Eurographics/ACM SIGGRAPH Symposium on Computer Animation (2004)
R. Boulic, D. K. Pai (Editors)

Interactive Motion Deformation with Prioritized Constraints

Benoît Le Callennec and Ronan Boulic

Virtual Reality Lab, Swiss Federal Institute of Technology, Lausanne, Switzerland †

Abstract

In this paper, we present an interactive motion deformation method to modify animations so that they satisfy a set of prioritized constraints. Our approach successfully handles the problem of retargetting, adjusting a motion, as well as adding significant changes to preexisting animations. We introduce the concept of prioritized constraints to avoid tweaking issues for competing constraints. Each frame is individually and smoothly adjusted to enforce a set of prioritized constraints. The iterative construction of the solution channels the convergence through intermediate solutions, enforcing the highest prioritized constraints first. In addition, we propose a new, simple formulation to control the position of the center of mass so that the resulting motions are physically plausible. Finally, we demonstrate that our method can address a wide range of motion editing problems.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - Animation

1. Introduction

Producing high-quality character animations is still a research area of interest. One popular technique to achieve this is *motion capture*, as it is able to create realistic animations in a short period of time. Unfortunately, one of its major advantages is also its major drawback: the animation is recorded by mimicking the motion of a performer. Thus, the final animation must be planned before the capture is done and is only valid for virtual humans having specific proportions. That is why these animations are not directly reusable and need additional adaptations.

Recently, motion databases have become commercially available. Given a motion database, it now becomes a new challenge to create the animations we need for each virtual human we want to animate. Graph-based motion synthesis [AF02][KSG02][LCR*02] and blending techniques [KG03][PSS02][Per95] consider the database as a whole to construct new motions. The resulting animations are built from finely crafted combinations of the input data. These techniques are well-suited for video games for example. Conversely, motion editing techniques adapt a single animation to fit specific needs [Men99]. Many issues have been addressed over the recent years, like motion retargetting to different characters [Gle98] [MBBT00], and motion editing under various constraints [CK00] [LS99] [TSK02].

Our method aims at adding significant deformations to an input motion, while retaining as many of its characteristics as possible. The initial animation is deformed using prioritized constraints. The key feature of this technique is that prioritized constraints are sorted into priority-layers. Constraints belonging to the highest priority-layer are enforced first. Then, those of the next priority-layer are satisfied as much as possible without disturbing the previous ones, and so on... All the well-known constraints can be integrated in this framework: end-effector position and/or orientation, center of mass position, attraction toward the input motion... As a consequence, the animator is able to deform an animation in a very flexible way, with an arbitrary number of priority-layers and constraints. For example, it becomes possible to deform a walking motion into a stairs climbing motion, with irregular stairs' height, while ensuring a believable mapping of the character's balance in this new environment. For this latter feature, we propose a simple approach to estimate the desired position of the Center of Mass (CoM) in the target environment. The user can choose a default readjustment of unbalanced postures and/or add new effects.

In our framework, each frame is individually deformed with an Inverse Kinematics solver so that a set of predefined constraints is satisfied. Being a per-frame approach, a filter-

† email: {benoit.lecallennec|ronan.boulic}@epfl.ch

ing process is used to smooth the results when needed. This algorithm can be repeated, within an interactive design loop, to reenforce important constraints.

The next section reviews related work. Section 3 introduces our notation and presents our Inverse Kinematics solver. Section 4 details the major characteristics of the constraints handled by our framework, while section 5 describes the motion deformation algorithm. In section 6, we demonstrate how our method can be applied to deform a wide range of motions. Finally, we discuss its limitations in section 7 and we conclude the paper in section 8.

2. Related Work

Inverse Kinematics

Finding the joint variables for desired position and orientation of end-effectors has long been studied in Robotics. Efficient analytic solutions have been proposed for arm-like articulated chains [Pau81]. The same type of approach has been retained to compute the configuration of human limbs [BKK*85] [TGB00]. It is known as analytic Inverse Kinematics, and is widely used to position the hands and feet of animated characters.

Similar approaches, exploiting the output of multiple sensors attached to the limbs, are used in motion capture [BRRP97] and/or real-time interaction contexts [MHBT97]. Specialized analytic IK solvers improve the believability of the solution in contexts such as walking [SM01], computer puppetry [SLSG01], or footskate cleanup [KSG02].

However, despite its intrinsic efficiency, analytic IK suffers mostly from lack of flexibility. Numeric IK approaches have addressed this issue by solving a constraint optimization problem so that a locally optimal solution is found for an arbitrary number of end-effectors and joints [BMW87] [ZB94]. In addition, it is possible to optimize decisive criteria for the believability of the resulting posture, like static balance [BMT97]. A recent effort has shown the feasibility of exploiting the Null Space of the constraints' Jacobian to attract the posture toward an input movement [CK00]. A similar approach was used in [YN03b] to divide the constraints into two priority layers. Our contribution pushes this research direction further by considering an arbitrary number of prioritized constraints.

Constraints Definition for Character Animation

From our point of view, the most general formulation of constraints was introduced by Witkin et al. in [WFB87]. In their formulation, constraints are defined as energy functions which are then minimized. The initial problem formulation of spacetime constraints [WK88] was made to enforce physical constraints. [PW99] used Newton's laws on a simplified character to minimize computation costs. In [RGBC96], Rose et al. minimized energy consumption to obtain realistic transitions between motions, while Liu and Popović [LP02] used minimum mass displacement and momentum conservation. In [SKG03] and [TSK02], the authors considered the conservation of the zero moment point, which leads to realistic motion during constrained stages. Geometric constraints are more intuitive because they directly specify a goal for a specific body part: a point can be constrained to a specific position [Gle97], [Gle98], or can be constrained to move along a line [WFB87]. [LP02] and [WP95] used keyframes to specify constraints on motions. An example of a gaze constraint has been proposed in [CKB99].

Constraint-based Motion Editing Techniques

The research area of motion editing has become very active in the last years. One special class of such techniques is said to be constraint-based: the important features we need to preserve or to enforce are explicitly defined as constraints. While being considered as a pure signal processing technique, the method presented in [WP95] used keyframes to constrain the translation and the rotation of all joints. In [LS99], Lee and Shin introduced the *Per-Frame Plus Filtering* class of motion editing techniques. In [Gle98], Gleicher used a set of predefined constraints as input of a large constrained optimization problem. He used a similar technique in [Gle01b]. Choi and Ko [CK00] used an Inverse Kinematic solver to enforce constraints at each frame. A similar technique was described in [SLSG01] where a concept of importance is introduced to choose, at each frame, whether a constraint is relevant or not. Finally, Gleicher [Gle01a] introduced a taxonomy of constraint-based techniques.

3. Inverse Kinematics

3.1. Preliminaries

Motions may be represented as a continuous function of time $\mathbf{m}(t) = (\mathbf{p}(t), \mathbf{q}_0(t), \cdots, \mathbf{q}_n(t))$ where $\mathbf{p}(t)$ and $\mathbf{q}_0(t)$ represent the global position and orientation of the root node and $\mathbf{q}_i(t)$ is the local transformation of the i^{th} joint. For a given time t_α, the character's posture is defined as $\mathbf{m}(t_\alpha) = (\mathbf{p}(t_\alpha), \mathbf{q}_0(t_\alpha), \cdots, \mathbf{q}_n(t_\alpha))$.

3.2. Overview of the Inverse Kinematics Solver

In our method, we use a specific Inverse Kinematics solver to handle prioritized constraints. Given an articulated figure in a posture $\mathbf{m}(t_\alpha)$ and a task to satisfy $X = (x_0, \ldots, x_p)$ with x_i the i^{th} constraint, we have to determine a solution to the following non-linear equation:

$$X = F(\mathbf{m}(t_\alpha))$$

Our Inverse Kinematics solver is based on the well-known *inverse rate control* method [Whi69]. Then, we need to solve:

$$\Delta\mathbf{m}(t_\alpha) = J^{\dagger^\lambda}\Delta X + P_{N(J)}z \qquad (1)$$

with J the Jacobian, $P_{N(J)}$ the projection onto the Null-space of J, $J^{\dagger_\lambda}$ the damped pseudo-inverse [Mac90] and z an arbitrary vector expressing a variation of posture. z is defined as the optimization vector and is often used as the lowest prioritized-constraint to attract the solution toward a reference posture.

In the case of multiple constraints, the optimal solution should ideally satisfy all of them. However, this may not be possible, either because one of the constraints is not achievable, or because some constraints are not achievable simultaneously, whereas they can separately. In this latter case, these constraints are said to be in conflict. These conflicts may arise when one or more joints are shared by several constraints. While *weighting strategies* try to minimize the distributed error, a *priority strategy* sorts the constraints by order of priority to satisfy the most important constraints first.

The final algorithm extends the equation 1:

$$\begin{aligned}
\Delta\mathbf{m}(t_\alpha)_1 &= \tilde{J}_1^{\dagger_{\lambda_1}}\Delta x_1 \\
\text{For i=2..k} & \\
\Delta\mathbf{m}(t_\alpha)_i &= \Delta\mathbf{m}(t_\alpha)_{i-1} + \tilde{J}_i^{\dagger_{\lambda_i}}(\Delta x_i - J_i\Delta\mathbf{m}(t_\alpha)_{i-1}) \\
&\quad \text{with } \tilde{J}_i = J_i P_{N(J_{i-1}^A)} \\
\text{where } P_{N(J_i^A)} &= P_{N(J_{i-1}^A)} - \tilde{J}_i^{\dagger}\tilde{J}_i \\
\text{and } P_{N(J_1^A)} &= I_n
\end{aligned}$$

where i is the current priority-layer, k the total number of priority-layers and J_i^A the *augmented Jacobian* defined by:

$$J_i^A = \begin{bmatrix} J_1 \\ J_2 \\ \vdots \\ J_i \end{bmatrix}$$

For further details, the reader may refer to [BB04].

One major advantage of our IK solver is its great flexibility. For example we can precisely define which joints will be recruited to satisfy a specific constraint.

4. Constraints Definition and Specification

In this section, we detail important characteristics of the constraints we use in our motion deformation tool. Every motion editing tool must provide constraints to control the position as well as the orientation of end-effectors. In addition, the control of the CoM of the character could be of interest to adjust unbalanced postures or to simulate effects such as "walking against the wind".

4.1. Flexible Recruiting Level

A common approach in motion editing techniques is to directly consider the underlying structure. For example, human-like characters can be advantageously partitioned into smaller sub-hierarchies (limbs, spine,...). These latter are then controlled in a more efficient way by an Inverse Kinematics solver dedicated to this problem. As we aim at providing a general framework to control any kind of hierarchy, this specialization is not satisfactory. Conversely, considering all the possible joints to control the position of an end-effector may prove to be counterproductive. While controlling the position of the wrist for example, it is important to choose whether we want the spine to participate or not. Thus, the present approach allows a constraint to "recruit" all or part of the joints from its parent joint up to the root of the hierarchy. We can then easily define kinematics chains of varying lengths providing the user with more control over the final results.

The joint recruiting is submitted to one strict design rule that guarantees the enforcement of the hierarchy of priorities. It concerns those parts of the skeleton where multiple constraints may recruit part of their joints; in these regions, the rule requires joints sets associated to high priority constraints to include any joints set associated to lower priority constraints. Failing to do so may lead to diverging configurations. The problem of multiple overlapping kinematics chains was first addressed in [BB98]. Instead of using an algorithmic scheme to solve this problem, we enforce a minimal recruiting rule in order to obtain an effective enforcement of the scale of priorities. Let C_i be a constraint of priority i, $Rec(C_i)$ its associated set of recruited joints. Then, for $a > b$ we have:

$$Card(Rec(C_a) \cap Rec(C_b)) > 1 \Rightarrow Rec(C_b) \subset Rec(C_a) \quad (2)$$

4.2. Shape-Constraints Definition and Representation

Constraint-based motion editing techniques require the user to specify the important features the final motion should achieve. Providing interactive tools to construct such constraints is a key point when designing motion editing methods. Systems failing to do so tend to be useless as it is not reasonable to ask the user to specify constraints in mathematical form. In particular, constraints representing the trajectory of end-effectors are very useful. As these constraints represent trajectories of end-effectors, they must be continuous in space as well as in velocity except at specific points. We define a *shape constraint* as a pair of curves $(\mathbf{S}(u), \mathbf{T}(t))$. $\mathbf{S}(u)$ defines the trajectory of the end-effector in space and $\mathbf{T}(t)$ its timing.

The Trajectory Curve The trajectory $\mathbf{S}(u)$ of an end-effector is represented as a *Kochanek-Bartels* spline [KB84]. This class of C^2 continuous interpolating cubic splines offers great flexibility at control points. Indeed, their tangents are directly influenced by three parameters: the *tension*, the *continuity* and the *bias*. These parameters are useful to explicitly change (or even break) the continuity of end-effectors' trajectories. Suppose for example that we need to adjust the

wrist position so that it reaches a stationary location over a period of time. In this case, there is no need for the trajectory to be continuous. The control parameters are then adjusted to break the continuity at that point (the tension is set to 1). In addition, we need to reparameterize $\mathbf{S}(u)$ so that the corresponding interval of time corresponds to the same parameter u.

The Time Curve The *time curve* $\mathbf{T}(t)$ is defined as an increasing piecewise linear function. This function establishes the correspondence between each time of the animation and the trajectory curve $\mathbf{S}(u)$. More formally, given a time t_α, the corresponding 3D position P in the shape-constraint is then defined as:

$$P = \mathbf{S}(\mathbf{T}(t_\alpha)) \tag{3}$$

Finally, the construction of a shape-constraint that holds over a period of time $[t_{begin}, t_{end}]$ consists of the following steps:

1. The first (resp. the last) control point of $\mathbf{S}(u)$ is created at the initial end-effector's position at time t_{begin} (resp t_{end})
2. The tangent of the first (resp. the last) control point is adjusted with respect to the velocity of the end-effector in the input motion.
3. For each end-effector target locations defined by the user, a control point is added to $\mathbf{S}(u)$.
4. For each control point where the end-effector should remain stationary over a period of time, the continuity of the spline is broken by setting its *tension* to 1.
5. Finally, $\mathbf{T}(t)$ is computed so that $\mathbf{S}(\mathbf{T}(t_\alpha)) = P$ for $t_{begin} \leq t_\alpha \leq t_{end}$, with P a 3D location.

Note that the end-user only needs to define the timing of the shape-constraint as well as the position and timing of the points the shape-constraint has to pass by. The trajectory and timing curves are then automatically computed.

4.3. Balance Control

Most of the constraint-based motion editing tools do not consider physical properties when solving for a solution [Gle01b] [SLSG01] [CK00] [LS99] [WP95] [BW95]. These approaches usually make this sacrifice to achieve simplicity and computational efficiency. Few methods constraint the Zero Moment Point of the character to remain inside its support polygon [TSK02][DN99][KB96]. However, this concept often relies on expensive numerical methods. In [SKG03], the authors make the sacrifice of physical correctness to achieve interactive rates. However, the ZMP is well-defined for planar ground but is difficult to generalize to uneven terrains [SB04]. Other methods [LP02] [PW99] use a sub-set of physical laws to improve physical realism. [YN03a] and [PR01] use a dynamics filter to track a reference motion while enforcing dynamic constraints. In [ZH02], the authors maintain balance either using a balance controller pushing the CoM toward the center of support, or adjusting the desired reference motion by offsetting the joint angles of the legs similarly as in [WH00]. Finally Pai and Patton demonstrated the relationship between velocity and position of the CoM and proposed a prediction model to decide whether the balance can be maintained or not[PP97].

In our framework, we choose to control the CoM's position through Inverse Kinetics [BMT96]. Unbalanced postures are then adjusted to improve realism or to apply some additional effects. The final CoM's position $\mathbf{P}'_{CoM}$ first needs to be estimated. For this, we consider the initial CoM's position $\mathbf{P}_{CoM}$ as well as a set of points A on the body. A may be different from one motion to another. Then, $\mathbf{P}'_{CoM}$ is defined using the relation:

$$N_1(\mathbf{P}'_{CoM} - \frac{\sum_{i=1}^{Card(A)} \mathbf{A}'_i}{Card(A)}) = N_2(\mathbf{P}_{CoM} - \frac{\sum_{i=1}^{Card(A)} \mathbf{A}_i}{Card(A)})$$

where $N1$ and $N2$ are normalization functions, and $\mathbf{A}_i$ (resp. $\mathbf{A}'_i$) the position of the i^{th} joint of A in the input motion (resp. output motion).

$\frac{\sum_{i=1}^{Card(A)} \mathbf{A}_i}{Card(A)}$ is the position of the barycenter of A in the input motion and $\frac{\sum_{i=1}^{Card(A)} \mathbf{A}'_i}{Card(A)}$ is its corresponding position in the output motion.

Figure 1 shows the initial trajectories of a character's CoM and the barycenter of A for a simple walking animation. In this example, A is composed of the heel and toe joints of each leg.

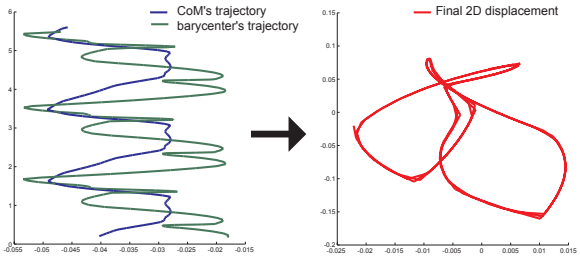

Figure 1: ***Left****: Input trajectories of the CoM and the barycenter of A for a walking motion* ***Right:*** *2D displacement between the CoM and the barycenter of A. The units are different on both axes to show the details.*

A *3D displacement map* may be added to the resulting curve to produce new behaviors. Notice that the choice of A is crucial as it directly influences the deformed motion. However, in most cases, it remains intuitive: for example, for classical walking or running motions, A should contain points of the feet.

While this method cannot ensure that the character is dynamically balanced, it has the advantage that the balance constraint is treated as any other constraint in our framework. Moreover, as the deformed motion is usually close

enough the initial one, the adjustments are small, and simply constraining the CoM position to its original trajectory often provides good results. However, for highly dynamic motions, or when adding drastic changes to the initial animation, physical laws should be taken into account.

5. Motion Deformation Method

5.1. Overview of the method

Our purpose is to find a *displacement map* **d(t)** so that the output motion $\mathbf{m}_{output}(t) = \mathbf{m}_{input}(t) \oplus \mathbf{d(t)}$ satisfies a set of predefined constraints C. A common method, first introduced by Lee and Shin [LS99], considers the input motion as a set of independent character postures. Each of these postures is individually adjusted so that the constraints C are satisfied. Finally, the *displacement maps* are low-pass filtered to ensure temporal consistency between neighboring frames. This method has shown very good results when editing preexisting motions, but is not practical if the final postures deviate significantly from the input ones. Indeed, as the IK solver starts from the original posture, neighboring deformed postures may be quite dissimilar and the low-pass filtering step inevitably leads to unrealistic results. Instead, our approach is more similar to [CK00]. We make the assumption that each character's posture in the output motion is similar to the previous one. Regarding the joints that were not recruited by constraints, they should exactly reproduce their original motion. Then, for a given instant of time t_α, the starting state of the deformed posture $\mathbf{m}_{output}(t_\alpha) = (\mathbf{p}'(t_\alpha)), \mathbf{q}_0'(t_\alpha), \cdots, \mathbf{q}_n'(t_\alpha)$ is defined as:

$$\mathbf{m}_{output}(t_\alpha) = (\mathbf{p}^*(t_\alpha), \mathbf{q}_0^*(t_\alpha), \cdots, \mathbf{q}_n^*(t_\alpha)) \oplus \mathbf{d}(t_\alpha) \quad (4)$$

where (the same holds for $\mathbf{p}^*(\mathbf{m}(t_\alpha))$):

$$\mathbf{q}_i^*(t_\alpha) = \begin{cases} \mathbf{q}_i'(t_\alpha - \Delta t) & \text{if the joint is controlled} \\ \mathbf{q}_i(t_\alpha) & \text{otherwise} \end{cases}$$

for $0 \leq i \leq n$. Δt is the time interval between two consecutive frames. Each adjusted posture is attracted toward its corresponding one in the input motion, thanks to the optimization vector described in section 3. Finally, the result is low-pass filtered to avoid discontinuities.

5.2. Enforcing the Continuity

Adjusting each frame individually may violate the *inter-frame consistency*. Thus, to ensure that our approach effectively produces natural looking motions, we need to low-pass filter the deformation we want to add to the original animation. During the IK step, each joint is attracted toward its original value using the optimization vector. Only the translation part of the root is not considered in this optimization vector. The translation components are then low-pass filtered using a FIR filter. In cases where the discontinuity was too important, this filtering process significantly alters important constraints and we need one more pass.

Another issue when dealing with continuity is the activation/deactivation of constraints. In our framework, the positional constraints are not considered as location in space but instead as continuous trajectories. So, to enforce a positional constraint at a specific time t_α, we define a shape-constraint between times $t_\alpha - \Delta t_{c_b}$ and $t_\alpha + \Delta t_{c_e}$ where Δt_{c_b} is the duration to go from the original trajectory to the goal location and Δt_{c_e} the duration to go from the goal location back to the original trajectory.

6. Experimental Results

The final system is integrated into Alias^{TM}/Maya5 as plug-ins and MEL scripts. Figure 2 shows a screenshot of the application. For this particular example, it took less than one

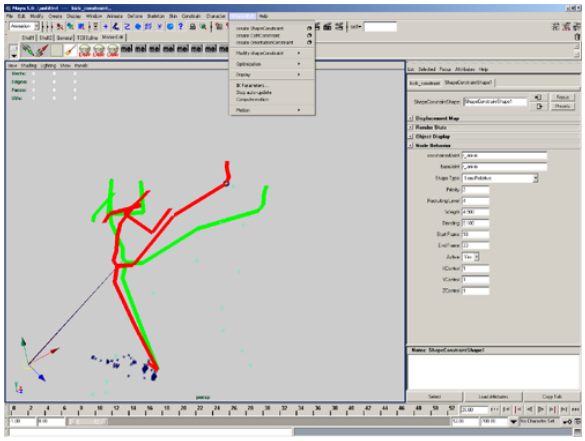

Figure 2: *Motion deformation system integrated into Alias^{TM}/Maya5*

minute to create the needed constraints. Indeed, the end-user only needs to specify the timing of each shape-constraint as well as the position and timing of the points it has to pass by. The others constraints are simply created using MEL commands. Moreover, assigning a priority to each constraint is straightforward as the important point is the relative priority between them. For example, a set of constraints C_1, C_2 and C_3 with priority 10, 2 and 15 gives the same results if the priorities were 2, 1 and 3. Finally, though it was not used to generate the results presented in this section, the user can assign a weight to each constraint. Thus, when two or more constraints with the same priority-level are conflicting, the conflict is solved using the usual weighting strategy.

We used our motion deformation algorithm to create a wide range of animations. All the animations are generated on a IBM T40p (Pentium M 1.6 GHz, 1Go RAM, ATI mobility FireGL 9000).

Reaching

We generated the animation shown in figure 3 by deforming a motion-captured walking animation. This motion is composed of 145 frames (6 seconds). We enforce footplants us-

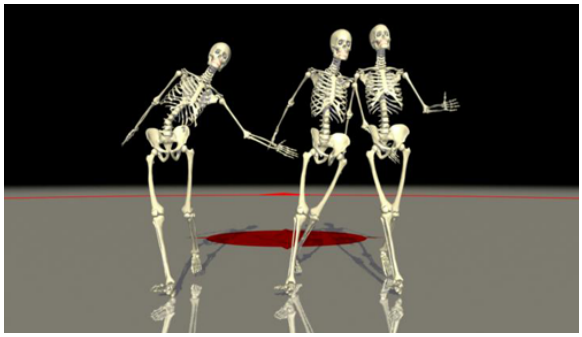

Figure 3: ***Left:*** *Final deformed motion* ***Middle:*** *Deformed motion without CoM constraint* ***Right:*** *Input motion*

ing four high-priority shape-constraints. The left arm is then controlled to reach a specific location with a middle-priority shape-constraint onto the wrist. This constraint is activated during 40 frames only. Finally, the CoM is constrained to follow its original trajectory to avoid unbalanced postures (such as the one shown in the middle of figure 3). The whole deformed animation is generated in approximately 9 seconds without visualization and 12 seconds with visual feedback. If we only modify the reaching constraint, the final animation is then recomputed in less than 4 seconds (with visualization) providing immediate feedback to the user.

Stepping

The figure 4 shows the difference between the weighting and the priority strategies. Footplants are constrained during the whole animation. The ankle is also constrained to force the character to raise the leg. At this instant, both shape-constraints are conflicting. Thus, we assign a higher priority to the shape-constraint controlling the ankle. This way it is easy to define global low-priority constraints on the whole animation and local high-priority constraints to deform a short part of the motion. The final animation is computed in approximately the same time as in the previous example.

Karate Motion

In this example, the right foot is constrained to follow its original trajectory (high-priority). The left ankle of the character is constrained to reach a higher location (low-priority). Finally, the CoM is controlled to enforce balance (middle-priority). Once again, the priority level assigned to each constraint allows to decide whether to keep the left foot planted, or to constraint the right foot the reach its goal at all costs. In figure 5 we demonstrate that controlling the CoM position

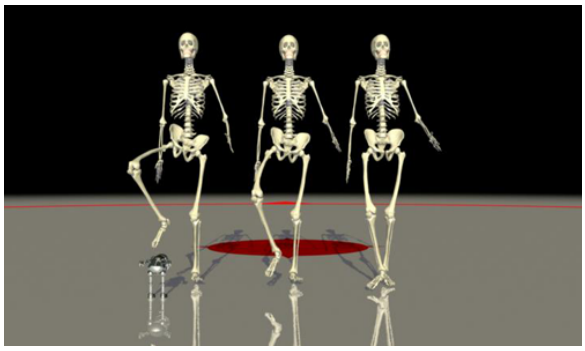

Figure 4: ***Left:*** *Deformed motion with prioritized constraints: the goal location is reached.* ***Middle:*** *Deformation with weighted constraints: the final ankle position corresponds to a weighted average of the constraints.* ***Right:*** *Original motion*

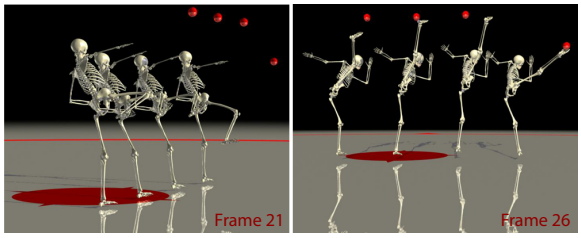

Figure 5: ***Left:*** *Deformed motion. The ankle's goal cannot be reached without disturbing higher priority constraints.* ***Middle left:*** *The CoM is not controlled anymore resulting to unbalanced postures.* ***middle right:*** *Resulting motion using weighting constraints. The location of the right foot is disturbed.* ***Right:*** *Original motion*

as well as assigning priority to constraints may facilitate the process of motion deformation. In figure 6, we constrained

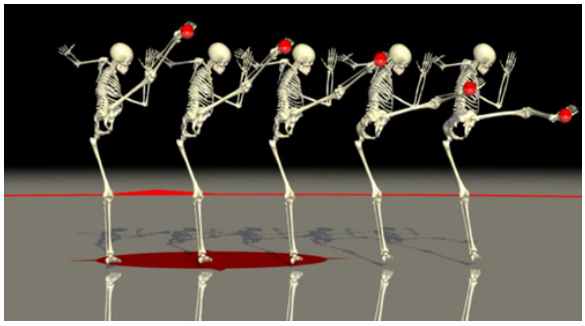

Figure 6: *Reaching different goals.*

the right ankle to reach goals with different heights. All these examples were generated in less than 2 seconds with visualization.

Putting it all together

Finally, we applied our method to a walking animation to obtain a "climbing stairs" motion. We used different classes of constraints. We used four shape-constraints to design the "walking on stairs pattern". One additional shape-constraint is used to constraint the relative position of the right elbow with respect to the torso. The orientation of the right arm is also constrained as well as the CoM of the character. The final result is shown in figure 7. The whole animation needed

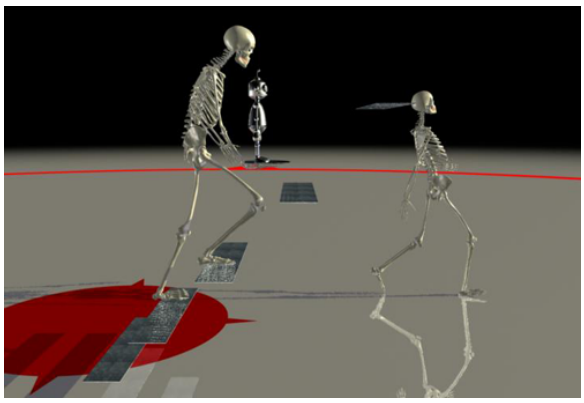

Figure 7: *Example of a deformed animation with five shape-constraints, one rotational-constraint, and control of the position of the CoM.*

two passes as the filtering process significantly disturbed the footplant constraints. Finally, the whole animation was generated in approximately 40 seconds without visualization.

7. Discussion and future works

Filtering The actual filtering process is simple but suffers from one drawback: our method becomes inherently off-line. We will investigate the advantages/disadvantages of using on-line filters [TSK02] [SLSG01].

Quality of the input motion As we directly rely on the input motion to estimate the CoM's position in the resulting animation, it becomes difficult to use in cases where the original motion is too noisy. We will improve the robustness of our approach by first cleaning the input motions (filtering and enforcing important constraints).

Footplants Definition Using wireframe models is common place in animation. However, this kind of approach tends to hide important problems. For example, consider the problem of defining a simple footplant constraint. While defining two positional constraints on each foot is visually accurate for wireframe models, it becomes inefficient for anatomical ones. Indeed, while both positional constraints are met, it still remains one rotational degree of freedom along the axis joining these constraints. The definition and addition of accurate footplant constraints to our framework is left for future work.

8. Conclusions

In this paper we have presented an interactive method for adding significant changes to an animation. Our approach improves classical motion editing techniques, as animators can add large deformations without ending with unbalanced results. Moreover, the priority concept greatly helps when animators need to arbitrate conflicting constraints.

We have introduced a versatile class of constraints: the *shape-constraints*. The end-user is able to design a wide range of trajectories in space. Moreover, these may contain stationary points breaking (or not) the continuity of the trajectory. Furthermore, these constraints can be expressed in a reference frame allowing relative constraints between joints, to shift a joint position or to define an absolute trajectory. We have also presented a simple approach to adjust unbalanced postures by controlling the position of the center of mass thanks to Inverse Kinetics.

Our algorithm allows to assign a priority to each constraint. This priority is used to arbitrate conflicts between overlapping constraints (i.e. controlling a common sub-set of joints over an interval of time). Our scheme ensures that high-priority constraints won't be disturbed by low-priority ones.

While we have mainly focused our discussion on motion deformation, our method is also well-suited to deal with retargetting problems.

Acknowledgments

The authors would like to thank Thierry Michelot for the design of the character and the scene as well as Stéphanie Noverraz for careful proofreading. Special thanks to the reviewers for their helpful comments. The Maya licences has been granted by Alias through their research donation program. This research was supported by the Swiss National Science Foundation under the grant 200020-101464.

References

[AF02] ARIKAN O., FORSYTH D. A.: Interactive Motion Generation From Examples. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (2002), pp. 483–490.

[BB98] BINDIGANAVALE R., BADLER N. I.: Motion Abstraction and Mapping with Spatial Constraints. *Lecture Notes in Computer Science 1537* (1998), 70–83.

[BB04] BAERLOCHER P., BOULIC R.: An Inverse Kinematic Architecture Enforcing an Arbitrary

Number of Strict Priority Levels. *The Visual Computer 20*, 6 (2004).

[BKK*85] BADLER N. I., KOREIN J. D., KOREIN J. U., RADACK G. M., BROTMAN L. S.: Positioning and animating human figures in a task-oriented environment. *The Visual Computer 1*, 4 (Dec. 1985), 212–220.

[BMT96] BOULIC R., MAS R., THALMANN D.: A robust approach for the control of the center of mass with inverse kinetics. *Computers and Graphics 20*, 5 (Sept.–Oct. 1996), 693–701.

[BMT97] BOULIC R., MAS R., THALMANN D.: Complex character positioning based on a compatible flow model of multiple supports. *IEEE Transactions on Visualization and Computer Graphics 3*, 3 (July/Sept. 1997), 245–261.

[BMW87] BADLER N. I., MANOOCHEHRI K. H., WALTERS G.: Articulated figure positioning by multiple constraints. *IEEE Computer Graphics and Applications 7*, 6 (June 1987), 28–38.

[BRRP97] BODENHEIMER B., ROSE C., ROSENTHAL S., PELLA J.: The process of motion capture – dealing with the data. In *Computer Animation and Simulation '97* (1997), Thalmann D., van de Panne M., (Eds.), Eurographics, Springer-Verlag Wien New York, pp. 3–18.

[BW95] BRUDERLIN A., WILLIAMS L.: Motion Signal Processing. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1995), pp. 97–104.

[CK00] CHOI K.-J., KO H.-S.: Online Motion Retargetting. *Journal of Vizualisation and Computer Animation 11* (2000), 223–235.

[CKB99] CHOPRA-KHULLAR S., BADLER N. I.: Where To Look? Automating Attending Behaviors of Virtual Human Characters. In *Proceedings of Conference on Autonomous Agents* (may 1999), pp. 16–23.

[DN99] DASGUPTA A., NAKAMURA Y.: Making feasible walking motion of humanoid robots from human motion capture data. In *Proceedings of IEEE International Conference on Robotics and Automation* (may 1999), pp. 1044 – 1049.

[Gle97] GLEICHER M.: Motion Editing with Spacetime Constraints. In *Proceedings of ACM Symposium on Interactive 3D Graphics* (apr 1997), pp. 139–148.

[Gle98] GLEICHER M.: Retargetting motion to new characters. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1998), pp. 33–42.

[Gle01a] GLEICHER M.: Comparing constraint-based motion editing methods. *Graphical models 63*, 2 (Mar. 2001), 107–134.

[Gle01b] GLEICHER M.: Motion path editing. In *Proceedings of ACM Symposium on Interactive 3D Graphics* (2001), pp. 195–202.

[KB84] KOCHANEK D. H. U., BARTELS R. H.: Interpolating Splines with Local Tension, Continuity and Bias Control. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1984), pp. 33–43.

[KB96] KO H., BADLER N. I.: Animating Human Locomotion with Inverse Dynamics. *IEEE Computer Graphics and Applications 16*, 2 (mar 1996), 50–58.

[KG03] KOVAR L., GLEICHER M.: Flexible Automatic Motion Blending with Registration Curves. In *Proceedings of ACM SIGGRAPH/EUROGRAPHICS Symposium on Computer Animation* (2003), pp. 214–224.

[KSG02] KOVAR L., SCHREINER J., GLEICHER M.: Footskate Cleanup for Motion Capture Editing. In *Proceedings of ACM SIGGRAPH/EUROGRAPHICS Symposium on Computer Animation* (2002), pp. 97–104.

[LCR*02] LEE J., CHAI J., REITSMA P. S. A., HODGINS J. K., POLLARD N. S.: Interactive Contol of Avatars Animated With Human Motion Data. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (2002), pp. 491–500.

[LP02] LIU C. K., POPOVIC Z.: Synthesis of Complex Dynamic Character Motion from simple animation. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (2002), pp. 408–416.

[LS99] LEE J., SHIN S. Y.: A Hierarchical Approach to Interactive Motion Editing for Human-Like Figures. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1999), pp. 39–48.

[Mac90] MACIEJEWSKI A. A.: Dealing with the ill-conditioned equations of motion for articulated figures. *IEEE Computer Graphics and Applications 10*, 3 (May 1990), 63–71.

[MBBT00] MONZANI J.-S., BAERLOCHER P., BOULIC R., THALMANN D.: Using an Intermediate Skeleton and Inverse Kinematics for Motion Retargeting. In *Proceedings of Eurographics* (2000).

[Men99] MENACHE A.: *Understanding Motion Capture for Computer Animation and Video Games*. Morgan Kaufmann Publishers Inc., 1999.

[MHBT97] MOLET T., HUANG Z., BOULIC R., THALMANN D.: An Animation Interface Designed for Motion Capture. In *Proceedings of Computer Animation and Social Agents* (1997), pp. 77–85.

[Pau81] PAUL R. P.: *Robot Manipulators: Mathematics, Programming and Control*. MIT Series in Artificial Intelligence. The MIT Press, 1981.

[Per95] PERLIN K.: Real time responsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics 1*, 1 (1995), 5–15.

[PP97] PAI Y.-C., PATTON J.: Center of mass velocity-position predictions for balance control. *Journal of Biomechanics 30*, 4 (1997), 347–354.

[PR01] POLLARD N. S., REITSMA P. S. A.: Animation of Humanlike Characters: Dynamic Motion Filtering with a Physically plausible Contact Model. In *Proceedings of Yale Workshop on Adaptative and Learning Systems* (2001).

[PSS02] PARK S. I., SHIN H. J., SHIN S. Y.: On-line Locomotion Generation Based on Motion Blending. In *Proceedings of ACM SIGGRAPH/EUROGRAPHICS Symposium on Computer Animation* (2002).

[PW99] POPOVIC Z., WITKIN A.: Physically Based Motion Transformation. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1999), pp. 11–20.

[RGBC96] ROSE C., GUENTER B., BODENHEIMER B., COHEN M. F.: Efficient Generation of Motion Transitions using Spacetime Constraints. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1996), pp. 147–154.

[SB04] SARDAIN P., BESSONNET G.: Forces acting on a biped robot. center of pressure - zero moment point. *IEEE Transactions on Systems Man and Cybernetics part A* (2004).

[SKG03] SHIN H. J., KOVAR L., GLEICHER M.: Physical Touch-up of Human Motions. In *Proceedings of Pacific Graphics* (oct 2003).

[SLSG01] SHIN H. J., LE J., SHIN S. Y., GLEICHER M.: Computer puppetry: An importance-based approach. *ACM Transactions on Graphics 20* (2001), 67–94.

[SM01] SUN H. C., METAXAS D. N.: Automating Gait Generation. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (2001).

[TGB00] TOLANI D., GOSWAMI A., BADLER N. I.: Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical models 62*, 5 (Sept. 2000), 353–388.

[TSK02] TAK S., SONG O.-Y., KO H.-S.: Spacetime Sweeping: An Interactive Dynamic Constraints Solver. In *Proceedings of Computer Animation and Social Agents* (2002), pp. 261–270.

[WFB87] WITKIN A., FLEISCHER K., BARR A.: Energy constraints on parameterized models. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1987), pp. 225–232.

[WH00] WOOTEN W. L., HODGINS J. K.: Simulating leaping, tumbling, landing and balancing humans. In *Proceedings of IEEE International Conference on Robotics and Automation* (apr 2000).

[Whi69] WHITNEY D.: Resolved motion rate control of manipulators and human prostheses. *IEEE Transactions on Man-Machines Systems MMS-10*, 2 (1969), 47–53.

[WK88] WITKIN A., KASS M.: Spacetime Constraints. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1988), pp. 159–168.

[WP95] WITKIN A., POPOVIC Z.: Motion Warping. In *Proceedings of ACM SIGGRAPH, Annual Conference Series* (1995), pp. 105–108.

[YN03a] YAMANE K., NAKAMURA Y.: Dynamics Filter-Concept and Implementation of On-line Motion Generator for Human Figures. *IEEE Transactions on Robotics and Automation 19*, 9 (jun 2003), 421–432.

[YN03b] YAMANE K., NAKAMURA Y.: Natural Motion Animation through Constraining and Deconstraining at Will. *IEEE Transactions on Visualization and Computer Graphics 9*, 3 (jul 2003), 352–360.

[ZB94] ZHAO J., BADLER N. I.: Inverse Kinematics Positioning Using Nonlinear Programming for Highly Articulated Figures. *ACM Transactions on Graphics 13*, 4 (oct 1994), 313–336.

[ZH02] ZORDAN V., HODGINS J.: Motion Capture-Driven Simulations that Hit and React. In *Proceedings of ACM SIGGRAPH/EUROGRAPHICS Symposium on Computer Animation* (2002), pp. 89–96.