

Generating Novel, Stylistically Consonant Variations on Human Movement Sequences

University of Colorado
Department of Computer Science
Technical Report CU-CS-1029-07

April 2007

Joshua M. Stuart
Elizabeth Bradley
David Capps
Jeffrey Luftig

In review, *Journal of Artificial Intelligence Research*.

Generating Novel, Stylistically Consonant Variations on Human Movement Sequences

Joshua M. Stuart

Biomolecular Engineering
129 Baskin Engineering
University of California at Santa Cruz
Santa Cruz, CA 95064 USA

Elizabeth Bradley

Department of Computer Science
University of Colorado
Boulder CO 80309-0430 USA

David Capps

Dance Program/Music Department
Hunter College
695 Park Avenue
New York, NY 10021

Jeffrey Luftig

Department of Engineering Management
University of Colorado
Boulder CO 80309-0433 USA

Submitted to the *Journal of Artificial Intelligence Research*.

Abstract

A common task in dance, martial arts, animation, and many other movement genres is for the character to move in an innovative and yet stylistically consonant fashion. This paper describes a corpus-based method for automatically generating such movement sequences. Our algorithms use the mathematics of chaos to achieve innovation and simple machine-learning techniques to enforce stylistic consonance. We first construct a mapping between an original motion sequence and a chaotic attractor. Chaos’s hallmark *sensitive dependence on initial conditions* guarantees that variations generated from this mapping are different from the original; the fixed structure of the attractor guarantees that the variations resemble the original in both esthetic and mathematical senses. The effects of this mimic a choreographic variation technique that dates from the early 1960s: pasting together segments of a given dance in a new order. As one would imagine, that mechanism can create abrupt transitions at the segment boundary, which calls for some smoothing. Simple interpolation—cf., morphing—is not a good solution here, as it is guided by mathematical constraints (smoothness, shortest path, etc.) and so its results are not necessarily consistent with a specific movement style. Our algorithms employ simple graph-theoretic methods to learn the “grammar” of joint movements in a given corpus, and then search those graphs to find stylistically consistent interpolation sequences between pairs of body postures.

1 Introduction

Musical or choreographic variation—one of the most fundamental of compositional techniques—are based upon two phases of work: establishment of a grounding theme and a series of structured departures

from that theme. Classic examples are Bach’s *Goldberg Variations* for the keyboard or Balanchine’s *The Four Temperaments* ballet. The properties of dynamical chaos provide unusual mechanisms for achieving these effects. The characteristic patterns of strange attractors can be used to capture the underlying theme of the sequence and chaos’s sensitive dependence on initial conditions can be used to produce variations. This intriguing notion was proposed by Diana Dabby in the mid-1990s in the context of music and image[20, 21]. Inspired by Dabby’s work, we set out to apply some similar ideas to dance. The result was the pair of tools that are described in this paper: CHAOGRAPHER, which produces chaotic variations of keyframed movement sequences, and MOTIONMIND, which smooths any abrupt transitions that may be introduced by that variational mechanism.

The core of CHAOGRAPHER’s strategy is to create a mapping that associates a sequence of body positions with a chaotic attractor, and then use that mapping to generate variations. The fixed attractor structure guarantees that the variations will resemble the original in a mathematically precise sense, while sensitive dependence on initial conditions guarantees that each variation will be different. From an esthetic standpoint, variations generated in this fashion are both pleasing and strikingly reminiscent of the original sequences. Loosely speaking, they resemble the original pieces, but with shuffled subsequences. Our work differs from Dabby’s in several ways, beginning with the representation. Musical notation is well established and straightforward, but capturing the state of the human body is far more complicated, as described in Section 2. The mathematics of our mapping is different from Dabby’s in some formal ways that are detailed in [14, 15]. And while musical instruments can play arbitrary pitch sequences, subject to instrument range and performer ability, kinesiology and style impose a variety of constraints on consecutive body postures in movement genres. This becomes particularly important when subsequences of a piece have been shuffled, as in CHAOGRAPHER’s algorithm, because the ending posture in one subsequence may be very distant, in “body space,” from the beginning posture of the subsequence that follows it. This issue, which arose in the dance world in the 1960s when choreographers began exploring the use of randomization in performance, was the catalyst for the line of research described in Section 4. The end result of that line of work was the MOTIONMIND tool, which uses directed graphs and Bayesian networks to capture the patterns that are inherent in a corpus of movement sequences. These data structures can then be used to generate original *stylistically consonant* movement sequences. One can do this via a directed search to find a “tweening” sequence between two prescribed positions, or one can generate free-form original movement simply by “walking” these graphs.

CHAOGRAPHER’s shuffles and MOTIONMIND’s stylistically consonant movement sequences are presented and discussed in sections 3.2 and 4.3, respectively. As described in section 6, all of these ideas apply to other kinds of sequences that have characteristic patterns, such as flight simulators.

2 Representing Human Motion

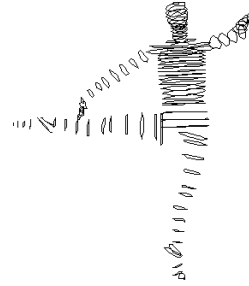
The human body moves using a complex combination of gross and fine articulations and many different representations have been developed to capture its state. To mathematically model the body using a tractable number of parameters, the representation used in our work simplifies the body’s degrees of freedom into 23 main joints. This model neglects some of the smaller joints (e.g., the individual fingers) and treats the spine as four rigid segments rather than 24 individual vertebrae. The orientation of each joint is specified with a *quaternion*, a standard representation in rigid-body mechanics that dates back to Hamilton[30]. A quaternion $q = (r, \vec{u})$ consists of an axis of rotation \vec{u} and a scalar r that specifies the angle of rotation of the joint about \vec{u} . A single body position, in this representation, translates to 23 descriptors (**pelvis**, **right-wrist**, etc.), 92 floating-point numbers (four for each joint), and a variety of information about the position and orientation of the center of mass. An example is shown in Figure 1(a). We represent motion as a series of quaternion-based snapshots, or *keyframes*, which are evenly spaced in time. Such a sequence can be rendered graphically into an animation using a variety of software packages; in this paper, we use LifeForms (tm), a commercial animation tool that is common in the dance community. See Figure 1(b) for an example of LifeForms output. This tool uses spline

```

Location, 0, 1, 0, 0
pelvis, 0, 0, 0, 1
atlantal, 0, 0, 0, 1
cervical_1, 0, 0, 0, 1
...
right sternum, 0, 0, 0, 1
right shoulder, -.23485, -.68735, .12987, .22236
...

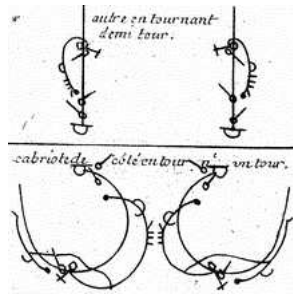
```

(a)

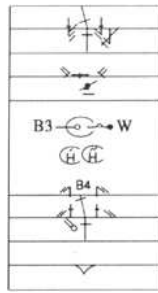


(b)

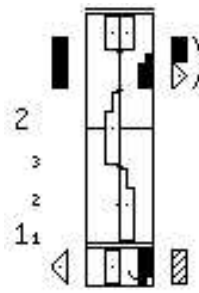
Figure 1: Representing and rendering human motion: (a) a quaternion-based description of a body posture (b) a LifeForms rendering of that posture.



(a)



(b)



(c)

Figure 2: Human movement representations used in the dance community: (a) the Feuillet notation for Baroque-era court dance (b) Benesh notation (c) labanotation. These images are reproduced, with permission where copyright exists, from [26], [1], and [2], respectively.

interpolation to smooth or “tween” between the keyframes, which can be spaced coarsely and unevenly in time when generated by human animators. The computer science community has many other tools (e.g., AutoDesk’s Maya) and representations (e.g., ASF/AMC, originated by Acclaim, or Biovision’s BVH) for capturing and rendering human motion, some of which are designed for use in animation or motion-capture work, where data are gathered or produced automatically and frame rates are very high.

Dance notation has a very interesting history, and its representations are very different than the type of keyframed sequences described in the previous paragraph. Rather than describing *position*, for instance, these representations specify *motion*, and most do not focus on individual joints. In the Baroque era, for instance, Feuillet notation was used to record the floor plan and movement sequences of European court dances. An example is shown in Figure 2(a). Symbols referring to fairly complex pre-determined movement sequences—“steps” like a demi-tour¹—were placed along maps of the movement of the body as a whole through space, and often keyed directly to notations of musical accompaniments. Importantly, this system relied on fore-knowledge of the steps themselves, which for the most part had verbal names that were central parts of training methodologies for dancers. Steps that were not in this vocabulary could not be represented with this notation. This specialization, while reducing expressiveness and flexibility, makes Feuillet notation comparatively easy for human practitioners to learn and use.

In the 20th century, Rudolf Benesh and Rudolf Laban created movement representations that were

¹“half turn”

much more general and expressive. Both were designed to be style-neutral, relying solely on knowledge of the dynamics of the human body, rather than on experiential familiarity with the subject movement itself and its specialized semantics (e.g., names of specific steps, such as pliés). These notational systems are harder for humans to learn because they are more expressive, but they are much more widely useful. Benesh notation[17], which is currently utilized particularly by ballet dancers, graphs positions on a staff similar to musical notation using symbols to indicate posture, body part by body part. See Figure 2(b) for an example frame of Benesh notation. In this representation, “movement lines” are added to augment the visualization of action by indicating changes between key frame positions. Time is represented, as in music notation, moving left to right along the staff, with the equivalent of musical “rests” and time signatures indicating the duration of transition from position to position. Spatial coordinates and facings are added below the staff, and expressive qualities like force or strength are indicated via symbols common in musical notation, like *fff* (fortissimo.)

Labanotation[36], widely used in a variety of movement disciplines and shown in Figure 2(c), also indicates time via a linear staff. Unlike Benesh and music notation, the staff is vertical, moving from bottom to top. It focuses not on sequences of positions, but on directional actions through space: a particular symbol describes a direction of movement in relation to the core and front facing of the body, and the placement of the symbol within the staff attaches that action to a particular body part. The vertical scale of the symbol indicates the duration of the action; a lexicon of graphic modifiers specifies details such as degree of rotation, degree of contraction or extension, and contact with objects or other bodies. Significantly, Labanotation places information about the action of the weight bearing structures (typically feet, but possibly knees, shoulders, pelvis) at the core of the staff, so that displacement through space (locomotion), indicated by the same symbols applied to body parts, is immediately visualized. Laban also devised a system of analysis and description of performative qualities of movement that generalizes the actual actions but specifies in great detail the experiential qualities of the movement. In this “Laban Movement Analysis” scheme, observations of time (fast/slow), space (indirect/direct) and force (light/strong) are combined to generate a grid of possibilities such as punch (fast, direct, strong) or flick (fast, indirect, light), which are indicated in the score via specialized symbols.

The tension between these different notational approaches—particularly the tradeoff between expressiveness and unwieldiness—brings up a variety of issues that are interesting from an artificial intelligence standpoint. Many dancers and choreographers find Labanotation difficult to use, for example. The decomposition of the body into parts is just not a natural way for people to think about motion. Style-specific representations like Feuillet notation, on the other hand, are much more natural for their user communities, but they obviously fail if the move to be described is not in their vocabulary. Labanotation can capture arbitrary movements, but its complexity makes it awkward for its users. The tradeoff is fundamental, and it is intertwined with many other influences: our learned and implicit neuromuscular patterns (coordination, symmetry), coupled with the physics of the world in which we move (momentum, friction, gravity). All of this makes the notion of a movement “motif”—a concept that dancers use freely and effectively, but that completely defies definition—quite slippery. For computer modeling in the digital age, when vast corpora of motion-capture data and powerful machine-learning techniques are available, a highly expressive representation is essential. But distilling the patterns of coordination, style, and motif out of the results—and then using them to create original stylistically consonant movement—is nontrivial. This cluster of issues will resurface at several points in this paper.

Our choice of human-motion representation was driven by the need for generality and automation: we wanted to be able to represent arbitrary movements, and in a way that meshed easily with available animation software. We build finite representations of movement patterns by discretizing the joint angles in the body. This solution is similar to both Benesh and Labanotation in that it decomposes the body into small elements. The elements are different, though: joints (like the elbow or wrist) instead of body *parts* like the head or the upper arm. Like Benesh, our representation describes keyframed positions of these elements, not their motions. Unlike any of the schemes employed by dancers, we use numbers, rather than colors and shapes of icons, to describe the state of those elements. In our algorithms, each joint λ can take on a finite number M^λ of allowed orientations; in practice, $M^\lambda < 400$. Formally, we define Q^λ as the set of *allowed* orientations for joint λ and then replace the *actual* orientation of the

joint with the closest quaternion in Q^λ . We can express a body position \vec{b} as a discretized vector \vec{s} by setting each of its components s_λ equal to the quaternion in Q^λ that is closest to b_λ . We can do this in $\log(M^\lambda)$ time using K-D trees[28] to represent the Q^λ sets.

The strategy described at the end of the previous paragraph is analogous to “snapping” objects to a grid in computer-drawing applications. While this quantization is useful for computer movement representation, it has several interesting problems when used to capture human motion. Deriving a kinesiological and esthetically successful discretization of joint states, for instance, was unexpectedly difficult. Simply performing an even quantization of the quaternion variable values—that is, classifying all orientations between, say, (**right-knee**, 1, 1, 0, 1) and (**right-knee**, 1, 1, 0.2, 1) as an equivalence class and representing them in the algorithms as a single posture—produced visibly awkward results. The individual frames were all but indistinguishable when viewed side by side, but the *animations* looked quite different. An example can be found at www.cs.colorado.edu/~lizb/chaotic-dance.html. The human visual perception system appears to be very sensitive to small variations in quaternion coefficients of motion sequences: small changes in a single coefficient can violate the motif of the motion. In ballet, for instance, linear motion is the rule and zigzags—“jaggies” introduced by a quantization scheme—can be quite startling. The same issue arose when we transformed the quaternion data to Euler angles and then did the discretization in that space. The solution on which we eventually settled uses a non-uniform quantization scale for Q^λ that was created by hand by an expert dancer. This scale contained more ‘clicks’ in some joint-angle ranges than in others, and it differed from joint to joint. Note that this discretization issue would not arise if we had used a coarser-grained, stylistically rooted representation like Feuillet, where a single symbol describes a predefined, atomic piece of movement. But such a representation would obviously have constrained both what we could represent and how we could manipulate it—in ways that are inappropriate for this work. Moreover, representing movement using a notation that had a unique description for every possible body position would create a very large state space, and learning in that space would require enormous amounts of training data.

As described so far, our representation captures joint orientations statically and in isolation, completely disregarding the kinesiological constraints that govern how they move together. For example, if the shoulder is in its resting position with the palm facing the thigh, the elbow can bend freely. If the upper arm is turned 180 degrees on its long axis—rotating the thumb inwards, past the leg, until it points backwards—the elbow cannot bend far before the anterior shoulder ligaments complain. Other parts of the body, too, affect this reasoning—via connection/flexibility or even via collision. The upper leg and hip, for instance, can physically interfere with the elbow movement in the example above. Considering each joint’s movement in isolation also disregards the notion of movement *style*, which can be understood as another limiter of movement possibilities, similar to joint anatomy. For example, in classical ballet, the elbow/arm unit seldom crosses the midline of the body and the torso almost never articulates regions of the spine in isolation from one another. This is a function of relationships between sets of joints and cannot be deduced from individual joint-angle sequences, considered in isolation. To produce stylistically consonant motion, then, one must also consider the correlations between joints.

In order to capture these constraints, we needed to incorporate a good model of joint coordination—one that takes into consideration both kinesiological and stylistic delimiters—into our representation of human motion. The most complete and general approach to this problem would be to model the interactions between each joint and every other joint in the body, or perhaps even between *combinations* of joints (e.g., whether the right hand crosses the centerline of the body). Doing so, however, engenders a combinatorial explosion in the search space. There are sensible ways to reduce the complexity of the problem. To a first approximation, for instance, a joint is not influenced by every other joint in the body. The orientation of the wrist, for instance, strongly affects the orientation and movement of the fingers but has little effect on the toes. We put this simplifying assumption into effect by using a Bayesian network[32] to explicitly represent the relationships between how different joints move. This network reflects the structure and physics of the human body: a tree with the pelvis at the root. Three branches lead from this root to nodes corresponding to the right hip, the left hip, and the lower spine. Each hip joint is the parent node to a knee, and so on. We assign a conditional probability distribution, estimated from the a corpus of human motion, to every (parent,child) pair in the tree. For every combination

of states that a parent λ and its child μ can assume, the distributions estimate the probability that joint μ is in orientation r given that joint λ is in orientation q , for every pair of discretized quaternions $q \in Q^\lambda, r \in Q^\mu$. This is of course only a rough approximation, and a better idea would be to learn the true relationships that are implicit in the inter-joint correlations. That would be a more general way to capture kinesiology and style, and it could even elucidate hidden linkages (e.g., a hip injury that causes unusual shoulder movement). We are currently exploring this idea using dynamic Bayesian networks[22] and a corpus of motion-capture data that is described at more length in the conclusion.

Note that none of this analysis and representation of inter-joint coordination would be needed if we had used a style-linked strategy like Feuillet notation, where the atomic representational units are rich enough to encapsulate every aspect of the corresponding dance steps. This very richness, however, makes these representations completely inflexible about any departure from those fixed steps.

3 Chaotic Variations on Movement Sequences

Given a human motion sequence, represented as described in the previous section, the algorithms described in this section use the mathematics of chaos to generate a new sequence that resembles the original in the sense of a variation on a theme. [15] describes the mathematical details of this approach and discusses its implications from the standpoint of nonlinear dynamics. This section begins with a brief overview of chaotic dynamics, then summarizes the chaotic variation algorithm and briefly reviews its results.

3.1 Chaos 101

Chaos is a type of complex and yet highly patterned behavior that arises in deterministic dynamical systems². One of the canonical examples in the field is the Lorenz system[43], a set of three coupled ordinary differential equations that comprise a highly simplified model of flow in a heated fluid:

$$\begin{aligned} \dot{x}(t) &= 16y(t) - 16x(t) \\ \dot{y}(t) &= 45x(t) - y(t) - x(t)z(t) \\ \dot{z}(t) &= x(t)y(t) - 4z(t) \end{aligned} \tag{1}$$

The *state variables* of this system— x , y , and z —are physical quantities like the convective intensity in the fluid, which vary with time as the dynamical system evolves. Their time derivatives are indicated with dots: $\dot{x}(t)$ and so on. The constant values in the terms on the right-hand side of the system of differential equations represent different physical conditions for the problem (e.g., the value 45, which represents how much heat is being applied to the fluid). The xy and xz terms on the right-hand side of equations (1) make this system nonlinear, which is a necessary condition for chaos.

If one starts the system (1) at some initial condition $[x(t_0), y(t_0), z(t_0)]^T$, follows the trajectory of its evolving state $[x(t), y(t), z(t)]^T$ for $t > t_0$, and plots that trajectory in the *state space*—whose axes are the state variables—one sees a *strange attractor*, as shown in Figure 3(c) or (d). Strange attractors have a variety of fascinating properties:

- they are covered densely by any trajectory starting in their *basins of attraction*,
- their trajectories exhibit *sensitive dependence on initial conditions*,
- they often have some fractal structure, and

²Those whose state evolves in a manner that is fully determined by the previous state.

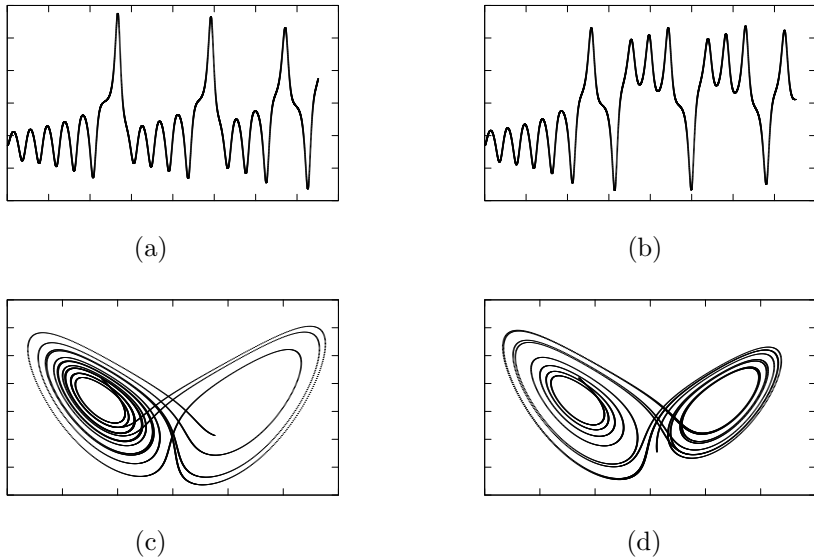


Figure 3: The hallmarks of chaos: sensitive dependence on initial conditions in the context of a fixed, highly characteristic attractor geometry. (a) and (b) show time-domain plots of the x -components of trajectories from two nearby initial conditions in the canonical Lorenz system; (c) and (d) show the same trajectories in the state space (plotted here in an x vs. z projection). A small perturbation has a large effect on the time course of the system state, but the overall time-asymptotic pattern in the state space remains the same.

- they cannot be calculated in closed form, but rather the equations must be solved numerically in order to produce them.

See [12, 53] for introductions or [54] for a more-comprehensive treatment of these concepts.

The critical features of chaos, for the purposes of this paper, are the first two bullets of the list above. If one starts the Lorenz system from some other, nearby initial condition $[x(t_0) + \epsilon, y(t_0) + \epsilon, z(t_0) + \epsilon]^T$, the associated trajectory will relax to and cover the same attractor—unless, of course, the change in initial conditions bumps the trajectory out of the basin of attraction—but it will trace out that attractor’s pattern *in a very different order*. Figure 3 shows how this plays out; note the obvious difference between the top two plots, which show the time evolution of two nearby initial conditions, and contrast that to the similarity of the structure of the bottom pair of images, where the same information is plotted in the state space. This is the so-called *butterfly effect*: a small perturbation can have a very large effect upon the evolution of a chaotic system. Edward Lorenz first reported this behavior in a 1963 paper[43] entitled “Deterministic Nonperiodic Flow.” The term “chaos” was coined twelve years later[41].

This fixed attractor structure provides an element of order and predictability in a chaotic system: any trajectory that is started in an attractor’s basin will follow the same overall, time-asymptotic pattern. Sensitive dependence, however, makes chaotic systems effectively unpredictable, even though the future evolution of the system state is fully determined by the current state. One *can* prove that the trajectory will cover the attractor, but there is no way—without infinite-precision arithmetic and perfect measuring devices—to determine where that trajectory will be at a given time, nor the order in which it traces out the lobes and curves of the attractor. In essence, sensitive dependence on initial conditions magnifies small-scale effects that we do not see, such as floating-point arithmetic, into large-scale effects that we can measure. This combination of large-scale order and small-scale “mixing” is not only ubiquitous in science and engineering, but also highly intriguing, and it has a variety of practical applications, ranging from spacecraft control to heart-attack prevention[11, 52]. The following section explains how to exploit these properties to create chaotic variations on movement sequences.

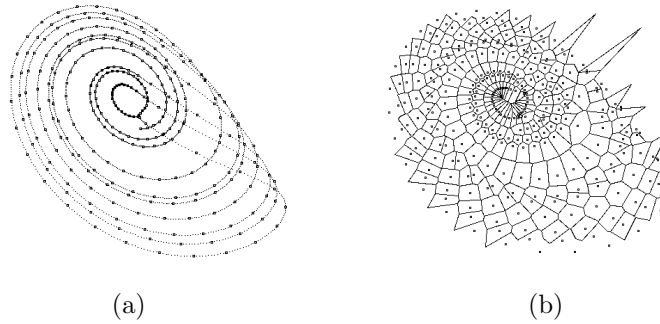


Figure 4: Generating a tiling of a chaotic attractor: A Voronoi diagram is constructed from the points of a reference trajectory on the attractor—the dots in part (a)—to divide the space occupied by that attractor into non-overlapping patches or *cells*. The order in which the original trajectory traverses those cells defines the temporal order of the *cell itinerary* that corresponds to that reference trajectory. The trajectory points in this image are projected into 2D for this presentation, but the attractor and the Voronoi diagram are both 3D in reality.

3.2 Chaographer

CHAOGRAPHER’s task is to create a mapping of a dance onto a chaotic attractor, then use that mapping to generate new dances. To accomplish this, it first generates a *reference trajectory*, like the one in Figure 4(a), to define the attractor geometry. It then “wraps” a keyframed sequence of body positions, represented as described in Section 2, around that attractor, associating successive keyframes in the sequence with successive state-space patches that are traversed by the reference trajectory. In order for this mapping to be useful for generating variations, these patches must cover the attractor and they must not intersect. We chose to implement this patchwise division of state space using a Voronoi diagram[46] on the reference trajectory points, as shown in Figure 4(b). A Voronoi cell around one of these points is the region of state space that is closer to that point than to any other. One draws Voronoi cells by constructing and intersecting the perpendicular bisectors of every adjacent pair of points, as shown in Figure 4(b), but our actual implementation is a nearest-neighbor calculation that, again, uses K-D trees[28] to reduce computational complexity.

The cells of a Voronoi diagram like Figure 4(b) define a *partition* of the space occupied by the attractor, which has some interesting mathematical implications that we discuss elsewhere[15]. The reference trajectory defines a *cell itinerary* on these cells: one can imagine it lighting up those cells in some fixed sequence as it moves through them. A trajectory from some other initial condition in the basin of attraction of the attractor will move through those same cells, but in a different order—and therein lies the variation mechanism. We map the keyframes of the original sequence to successive cells in the partition, as shown in Figure 5. To generate a variation, we simply start another trajectory at some different initial condition and play back the postures associated with each cell that it touches.

Figure 6 shows an example of a chaotic variation produced in this manner. The first three frames of the variation follow the original verbatim; the sequence order then diverges, inserting a single frame from the end of the original, then repeating a two-frame subsequence. It is impossible to appreciate this kind of result from a textual description or a static picture of a series of frames; see the animations at www.cs.colorado.edu/~lizb/chaotic-dance.html. Chaotic variations generated in this fashion bear an obvious resemblance to the original movement sequences, which makes sense because both trajectories are moving on the same attractor, onto which the original motion sequence has been mapped. The variations are also clearly different from the originals. Broadly speaking, the variations resemble the originals with some shuffling of coherent subsequences. Note that this kind of *ordinal* shuffle does not appear in classic musical variations like Bach’s. Later work, such as Lukas Foss’s Baroque Variations or “sampled” music like that of Pierre Schaefer’s “musique concrète,” does use ordinal shuffles—sometimes of multiple pieces—in developing musical themes. While this kind of ordinal shuffle does not fall under

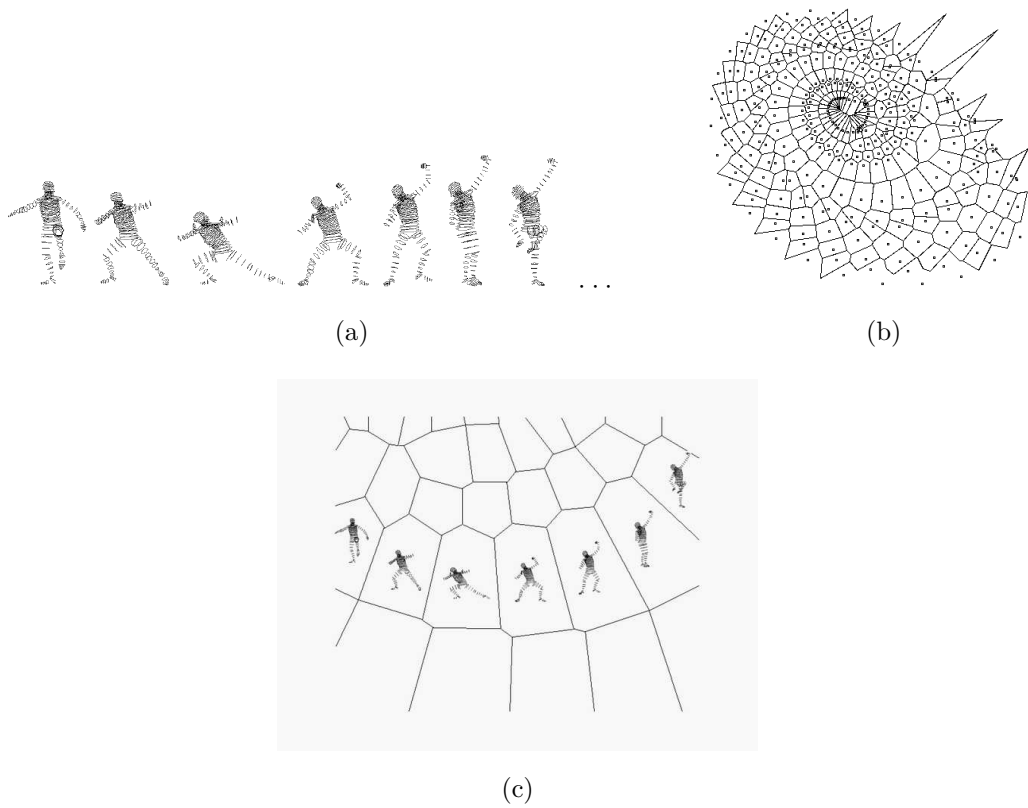


Figure 5: A mapping that links a keyframed dance sequence and a chaotic attractor. Successive body positions from the movement sequence (a) are mapped to successive cells of Voronoi-partitioned chaotic attractor (b), producing a mapping that is schematized in (c)

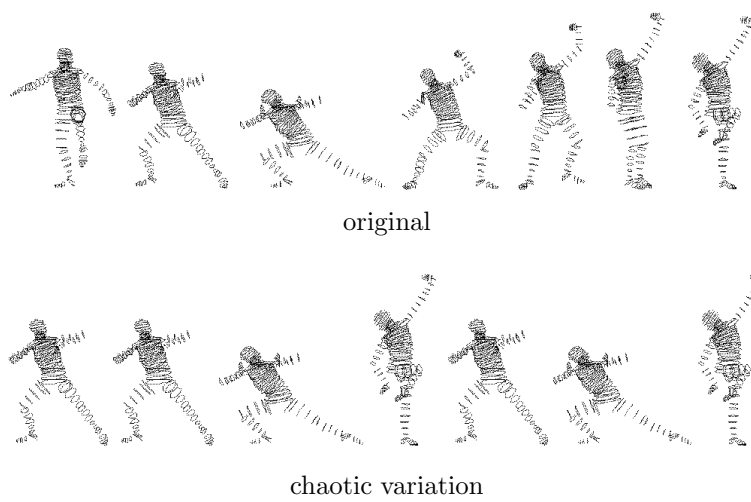


Figure 6: A chaotic variation generated using the strategy schematized in the previous figure.

the classic definition of variation³, it has recently moved very much into the mainstream. Hip hop DJs, for instance, often use turntables to create a new piece by looping and mixing segments of an existing one.

The modern dance community has also explored the idea of an ordinal shuffle. In the early 1960s, the noted choreographer and dancer Merce Cunningham began to experiment with *aleatory* choreography: techniques of constructing movement and choreographic sequences that incorporated chance. This was part of a revolutionary shift in dance composition that introduced the idea that movement could be decomposed and manipulated by means other than the kinesthetic logic that is rooted in the body’s neuromuscular system, thereby removing much of its thematic nature[18]. One of Cunningham’s strategies was to compose motion sequences for different quadrants of the body and then combine them in arbitrary ways; in another, he used coin tosses or other randomization techniques to shuffle the temporal order of the phrases of a dance and invent new movement possibilities. Many of these techniques have since entered the dance vernacular; chunkwise shuffling, in particular, is used to this day by novices and professionals alike for generating choreographic materials. The chaotic variation technique described in the previous paragraphs essentially automates that process. An important difference is the chunking: a choreographer’s notion of what comprises an atomic piece of dance depends on esthetic and stylistic constraints, among other things. CHAOGRAPHER simply ends the subsequence when it crosses into a neighboring patch—an effect that follows solely from the mathematical landscape of the chaotic system, not the dance—and chooses the starting point of the next chunk by examining which patch it has now entered. As in Cunningham’s techniques, this provides a mechanism for innovation. In our experience, the dance community has been quite receptive to the notion of mathematically generated movement; not only has our work been warmly received at Dance/Technology conferences (e.g., [13]), but dancers have even adopted moves created by our algorithms. The music community, in contrast, was initially quite resistant to the notion of applying chaos to respected composers like Bach[19].

It is interesting to contrast chaotic variations with random shuffles of the keyframes in the same original sequence, particularly in view of the mathematical techniques that are typically used by rendering software to transform a keyframed sequence into a smooth animation. The critical difference is that while the randomly shuffled sequences contains the same postures as the original piece, they do not preserve any of its subsequence structure. An example is available on the on the website listed above. Like many animation tools, the LifeForms software that we use to generate these movies uses splines to interpolate (or ‘tween’) between keyframes, freeing its user from the onerous task of specifying body postures at the animation frame rate. The effect of this is to smooth the transitions in the random shuffles, creating some apparent structure. This is simply an artifact of the LifeForms animation process, however, and the random shuffles bear little temporal resemblance to the original at any timescale beyond that of a single pair of keyframes.

Note that the chaotic mapping does *not* constitute a model of the motion, nor are we claiming that human motion is chaotic. Rather, the chaotic attractor—and the mapping of the motion sequence onto it—is simply a ‘blender’ that chops up and mixes the motion sequence. One could also accomplish this using random variables: i.e., index randomly into the sequence, play a chunk of random length, repeat. The results of this, as shown on our website, are similar to those produced by our scheme—indeed, Von Neumann’s original random number generator was a chaotic system, as were Cunningham’s “randomization” mechanisms—but using chaos in the manner that is described in this section enforces more constraints on the randomization so that original patterns are more frequently generated.

Chunkwise shuffles of motion sequences have an unusual and important property: the ending position of one chunk and the starting position of another may be quite different, requiring the dancer to make an abrupt transition if the two are pasted together. The transition between the third and fourth posture of the variation sequence of Figure 6, for instance, would be somewhat wrenching⁴. The adagio sequence on our webpage is a particularly good demonstration of this effect. These kinds of abrupt transitions

³“In the context of Western art music, the term commonly means elaboration of the melody or accompaniment; other kinds of modifications, such as development or transformation, are often considered to be outside the scope of variation.” [48].

⁴Recall that frames in this sequence are evenly spaced in time.

occurred in Cunningham’s phrase-shuffled work as well, and caused it to be met by substantial resistance from audiences, dancers, and critics when it was first introduced. (It is now a well-accepted phenomenon in modern dance, though, and is no longer startling to viewers from that community.) The following section presents a corpus-based technique for movement interpolation that can be used to fill in these kinds of gaps in a manner that remains faithful to the style of the movement genre.

4 Stylistically Consonant Interpolation

Given a pair of body positions, A and B, and a corpus of keyframed movement sequences, the algorithms described in this section—embodied in the MOTIONMIND tool—construct a movement sequence that starts at A, ends at B, and is consistent with the style that is implicit in that corpus. The simplest way to approach this is to use traditional interpolation techniques like splines to interpolate between the discretized, quaternion-valued representations \vec{s}_A and \vec{s}_B that correspond to these positions. The morphing techniques that have been so effective in computer graphics (e.g., [61]) do exactly that; as mentioned on page 10, the animation software that we use to produce movies also uses splines to interpolate or ‘tween’ between its keyframes. Simple interpolation techniques, however, do *not* work well for human motion. Splines, for instance, take the shortest path, subject to various continuity constraints, through the interpolation space, but human joints may not move in those ways. The shortest path from a head rotation of -120 degrees to one of +120 degrees—from looking over one shoulder to looking over the other—would be a rotation through facing backwards, which is not physically possible. Spline-based animation tools often produce these kinds of glitches. Clearly some sort of algorithm that captures both real physical constraints and behavioral patterns is necessary; building such an algorithm *a priori*, however—as in the work of Jessica Hodgins and collaborators (e.g., [33, 34])—is extremely difficult. Another approach is to use machine-learning techniques to build models of movement patterns from corpora of human motion, and that is what we have chosen to do.

A critical design parameter in any such strategy is the choice of state variables—in our case, the descriptive granularity. We first tried working with entire body positions and treating them as atomic objects, but the results were unsatisfying[15]. This approach was too highly constrained, from a creative standpoint: it could only use full-body postures that had been observed in the corpus, verbatim. And the enormity of the associated state space—one state for every possible full-body position—means that training any machine-learning strategy for it would require a huge amount of data. A better idea is to interpolate in a *joint-wise* fashion: to build graphs to capture the orientations and motions of each joint, and Bayesian networks to capture their inter-relationships. This approach, which motivated the representation described in Section 2, can patch together joint orientations observed in different keyframes to obtain body-posture sequences that are novel, kinesiologyically valid, and stylistically consonant. A preliminary version of this strategy was described in [55]; the rest of this section describes our final version and assesses the results using a simple Turing Test.

4.1 Capturing Joint Movement Patterns

We capture the patterns in a joint’s motion using a transition graph built from a movement corpus. Vertices in these graphs represent particular joint orientations, represented as described in Section 2. Edges correspond to the movement of the joint from one orientation to another. The corpus is used both to identify orientations that the joint assumes and to estimate the corresponding transition probabilities between orientations. Note that our graphs are finer grained than the “motion graphs” that are used in the graphics community[39], whose vertices represent temporal subsequences of the original motion and whose edges capture transitions between those clips. Our graphs capture a *jointwise* decomposition of the motion. To build them, we first discretize every body position in the corpus, so that a consecutive pair of body positions (\vec{a}, \vec{b}) , each consisting of 23 continuous-valued quaternions, becomes the discretized pair (\vec{s}, \vec{t}) where \vec{s}, \vec{t} each consist of 23 *discretized* quaternions. We then build a transition graph G^λ for each joint λ ; G^λ contains M^λ vertices—one for each allowed orientation of that joint. We

record the fact that joint λ is allowed to move from a_λ to b_λ by introducing an edge in G^λ from vertex s_λ to vertex t_λ . We then assign a weight to this edge that models the “unlikeliness” with which such a transition occurs in the corpus, calculated using the negative log-likelihood of observed transitions. At the same time, we build a Bayesian network that, for each joint orientation, keeps track of the observed orientations of all of the “parent” joints in the body. (Recall that this strategy, described at the end of Section 2, models inter-joint coordination by capturing the conditional probability distribution of observed orientations of joints that are parent-child pairs in a network whose topology mimics the human body.) See [55] for the mathematical details of all of these calculations.

Figure 7 shows a transition graph for the right ankle that was constructed in this fashion from a 575-posture corpus of cartwheels. The complex topology of this graph—and of the graphs corresponding to the other joints in the body, which are equally complicated—reflect the intricacy of human motion. Even though joint angles are quantized, the number of joints and associated degrees of freedom makes the number of vertices quite large. The branching factor is also high; a joint can move at different speeds along any of its degrees of freedom, accessing a large number of different next states. Long, linear vertex chains like the one that is magnified in Figure 7 are introduced into the joint transition graph when one movement sequence in the corpus progresses through orientations that do not occur in any other sequence. This is a common effect in small corpora. A large, rich corpus translates to a heavily connected graph, reflecting that a given joint orientation has been accessed along different paths. This has some interesting effects on the results, as described in the next section. If the motion is much slower than the frame rate, the graphs become variants of birth-death chains, wherein a joint can either stay where it is or move to its immediate-neighbor states in joint-angle space. This situation arises in motion-capture data, where frame rates exceed 120 per second, and is discussed at more length in Section 6.

Taken together, these joint transition graphs and Bayesian networks capture the patterns and correlations in joint movement. The search strategy in the following section uses that information to construct stylistically consonant full-body movement sequences between two prescribed positions. One can also, as described in Section 6, generate *free-form* movement that follow the corpus’s patterns by walking these graphs in different ways.

4.2 Using Joint Movement Patterns to Interpolate

We use a memory-bounded A* search strategy[60] to find an interpolation subsequence that moves smoothly between two discretized body postures \vec{s}_A and \vec{s}_B . Recall that A* finds a path from an initial state to a goal state by progressively generating successors of the current state in the search. The algorithm places successor states on a priority queue, sorted according to a score that estimates the cost of finding a goal state. In the next iteration, the state with the best score is drawn from the priority queue, its successor states are computed and added to the queue, and the procedure is repeated until a goal state is found or until the queue is empty. In this application, the states in the A* search space are *body states*: that is, 23-vectors of discretized quaternions that represent full body positions. Note that all 23 joints must reach the goal state \vec{s}_B *at the same time*, which makes the search more interesting—and more complicated.

To generate successors of some body state \vec{s} , we generate joint orientations recursively using the joint graphs and the Bayesian network. We first use the pelvis graph to generate the possible orientations of the pelvis⁵. Next, for each possible state x of the pelvis, we generate the possible states of each of the pelvis’s child joints (the hips and lumbar spine), *given that the pelvis will be in state x* . This recursive, top-down generation of body states allows us to filter out joint states that have zero or low probability from the search, thus keeping the effective average branching factor of each joint as small as possible.

The scoring for this search also involves both the joint graphs and the Bayesian network. Recall that a joint graph captures the movements of an individual joint, on the time-scale of a single keyframe. Edge

⁵The successors of the joint-state s_λ are those vertices in G^λ that are connected to s_λ by an edge directed away from s_λ .

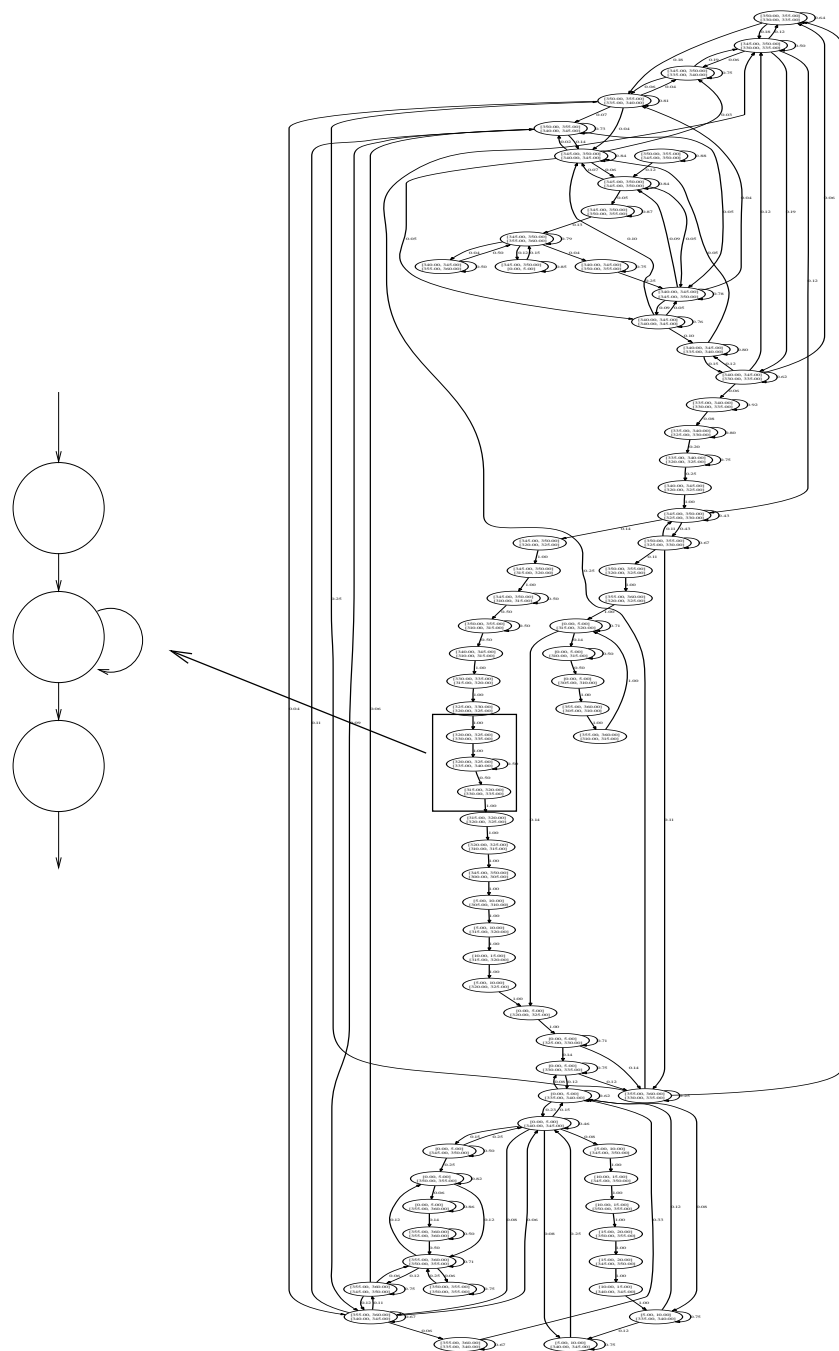


Figure 7: A transition graph that represents the movement patterns of the right ankle in a corpus of cartwheels from the Graphics Lab website at the Carnegie-Mellon University[3]. The numbers in each state identify the discretized orientation of the joint; edge weights represent the probabilities with which the ankle moves between states. Isolated vertices—i.e., ankle orientations that were not observed—have been omitted in the interests of clarity. The left-hand image shows a magnified view of the boxed area in the right-hand image.

weights in each joint graph correspond to the unlikeliness of the associated movement, so that smaller weights connect joint orientations that are more frequent. (The unlikeliness is defined as one minus the probability that the joint takes the transition, as estimated from a given corpus.) The Bayesian network essentially modifies these probabilities, based on orientations of parent joints, to reflect the inter-joint coordination of the human body. To score a single whole-body movement, we use both of these data structures and sum the resulting unlikeliness values of the movement’s constituent joint motions. To score a movement *sequence*—a partial path between body positions \vec{a} and \vec{b} , for instance—we simply sum the scores of the single-keyframe whole-body movements that make up that sequence. To estimate the score for the path from some intermediate state to the goal state, we use Dijkstra’s algorithm[23] to find the shortest-path costs for each joint and then sum those costs. As required for A*, this heuristic will always underestimate the true cost to the goal state; see the appendix for a short proof.

As it expands successors, MOTIONMIND’s memory-bounded A* search (MA*) stores only a certain fixed number of partial paths at any given time and “forgets” any paths that have a score worse than the minimum score in the current queue. This constitutes a kind of beam search because it only looks at local body states near its current set of discovered paths. It is not guaranteed to find the globally optimal solution—which is not a goal here—but it can save time avoiding low-probability (i.e., less-promising) paths of body states. At each step of the search, MA* must generate K successors, score each of these, and add them to the queue. The naive approach is to score an exponential number of successor states. If the average branching factor, b , of each joint is small, such a brute force approach may be possible. The corpus used for this paper generates joint orientation transition graphs with an average branching factor of approximately 2. Thus, the MA* search had to score fewer than $K = 2^{23}$ possible states. While the b^{23} term does not grow asymptotically with the length of the number of paths considered, it is a large constant that adds significantly to the running time. As the average branching factor of the joints increases—for example using a richer movement corpus, such as motion-capture data—successor generation using this naive approach may make the entire search infeasible. The running time of the search is the product of K , the total number of successor states generated at any given point in the search, and L , the number of total partial paths considered before finding the first goal state. In the worst case, L is exponential in the number of body states in the graph and super-exponential in the number of joint orientations: $L = b^{\prod_{\lambda=1}^{23} M_{\lambda}}$. In practice, though, L is often much less even when the joint graphs are sparse. If this becomes a problem, one could add a parameter to terminate the search if the goal state is not reached before a tolerated number of paths have been explored.

Although not implemented in the current version of MOTIONMIND, we note that the generation of body states lends itself neatly to a dynamic programming solution, which would make successor state generation efficient when the set of possible joint orientations is large. A Baum-Welch forward and backward pass[9] can find the top ν most probable body states in time $K = 2 * 23 * O(b + \nu)$ instead of the exponentially many in the worst-case example above. Rather than score all possible successors of the current body state in each step of the search, then, we could generate the best-scoring ones and put them on the priority queue if their scores are better than the minimum score already in the queue.

There are a variety of opportunities for improvement here. Our search strategy is greedy: it ignores the cost of paths and scores nodes in the search based solely on the estimated distance between them and the goal. This can create “inefficiencies” in the interpolation sequences—places where the animated character appears to be headed towards the goal state, but then moves away. An appealing alternative would be to incorporate the path weights up to the current point in the solution as part of the scoring function, which should allow the search algorithm to find shorter, more-direct sequences. The esthetics place an extra layer of requirements and constraints on these kinds of “improvements,” however: simply taking the highest-probability branch can be a significant source of cliché. Incorporating more of the physics of motion into the process—e.g., adding constraints on the position, velocity, and acceleration of the center of mass, so the momentum of the body is conserved as it passes through the interpolated sections of the movement—makes practical sense, but does not fit smoothly into the A* framework. The topology that we chose for the Bayesian network may not accurately reflect the human body; the middle back, for instance, may influence the arms directly, rather than through the upper back and shoulder linkage in the child-parent scheme described above. It would clearly be better to learn the

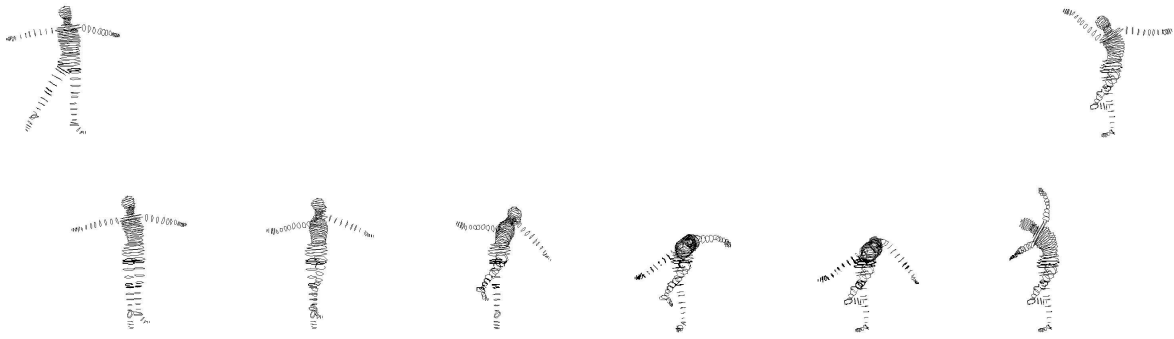


Figure 8: Stylistically consonant interpolation: the six-keyframe sequence at the bottom of the figure interpolates between the two body positions at the top in a manner that is consistent with the patterns in a ballet corpus.

inter-joint coordination patterns from the corpus, say using dynamic Bayesian networks, rather than making a *priori* assumptions about them. We are in the process of exploring this approach.

4.3 Stylistically Consonant Interpolation: Results and Evaluation

Figure 8 shows an example of the results produced by the algorithms described in Sections 4.1 and 4.2. The task presented to MOTIONMIND is to interpolate between a specified pair of body positions in a manner that is consonant with a corpus of ballet sequences. The starting and ending poses are shown at the top of the figure. The ‘before’ and ‘after’ torso positions (top left and top right, respectively) are only subtly different, but the weight-bearing and foot positioning differs significantly between the two, as are the arm positions and head orientations. The corpus included 38 short ballets comprising 1720 individual body positions, drawn primarily from the LifeForms PowerMoves CD. We used the representation of Section 4.1 to capture the joint-movement patterns in this corpus, then used the algorithms of Section 4.2 to search the resulting graphs for a path between these two positions. The result was the six-frame interpolation sequence shown across the bottom of Figure 8, which moves between the specified positions in a manner that is consistent with the style of this corpus. First, the right leg comes forward to the extended forward low direction, making possible the succeeding forward shift of weight into a lunge⁶ stance. The torso/arm unit follows with a forward/side/back (called a “port de bras”) sequence that is often associated with the lunge position in the ballet lexicon. The transition to the given final pose is solved with a simple lift of the right arm. This sequence satisfies a number of stylistic tendencies in classical ballet:

- a contralateral relationship between upper and lower limbs
- a spatially direct cause/effect relationship between leg gesture and shift of weight
- large-kinesphere articulation of torso only in the absence of locomotion
- torso gesture in peripheral pathway; absence of isolational use of torso parts
- peripheral pathways of arm gestures, using the arms as units rather than articulating extensively between upper and lower arm, generally in response to torso posture movement
- constant outward rotation of legs
- larger proportion of gestural action than locomotor action

⁶one knee bent, the other extended, with both legs weightbearing

Note that these characteristics are not programmed into MOTIONMIND *a priori*. They are present in the interpolated sequence only because our algorithms are able to effectively extract and use the patterns that are in the corpus. Many port-de-bras appear in that corpus, for example, but none of them are associated with the specific lower-body position that appears in this interpolated sequence. This is a key point: MOTIONMIND has invented a physically *and stylistically* appropriate way to move the dancer between the given positions. Recall that these postures were not simply pasted in verbatim from the corpus; they were synthesized *joint by joint* using the transition graphs and Bayesian network-directed A* search, and their fit to the genre is strong evidence of the success of the methods described in the previous section. We have used different variants of this strategy to interpolate between dozens of other pairs of postures. The interpolation subsequences so constructed included a variety of stylistically consistent and often innovative sequences; among other things, the interpolation algorithms used *relévés*, *pliés*, and *fifth-position rests*⁷ in highly appropriate ways—and all with no hard coding.

From a purely functional standpoint, the results have some room for evolution, as the sequences do include a few transitions that appear awkward to many human viewers. And occasionally an interpolated sequence is extremely long, using a great deal of seemingly unrelated motion to accomplish an apparently simple movement task. In one such instance, where the task was a simple 90-degree rotation of the right upper arm around its long axis, the algorithm constructed an 65-move sequence that involved much leg and trunk movement. While this sequence was stylistically consonant, it was highly discursive. This outcome was a by-product of working with a corpus that was really too small to be representative of human motion. As mentioned in conjunction with Figure 7, idiosyncratic motions—joint angles that appear only once in the corpus, in a single movement sequence—leave isolated, linear vertex chains in the graphs. This forces the search algorithms to use the entire linear sequence in order to access any of the vertices that appear in it. And since the search algorithm is designed to return equal-length paths for each joint, an idiosyncratically long sequence in one joint will force MOTIONMIND to “pad” all the other joints’ movement sequences to match. While this filled-in movement does abide by the constraints of the corpus and is thus stylistically consonant, it can be discursive. When the corpus is larger, the resulting movement graphs are generally more richly connected, which gives the search algorithms more leeway and reduces the occurrence of discursive motion in the results. Until recently, motion corpora were quite limited, so this was a real issue in our work. Motion-capture technology has become much more widespread, though, and many laboratories are producing richer corpora. As described in Section 6, we are working with the Graphics Lab at Carnegie Mellon to gather motion-capture data on different kinds of movement genres, ranging from ballet to contact improv and the martial arts.

Taking the Bayesian network constraints out of the search heuristic had some extremely interesting effects. To the layman’s eye, the resulting sequences look jerky and unappealing, so we expected negative comments about them from professional dancers. However, it seems that an uncoordinated path through a classical ballet corpus is a very good way to generate unconventional sequences, and the results can be inventive and appealing to modern dancers: “Wow! I’m going to use *that* move in my next piece!” In retrospect, this makes some sense: the modern dance genre actively works at violating expectations of movement appropriateness that have been received from traditional forms like ballet. (Choreographer Martha Graham went as far as to praise those potentially unappealing movements as “divine awkwardnesses.”)

Animated movies of all of the sequences discussed in this section are available on the web[4].

4.4 A Turing Test

As a final piece of evaluation, we offer a simple Turing Test of our results. In the 1950s, Alan Turing suggested that one could evaluate machine intelligence using a blind test, in which human subjects are

⁷A *relève*, a rising up onto the balls of the feet, is a common component of direction shift in ballet. A *plié* is a lowering of the center of gravity, generally through a knee bend. Fifth position is a two-footed stance with legs rotated outward and crossed toward the mid-line.

Source of Variation (SV)	Degrees of Freedom (df)	Sums of Squares (SS)	Mean Squares (MS)	F	p-value	Importance
Main Effects:						
Method (A)	1	51.07	51.07	4.095	0.0528 (<i>df</i> = 1, 27.7)	$\omega^2 = 4.00\%$
Class within Method (B(A))	8	15.99	2.00	1.52	0.1654	N.A.
Sequence within Method (C(A))	18	212.28	11.79	8.93	0.000	$\sim 16.97\%$
Interaction Effects:						
Class by Sequence within Method (BC(A))	72	95.37	1.32	3.80	0.0000	$\sim 5.80\%$
Residual (e)	2070	718.37	0.35			
Blocked Effects:						
Rater within Method \times Class (D)	230	183.7851	0.80			

Table 1: ANOVA – Question 1: Awkwardness

presented with a program’s output and asked to determine whether or not that output was produced by another human[8]. In our test, we showed twenty short motion sequences—ten constructed by a human (‘Human’) choreographer/ animator and ten constructed by these interpolation algorithms (‘Computer’)—to several classes of students at the University of Colorado at Boulder and at Harvard University between 1999 and 2004. The classes were unequal in size, ranging from 19 to 34 subjects. Each subject (‘Rater’) was asked to rate each sequence on three constructs based on the following ordinal scales:

<i>Awkwardness</i>	awkward	1 2 3 4 5	natural
<i>Physical Plausibility</i>	impossible	1 2 3 4 5	possible
<i>Esthetic Appeal</i>	unpleasing	1 2 3 4 5	pleasing

This design resulted in the generation of a Model III (Mixed Effects) 3 Way Analysis of Variance (ANOVA) for each of the three questions. ‘Method,’ the treatment (*A*) was, of course, a fixed effect. ‘Class’ (*B*) and motion ‘Sequence’ (*C*) were treated as random effects, as was the resultant Interaction between them (*BC*). Within subject variability (*D*) was treated as a blocked effect. Class and Sequence, as well as their Interaction, were nested within Method. The Appropriate Error Terms (AETs) for each of the three effects tested were established using the methodology described by Scheffe[51]; the Quasi-F ratio for Method was constructed using the approach described by Winer[59]. As the data were ordinal in nature, the raw scores generated by each subject were transformed (X_T) using the square root transformation recommended by Dixon and Massey[24] prior to executing the ANOVA.

Complete appreciation of the results of the three ANOVA tables that follow requires an understanding that an algebraic combination of the Class and Sequence effects, as well as their interaction, constitute the AET for Method (that is, $B(A) + C(A) - BC(A)$). Where Method was determined to be statistically significant ($\alpha = 0.05$), the importance of the observed difference was calculated as an Omega-Squared (ω^2) coefficient, as detailed by Winer[59]. For each of those cases where a random effect was found to be significant, an Intraclass Correlation coefficient (ρ_I) was calculated as a measure of the importance of the observed effect[37]. This was done only to facilitate the interpretation of the results associated with the Method effect.

Table 1 presents the results of the ANOVA for Question 1: ‘Awkwardness.’ As shown by this analysis, a statistically significant difference in the mean transformed ratings associated with each Method was observed. Table 2 presents the results of the ANOVA for Question 2: ‘Physical Plausibility.’ As shown by this analysis, there was no statistically significant difference observed in the mean transformed ratings

Source of Variation (SV)	Degrees of Freedom (df)	Sums of Squares (SS)	Mean Squares (MS)	F	p-value	Importance
Main Effects:						
Method (A)	1	16.62	16.62	1.19	0.2859 (<i>df</i> = 1, 24.63)	ω^2 =N.A.
Class within Method (B(A))	8	9.85	1.23	0.94	0.3172	N.A.
Sequence within Method (C(A))	18	252.40	14.02	10.70	0.000	~22.02%
Interaction Effects:						
Class by Sequence within Method (BC(A))	72	94.13	1.31	4.09	0.0000	~6.32%
Residual (e)	2070	654.04	0.32			
Blocked Effects:						
Rater within Method \times Class (D)	230	154.7447	0.67			

Table 2: ANOVA – Question 2: Physical Plausibility

associated with each Method. The difference in the transformed mean ratings tested corresponded to raw data averages of 3.36 for the Computer-generated sequences, versus 3.68 for the Human-generated sequences. Table 3 presents the results of the ANOVA for Question 3: ‘Esthetic Appeal.’ As shown by this analysis, a statistically significant difference in the mean transformed ratings associated with each Method was observed. As shown by the analyses reflected by the three ANOVA output tables, a statistically significant difference in the mean transformed ratings associated with each Method was observed for Awkwardness and Esthetic Appeal, but no statistically significant effect could be detected for Physical Plausibility. Table 4 shows the mean and median for the transformed and raw data values across all Classes and Sequences.

In summary, the Human-generated sequences were found by the Raters to be significantly more natural and more pleasing than the Computer-generated sequences. While no statistically significant difference in the two sets of sequences could be discerned in Physical Plausibility, it is interesting to note that the observed differences in the two sequence means were consistent with the differences associated with Esthetic Appeal and Awkwardness. In terms of importance as a function of explained variability, the greatest difference observed between Human and Computer generated sequences was associated with Esthetic Appeal. The inability to detect a significant difference between the levels of Method studied, as well as the relatively low level of importance detected for the two cases where a statistically significant difference was, in fact, observed, is due to the same condition. Specifically, the explained variability within each method due to differences between Sequences was relatively large, and was accompanied by a significant interaction between Class and Sequence. In other words, not only was there a large difference in how each Sequence was rated within their respective Method levels, but each Class did not evaluate each of the Sequences uniformly. Both of these contributions to variability in the model are components of the AET associated with Method. Therefore, it might be reasonable to surmise that had there been more uniformity among the Sequences associated with the Methods tested, the differences due to Method may have appeared even more pronounced, and the superiority of Human-versus Computer-generated effects on all three questions tested may have been observed.

5 Related Work on Movement, Style, and Variation

Human movement has been studied in great detail by the graphics, vision, and machine-learning communities, as well as in biomechanics. A large amount of this effort has been devoted to recognizing

Source of Variation (SV)	Degrees of Freedom (df)	Sums of Squares (SS)	Mean Squares (MS)	F	p-value	Importance
Main Effects:						
Method (A)	1	114.61	114.61	14.17	0.0023 ($df = 1, 13.25$)	$\omega^2 = 10.2\%$
Class within Method (B(A))	8	12.33	1.54	1.69	0.1156	N.A.
Sequence within Method (C(A))	18	134.35	7.46	8.20	0.0000	$\sim 12.01\%$
Interaction Effects:						
Class by Sequence within Method (BC(A))	72	65.73	0.91	2.90	0.0000	$\sim 4.11\%$
Residual (e)	2070	649.31	0.31			
Blocked Effects:						
Rater within Method \times Class (D)	230	148.84	0.65			

Table 3: ANOVA – Question 3: Esthetic Appeal

Question	Descriptive Statistics: Transformed Data		Descriptive Statistics: Raw Data	
	Mean	Median	Mean	Median
Awkwardness				
Computer	3.545	3.732	2.81	3.00
Human	3.847	3.732	3.33	3.00
Physical Plausibility				
Computer	3.853	4.236	3.36	4.00
Human	4.021	4.236	3.68	4.00
Esthetic Appeal				
Computer	3.535	3.732	2.76	3.00
Human	4.236	4.236	3.58	4.00

Table 4: Means and Medians

and tracking the body and its parts in various kinds of 2D image data. These are hard and interesting problems, but different from the one addressed here. Our *input* is a 3D model in the form of limb lengths and joint angles—the kind of output that is produced by software that deduces body position from a collection of still images—and our goal is to model and use the progressions and correlations in those data.

Representations are always key, and several groups (e.g., Mataric *et al.*[27] or Pentland[56]) have focused on how to construct good primitives for representing movement. Our representations and algorithms, in contrast, are not intended to help *humans* understand motion or style; they are designed simply to allow a machine to duplicate it. This means that they can work at a finer grain—individual joint movements, rather than movement clips or even motifs—and that they need not operate under the kinds of constraints that arise in, say, inverse kinematics problems.

There have been hundreds of good papers on recognizing, analyzing, quantifying, and understanding various aspects of motion, especially gait (e.g., [40]) and hand gesture (e.g., [10]). We are interested in how the *whole body* moves, as well as how it moves differently in different movement genres, and our goal is to generate original movement that follows a given style, not to recognize the motion—let alone understand or decompose it in any detail. Several other groups have produced interesting results in this area. Some of this work, not surprisingly, has used inverse kinematics (IK). [62], for instance, uses planning and a data-driven constraint-based IK to attain naturalness. [44] also uses IK, tuning a couple of its parameters (joint stiffness, hip swing) to alter the style of a given movement sequence in physically meaningful ways—e.g. to introduce a limp into a walk. Other approaches that have proved to be useful in creating and/or adapting “natural-looking” motion include optimization of appropriate objective functions[25], combinations of IK and optimization[45], dynamical modeling & control theory[34], and detailed musculoskeletal modeling[38]. All of these approaches devote significant analytical effort to the modeling process, whereas we learn the models from the motion-capture data. This makes our approach better at capturing the vagaries of real-world movement data, and thus easier to apply to a new body—one that may violate *a priori* modeling assumptions in subtle ways. The work of the Graphics Lab at Carnegie Mellon is probably the most closely related to ours. In [49], for instance, motion is decomposed by body part, modeled individually using different machine-learning techniques, and combined via an ensemble method that strives for naturalness. Our techniques are different, but the goals are similar and the methods require a bit less tuning.

In the past few years, some other strategies have been proposed specifically for learning movement style from a corpus of examples: using singular-value decomposition[58], principal-components analysis[57], hidden Markov models[16], probability distributions over the space of all possible poses[31], linear models[35], and even nonlinear optimization techniques[42]. The representation used in our work (joint graphs + Bayesian networks) has a significant advantage over these approaches in that it allows us to make good use of having multiple instances of a given movement, or of a given movement style. This is critically important to physical plausibility of the results, and hence to generalization of the strategy.

The movie industry, of course, has devoted a tremendous amount of effort to computer-generated variations on motion sequences. WETA, for instance, used MASSIVE in the *Lord of the Rings* trilogy to generate thousands of battles sequences automatically. None of this work has used dynamical chaos—the fundamental variational technique in the work described here. As described in Section 1, chaos *has* been used to create *musical* variations, and that was the catalyst for the project described in this paper. Chaos has also been used to generate music from scratch as well (e.g., [47]), but the results are not at all consonant with any established musical style.

There are a few other groups working at the intersection of computer science and dance, including NYU, Arizona State, and Ohio State. The NYU group shares our specific interest in using motion-capture data for analysis and synthesis purposes; among other things, they are working on a way to deduce Laban Movement Analysis characterizations (see page 4) automatically[5].

6 Conclusions and Future Directions

The algorithms described in this paper do a surprisingly good job of duplicating some of the efforts of human choreographers. CHAOGRAPHER uses a chaotic mapping to shuffle an original sequence, introducing a strong element of variation. The stretching and folding along the attractor guarantee that the ordering of the movements in the chaotic variation differs from that in the original sequence. At the same time, the fixed geometry of the attractor ensures that the variation resembles the original piece—not in the classical theme-and-variations formula used in Western music and dance, but via an *ordinal shuffle*, a technique that became popular among human composers of music and dance in the late 20th century. Because ordinal shuffles reorder subsequences of a piece, however, they can introduce abrupt transitions, where the ending posture of one subsequence is physically very different than the beginning posture of the subsequence that follows it. To smooth these transitions, we developed MOTIONMIND, a group of corpus-based schemes that employ directed graphs and Bayesian networks to capture and enforce the dynamics of a given group of movement sequences. To interpolate between two positions, one searches these graphs; to generate free-form movement, one walks them. Either way, the resulting movement sequences are both novel and also consistent with the style that is inherent in the corpus from which they were built.

The notion of stylistic consonance is not easy to define or measure. Dancers and choreographers have studied in great detail the ways—at the individual, artistic, and cultural levels—in which movement tendencies and choices tend to fall into discernible ranges that are relatively narrow in relation to the range of possibilities of the human body in time and space. As a result, there are some standard rules, procedures, and patterns in certain dance and martial arts genres that can be used to evaluate MOTIONMIND’s results. As described in [55], we went through a variety of dances, frame by frame, with experts who were trained in “movement analysis,” a certificate program offered by many dance departments. With a few exceptions, these experts proclaimed the sequences to be stylistically consonant. The Turing Test in Section 4.4 did not completely agree with this. The subjects in that test saw a clear difference between human- and MOTIONMIND-generated sequences in terms of awkwardness and appeal, but rated them similarly in terms of physical plausibility. Intriguingly, expert dancers have a different take on this than non-dancers: they noted the differences, but *perceived the awkwardnesses as appealing*. We only had data from six subjects in this category, however, so those results are inconclusive and those data are not included in Section 4.4. We are in the process of redesigning our Turing Test to explore these issues further.

Analyzing CHAOGRAPHER’s results is even more problematic, as the dance community has no guidelines for quantifying the choreographic effectiveness of a particular ordinal shuffle. When we showed our results to expert dancers, though, we were delighted to find that our algorithms had duplicated an approach used by one of the most innovative and respected choreographers of the 20th and 21st centuries. “No one has revised ‘the fundamentals’ more fundamentally than Merce Cunningham; for example: [...] compositional practices based on the use of ‘chance operations,’ (producing new strategies for linking together disparate phrases of movement)...” [18]. Interestingly, humanists fall into nonlinear dynamics terminology to describe Cunningham’s work—and some of John Cage’s musical techniques: “*Chance operations* is the marvelously oxymoronic phrase that Cunningham and Cage employ to describe the ‘rules’ (or ‘operations’) that govern these interactions. As with many complex systems, the resulting behavior is both deterministic and unpredictable.” [18]. Some interesting quotes from Cunningham’s own thinking appear in appendix A. An original performance piece entitled CON/CANTATION, which was based on CHAOGRAPHER’s variations of an original dance composed by the authors of this paper, premiered in Boston in April 2007 to warm audience reception[4].

In terms of runtime, evaluation is straightforward. CHAOGRAPHER shuffles a thousand-frame sequence in less than a minute on any modern CPU⁸. MOTIONMIND is far more intensive. It takes tens of seconds to build the joint-transition graphs and Bayesian networks from corpora containing on the order of a thousand body positions, but searching those data structures for an interpolation path can require

⁸The run time is generally $n \log n$ in the number n of frames in the input sequence.

much longer. These numbers depend on the number λ of joints in the representation, the number M_λ of unique orientations that each joint can take on, and the branching factor (b) of the graphs that track those transitions. In our current implementation, $\lambda = 23$, M_λ ranges from 50 to a few hundred, and b is roughly two. Bearing in mind that the worst-case computational complexity of the search is $O(KL)$, where $K = b^{M_\lambda}$ and $L = b^{\prod_{\lambda=1}^{23} M_\lambda}$ (see page 14), it is clear that this will be a serious issue if the corpus is richer.

Motion-capture systems, one way to gather better data about human movement, reconstruct a three-dimensional model of a moving object from a series of simultaneous views from cameras arrayed around that object. Reflective markers are used to identify salient locations in the object—joint positions, in the case of a human performer. The motion-capture (“mocap”) software reconstructs the 3D positions of these markers, tracks how they move, and deduces how the connections between them (i.e., the limbs) evolve. This is vastly less onerous than entering each frame by hand with animation software—the manner in which the corpora used in this paper were produced—so mocap data are generally more comprehensive and more realistic than animations. This increased realism should enable MOTIONMIND to produce better results. Recall from page 16, for instance, that a particular joint orientation that appears only once in the corpus may force the interpolation algorithms to produce discursive movement. If the graphs were built from a corpus that adequately sampled the full richness of human motion, each joint orientation would likely be visited many times in many different progressions, giving the interpolation algorithms more leeway in the search. Though this will increase the computational complexity, as discussed below, mocap data is very much worth the trouble because it captures what real people are doing, and in accurate and anatomically appropriate ways.

To explore this further, we are collaborating with the Carnegie-Mellon Graphics Lab to obtain mocap data. These data sets use 52 markers that delineate 30 body parts and are recorded at 120 frames/second. This is far better data than the corpus that was used to produce Figure 8 (1720 frames of ballet, each with 23 joints, entered by hand by human animators). Indeed, it is almost too good. The high frame rate—which makes the data pile up very rapidly—not only is unnecessary for the types of motion that we study, but it actually affects the topology of the graphs. Each position is sampled many times and every micro-movement is recorded and re-recorded, so every vertex has a high-probability self-loop edge and only a few nearby low-probability successor states. For these reasons, we downsample the mocap data to 5Hz. Even so, it overwhelms our search algorithms. Because the joints’ motions are richer, the graphs contain many more states (M_λ is many hundreds instead of 50-100) and they have a higher branching factor— $n \approx 3$ on the average, if the quantization step is five degrees, instead of the $n \approx 2$ in the corpora used to create Figure 8. In graphs like this, finding an interpolation path for even a highly simplified body⁹ can take tens of minutes. We are working on improving our algorithms to handle this complexity.

Joint coordination is the key here, we believe—not only to prune the search space, but also to capture the essence of movement style. Our simple, static Bayesian network is an extremely rough approximation of coordination, but even that turned out to be important—not only in streamlining the search, but in achieving stylistic consonance. When we removed the Bayesian network from the A* score, the search time increased¹⁰ and the results did not adhere to the movement style at all. The results of a coordination-free search of graphs built from a ballet corpus, for instance, looked like a combination of modern and Irish step dance. Coordination is clearly fundamental to movement, and one really should learn it from the data, rather than assuming its structure *a priori* as we did. To this end, we are currently exploring the use of dynamic Bayesian networks[29] to extract these coordination patterns automatically from mocap data. It will be important to learn not only which pairs of joints are related, but also examine *combinations* of joint movement. In ballet, as mentioned previously, the hand rarely crosses the centerline of the body. This is a relationship between two different sets of joints: the {shoulder, elbow, wrist} combination, which specifies where the hand is, and whatever pair of joints one chooses to define the body’s centerline. This is related to the inverse kinematics problem in robotics, and also to the notion of what comprises a movement motif. There is evidence that human

⁹pelvis, femurs, lumbar spine

¹⁰Recall that the Bayesian network effectively reduces b in the $O(bM_\lambda)^\lambda$ equation.

motion occupies only a subset of the high-dimensional space: “most dynamic human behaviors are intrinsically low dimensional with, for example, legs and arms operating in a coordinated way[50]”. Note that stylistically rooted representations like Feuillet notation avoid all of these issues completely because a huge amount of coordination is implicit in the packets of movement that are captured by their symbols.

All of this begs another important question: whether or not analyzing movement and/or enforcing stylistic consonance *at the joint level* is the right thing to do at all. It is not clear, for instance, how to tell from a joint transition graph like Figure 7 whether the corpus involved ballet or modern dance, or whether the dancer has a sore ankle. One can do some simple reasoning about the graphs: presumably a trained dancer would have wider range of motion than a novice, which would translate to more vertices and probably a higher branching factor. Beyond that, analysis becomes difficult. The issue, again, is that joint *coordination*, not orientation, appears to be the key to style—and not just individual joints, or pairwise combinations of joints. Rather, movement “motifs” are an emergent phenomenon of interactions between different *sets* of joints, all moving under the constraints to physics (gravity, balance, momentum, etc.). These influences do leave their signatures in the graphs described in Section 4, which makes those graphs useful for computer algorithms like MOTIONMIND whose goal is to construct stylistically consonant interpolations. They are not, by themselves, a very good tool for *understanding* motion.

Interpolation is not MOTIONMIND’s only product; one can also use the joint-transition graphs and Bayesian networks described in Section 4.1 to generate free-form original movement that fits a given genre. The obvious strategy for doing this fails in interesting ways, however. A simple graph walk that follows the highest-probability edge sequence produces clichéd movement. If the keyframe rate is much faster than the movement time scales, as in motion-capture data, the results are even worse: because the highest-probability edge from any vertex is always the self loop, every joint freezes in place. Injecting some randomness into the walk makes things more interesting. Note that this kind of graph walk does not suffer from the computational complexity problems mentioned above, which derive from the constraints of the directed search (*viz.*, the combinatorial explosion engendered by the need to have all the joints move in synchrony between the designated orientations).

Dance is not the only application of these ideas. CHAOGRAPHER and MOTIONMIND can be usefully applied to any sequences that have characteristic patterns. Colleagues of ours have used our code to generate chaotic variations of the words in Shakespeare’s sonnets and Bush’s 2004 inauguration speech, as well as frames in movies and various image decompositions. There are also many potential engineering applications. A flight simulator, for example, presents pilots with scenarios comprised of timed subsystem failures. Certain patterns are common (e.g., engine 1 coughs and then fails). Expert simulator training personnel extract these patterns from records of emergency situations and then use them to craft the scenarios presented to the trainees. The methods presented in this paper could be used to learn these patterns from corpora of flight recorder data and concoct new training scenarios that put them together in unexpected—and yet consistent—patterns.

Acknowledgements

The authors would like to thank P. Clary, D. Dabby, A. Hogan, D. Jurafsky, N. Rojas-Adame, A. Rubin, E. Schell, S. Schroeder, M. Seltzer, E. Sheehan, D. Trowbridge for code, suggestions, and comments. Jessica Hodgins and the members of the CMU Graphics Lab have been instrumental in helping us with motion-capture data, references, and many other things. This work was supported in part by NSF NYI #CCR-9357740, a Packard Fellowship in Science and Engineering from the David and Lucile Packard Foundation, a Dean’s Small Grant from the College of Engineering and Applied Sciences at the University of Colorado, and a Radcliffe Fellowship from the Radcliffe Institute for Advanced Study.

Appendix A* — Search Heuristic

Proof that Dijkstra's shortest path distances are optimistic

Let h be the current estimate of the cost to the goal state, which we can write as the sum of the shortest path costs for each joint: $h = h_1 + h_2 + \dots + h_k$, where h_i is the shortest path cost of joint i . Suppose there is a path from the current state to the goal state with a lower cost, z . This cost can also be written as a sum of the costs of the joint transitions: $z = z_1 + z_2 + \dots + z_k$, where each z_i is the cost for joint i to transition from the current state to the goal state. This implies that there is at least some joint i where $z_i < h_i$. This is a contradiction since this means there is a lower cost path for joint i not found by Dijkstra's algorithm.

Appendix A – Thoughts on Ordinal Variation in Music and Dance

Merce Cunningham on dance:

Regarding *Solo suite in space and time* (1953):

- “The spatial plan for the dance, which was the beginning procedure, was found by numbers the imperfections on a piece of paper (one for each of the dances) and by random action the order of the numbers. The time was found by taking lined paper, each line representing five inch intervals. Imperfections were again marked on the paper and the time lengths of phrases obtained from random numbering of the imperfections in relation to the number of seconds.”
- “You do not separate the human being from the actions he does, or the actions which surround him, but you can see what it is like to break these actions up in different ways, to allow the passion, and it is passion, to appear for each person in his own way.”
- “A large gamut of movements, separate for each of the three dances, was devised, movements for the arms, the legs, the head and the torso which were separate and essentially tensile in character, and off the normal or tranquil body-balance. The separate movements were arranged in continuity by random means, allowing for the superimposition (addition) of one or more, each having its own rhythm and time length. But each succeeded in becoming continuous if I could wear it long enough, like a suit of clothes.”

Referring to solos from 1953: “All were concerned with the possibility of containment and explosion being instantaneous. The trilogy used chance procedures in the choreography, sometimes in the smallest of fragments and at others in large ways only.”

Referring to *Space* (1963): “In *Space*, the dances had possibilities for improvisation within a space scale.... Within a section the movements given to a particular dancer could change in space and time and the order the dancer chose to do them in could come from the instant of doing them....”

Hip hop culture:

“Cutting and pasting is the essence of what hip-hop culture is all about for me. It's about drawing from what's around you, and subverting it and decontextualizing it.” DJ Shadow[6].

“I look at all the different parts and see how I can organize them in a way. It's like maths. Very mathematic. It's like graphs!” Blockhead[7].

References

- [1] <http://www.benesh.org>.
- [2] <http://www.dancenotation.org>.
- [3] <http://mocap.cs.cmu.edu>.
- [4] <http://www.cs.colorado.edu/~lizb/chaotic-dance.html>.
- [5] movement.nyu.edu/projects/labani.html.
- [6] <http://to-the-quick.binghamton.edu/issue%202/sampling.html>.
- [7] <http://www.trip-hop.net/interview-10-Blockhead.html>.
- [8] Computing machinery and intelligence. *Mind*, 59:434–460, 1950.
- [9] L. Baum. An equality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3:1–8, 1972.
- [10] M. J. Black and A. D. Jepson. Eigenttracking: Robust matching and tracking of articulated objects using a view-based representation. In *ECCV (1)*, pages 329–342, 1996.
- [11] E. Bradley. Autonomous exploration and control of chaotic systems. *Cybernetics and Systems*, 26:299–319, 1995.
- [12] E. Bradley. Causes and effects of chaos. *Computers and Graphics*, 19:755–778, 1995.
- [13] E. Bradley, D. Capps, and A. Rubin. Computers and choreography. In *International Conference on Dance and Technology*, pages 12–16, Tempe, AZ, 1999.
- [14] E. Bradley and J. Stuart. Using chaos to generate choreographic variations. In *Proceedings of the Fourth Experimental Chaos Conference*, pages 451–456, 1997.
- [15] E. Bradley and J. Stuart. Using chaos to generate variations on movement sequences. *Chaos*, 8:800–807, 1998.
- [16] M. Brand and A. Hertzmann. Style machines. In *Proceedings of SIGGRAPH*, 2000.
- [17] M. Causley. *An Introduction to Benesh Movement Notation*. Max Parrish, London, 1968.
- [18] R. Copeland. *Merce Cunningham: The Modernizing of Modern Dance*. Routledge, 2004.
- [19] D. Dabby. Personal Communication.
- [20] D. Dabby. Musical variations from a chaotic mapping. *Chaos*, 6:95–107, 1996.
- [21] D. Dabby. A chaotic mapping for musical and image variation. In *Proceedings of the Fourth Experimental Chaos Conference*, 1997.
- [22] T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5:142–150, 1989.
- [23] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.
- [24] W. Dixon and F. Massey. *Introduction to Statistical Analysis*. McGraw-Hill, 1983.
- [25] A. Fang and N. Pollard. Efficient synthesis of physically valid human motion. In *Proceedings of SIGGRAPH*, 2004.

- [26] R. Feuillet. *Chorégraphie, ou L'art de décrire la danse, par caracteres, figures et signes desmonstratifs, avec lesquels on apprend facilement de soy même toutes sortes de dances ...* Chez l'auteur...et chez Micel Brunet, 1700.
- [27] A. Fod, M. Mataric, and O. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12:39–54, 2002.
- [28] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3:209–226, 1977.
- [29] N. Friedman, K. Murphy, and S. Russell. Learning the structure of dynamic probabilistic networks. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 139–147, 1998.
- [30] H. Goldstein. *Classical Mechanics*. Addison Wesley, Reading MA, 1980.
- [31] K. Grochow, S. Martin, A. Hertzmann, and Z. Popovic. Style-based inverse kinematics. *ACM Transactions on Graphics*, 23:522–531, 2004.
- [32] D. Heckerman. Bayesian networks for knowledge discovery. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*. MIT Press, 1996.
- [33] J. Hodgins and W. Wooten. Animating human athletes. In Y. Shirai and S. Hirose, editors, *Robotics Research: The Eighth International Symposium*, pages 356–367. Springer-Verlag, 1998.
- [34] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. Animating human athletics. In *Proceedings of SIGGRAPH*, 1995.
- [35] E. Hsu, K. Pulli, and J. Popovic. Style translation for human motion. In *Proceedings of SIGGRAPH*, 2005.
- [36] A. Hutchinson. *Labanotation*. The Dance Notation Bureau, NY, NY, 1970.
- [37] R. Kirk. *Experimental Design: Procedures for the Behavioral Sciences*. Brooks-Cole, 1968.
- [38] T. Komura and Y. Shinagawa. Attaching physiological effects to motion-captured data. In *Proceedings of Graphics Interface (GRIN)*, pages 27–36, 2001.
- [39] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *Proceedings of SIGGRAPH*, 2002.
- [40] L. Lee and W. Grimson. Gait analysis for recognition and classification. In *Proceedings of the IEEE Conference on Face and Gesture Recognition*, pages 155–161, 2002.
- [41] T.-Y. Li and J. A. Yorke. Period three implies chaos. *American Mathematical Monthly*, 82:985–992, 1975.
- [42] C. Liu, A. Hertzmann, and Z. Popovic. Learning physics-based motion style with nonlinear inverse optimization. In *Proceedings of SIGGRAPH*, 2005.
- [43] E. N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20:130–141, 1963.
- [44] M. Meredith and S. Maddock. Adaptive motion-capture data using weighted real-time inverse kinematics. *ACM Computers in Entertainment*, 3, 2005.
- [45] M. Neff and E. Fiume. Methods for exploring expressive stance. In *Proceedings of the ACM/Eurographics SIGGRAPH Symposium on Computer Animation*, pages 49–58, 2004.
- [46] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, 1985.

- [47] J. Pressing. Nonlinear maps as generators of musical design. *Computer Music Journal*, 12:35–46, 1988.
- [48] D. Randel, editor. *The New Harvard Dictionary of Music*. Harvard University Press, Cambridge, MA, 1986.
- [49] L. Ren, A. Patrick, A. Efros, J. Hodgins, and J. Rehg. A data-driven approach to quantifying natural human motion. In *Proceedings of SIGGRAPH*, 2005.
- [50] A. Safonova, J. Hodgins, and N. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *Proceedings of SIGGRAPH*, 2004.
- [51] H. Scheffe. *The Analysis of Variance*. John Wiley & Sons, 1959.
- [52] T. Shinbrot. Progress in the control of chaos. *Advances in Physics*, 44:73–111, 1995.
- [53] I. Stewart. *Does God Play Dice?: The Mathematics of Chaos*. Blackwell, Cambridge MA, 1989.
- [54] S. Strogatz. *Nonlinear Dynamics and Chaos*. Addison-Wesley, Reading, MA, 1994.
- [55] J. Stuart and E. Bradley. Learning the grammar of dance. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 547–555, 1998.
- [56] M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3:71–86, 1991.
- [57] R. Urtasun, P. Glardon, R. Boulic, D. Thalmann, and P. Fua. Style-based motion synthesis. *Computer Graphics*, 23:799–812, 2004.
- [58] M. Vasilescu. Human motion signatures: Analysis, synthesis, recognition. In *Proceedings of the International Conference on Pattern Recognition*, 2002.
- [59] B. Winer. *Statistical Principles in Experimental Design*. McGraw-Hill, 1962.
- [60] P. Winston. *Artificial Intelligence*. Addison Wesley, Redwood City CA, 1992. Third Edition.
- [61] G. Wolberg. *Digital Image Warping*. IEEE Society Press, 1990.
- [62] K. Yamane, J. Kuffner, and J. Hodgins. Synthesizing animations of human manipulation tasks. In *Proceedings of SIGGRAPH*, 2004.