

Animating by Multi-level Sampling

Katherine Pullen, Christoph Bregler
Computer Science Department
353 Serra Mall
Stanford University
Stanford, CA 94306-9010
{pullen, bregler}@graphics.Stanford.edu
(650) 725-6359, FAX: (650) 725-1449

Keywords: Analysis/Synthesis, Motion-Capture based Animation, Signal-Processing, Noise in Animation, Kangaroos

Abstract

In this paper we describe a method for synthesizing joint angle and translation data based on the information in motion capture data. The synthetic data is realistic not only in that it resembles the original training data, but in that it has random variations that are statistically similar to what one would find in repeated measurements of the motion. To achieve this result, the training data is broken into frequency bands using a wavelet decomposition, and the information in these bands is used to create the synthetic data one frequency band at a time. The method takes into account the fact that there are correlations among numerous features of the data. For example, a point characterized by a particular time and frequency band will depend upon points close to it in time in other frequency bands. Such correlations are modeled with a kernel-based representation of the joint probability distributions of the features. The data is synthesized by sampling from these densities and improving the results using a new iterative maximization technique. We have applied this technique to the synthesis of joint angle and translation data of a wallaby hopping on a treadmill. The synthetic data was used to animate characters that have limbs proportional to the wallaby.

1. Introduction

There are three main methods by which computer animations are created. Most commonly, the motions are produced by key-

frame interpolation. Good results in this case depend upon the skill and hard work of the animator. In order to reduce the burden on the animator, there has been a large amount of research to develop techniques based on physical simulation. These methods have been most useful for animating inanimate objects such as cloth deformations or falling objects, where the physics of the situation determines the motion. However, applying physical simulation to character animation is much more difficult, as a true physics based model of a human is complex, and an animator is often interested in the subtleties of the motion that give his or her character personality that would be extremely difficult to model. As a result, there has been an increasing interest in motion capture based animation, in which the motion parameters of a live actor are captured by sensors. This technique is gaining popularity as the sensor technology improves and becomes more readily available. If motion capture is done with enough accuracy, it will record all necessary subtleties that are important to convey the desired life-like appeal and personality of the character.

The disadvantage of motion-capture based animation is that the motions that are recorded may not be exactly what the animator wants. The motion may need to be fit to a character of different proportions than the actor or be altered in other ways to match the situation being animated. As a result, there has been much research recently to develop methods for manipulating motion capture data. For example, motion signal processing, motion editing, and motion retargeting techniques allow an animator to adapt the movement to different environments and skeleton dimensions while keeping the essence of the original motion intact.

One aspect of motion-capture data that has not been explored in as much depth is the inherent randomness of life-like motions. For example, if we execute the same gesture twice, it will not come out the same each time. Or when we walk, not every step is identical; there are slight variations in stride length, how we swing our arms or move our head from step to step, and these subtle motions are part of what give the motion life and personality. When motion capture data is applied to a repetitive movement, is it usually cyclified. In other words, the same cycle of one step is used over and over again to make a character walk across the floor, which does not accurately represent the natural variations. There has been some work on adding noise to motion to increase the life-like quality of such movement, but we were interested in exploring this area from a different perspective, which is the topic of this paper.

Our goal was to start from motion capture data of a repetitive motion and synthesize a set of motion curves for joint angles and translations that are statistically similar to this training data. The generated data should preserve features of the original data that are important to make it appear real, and yet not be an exact copy. The variations in the synthetic motion should reflect the variation that one would expect in the real data.

Fluctuations in motion capture data may occur over different time scales. There are low frequency variations that we perceive as changes in stride length, and higher frequency fluctuations that we see as “jitter”. In order to separate the effects of these different types of noise, we decomposed the data into frequency bands. The value of the data points in each band are not independent; in fact they are highly correlated. In addition, there are strong correlations among other features of the data. For example, a point in the knee angle data at a given time and in a given frequency band will depend upon the points in the hip angle data characterized by similar times and frequency bands. We take these correlations into account by creating conditional probability distributions among the various features of the data.

The synthesis procedure is initiated by sampling randomly from conditional probability distributions among lower frequencies and other joints, and completed by using a new iterative technique to maximize the probability of occurrence of each value. To test the method, we use as training data the motion of a wallaby (a small species of kangaroo) hopping on a treadmill. The final product is an animation of a small crowd of hopping creatures with limb lengths designed to

be proportional to that of the wallaby. The creatures all move as individuals, in that each hops as if it were animated with the original data, and yet no two are alike because of slight variations in their strides.

In section 2 we survey related approaches, in section 3 we describe the technique we used for the synthesis, and in section 4 we discuss our results and conclusions.

2. Related Work

Our work draws on ideas or is related to four different areas which we summarize here:

2.1 Animating with Noise

Perlin and Goldberg [Per96] demonstrated an animation system called *Improv* that enables an expert to create interactive animations driven by procedural techniques. One important feature of this system is the use of noise functions that add random variability to the resulting animations. This is based on earlier work in visual texture synthesis with so called *Perlin-noise* [Per85]. An animator must tune the appropriate noise functions, and the result is life-like and very impressive animations. Another approach based on hand-crafted noise functions that are motivated by biomedical studies is proposed by Bodenheimer et al. [Bod99] and applied to cyclic running motion. The motion is created by dynamical simulations described in Hodgins et al [Hod95], and these simulations are perturbed by different noise functions. We also are interested in noise and fluctuation in motion, but our work differs from the above in that we wanted to find a way to extract the variation from the data itself to avoid or minimize the step of having to tune the noise to the animation in question.

2.2 Varying Existing Motions

Many techniques have been proposed that start with existing motions, often obtained from motion-capture data, and vary the motions to adapt to different constraints or change stylistic features. For example Unuma et al. [Unu95] developed a functional model in the Fourier domain in which they perform rescaling, interpolation, and extrapolation of human locomotion. Bruderlin and Williams’s [Bru95] Motion Signal Processing applies several image processing techniques to motion data. Related to our approach is their use of multi-resolution

filtering techniques and the modification of different frequency bands to change the style of the motion. Examples of other techniques that vary existing motions are Motion Warping Techniques by Witkin and Popovic [Wit95], and Motion Retargetting techniques by Gleicher [Gle98].

2.3 Texture Analysis / Synthesis

Our original motivation for this new motion synthesis technique comes from methods used for visual texture analysis and synthesis. One of the first presented approaches related to our technique is work by Heeger [Heg95] using multi-level Laplacian pyramids. In an analysis phase statistics across all resolution levels are estimated using histograms. In the synthesis phase, a new image is generated by starting with white noise, and iteratively histogram matching the new image to the statistics of the original image. More elaborate treatment of this problem has been reported by Simoncelli et al. [Sim98], DeBonet [Deb97], and Efros et al [Efr99]. Specifically DeBonet introduces the use of a so-called Parent Structure, that forces the resynthesis to take into account the fact that that higher frequency statistics are conditionally dependent on lower frequency statistics.

2.4 Density Estimation

Most texture synthesis techniques use histograms to represent the conditional probability distributions among features in the images. In the realm of motion statistics, other density estimation techniques such as those based on Mixture of Gaussians and Hidden Markov Models have been used. For instance, Brand [Bra99] demonstrated how to use such representations and a new entropic prior for the estimation of speech and mouth motions, and applied it to a mapping problem from audio input to facial motion output. In contrast to these methods, we estimate conditional dependencies across multi-resolution layers using so-called kernel-based density representations [Bis95]. As we describe below, we found that kernel-based densities represent an ideal trade-off between smoothing and generalizing the data while still keeping the subtle variations in the motion.

3. Technical Approach

In the following sections, we describe the method we used to synthesize motion curves for our animations.

3.1 Data

Our input data was a video of a wallaby hopping on a treadmill. The reason for choosing this motion for our initial studies is that when a wallaby hops, he does so with his legs together. As a result, it is a very good approximation to model the motion in two dimensions, which simplifies the problem. The video sequence showed about 8 hops in a row; such a database is necessary to provide samples of how the motion naturally fluctuates from hop to hop. We use a vision based kinematic tracking technique that is an extension of [Bre98] to determine the hip, knee, and ankle joint angle motions and the horizontal and vertical translations of the wallaby from the video. Figure 1 shows several frames from the video, in which we overlaid the estimated kinematic chain model of the leg.

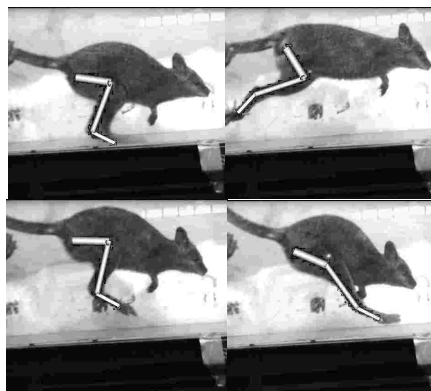


Figure 1: Example frames from original data with markers showing the captured joint angles.

3.2 Separation into Frequency Bands

In order to separate the effects of the low frequency fluctuations from the high frequency jitter, we chose to decompose the data into

frequency bands. We found that any number of decompositions would work in the synthesis method described below. For example, we tried different types of wavelets and the Laplacian pyramid, with similar results. We applied the decomposition to each of the degrees of freedom of the motion, in this case the hip, knee, and ankle joint angles, and the x and y translations. Note our convention of labeling the lower frequencies with higher numbers. Figure 2 shows the band-pass output of the wallaby hip angle data for levels 1-5.

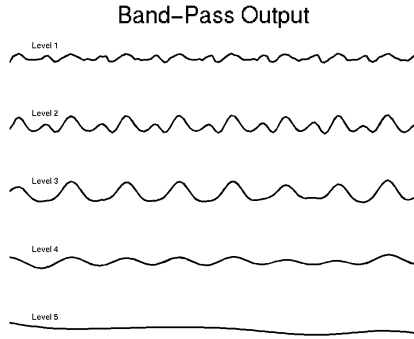


Figure 2: Frequency Decomposition of hip angle data

3.3 Representing Cross-Level and Cross-Joint Correlations with Kernel Based Densities

We cannot treat each frequency band and each joint angle motion separately. The value of a point characterized by a given time, wavelet level, and joint will depend upon the values of points at nearby times in the same and other wavelet levels and joints. Yet, because there are always random variations in real data, these dependencies are not deterministic. We can represent the dependencies with multivariate densities. For example, the value of the knee angle at time point t will depend on the value of the hip angle at the same time (see figure 4). We can represent this dependency by a multivariate probability density $p(\text{knee}, \text{hip})$. Most texture-analysis and synthesis techniques use histograms to represent such densities. We choose instead to use Kernel-based densities [Bis95]. As we will describe later, such densities are very well suited for iterative techniques that try to find a local maximum in the density.

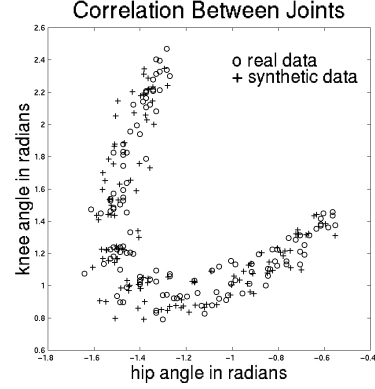


Figure 3: it is important that the synthesis technique maintains the correlations between joint angles that are present in the original data. This plot shows the knee versus hip at each time point in the original data and the synthetic data.

Kernel-based density representations are non-parametric densities, in contrast to a parametric density such as a Gaussian density. The advantage of parametric densities are their compactness, in that one function represents all of the data. For example, in figure 3 we have plotted the knee angle versus the hip. A parametric density would fit a function such as a two-dimensional gaussian to all of the data. Clearly in this case that would not be appropriate, as there is a definitely non-gaussian pattern to the shape of the distribution of points. Yet it is not clear what shape of function *would* work, as it is an unusual shape. In addition, when we consider three or more features together, as we wish to do in this synthesis method, the data become sparse, further complicating efforts to fit a single function to the distribution.

To get around such problems, we can use kernel-based densities, which do not suffer from these shortcomings if the kernel itself is tuned to the right resolution level of the problem. Continuing our example of the dependence of the knee on the hip, let x_k represent a vector of values of hip angle data, and y_k represent a vector of values of knee angle data. Then a Kernel-based density for this case would be defined by:

$$P(k,h) = \sum K(x - x_k)K(y - y_k)$$

where x and y are particular hip and knee angle values, respectively, that we are interested in knowing the probability of occurring together. The kernel K can be any one of a number of functions; in this work we have chosen a

Gaussian. The sigma of the gaussian is a parameter that is chosen to find the right balance between generalizing the data and keeping the subtleties. If the spread of the Gaussian is very large, the resulting density will be very smooth and we might wash out important details. If the spread of the Gaussian is very small, the density peaks very highly at the training points, but vanishes everywhere else. See figure 4 for examples of representing the knee versus hip angle data as a kernel-based density representation. In figure 4a the kernel is too small to generalize the data, and in 4d it is too large to see the shape of the distribution; the subtleties are smoothed out. In 4b and 4c are more reasonable representations of the data, depending on the purpose.

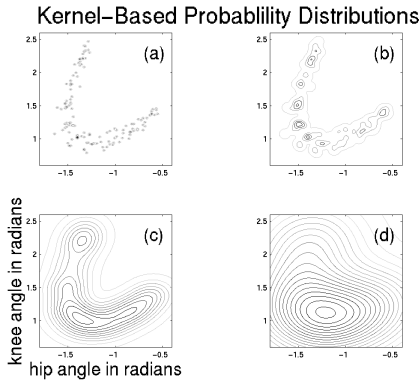


Figure 4: Representation of the knee versus hip angle data as a gaussian kernel-based probability distribution, displayed as contour plots. Darker colors indicate regions of higher probability. The raw data is the same as is plotted in figure 4. (a) sigma = .01 radians (b) sigma = .05 radians. (c) sigma = .1 radians (d) sigma = .5 radians.

3.4 Synthesis of Data

To describe the method, in this section we will consider the example of synthesizing the wallaby knee angle. We form a wavelet decomposition of the knee angle training data to create a set of band-pass data. We choose the lowest band-pass level that appears relatively random, the 5th in this case (see figure 2), and create it with random numbers that are scaled to match the data. The next level, the 4th in this case, is created with a sine wave with noise added. The noise comes

from the 4th level of a band-pass decomposition of white noise. The rest of the levels of the knee angle data are computed one wavelet level at a time using the multivariate kernel-based densities that represent the statistics of the training data. We use a cycle of initialization and optimization that are described below.

Initialization. Suppose we are synthesizing a point at time t in wavelet level L of the knee angle data. To make an initial guess as to what the value of this point should be, we use the following features: the point at time t in level $L+1$ for the knee angle data; and the point at time t in level L for the hip angle data. If we let $k_L(t)$ represent the knee angle data point in level L at time t , and $h_L(t)$ represent the hip angle data point in level L at t , we can write a conditional probability for occurrence of a particular value of the knee angle data as

$$P(k_L(t) | k_{L+1}(t), h_L(t)).$$

We sample randomly from this distribution for each point in time to create an initial guess for the knee angle data in level L .

Iterative Optimization. Because the sampling process used only a few features of the data, we do not yet have a solution that lies in a local maximum of the density when we include all of the features that are important. In particular, we have not taken account of the fact that each point depends on the points before and after it in time, in other words, the smoothness of the motion data. We now include these features and iteratively optimize for values with higher probability densities.

Again, we use a multivariate kernel based probability distribution to represent the data. However this time we use different features than we did in the initialization. Because we now have an initial guess for the L^{th} level at all time points, we can use points in time that follow as well as precede the point at time t we are optimizing. If we are optimizing the point at time t for the knee angle data, we use the points at time $t-2$, $t-1$, $t+1$, $t+2$, and the point in the hip angle at the same level at time t . Using the same notation as above, we could write the resulting conditional probability density as

$$P(k_L(t) | k_L(t-2), k_L(t-1), k_L(t+1), k_L(t+2), h_L(t)).$$

The optimization is performed by computing the derivative of this conditional probability distribution, evaluated at time t . This quantity is used to compute a step direction toward a higher density of probability. We used a constant step size of roughly $1/50$ the spread of the data for best results, and the procedure usually took about 10-20 iterations to converge. After all of the levels have been synthesized, they are added together to form the final knee angle data.

This same procedure was used to synthesize data for the hip angle, knee angle, ankle angle, x translation, and y translation. The hip angle was the first synthesized, so when initializing and optimizing no other joint angle data was included as a feature. The ankle was correlated to the knee, the x translation to the ankle angle, and the y to the hip angle.

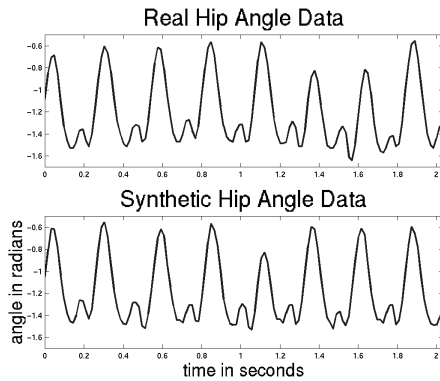


Figure 5: Comparison of real and synthetic hip angle data

3.5 Constraints

So far the synthesis technique has not taken into account the fact that the feet must contact the floor, not penetrate the floor, and not slide during contact. To enforce these constraints, we choose the points in time where the foot should remain in contact with the floor, and adjust the translation of the whole body to meet this position. Because the translational adjustment necessary turned out to be minor, no adjustment to the angles was necessary. To be sure the translations remained smooth, we took 3-5 time points before and after the times during which the feet were fixed to be in contact and subjected them to the same optimization procedure described above, but with the contact points fixed.

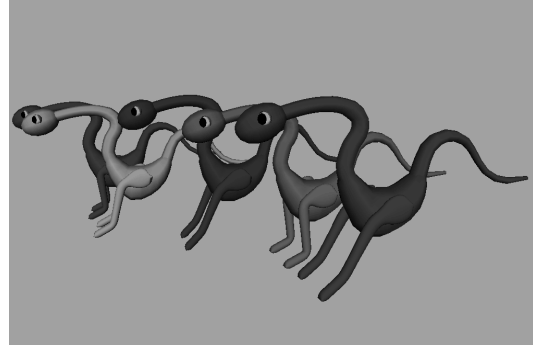


Figure 6: Example frame of the output animations

4 Results and Discussion

The technique described above was applied to the wallaby motion data. Starting with 128 points of training data, we synthesized 512 points of artificial data. We repeated the procedure several times independently, and each set of synthetic data looked as if it could have been real data, and yet the sets were different from one another. In other words, in addition to representing realistic motions, they had realistic noise and fluctuations. In figure 5 we show an example of synthetic hip angle data from one of these sets, and compare it to the hip angle training data. The synthesis technique maintained the proper correlations between the important features such as between joints, as can be seen, for example, by comparing the synthetic and real data in figure 3. We applied five sets of the synthetic angle and translation data to five computer models of a creature with limb lengths that are proportional to that of the wallaby. The creatures all hopped in a realistic manner, but not all exactly in the same way. A frame of the animation is pictured in figure 6.

This method of synthesizing data from a set of motion capture data may be useful for animation of repetitive motion such as walking or running when one wants to animate more steps than there is real data for. Another use may be in crowd sequences, in which the animator does not want all of the characters to move in exactly the same way, but wants to use a given set of motion capture data as the basis for the motion.

The next step is to apply this method to more complex three-dimensional motion. Work is in progress to use it for human walking, and so far the same procedure appears to work. The main difference is in that correlations to more joints must be included in the sampling. We also would

like to further integrate the optimization part of the synthesis with other non-probabilistic constraints to allow the animator to control features such as direction or stride length. It may also be possible to manipulate the motion data by enhancing or changing parts of it while keeping the rest as close to a “natural” motion as possible by maximizing the kernel-based probability distributions as described in the paper. Ultimately we hope to conduct human subject studies to further evaluate the quality of the generated motions.

Acknowledgements

We would like to thank Anna Rabinowicz for collecting and providing the video recordings, and NSF and the Stanford IMTV project sponsored by Intel, Sony, and Interval for partial support.

References

- [Bis95] C. Bishop, Neural Networks for Pattern Recognition, Oxford, 1995
- [Bod99] B. Bodenheimer, A. Shleyfman, J. Hodgins, The Effects of Noise on the Perception of Animated Human Running", Computer Animation and Simulation '99, Eurographics Animation Workshop, Sept. 1999.
- [Bra99] M. Brand. Voice puppetry. Proc. SIGGRAPH 1999.
- [Bre98] C. Bregler, J. Malik, Tracking People with Twists and Exponential Maps, Proc. Computer Vision and Pattern Recognition, 1998.
- [Bru95] A. Bruderlin, L. Williams, Motion Signal Processing, Proc. SIGGRAPH 1995.
- [Deb97] J.S. De Bonet, Multiresolution sampling procedure for analysis and synthesis of texture images. Proc. SIGGRAPH 1997.
- [Efr99] A. A. Efros, T.K. Leung, Texture Synthesis by Non-parametric Sampling, Proc. Int. Conf. Computer Vision, 1999.
- [Gle98] M. Gleicher, Retargetting Motion to New Characters, Proc. SIGGRAPH 1998.
- [Heg95] D.J. Heeger, J.R. Bergen, Pyramid-based texture analysis/synthesis. Proc. SIGGRAPH 1995.
- [Hod95] J. Hodgins, W.L. Wooten, D.C. Brogan, J.F. O'Brien, Animating human athletics. Proc. SIGGRAPH 95.
- [Per85] K. Perlin, An Image Synthesizer, Proc. SIGGRAPH 1985.
- [Per96] K. Perlin, A. Goldberg, Improv: A System for Scripting Interactive Actors in Virtual Worlds, Proc. SIGGRAPH 1996.
- [Sim98] E.P. Simoncelli, J. Portilla, Texture characterization via joint statistics of wavelet coefficient magnitudes. Proc. Int. Conf. Image Processing, 1998.
- [Str97] G. Strang, Wavlets and Filter Banks. Wellesley,-Cambridge Press 1997.
- [Unu95] M. Unuma, K. Anjyo, R. Tekeuchi, Fourier principles for emotion-based human figure animation. Proc. SIGGRAPH 1995.
- [Wit95] A. Witkin, Z. Popopic, Motion Warping, Proc. SIGGRAPH 1995
- [Zhu98] S.C. Zhu, Y. Wu, D.Mumford, Filters, random fields, and maximum entropy, Int. Journal of Computer Vision, 27(2):1-20, 1998.