Retargetting Motion to New Characters

Michael Gleicher *

Autodesk Vision Technology Center

Abstract

In this paper, we present a technique for *retargetting* motion: the problem of adapting an animated motion from one character to another. Our focus is on adapting the motion of one articulated figure to another figure with identical structure but different segment lengths, although we use this as a step when considering less similar characters. Our method creates adaptations that preserve desirable qualities of the original motion. We identify specific features of the motion as constraints that must be maintained. A spacetime constraints solver computes an adapted motion that re-establishes these constraints while preserving the frequency characteristics of the original signal. We demonstrate our approach on motion capture data.

CR Categories and Subject Descriptors: I.3.7 [Computer Graphics]: Three Dimensional Graphics and Realism - Animation

Additional Keywords: motion editing, motion signal-processing, spacetime constraints, motion capture.

1 Introduction

In this paper, we present techniques for *retargetting* motion: the problem of adapting an animated motion from one character to another. Our goal is to re-use motions created for one character on other characters, independently of how that motion was created. We aim to preserve as many of the desirable properties of the original motion as possible. That is, if we begin with the motion of a tall adult person, we expect to end up with a motion of a small child walking like an adult, or a crocodile swing dancing as if it were an adult human. Admittedly, this faithfulness to the original motion is not always artistically desirable. However, we prefer to relegate the difficult creative decisions (How do crocodiles dance?) to the user's selection of an initial motion.

Our focus is on applying motion created for one articulated figure to another figure with identical structure (connectivity of limbs, types of joints, number of degrees of freedom) but different segment lengths. Even when two articulated figures share structure, the motion of one may not trivially apply to the other and therefore require adaptation. Good adaptations preserve important aspects of the motion by altering less important ones: in a walking motion, it is important that the feet touch the floor, not that the pelvis is 32 inches above the floor as in the original. The important properties of

---

*Autodesk VTC, 2465 Latham St, Mountain View, CA 94040. gleicher@cs.cmu.edu, http://www.gleicher.com/mike

a given motion may not always be simple; realism, grace, like–in–Singing–in–the–Rain–ness, or other high-level properties may be desirable to preserve during adaptation. In practice, we are limited by our ability to define high-level qualities of the motion mathematically, by our ability to compute adaptations efficiently when the metrics become complex, and by the amount of effort we wish to expend in identifying (or having the user identify) these properties. These issues motivate a more pragmatic approach to retargetting.

This paper presents a method for finding the adaptations needed to retarget motions from one articulated figure to another. We accomplish this by requiring the basic features of the motion – for example that the feet touch the floor when walking – to be identified as constraints. If the constraints are violated when the motion is applied to a different figure, we find an adaptation to the motion that re-establishes the constraints in a manner that fits with the motion. Our premise is that by maintaining the basic features and avoiding uncharacteristic (in a basic signal-processing sense) changes, we find adaptations that generally preserve the desirable characteristics of a motion, without explicitly modeling them.

The core of our retargetting method is a numerical solver that computes an adaptation to the original motion. The adaptation re-establishes the constraints while attempting to avoid adding any undesirable artifacts. Our solver is a spacetime constraints method that considers the entire motion simultaneously, computing whole motions, not just individual frames. To preserve the qualities of the original motion, we minimize the magnitude of the changes and restrict their frequency content.

After a review of previous work, we introduce our method in Section 3, and summarize the technique in Section 4. Section 5 describes how the method can be applied to creating motions when the character is changing (morphing). In section 6, we discuss issues in solving the non-linear constraint problems. We provide a gallery of examples in Section 7 and consider the problem of retargetting a motion to a character with different structure in Section 8.

1.1 An Example

We motivate our approach with an example: retargetting motion capture data of an actress walking up to, picking up, and carrying away a box. During pre-processing, we augment the motion data by specifying constraints that are essential to the action: the hands must grab the box in the middle frame, the hands must remain the correct distance apart while carrying the box, and the feet must be planted and not skid when they are on the ground.

Without adaptation, our motion capture data does not apply to figures of different sizes or proportions than our actress: the resulting motions have the feet skating and the hands failing to reach the object. Our method enables us to re-use this data on figures of varying proportions, as shown in Figure 1. The method computes an adapted motion for each new character using the approach detailed in Section 3. Because the technique looks at the entire motion, it can make adjustments based on all the requirements. For example, it adjusts the footplant positions so that the characters reach the box using natural footstep sizes.

Our approach makes many sacrifices to achieve practicality. We tell our solver little about the original motion or general motion properties, and our choice of the mathematical problem is heavily
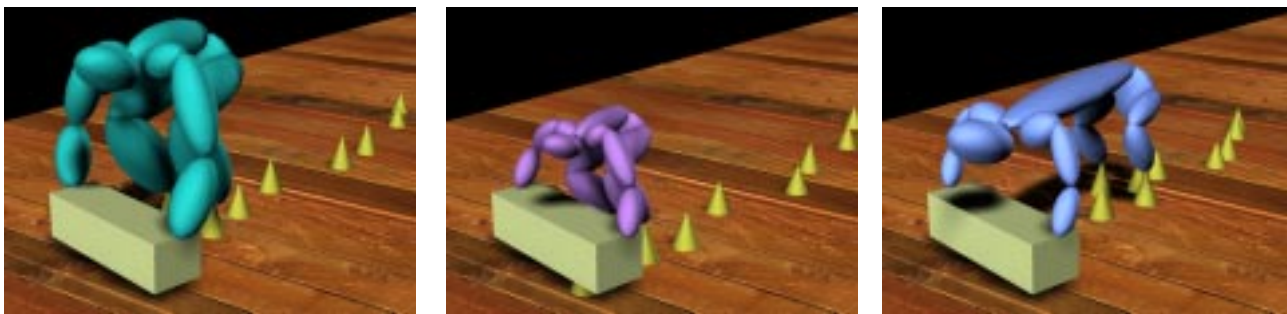
Figure 1: Differently sized characters pick up an object. Their positions are determined by the position of the object. The left shows the original actress. The center shows a figure 60% as large. The right shows a figure with extremely short legs and arms and an extremely long body. The yellow cones represent footplant positions.

influenced by what can be solved efficiently. We sometimes pay for these sacrifices in the quality of the resulting motions. For example, because our system did not consider gravity or posture we get an unrealistically unbalanced result in the right frame of Figure 1. The payoff is that our approach provides a practical solution to the retargetting problem and a framework in which to employ more sophisticated constraints, like balance, in the future.

2   Previous Work

Few techniques specifically address the retargetting problem. Generally, users are forced to adapt motions using the same tools that are used for motion creation: each frame or key must be manually tweaked. Some commercial systems, such as Kinetix's Character Studio [11], are beginning to support retargetting. For example, Character Studio can adjust keyframes to maintain footplants and balance when a motion is re-applied to a new character.

Hodgins and Pollard [9] address a variant of the motion re-use problem, adjusting parameters of a physical simulation to adapt a controller for use with a new character or a character that is changing. In general, procedural- and simulation- based approaches to animation offer representations independent of the character and therefore may be used generate new motions for new characters. Many of the procedural and simulation controllers are able to adjust to different characters easily. Such methods do not address the retargetting problem: they can generate new motions for new characters but not reuse existing motions. Re-generation of motion risks losing qualities in the original. Our goal is to create methods that adapt existing motions obtained from a variety of sources, including motion capture and keyframing as well as simulation and procedural generation.

Recently, there is an interest in tools that allow motion to be altered in ways that are independent of how it was created. At their core, these tools treat animated motions as time-varying signals and apply signal processing techniques to these signals. Litwinowicz's Inkwell system [12] first demonstrated the utility of applying signal processing methods to animation data. Perlin [17] showed how existing motions could be blended together, and how the addition of noise to a motion could be used to transform it. Bruderlin and Williams [2] showed that many signal processing techniques could be applied to motion. Simultaneously, other authors showed some of these methods in greater detail. Unuma et al. [21] showed how band-pass filtering methods could adjust emotional content, and Witkin and Popović [23] introduced motion-warping, a variant of Bruderlin and Williams' motion displacement mapping.

2.1   Spacetime Constraints

The spacetime constraint approach, introduced by Witkin and Kass [22], poses the motion synthesis problem as a constrained optimization: what is the best motion that meets a specified set of constraints? Cohen [3] extended this with a more complete system that allowed the user to focus the solution process. Recently, Rose et al. [19] applied the approach to the problem of generating transitions between motion segments, and Gleicher and Litwinowicz [7] showed how the methods can be used for adjusting motions so that the characters have new goals. Gleicher [6] extends this work by simplifying the spacetime problem to achieve interactive performance for interactive editing.

What differentiates spacetime from other constraint methods is that it poses a single large problem over a duration of motion, rather than on an individual frame. The original spacetime work, as well as most that followed, used spacetime to derive physically valid motions: constraints enforced Newton's laws, and the objective function minimized energy consumption. Previously, we [6] have suggested removing the physical constraints to achieve better performance and to apply the techniques to non-physical motions.

Although Ngo and Marks[15] re-used the term spacetime constraints to describe their work, their method belongs to a different family of approaches that generates control systems that create motions, rather than generating the motions themselves. We prefer to reserve the term *spacetime constraints* for methods that compute specific motions.

3   An Approach to Retargetting

In this section, we motivate and describe our approach to retargetting the motion between articulated figures with identical structure but different segment lengths. We assume that the configuration of an articulated figure is specified by a position for the root of the hierarchy and the angles of its joints. We will denote these configurations as a vector that concatenates all of these parameters, often denoted by $\mathbf{q}$, or by $\mathbf{q}^t$ to refer to its value at time t. A motion is a vector-valued function that provides a configuration given a time. While we often represent the initial motion as a dense array of samples or as a set of key values that are interpolated, our methods are independent of how this motion is obtained. We refer to the retargetted motion as $\mathbf{m}(t)$, and often use the concept of a motion displacement which represents the difference between two motions, e.g. $\mathbf{m}(t) = \mathbf{m_0}(t) + \mathbf{d}(t)$.

Because the target character has the same parameters as the original, reusing the original motion data will cause the new character to move its limbs as the original, but not necessarily lead to a desirable result as shown in the example of Figure 2. Because the length of the limbs are different, the parts of the new character do
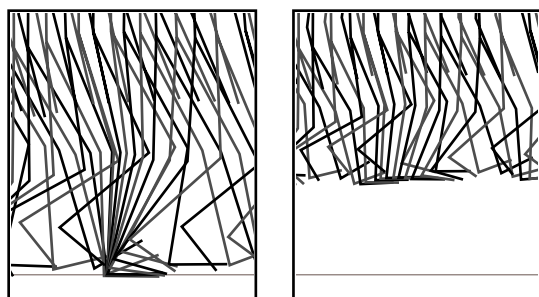
Figure 2: Left: Frames from a rotoscoped walking motion are shown. Right: Applying this motion to a character that is 60% of the size of the original yields a motion that skates along horizontally above the floor.
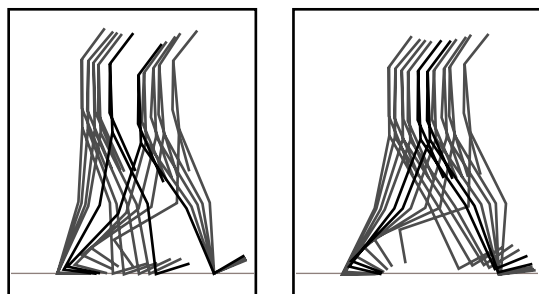

Figure 3: Adaptations are applied to the motion of Figure 2 to re-establish the constraints. The figure shows five frames before and after a heel strike, with the frame immediately before and after the heel strike darkened. A constraint on the heel's position applies on the frames after the strike. Left: inverse kinematics is applied to individual frames, causing a noticeable discontinuity. Right: our approach re-establishes the constraints while maintaining the frequency characteristics of the original motion.

not end up in the same place as in the original. Therefore, they may fail to interact correctly with other objects in the world or may move differently. In the example these problems appear as the feet not touching the floor and "skating" horizontally when planted, as seen in Figure 2. The naive retargetting fails to preserve important properties of the initial motion.

3.1 Inverse Kinematics

The principal problem with the naively retargetted motion is that it violates some of the constraints that we expect in a satisfactory walking motion. For example, a walking motion requires character's feet to touch the floor and to not skid during footplants. Retargetting must re-establish these constraints.

Inverse kinematics (IK) is a common technique for positioning end effectors of articulated figures in individual frames of an animation. An IK solver could be used to adjust the configuration of the character to meet the constraints in each frame. Figure 3 shows the result of such a retargetting approach, re-establishing the planted foot positions. Because the IK solver considers each frame independently, it makes different alterations to each frame. This lack of consistency adds many undesirable artifacts to the motion. For example, because frame i does not know that a foot will be planted in frame $i + 1$, it cannot move towards this constraint, so that in frame $i + 1$, the foot will snap to its new location. Even within a footplant, there is a lack of consistency: on each frame the solver will use a different combination of straightening the leg and lowering the pelvis. These artifacts appear as high frequency "jerkiness," shown for the example in Figure 3.

3.2 Motion Frequency Response

The problem with the IK solution is that we have added high frequencies to a primarily smooth motion. Extending the leg from bent to straight in $1/15th$ of a second might be acceptable if this were a karate master's kick, but, this discontinuity is inappropriate in our walking motion. Generally, the high frequencies of a motion (or the lack thereof) are important, and therefore must not be disturbed. An adaptation that removed the snap from a karate kick might be just as inappropriate as adding the snap to our slow walking motion.

The importance of preserving the high-frequency content of a motion (or the lack thereof) is an explanation for the success of motion-displacement mapping [2, 23] (also called motion-warping) techniques.[1] The key spacing of the displacement curves restricts their frequency content such that the high frequencies of the motion are not disturbed.

Changes should not necessarily be made at the lowest possible frequency. Consider retargetting a motion where a smaller character must grab an object in the middle frame, but there are no other constraints on the arm. To meet the constraint, the character must extend his arm in this one frame. This alteration can be made at any frequency: the single frame can be adjusted (e.g. the arm shoots out for the $1/30th$ of a second), or the adjustment can be applied to the whole motion (e.g. the arm is extended while the character walks up to the object to pick it up). While the extreme high-frequencies of the former are undesirable, so are the extreme low frequencies of the latter (the added signal has only a DC component).

A simple approach to avoiding the addition of high frequencies is to low-pass filter the displacement signal generated by the inverse kinematics process. Unfortunately this change does not necessarily maintain the constraints that IK was used to achieve as shown in Figure 4.

3.3 Motivating Spacetime

The failure of the per-frame approach to meet the needs of automatic retargetting suggests that we require a constraint-based method that can take into consideration a span of the motion, e.g. spacetime constraints. The more global view of such a method allows it to consider relationships among multiple frames. Spacetime constraint's use of constrained optimization allows us to address both parts of the retargetting problem: establishing the constraints on the motion, while minimizing the changes our original motion.

The spacetime constraints approach poses the retargetting problem mathematically. We seek a motion $\mathbf{m}(t)$ that, subject to satisfying a set of constraints on the motion $\mathbf{f}(\mathbf{m}(t)) = 0$ and $\mathbf{f}_i(\mathbf{m}(t)) \geq 0$ (we divide the constraints as equality and inequality constraints for notational convenience), minimizes an objective function $g(\mathbf{m})$. For retargetting, the objective compares the motion with the original motion, $\mathbf{m_0}(t)$. By encoding the retargetting problem in this form, we can use numerical methods to solve the constrained optimization problem for our desired result.

Because the spacetime approach looks at the entire motion, it can make choices based on other parts of the motion. For example, it can move footplants based on where the character needs to end up. Such look-ahead and -behind is not possible in approaches that consider each frame independently.

3.4 Spacetime in Practice

Ideally, the constrained optimization problem would fully encode our desires mathematically: there would be a single solution that was the desired motion. Realizing this ideal requires a rich set of constraints and objectives. For example, we could find constraints that enforce the laws of physics, biomechanical limitations

---

[1]Albeit, one that is not emphasized in [2] but is a motivation for [23].
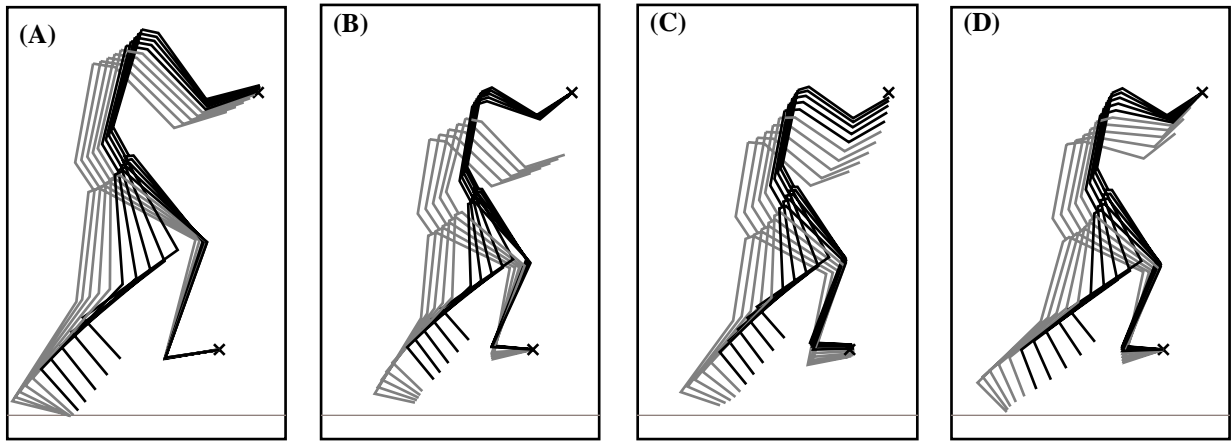
Figure 4: Ten frames of a ladder-climbing motion are shown. In the last 5 frames (shown darker), the hand is constrained to be attached to the handhold. (A) shows the original motion capture data. (B) shows the motion adapted to a smaller character by applying Inverse kinematics (IK) to each frame, causing a noticeable snap. (C) shows low-pass filtering of the results of the IK process. This removes the snaps at the expense of violating the constraints. (D) shows our approach applied to the example.

due to strength, and proper ballet form. We could define objective functions that measure visual properties such as "grace," "Charlie–Chaplin–ness," and "like–Joe–did–it–yesterday–ness." We would aim to maintain the constraints that were satisfied in the original motion while minimizing the amount of change in the important properties.

There are central difficulties in realizing the spacetime ideal for retargetting: first, some properties are difficult to encode mathematically as constraints or objectives either because the forms of the equations are complex or because they elude a mathematical encoding; second, we may not know all the properties required, such as the mass distribution of an imaginary character or the physical laws of an imaginary world; third, we must decide which properties are important in a given setting; fourth, many of the properties and constraints may be specific to a small set of examples, and therefore not worth the effort to define.

Even if we encoded the desired animation completely in a constrained optimization, we still need to find the solution to these problems. Generally, richer sets of constraints and objective functions are likely to lead to more difficult problems to solve. The challenges of solution lead us to take a pragmatic view in defining spacetime problems. An extreme case of this pragmatism is our work on spacetime editing [6] where many sacrifices were made in order to achieve interactive performance.

Our approach to spacetime for retargetting is motivated by the pragmatic issues of defining, specifying, and solving constraints and objectives. We use constraints to define specific features of the motion that must be maintained and use the objective function to limit certain generally unacceptable types of changes. Besides the constraints and objectives, we have two more pragmatic tools that we can use to help define a spacetime problem with the desired solution: the representation used for the motion and the starting point for the constrained optimization. We will discuss these four in more detail in the following sections.

3.5 Sources of Constraints

Constraints are the primary tool used to identify features of the original motion that must be present in the retargetted result. In general, our constraints will either come from restrictions on the character (e.g. the elbows do not bend backwards), the environment (nothing should be below the floor), or the motion (the character must pick up the box in frame 50). Specification of these constraints typically involves only a small amount of work in comparison with

the tasks of creating the characters and motions, especially with semi-automatic detection (for example finding footplants), graphical specification, and generic constraints (e.g. we use the same joint limits for most humanoid characters). Constraints are generally defined once for each motion, and this one set of constraints is used for any retargettings (or editing, using the techniques of [6]) done with the motion. Even with these tools, augmenting our characters and motions with constraints does require some additional work. However, we feel this incremental effort is worthwhile because of the potential for reuse afforded by augmentation.

Mathematically, constraints are differentiable functions of the parameters of the character. Although it is not required by the methods, our implementation always places constraints on configurations at particular instants of time. Variational constraints, that is constraints that are to hold over a range of the motion curves, are approximated by sampling. Therefore, constraints are generally written as $f(\mathbf{q}^{t_i}) \diamond c$, where $\diamond$ is $\{\leq, \geq, =\}$ and c is a constant. Some constraints consider two instants in time, and therefore have the form $f(\mathbf{q}^{t_i}, \mathbf{q}^{t_j}) \diamond c$.

In our system, the user never needs to see an equation: the system includes a variety of pre-defined constraints that can be applied to a motion through a graphical user interface or via a scripting language. We have emphasized finding (and using) constraints that we believe are applicable over a wide range of motions. Some of these include:

1. a parameter's value is in a range (useful for joint limits);

2. a point on the character (such as an end-effector) is in a specific location (useful for footplants or grabbing an object);

3. a point on the character is in a certain region (for example, above the floor);

4. a point on the character is in the same place at two different times (useful to prevent skidding), although this position is unspecified so that it can be adjusted;

5. a point on the character is following the path of another point;

6. two points are a specified distance apart (useful for when a character is carrying an object of a fixed size);

7. the vector between two points has a specified orientation.

The architecture of our system is designed to minimize the effort required to add new types of constraints, although this does require programming and must be done at compile-time.

In developing a new type of constraint, it is important to make restrictions in ways that are invariant of other aspects of the motion. For example, if one defines a footplant by the positions of the heel and toe strikes, the constraint cannot be satisfied if the foot size is changed. Similarly, we often do not care where a footplant is, providing that is is on the floor and that the foot does not skate while planted. For the examples in this paper, we will distinguish between footplant constraints that maintain the position on the floor and those that only restrict height and skating. When the solver is permitted to move footplants, the resulting motion may cover a different distance, e.g. if the footsteps of a walk are made smaller, the character will travel a shorter distance since the system does not generate new footsteps.

3.6 Objective Functions

Since there are typically many possible motions that satisfy the constraints, we use an objective function to select the best choice. For retargetting, a simple objective is "minimize the amount of noticeable change." This does not necessarily lead to a simple, generic manifestation: consider a ballet motion where a very slight bend of the knee might be a very noticeable deviation from the otherwise perfect form of the original with its straight leg. However, our strategy is to use constraints to prevent specific changes that are unwanted, and use the objective function to avoid undesirable frequency content and unnecessary large alterations, as discussed in Section 3.2. We avoid designing objective functions tuned to specific high-level goals.

The most basic comparative objective function would be to compare the values of the parameters, matching pose in parameter space. For example,

$$g(\mathbf{m}) = \int_t (\mathbf{m}(t) - \mathbf{m}_0(t))^2 = \int_t \mathbf{d}(t)^2, \qquad (1)$$

minimizes the magnitude of signal differences in the motions over time. This objective is similar to performing per-frame inverse kinematics as it provides no coupling between constraints at different times. The minimum magnitude solution effectively maximizes high frequency content. Intuitively, it prefers not to "waste" change preparing to meet goals at other times. Other frequency criteria can be implemented with an objective function that minimizes the output of a filter that selects undesirable frequencies.

In practice, we find that pragmatic concerns outweigh most other choices in the design of an objective function. For the experiments described in this paper, we use the objective function to minimize the magnitude of the changes, approximating Equation 1. Methods described in the next section restrict high frequency content of the changes. This tactic affords the use of more efficient solving techniques (as we will describe in Section 6).

3.7 Representation

Another issue in a spacetime approach is how to represent the motions so that the optimization problems can be solved effectively. Liu et al. [13] first made use of a carefully selected representation by using wavelets to speed computations. Gleicher and Litwinowicz [7] introduced the use of motion-displacement maps as a representation for spacetime problems where the objective function related two motions. This approach defines

$$\mathbf{m}(t) = \mathbf{m}_0(t) + \mathbf{d}(t)$$

and uses the solver to find $\mathbf{d}(t)$. The approach has a number of advantages. First, it decouples the solution from the form of the initial motion, providing generality. Secondly, it simplifies placing constraints and objectives on the changes. Third, it allows a representation for $\mathbf{d}(t)$ to be chosen that includes constraints on the changes so they do not need to be expressed as explicit functions.

To constrain the displacement signal not to include high frequencies, we use a representation for it that cannot represent the high frequencies: specifically, cubic B-splines [14] with control point spacing determined by the desired frequency limits. The control points of the displacement curve need not be uniformly spaced: we can place controls closer together for portions of the motion where higher frequencies are acceptable. Similarly, we do not need to use the same key spacing for all parameters, for example, if a chef is chopping, we might allow high frequencies in the motion of his arm (to accommodate the abrupt motions of the knife), and only permit smoother changes to the rest of his body.

The spacing of B-spline control points allows us to determine the frequency response of our adaptations, although we do not have the fine control afforded by carefully crafted filters placed in an objective function. We must determine how to place the control points to achieve the desired effect. For our experiments, we have limited our choices to using the uniformly spaced control points on all parameters of a motion.[2] For the examples in this paper, we further restrict ourselves to control points spaced every 2, 4 or 8 frames. We have developed a simple heuristic method for determining which of these to apply: we compute a bandpass decomposition of the original motion (as described in [2]) and choose the key spacing that coincides with the lowest, that is highest-frequency, level of the pyramid whose energy contribution exceeds a threshold. While this simple heuristic has resulted in the correct recommendation for almost all of our examples, the speed of our solver makes it practical to produce all three adaptations and to select the one that gives the most appealing result.

With the constraints imposed by the restricted representation, there may not be a solution to the constraints. In such cases, there is a fitting problem: find the frequency-limited signal that comes closest to satisfying the constraints (where the constraints are the explicit equations from Section 3.5). In such a scheme, the nature of the mathematical problem is flipped: our constraint is the frequency response, and our optimization objective attempts to minimize the residual of the constraints. We use a least-squares metric for the residual which enables simpler solution methods, as we will discuss in Section 6.

3.8 Starting Points

Cohen [3] pointed out the importance of having good starting points for spacetime problems. Seitz and Dyer [20] observed the utility of a previously captured motion as a starting point for speeding their numerical solutions. With our retargetting approach, the initial estimate of the solution is even more critical because our simple objective function explicitly defines the result in terms of the initial estimate. To improve the quality of our results, we must apply some simple transformations to the original motion so it better estimates the desired result. The process described in this section is summarized in Figure 5.

Simply re-using the initial motion is possible because our figures share the same parameters. For articulated figures, most of the parameters are angles and are independent of the scaling of the limbs: the angular value for a straight leg is the same, no matter how long the thigh and calf are. However, the positional offset of the root of the hierarchy is not scale-independent. The translation is a distance

---

[2]This was problematic only for the example of Figure 1 where the footsteps have different frequency content than the grabbing motion. The artifacts of this problem are subtle.
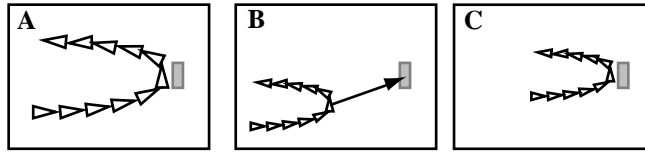
Figure 5: A: An abstracted aerial view of a character walking up to, picking up, and carrying away an object. B: When the motion is scaled about the origin (the lower left corner of the frame), the character does not come close to the object. C: Because the position of the object is the only constraint that specifies a position for the character, the entire motion can be translated.

(from the origin), and therefore should be scaled as the limbs were. Such scalings are difficult to create with the additive displacement maps, so we perform the scaling as a separate step. If the character is scaled uniformly and does not interact with the world (or if the world is scaled similarly), the scaling is sufficient for retargetting. In cases where the character is scaled non-uniformly, we make an estimate of the overall scaling to apply to the positions.

Multiplying the positional parameters scales the motion around an arbitrary point, the center of the coordinate system. Typically, there is a better center for the scaling. For example, we might scale the heights around the floor, which may not be zero. We recenter the scaling of the positional parameters by adding a translational component to them.

To find the translation, we note that a constant positional shift of a motion is not noticeable, except in conjunction with constraints that relate the character to the world. Therefore, if we could re-establish the constraints by a simple shift of the motion, this would be ideal. We find the shift of the motion that comes closest to re-establishing the constraints by computing the average of the displacements. By displacement, we refer to the vector between the point on the character and a position that it is attached to. Constraints only have displacements for axes that they restrict with a fixed position, for example, a footplant constraint may only specify the vertical direction if it only places non-skid restrictions on the other axes.

Since the center of scaling might not be constant over the whole motion, we compute a translational signal to add to the positions. We perform the displacement averaging process on each frame individually. Adding the per-frame constraint displacements to the motion may add undesirable high frequencies. Therefore, we interpolate the offsets to frames that do not have any displacements and apply low-pass filtering to remove high frequencies.

The utility of interpolation can be seen in the example of Section 1.1 where a figure walks up to, picks up, and carries away an object. In this example, the only constraint on the figure's position on the floor is provided by the constraint that the hands touch the object on the middle frame of the motion. When the motion is scaled, the entire motion is moved far away from the goal point. Interpolating the displacement of this one constraint shifts the entire motion back to the object, as shown in Figure 5. The desirability of constant shifts is unique to position; for angles it can have the undesirable behavior described in Section 3.2.

4 The Motion Retargetting Method

To summarize, our approach to retargetting motion to another articulated figure with different limb lengths consists of the following steps:

1. Begin with an initial motion with identified constraints.

2. Find an initial estimate $\mathbf{m_1}(t)$ of the solution by scaling the translational parameters of the motion, and then adding a translation to define the center of scaling. This translation is computed by finding the constraint displacements of the

scaled motion for the target character, interpolating these values, and smoothing.

3. Choose a representation for the motion-displacement curve based on the frequency decomposition of the original motion.

4. Solve the non-linear constraint problem for a displacement that when added to the result of step 2 provides a motion that satisfies the constraints.

5. *(optional)* If the result of step 4 does not satisfy the constraints sufficiently, solve using the result of the step $(\mathbf{m_1}(t) + \mathbf{d}(t))$ as the initial motion, and a denser set of control points for the new displacement.

5 Motion for Morphing

The same methods that are used to adapt a character to new segment lengths can be used when the target lengths are not constant, i.e. when the target character is morphing. A simple example of a motion generated for morphing is shown in Figure 6. A more complex example is shown in Figure 10.

The difference between motion for morphing and standard retargetting is that the segment lengths of the target character is not constant over the motion. Therefore, it is better to use a different scaling amount on each frame in Step 2. As with the constant case, we estimate the scale in the event that the limb scalings are non-uniform. To apply this time-varying scale to the character's position, we scale the changes in translation between frames by the scale of the character in the frame, and add these changes together to find the characters positions.

6 Solving the Non-Linear Optimization

The key computation of the retargetting approach is the solution of the spacetime constraint problem. In this section, we briefly discuss our solver implementation. We emphasize that our approach casts retargetting as a standard mathematical problem, constrained optimization, for which there is a rich literature of solution methods. For a more detailed discussion of solution methods, we suggest a text on the subject such as Fletcher [4] or Gill et al. [5].

For simplicity of our discussion, we consider only equality constraints as we implement inequality constraints using an active set method [4] that creates inequality constraints by switching sets of equality constraints on and off. The constrained optimization problem we solve is generically:

$$\text{minimize } g(\mathbf{x}) \text{ subject to } \mathbf{f}(\mathbf{x}) = \mathbf{c}. \tag{2}$$

The unknown in our spacetime problem is the motion-displacement curve, or more precisely, the values for the B-Spline control points of the displacement curve. The vector of parameters $\mathbf{x}$ is the concatenation of these points. We must express all of the constraints and objectives in terms of these variables, and solution methods require us to compute the values and derivatives of
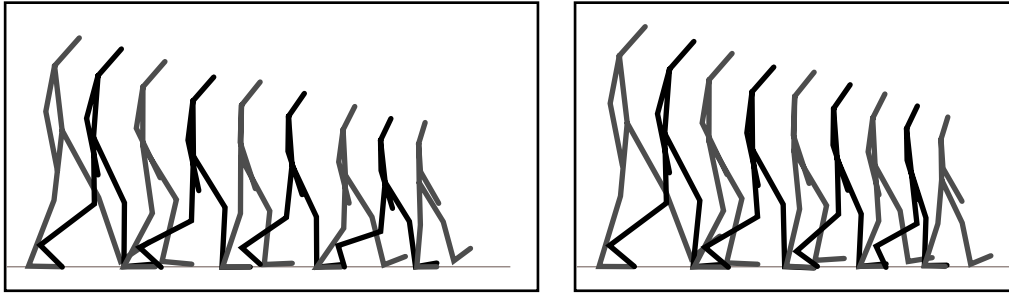
Figure 6: The retargetting process is used to adapt the motion of Figure 2 as the character morphs to 60% of its original size. Left: the footplant positions are fixed to be the same as the original motion. Right: the solver repositions the footplants.

these functions. We approximate the objective of Equation 1 as a weighted sum of squares of the controls

$$g(\mathbf{x}) = \frac{1}{2}\mathbf{x}\mathbf{M}\mathbf{x} \tag{3}$$

where $\mathbf{M}$ is a diagonal matrix. We usually compute the entries in $\mathbf{M}$ to account for differing sensitivities in the variables as described in [6] and [7]. The importance of the choice of $\mathbf{M}$ is reduced by the large number of constraints, both explicit in equations and implicit in the representation, in the retargetting problems.

Since our constraints are always defined on instants of time, the sampling of the continuous variational problem is implicit in their definition. While the expressions for individual constraints may grow complicated, we note that they are composed of smaller pieces that are more manageable. For example, a constraint specifying the height of a characters foot would combine the kinematic function that takes the character's parameters and returns the foot height $f_k(\mathbf{q})$ composed with the function that computed the value of the parameters at the instant of time in question $\mathbf{q}^{t_i} = \mathbf{m_0}(t_i) + \mathbf{d}(t_i)$, which in turn must sample the B-splines $\mathbf{d}(t_i) = \mathbf{b}(t, \mathbf{x})$. Through the use of automatic differentiation [8, 10], we can construct these pieces independently.

Most previous spacetime work has used constrained optimization solvers that are variants of sequential quadratic programming (SQP). This standard method is described in texts such as [4], as well as spacetime papers such as [22] and [3]. In [6], we provided a variant of SQP that is more efficient for cases where the objective function has the special form of Equation 3. Our system includes solvers that operate both ways.

An alternative solution approach focuses on minimizing the constraint residual $r = 1/2(\mathbf{f}(\mathbf{x}) - \mathbf{c}) \cdot (\mathbf{f}(\mathbf{x}) - \mathbf{c})$ (because of the implicit constraints of the representation, it is unreasonable to expect that there will be an exact solution to the explicit, equational constraints). Because the constraints may not fully determine the solution, for example on a walking motion the legs may be over determined while there are no constraints on the arms, we add additional constraints that specify that each variable should have a zero value. These constraints receive a smaller weighting. Such problems are called damped least-squares problems [5, 16], and can be solved by performing an unconstrained minimization on the residual

$$r = \frac{1}{2}(\mathbf{f}(\mathbf{x}) - \mathbf{c}) \cdot (\mathbf{f}(\mathbf{x}) - \mathbf{c}) + \epsilon \frac{1}{2}\mathbf{x} \cdot \mathbf{x}, \tag{4}$$

where ϵ is a small constant, or a diagonal matrix of weights.

Our non-linear least-squares solver iteratively improves on an estimate of the solution. At each step, we construct a linear approximation of the constraint problem using Taylor expansion around the current estimate for $\mathbf{x}$,

$$\mathbf{f}(\mathbf{x} + \boldsymbol{\Delta}) \approx \mathbf{f}(\mathbf{x_i}) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \boldsymbol{\Delta},$$

which gives us a linearized version of the constraint equations,

$$\mathbf{J}\boldsymbol{\Delta} = \mathbf{f}(\mathbf{x}) - \mathbf{c}.$$

This linear least-squares problem can be solved in a variety of ways. We solve for $\boldsymbol{\Delta}$ using a damped pseudo-inverse

$$(\mathbf{J^T J} + \epsilon\mathbf{I})\boldsymbol{\Delta} = \mathbf{J^T}(\mathbf{f}(\mathbf{x}) - \mathbf{c}). \tag{5}$$

Because Equation 5 is a positive definite linear system, we can solve it efficiently using either a Cholesky decomposition [18] or conjugate gradient solver[1]. We use the latter exclusively as it allows us to exploit the sparsity in the matrix to achieve good performance.

In both our constrained-optimization and least-squares solvers we use a line search [18] to determine how to use best the results of the linear subproblem. That is, once we compute $\boldsymbol{\Delta}$, we determine a value of k such that $\mathbf{x} + k\boldsymbol{\Delta}$ best satisfies the non-linear constraints.

In most cases, we find the least-squares solver to be faster than either of the SQP style solvers while providing equivalent results. For the rest of the paper, we will refer to the solvers as SQP (for the solver similar to that described in [3]), LMULT (for our implementation of the method in [6]), and least squares (for the pseudo-inverse based solver). The running times of the iterative methods used in our solvers depend on many factors, including number of variables, number of constraints, sparsity, and desired stopping tolerance. Small changes, especially in tolerance, can cause dramatic changes in solver times.

7 Examples

We have used the retargetting approach of this paper on a number of examples. While there is nothing specific to motion capture data in our approach, our examples are exclusively done on performance data because of its availability. Other than the rotoscoped 2D walking motion of Figure 2, the motions in this paper were captured with an optical motion capture system at a commercial studio. In all examples, the 120 Hz motion capture data was downsampled to 30Hz. Marker positions were converted to articulated figure parameters using our experimental automated software.

Because of the differences in processing technologies, we have some diversity in the parameters for the figures in different motions. In all cases, we use Euler angle representations for the joints. We do not have positional information for the hands. Therefore, we treat the end of the forearm as the "hand." Similarly, some motion data is missing information for the feet, in which case the ankles are used as the end effectors. For many of the motions, we did not compute the head and neck parameters as they do not affect the computations. Joints generally have three degrees of freedom, except for the elbows, knees, and ankles which have one or two parameters.
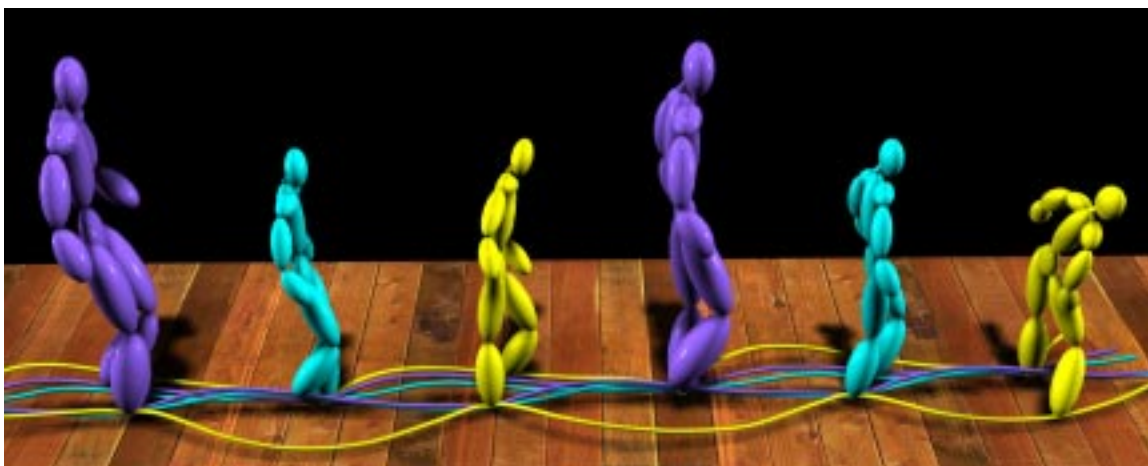
Figure 7: A walk adapted to a figure 60% of the size of the original actor. The smaller character is forced to use the original footplant positions. When the displacement keys are too distant, overfitting causes the wide swings shown in the alternate (yellow) foot traces. Proper key spacing (blue) results in a motion similar to the original (purple).

When given, timing information refers to our prototype system running on an Apple Power Macintosh 8500/180 computer with a 180Mhz PowerPC 604e processor and enough physical memory to complete the retargetting without paging. Timings are reported for the task of solving the non-linear optimization as the other parts of our retargetting approach take negligible amounts of computation.

7.1 Walking

The initial 2D walking motion of Figure 2 was created by rotoscoping marker points and using a capture process like that described in Section 8 to compute the parameters of the articulated figure. Our character has 14 degrees of freedom (2 for position and 12 joint angles), and the motion is 15Hz. On the 82 frame motion, footplant constraints on the heels and toes give 146 scalar constraints, to which we add 328 inequality constraints to keep the feet above the floor in each frame, and 1968 joint limit constraints.

Our 3D walking example is similar. The character has 34 degrees of freedom, and does not have hands or feet. Because the "feet" in the motion are actually ankles, they were not planted in the original motion and skated. We therefore used our solver to establish these constraints initially. Including joint limits and feet-above-floor constraints there are a total of 4193 scalar equations on the 112 frame motion, although during solving there are generally only 354 active constraints.

We have adapted the walking motion to a number of differently proportioned figures. An example is shown in Figure 7. With fixed footplant positions to match the tall figure, the shorter legged figures must take unnaturally long strides, seen in Figure 8. As predicted by the pyramid level heuristic of Section 3.7, a key spacing of 4 provides a better result to spacings of 8 or 2. With a key spacing of 8 there is considerable over-fitting that can be clearly seen in the yellow foot path traces of Figure 7. A key spacing of 2 provides a motion that is reasonable, however, the character seems to slow down with each step. While this is different from the original motion, the character is taking very large steps, so it seems natural for it to regain its balance each time. Our system was able to generate all 3 motions in under 10 seconds of solution time, so it is practical to create all 3 motions and choose the one we find visually most desirable.
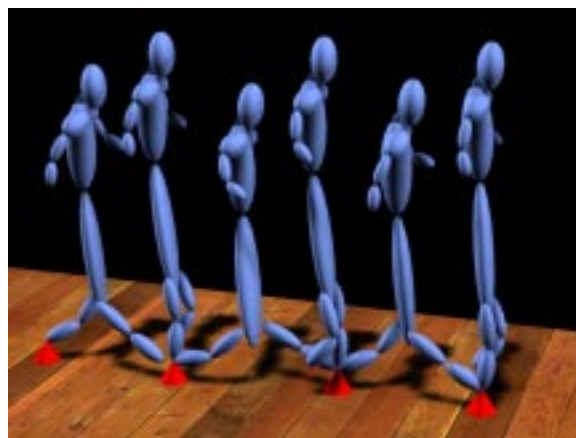

Figure 8: Forcing a character with short legs to walk in the footsteps of a longer-legged character leads to an unnatural motion.

7.2 Climbing a Ladder

The ladder example, shown in Figure 4, gives constraints on both hands and feet. The figure has 35 degrees of freedom, no hands, and no neck or head. We use fixed position constraints for the footplants and handplants on the ladder. The least-squares solver takes approximately 9 seconds for keys spaced every other frame, and 7 seconds every fourth frame. The LMULT solver takes 6 and 4 seconds, although its answers do not satisfy the constraints as accurately. With the key spacing of 4, the LMULT solution has some constraints being violated by over half an inch, while the least-squares solution satisfies all constraints to within a quarter of an inch.

The fixed position of the hand and footplants on the ladder lead to slightly unnatural motions: the small figure must reach over its head to grasp the handholds and sometimes stands on its tip-toes to reach. We have implemented some less restrictive constraints: footplants that the solver can move along the ladder step (so the width of the steps is not an issue) and hand-holds that can be positioned along the rail. These constraints are relatively special purpose: they probably will be useful for ladder climbing motions. The motion obtained from using these constraints more closely resembles the original motion, although it is still unnatural as the ladder is very
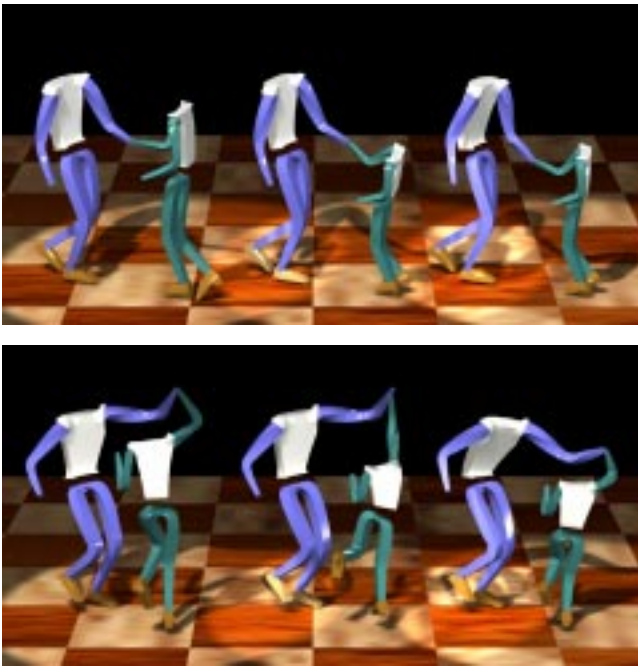
Figure 9: Two frames are shown of a swing dance motion adapted to a smaller female character. Left: original motion. Center: only female motion adapted. Right: both characters adapted.

large in comparison to the resized character.

7.3 Swing Dancing

When there are two characters in a scene, we may wish to adapt both together, even if only one changes size. For example, consider the swing dance motion in Figure 9. In this motion, the hands of the two characters must remain connected, in addition to the foot-plant constraints. If we change the size of the female figure without changing the motion of the male figure, the smaller figure gets lifted by the hand-hold when spinning. If we adjust both motions simultaneously, the male's part is adapted, and the female's spin is less noticeably forced. In Figure 10, the female shrinks in size while spinning and the male part responds accordingly.

On the 276 frame motion, we use 1200 equality constraints for the female character's footplants (which are free to be repositioned by the solver) and the connection between the characters hands. We only allow the upper body of the male character to be altered. If we adapt just the female motion, there are 33 parameters. Adapting both motions gives 44 parameters per key. The least-squares solver took approximately 14 seconds, while the LMULT solver ran for slightly over a minute, but with a solution that better satisfies the constraints (all to within an eighth of an inch).

8 Differing Characters

When the characters share structure there is a direct mapping between the parameters of one to the other. The more general retargetting problem is harder. When we apply a human motion to a figure with a different structure, there are creative choices in how the motion applies. What will the character use for knees? How do we choose a motion for the parts of the character that the human does not have? These creative choices correspond to mathematical problems: there may be different types of degrees of freedom, and there may be different numbers of degrees of freedom.
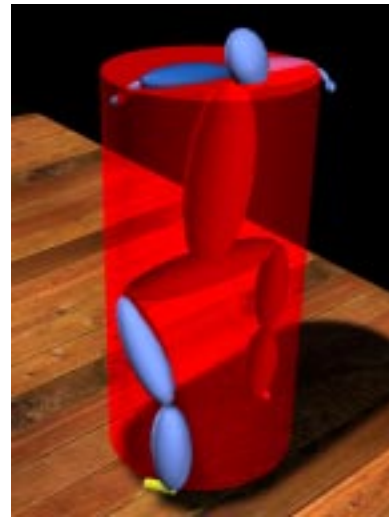

Figure 11: A walking motion is adapted from a human to a soda can by first adapting it to a human with the proportions of a can, then using this motion to drive the motion of the can (shown transparently surrounding the humanoid).

Our initial attempts at "automatic anthropomorphism" allow the user to make the creative choices, while having the system do the more tedious aspects. The user identifies correspondences between externally visible features of the characters, not the degrees of freedom that determine their positions. For example, we identify points on the new character that will serve as its feet when it walks, even if the foot is not at the end of a two-segment leg like the human. These correspondences pose a constraint problem, almost identical to the problem of motion capture processing: we must compute a motion that puts the character's features in the right location in each frame.

We can use the same spacetime constraints techniques that we have used for retargetting for the anthropomorphic case. Our constraints connect each feature on the new character to its corresponding feature on the original in each frame. If there are fewer degrees of freedom on the character, the motions will not be able to match exactly, and we find the "best matching motion" in a least squares sense. We have not yet developed a method for handling extra degrees of freedom.

For the spatial correspondences to apply, the characters must be approximately the same size. We use the retargetting methods of this paper to adapt the initial human motion to a new figure that has proportions more similar to the target character. We then use this motion as the source of constraints to compute the target motion. Figure 11 shows an example in which we adapt a human motion to a rigid can (a cylinder with the same proportions as a soda can). We correspond three points on the can to the human: the ends of the legs are connected to points on the bottom of the can, and the center of the hips is attached to the center of the can. Even with the can's extremely limited degrees of freedom (it is a rigid body), it can convey a sense of the original human motion. In our tests, we have made the can walk, skip, and run.

9 Discussion

In this paper we presented an approach to retargetting motions from one character to another by posing the problem of computing an adaptation as a constrained optimization. To realize the approach in a practical manner, we used geometric constraints and a simple objective function. This pragmatic strategy dodges difficulties in

Figure 10: The female character morphs into a smaller character during her spin.

using spacetime constraints. We compute retargettings of complex motions despite: not having developed mathematical encodings of concepts such as "grace" and "Charlie–Chaplin–ness" in motion; not having presented too many choices of constraints and objectives to users; and not having solved optimization problems for which we do not have efficient solution methods.

While our pragmatism pays off in the practicality of the method, we sometimes pay a cost in the quality of the resulting motions. Some of the problems we see are artifacts of the specific simple objective we have chosen and our reliance on simple frequency limits on the adaptations. For instance, in the example of Figure 1 the balance between reaching, bending, and positioning is chosen by artifacts of the representation of the character's configuration and different spatial frequencies in reaching and walking make selection of a single frequency limit for the adaptation problematic. Other problems occur because we have no guarantees on the many properties we do not explicitly model in our constraints and objective. For instance, our lack of physics constraints can lead to unrealistic situations like Figure 8 and the right image of Figure 1. Richer sets of constraints and objective functions, combined with improved solvers for the resulting numerical problems and techniques to avoid the burden of specification, would cause our approach to provide better results for a wider range of motions.

Acknowledgments

References

[1] Richard Barrett, Michael Berry, Tony Chan, James Demmel, June Donato, Jack Dongarra, Victor Eikhout, Roldan Pozo, Charles Romine, and Henk van der Vorst. *Templates for the solution of linear systems: Building Blocks for Iterative Methods*. SIAM, 1994.

[2] Armin Bruderlin and Lance Williams. Motion signal processing. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 97–104, August 1995.

[3] Michael F. Cohen. Interactive spacetime control for animation. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 293–302, July 1992.

[4] Roger Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, 1987.

[5] Phillip Gill, Walter Murray, and Margaret Wright. *Practical Optimization*. Academic Press, New York, NY, 1981.

[6] Michael Gleicher. Motion editing with spacetime constraints. In Michael Cohen and David Zeltzer, editors, *Proceedings 1997 Symposium on Interactive 3D Graphics*, pages 139–148, apr 1997.

[7] Michael Gleicher and Peter Litwinowicz. Constraint-based motion adaptation. *Journal of Visualization and Computer Animation*, to appear.

[8] Michael Gleicher and Andrew Witkin. Supporting numerical computations in interactive contexts. In Tom Calvert, editor, *Proceedings of Graphics Interface '93*, pages 138–145, May 1993.

[9] Jessica Hodgins and Nancy Pollard. Adapting simulated behaviors for new characters. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, pages 153–162, August 1997.

[10] Masao Iri. History of automatic differentiation and rounding error estimation. In Andreas Griewank and George Corliss, editors, *Automatic Differentiation of Algorithms: Theory, Implementation and Application*, pages 3–16. SIAM, January 1991.

[11] Kinetix Division of Autodesk Inc. Character studio. Computer Program, 1997.

[12] Peter C. Litwinowicz. Inkwell: A $2\frac{1}{2}$-D animation system. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 113–122, July 1991.

[13] Zicheng Liu, Steven J. Gortler, and Michael F. Cohen. Hierarchical spacetime control. In Andrew Glassner, editor, *SIGGRAPH 94 Conference Proceedings*, Annual Conference Series, pages 35–42, July 1994.

[14] Michael Mortenson. *Geometric Modelling*. John Wiley & Sons, second edition, 1997.

[15] J. Thomas Ngo and Joe Marks. Spacetime constraints revisited. In James Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 343–350, August 1993.

[16] Christopher Paige and Michael Saunders. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, March 1982.

[17] Ken Perlin. Real time responsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):5–15, March 1995. ISSN 1077-2626.

[18] William Press, Brian Flannery, Saul Teukolsky, and William Vetterling. *Numerical Recipes in C*. Cambridge University Press, Cambridge, England, 1986.

[19] Charles F. Rose, Brian Guenter, Bobby Bodenheimer, and Michael F. Cohen. Efficient generation of motion transitions using spacetime constraints. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 147–154, August 1996.

[20] Steven Seitz and Chuck Dyer. Analogically-guided animation. Masters Project Report, Department of Computer Science, University of Wisconsin, May 1993. unpublished.

[21] Munetoshi Unuma, Ken Anjyo, and Ryozo Takeuchi. Fourier principles for emotion-based human figure animation. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 91–96. ACM SIGGRAPH, Addison Wesley, August 1995.

[22] Andrew Witkin and Michael Kass. Spacetime constraints. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 159–168, August 1988.

[23] Andrew Witkin and Zoran Popović. Motion warping. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 105–108, August 1995.