Cage-based Motion Recovery using Manifold Learning

Estelle Duveau, Simon Courtemanche, Lionel Reveret, Edmond Boyer

Cage-based Motion Recovery using Manifold Learning

Estelle Duveau Simon Courtemanche Lionel Reveret Edmond Boyer
LJK, INRIA Grenoble Rhône-Alpes, France

Abstract

We present a flexible model-based approach for the recovery of parameterized motion from a sequence of 3D meshes without temporal coherence. Unlike previous model-based approaches using skeletons, we embed the deformation of a reference mesh template within a low polygonal representation of the mesh, namely the cage, using Green Coordinates. The advantage is a less constrained model that more robustly adapts to noisy observations while still providing structured motion information, as required by several applications. The cage is parameterized with a set of 3D features dedicated to the description of human morphology. This allows to formalize a novel representation of 3D meshed and articulated characters, the Oriented Quads Rigging (OQR). To regularize the tracking, the OQR space is subsequently constrained to plausible poses using manifold learning. Results are shown for sequences of meshes, with and without temporal coherence, obtained from multiple view videos preprocessed by visual hull. Motion recovery applications are illustrated with a motion transfer encoding and the extraction of trajectories of anatomical joints. Validation is performed on the HumanEva II database.

1. Introduction

Motion recovery using multiple view environments has known important developments in Computer Vision and Graphics. Applications to biomechanics and character animation require a reliable and interpretable measurement of motion. In that respect, the recovery of articulated motion appears as a relevant goal since the underlying skeleton provides a deterministic interpretation of motion. As a preprocess, it is now common to calibrate multiple views to obtain streams of 3D meshes from visual inputs such as silhouettes typically using space carving or visual hull [4]. The resulting information from visual hull is not temporally structured yet nor related to interpretable parameters in terms of motion. In contrast to existing methods for mesh tracking that use the coupling of an articulated skeleton and skinning to deform a known template, e.g. [20, 19], we propose here to relax the strong parametric constraint of the skeleton by making use of barycentric coordinates. In barycentric coordinates, a low polygonal version of the reference template, the *cage*, controls the deformation of the mesh by interpolation. We use Green Coordinates for their known properties of good conformal behavior [13]. The output is thus a sequence of temporally coherent coordinates of the vertices of the cage, controlling the deformation of a mesh template and constrained to lie on a pose manifold. To fulfill the goal of motion recovery, the cage has been carefully designed to follow a morphologically standardized representation of the body shape. It can thus be typically converted to an articulated skeleton representation.

For most methods, the accuracy of articulated tracking may still be challenged by lighting conditions, poor camera positioning or occlusions. One of the key factor is therefore the ability to bring an efficient regularization to the tracking problem. Latent spaces have been successfully used to constrain the manifold of plausible poses for human body tracking from monocular video or pose editing in animation. In this paper we investigate a similar strategy but in the novel context of motion recovery as a postprocess of 3D reconstruction of meshes from multiple view silhouettes. The direct link between the morphological cage and an articulated skeleton allows the use of available databases of motion capture data for the automatic learning of the manifold.

Our main contribution is thus the derivation of a mesh tracking algorithm, based on a Green Coordinates encoding, and regularized by a manifold learned whether by hand labeling of poses or automatically from motion capture data. We build on the formalism introduced by Prisacariu and Reid [14][15] for a rigorous derivation of the gradients of the object measurement with respect to the space of latent variables. We also provide the exact formulation of the gradients. We demonstrate our approach on different human performances : on temporally coherent clean meshes for consistency test [20], on the HumanEva II dataset [17] for objective evaluation and on non temporally coherent meshes obtained from a personal set-up adapted to visual hull for biometrics test.

2. Related Work

2.1. Motion recovery from a sequence of meshes

For mesh tracking, feature-level methods such as the patch-based approach of [2] offer a maximum flexibility. This approach is robust to large changes in the topology of the analyzed scene as no assumption is made on the internal structure but it does not fulfill the objective of motion recovery as no interpretable parametrization can be directly derived from this formulation. Following the parametrization encoded by our normalized cage, we deliver a more structured output. Furthermore, in case of very noisy input meshes, an approach that is too local risks providing unreliable results with non recoverable drift. This effect appears clearly on the HumanEva dataset [17], where the camera locations are not optimal for a preprocess step using visual hull reconstruction, leading to noisy input meshes.

Taking an opposite direction to feature-level approaches, the skeleton-based methods impose a strong prior on the parameter space. This approach has shown successful results such as in [20] and [8]. In both cases, a skeleton of a human or animal character has to be aligned on a reference template mesh. A skinning algorithm deforms the mesh, which is matched against 2D silhouettes. We propose here an alternative method for the deformation phase using Green Coordinates [13] and a 'cage' proxy to control the shape. The skinning algorithm is known to be sensitive to a careful tuning of vertices weights, where the automatic phase usually needs for manual adjustment. The vertices weights automatically provided by the Green Coordinates method show superior stability and do not require further tuning.

Tracking mesh data using a cage for barycentric-coordinates has first been proposed by [16] and recently improved by [18]. Unlike these works, we make no assumption on the temporal coherence of the input 3D data. Another major difference is in the regularization strategy. In [16], the cage deformation was constrained by a Laplacian smoothing. It has been reported that it did not prevent the cage from collapsing when tracking long sequences. This is our motivation for investigating a more robust regularization strategy using manifold learning.

2.2. Manifold learning for human motion tracking

The success of manifold learning for robust tracking of human motion from monocular videos has been typically proved with works such as [19] which uses GPLVM. For animation, an application of manifold learning has also been shown for 3D pose editing [9]. Other methods for animation editing using low dimensional subspace embedding exist. Among recent works, [3] uses different encodings for shape, pose and time. However, it requires an animation sequence as input whereas our learning samples are few and sparse. In our case, we therefore use a similar to [9] manifold learning approach, but in the novel context of cage-based animation from a sequence of meshes. We build on [14], who uses GPLVM to segment 2D images, by using a non-linear manifold as a deformation prior to track a cage-based animation. In particular, closed-form solutions of gradients are provided for a straightforward implementation of a simple optimization based on gradient descent.

The rest of the paper is organized as follows : in section 3, we describe how our template is defined and deformed. In section 4, we explain the learning procedure to obtain a latent space of coherent motion. In section 5, we detail latent space-based tracking. Section 6 shows results obtained with our method and evaluation.

3. Deformation Model

For our model-based approach, we need a shape model and a deformation model. Our shape model is a 3D mesh $\mathcal{M}$. To deform this mesh, we use a cage-based approach as it has better properties in terms of shape preservation and smoothness than skinning. In addition, it makes the motion parameters independent on the actual model as the same cage is used for every specific human model. Our template surface is defined as a mesh embedded in a coarse bounding mesh. The template mesh is called the *model*, the coarse bounding mesh is called the *cage*. The barycentric coordinates used to generate the model from the cage are given by the Green coordinates [13]. As mentioned in [13], unlike other barycentric coordinates, Green coordinates are empirically shown to be quasi-conformal in 3D. There are no requirements on the model : the only information used is the position of its vertices. Green Coordinates behave properly if there are no self-intersections in the cage when computing the coordinates. However, once the coordinates are computed, self-intersecting deformations of the cage do not create artefacts. Model vertices are expressed as a linear combination of cage vertices $\mathcal{V} = (v_1 \dots v_n)$. Therefore a point P of the model is deformed according to :

$$P \mapsto \sum_{i=1}^{n} \phi_i(P) v_i + \sum_{j=1}^{m} \psi_j(P) n(t_j) \tag{1}$$

where $(t_1 \dots t_m)$ are the faces of the cage, $n(t_j)$ is the normal of face t_j and $\{\phi_i\}_{i=1\dots n}$, $\{\psi_j\}_{j=1\dots m}$ are the Green coordinates.

Being a coarse approximation of the model, the cage is easy to deform. However, it lacks an underlying structure to intuitively generate and deform it in a volume and shape-preserving way. Works such as [1] allow the user to interact with the cage using a small number of position and orientation constraints. However, how to generate the cage remains an issue. Inspired by skeletal joints, in order to provide a standard way of building and deforming the cage, we define oriented 3D quads located at main joints. These quads

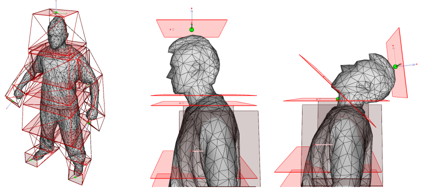


Figure 1: The Oriented Quads Rigging (OQR): template model and example of deformation.

have 6 degrees of freedom (3 in translation, 3 in rotation) to enforce a structure. Their location corresponds to the interface between rigid areas of the model, intuitively, where joints would be located in joint-based animation.

Each oriented quad has a constant size which is not modified during the motion. The cage is then built procedurally from the vertices defined by the quads. Deforming the model then consists in setting the translation and orientation of each quad, which procedurally defines the vertices of the cage. The topology of the cage is computed once by defining relations between quads. The geometry of the cage is updated every time the OQR is modified with the positions of the vertices of the quads. For the sake of brevity, details could not be included in the present paper, which focuses on motion recovery. Finally, this cage defines the vertices of the model by linear combination of the vertices and normals of the cage (see Equation (1)).

As a result, the number of degrees of freedom of the deformation is greatly reduced. From 3 times the number of vertices of the model, we reduce it to 6 times the number of quads. In the example on Figure 1, it is reduced from 3×10002 to 6×21. This approach can be interpreted as an intermediate representation of an articulated character, between a hierarchical skeleton bind to a mesh using skinning and a standard non-hierarchical mesh. For the remainder of the paper, we refer to this representation as Oriented Quads Rigging (OQR).

4. Learning a manifold of deformations

Thanks to the constant size of the quads, some unrealistic deformations such as shrinking the circumference of an arm are avoided. However, nothing preserves the length of the limbs or non physiological configuration. The learning of a non-linear manifold of the possible positions and orientations of the quads allows to enforce such constraints. As in [14], where Gaussian Process Latent Variable Models are used to segment 2D images, we use GPLVM to learn a manifold of deformations. A Gaussian Process Latent Variable Model (GPLVM) is a low dimensional representation of the original data called the latent space. Each data point y_i of dimension d can be represented by a latent point x_i of dimension q, where q is smaller than d. Conversely, every

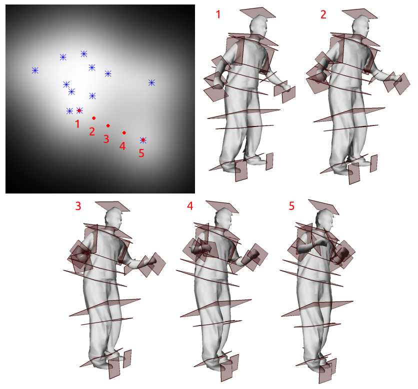


Figure 2: Example of 2D-latent space and corresponding deformations. The gray level corresponds to the variance at a given point in latent space. Blue crosses are the latent coordinates of the learned data-set. Red points and their corresponding reconstructions show that the latent space is suitable for interpolation between learned poses.

point x in the latent space projects to a data point through a Gaussian distribution defined by its mean μ and its variance σ^2. As derived in [11], for a data set of size n, we have :

$$P(Y|X,\theta) = \prod_{i=1}^{n} \mathcal{N}(y_i|0, K(\theta)) \tag{2}$$

where $Y = [y_1 \dots y_n]$, $X = [x_1 \dots x_n]$ and $K_{i,j} = k(x_i, x_j)$ with k being the covariance function defined by the hyper-parameters θ. For instance, if k is a radial basis function, as in our case, it is written :

$$k(x_i, x_j) = \alpha exp(-\frac{\gamma}{2}(x_i - x_j)^T(x_i - x_j)) \tag{3}$$

and the hyper-parameters are α and γ. To learn the GPLVM, Equation (2) is maximized with respect to both X and θ.

In our case, each data point y_i describes one deformation by the positions and orientations of the quads in the OQR. We align all the shapes with respect to the position and orientation of the quad at the pelvis. We then learn the latent variables and hyper-parameters by maximizing Equation (2). An example of a 2D-latent space can be seen on Figure 2. We impose back-constraints so that points close in data space stay close in latent space, which is not guaranteed in the usual formulation of GPLVM [12].

5. Tracking

To recover the motion, we must fit the model to the current representation of the shape, namely the *target*, which is, in our case, a 3D mesh. To do so, we derive a differentiable energy that measures how well the model fits the target, the model being the template mesh controlled by the

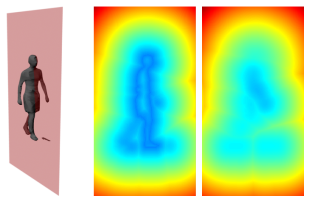


Figure 3: Example of distance fields - model and slicing plane (left), corresponding slice of signed distance field (right) as well as another slice in the sagittal plane.

cage. We differentiate this energy in the latent space to recover the deformation of the model. By doing so, we stay in the learned manifold i.e. in the space of plausible shapes.

5.1. Distance between two meshes

We make no assumptions on the temporal coherence of the target meshes. To measure how well the model mesh fits the target mesh, we thus need to measure the distance between two different meshes having potentially two different topologies. As we will minimize this distance, we need a function which is differentiable with respect to the model mesh. To that end we use a distance field built such that the target mesh is its zero level-set. Distance fields have the advantages of being easy and fast to compute. They are differentiable everywhere and can be used to measure the distance between meshes of different topologies.

Other approaches exist to deal with distances between non-corresponding meshes. One can first estimate vertex-to-vertex sparse correspondences, then use those pairs of vertices to compute the distance between the two meshes. However, computing correspondences is costly and can fail. Another approach is to directly use a mesh to mesh distance, such as the Hausdorff distance. However, such distances are often hard to compute and can not be differentiated because it implies finding optima. We thus use a distance field to avoid these issues [10]. Figure 3 shows examples of slices of distance fields that we obtain on noisy visual hulls.

Once the distance field is computed, we can define the distance function that has to be minimized. For a given mesh $\mathcal{M}$, the distance to the target $\mathcal{T}$ is computed as the sum of the distances of each vertex of $\mathcal{M}$ to $\mathcal{T}$. The distance of a vertex to $\mathcal{T}$ is the value of the distance field at the voxel to which the vertex belongs. Thus the distance function takes the following form :

$$d_{DF}(\mathcal{M}, \mathcal{T}) = \sum_{P_i \in \mathcal{M}} d_{DF}(P_i, \mathcal{T}) \tag{4}$$

where $d_{DF}(P_i, \mathcal{T})$ is the value of the distance field generated from $\mathcal{T}$ at position P_i.

5.2. Shape Optimization

To recover the deformation of the model, we could minimize the distance between the model and the target in the unconstrained highly-dimensional OQR space. To limit the evolution of the model to plausible deformations, we differentiate this distance in the latent space. As a result, we stay in the low-dimensional learned manifold i.e. in the space of plausible poses.

We define an energy function to minimize with two terms. The first one measures how close the model and the target are. This is given by their distance as defined in Section 5.1. The second term takes advantage of the fact that at each point in the latent space, we can compute the variance which indicates how likely this latent point is to generate a valid data point. We use this variance as a regularization term to ensure that the shape generated from the latent point is likely to exist. We therefore try to find :

$$\hat{x} = \arg\min_x (d_{DF}(\mathcal{M}(x), \mathcal{T}) - \log(\sigma^2(x))) \tag{5}$$

where x is the latent point, $\mathcal{M}(x)$ is the model generated from the latent point x and σ^2 is the variance at point x.

We do this by differentiating this energy with respect to the latent variables. This is done by using the chain rule :

$$\frac{d(d_{DF}(\mathcal{M}, \mathcal{T}))}{dx} = \frac{d(d_{DF}(\mathcal{M}, \mathcal{T}))}{d\mathcal{M}} \frac{d\mathcal{M}}{d\mathcal{V}} \frac{d\mathcal{V}}{dy} \frac{dy}{dx} \tag{6}$$

where $\mathcal{V}$ is defined in Section 3 as the vertices of the cage and y is the deformed shape generated from x i.e. the set of positions and orientations of the oriented quads.

$\frac{d(d_{DF}(\mathcal{M}, \mathcal{T}))}{d\mathcal{M}}$ is computed by finite differences.

$\frac{d\mathcal{M}}{d\mathcal{V}}$ is computed by differentiating Equation (1) with respect to the vertices of the cage.

Each vertex in $\mathcal{V}$ is connected rigidly to one and only one oriented quad. $\frac{d\mathcal{V}}{dy}$ follows trivially.

y is the mean of the Gaussian distribution by which the latent point x projects to a data point :

$$y = Y^T K^{-1} K_x \tag{7}$$

where K is the covariance in-between all the learned latent points $X = [x_1 \dots x_n]$ and K_x is the covariance between x and all the learned latent points. Differentiating Equation (7) gives :

$$\frac{dy}{dx} = Y^T K^{-1} \frac{dK_x}{dx} \tag{8}$$

Each element of K_x is the covariance function k applied on x and one learned latent point. As a result, $\frac{dK_x}{dx}$ follows from Equation (3).

As for the regularization term, x projects to a data point with a mean as expressed in Equation (7) and a variance σ^2 such that :

$$\sigma^2 = k(x, x) - K_x^T K^{-1} K_x \tag{9}$$

As a result :

$$\frac{d\sigma^2}{dx} = \frac{dk(x,x)}{dx} - 2K_x^T K^{-1} \frac{dK_x}{dx} \tag{10}$$

Thanks to Equations (6) and (10), a simple first order method such as gradient descent can be used to minimize Equation (5) and recover the deformation of the model.

5.3. Pose Optimization

So far, we have captured the deformation of the shape as encoded in the latent space. As explained in Section 4, this shape is defined up to the global transformation Γ of the pelvis. As a result, we also need to recover the global translation and orientation of the model.

To do so, we minimize the energy without the regularization term by differentiating Equation (4) with respect to the pose parameters :

$$\frac{d(d_{DF}(\mathcal{M}, \mathcal{T}))}{d\Gamma} = \frac{d(d_{DF}(\mathcal{M}, \mathcal{T}))}{d\mathcal{M}} \frac{d\mathcal{M}}{d\Gamma} \tag{11}$$

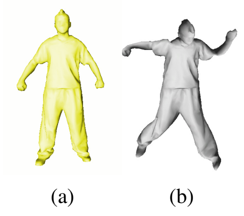
Pose and shape optimization can be done either jointly or iteratively until convergence. Joint optimization is more correct but converges more slowly as the search space is larger. Empirically, iterating between pose and shape optimization works faster and better, especially when tracking a sequence. Indeed, the rigid transformation and the deformation between two frames are small enough that optimizing first for the pose does not lead to a pose from which finding the good shape is impossible, and vice-versa.

6. Experiments and Discussion

We tested our method on three test sets. We first tested our approach on a temporally coherent mesh of high quality. We then tackled more challenging data by tracking a sequence of noisy visual hulls, without temporal coherence, learning the motion manifold automatically from the CMU Motion Capture Database (http://mocap.cs.cmu.edu/). This was first evaluated on the HumanEva II dataset [17]. We finally used it for gait analysis. All experiments have been done with the deformation model of Figure 1. It consists in a rigging of 21 oriented quads, which generates a cage of 86 vertices and 166 faces.

6.1. Evaluation on a temporally coherent sequence

We first tested our deformation optimization without the pose recovery. To do so, we use a bouncing sequence. The sequence of target meshes is made of a temporally coherent animated mesh of 10000 vertices and 20000 faces. We define the template mesh as one of these meshes simplified to 4000 vertices and 8000 faces. The training was done by manually locating the oriented quads on 16 frames of the sequence. We learn a 10-D manifold from these samples. Figure 4 shows the progression of the gradient descent towards



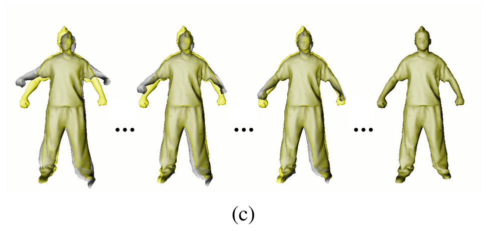


Figure 4: (a) Target model - (b) Initial pose - (c) Samples on the convergence trajectory.

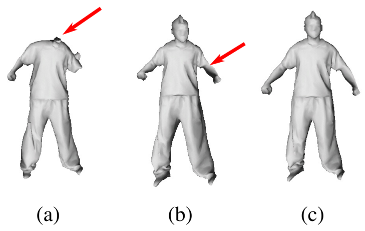


Figure 5: With the same target model and initial pose as in Figure 4 - (a) Result of the optimisation in the oriented quad space. Notice how the head collapses - (b) Result of the optimisation in the 10-D latent space without any regularisation term. Notice how the arms shorten - (c) Result with the regularisation term.

an optimal deformation. We can notice that, since we are optimizing in the latent space, each of the sample deformations is a valid deformation. Figure 5a shows the advantage of tracking in the manifold. As there is no structure on the cage, optimizing with respect to the oriented quads creates a shrinking of the cage as the quads are attracted to the closest patch of surface without any influence from the position and orientation of the other oriented quads. We can see on Figure 5 the influence of the regularization term of the energy. Without it, the model mesh is attracted to the shape that is the closest to the target mesh and can be generated from the latent space without any regard for the likelihood of this shape. Using the variance as a regularization term forces the optimal latent point to be more likely to generate a valid deformation.

As explained in Section 5.3, we also recover the pose. In practice, we noticed that pose recovery and deformation re-

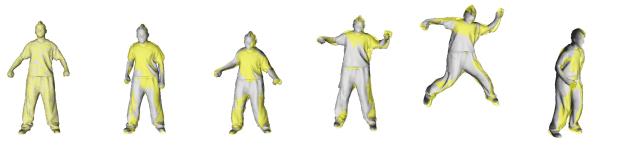


Figure 6: Optimisation of both pose and deformation. The tracking (gray) is overlaid over the target (yellow).

covery done iteratively works better than joint optimization. Figure 6 shows results on the bouncing sequence.

6.2. Automatic learning on CMU database and validation on the HumanEva II dataset

Automatic learning : Instead of creating the training data by manually fitting the OQR to meshes, we also propose a method that uses existing databases of movements. From walking sequences in the CMU Motion Capture Database (http://mocap.cs.cmu.edu/), we automatically generate 120 training samples by rigidly mapping the quads of the OQR to the skeleton. It has to be done only once for a given frame of one of the subject's performance. After that, more samples can be automatically added to the training set using any available motion capture data samples for this subject. This training data is then used to build a 2D-manifold.

HumanEva II : We use the walk sequence of subject S4 in the HumanEva II dataset [17] for a quantitative evaluation. This data consists of 4 synchronized color cameras. Using the background subtraction software and the mixture of Gaussian distribution model provided with the dataset, we extracted the silhouettes from the videos. From these silhouettes, we generated the visual hulls using the Exact Polyhedral Visual Hull method [6]. As the scene was captured with 4 cameras facing each other, the visual hulls are a very coarse approximation of the observed shape as can be seen on Figure 7a. The model used for cage-based deformation is the mesh model for subject S4 generated by Stefano Corazza and the Stanford BioMotion Group using [4].

Local adjustment : One limitation of our method is its dependence on the training data. As with every manifold-based method, we are limited to what the manifold can reconstruct. To account for this limitation, we can first optimize with respect to latent variables to get as close as possible to the target mesh. Once this has converged, a few more iterations can be done with respect to the oriented quads to locally adjust the cage by differentiating the energy function with respect to the oriented quads :

$$\frac{d(d_{DF}(\mathcal{M},\mathcal{T}))}{dy} = \frac{d(d_{DF}(\mathcal{M},\mathcal{T}))}{d\mathcal{M}} \frac{d\mathcal{M}}{d\mathcal{V}} \frac{d\mathcal{V}}{dy} \quad (12)$$

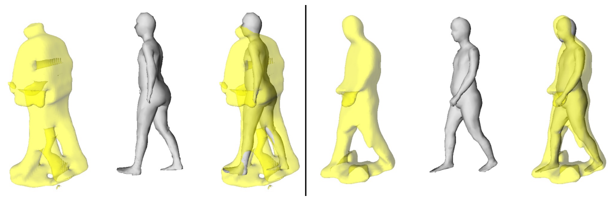


Figure 7: Examples of tracking on the HumanEva II dataset : (from left to right) visual hull, result and overlay of the model on the visual hull. (a) The location of the cameras can lead the visual hull to be a very coarse approximation of the observed shape. (b) When the visual hull quality increases, the tracking becomes more precise.

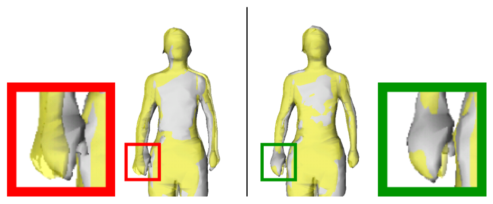


Figure 8: Example of local adjustment of the oriented quads - (a) before local adjustment - (b) after local adjustment.

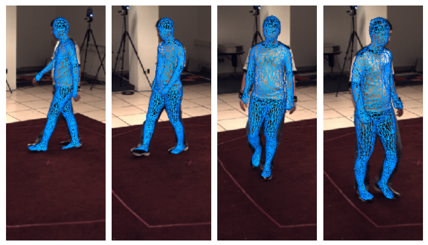


Figure 9: Reprojection on camera 3 of the tracked model for frames 5, 45, 102 and 148 of the sequence for subject S4 in the HumanEva II dataset.

An example of local adjustment is shown in Figure 8, where a 2D latent space does not capture enough variance to track the mesh properly. Table 1 shows the 3D error as specified in the HumanEva II dataset with and without the local adjustment for the first 150 frames. As our approach is not skeleton-based, the positions of the joints are computed by cage-based animation. On the first frame where motion capture data is available i.e. where the positions of the joints are known, the Green Coordinates are computed on these positions and then used to compute the new positions of the joints when the cage is deformed. With local adjustment, the mean error drops from 90mm to 38.6mm.

Figure 9 shows reprojections of the model tracked on the

Method	Frames	Mean error (mm)	Median error (mm)	Standard deviation (mm)
Corazza [5]	1-150	80.0		13.0
Gall [7]	2-1258	32.01		4.53
Our method - no local adjustment	1-150	90.0	87.3	24.1
Our method	1-150	38.6	39.2	4.1
Our method	1-300	85.2	55.8	55.8

Table 1: Error for sequence of subject S4 of the HumanEva II dataset. The mean, median and standard deviation in mm is measured for the first 150 frames without (third row) and with (fourth row) the local adjustment. The error increases after frame 150 (bottom row) as the quality of the visual hulls decreases (Figure 7a) to reach the levels of tracking without local adjustment.

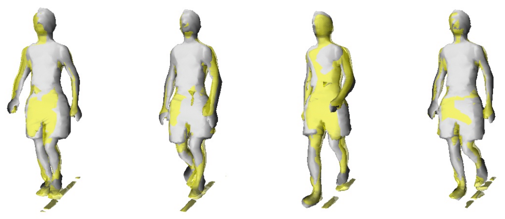


Figure 10: Pose and shape recovery - The tracking (gray) is overlaid over the target (yellow). The training data is built automatically from the CMU motion capture database and the target meshes are independent visual hulls.

sequence. We refer to the video for the whole sequence. For the whole walking sequence, the error is 85.2mm (Table 1). However, when considering only the first 150 frames, where the visual hulls are of better quality (Figure 7b), the error drops to 38.6mm. The error for the whole sequence is about the same as the error for the first 150 frames without local adjustment. This can be explained by the non-discriminative quality of the target mesh. Indeed, the coarseness of the visual hull (Figure 7a) prevents any fine adaptation of the shape by local adjustment. Table 1 also reports the results found in [5] and [7]. The method of Corazza et al. [5] also takes as input a series of visual hulls. The results are limited to the first 150 frames as re-initialisation is necessary after that due to the poor quality of the visual hulls. Thanks to the use of a manifold of deformations, even though the error increases after frame 150, we are still able to attain good results without re-initialisation. To our knowledge, Gall et al. [7] report the best results. However, they also use information not taken into consideration in this paper such as the appearance of the model.

6.3. Applications

Motion analysis: For gait analysis, walking and jogging sequences are selected in the CMU motion capture database

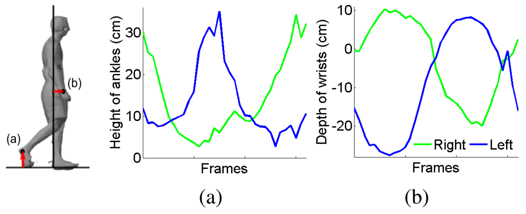


Figure 11: Application : motion analysis - (a) Absolute height of the right and left ankles - (b) Position relative to the pelvis in the sagittal plane of the right and left wrists.

to learn a 10-D manifold for locomotion as explained in Section 6.2. The same subject is recorded performing walks and jogs at various speeds. Visual hulls are extracted from these sequences using [6]. The target meshes are visual hulls of about 7000 vertices and 15000 faces with holes and noise. We use one of those visual hulls cleaned and remeshed to 2500 vertices and 4500 faces as a model mesh. Examples of tracking are shown on Figure 10. From this sequence, we can extract locomotion cycles. Figure 11 shows the height of the feet and the swinging motion of the arms during a cycle. Because of the noise from the treadmill that can be seen on Figure 10, the trajectories of the feet are slightly noisier that those of the hands. We refer to the video for the results on varying speeds of walks and jogs.

Motion transfer : Once we have captured our parameterized performance, we can put any model we want in the cage to obtain a new animation (*motion transfer*). We can edit the original shape or put a texture on it as there is no drift in vertices position. This process is fully automatic as the computation of the Green Coordinates is automatic and gives smooth deformations. On the contrary, skinning usually requires user intervention to correct skinning weights, which makes the process of changing the model more tedious. Figure 12 shows an example of the transfer of the tracking shown in Figure 6 to a different 3D model.

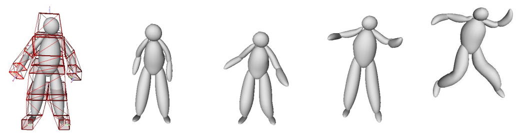


Figure 12: Application : motion transfer from Figure 6.

7. Conclusion

In this paper, we tackle the problem of motion recovery from a sequence of 3D meshes. To this end, we have proposed a method using a cage-based deformation constrained by manifold learning. The use of a manifold regularizes the tracking problem, leading to a more robust solution. We use oriented quads as an underlying structure for cage-based animation. This gives an intuitive way to generate and deform cages, the Oriented Quads Rigging (OQR). On top of that, the use of cage-based animation allows for the tracking of any model, not just models with an intrinsic skeletal structure. Experimental results demonstrate the validity of our approach. We have evaluated our method on temporally coherent data and tested its usability by generating the training data from an existing motion capture database. However, our current method has not considered the fact that classic GPLVM struggles with learning a good model for complex datasets and becomes intractable for large datasets. Using a stochastic approach as in [21] may overcome these issues and lead to a better tracking on complex sequences.

Acknowledgments This work has been partially funded by grant ANR-2010-BLAN-0206-01 MORPHO of the French National Research Agency.

References

[1] M. Ben-Chen, O. Weber, and C. Gotsman. Variational harmonic maps for space deformation. *ACM Trans. Graph.*, 28(3):34:1–34:11, July 2009. 2

[2] C. Cagniart, E. Boyer, and S. Ilic. Probabilistic deformable surface tracking from multiple videos. In *ECCV*, 2010. 2

[3] T. J. Cashman and K. Hormann. A continuous, editable representation for deforming mesh sequences with separate signals for time, pose and shape. *Computer Graphics Forum*, 31(2):735–744, 2012. Proceedings of Eurographics. 2

[4] S. Corazza, E. Gambaretto, L. Mundermann, and T. Andriacchi. Automatic generation of a subject-specific model for accurate markerless motion capture and biomechanical applications. *Biomedical Engineering, IEEE Transactions on*, 57(4):806 –812, april 2010. 1, 6

[5] S. Corazza, L. Mndermann, E. Gambaretto, G. Ferrigno, and T. Andriacchi. Markerless motion capture through visual hull, articulated icp and subject specific model generation. *International Journal of Computer Vision*, 87:156–169, 2010. 10.1007/s11263-009-0284-3. 7

[6] J.-S. Franco and E. Boyer. Exact polyhedral visual hulls. In *British Machine Vision Conference (BMVC'03)*, volume 1, pages 329–338, Norwich, Royaume-Uni, 2003. 6, 7

[7] J. Gall, B. Rosenhahn, T. Brox, and H.-P. Seidel. Optimization and filtering for human motion capture. *International Journal of Computer Vision*, 87:75–92, 2010. 10.1007/s11263-008-0173-1. 7

[8] J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel. Motion capture using joint skeleton tracking and surface estimation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1746 –1753, june 2009. 2

[9] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović. Style-based inverse kinematics. *ACM Trans. Graph.*, 23:522–531, Aug. 2004. 2

[10] M. Jones, J. Baerentzen, and M. Sramek. 3d distance fields: a survey of techniques and applications. *Visualization and Computer Graphics, IEEE Transactions on*, 12(4):581 –599, july-aug. 2006. 4

[11] N. D. Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Machine Learning Research*, 6:1783 – 1816, 2005. 3

[12] N. D. Lawrence and J. Quiñonero Candela. Local distance preservation in the gp-lvm through back constraints. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 513–520, New York, NY, USA, 2006. ACM. 3

[13] Y. Lipman, D. Levin, and D. Cohen-Or. Green coordinates. *ACM Trans. Graph.*, 27:78:1–78:10, August 2008. 1, 2

[14] V. Prisacariu and I. Reid. Nonlinear shape manifolds as shape priors in level set segmentation and tracking. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2185 –2192, june 2011. 1, 2, 3

[15] V. A. Prisacariu and I. Reid. Shared shape spaces. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2587 –2594, nov. 2011. 1

[16] Y. Savoye and J.-S. Franco. Cage-based Tracking for Performance Animation. In *ACCV'10 :the Tenth Asian Conference on Computer Vision*, Queenstown, New Zealand, 2010. 2

[17] L. Sigal, A. O. Balan, and M. J. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *Int. J. Comput. Vision*, 87(1-2):4–27, Mar. 2010. 1, 2, 5, 6

[18] J.-M. Thiery, J. Tierny, and T. Boubekeur. Cager: From 3d performance capture to cage-based representation. In *ACM SIGGRAPH 2012 - Talk Program*, 2012. 2

[19] R. Urtasun, D. J. Fleet, and P. Fua. 3d people tracking with gaussian process dynamical models. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, pages 238–245, Washington, DC, USA, 2006. IEEE Computer Society. 1, 2

[20] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics*, 27(3):97, 2008. 1, 2

[21] A. Yao, J. Gall, L. van Gool, and R. Urtasun. Learning probabilistic non-linear latent variable models for tracking complex activities. In *Proceedings of Neural Information Processing Systems (NIPS)*, Granada, Spain, December 2011. 8