shape segmentation [14], but the emphasis in these methods is more on the continuous matching problem between general static poses than the temporal evolution aspect. A goal of our method is to leverage both the spatial and dynamic cues of spatio-temporal sequences for segmentation and tracking.

Motion Segmentation and Factorization A large body of work exists in the vision community for segmenting motion in 2D images using various cues such as color, and optical flow [3]. This comes with intrinsic image-domain problems, such as occlusions, light perception, with less than ideal image domain specific solutions such as occlusion boundary detection or image domain smoothing. The latter is a way of using spatial coherence, but is best performed in 3D as soon as 3D information or multiple views are available. Indeed 3D rigid segmentation has been addressed in the vision community [16], including the analysis of kinematic chains [18], but mainly for the case of sparse set of 2D feature points, sometimes lifted in 3D through matching, if multiple views are available. Again we see benefit in taking this type of approach further, by using spatial continuity together in surface matching and tracking.

3. Modeling

3.1. Model Overview

We assume an object of interest is observed moving through a set of 3D point clouds, obtained from a vision system. We wish to identify the rigid parts of the object, and their motion over time. Given a reference mesh of the object, these two goals translate to two association problems. First, associate each reference vertex to a rigid part, determining a *rigid segmentation* of the reference mesh. Second, associate each 3D observation at each time t to a vertex of the reference mesh, thus determining the *motion* of reference mesh vertices at time t. A key difficulty here is that both problems are considered simultaneously and intertwined, as shown in the proposed graphical model Fig. 2.

Let $X = \{X_v\}_{v\in\mathcal{V}}$ be the set of 3D coordinates associated to each vertex of reference mesh, with $\mathcal{V}$ the set of vertices. Let $O = \{O_o^t\}_{t\in\mathcal{T},o\in\mathcal{O}^t}$, be the sets of sparse 3D surface points observed, where $\mathcal{T}$ is the time span considered and $\mathcal{O}^t$ the set of observations at time t.

The segmentation problem is governed by a set of parameters $C = \{C_k\}_{k\in\mathcal{K}}$ describing the distribution of rigid parts over the reference mesh, and a set $\boldsymbol{k}_v$ of unobserved *rigidity selection variables* per-vertex v of the reference

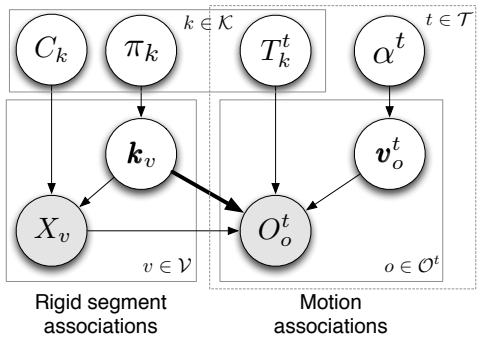

Figure 2. "Naive" model

mesh, describing an association hypothesis of v to a rigid cluster. As a modeling simplification, in this paper we consider that the reference mesh points are to be classified according to their 3D coordinates X_v using a Gaussian Mixture Model, where $C = \{C_k\}_{k \in \mathcal{K}}$ are the parameters (mean and covariance matrices of the GMM), with mixing coefficients π_k. This simple model discriminates rigidities as quasi-convex subvolumes, which is sufficient to validate the method.

The motion problem is parameterized by a set of rigid transforms $T = \{T_k^t\}_{k \in \mathcal{K}, t \in \mathcal{T}}$, for each rigid cluster $k \in \mathcal{K}$ and time $t \in \mathcal{T}$, and a set of discrete *vertex selection* variables $\boldsymbol{v}_o^t$ describing the reference vertex association hypothesis for each observation o at time t. $\boldsymbol{v}_o^t$ can take any value in $\{\mathcal{V} \cup \emptyset\}$, including a garbage association $\emptyset$ for the case where the o'th observation is an outlier. The prior proportion of outliers at time t is noted α^t, to be learned by our model. Importantly, a given observed 3D point O_o^t is treated as the noisy measurement of a reference vertex coordinate X_v, once displaced by the transform $T_{\boldsymbol{k}_v}^t$ associated to that vertex's cluster $\boldsymbol{k}_v$. $\boldsymbol{k}_v$ is thus required to make this prediction, inducing the bold causal link represented in the graph.

3.2. Resolution and Tractability

The full problem can be cast as a likelihood maximization over C_k and T_k^t, given the knowledge of the reference mesh X and all observations O. Because of the presence of latent variables, the solution can be found using Expectation Maximization (EM) [7]. The segmentation and motion association problems are interrelated through the variable group $\boldsymbol{k}_v$: in other words, for a cluster k to have a high likelihood, it needs to cluster spatially consistent points on the reference mesh, while simultaneously predicting a subset of each t's observations from these points *with a single rigid transform* T_k^t. This problem is thus significantly harder to solve than anyone of the association problems alone.

The difficulty materializes for the E-step of the full problem, which needs to compute a distribution over the selection possibilities $\boldsymbol{k} = \{\boldsymbol{k}_v\}_{v \in \mathcal{V}}$ and $\boldsymbol{v} = \{\boldsymbol{v}_o^t\}_{t \in \mathcal{T}, o \in \mathcal{O}^t}$ given the current parameter estimates (i.e. the full posterior $p(\boldsymbol{kv}|XOCT\pi)$). The current model implies that probabilities over rigid associations $\boldsymbol{k}$ and vertex association $\boldsymbol{v}$ are interdependent, because of the bold causal link in Fig. 2: the posterior $p(\boldsymbol{kv}|XOCT\pi)$ doesn't factorize as a product of simpler distributions, and thus yields an intractable E-step. Favorable factorization is identified to occur when the selection variables satisfy the *D-separability* criterion [2], i.e. when a clearly separated subset of selection variables is used to predict separate measurements, as is the case e.g. for simple GMMs.

3.3. Tractable Model

To obtain a tractable EM algorithm, we assign each observation o with a local duplicate of the segmentation problem, as shown in Fig. 3. Thanks to this duplicate, the causal links involving segmentation and motion associations in the model will now be contained as per-observation terms, as opposed to Fig. 2 where observation predictions needed potentially arbitrary rigid segment selection variables $\boldsymbol{k}_v$.

In other words, each observation o is given its own independent set of rigidity selection variables $\{\boldsymbol{k}_o^t\}_{t \in \mathcal{T}, o \in \mathcal{O}^t}$, governed by the same rigidity mixing priors π_k. The method thus partitions not only static reference mesh vertices among rigid GMM clusters, but also the 3D observations themselves through this set of redundant selection variables. The set of reference mesh coordinates is also locally duplicated as $X_{o,v}^t$. We write the predictive distributions over $X_{o,v}^t$ so as to allow only the associated coordinate $X_{o,\boldsymbol{v}_o^t}^t$ of the reference mesh vertex $\boldsymbol{v}_o^t$ to be predicted by the GMM's $\boldsymbol{k}_o^t$'th component. This allows each observation o to contribute an additional sample to the static segmentation GMM, as selected by its selection variables $\boldsymbol{k}_o^t$ and $\boldsymbol{v}_o^t$.

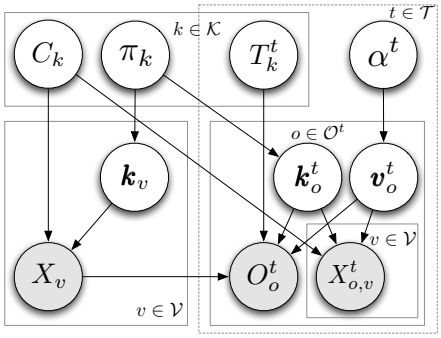

Figure 3. Complete model

For readability, we introduce $\Theta = \{CT\pi\alpha\}$ the set of method parameters to be estimated, $Z = \{\boldsymbol{vk}\}$ the set of latent variables, and $M = \{OX\}$ the set of known variables. We can then write the model likelihood as follows:

$$p(MZ|\Theta) = \prod_{v\in\mathcal{V}} p(\boldsymbol{k}_v|\pi)\, p(X_v|C\,\boldsymbol{k}_v) \tag{1}$$
$$\prod_{o,t} p(\boldsymbol{v}_o^t|\alpha)\, p(\boldsymbol{k}_o^t|\pi)\, p(O_o^t|X_v\, T^t\, \boldsymbol{v}_o^t\, \boldsymbol{k}_o^t) \prod_{v\in\mathcal{V}} p(X_{o,v}^t|C\,\boldsymbol{k}_o^t\,\boldsymbol{v}_o^t)$$

Note that $\prod_{v\in\mathcal{V}} p(X_{o,v}^t|C\,\boldsymbol{k}_o^t\,[\boldsymbol{v}_o^t \neq \emptyset])$ simplifies to $p(X_{o,\boldsymbol{v}_o^t}^t|C\,\boldsymbol{k}_o^t\,[\boldsymbol{v}_o^t \neq \emptyset])$ in this expression, by choosing reference coordinates for $v \neq \boldsymbol{v}_o^t$ to be predicted uniformly in space regardless of C, and thus discarded, which effectively enables to select the $\boldsymbol{v}_o^t$'th coordinate as a rigid GMM sample, as previously mentioned. To account for the outlier selection case, we set $p(X_{o,v}^t|\boldsymbol{v}_o^t = \emptyset) = \mathcal{U}(X_{o,v}^t, S) = \frac{1}{V}$ regardless of other parameter values, that is, uniform spatial prediction over the reference bounding volume.

3.4. Choosing parametric distributions

Selection distributions $p(\boldsymbol{k}_v|\pi)$, $p(\boldsymbol{k}_o^t|\pi)$ and $p(\boldsymbol{v}_o^t|\alpha^t)$. The rigidity selection variables are drawn from their corresponding mixture prior : $p(\boldsymbol{k}_v = k|\pi) = \pi_k$, resp. $p(\boldsymbol{k}_o^t = k|\pi) = \pi_k$. $p(\boldsymbol{v}_o^t|\alpha^t)$ models the expected proportion of outliers in the data governed by α^t: $p(\boldsymbol{v}_o^t = \emptyset|\alpha^t) = \alpha^t$, $p(\boldsymbol{v}_o^t \neq \emptyset|\alpha^t) = \frac{1-\alpha^t}{|\mathcal{V}|} = \bar{\alpha}^t$. Note that there is no outlier class in $\mathcal{K}$: we assume the reference mesh is not noisy, as such all of its vertices should be classified according to static clusters C.

Observed vertex displacement term $p(O_o^t|X\,T^t\,\boldsymbol{v}_o^t\,\boldsymbol{k}_o^t)$. We assume that each $T_k^t = \{R_k^t, \sigma_k\}$, with R_k^t a rigid transformation matrix from the canonical pose to the curent pose. As in other similar methods [4], we assume that the predicted position is perturbed by Gaussian noise. The noise in this case is better modelled as isotropic, as we expect no dominant noise orientation in the way displaced vertices can predict each observed datum. Thus, when $\boldsymbol{v}_o^t \neq \emptyset$:

$$p(O_o^t|X\,T^t\,[\boldsymbol{v}_o^t \neq \emptyset]\,\boldsymbol{k}_o^t) = \mathcal{N}(O_o^t|R_{\boldsymbol{k}_o^t}^t(X_{\boldsymbol{v}_o^t}), \sigma_{\boldsymbol{k}_o^t}^2\mathbf{I}). \tag{2}$$

When $\boldsymbol{v}_o^t = \emptyset$, we use a spatial uniform distribution over $\mathbb{R}^3$ to model outliers predictions (they don't depend on model parameters T and C):

$$p(O_o^t|\boldsymbol{v}_o^t = \emptyset) = \mathcal{U}(O_o^t, V) = \frac{1}{V}, \tag{3}$$

where V is the reference mesh bounding volume.

Mesh classification terms $p(X_v|C\,\boldsymbol{k}_v)$ and $p(X_{o,\boldsymbol{v}_o^t}^t|C\,\boldsymbol{k}_o^t)$. With the GMM parameterization we note $C_k = \{\mu_k, \Sigma_k\}$ over $\mathbb{R}^3$:

$$p(X_v|C\,\boldsymbol{k}_v) = \mathcal{N}(X_v|\mu_{\boldsymbol{k}_v}, \Sigma_{\boldsymbol{k}_v}), \tag{4}$$
$$p(X_{o,\boldsymbol{v}_o^t}^t|C\,\boldsymbol{k}_o^t) = \mathcal{N}(X_{o,\boldsymbol{v}_o^t}^t|\mu_{\boldsymbol{k}_o^t}, \Sigma_{\boldsymbol{k}_o^t}), \tag{5}$$

where μ and Σ are the centroid and covariances modeling the cluster shape. Note that we keep the full covariance matrix description in this case as we expect spatial clusters to have a variety of shapes over the reference mesh, not necessarily circular. Other more complex prediction models could be explored in future work.

3.5. Final likelihood expression

By substituting the different terms, and using an indicator variable $\delta_{\boldsymbol{v}_o^t}$ with value 1 if $\boldsymbol{v}_o^t = \emptyset$ and 0 otherwise, and $\bar{\delta}_{\boldsymbol{v}_o^t} = 1 - \delta_{\boldsymbol{v}_o^t}$, we obtain:

$$p(MZ|\Theta) = \prod_{v\in\mathcal{V}} \pi_{\boldsymbol{k}_v}\, \mathcal{N}(X_v|\mu_{\boldsymbol{k}_v}, \Sigma_{\boldsymbol{k}_v}) \tag{6}$$
$$\prod_{o,t} \{\bar{\alpha}^t\, \pi_{\boldsymbol{k}_o^t}\, \mathcal{N}(X_{o,\boldsymbol{v}_o^t}^t|\mu_{\boldsymbol{k}_o^t}, \Sigma_{\boldsymbol{k}_o^t})\, \mathcal{N}(O_o^t|R_{\boldsymbol{k}_o^t}^t(X_{\boldsymbol{v}_o^t}), \sigma_{\boldsymbol{k}_o^t}^2\mathbf{I})\}^{\bar{\delta}_{\boldsymbol{v}_o^t}}$$
$$\{\alpha^t\, \mathcal{U}(X_{o,v}^t, S)\, \mathcal{U}(O_o^t, V)\}^{\delta_{\boldsymbol{v}_o^t}}$$

4. Inference

Finding the optimal clustering and rigid parameters translates to maximizing the following log likelihood:

$$\Theta^* = \max_{\Theta} \ln p(M|\Theta) \tag{7}$$

The following EM helper function $Q(\Theta|\Theta^j)$ can be defined to solve this goal with the latent variables Z [2], by substitution of the different variable groups (state, latent, observed) in the generic definition:

$$\begin{aligned} Q(\Theta|\Theta^j) &= E_{Z|M\Theta^j}\{\ln p(MZ|\Theta)\} \\ &= \sum_Z p(Z|M\Theta^j) \ln p(MZ|\Theta). \end{aligned} \tag{8}$$

The associated E- and M-steps are as follows:

$$\textbf{E-Step:} \quad \text{Compute } p(Z|M\Theta^j), \tag{9}$$
$$\textbf{M-Step:} \quad \Theta^{j+1} = \max_{\Theta} Q(\Theta|\Theta^j). \tag{10}$$

4.1. E-step updates

Let us analyse the expression of $p(Z|M\Theta^j)$. First let us note that, by construction, the groups of selection variables $\{\boldsymbol{k}_v\}_{v\in\mathcal{V}}$ and $\{\boldsymbol{k}_o^t, \boldsymbol{v}_o^t\}_{o\in\mathcal{O}^t, t\in\mathcal{T}}$ are now D-separable according to the complete graphical model (Fig. 3) and thus independent under this posterior distribution:

$$p(Z|M\Theta^j) = \prod_{v\in\mathcal{V}} p(\boldsymbol{k}_v|M\Theta^j) \prod_{t\in\mathcal{T}, o\in\mathcal{O}^t} p(\boldsymbol{k}_o^t\boldsymbol{v}_o^t|M\Theta^j). \tag{11}$$

The E-step consists in tabularizing these two sets of distributions, seen as functions of $\boldsymbol{k}_v$, and ($\boldsymbol{k}_o^t$,$\boldsymbol{v}_o^t$):

$$\beta(\boldsymbol{k}_v) = p(\boldsymbol{k}_v|M\Theta^j) = \frac{\pi^j_{\boldsymbol{k}_v}\,\mathcal{N}(X_v|\mu^j_{\boldsymbol{k}_v},\Sigma^j_{\boldsymbol{k}_v})}{\sum_{\boldsymbol{k}_v}\pi^j_{\boldsymbol{k}_v}\,\mathcal{N}(X_v|\mu^j_{\boldsymbol{k}_v},\Sigma^j_{\boldsymbol{k}_v})} \quad (12)$$

$$\gamma(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o \neq \emptyset) = p(\boldsymbol{k}^t_o\boldsymbol{v}^t_o|M\Theta^j) \quad (13)$$
$$= \frac{1}{w}\pi^j_{\boldsymbol{k}^t_o}\,\bar{\alpha}^{t,j}_{\boldsymbol{v}^t_o}\,\mathcal{N}(X^t_{o,\boldsymbol{v}^t_o}|\mu^j_{\boldsymbol{k}^t_o},\Sigma^j_{\boldsymbol{k}^t_o})\,\mathcal{N}(O^t_o|R^{t,j}_{\boldsymbol{k}^t_o}(X_{\boldsymbol{v}^t_o}),{\sigma^j_{\boldsymbol{k}^t_o}}^2\mathbf{I}),$$

$$\gamma(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o = \emptyset) = \frac{1}{w}\alpha^{t,j}\,\mathcal{U}(X^t_{o,v},S)\,\mathcal{U}(O^t_o,V), \quad (14)$$

where w is a normalization constant ensuring γ is a distribution over $(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)$. Note that the β-functions correspond to the usual E-step for GMMs, as applied here to reference mesh point clustering, while the γ-functions can be seen as a variant accounting for the transformation coherence and outlier rate, for each pair $(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)$. The state space for each γ-function is quite large, thus we shall compute it sparsely, with the most likely $(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)$ pairs. An algorithm to retrieve the best pairs will be discussed in §5.

4.2. M-step Updates

Substituting the definitions of $\beta(\boldsymbol{k}_v)$ and $\gamma(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)$, the expression of $Q(\Theta|\Theta^j)$ can be written as a sum of terms each involving only one of the maximization variables:

$$Q(\Theta|\Theta^j) = \sum_Z \prod_{v\in\mathcal{V}}\beta(\boldsymbol{k}_v) \prod_{t\in\mathcal{T},o\in\mathcal{O}^t}\gamma(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)\ln p(MZ|\Theta)$$
$$= \sum_v\sum_{\boldsymbol{k}_v}\beta(\boldsymbol{k}_v)\{\ln\pi_{\boldsymbol{k}_v} + \ln\mathcal{N}(X_v|\mu_{\boldsymbol{k}_v},\Sigma_{\boldsymbol{k}_v})\} \quad (15)$$
$$+\sum_{o,t}\sum_{\boldsymbol{k}^t_o,\boldsymbol{v}^t_o\neq\emptyset}\gamma(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)\{\ln\pi_{\boldsymbol{k}^t_o} + \ln\mathcal{N}(X^t_{o,\boldsymbol{v}^t_o}|\mu_{\boldsymbol{k}^t_o},\Sigma_{\boldsymbol{k}^t_o})\}$$
$$+\sum_{o,t}\sum_{\boldsymbol{k}^t_o,\boldsymbol{v}^t_o\neq\emptyset}\gamma(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)\{\ln\bar{\alpha}^t + \ln\mathcal{N}(O^t_o|R^t_{\boldsymbol{k}^t_o}(X_{\boldsymbol{v}^t_o}),\sigma^2_{\boldsymbol{k}^t_o}\mathbf{I})\}$$
$$+\sum_{o,t}\sum_{\boldsymbol{k}^t_o,\boldsymbol{v}^t_o=\emptyset}\gamma(\boldsymbol{k}^t_o,\boldsymbol{v}^t_o)\ln\alpha^t + \text{const.}(\Theta).$$

$Q(\Theta|\Theta^j)$ can be maximized by maximizing separately the group of sum terms corresponding to each of the parameters in $\Theta = \{CT\pi\alpha\}$. Fortunately, most of the updates can be computed in closed form, yielding a set of update equations for the M-step presented below. For concision and clarity, the complete derivations have been omitted in this paper, but will be made available as technical report.

Updating π_k, the mixing coefficients. Defining N and N_k as follows, the π_k update can be written:

$$N = |\mathcal{V}| + \sum_{t\in\mathcal{T}}|\mathcal{O}^t|, \quad (16)$$

$$N_k = \sum_{v\in\mathcal{V}}\beta_v(k) + \sum_{o,t,v\in\mathcal{V}}\gamma^t_o(k,v), \quad (17)$$

$$\pi^{j+1}_k = \frac{N_k}{N}. \quad (18)$$

Updating (μ_k,Σ_k), the spatial cluster parameters. It can be shown that the update equations maximizing (μ_k,Σ_k) are as follows:

$$\mu^{j+1}_k = \frac{1}{N_k}\left(\sum_{v\in\mathcal{V}}\beta_v(k)X_v + \sum_{o,t,v\in\mathcal{V}}\gamma^t_o(k,v)X^t_{o,v}\right), \quad (19)$$

$$\Sigma^{j+1}_k = \frac{1}{N_k}\sum_{v\in\mathcal{V}}\beta_v(k)(X_v-\mu^{j+1}_k)(X_v-\mu^{j+1}_k)^\top \quad (20)$$
$$+\frac{1}{N_k}\sum_{o,t,v\in\mathcal{V}}\gamma^t_o(k,v)(X^t_{o,v}-\mu^{j+1}_k)(X^t_{o,v}-\mu^{j+1}_k)^\top.$$

The form of these updates (including π_k) make sense considering the form of the classical GMM Gaussian update equations, by noting that in fact the samples of our Gaussian updates are given by all reference mesh coordinates X_v on one hand, and the coordinates $X^t_{o,v}$, as selected through the various vertex and rigidity matching hypotheses for each observation, on the other hand. This explains the form of our mean and covariance updates, a weighted combination of these terms in two summation groups.

Updating α^t, the outlier rate for each time step. Setting the partial derivative to 0 for each α^t, we obtain:

$$\alpha^{t,j+1} = \frac{1}{|\mathcal{O}^t|}\sum_{o,k}\gamma^t_o(k,\emptyset), \quad (21)$$

which corresponds quite naturally to summing the (already normalized) weights of outlier observation assignments for each time step considered.

Updating T^t_k, the transform parameters. Obtaining the transform parameters for each instant and rigid part can be obtained by maximizing the corresponding terms in $Q(\Theta|\Theta^j)$ with respect to R^t_k :

$$R^{t,j+1}_k = \max_{R^t_k}\sum_{o,v\in\mathcal{V}}\gamma^t_o(k,v)\ln\mathcal{N}(O^t_o|R^t_k(X_v),\sigma^2_k\mathbf{I})$$
$$= \max_{R^t_k}\sum_{o,v\in\mathcal{V}}\gamma^t_o(k,v)\|O^t_o - R^t_k(X_v)\|^2. \quad (22)$$

The latter can be recognized as a weighted orthogonal Procrustes problem. Only a subset of pairs $(k,v)\in\mathcal{P}$ yield non-zero $\gamma^t_o(k,v)$ values. We collect these values in a size-$|\mathcal{P}|$ diagonal matrix W, and collect the X_v and O^t_o coordinates appearing in the corresponding sum terms in (22) in two $3\times|\mathcal{P}|$ matrices $\mathbf{x}$ and $\mathbf{o}$. The rotational and translational components $(R\ \mathbf{t})$ of $R^{t,j+1}_k$ can then be retrieved using the SVD of a 3×3 matrix defined as follows:

$$\hat{\mathbf{x}} = \mathbf{x}W - \bar{\mathbf{x}}, \quad (23)$$
$$\hat{\mathbf{o}} = \mathbf{o}W - \bar{\mathbf{o}}, \quad (24)$$
$$R = VU^\top \text{ with the SVD of } \hat{\mathbf{x}}\hat{\mathbf{o}}^\top = UDV^\top, \quad (25)$$
$$\mathbf{t} = \bar{\mathbf{o}} - R\bar{\mathbf{x}}, \quad (26)$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{o}}$ are reweighted, zero-centered versions of $\mathbf{x}$ and $\mathbf{o}$.

Updating the rigidity variances σ_k. To update σ_k, we differentiate (22) w.r.t. σ_k^2, obtaining:

$$\sigma_k^{j+1\,2} = \frac{\sum_{o,v} \gamma_o^t(k,v) \|O_o^t - R_k^{t,j+1}(X_v)\|^2}{\sum_{o,v} \gamma_o^t(k,v)}. \quad (27)$$

5. Implementation

5.1. Computing sparse γ-tables

To compute each γ table sparsely for a given observation o, we need to identify the best set of hypotheses (k, v). γ can be seen as a function attributing a matching weight to each vertex-cluster pair. To retrieve the best candidate matches, we solve a series of K-Nearest Neighbor problems, by constructing a search space with $|\mathcal{K}| \times |\mathcal{V}|$ candidates, corresponding to each (k, v) hypothesis, i.e. each reference vertex transformed with each cluster transform. Each candidate is normalized using the respective cluster covariances to retrieve nearest neighbors in the sense of Mahalanobis distances. Then, for each observation in $\mathcal{O}^t$, we look for the K-nearest neighbors where K is a supplied parameter (in practice fixed to 5 or 10), to keep the K-best γ values. Nearest Neighbor structures thus need to be built only once per EM iteration. Note that as a result, every update sum and expression involving (k, v) can be sparsely evaluated.

5.2. Processing Time Sequences

The proposed framework can be used to process sequences of input observations with proper incremental initialization. Because EM converges to local minima and is initialization dependent, the framework is best used in sequence, starting from a position close to the reference mesh. With such starting conditions, the algorithm easily finds correct solutions, with a simple initialization strategy for all experiments, e.g. Fig. 1. We initialize cluster parameters with centroids on a randomly selected reference mesh vertex, with covariance matrices randomly generated with the order of magnitude of the bounding box. Cluster transforms are initialized to identity, and mixing coefficients to uniform. Sequences can then be processed using a chosen sliding window of size W, where each new observation set entering the window is initialized with the transforms and cluster parameters computed at the previous time step with some random perturbation. We obtain promising results with these simple strategies; arguably better initializations could be explored in future work.

6. Experimental Results

We validate the approach using two synthetic datasets and several real datasets, also shown in the supplemental video[1]. Between datasets, only values of the sliding window size and expected number of rigidities $|\mathcal{K}|$ are to be set manually, although strategies to infer $|\mathcal{K}|$ could be applied from existing EM literature in future work. Best and more stable results were obtained by estimating identical variances for all transforms T. The computed parameters and E-step tables encode a continuous mesh deformation defined by the spatial clusters C, the β-tables which encode class probabilities with respect to C, and the transforms T. This deformed position can be computed for each reference vertex v as the expectancy of its transformed position under $\beta(\boldsymbol{k}_v)$, or $\sum_{\boldsymbol{k}_v} \beta(\boldsymbol{k}_v) X_v$, where β-tables are analog to skinning weights. Computation times on a single 2.53Ghz-core range from 100msec per iteration for simple datasets (CYLINDER) to a second per iteration (LOCK). The number of iterations before convergence is usually 10-40, with occasionally higher numbers (80-100) for the larger configuration changes within a sequence.

6.1. Validation on synthetic datasets

We validate the approach on synthetic datasets to evaluate performance of the method under controlled conditions. The first dataset used is a deformable cylinder with 1300 vertices, comprised of three rigid subcomponents (Fig. 4) folding over 22 frames. The first joint folds over frames $1-10$, then both joints fold subsequently. We process the dataset with a purely vertical reference position, so as to test the ability of the method to retrieve rigid parts unbiased

[1] See http://hal.inria.fr/inria-00583131/en

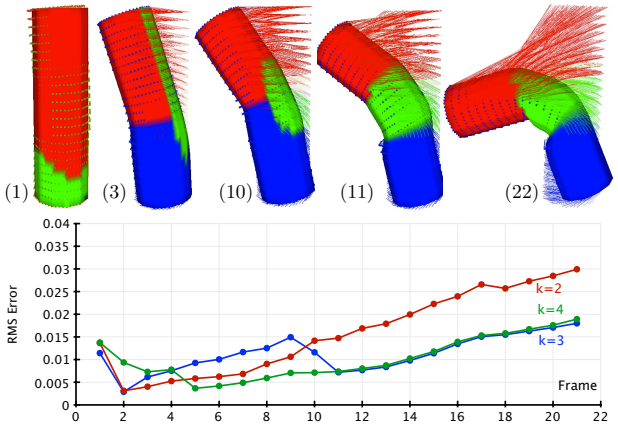

Figure 4. Various frames of the cylinder dataset. Mesh vertices are colored according to the most likely rigidity according to its β table. Color lines are drawn from each observation to most probable matching reference positions and rigidity hypothesis (color) according to its γ-table. Observation points are shown in blue.

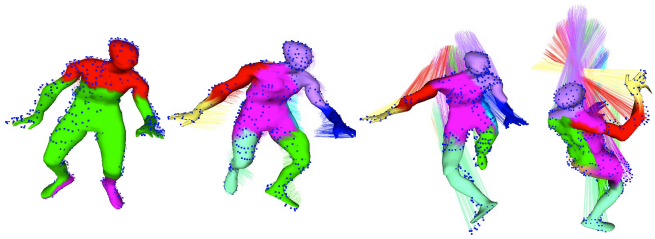

Figure 5. Frames 1,5,20 and 35 of the dancer synthetic dataset.

by the reference geometry. The sequence shown in Fig. 6 is processed with $|\mathcal{K}| = 3$ clusters and a sliding window of size $W = 3$, although similar results have been produced with $W = 1$ and $W = 5$. RMS errors between observations and expected reference mesh positions given the model, have been shown for various values of k. In frame (1) the cylinder has barely deformed and the method broadly uses 2 of the rigidities, RMS error is relatively high at initialization. From frame (2)-(9), the method places the red and blue clusters correctly and puts the third cluster in the axis of the folding on frame (10), which is a plausible solution reducing error in the fold. From frame (11) to (22), the method properly identifies the three rigid segments. Note how the RMS error drops from frame (11) on as soon as the new joint is solicited, when setting $k \geq 3$: this shows that the method did indeed detect the new rigidity upon occurrence, reducing positional error with respect to using only two rigidities. We use a second synthetic sequence with 200 frames, DANCER, to test the longer term behavior of the method. The sequence is tracked and segmented using 10 rigid parts. The human model in the sequence has 3000 vertices, only 1000 of which are samples randomly at each frame to simulate missing data. Despite the scarce number of parts, the method adapts the location of rigidities to observed degrees of freedom being solicited (e.g. a single rigid movement per leg often is sufficient to model the deformation). The method was able to track the model up to frame 120, despite large body motions and rotations such as shown frame 35 in Fig. 5.

6.2. Real datasets

We have processed various real datasets, LOCK (Fig. 1), BALL, DOG, and CRANE (Fig. 6). These mesh sequences were obtained from vision systems using visual hulls (BALL, DOG) or silhouettes and photoconsistency methods (CRANE, LOCK)[15]. All datasets use a sliding window of size $W = 2$, keeping observations 5 frames apart for better coverage of observations on different poses. In all sequences the reference mesh used was the first model in the sequence, reduced to 3000 vertices for faster computation. Observations point clouds were constructed by extracting 1000 random unaligned points from sequence meshes, to illustrate method resilience to sparser data. All sequences were processed with $k = 15$ except for ball which was processed with $k = 10$. It was observed during experiments that the method performs better with lower rigidity counts: the convergence is better constrained in those cases. The method is able to process about 20 to 30 frames in motion for each sequence before significant artifacts occur. It finds rigid segmentations plausible with the shape and motion observed in the sequences, including arms, legs, forearms and forelegs for all sequences where they are observed moving. For example in the BALL sequence in Fig. 6(a), the main moving parts are the adult's upper body, and both person's forearms, which is consistent with the segmentation found. Results show that outliers are identified, such as the shadow volume in the BALL sequence or the small ball tossed in the DOG, which was not part of the reference mesh.

7. Limitations and Future Work

The method yields very encouraging results. Further improvement of the method is still possible. First the number of rigidities could be themselves learned with various strategies. The algorithm could easily monitor rigidity mixing weights and assignments, and their contribution to the total log-likelihood, to remove, split or merge rigidities. Second, although the simple initialization strategy used works in most cases, it doesn't guarantee systematic escapes of local maxima of the log-likelihood function, which sometimes translates to suboptimal segmentations being transferred from one frame to another. A hierarchical strategy could be explored to improve convergence properties. Third, terms related to surface regularization could be added. The fact that clusters softly partition the reference mesh into rigidities, does yield a weak form of surface regularization; this is not always enough to prevent folds and loss of tracking on complex sequences. Specific additional terms related to surface and inter-cluster regularization could be added to improve long-term robustness of the method. Finally, our proof-of-concept implementation is single-core, but virtually all update equations involve only individual variables and could be massively parallelized (GPU), bringing this method within reach for real-time applications.

8. Discussion

We have presented a novel learning technique to simultaneously address spatio-temporal mesh tracking and rigid segmentation. The framework developed is based on a principled graphical model and EM maximization algorithm, which performs unsupervised matching of unaligned observed point clouds sequences to a reference mesh, detects outliers, while segmenting the reference mesh and computing its rigid parameters. Resolution is done with simple, efficient and highly parallelizable EM updates. We are con-

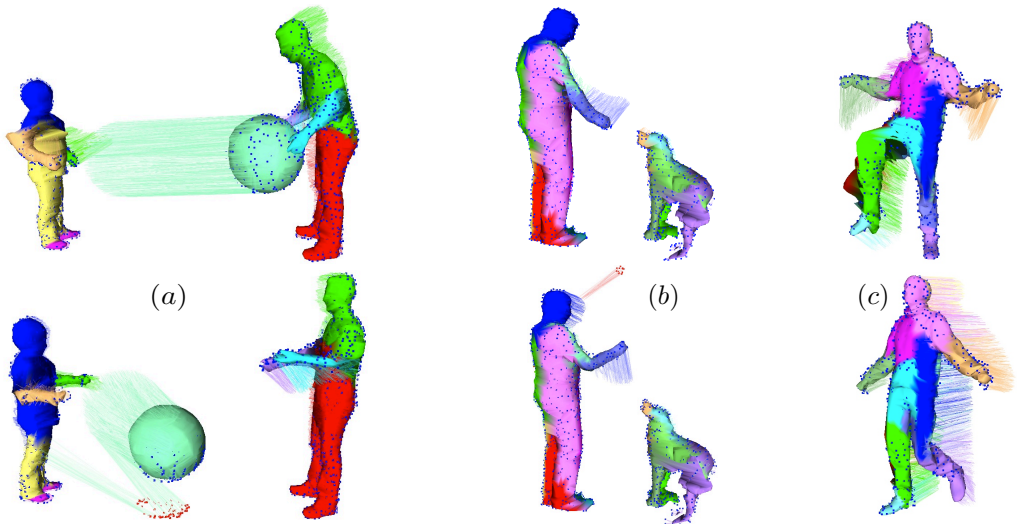

Figure 6. (a) BALL and (b) DOG sequence, courtesy 4drepository.inrialpes.fr. [1] (c) CRANE sequence, courtesy CSAIL-MIT [17]. Observation points are shown in blue if inlier, red if outlier (such as points on the ball's shadow in (a) or the ball tossed by the person in (b)). The method sometimes exhibits artifacts such as top (c) on the leg, due to an occasionally misfitted rigidity.

fident that the framework opens a new direction to cooperatively segment and track spatio-temporal mesh sequences.

9. Acknowledgements

This work has been partially funded by grant ANR-10-BLAN-0206 of the French National Research Agency.

References

[1] J. Allard, J.-S. Franco, C. Ménier, E. Boyer, and B. Raffin. The grimage platform: A mixed reality environment for interactions. In *ICVS*, jan 2006.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *ECCV*, volume 6315 of *Lecture Notes in Computer Science*, pages 282–295. Springer, 2010.

[4] C. Cagniart, E. Boyer, and S. Ilic. Probabilistic deformable surface tracking from multiple videos. In *ECCV*, pages 326–339, 2010.

[5] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun. Performance capture from sparse multi-view video. In *SIGGRAPH*, 2008.

[6] E. de Aguiar, C. Theobalt, C. Stoll, and H.-P. Seidel. Marker-less deformable mesh tracking for human shape and motion capture. In *CVPR*, pages 1–8, 2007.

[7] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. In *Journal of the Royal Statistical Society: Series B*, 1977.

[8] Y. Furukawa and J. Ponce. Dense 3d motion capture from synchronized video streams. In *CVPR*, 2008.

[9] V. Jain, H. Zhang, and O. van Kaick. Non-rigid spectral correspondence of triangle meshes. *International Journal on Shape Modeling*, 13(1):101–124, 2007.

[10] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162, 1994.

[11] T.-Y. Lee, Y.-S. Wang, and T.-G. Chen. Segmenting a deforming mesh into near-rigid components. *Vis. Comput.*, 22:729–739, September 2006.

[12] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, pages 519–528, 2006.

[13] A. Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27(6):1539–1556, 2008.

[14] A. Sharma, E. von Lavante, and R. Horaud. Learning shape segmentation using constrained spectral clustering and probabilistic label transfer. In *ECCV*, volume 6315, pages 743–756, 2010.

[15] J. Starck and A. Hilton. Correspondence labelling for wide-timeframe free-form surface matching. In *ICCV*, 2007.

[16] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2007.

[17] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. *ACM Trans. Graph.*, 27(3):1–9, 2008.

[18] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and nondegenerate. In *ECCV*, pages 94–106, 2006.