Using Chaos to Generate Variations on Movement Sequences

Elizabeth Bradley and Joshua Stuart[1]

Department of Computer Science
University of Colorado
Boulder, Colorado, USA 80309-0430
`[lizb,stuartj]@cs.colorado.edu`

Chaos, **8**:800-807 (1998)

Abstract

We describe a method for introducing variations into predefined motion sequences using a chaotic symbol-sequence reordering technique. A progression of symbols representing the body positions in a dance piece, martial arts form, or other motion sequence is mapped onto a chaotic trajectory, establishing a symbolic dynamics that links the movement sequence and the attractor structure. A variation on the original piece is created by generating a trajectory with slightly different initial conditions, inverting the mapping, and using special corpus-based graph-theoretic interpolation schemes to smooth any abrupt transitions. Sensitive dependence guarantees that the variation is different from the original; the attractor structure and the symbolic dynamics guarantee that the two resemble one another in both aesthetic and mathematical senses.

[1]Supported by NSF NYI #CCR-9357740, ONR #N00014-96-1-0720, and a Packard Fellowship in Science and Engineering from the David and Lucile Packard Foundation.

Lead Paragraph

This paper describes a chaotic symbol-sequence reordering technique that creates variations on predefined motion sequences. This work was inspired by a similar scheme, proposed by Diana Dabby, that uses a closely related procedure to generate musical variations. We use special symbols to represent human body postures, encoding the position of each of the main joints with a *quaternion*, **which defines an axis and an angle of rotation. The quaternion symbol sequence representing the body positions in a dance piece, martial arts form, or other motion sequence is mapped onto a chaotic attractor, establishing a symbolic dynamics that links the movement progression and the attractor geometry. Using this mapping of body positions to state-space regions, we create a variation by following a new trajectory around the attractor and inverting the symbolic mapping. Any abrupt transitions in the chaotic variation are smoothed using corpus-based interpolation schemes that are faithful both to the dynamics of the movement genre and thus implicitly to the kinesiology of the human body. Sensitive dependence on initial conditions guarantees that the variation is different from the original; the attractor structure, the symbolic dynamics, and the interpolation scheme guarantee that the two resemble one another in both the aesthetic and the mathematical senses.**

Figure 1: A seven-position ballet jump sequence

1 Introduction

This paper describes a chaotic symbol-sequence reordering technique that creates variations on predefined motion sequences. This work was inspired by a similar scheme, proposed by Diana Dabby[3, 4], that uses a closely related procedure to generate *musical* variations. We map a progression of symbols representing the body positions in a dance piece, martial arts form, or other motion sequence onto a chaotic attractor, establishing a symbolic dynamics that links the movement progression and the attractor geometry. We create a variation by following a new trajectory around the attractor and inverting the symbolic mapping. Any abrupt transitions in the chaotic variation are smoothed using corpus-based interpolation schemes that are faithful both to the dynamics of the movement genre and thus implicitly to the kinesiology of the human body. Sensitive dependence on initial conditions guarantees that the variation is different from the original; the attractor structure, the symbolic dynamics, and the interpolation scheme guarantee that the two resemble one another in both the aesthetic and the mathematical senses.

To establish the mapping between an N-move motion sequence, like the ballet jump shown in figure 1, and a chaotic attractor, we first integrate a chaotic ODE system $\dot{x} = f(x),\ x(t) \in R^n$ numerically from some initial condition x_0. We then use a Voronoi diagram to partition the state-space region occupied by the ω-limit set $\phi_\omega(x_0)$ of this trajectory into N cells. Finally, we label the *itinerary* of cells traced out by $\phi_\omega(x_0)$ with special symbols that represent the sequence of body positions in the predefined movement sequence. To create a variation, we then generate a new trajectory $\phi(x')$ from an initial condition x' near the attractor and invert the mapping: at each timestep, the body position corresponding to the cell in which $\phi(x')$ falls is sent to an animation tool.

The core of this algorithm follows directly from Dabby's work on musical variation, wherein a simple metric on a projection of a chaotic trajectory with a brief initial transient is used to induce a symbol dynamics that captures the structure of the piece[2]. This mapping, which "wraps" the pitch sequence around an attractor by associating the notes in the sequence with state-space patches bounded by successive points of the

[2]The set of symbols is taken from the musical notes of the pitch sequence.

chaotic trajectory, can be used to re-create the original piece (by using the same dynamic system and initial condition, generating a trajectory, determining which patches it visits, and playing the associated notes) or — with a different initial condition — to generate a variation. Variations generated in this fashion, whether musical or choreographic, are both aesthetically pleasing and strikingly reminiscent of the original sequences. The stretching and folding along the attractor guarantee that the ordering of the pitches or movements in the variation is different from the original sequence; at the same time, the fixed geometry of the attractor ensures that a chaotic variation of Bach's Prelude in C Major or of a short Balanchine ballet sequence resembles the original piece in the classical sense of a variation on a theme.

Primarily because of the requirements of the application, many of the issues and tactics described in the paper — together with much of the mathematics — are very different from Dabby's work. The symbol set is one obvious distinction. There is a simple, well-established notational scheme for music, but body positions are much harder to represent; we use representational techniques from rigid-body mechanics to solve this problem. The mathematics of the mapping is also different in some formally important ways; we work with a full, formal symbolic dynamics *on the attractor*, derived using computational geometry techniques. Finally, while musical instruments can play arbitrary pitch sequences, subject to instrument range and performer ability, both kinesiology and aesthetics impose a variety of constraints on consecutive body postures in dance and martial arts genres. To address this problem and smooth any abrupt transitions introduced by the chaotic symbol-sequence reordering technique, we have developed a class of corpus-based interpolation schemes that use graph-theoretic methods to capture and enforce the dynamics of a given group of movement sequences.

The results produced by these mapping and interpolation algorithms have intrigued both dynamicists and dancers. The chaotic variations bear an obvious resemblance to the originals, and yet they are also clearly different[3]. Broadly speaking, the variations resemble the originals with some shuffling of coherent subsequences. When contrasted to random shuffles of the same sequences, the properties of this scheme become even more apparent: the randomized "variations" bear little temporal resemblance to the original. It is impossible to appreciate these results from a textual description; please see the animations at `http://www.cs.colorado.edu/~lizb/chaotic-dance.html`.

2 Linking the Attractor Geometry and the Movement Sequence

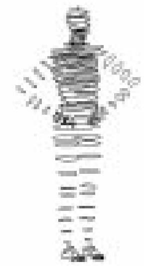
2.1 Symbolic Dynamics and Body Positions

A point in a state-space trajectory of a dynamic system can be described at different precisions, ranging from a tuple of real numbers to a symbol that identifies a large

[3]A well-known dynamicist opined "It looks like Al Gore doing the macarena."

```
{pelvis,          0, 0, 0, 1;
 ...
 right-shoulder, -0.24, -0.65, -0.33, 0.65;
 right-elbow,     0.76, 0, 0, 0.65;
 ...
 left_toes,       0, 0, 0, 1 }
```

(a)



(b)

Figure 2: Symbolic representation of the human body: (a) the descriptor/quaternion symbol, which specifies a vector and an angle of rotation around it for each of the 23 main joints in the body (b) the corresponding graphical representation used by the Life Forms animation tool.

state-space region. Though the coarse-grained nature of the latter abstracts away much detailed information about the dynamics, it preserves many of its invariant properties; see, e.g., Hao[8] for details. Establishing such a *symbolic dynamics*[11] presents two problems: the partition and the ordering. This paper offers novel and unusual solutions to both: we use computational geometry techniques on points of a trajectory to obtain a good partition, and we use the natural progression of body positions in a movement sequence to induce the symbols and their ordering.

The symbol set used in our algorithms represents the position of each of the 23 primary joints in the human body with a *quaternion* — a standard representation in rigid-body dynamics, dating back to Hamilton[7]. A quaternion $Q(r, \vec{u})$ consists of a three-space vector $\vec{u}$ and a scalar r that specifies the angle of rotation around that vector. Thus, a body position symbol $\mathcal{S}$ is quite complicated: 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. See figure 2 for an example. This complexity is simply a reflection of the representational task involved; Labanotation[9], the graphically intricate system used by professional dance notators, is even more baroque, and attaining proficiency in its use requires years of practice. Note that this quaternion-based symbol set can be easily adapted to other body topologies, such as those of insects. The code has been designed to support arbitrary topologies, so this variation procedure is by no means limited to human movement sequences.

2.2 Tiling the Attractor

Creating a partition for the purposes of chaotically shuffling a movement sequence requires tiling the state-space region occupied by the attractor with N nonoverlapping cells, where N is the number of postures $s_i \in \mathcal{S}$ in the movement sequence $s_1, s_2, \ldots s_N$. To accomplish this, we first integrate a chaotic ODE system $\dot{x} = f(x)$, $x(t) \in R^n$ with 4^{th}-order Runge-Kutta[14] from some initial condition x_0 and let the transient die out.

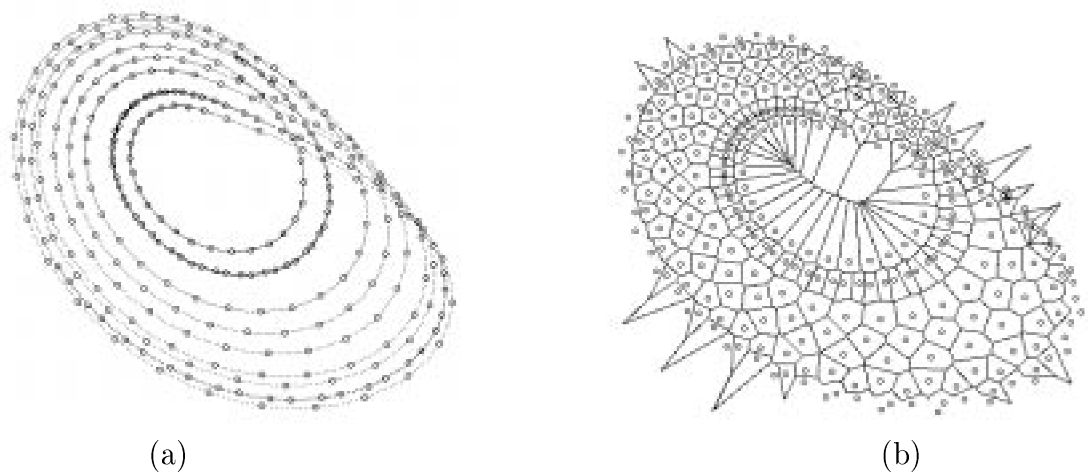


Figure 3: A Rössler trajectory for a 300-move movement sequence and the associated Voronoi-diagram partition of the attractor. The perpendicular bisectors of the circled trajectory points in (a) yield the Voronoi diagram in (b). Line segments that extend beyond the bounding box of part (b) have been omitted from this plot.

The requirements of the mapping limit the number of cells to N, but the partition requires that the collection of cells cover the attractor, and spurious numerical effects preclude simply increasing the time step until a fixed-length (N-point) trajectory covers a given attractor. We address this by fixing the timestep and trajectory length — choosing values for these parameters that assure that the transient has died out, the attractor is covered, and the dynamics include no spurious numerical effects — and using a "skip" parameter, m, to control the spacing of the trajectory points that are used to construct the cells. In figure 3, for instance, $m = 10$, so every eleventh trajectory point (shown circled) generates a cell. The specific algorithm that we use to actually construct the cells, the Voronoi diagram[13], is drawn from the field of computational geometry; it involves constructing and intersecting the perpendicular bisectors of every adjacent pair of points, as shown in figure 3. Note that a Voronoi diagram is essentially the dual of a Delaunay triangulation; see Preparata and Shamos[13] for more details on this branch of computational geometry. The actual implementation uses K-D trees[6], rather than the usual Voronoi diagram construction algorithm, to reduce the computational complexity in the step of the algorithm that finds all the nearest neighbors from $O(N^2)$ to $O(N \log N)$.

2.3 Establishing and Using the Mapping

Given an N-position movement sequence, expressed in terms of the quaternion-based symbol set described in section 2.1, and an N-cell Voronoi tiling of a chaotic attractor — like the $N = 300$ example shown in figure 3 — establishing a mapping that links the attractor geometry and the movement sequence simply amounts to equating indices: the first entry c_1 in the *itinerary* $c_1, c_2, \ldots c_N$ of cells traversed by the trajectory $\phi_\omega(x_0)$ is

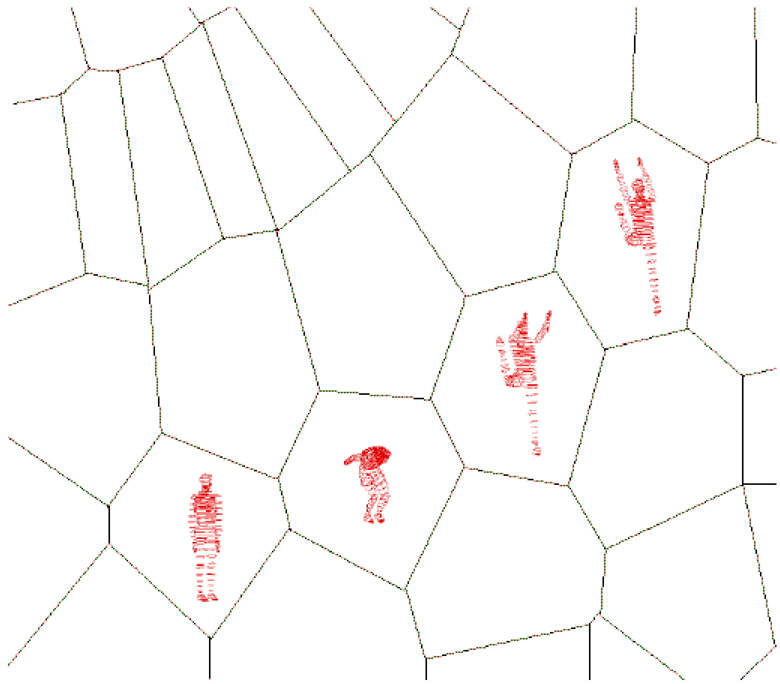


Figure 4: Part of the chaotic mapping that links a longer version of the jump sequence of figure 1 and the Rössler attractor geometry of figure 3. Only a few body positions are shown here; in the full mapping, each Voronoi cell is labeled with a posture.

labeled with the symbol s_1 that describes the first position in the movement sequence, and so on[4], as depicted schematically in figure 4. Using this mapping of body postures s_i to state-space patches c_i to create a variation is equally simple, but somewhat more computationally expensive: we generate another trajectory $\phi(x')$ from an initial condition x' near the attractor, use the K-D tree to determine the Voronoi cell c_j in which its first point falls, output the associated body-position symbol s_j to an animation tool, skip m points, and repeat to the desired variation length. This procedure is quite rapid: for a 1000-position movement sequence, the entire chaotic symbol-sequence reordering procedure requires 18 seconds on an HP9000/735 workstation running HP-UX v10.20; without the K-D tree, it takes 30.2 seconds. The K-D tree advantage grows ($N \log N$ vs. N^2) with the sequence length: for a 9000-move sequence, the times are 156.2 and 2324.2 seconds, respectively.

[4]This representation — a symbolic dynamics on an attractor, induced by a movement sequence — preserves conjugacy to the R^n dynamics of $\phi_\omega(x_0)$. This is interesting in that it implies that the animations on the website are formally equivalent, in a precise mathematical sense, to the "real" dynamics on the Rössler and Lorenz attractors[8, 10].

3 Interpolation

The symbol-sequence reordering scheme described in the previous section introduces abrupt transitions in the chaotic variation — places where consecutive body positions would require physically impossible or stylistically illegal moves. The interpolation scheme described in this section inserts new body postures into these gaps in order to smooth the progression. A simple and obvious way to do this would be to use splines or some other purely mathematical interpolation technique on the quaternion data. This does not, however, address the problem of stylistic or kinesiological illegality. A spline-based interpolation may not, for instance, adhere to the requirement that the elbow only bends 180 degrees. Using kinematics, together with state variable bounds that express physiological constraints, would be a better approach, but $F = ma$ cannot, for example, enforce the requirement that ballet motion is linear[5]. To solve these problems, we use a corpus of human movement (e.g., one composed of ten Balanchine ballets, if one is working with dances of that particular genre, or one composed of four karate kata if this chaotic variation method is to be applied to karate sequences) to select a sequence of postures that would naturally occur between the two positions that frame the abrupt transition. The composition of the corpus will, of course, affect the nature of the interpolation; smoothing abrupt transitions in ballet pieces using an interpolation scheme that is mathematically rooted in a karate corpus will negate the very aesthetic resemblance that the core of the algorithm is designed to preserve. On the other hand, this might be an interesting source of innovation, whereby one could mathematically mix two or more styles.

To this end, we build a labeled, directed graph $G(V, E)$ that captures the structure of the movement sequences in the corpus. Each body position in the corpus is represented by one vertex v_i and each transition between successive postures is represented by an edge e_{ij} between the corresponding vertices. Figure 5 shows an example of such a graph. In this formulation, an illegal transition — defined as one that is not present in the corpus — is a pair of vertices v_l and v_n that are not linked by a single edge e_{ln}, such as the postures labeled **a** and **c** in figure 5. When such a transition is encountered in the chaotic variation, we use $G(V, E)$ to compute an interpolation subsequence that starts with v_l, ends with v_n, and is consistent with the corpus. Specifically, we use a forward-backward modified Dijkstra's algorithm to find the shortest path in G between v_l and v_n, and then insert the body positions corresponding to the vertices traversed by that path into the gap in the original sequence. In figure 5, for example, there are two two-edge paths that link postures **a** and **c**: $\{\mathbf{a} \rightarrow \mathbf{b} \rightarrow \mathbf{c}\}$ and $\{\mathbf{a} \rightarrow \mathbf{e} \rightarrow \mathbf{c}\}$. The abrupt transition $\{\mathbf{a} \rightarrow \mathbf{c}\}$ could be patched with either of these two subsequences. Our software provides a viewing option that highlights these inserted sequences in a different color — and editing tools for cutting and pasting — so choreographers and animators can identify the insertions and discard, approve, and/or change the results[6].

[5]In ballet, body parts tend to describe piecewise-linear paths through space, emphasizing the positions at the junctions of those linear segments; in modern dance, on the other hand, arcing motions are much more common.

[6]Given that animations and movement sequences are almost always linked with soundtracks, it might

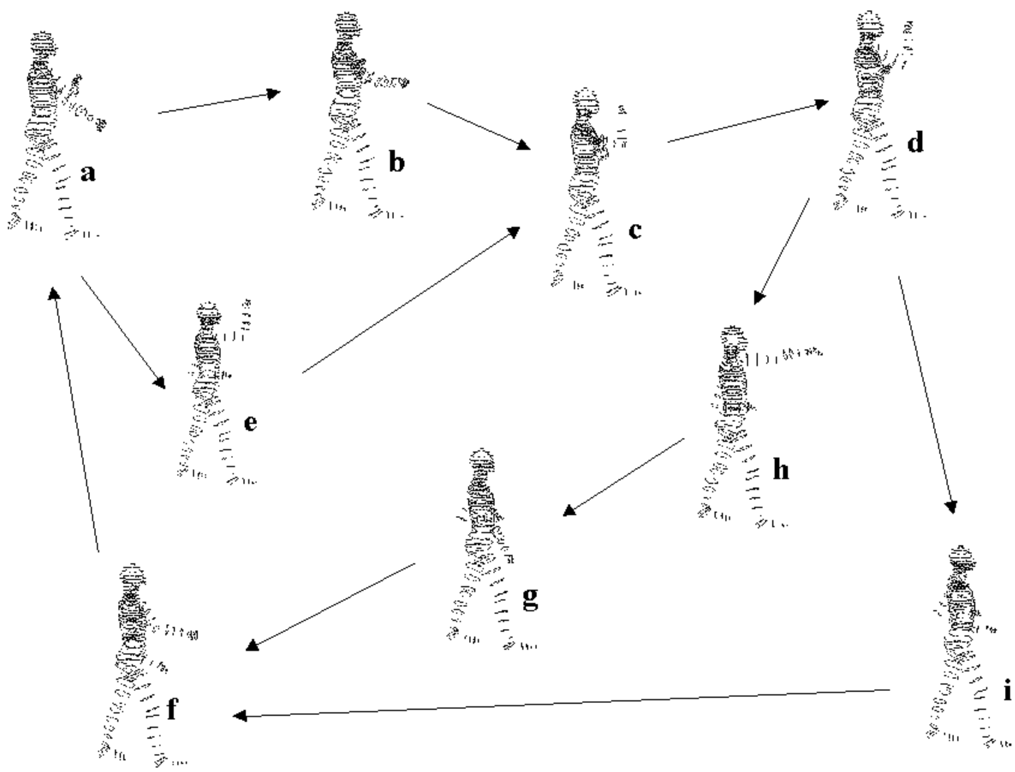


Figure 5: A labeled, directed graph representing a small corpus of human movement. Vertices represent postures present in the corpus and edges depict observed movement sequences between those postures.

Dijkstra's algorithm[5] finds the shortest path from a single source vertex to all other vertices in a graph; if more than one "shortest" path exists, as in figure 5, it returns the first one it encounters. For the forward version, v_l is used as the source vertex. For the backward version, v_n is used as the source and the orientations of the edges in G are reversed. The forward and backward algorithms are invoked simultaneously; each one progressively deepens its search until a common vertex v_m is encountered. The paths from v_l to v_m and from v_m to v_n are then merged to give the desired shortest path from v_l to v_n. The worst case total running time of this algorithm is $O((V+E)\log_2 V)$[2].

To better model and enforce the nuances of a particular movement style, we are improving upon this scheme in two ways. The first involves finer-grained physical representation. Currently, the atomic representational unit is a full body position; the next version will perform *joint-wise* interpolation instead — e.g., bridging a gap by moving the shoulder from its quaternion position in v_l to its quaternion position in v_n according to the rules for *shoulder* movement that are implicit in the corpus, repeating for the elbow, and so on, rather than searching for and patching in *full body positions*. The second improvement involves probabilistic analysis of the transitions in the corpus. $G(V, E)$ is currently a simple labeled, directed graph, where transition legality is represented by the presence or absence of an edge; to this, we are adding edge weights that represent a measure of the probability of each inter-move transition. One logical choice for these weights is the negative log-likelihood:

$$w_{ij} = -\log(f_i) + \log(f_j) - \log(f_{ij}),$$

where f_i and f_j are the frequencies of postures i and j and f_{ij} is the frequency with which posture j follows posture i. Small values for w_{ij} correspond to transitions that are more likely to occur. With this addition, the interpolation scheme can find more-natural subsequences with which to smooth abrupt transitions. For instance, a five-edge path may have a much higher probability than a two-edge path if the latter is only observed rarely in the corpus, and adding edge weights to G allows the interpolation scheme to enforce that constraint. There is, however, an important disadvantage to a probabilistic scheme like this: choosing only the most-likely moves may engender cliché.

Fine-grained, jointwise interpolation with log-likelihood edge weights, as described in the previous paragraph, is theoretically a good solution for the problem at hand, but its computational complexity is prohibitive. For example, if each joint can be in one of only ten possible orientations, then G could contain $O(10^{88})$ vertices. One way to manage this complexity is to use a hierarchical data structure that exploits the structure and physics of the human body — the notion, for example, that the position of the wrist strongly affects the position of the fingers but has little effect on the toes. The physical structure of the human body is depicted graphically in part (a) of figure 6. We use a tree to represent this structure in the form of dependency relationships between joints

make sense to store the sound associated with each position s_i and then reproduce it whenever that position appears in the variation. On the other hand, the sound and movement genres may operate under wholly different sets of constraints, in which case this would create nonsense sound — e.g., reordering the syllables of a word.

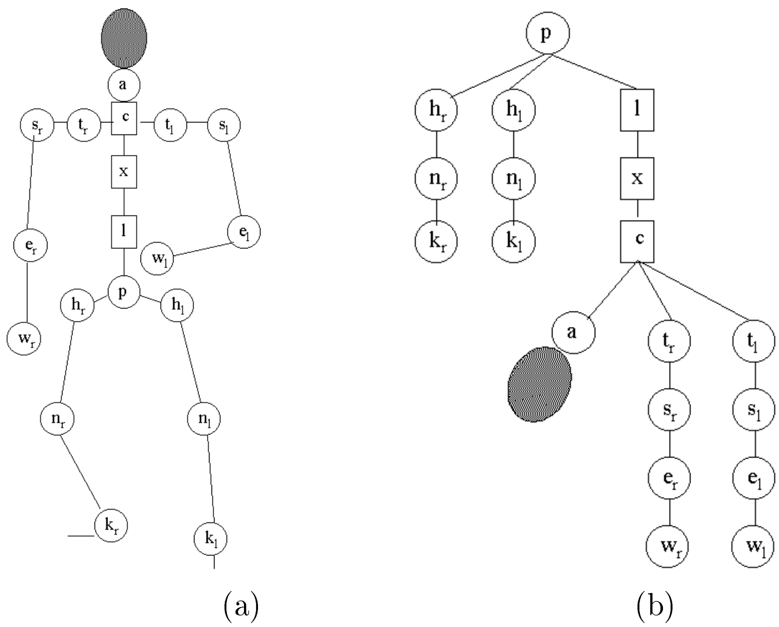
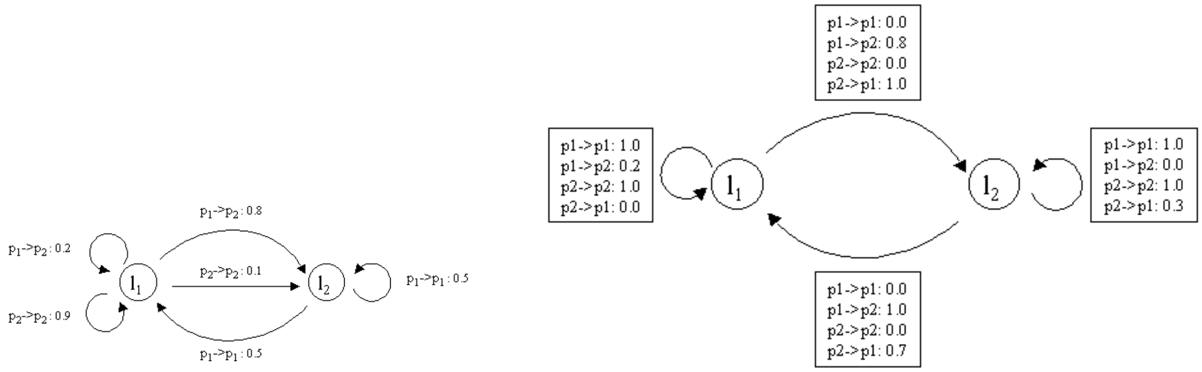


Figure 6: A hierarchical representation of the human body that helps mitigate the complexity of jointwise interpolation. The shaded ellipse is the head; toes and fingers have been omitted for clarity. Part (a) depicts the body and its major joints. Bilateral joints are identified with subscripts according to the side of the body on which they fall (e.g., right and left elbows e_r and e_l). Part (b) shows the dependencies induced by gravity and topology: for instance, the position of the pelvis influences the positions of both hips h_r and h_l and the lumbar spine l, but the right and left ankles k_r and k_l do not directly influence one another.



(a) (b)

Figure 7: Capturing typical joint movement patterns: each joint is associated with a graph that contains a vertex for each of its observed states, together with a set of edges that define how that joint reacts to movements of its parent. The graph in part (a) reflects the patterns in how the lumbar spine joint l reacts to movements of its parent joint p, the pelvis. To manage graph complexity, we aggregate topologically identical edges and attach a progression/probability table to each, as shown in part (b).

— a type of *influence diagram*[12] — as shown in part (b) of the figure. The pelvis is the root of this tree; three branches lead from this root to nodes corresponding to the right thigh/hip joint, the left thigh/hip joint, and a joint representing the lower spine[7]. Each hip joint is the parent node to a knee (n), and so on.

Associated with each node of this tree is a graph that contains a vertex for each observed state of the corresponding joint, together with a set of edges that define how that joint reacts to movements of its parent joint. Figure 7(a), for instance, shows the graph associated with the lumbar-spine node l in a tree built from a corpus where both that joint and its parent (the pelvis) can take on two orientations: $\{l_1,\ l_2\}$ and $\{p_1,\ p_2\}$ respectively. If the lumbar spine is in position l_1 and the pelvis moves from p_1 to p_2, then the lumbar spine will either stay in position l_1 (with probability 0.2) or move to position l_2 (with probability 0.8). This figure are highly simplified; graphs constructed from real movement sequences have many more edges. Figure 8, for example, shows a directed graph representing how the shoulders move in a corpus of 38 short ballet sequences. This corpus included 919 positions, and the patterns in their progressions, as represented by the topology of the graph, are obviously quite complex. To handle this complexity, our representation aggregates topologically identical edges and attaches a progression/probability table to each, as shown in part (b) of figure 7. Note that the probabilities in each edge table may sum to values greater than 1.0, but the layered structure of the graph requires that, given a transition pair from the parent $p_i \to p_j$ (written `pi->pj` in figure 7), the sum of the weights of all $p_i \to p_j$-labeled edges leaving

[7]The sacrum and the five lumbar vertebrae are lumped together. This compromise sacrifices back suppleness for lowered complexity.

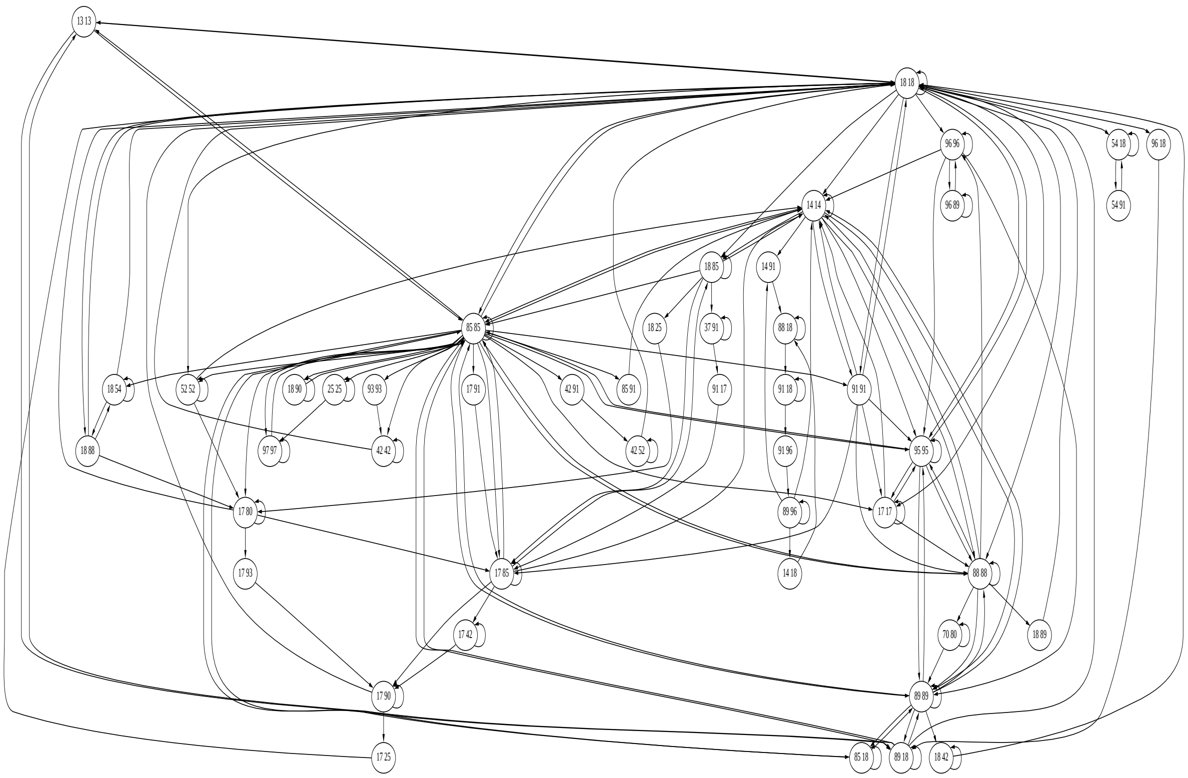


Figure 8: In real movement sequences, movement graphs have many more edges than the simplified example shown in the previous figure. The graph above, which represents the movements of the shoulders, was constructed from a small corpus of 38 short ballet sequences. The numbers in each state identify the discretized position of the joint. The parent-joint probability tables — the boxed `pi->pj` information in the previous figure — have been omitted in the interests of clarity.

any vertex must be 1.0.

Discretization of joint states was unexpectedly difficult. Simply discretizing the quaternion values — that is, classifying all positions between, say,

```
right-shoulder, 1, 1, 0, 1
```

and

```
right-shoulder, 1, 1, 0.2, 1
```

as an equivalence class and representing them in the algorithms as a single posture — produced visibly awkward animations. The human visual perception system appears to be very sensitive to small variations in quaternion coefficients: any change in a single coefficient seems to violate the “motif” of the motion. The same problem arose when we attempted a physically more-realistic discretization by transforming quaternion data to Euler angles and then discretizing θ, ϕ, and ψ instead. The solution on which we eventually settled uses a discretization library that was created by hand by an expert dancer.

Many of the techniques in this section, as well as others on which we are currently working, were inspired by solutions to similar problems that arise in molecular biology (e.g., DNA sequence analysis) and computational linguistics (e.g., learning a grammar from a corpus and then using it to construct meaningful sentences). For example, one can view the hierarchical graph structure in figures 7 and 8 as a set of first-order Markov chains, in which a single chain represents the orientation of each joint in the body. Each Markov chain contains a different set of state transitions and transition probabilities for each transition pair in the joint's parent node.

4 Results

Figure 9 shows the simple ballet jump sequence of figure 1, a chaotic variation of that jump generated with the Lorenz system, and a smoothed version of that variation. The sequence shown in the middle row of figure 9 was derived from the original using the chaotic symbol-sequence reordering scheme described in section 2. An abrupt transition is visible between the third and fourth moves of this variation; the corpus-based graph-theoretic interpolation scheme described in section 3 inserted two new moves to produce the smoothed sequence shown at the bottom of the figure. Note that the inserted moves define a very natural way to move between the two body positions that frame the abrupt transition.

While it is clear from the figure that the jump positions are indeed shuffled and that the interpolated version is indeed smoother, it is impossible to appreciate these results from a static portrayal of such a short sequence; please see the web site listed at the end of the introduction for a variety of animated variations — including the jump shown in figures 1 and 9, a popular dance progression (the macarena), a martial arts “form” drawn from the discipline of kenpo karate, and a medley of all three of these

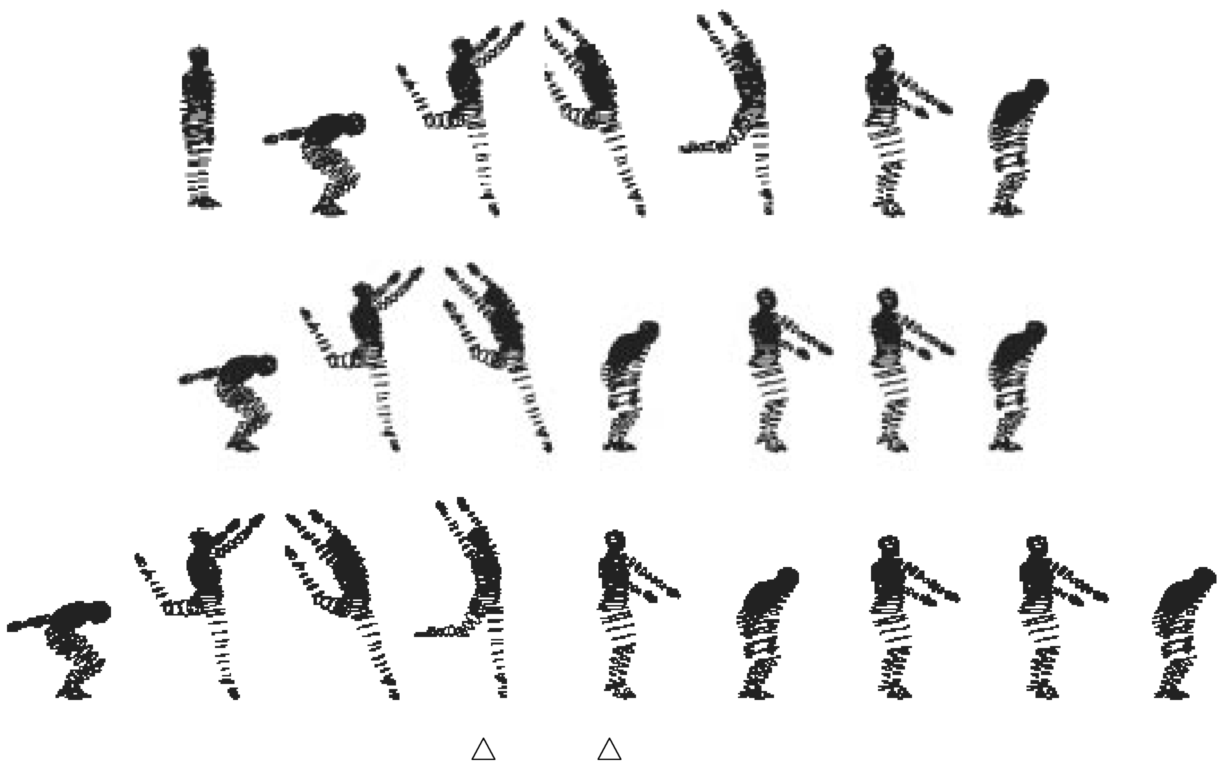


Figure 9: A ballet jump: the original sequence (above), a chaotic variation on that original (middle), and an interpolated version of that variation (below). The moves identified by arrows in the lower sequence were inserted by the interpolation scheme to smooth an abrupt transition between the third and fourth moves in the chaotic variation above it.

movement sequences. Variations were constructed on each of these four pieces using two different chaotic systems — Lorenz and Rössler — in order to explore how the attractor geometry affects the variation. Loosely speaking, variations resemble the originals with some shuffling of coherent subsequences; this is most obvious in the medley, where the variation clearly shifts back and forth between genres. Where there is an obvious genre, such as the karate sequence, martial arts experts report that the variations fit that genre. In fact, the point of using this sequence was the distinct, well-defined structure of individual martial arts genres, and the (achieved) goal was to determine whether variations generated on kenpo karate sequences still looked like kenpo — and not like shokotan karate or tae kwon do[8]. We also present *randomly* shuffled versions of each of the four pieces in order to demonstrate, by contrast, how much structure is retained by the chaotic variation scheme. Perhaps the most telling comparison is between the chaotic and randomized versions of the medley; segments of the individual dances are clearly visible in the former and all but absent in the latter. (What structure *is* present comes from the Life Forms animation tool, which performs a spline-based interpolation between posture snapshots in order to fill in frames in the movie.)

These results set off a variety of interesting questions. For instance, a shorter movement sequence implies larger cells and hence a "coarser" symbolic dynamics; this has interesting effects on how smoothly the cell itinerary of $\phi(x')$ moves and shifts along the original attractor, with corresponding implications for the animation and its resemblance to the original piece. The attractor geometry plays a mathematically and visually obvious role in the character of the variation; note the differences between the Lorenz and Rössler pieces on the web site. It appears that the latter contains longer coherent original subsequences than the former, which is consistent with the Möbius-band nature of the Rössler attractor, in comparison to the bilaterally symmetric two-lobed Lorenz geometry. We are in the process of analyzing the two pieces statistically in order to determine whether these apparent patterns are real or illusory. Finally, the implementation allows for arbitrary body topologies, so this scheme is by no means limited to *human* motion sequences — though one would, of course, have to adapt the quaternion-based symbol to the topology of the limbs and joints that are involved.

Besides the animations and the associated explanation and analysis, the web site also contains a simple animation package and the symbol-sequence reordering code itself. We encourage the readers (and their students) to create new animations and/or try different ODE systems, initial conditions, and so on. New animations are particularly useful and extremely welcome; the dance world has not yet embraced the notion of computer animation, so the current critical limitation in this project is the inadequacy of the existing corpus — on which our interpolation scheme depends[9].

[8]These genres use many of the same postures as kenpo, but in very different sequences.

[9]The chaotic symbol-sequence reordering code does not yet handle center-of-mass interpolation smoothly, so movement sequences where the body moves from place to place will appear choppy. We are currently working on this, but the issues involved — kinesiology, in particular — make it quite difficult to solve.

5 Conclusion

Evaluation of these results is necessarily somewhat subjective. We have shown these animations to hundreds of people, including dozens of dancers and martial artists. The consensus is that the variations not only resemble the original pieces, but also are in some sense pleasing to the eye. They are both different from the originals and faithful to the dynamics of the genre: there are no jarring transitions or out-of-character moves. This is a non-trivial accomplishment. A previous attempt to use mathematics to generate choreographic variations — a subsequence randomization scheme introduced by the now well-known choreographer Merce Cunningham in the 1960s — met with a strongly negative reception in the dance world[10].

From a scientific viewpoint, this scheme is interesting for several reasons. It involves a formal (albeit unusual) application of symbolic dynamics, the properties of chaotic attractors, and rigid-body mechanics: the partition for the symbolic dynamics is generated automatically using computational geometry techniques and the natural order of the movement sequence and the symbol set relies on a representational device invented by Hamilton himself. By applying methods from graph theory, statistics, and computational linguistics to a corpus of movement sequences from a particular genre, the interpolation scheme described here smooths awkward transitions in a physically *and stylistically* coherent fashion. Last, but certainly not least, showing these results in a classroom is an enormously effective way to motivate students to learn the mathematics of rigid-body dynamics, chaos, and context-dependent grammars.

Acknowledgements: Stephen Schroeder discretized each joint's position by hand, and Eric Schell and Meenakshy Chakravorty helped set up the website and write the associated animation code. The authors would like to thank David Capps and Nadia Rojasadame of the University of Colorado Department of Theater and Dance and Dan Jurafsky of the University of Colorado Department of Linguistics for helpful conversations, Margo Seltzer of Harvard University for suggesting and demonstrating the karate sequences, and Diana Dabby of M.I.T. for numerous fruitful and thought-provoking suggestions.

References

[1] D. Capps. University of Colorado, Department of Theater and Dance, personal communication, 1998.

[2] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, 1990. pp 527-531.

[3] D. Dabby. Musical variations from a chaotic mapping. *Chaos*, 6:95–107, 1996.

[10] Since that time, *aleatory* choreography — wherein randomization schemes are used to shuffle sequences — "has by now become one of the important currencies of dance composition approaches."[1]

[4] D. Dabby. A chaotic mapping for musical and image variation. In *Proceedings of the Fourth Experimental Chaos Conference*, 1997.

[5] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[6] 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.

[7] H. Goldstein. *Classical Mechanics.* Addison Wesley, Reading MA, 1980.

[8] B.-L. Hao. Symbolic dynamics and characterization of complexity. *Physica D*, 51:161–176, 1991.

[9] A. Hutchinson. *Labanotation.* The Dance Notation Bureau, NY, NY, 1970.

[10] D. Lind and B. Marcus. *An Introduction to Symbolic Dynamics and Coding.* Cambridge University Press, 1995.

[11] M. Morse and G. A. Hedlund. Symbolic dynamics. *Am. J. Math.*, 60:815, 1938.

[12] R. M. Oliver and J. Q. Smith, editors. *Influence Diagrams, Belief Nets and Decision Analysis.* Wiley, 1990.

[13] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, 1985.

[14] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes: The Art of Scientific Computing.* Cambridge University Press, Cambridge U.K., 1988.