Motion Pattern Preserving IK Operating in the Motion Principal Coefficients Space

Schubert R. Carvalho
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Virtual Reality Lab
CH-1015 Lausanne, Switzerland
schubert.carvalho@epfl.ch

Ronan Boulic
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Virtual Reality Lab
CH-1015 Lausanne, Switzerland
ronan.boulic@epfl.ch

Daniel Thalmann
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Virtual Reality Lab
CH-1015 Lausanne, Switzerland
daniel.thalmann@epfl.ch

ABSTRACT

We introduce a constraint-based motion editing technique enforcing the intrinsic motion flow of a given motion pattern (e.g., golf swing). Its major characteristic is to operate in the *motion* Principal Coefficients (PCs) space instead of the *pose* PCs space. By construction, it is sufficient to constrain a single frame with Inverse Kinematics (e.g., the hitting position of the golf club head) to obtain a motion solution preserving the motion pattern style. Such an approach also reduces the size of the Jacobian matrix as the motion PCs space dimension is small for a coordinated movement (e.g. up to 9 for the golf swing illustrating this paper). We compare this approach against a per-frame prioritized IK method regarding performance, expressivity and simplicity.

Keywords

Motion Editing, Principal Coefficients, Principal Component Analysis, Inverse Kinematics.

1 INTRODUCTION

Technological advances in motion capture techniques make it possible to provide high quality motions for computer animation. However, the captured animations almost always attend to specific needs. Editing and reusing these motions in new situations (e.g., retargeting to new characters or to new environments) became an increasing area of research. Unfortunately, making little adjustments into a good motion may introduce some undesirable artifacts or discontinuities, that can modify the intrinsic style of the motion. Hence, providing ways to preserve the characteristics of the original animation and to impose continuity on the edited animation are the major challenges of motion editing techniques.

In this paper, we present a new method for motion editing. As in traditional per-frame constrained-based approaches the user can make adjustments to a character with direct manipulation, for example repositioning the character's end-effectors. However, two important differences arise between our approach and traditional per-frame constrained-based techniques. First, our method considers just the frame that should be edited, instead of a range of frames around the edited one. Second, to compute the new end-effector's position the system takes into account the style or characteristics of the original animation during optimization.

Although, traditional constrained-based techniques uses just the information of the input motion, we use a database of motion captured animations (not upper body cleaned) of the same behavior of the edited one. These motions are used to estimate a set of Principal Components (PC) and Principal Coefficients (PCs) [Jol86]. The PCs space is used to model the characteristics of the edited animation. The PC space is used for two purposes: 1) to bound the inverse kinematics (IK) solutions closer to the pose space of the edited animation; and 2) to reduce the size of the Jacobian matrix to invert. Our experiments indicate that our approach preserves the characteristics of the edited motion and impose continuity between frames just constraining the pose that should be edited. We also assess the quality of the edited animations against a traditional per-frame prioritized IK method [CB06].

The next section reviews related work. Section 3 provides the motivation and the overview of our approach. Section 4 describes our approach for motion editing in more details. Section 5 describes our motion normalization schema. Then, we illustrate our method with results in section 6. Finally, we conclude this paper by discussing current limitations of our method and possible extensions in section 7.

2 RELATED WORK

2.1 Motion Editing

Constrained-based techniques [Gle01] enables the user to specify constraints over the entire motion or at specific times, while editing an animation. An inverse kinematics (IK) solver computes the "best"motion, pose by pose, satisfying all these constraints. Lee and Shin [LS99] introduced the per-frame plus filtering class of motion editing techniques. Continuity between frames was maintained with a B-Spline filter. Kulpa et. al. [KMA05] proposed a motion adaptation methodology. Their method do not use any kind of filtering, so it provides very good results for real-time motion adaptation of virtual characters. To ease the adaptation, the underlying skeleton is divided into groups, each of which being an individual kinematic chain. This has the advantage to ease the computation of a solution. However, as no synergy exists between groups, it may lead to unrealistic results. Le Callennec and Boulic [CB06] proposed an off-line interactive motion editing technique with prioritized constraints. The characteristics of the original motion are preserved adding the difference between the motion before and after editing as the lowest priority constraint. Splines curves were used to impose smoothly varying constraints. Recently, Liu [LZqDMSH06] proposed a motion editing technique with collision avoidance to prevent the character's limbs to interpenetrate the body during editing. Displacement maps techniques and a Kalman filter were used to preserve the similarities between the edited and original motions, and to impose continuity between frames.

All these techniques do not explicitly model the characteristics of the original motion, mainly because style is difficulty to formulate. Our approach uses Principal Component Analysis(PCA) [Jol86] to represent style in the low-dimensional space of PCs. We also provide an IK solver, to treat geometric constraints, able to preserve motion patterns (e.g., the golf swing characteristics) during optimization, and a different approach to impose continuity between frames.

2.2 Principal Component Analysis

PCA is used in Computer Animation to treat different kinds of problems. Alexa and Müller [AM00] used PCA to represent animations as a set of principal animation components. Glardon [GBT04] used the PCA to synthesize walking, running and jumping motions into a low-dimensional space. Following this idea, Safanova et. al. [SHP04] proposed a motion synthesis framework able to synthesize new motions by optimizing a set of constraints in the low-dimensional space of the PCA. They used IK, just in the characters limbs, to prevent undesirable artifacts such as foot sliding. Urtasun et. al. [UGB+04] proposed a style based motion synthesis framework able to produce motions with different styles by extrapolating the PCs parameters. Urtasun and Fua [UF04] also used the PCA to improve 3D body tracking. Their formulation is similar in spirit to ours because they need to evaluate the PCs that describes a tracked motion style. They relaxed the overconstrained problem of tracking multiple effectors for each frame by allowing the Principal Coefficients to evolve overtime too. On the contrary, we relax the problem by constraining only one frame for which the constraint is particularly easy to specify, e.g. hitting frame in golf. This reduces computation time as the solution is searched in the Principal Coefficients space, which dimension is small compared to the posture space.

3 MOTIVATION AND OVERVIEW

The motion editing framework proposed in this paper is motivated by the following question: how to edit a motion by constraining just one pose without adding discontinuities. We investigated this problem by considering three issues: first, we need to mathematically represent the characteristics of the edited animation; second, we need a constraint formulation that takes into account these characteristics during optimization; third, we need to impose continuity between frames considering the characteristics of the edited animation.

Our aim for motion editing is to take a motion that is good, but is in need of some adjustments, e.g., the motion has the basic format that we want. This is almost always the case of captured motions, where simple adjustments should be made [Gle97]. Now, consider one adjustment: at a specific time of the motion, there is a new position that should be achieved. Consider Figure 8 as an example. We have a good golf swing motion, but we want that the golf club head hits the ball onto another position. We can associate a constraint to the golf club head, to change its position, and compute the new motion.

A common technique used to address such problems is inverse kinematics. IK gives the possibility to edit a motion by dragging the character's end-effector to a new position. When highly articulated figures are considered, IK algorithms are iterative. They try to find the best solution, for some set of constraints, iteratively minimizing some residual error (section 4.2). IK is also the core of constraint-based motion editing techniques [Gle01]. However, when IK is used to deform a motion frame by frame, motion continuity can be broken [Gle97]. Instead of using traditional techniques (i.e., splines curves [LS99]), we proposed a new approach (section 4.3) by exploiting the low-dimensional parametrization of a set of motions with the same behavior as the edited one (sectio 4.1).

The present framework handles geometric constrains, such as position or rotation of end-effectors. Geometric constraints are more intuitive because they directly specify a goal for a specific end-effector [CB06]. Constraints such as position of the center of mass are not specified,

because the solution is searched within the PC space of physically balanced motions.

4 A MODEL FOR EDITING

4.1 Motion Parametrization

In this section, we recall how a motion can be represented by PCA. The parameters of the PCA can be easily estimated from captured animations of people performing a specific activity (see, section 5 for more details). Let us define a pose, Θ, as a state vector describing the global position and orientation of the root node, and a set of joint angles:

$$\Theta = [\theta_1, P_1, \theta_j, ..., \theta_n] \tag{1}$$

where, P_1 and θ_1 represent the 3D global position and 3D orientation of the root node in the sacroiliac level and θ_j is the local transformation of the j^{th} joint expressed using exponential map formulation [Gra98].

A motion can be represented as a normalized motion vector Ψ of dimension: $D = n * N$. Where $N = 132$ is the total number of poses and $n = 93$ is the number of degrees of freedom of the skeleton including the 3D global position and 3D orientation of the root node. Ψ is a column vector of the form:

$$\Psi = [\Theta_{t_0}, ..., \Theta_{t_k}, ..., \Theta_{t_1}]^T, \qquad 0 \leq k \leq 1 \tag{2}$$

where, Θ_{t_k} is a posture corresponding to the normalized time t_k (see, section 5 for more details).

Given the motion database, we compute the mean motion $\Psi_\circ$ and the covariance matrix $C_{M\times M}$, where $M = 10$ is the size of the motion database. The eigenvectors of the covariance matrix are computed and a linear map is constructed to obtain the principal components $\mathbf{E}_{i,1\leq i\leq M}$ [Jol86]. Assuming this set of examples to be representative for a golf swing motion, any motion vector Ψ can be approximated as a linear combination of the mean motion and the PC:

$$\Psi \approx \Psi_\circ + \sum_{i=1}^{m} \alpha_i \mathbf{E}_i \tag{3}$$

where, $\alpha = (\alpha_1, ..., \alpha_m)$ are the Principal Coefficients that characterize the motion and $m \leq M$. m represents the number of PC required to reconstruct a desired percentage σ of the database. This percentage is defined as:

$$\sigma = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{\mathbf{M}} \lambda_i} \tag{4}$$

where λ_i is the eigenvalue corresponding to the $\mathbf{E}_i$ eigenvector, ordered in decreasing order. The parameters of the linear model are estimated only once off-line, before the editing process starts.

4.2 Constraining a Single Posture in the Motion PC Space

In this section, we demonstrate how standard IK methods can be adapted to our needs. If we define the pose, including the root node, of a virtual character as a n-dimensional vector (joint space, Eq. 1) and the position of the end-effector, $\mathbf{x}$, as a p-dimensional vector (task space), the inverse kinematics function can be defined as:

$$\Theta = f^{-1}(\mathbf{x}) \tag{5}$$

for a redundant manipulator, i.e., $n > p$, the problem is always ill-posed (there is not unique solution) and even if $n = p$ multiple solutions can exist [Cra86]. Hence, IK algorithms should determine just one solution to Eq. 5 given many possibilities. In the presence of redundant manipulators IK can be formulated as the solution of an optimization problem:

$$\mathbf{g} = \mathbf{x}(\Theta) \tag{6}$$

where $\mathbf{x}(\Theta)$ and $\mathbf{g}$ are p-dimensional vectors expressed in the task space. The task function $\mathbf{x}(\Theta)$ represents the position or orientation of an end-effector frame while $\mathbf{g}$ is the goal to be reached.

One approach to solve IK problems, with optimization criteria, is the well known Resolved Motion Rate Control (RMRC) proposed by [Whi69]. The RMRC computes an optimal change, $\Delta\Theta$, to Θ, for a small change, $\Delta\mathbf{x}$, in $\mathbf{x}$. The final pose can be found adding $\Delta\Theta$ with respect to Θ ($\Theta = \Theta + \Delta\Theta$). To use the RMRC technique we need to compute the Jacobian matrix of the forward kinematics function, $\mathbf{x} = f(\Theta)$, and invert it:

$$\Delta\Theta = J(\Theta)^{-1}\Delta\mathbf{x} \tag{7}$$

where, $J(\Theta) = \partial\mathbf{x}/\partial\Theta$ is the Jacobian matrix of size $(p\times n)$. To solve Eq. 7 in this way $J(\Theta)$ should be square $(n = p)$ and non-singular. But, for a redundant manipulator this is not the case. The damped pseudo-inverse technique can be used deal with this problem [Mac90]:

$$\Delta\Theta = J(\Theta)^{\dagger^{\xi}}\Delta\mathbf{x} \tag{8}$$

where, $J(\Theta)^{\dagger^{\xi}}$ is the damped pseudo-inverse. The damped factor, ξ, is added to prevent Jacobian's singularities and to stabilize IK solutions [Mac90]. Eq. 8 can be solved by Singular Value Decomposition (SVD) exploring the null space of the Jacobian matrix. However, this Jacobian is not able to restrict the solutions to the motion pattern space, e.g., the space of golf swing motions, because it searches for solutions into the complete joint space.

We alleviate this problem by exploiting the PC space restricted to the single frame we want to constraint. This approach reduces the Jacobian's size and consequently restricts the solution to the motion pattern space. For

example, as we want to preserve the golf swing characteristics as much as possible, we restrict the solution in the space of similar poses.

We underconstrain the problem by considering just the parameters corresponding to a specific normalized time, t_k. The pose to constrain can be represented as a function of the Principal Coefficients and the normalized time:

$$\Theta_{t_k} = Z\left(t_k, \alpha_1, .., \alpha_m\right) \tag{9}$$

So, to compute the new pose configuration for the displacement $\Delta\mathbf{x}$, we have:

$$\Delta\alpha = J(\alpha)^{\dagger^{\xi}}\Delta\mathbf{x} \tag{10}$$

where, $J(\alpha) = \partial\mathbf{x}/\partial\alpha$ is the Jacobian matrix, of size $(p \times m)$, relating the task increment with the α_i coefficients. Once, we have the new increment $\Delta\alpha$, the updated pose can be found computing Eq. 3 just for the normalized time of the constrained pose. The Jacobian, $J(\alpha)$, can be easily computed with the chain rule:

$$\left[\frac{\partial\mathbf{x}_l}{\partial\alpha_i}\right] = \left[\frac{\partial\mathbf{x}_l}{\partial\theta_j}\right] \cdot \left[\frac{\partial\theta_j}{\partial\alpha_i}\right]_{t_k} \tag{11}$$

The derivatives of $\left[\frac{\partial\mathbf{x}_l}{\partial\theta_j}\right]$ can be fast and easily computed [Bae01]. Besides, θ_j are linear combinations of the $\mathbf{E}_i$ principal components, so $\left[\frac{\partial\theta_j}{\partial\alpha_i}\right]_{t_k}$ is simply extracted from the jth coordinate of $\mathbf{E}_i$ [UF04] for the time t_k. We denote this Jacobian as $J(\mathbf{E})_{t_k}$ its size is only $n \times m$, to be compared to the much larger size $D \times m$ of the PC basis. The size of the Jacobian $J(\alpha)$ is much smaller than the Jacobian used in Eq. 8 (Figure 1).

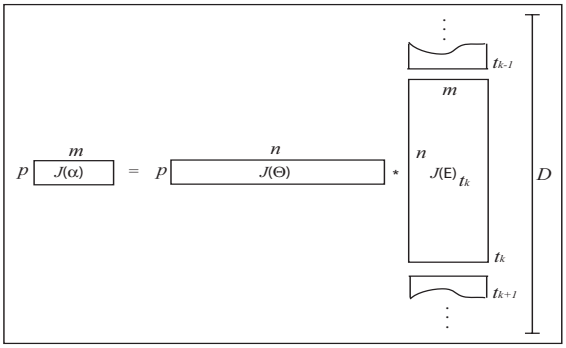


Figure 1. Obtaining $J(\alpha)$ from $J(\Theta)$ and the t_k slice of the Principal Components basis.

In the experiments reported in this paper (section 6), we have $p = 3$; $m = 9$ and $n = 93$. $J(\alpha)_{3\times9}$; $J(\Theta)_{3\times93}$; $J(\mathbf{E})_{t_k 93\times9}$ and the PC basis is as 12276×9 already said.

4.3 Continuity

When a motion is deformed at specific times continuity between frames should be maintained [Gle01]. In our case, continuity is enforced through the PCA parameters. As the α_i coefficients (Eq. 3) are considered invariants for all frames of the motion (Eq. 2). Then, if we modify the α_i coefficients for a pose Θ_{t_k}, the updated $\widetilde{\alpha}_i$, generate a full motion belonging the PC space.

More precisely, to deform a given motion, Ψ_D. We estimate its α_D coefficients projecting this motion in the linear model described by Eq. 3. The next step is to choose the pose $\Theta_{D_{t_k}}$, which should be edited, and extract its corresponding Jacobian, $J(\mathbf{E})_{t_k}$. Then, Eq. 10 is iteratively solved to compute the desired increment $\Delta\alpha_D$. The final step is to use the new $\widetilde{\alpha}_D$ coefficients ($\widetilde{\alpha}_D = \alpha_D + \Delta\alpha_D$) into Eq. 3 to impose continuity. Algorithm 1 describes the whole solution:

Algorithm 1 Motion deformation and continuity.

1: $\widetilde{\alpha}_D \leftarrow \alpha_D$; $\Delta\alpha_D \leftarrow \mathbf{0}$; $J(\mathbf{E})_{t_k} \leftarrow \mathbf{E}_{t_k}$
2: **while** not converged **do**
3: Compute $\{J(\Theta_{D_{t_k}}), \Delta\mathbf{x}\}$
4: $J(\widetilde{\alpha}_D) \leftarrow J(\Theta_{D_{t_k}})J(\mathbf{E})_{t_k}$
5: $\Delta\alpha_D \leftarrow J(\widetilde{\alpha}_D)^{\dagger^{\xi}}\Delta\mathbf{x}$
6: $\widetilde{\alpha}_D \leftarrow \widetilde{\alpha}_D + \Delta\alpha_D$
7: $\Theta_{D_{t_k}} \leftarrow \Psi_{\circ t_k} + \widetilde{\alpha}_D\mathbf{E}_{t_k}$
8: **end while**
9: $\widetilde{\Psi}_D \leftarrow \Psi_\circ + \sum_{i=1}^{m} \widetilde{\alpha}_{D_i}\mathbf{E}_i$

The convergence of algorithm 1 is achieved by tracking the error of the norm of the difference between the current state and the desired state: $\mathbf{e} = \|\mathbf{x}(\Theta) - \mathbf{g}\|$. We verified the algorithm's convergence for a simple example depicted by Figure 2. In this example, the golf club head should hits the ball onto another position defined by the user.

Figure 2. Left side: original pose $\Theta_{D_{t_i}}$. Right side: edited pose. This final pose was computed using 9 PC.

Figure 3 shows the convergence of Algorithm 1 as a function of the number of α_D coefficients. The convergence is faster when more than five dimensions are used (Table 1) as the solution posture is more likely to belong to the generated pose space for t_k. When less than six dimensions were used, we noticed the presence of some undesirable artifacts (e.g., feet sliding) during reconstruction (step 9 of the Algorithm 1). This problem is a consequence of the low percentage, corresponding to such a small number of PC, used during reconstruc-

tion [SHP04] and not of the quality of the updated $\widetilde{\alpha}_D$ coefficients.

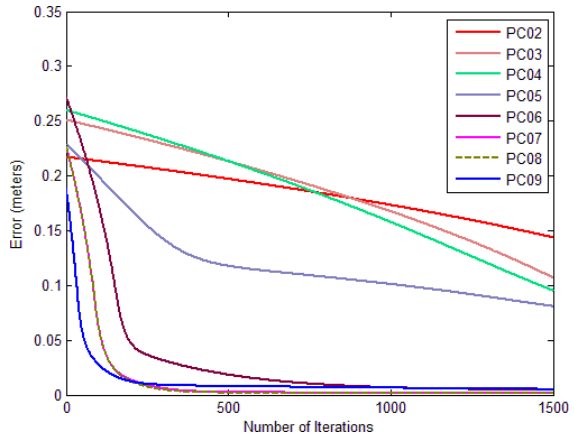


Figure 3. Convergence of algorithm 1. The convergence runs faster when more than five PC are used. We set 1500 iterations and chose a fixed value for the damping factor, $\xi = 10$.

PC	Time (seconds)	Nb. of Iterations
2	40.000	19106
3	110.000	50214
4	42.000	19929
5	9.000	4194
6	1.625	757
7	0.500	235
8	0.484	221
9	0.531	247

Table 1: Computation time, from lines 2 to 8 of the Algorithm 1, as a function of the number of PC. We used the values $\xi = 10$ and $\mathbf{e} < 0.01m$.

5 MOTION NORMALIZATION

A motion duration normalization is necessary in our approach because the PCA's parameters are estimated from complete motions. This is required as the single constrained posture should be as much as possible independent from duration variations of the original captured motions. For example, the golf club head should hit the ball at the same normalized time for all motions.

Let us recall the relationship between the motion index frame f_r, the frame-rate F_r (e.g., 25 fps), the motion duration T, and the normalized time:

$$f_r = TF_r t_k \tag{12}$$

Then, the total number of frames, F_T, can be expressed:

$$F_T = TF_r \tag{13}$$

The normalization is done in two stages. First, each motion Ψ_i has its corresponding total time T_i (Figure 4(a)). So, we preprocess the motion database to compute the mean time, T_μ (left side of Eq. 14), of all motions. We use this time directly into Eq. 13 to establish the final normalized size, which is $F = 132$. The normalization is done using quaternion spherical interpolation [Sho85] [1].But, for the root position we proceed with a simple linear interpolation.

The first step produces the normalized motions database, $\widetilde{\Psi}$ (Figure 4(b)). Second, we search in this normalized database for the key frame, K_{f_i}, for which the golf club should hits the ball. Then, we compute the mean key frame, K_{f_μ}, from all these motions:

$$T_\mu = \frac{1}{M}\sum_{i=1}^{M} T_i, \quad K_{f_\mu} = \frac{1}{M}\sum_{i=1}^{M} K_{f_i} \tag{14}$$

The mean key frame is used to compute the durations T_{μ_1} and T_{μ_2}, for the first half, $\left[K_{f_0}, ..., K_{f_\mu}\right]$, and second half, $\left[K_{f_{\mu+1}}, ..., K_{f_1}\right]$, of the final normalized motion, $\widetilde{\Psi}_{\mu_i}$ (Figure. 4(c)). Each interval is warped according to these two durations. The final normalized animations, $\widetilde{\Psi}_\mu$, are used to estimate the PCA's parameters (section 4.1).

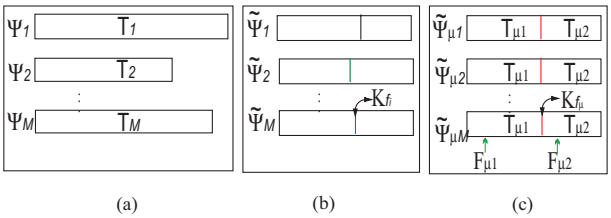


Figure 4. Motion normalization process.

6 EXPERIMENTS

We analyzed the performance of our approach in a number of experiments. We verified its performance, the style preservation of the edited motions, and its extrapolation capabilities. As a final experiment, we compare our model against a per-frame motion deformation technique [CB06] regarding performance, expressivity and simplicity.

6.1 Editing and Style

Figures 7(b) and 6 show the performance and style preservation results of the proposed model for motion editing. Figure 7(a) depicts the five goal positions (white balls) where the golf club head should hits the ball. The start position is the same for all five editing operations and is described by the blue ball. These positions were intentionally chosen to coincide with some of the positions of the motion database - to see if the edited motion would converge towards the original motion group or some of the others groups. For example, position 1

[1] The initial representation of the joint angles is exponential map. Before the interpolation, we convert them to quaternion. After the interpolation, we come back to exponential map to estimate the parameters of the linear model.

(Figure 7(a)) corresponds to the 3D position of the golf club head for the motion of group 2 (Figure 6), the input motion, which belongs to group 1, should hits the ball in that position. Figure 6 depicts the first two α_i coefficients of Eq. 3 for the 10 golf swing motion vectors. Note that, vectors corresponding to similar styles of golf swing tend to cluster. Figure 5 shows the percentage of the database (Eq. 4) for the first 9 PC. The 10th PC is negligible.

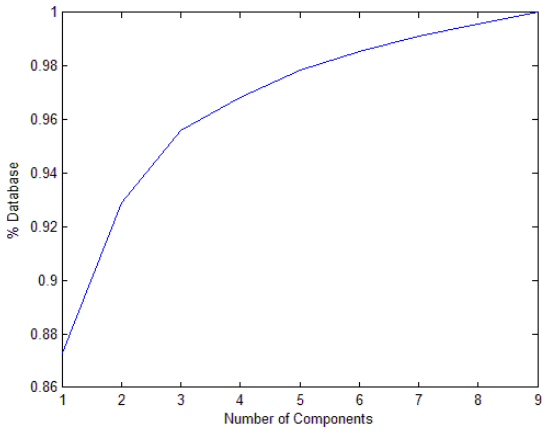


Figure 5. Percentage of the database.

Figure 7(b) shows the performance results. Each optimization took from 1 to 5 seconds approximately to converge. The algorithm's convergence depends of some parameters: the number of principal components (which we set to 8) and the intrinsic parameters of the optimizer (i.e., the damping factor, $\xi = 10$, the number of iterations, which we set to 1500, and the value of the tracked error, which we set to 0,01m). A high value of the regularization term ξ guarantees a stable but slow convergence. When ξ is zero the algorithm converges quickly (i.e, less than 12 iterations) but some undesirable poses and consequently discontinuities problems were produced for goals far from the initial end-effector position. This undesirable behavior happens because the IK solver tries to achieve unreachable poses, i.e, poses that are not in the database. The choice of a suitable value for ξ is thus one of the subtle difficulties to yield a good compromise between robustness and efficiency [Mac90, BB04]. The tracked error and the number of iterations were used to restrict the region of feasible poses during parameter's extrapolation. It means that, if the IK solver tries to achieve unreachable poses, the ones that are not in the motion database, the algorithm does not converge, but produces a result minimizing the norm of the error. Using these criterions a compromise between parameters' extrapolation and efficiency were established.

Finally, we verified that the edited motions preserved the same characteristics as the input motion (Figure 6). The optimization criteria used to solve the IK problem (Eq. 10) provides a local solution with minimum norm close to the starting state. Thereby, the final motion solution is found in the neighborhood of the starting state. This was checked by re-projecting the first two updated $\widetilde{\alpha}_{D_i}$ coefficients into the original golf motion database, as shown by Figure 6.

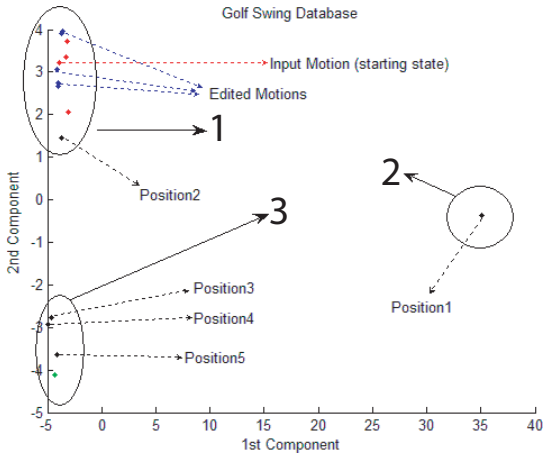


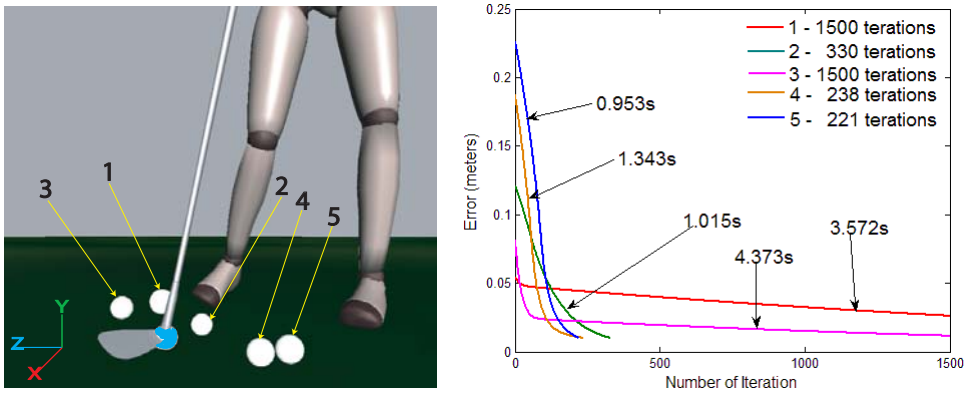
Figure 6. Clustering behavior of the motion database. Three groups of styles were found. The motions corresponding to the positions of Figure 7(a) are showed by black dots. The edited motions are showed by the blue dots. Three groups of style are showed.

6.2 Visual Comparison

In this section, we compare the visual quality of the edited motion. To edit a motion by using the per-frame method proposed by [CB06] the user has to specify the trajectory of the end-effector (modeled as spline curves) and the time intervals for which the motion will be deformed. On the contrary, by using our approach the user needs to specify the final goal position of the end-effector just for one frame - the one that should be deformed. Now, for visual comparison of both techniques, we asked the user to achieve the editing task of shifting the hit of the golf club head (in front of the left foot, at frame 94). The first line of Figure 8 shows the starting motion with the ball in the middle of the feet, the second line shows the animation result with our approach and the third line the result with the per-frame motion deformation method.

To edit this simple motion by using the per-frame approach, the user needed to add four more constraints in addition the golf club head constraint: three on the feet to prevent foot sliding and one to control the center of mass, to prevent unbalance poses. The user has to establish the ease-in and ease-out time intervals to prevent discontinuities. These time intervals increase computation cost. For this simple example, the computing cost was approximately 100 sec, without visualization. We also noticed inter penetration of the arms at frame 93, and discontinuities between frames (e.g., body balance) around the hit position. Of course, these problems can be fixed, but more tuning is needed.

On the contrary, with the proposed motion editing model only one constraint was necessary - the golf club head constraint - to achieve more realistic results. Our ap-



(a) (b)

Figure 7. Motion editing performance. The convergence is stopped when the end-effector is less than 1 cm of the goal. This threshold is used to delimit a region of feasible poses.

proach provided more natural poses (i.e., there is no inter penetration of the arms). The proposed model for continuity demonstrated more stable results, we did not noticed unbalanced poses or foot sliding. The computing cost was approximately 1 sec, without visualization. To summarize, if the motion needs just an adjustment of one key event, our approach can provide faster and more realistic results.

7 CONCLUSION AND FUTURE WORK

We have presented a new approach for interactive constraint-based motion editing, by constraining only a single pose in the motion space of the Principal Components and by imposing continuity exploiting this space. We demonstrated the performance, flexibility and simplicity of this approach through the editing of a golf swing motion and by comparing the results against a per-frame prioritized IK technique. Our approach provided faster and better quality results with less tuning from the user side.

One current limitation of our framework, compared with existing motion editing techniques [CB06], is that the constraint is applied to a single frame. Extending it to constrain multiple frames is possible but one has to keep in mind the small dimension of the PC space, that can quickly lead to an overconstrained problem. One possibility to alleviate this problem is to assign them a different priority [BB04]. Another possibility could be to solve them separately and interpolate the PC solutions between their associated frames. In any case the small computing cost for finding a solution in the presented context (less than 4s in general) makes our approach well suited for exploiting more versatile objectives.

8 ACKNOWLEDGEMENTS

Many thanks to Benoît Le Callennec, to providing access to his source code and for discussions about his method (with the support of the Swiss National Fund grant n^o 200020-109989). The authors would like to thank to Autodesk/Maya for their donation of Maya software. The authors would like to thank Raquel Urtasun and Pascal Fua for proving the data used in this paper and for part of the PCA source code. Thanks to Mireille Clavien for video production. This project was sponsored by the EPFL - Sport and Rehabilitation Engineering program.

9 REFERENCES

[AM00] M. Alexa and W. Muller. Representing animations by principal components. *Computer Graphics Forum*, 19(3):411–418, 2000.

[Bae01] Paolo Baerlocher. *Inverse Kinematics Techniques Of The Interactive Posture Control Of Articulated Figures*. Phd thesis, École Polytechnique Fédéral de Lausanne, 2001.

[BB04] P. Baerlocher and R. Boulic. An inverse kinematic architecture enforcing an arbitrary number of strict priority levels. *The Visual Computer*, 20(6), 2004.

[CB06] B. Le Callennec and R. Boulic. Interactive motion deformation with prioritized constraints. *Graphical Models*, 68 (2):175–193, March 2006. Special Issue on SCA 2004.

[Cra86] J. J. Craig. *Introduction to Robotics*. Addison-Wesley, Reading MA, 1986.

[GBT04] P. Glardon, R. Boulic, and D. Thalmann. A coherent locomotion engine extrapolating beyond experimental data. *Proceedings of CASA*, 2004.

[Gle97] M. Gleicher. Motion editing with spacetime constraints. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 139–ff., New York, NY, USA, 1997. ACM Press. ISBN 0-89791-884-3.

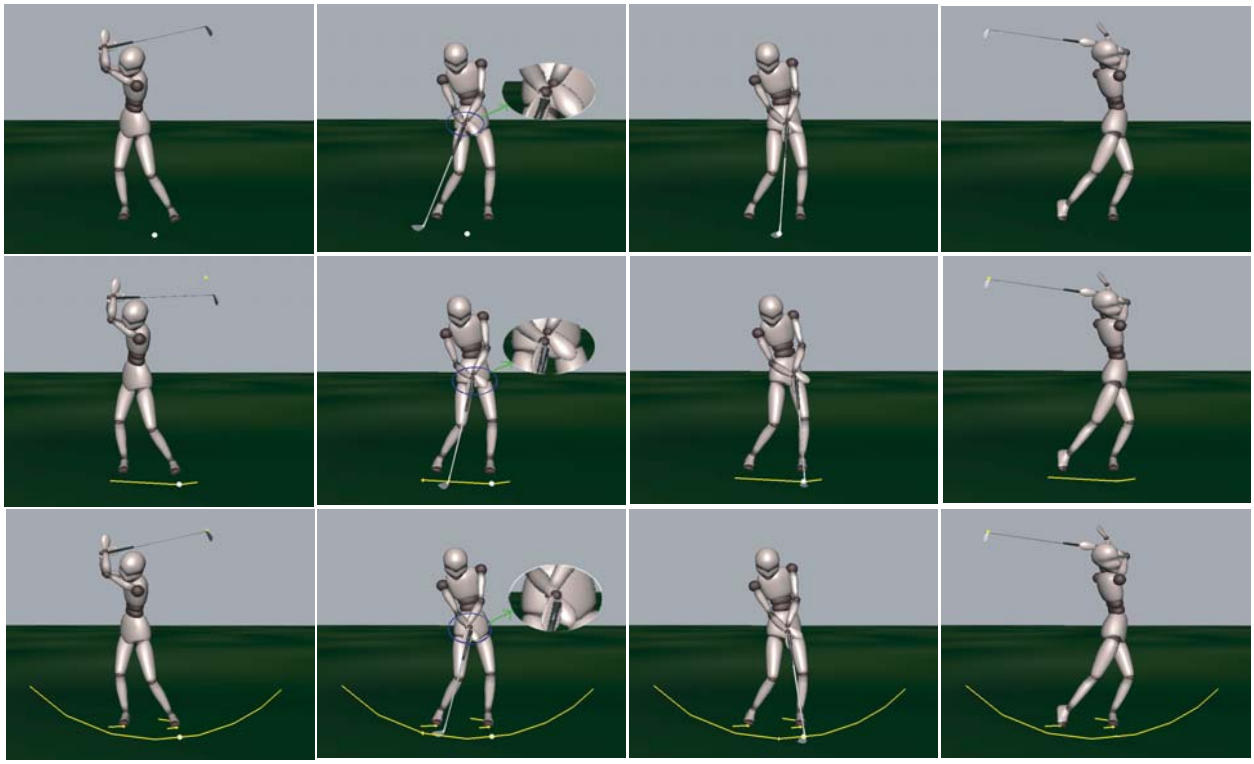


Figure 8. Animations results. Top row: original motion. Middle row: edited motion using the proposed approach. Bottom row: edited motion using the prioritized approach. Yellow lines show ease-in and ease-out time intervals. The prioritized approach produced a motion with inter penetration of the arms (second picture from left to right).

[Gle01] M. Gleicher. Comparing constraint-based motion editing methods. *Graphical models*, 63(2):107–134, 2001.

[Gra98] F. S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3): 29–48, 1998.

[Jol86] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[KMA05] R. Kulpa, F. Multon, and B. Arnaldi1. Morphology-independent representation of motions for interactive human-like animation. In *EUROGRAPHICS*, volume 24, pages 343–352, Number 2005.

[LS99] J. Lee and S.Y. Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of ACM SIGGRAPH*, 1999.

[LZqDMSH06] L. Liu, W. Zhao-qi, Z. Deng-Ming, and X. Shi-Hong. Motion edit with collision avoidance. In *Proceedings of the WSCG*, pages 303–310, January 2006.

[Mac90] A.A. Maciejewski. Dealing with the ill-conditioned equations of motion forarticulated figures. In *Computer Graphics and Applications, IEEE*, volume 10, pages 63–71, May 1990.

[Sho85] K. Shoemake. Animating rotation with quaternion curves. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, New York, NY, USA, 1985. ACM Press. ISBN 0-89791-166-0.

[SHP04] A. Safonova, J. K. Hodgins, and N. S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Trans. Graph.*, 23(3):514–521, 2004.

[UF04] R. Urtasun and P. Fua. 3d human body tracking using deterministic temporal motion models. In *European Conference on Computer Vision, Prague, Czech Republic*, May 2004.

[UGB+04] R. Urtasun, P. Glardon, R. Boulic, D. Thalmann, and P. Fua. Style-based motion synthesis. *Computer Graphics Forum (CGF)*, 23(4):799–812, November 2004.

[Whi69] D. E. Whitney. Resolved motion rate control of manipulators and human prostheses. In *IEEE Trans. Man-Mach. Syst*, volume 10 (2), pages 47–53, June 1969.