Motion Signal Processing

Armin Bruderlin[1]
Simon Fraser University

Lance Williams[2]
Apple Computer, Inc.

Abstract

Techniques from the image and signal processing domain can be successfully applied to designing, modifying, and adapting animated motion. For this purpose, we introduce multiresolution motion filtering, multitarget motion interpolation with dynamic timewarping, waveshaping and motion displacement mapping. The techniques are well-suited for reuse and adaptation of existing motion data such as joint angles, joint coordinates or higher level motion parameters of articulated figures with many degrees of freedom. Existing motions can be modified and combined interactively and at a higher level of abstraction than conventional systems support. This general approach is thus complementary to keyframing, motion capture, and procedural animation.

Keywords: human animation, motion control, digital signal processing.

1 Introduction

Motion control of articulated figures such as humans has been a challenging task in computer animation. Using traditional keyframing [27], it is relatively straightforward to define and modify the motion of rigid objects through translational and rotational trajectory curves. However, manipulating and coordinating the limbs of an articulated figure via keyframes or the spline curves they define is a complex task that draws on highly developed human skills. More general, global control of the character of an animated motion would be useful in fine-tuning keyframed sequences. Such global control would make predefined sequences more useful, and libraries of animated motion more valuable.

Much of the recent research in motion control of articulated figures has been directed towards reducing the amount of motion specification to simplify the task of the animator. The idea is to build some knowledge about motion and the articulated structure into the system so that it can execute certain aspects of movement autonomously. This has lead to the development of higher level control schemes [5, 6, 15, 22, 33] where the knowledge is frequently specified in terms of rules, and physically-based modeling techniques [8, 12, 18, 30, 31] in which knowledge is embedded in the equations of motion, constraints and possibly an optimization expression. Both approaches often suffer from lack of interactivity: they don't always produce the motion which the animator had in mind, and complex models have a slow interactive cycle. To increase the expressive power of such models, more control parameters can be introduced. Once again, higher-level editing tools for the trajectories of such control parameters would ease animators' burdens and generalize their results.

An alternative method to obtain movements of articulated figures is performance animation where the motion is captured from live subjects. Although a variety of technologies have been developed to fairly reliably measure performance data [19], the computer graphics literature makes scant mention of editing techniques for recorded motion. In the absence of effective editing tools, a recorded movement that is not quite "right" requires the whole data capture process to be repeated.

Because of the complexity of articulated movements and the limitations of current motion control systems as outlined above, we believe that it is desirable to develop tools that make it easy to reuse and adapt existing motion data. For this purpose, we adopt techniques from the image and signal processing domain which provide new and useful ways to edit, modify, blend and align motion parameters of articulated figures. These techniques represent a pragmatic approach to signal processing by providing analytic solutions at interactive speeds, and lend themselves to higher level control by acting on several or all degrees of freedom of an articulated figure at the same time.

In this paper, we treat a motion parameter as a sampled signal. A signal contains the values at each frame[1] for a particular degree of freedom. These values could come from evaluating a spline curve in a keyframing system, or be derived from the tracked markers in a motion capture system. In animating articulated figures we are often concerned with signals defining joint angles or positions of joints, but the signal-processing techniques we have implemented also apply to higher level parameters like the trajectory of an end-effector or the varying speed of a walking sequence.

In Section 2, we present the method of multiresolution filtering and its application to parameters of motion. Section 3 discusses multitarget interpolation, while pinpointing a severe problem of this technique when used for motion blending — the absence of an automatic alignment or registration of movements. A solution to this problem is given based on the principle of dynamic timewarping. Section 4 introduces waveshaping as a rapid nonlinear signal modification method useful for tasks such as mapping joint limits of articulated figures. Section 5 concludes the editing techniques we have developed with motion displacement mapping, an extremely general tool which permits editing of densely-sampled motion data with the ease of keyframing. Each of these sections provides illustrative examples. Finally, conclusions are given in section 6.

[1] School of Computing Science, Simon Fraser University, Burnaby, B.C. V5A 1S6, Canada, (armin@cs.sfu.ca).

[2] Apple Computer, Inc., 1 Infinite Loop, MS 301-3J, Cupertino, CA 95014, USA, (amberinca@aol.com).

[1] Sampled signals have values defined at regular intervals which, in a good animation system, should be completely decoupled from the nominal "frame rate" of the final product. We will speak of "frames" at the sample rate without intending any loss of generality.

2 Multiresolution Filtering

In the motion capture realm, most systems have provision for nonlinear impulse-noise removal filters (Tukey filters) as well as linear smoothing filters for noise reduction in digitized data. There has been less published discussion of the use of signal processing operations to edit or modify captured motion for creative purposes. The "lag, drag, and wiggle" recursive filters in Inkwell [17] represent more relevant previous work in the application of signal processing to keyframed 2D animated motion. These filters were used to stylize motion by invoking linear systems behavior without a more structured physical model, and permitted lively animated effects without unduly taxing the animator. In another related approach, Unuma et al. [28] apply Fourier transformations to data on human walking for animation purposes. Based on frequency analysis of the joint angles, a basic 'walking' factor and a 'qualitative' factor like "brisk" or "fast" are extracted. These factors are then used to generate new movements by interpolation and extrapolation in the frequency domain, such that now a walk can be changed continuously from normal to brisk walking.

"Multiresolution filtering" describes a range of digital filterbank techniques which typically pass a signal through a cascade of lowpass filters to produce a set of short-time bandpass or lowpass signal components. By applying filtering recursively to the output of successive filter bank stages, and downsampling lowpass components as appropriate, these filter banks can be quite efficient; they can produce short-time spectra at roughly the same $n \log(n)$ expense as the Fast Fourier Transform.

The method of multiresolution filtering has been extensively exercised by Burt et al. [4, 20] as an image representation method advantageous for certain kinds of operations, such as seamless merging of image mosaics and intra-image interpolation (noise removal). It has also been applied to temporal dissolves between images [26]. Images may be stored as lowpass (Gaussian) or bandpass (Laplacian) pyramids of spatial filterbands, where each level represents a different octave band of spatial frequencies. Operations like merging two images are then performed band-by-band before reconstructing the image by adding up the resulting bands. In this way, the fine detail of an image corresponding to the higher frequencies can be treated separately from the coarse image features encoded by the low frequencies.

In the currently popular wavelet parlance [7], Burt's Gaussian pyramid is a multiresolution analysis in terms of a cubic B-spline scaling function. The corresponding Laplacian pyramid is simply a bandpass counterpart, where each successively higher level of detail has an interpolated copy of the level beneath subtracted from it. The Laplacian pyramid can be computed directly in this way, or via a modified wavelet transform. Burt's method is more efficient for signals of more than one dimension [29]. As a general observation, for synthesis and modification (as well as many analysis tasks for computer vision), oversampled filter banks like Burt's are more useful than strict subband decompositions (where the number of coefficients does not exceed the number of samples in the original signal). A direct contrast is in the way small translations of an image are projected: sparse decompositions change radically with small offsets of the input image, whereas the Burt pyramids change smoothly. The reduction in coefficients attendant on a subband filterbank may speed numerical solution of some problems; a recent effort in the animation domain is the wavelet formulation of spacetime interpolation for physically- based keyframing by Liu et al. [18]. They did not use the the frequency decomposition to provide direct manipulation of motion, and we believe Burt's method is more appropriate for this purpose.

The first step in applying Burt's multiresolution analysis is to obtain the lowpass pyramid by successively convolving the image with a B-spline filter kernel (e.g. 5×5), while the image is subsampled by a factor of 2 at each iteration (as shown at the left of Figure 1, where G_0 is the original image). This process is repeated until the image size is reduced to one pixel, which is the average intensity, or DC value. The bandpass pyramid is then calculated by repeatedly differencing 2 successive lowpass images, with the subtrahend image being expanded first in each case (right of Figure 1, where L_0 is the highest frequency band). The image can be reconstructed without manipulation by adding up all the bandpass bands plus the DC. The same procedure can be performed on two or more images at the same time, whereby operations like merging are executed band by band before reconstructing the final result.

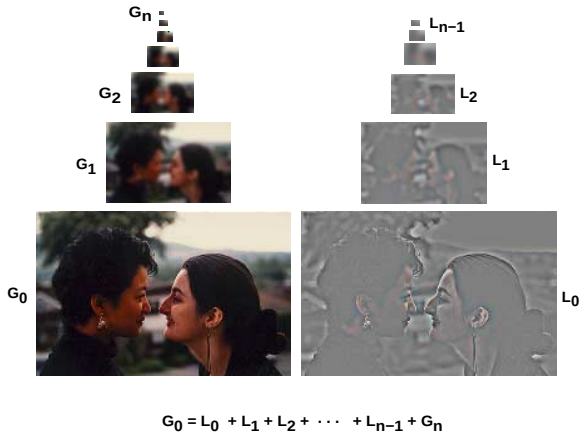

Figure 1: Left: lowpass pyramid; right: bandpass pyramid.

2.1 Motion Multiresolution Filtering

The principles of image multiresolution filtering are now applied to motion parameters of an articulated figure, motivated by the following intuition: low frequencies contain general, gross motion patterns, whereas high frequencies contain detail, subtleties, and (in the case of digitized motion) most of the noise. Each motion parameter is treated as a one-dimensional signal from which the lowpass (G) and bandpass (L) levels are calculated. An example is illustrated in Figure 2 based on the signal of the sagittal knee angle of two walking cycles generated with GAITOR [2].

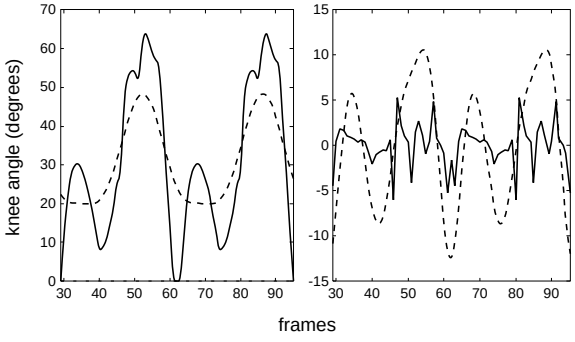

Figure 2: Left: lowpass G_0 (solid) and G_3 (dashed; B-spline kernel of width 5); right: bandpass L_0 (solid) and L_2 (dashed) of the sagittal knee angle for two walking cycles.

2.1.1 Filtering Algorithm

The length m (number of frames) of each signal determines how many frequency bands (fb) are being computed:

$$\text{let } 2^n \leq m \leq 2^{n+1}, \text{ then } fb = n.$$

Instead of constructing a pyramid of lowpass and bandpass sequences where each successive sequence is reduced by a factor of two, alternatively the sequences are kept the same length and the filter kernel (w) is expanded at each level by inserting zeros between the values of the filter kernel (a, b, c below) [3]. For example, with a kernel of width 5,

$$\begin{aligned} w_1 &= [c\ b\ a\ b\ c], \\ w_2 &= [c\ 0\ b\ 0\ a\ 0\ b\ 0\ c], \\ w_3 &= [c\ 0\ 0\ 0\ b\ 0\ 0\ 0\ a\ 0\ 0\ 0\ b\ 0\ 0\ 0\ c], \text{ etc.,} \end{aligned}$$

where $a = 3/8$, $b = 1/4$ and $c = 1/16$. Since we are dealing with signals rather than images, the storage penalty compared to a true pyramid is not as significant ($fb \times i$ versus $4/3 \times i$, where $i =$ number of data points in original signal), while reconstruction is faster since the signal does not have to be expanded at each level. We now state the motion multiresolution algorithm in detail. Steps 1 to 5 are performed simultaneously for each motion parameter signal:

1. calculate lowpass sequence of all fb signals ($0 \leq k < fb$) by successively convolving the signal with the expanded kernels, where G_0 is the original motion signal and G_{fb} is the DC:

   $$G_{k+1} = w_{k+1} \times G_k;$$

   This can be calculated efficiently by keeping the kernel constant and skipping signal data points (i ranges over all data points of a signal)[2]:

   $$G_{k+1}(i) = \sum_{m=-2}^{2} w_1(m)\, G_k(i + 2^k m);$$

2. obtain the bandpass filter bands ($0 \leq k < fb$):

   $$L_k = G_k - G_{k+1};$$

3. adjust gains for each band and multiply L_k's by their current gain values (see example below).

4. blend bands of different motions (optional, see multitarget interpolation below).

5. reconstruct motion signal:

   $$G_0 = G_{fb} + \sum_{k=0}^{fb-1} L_k\,.$$

[2]We implemented several treatments of the boundary of the signal, that is when $i + 2^k m$ lies outside the domain of the signal. The two most promising approaches have proved to be reflecting the signal, and keeping the signal values constant (i.e. equal to the first/last data point) outside its boundaries.

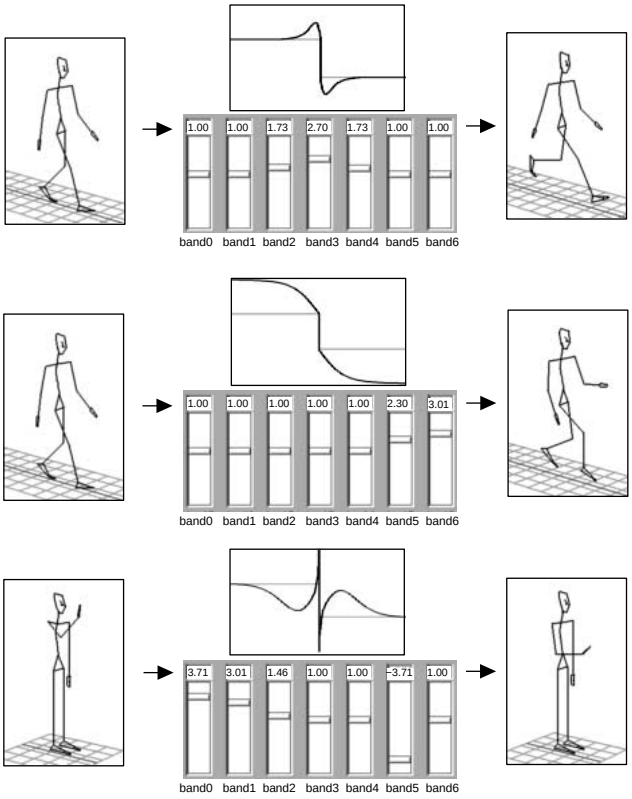

Figure 3: Adjusting gains of bands for joint angles; top: increasing middle frequencies; middle: increasing low frequencies; bottom: using negative gain value.

2.1.2 Examples

An application of motion multiresolution filtering is illustrated in Figure 3. Displayed like an equalizer in an audio amplifier, this is a kind of graphic equalizer for motion, where the amplitude (gain) of each frequency band can be individually adjusted via a slider before summing all the bands together again to obtain the final motion. A step function shows the range and effect of changing frequency gains. We applied this approach successfully to the joint angles (70 degrees of freedom) of a human figure. The same frequency band gains were used for all degrees of freedom. In the example illustrated at the top of Figure 3, increasing the middle frequencies (bands 2, 3, 4) of a walking sequence resulted in a smoothed but exaggerated walk. By contrast, increasing the high frequency band (band 0) added a nervous twitch to the movement (not shown in Figure 3), whereas increasing the low frequencies (bands 5, 6) generated an attenuated, constrained walk with reduced joint movement (Figure 3 middle). Note that the gains do not have to lie in the interval $[0, 1]$. This is shown at the bottom of Figure 3, where band 5 is negative for a motion-captured sequence of a figure knocking at the door, resulting in exaggerated anticipation and follow-through for the knock. We also applied the same filtering to the joint positions (147 degrees of freedom) of a human figure. Increasing the gains for the middle frequency bands of a walking sequence produced a slight scaling effect of the end effectors, and resulted in a squash-and-stretch cartoon walk (Figure 4).

From the examples, it becomes apparent that some constraints such as joint limits or non-intersection with the floor can be violated in the filtering process. Our motion-editing philosophy is to employ constraints or optimization after the general character of the motion has been defined (see displacement mapping in section 5 below; or a

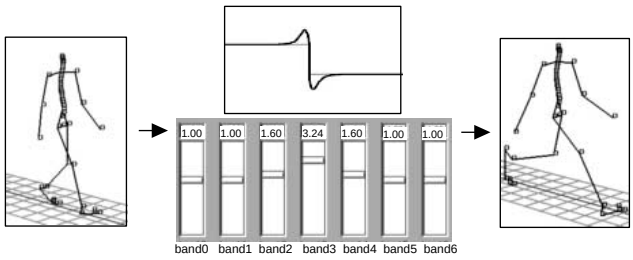

Figure 4: Adjusting gains of bands for joint positions.

more general optimization method [13]). Whereas being trapped in local minima is the bane of global optimization for most problems, animated motion is a good example of an underconstrained problem where the closest solution to the animator's original specification is likely the best. Of course, many animators disdain consistent physics, which is another good reason to decouple motion editing from constraint satisfaction.

Finally, we suggest that a multiresolution approach could also be quite useful in defining motion sequences, rather than simply modifying them. Much like an artist creating a picture blocks out the background first with a big brush, then adds more and more detail with finer and finer brushes, a generic motion pattern could be defined first by low frequencies, and then "finetuned" by adding in higher frequency refinements[3].

3 Multitarget Interpolation

Multitarget interpolation refers to a process widely used in computer animation to blend between different models. The technique was originally applied in facial animation [1, 21]. We might have a detailed model of a happy face, which corresponds parametrically to similar models of a sad face, quizzical face, angry face, etc. The control parameters to the model might be high level (like "raise left eyebrow by 0.7"), very high level (like "be happy"), or they might simply be the coordinates of the points on a surface mesh defining the shape of part of the face. By blending the corresponding parameters of the different models to varying degrees, we can control the expression of the face.

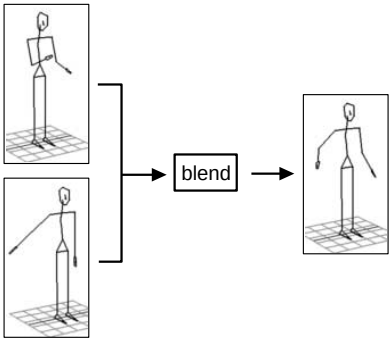

Figure 5: Example of multitarget motion interpolation.

3.1 Multitarget Motion Interpolation

We can apply the same technique to motion. Now we might have a happy walk, a sad walk, angry walk, etc., that can be blended freely to provide a new result. Figure 5 shows an example of blending two different motions of a human figure, a drumming sequence and a "swaying arm sideways" sequence. In this case, the blend is linear, i.e. add 0.4 of the drum and 0.6 of the arm-sway. In general, the blend can be animated by "following" any trajectory in time. Guo et al. [11] give a good discussion of this approach which they term parametric frame space interpolation. Our approach generalizes on theirs in that the motion parameters such as joint angles to be blended are completely decoupled from one another, and have no implicit range limits. Each component of an arbitrary ensemble of input parameters can have an independent blending coefficient assigned to it.

As indicated in step (4) of the multiresolution algorithm above, we can mix multitarget interpolation and multiresolution filtering to blend the frequency bands of two or more movements separately. This is illustrated in Figure 6 for the same two motions (a drum and an arm-sway) as in Figure 5. Adjusting the gains of each band for each motion and then blending the bands provides finer control while generating visually much more pleasing and convincing motion.

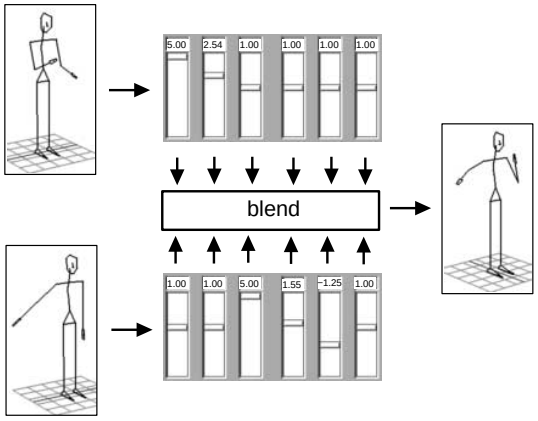

Figure 6: Multitarget interpolation between frequency bands.

However, there is a potential problem when applying multitarget interpolation to motion which relates to the notion of parametric correspondence as stated above: for all our face models to "correspond parametrically" implies that the parameters of each of the models has a similar effect, so that if a parameter raises the left eyebrow of face number one, a corresponding parameter raises the left eyebrow in face number two. If our parameters are simply surface coordinates, it means that the points on each surface correspond, so if the point at U, V coordinates $U1$, $V1$ is at the tip of the left eyebrow, the point at the same coordinates in any other face will also be at the tip of the left eyebrow.

In motion, parametric correspondence means much the same thing, except that now a correspondence with respect to time is required. If we are blending walk cycles, the steps must coincide so that the feet strike the ground at the same time for corresponding parameter values. If the sad walk is at a slower pace than the happy walk, and we simply blend them together without first establishing a correspondence between the steps, the blend will be a curious dance of uncoordinated motions, and the feet will no longer strike the ground at regular intervals; indeed, they are no longer guaranteed to strike the ground at all (see Figure 7). Thus, multitarget motion interpolation must include both a distortion (remapping a function in time) and a blend (interpolating among different mapped values). In the visual domain a transformation like this is termed a "morph."

Another example is illustrated in Figure 8; here the motion sequences of two human figures waving at different rates and intensities (a "neutral" and a "pronounced" wave) were first blended without timewarping. This resulted in a new wave with undesirable

[3] Personal communication, Ken Perlin, New York University, 1994.

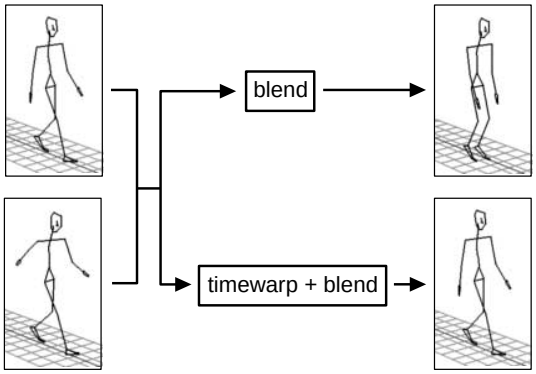

Figure 7: Blending two walks without (top) and with (bottom) correspondence in time.

secondary waving movements superimposed. After timewarping the neutral to the pronounced wave, the blend produced the neutral wave at the pronounced rate. In the following section we describe an automatic method for establishing correspondence between signals to make multitarget motion interpolation meaningful and useful.

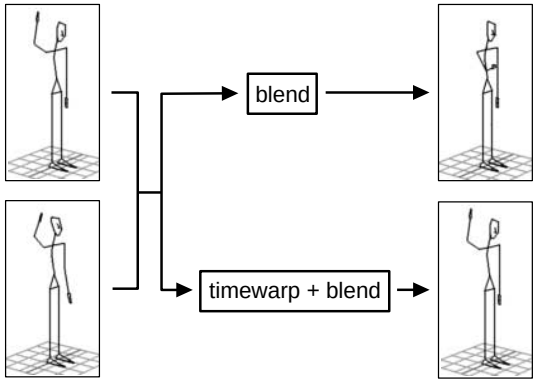

Figure 8: Blending two waves without (top) and with (bottom) correspondence in time.

3.2 Dynamic Timewarping

The field of speech recognition has long relied on a nonlinear signal matching procedure called "dynamic timewarping" to compare templates (for phonemes, syllables or words) with input utterances [9]. Apart from being subject to the usual random error, each acoustic input signal also shows variations in speed from one portion to another with respect to the template signal. The timewarp procedure identifies a combination of expansion and compression which can best "warp" the two signals together.

In our case, timewarping is applied in the discrete time domain to register the corresponding motion parameter signals such as joint angles. In Figures 7 and 8, the timewarping was done simultaneously for all 70 rotational degrees of freedom of the human figure for the duration of the movement sequences. If we have a military march and a drunken stagger, two new gaits can immediately be defined from the timewarp alone: the military march at the drunken pace, and the drunken stagger at the military pace. Figure 9 shows an example for one degree of freedom (knee angle) for the two walks warped in Figure 7. However, we are not limited to these two extreme warps, but may freely interpolate between the mappings of the two walks, and between the amplitudes of the signals through these mappings independently.

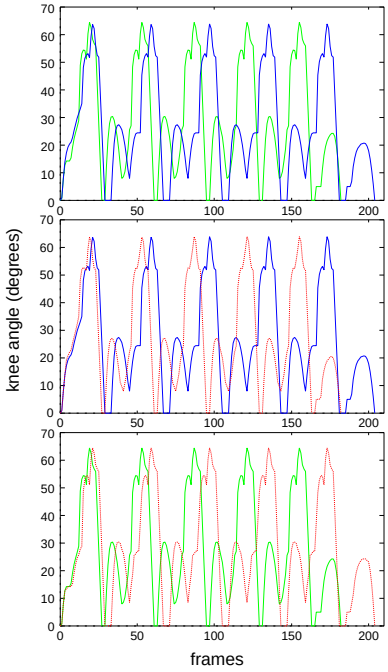

Figure 9: Top: sagittal knee angles curves of two walks; middle: red = blue curve warped to match green; bottom: red = green curve warped to match blue.

3.2.1 Timewarp Algorithm

The problem can be decomposed and solved in two steps: finding the optimal sample correspondences between the two signals, and applying the warp. The vertex correspondence problem is defined as finding the globally optimal correspondence between the vertices (samples) of the two signals: to each vertex of one signal, assign (at least) a vertex in the other signal such that a global cost function measuring the "difference" of the two signals is minimized. In this sense, the problem is related to contour triangulation [10] and shape blending [24], and is solved by dynamic programming optimization techniques. The solution space can be represented as a two-dimensional grid, where each node corresponds to one possible vertex assignment (see Figure 10). The optimal vertex correspondence solution is illustrated in the grid by a path from $(0, 0)$ to $(9, 9)$. In general, there are $O(n^n/n!)$ such possible paths[4].

When applying timewarping to recognition, the best fit to a canon of signals is computed; no subsequent use is made of a warped signal. The algorithms which perform the warp typically do so by forming a discrete, point-sampled correspondence [9]. For synthetic purposes, more continuous transformations and cost functions are appropriate [32, 25]. We adopted Sederberg's shape blending algorithm [24] which guarantees a globally optimal solution by visiting every node in the grid once ($O(n^2)$ with constant amount of work per node). Upon reaching node (n, n), the optimal solution is recovered by backtracking through the graph. Sederberg's "physically-based" approach measures the difference in "shape" of the two signals by calculating how much work it takes to deform one signal into the other. The cost function consists of the sum of local stretching and bending work terms, the former involving two, the latter three adjacent vertices of each signal. Intuitively, the larger the difference in distance between two adjacent vertices of

[4] This holds for the vertex correspondence problem, where we favor a diagonal move in the graph over a south-followed-by-east-move or an east-followed-by-a-south-move. For contour triangulation [10], where diagonal moves are denied, the complexity is $O((2n)!/(n!n!))$.

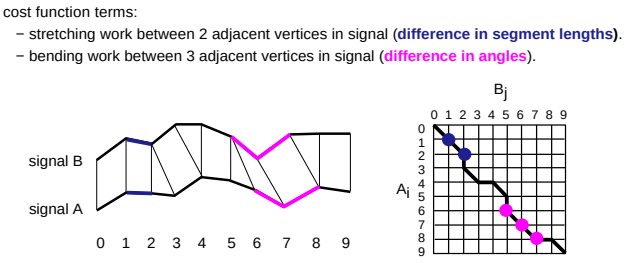

Figure 10: Vertex correspondence problem and cost functions.

one signal and the two vertices of the other (given by two adjacent nodes in the graph), the bigger the cost. Similarly, the larger the difference in angles between three adjacent vertices of one signal and the three vertices of the other (given by three adjacent nodes in the graph), the bigger the cost (for details, see [24]; an illustration is given in Figure 10). One additional check was introduced to make sure that we are really comparing corresponding angles in the two signals: if the middle of the three vertices used to calculate the angle is a local minimum in one signal and a local maximum in the other signal, then one of the angles (α) is inverted before calculating the cost term ($\alpha = 360\,\text{deg} - \alpha$).

The second part of the problem is to apply the warp given the optimal vertex correspondences. As in speech recognition [9], three cases are distinguished: substitution, deletion and insertion. This is indicated in the optimal path by a diagonal, horizontal and vertical line, respectively, between two nodes. For the following explanations, we assume that signal B is warped into A as shown in Figure 11, and the warped signal is denoted by B_w. Then if B_j and A_i are related by a substitution it follows that $B_{w_i} = B_j$. In case of a deletion, where multiple samples of B, $(B_j, B_{j+1}, \ldots, B_{j+k})$, correspond to one A_i, $B_{w_i} = \text{mean}(B_j, B_{j+1}, \ldots, B_{j+k})$. Finally, an insertion implies that one sample of B, B_j, maps to multiple samples of A, $(A_i, A_{i+1}, \ldots, A_{i+k})$. In this case, the values for $B_{w_i}, B_{w_{i+1}}, \ldots, B_{w_{i+k}}$ are determined by calculating a cubic B-spline distribution around the original value B_j.

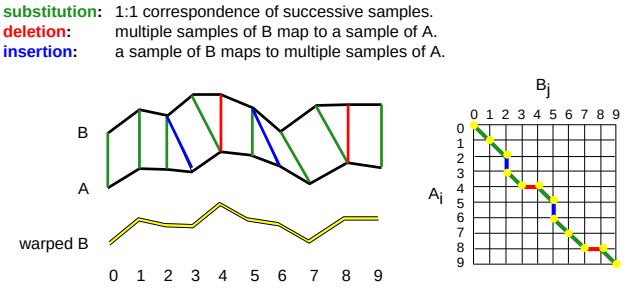

Figure 11: Application of timewarp (warp B into A).

4 Waveshaping

The transformations discussed so far are operations on the time history of a signal. Operations which are evaluated at each point in the signal without reference to its past or future trajectory are occasionally termed *point processes*. Such operations include scaling or offsetting the signal, but are more generally described as a functional composition. Familiar uses of functional composition in graphics include gamma correction and color-lookup, as well as tabular warping functions for images.

"Digital waveshaping" is the term applied to functional composition in computer sound synthesis. In this domain, a normalized input signal x (e.g. scaled to the range from -1 to $+1$) is directed through a discrete *shaping* function f (or waveshaping table) to synthesize steady-state or time-varying harmonic sound spectra. Although waveshaping is in general a nonlinear operation, its effects when applied to an input sine wave can be easily characterized [16]. In practical terms, if f is defined as the identity function $f(x) = x$, the signal will pass through unchanged. If f is slightly changed, say, to having a subtle bump near 0, then the signal x will be altered in that it will have slightly positive values where, and around where, it was zero before, thus x has now some bumps as well. If f is defined as a partial cycle of a cosine function going from minimum to maximum over the $[-1, +1]$ range, the values of x will be exaggerated in the middle and attenuated at the extremes. If f is a step function, x will be quantized to two values.

4.1 Motion Waveshaping

An example of how this idea can be adopted for animation is illustrated in Figure 12. Here the default identity shaping function has been modified to limit the joint angles for a motion sequence of an articulated figure waving. In the figure, "hard" limits are imposed: values of x greater than a limit value simply map to that value. An alternative is a "soft" limit: as values exceed the limit, they are mapped to values that gradually approach it. The implementation of our shaping function is based on interpolating cubic splines [14]; a user can add, delete and drag control points to define the function and then apply it to all or some degrees of freedom of an articulated figure.

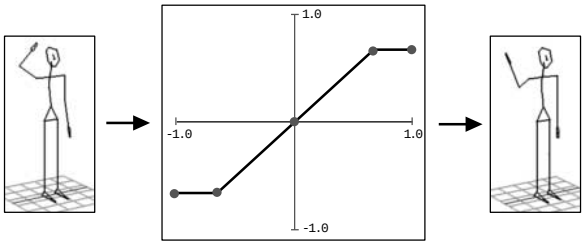

Figure 12: Capping of joint angles via a shape function.

Another application of waveshaping is to map the shape of input motions to a "characteristic" function. The shaping function in Figure 13 applied to the motion-captured data of a human figure sitting and drinking introduced extra undulations to the original monotonic reaching motion. In this way, it is possible to build up a library of shaping functions which will permit rapid experimentation with different styles of movement.

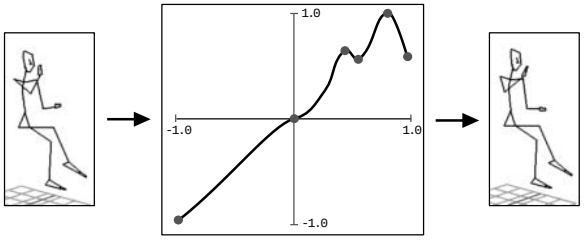

Figure 13: Adding undulations to motion via waveshaping.

5 Motion Displacement Mapping

Displacement mapping provides a means to change the shape of a signal locally through a displacement map while maintaining continuity and preserving the global shape of the signal. To alter a movement, the animator just changes the pose of an articulated figure at a few keyframes. A spline curve is then fitted through these displacements for each degree of freedom involved, and added to the original movement to obtain new, smoothly modified motion. The basic approach is illustrated in Figure 14. Step 1 is to define the desired displacements (indicated by the three vertical arrows) with respect to the motion signal; in step 2, the system then fits an interpolating cubic spline [14] through the values of the displacements (note that the first and last data points are always displacement points). The user can then adjust the spline parameters in step 3 before the system calculates the displaced motion satisfying the displacement points (step 4).

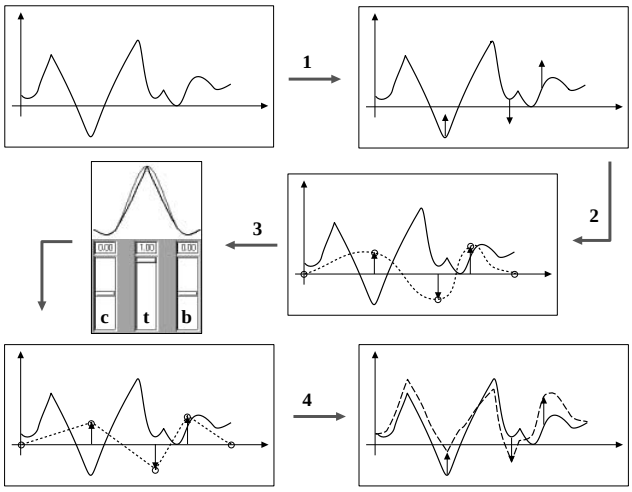

Figure 14: Steps in displacement mapping.

The displacement process can be applied iteratively until a desired result is achieved. Since the operation is cheap, a fast feedback loop is guaranteed. In the top part of Figure 15, we took the output of a multiresolution filtering operation on joint angles of a human walking figure, where some of the joint limits were violated and the feet did not make consistent contact with the ground, and read it into LifeForms [5], a system to animate articulated figures. There we adjusted some of the joints and translated the figure at a few keyframes for which displacement curves were quickly generated and applied to the motion of the figure as described above. To refine the resulting motion, a second loop was executed; a frame of the final result is shown on the top right of Figure 15. The same technique was used in modifying the rotoscoped motion of a human figure sitting and drinking (Figure 15, middle). Here, three out of the 600 motion-captured frames were modified to include some additional gestures of the arms and legs. In Figure 15, bottom, the joint angles for the arm and neck of a motion-captured knocking-at-a-door-sequence were changed for one frame via motion displacement mapping to obtain a knock at a higher impact point.

6 Conclusions

In this paper we have assembled a simple library of signal processing techniques applicable to animated motion. A prototype system has been implemented in the programming language C using the

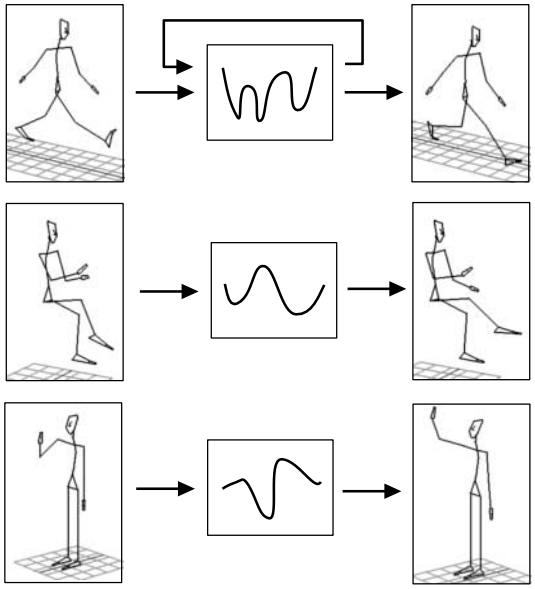

Figure 15: Examples of applying displacement curves.

Khoros application development environment [23]. The immediate goals of our motion-editing experiments have been fulfilled: the motion signal processing techniques provide a rapid interactive loop, and facilitate reuse and adaptation of motion data. By automating some aspects of motion editing such as time-registration of signals or increasing the middle frequencies for several degrees of freedom at the same time, these techniques lend themselves to higher level motion control and can serve as building blocks for high-level motion processing.

Of all the techniques introduced here, perhaps motion displacement mapping will prove to be the most useful; it provides a means by which a basic movement such as grasping an object from one place on a table can be easily modified to grasping an object anywhere else on the table. This allows simple and straightforward modification of motion-capture data through a standard keyframing interface. Timewarping as a non-linear method to speed up or slow down motion is useful in blending different movements. It could also play an important role in synchronizing various movements in an animation as well as in synchronizing animation with sound. Multiresolution filtering has been demonstrated as an easy tool to change the quality of a motion. Waveshaping represents a simple but efficient way to introduce subtle effects to all or some degrees of freedom. As the use of motion capture is becoming increasingly popular and libraries of motions are increasingly available, providing alternate methods for modifying and tweaking movement for reuse can be of great value to animators.

We believe that a wide range of animation tasks can be addressed with these techniques at a high level which is complimentary to and extends conventional spline tweaking tools:

- blending of motions is straightforward by using multitarget interpolation with automatic time registration of movements. This is a convenient way to build up more complex motions from elementary ones. For more fine-control, the frequency bands can first be computed for each motion before blending band-by-band while adjusting the frequency gains.

- concatenating motions is another practical application of multitarget interpolation, giving the user control over blending interval (transition zone) and blending coefficient. Multiresolution can be applied to concatenate band-by-band.

- capping of joint angles is a task easily accomplished by waveshaping. This tool is also well suited to apply user-defined undulations to all or some degrees of freedom to make a "bland" motion more expressive.
- some animation tasks which can be achieved with multiresolution analysis include "toning down" a motion by increasing the low frequency gains, "exaggerating" a movement by increasing the middle frequencies, producing a "nervous twitch" by increasing the higher frequencies, and generating "anticipation and follow-through" by assigning negative gain values. Because of immediate feedback, the user can quickly experiment with different combinations of gain values for specific movement qualities.
- editing of motion-captured data is very desirable yet very tedious in current systems. As mentioned above, displacement mapping provides an interface through which the animator can conveniently change such data at a few selected "keyframes" while preserving the distinctive "signature" of the captured motion.

References

[1] BERGERON, P., AND LACHAPELLE, P. Controlling facial expressions and body movements in the computer-generated animated short: Tony de Peltrie. In *Computer Graphics (SIGGRAPH '85), Course Notes: Techniques for Animating Characters* (July 1985).

[2] BRUDERLIN, A., AND CALVERT, T. Interactive animation of personalized human locomotion. In *Graphics Interface '93, Proceedings* (May 1993), pp. 17–23.

[3] BURT, P. Multiresolution method for image merging. In *Computer Graphics (SIGGRAPH '86), Course Notes: Advanced Image Processing* (August 1986).

[4] BURT, P., AND ADELSON, E. A multiresolution spline with application to image merging. *ACM Transactions on Graphics 2*, 4 (October 1983), 217–236.

[5] CALVERT, T., BRUDERLIN, A., DILL, J., SCHIPHORST, T., AND WELMAN, C. Desktop animation of multiple human figures. *IEEE Computer Graphics & Applications 13*, 3 (1993), 18–26.

[6] CASSELL, J. ET AL. Animated conversation: Rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. In *Computer Graphics (SIGGRAPH '94 Proceedings)* (July 1994), pp. 413–420.

[7] CHUI, C. K. *An Introduction to Wavelets, Series: Wavelet Analysis and its Applications.* Academic Press, Inc., 1992.

[8] COHEN, M. Interactive spacetime control for animation. In *Computer Graphics (SIGGRAPH '92 Proceedings)* (July 1992), vol. 26, pp. 293–302.

[9] DEMORI, R., AND PROBST, D. *Handbook of Pattern Recognition and Image Processing*. Academic Press, 1986, ch. Computer Recognition of Speech.

[10] FUCHS, H., KEDEM, Z., AND USELTON, S. Optimal surface reconstruction from planar contours. *Communications of the ACM 10*, 10 (1977), 693–702.

[11] GUO, S., ROBERGE, J., AND GRACE, T. Controlling movement using parametric frame space interpolation. In *Computer Animation '93, Proceedings* (1993), pp. 216–227.

[12] ISAACS, P., AND COHEN, M. Controlling dynamic simulation with kinematic constraints, behavior functions and inverse dynamics. In *Computer Graphics (SIGGRAPH '87 Proceedings)* (1987), vol. 21, pp. 215–224.

[13] KASS, M. Condor: Constraint-based dataflow. In *Computer Graphics (SIGGRAPH '92 Proceedings)* (1992), vol. 26, pp. 321–330.

[14] KOCHANEK, D., AND BARTELS, R. Interpolating splines with local tension, continuity and bias control. In *Computer Graphics (SIGGRAPH '84 Proceedings)* (1984), vol. 18, pp. 33–41.

[15] KOGA, Y., KONDO, K., KUFFNER, J., AND LATOMBE, J.-C. Planning motions with intentions. In *Computer Graphics (SIGGRAPH '94 Proceedings)* (July 1994), pp. 395–408.

[16] LEBRUN, M. Digital waveshaping synthesis. *Journal of the Audio Engineering Society 27*, 4 (1979), 250–266.

[17] LITWINOWICZ, P. Inkwell: A 2 1/2–D animation system. In *Computer Graphics (SIGGRAPH '91 Proceedings)* (1991), vol. 25, pp. 113–122.

[18] LIU, Z., GORTLER, S., AND COHEN, M. Hierarchical spacetime control. In *Computer Graphics (SIGGRAPH '94 Proceedings)* (July 1994), pp. 35–42.

[19] MEYER, K., APPLEWHITE, H., AND BIOCCA, F. A survey of posistion trackers. *Presence: Teleoperators and Virtual Environments 1*, 2 (Spring 1992), 173–200.

[20] ODGEN, J., ADELSON, E., BERGEB, J., AND BURT, P. Pyramid-based computer graphics. *RCA Engineer 30*, 5 (1985), 4–15.

[21] PARKE, F. ET AL. State of the art in facial animation. In *Computer Graphics (SIGGRAPH '90), Course Notes* (August 1990).

[22] PHILLIPS, C., AND BADLER, N. Interactive behaviors for bipedal articulated figures. In *Computer Graphics (SIGGRAPH '91 Proceedings)* (1991), vol. 25, pp. 359–362.

[23] RASURE, J., AND KUBICA, S. The Khoros application development environment. Tech. rep., Khoral Research, Inc., 4212 Courtney NE, Albuquerque, NM, 87108, USA, 1993.

[24] SEDERBERG, T., AND GREENWOOD, E. A physically-based approach to 2–D shape blending. In *Computer Graphics (SIGGRAPH '92 Proceedings)* (1992), vol. 26, pp. 26–34.

[25] SERRA, B., AND BERTHOLD, M. Subpixel contour matching using continuous dynamic programming. *IEEE Computer Vision and Pattern Recognition* (1994), 202–207.

[26] STEIN, C., AND HITCHNER, H. The multiresolution dissolve. *SMPTE Journal* (December 1988), 977–984.

[27] STURMAN, D. A discussion on the development of motion control systems. In *Graphics Interface '86, Tutorial on Computer Animation* (1986).

[28] UNUMA, M., AND TAKEUCHI, R. Generation of human motion with emotion. In *Computer Animation '93, Proceedings* (1993), pp. 77–88.

[29] VELHO, L. *Piecewise Descriptions of Implicit Surfaces and Solids*. PhD thesis, University of Toronto, Computer Science, 1994.

[30] WILHELMS, J. Virya–a motion control editor for kinematic and dynamic aniamtion. In *Graphics Interface '86, Proceedings* (1986), pp. 141–146.

[31] WITKIN, A., AND KASS, M. Spacetime constraints. In *Computer Graphics (SIGGRAPH '88 Proceedings)* (1988), vol. 22, pp. 159–168.

[32] WITKIN, A., TERZOPOULUS, D., AND KASS, M. Signal matching through scale space. *International Journal of Computer Vision* (1987), 133–144.

[33] ZELTZER, D. Towards an integrated view of 3–D computer character animation. In *Graphics Interface '85, Proceedings* (1985), pp. 105–115.