Learning the Grammar of Dance

Joshua M. Stuart
Elizabeth Bradley
Department of Computer Science
University of Colorado
Boulder CO 80309-0430 USA

Abstract

A common task required of a dancer or athlete is to move from one prescribed body posture to another in a manner that is consistent with a specific style. One can automate this task, for the purpose of computer animations, using simple machine-learning and search techniques. In particular, we find kinesiologically and stylistically consistent interpolation sequences between pairs of body postures using graph-theoretic methods to learn the "grammar" of joint movements in a given corpus and then applying memory-bounded A* search to the resulting transition graphs — using an influence diagram that captures the topology of the human body in order to reduce the search space.

1 INTRODUCTION

A common task required of a dancer or athlete is to move from one prescribed body posture to another in a manner that is consistent with a specific style. If these postures are "far apart," as measured by some metric that takes into account both the kinesiology of the body and the style of the movement genre, this can be nontrivial. For the purposes of computer-generated animation, there are a variety of ways to generate movement sequences that accomplish this kind of task. One can, for instance, use mathematical interpolation techniques like splines to move individual body parts from one position to another, but these kinds of methods do not address the problem of kinesiological illegality (e.g., that the knee only bends 180 degrees, or that arms cannot pass through ribcages). Many animation packages, such as Life Forms (`http://fas.sfu.ca/lifeforms.html`), use an augmented spline approach that relies on a table of kinematic constraints to avoid illegal movements, but this type of approach is somewhat *ad hoc*. A more-general way is to use the physics of the body: derive the associated differential equations — a torque balance for each joint, say — and solve the equivalent boundary-value problem. Approaches like this[10] are extremely interesting and highly promising, but also very difficult; deducing the control equations that humans use to recover

their balance after a jump, for example, is a Ph.D. thesis-level problem[13]. *Stylistically* faithful interpolations would be even harder to implement; neither splines nor $F = ma$ can easily capture or enforce, for instance, the requirement that classical ballet emphasizes position over motion[1], and developing a mathematics- or physics-based approach that does so would be all but impossible. In this paper, we propose an alternative solution to this problem: a class of corpus-based interpolation schemes that generate a kinesiologically *and stylistically* consistent movement sequence between two specified body positions by learning and then enforcing the dynamics of a particular movement genre.

The primary motivation for the development of these methods was our work on a mathematical technique[1, 2] that automatically creates variations on predefined motion sequences — an idea that was inspired by a similar scheme[5, 6] that uses a related procedure to generate *musical* variations. We use the mathematics of nonlinear dynamics to shuffle a predefined movement sequence by "wrapping" a progression of special symbols representing the body positions in a dance piece, martial arts form, or other motion sequence around a chaotic attractor. This establishes a symbolic dynamics that links the movement progression and the attractor geometry, as shown in figure 1. By definition, trajectories from different starting points[2] travel along the same attractor but *in a different order*. This property lets us use the mapping depicted in figure 1(d) to create a variation: we simply follow a *new* trajectory around the attractor and invert the symbolic mapping, "playing" the body position for each cell the trajectory enters. Variations generated in this manner, whether musical or choreographic, are both aesthetically pleasing and strikingly reminiscent of the original sequences. The stretching and folding of the chaotic dynamics 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 are related to the original piece in a sense reminiscent of the classic "variation on a theme." Broadly speaking, the chaotic variations resemble the originals with some shuffling of coherent subsequences. This is the primary source of the stylistic originality of the chaotic variation scheme — in fact, this type of subsequence shuffling is a well-established creative mechanism in modern choreography. One problem with any choreographic technique, automated or not, that involves subsequence reordering, however, is that the transitions at the subsequence boundaries can be quite jarring. Figure 2, for example, shows a short section of a chaotically generated variation on a short ballet adagio. Note the abrupt transition between the fifth and sixth moves of the variation.

The interpolation algorithms that are the topic of this paper can smooth these kinds of transitions in a manner that is both kinesiologically and stylistically consistent. These graph-theoretic methods "learn" the grammar of joint

---

[1]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, the motion *between* the endpoints is the important feature.

[2]within the basin of attraction, of course
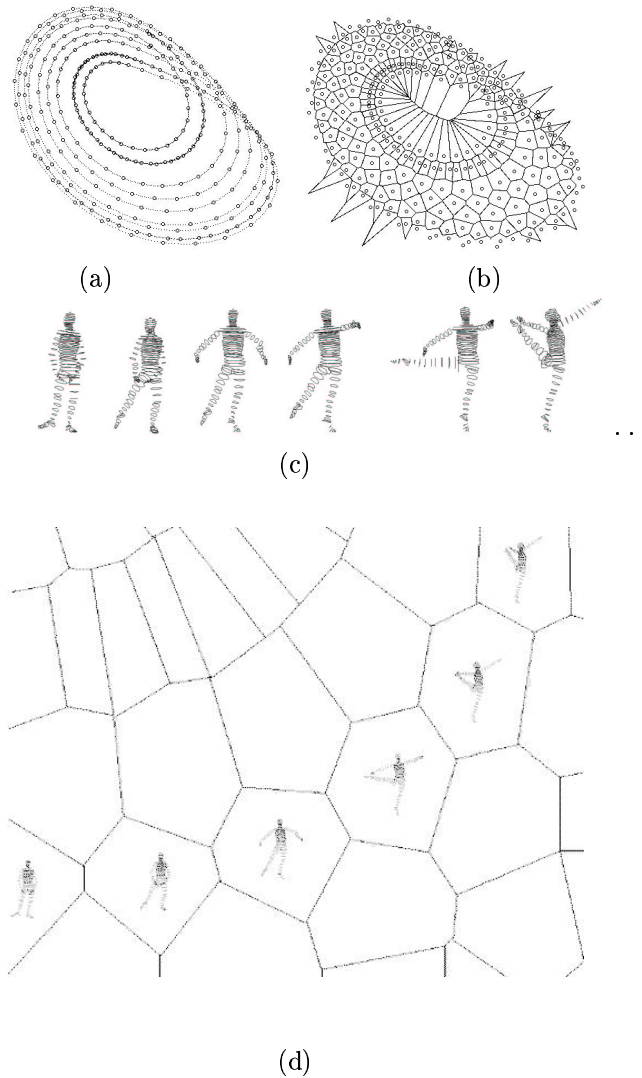
(a)   (b)

(c)

(d)

Figure 1: A chaotic mapping that links a short ballet adagio and the chaotic Rössler attractor. A Voronoi diagram is used to divide the region covered by the trajectory shown in part (a) into cells, yielding the tiling shown in part (b). The order in which the original trajectory traverses those cells defines the temporal order of the *cell itinerary* that corresponds to that trajectory. Successive body positions from the predefined movement sequence (c) are mapped to successive cells in that itinerary, linking the structure of the movement sequence and the attractor geometry. A small section of the overall mapping is shown in part (d).

Figure 2: Part of a variation on a short ballet sequence, generated using the chaotic shuffling procedure diagrammed in the previous figure. Note the abrupt transition between the fifth and sixth frames. The interpolation schemes described in this paper can be used to smooth such transitions in a kinesiologically and stylistically consistent fashion.

movements in a given corpus and then apply memory-bounded A* search — using an influence diagram that models the relationships of the joints in the human body in order to reduce the otherwise-intractable search space — to find an appropriate interpolation sequence between two given body positions. The search is complicated by the fact that joint positions cannot be interpolated in isolation: the movement patterns of the ankle, for instance, are strongly influenced by whether or not the foot is on the ground — information that is implicit in the positions of the pelvis, knees, etc. This requires that the expansion of nodes in the search be context dependent in a somewhat unusual way. The resulting interpolation procedures, which were developed and evaluated in close collaboration with several expert dancers, are quite effective at capturing and enforcing the dynamics of a given group of movement sequences.

2 CORPUS-BASED INTERPOLATION ALGORITHMS FOR MOVEMENT SEQUENCES

The interpolation schemes described in this section use corpora of human movement — a corpus composed of ten Balanchine ballets, for instance, if one is working with dances of that particular genre[3] — to select a movement sequence that would naturally occur between a given pair of body postures. The basic algorithms involved are fairly straightforward, but the application requires some unusual tactics and variations. We first examine the corpus, capturing typical progressions of joint positions in a set of transition graphs. Then, given a pair of body postures, we use a variant of the A* algorithm to search these graphs for interpolation subsequences. A typical interpolation sequence might, for in-

---

[3]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 this approach strives to preserve. On the other hand, this might be an interesting source of innovation, whereby one could mathematically mix two or more styles.

stance, first move the shoulder from its position in the fifth frame of figure 2 to its position in the sixth frame according to the rules for *shoulder* movement that are implicit in the corpus, then repeat for the elbow, and so on.

Our original approach[1] was much more coarse-grained; the atomic representational unit was a full body position and the patterns in the corpus were represented in a single graph that had one vertex for each observed posture. This approach was both impractical and unsatisfying. Firstly, it did not scale well with corpus size because the number of unique body positions is so large. Secondly, it could only populate interpolation sequences with verbatim copies of full-body positions that appeared in the corpus. The methods described in this paper, on the other hand, construct the body positions in the interpolation sequence in a joint-wise manner and on the fly. This scheme not only avoids the storage problems of the previous approach, but also allows innovation: it can generate sequences that contain body positions that do not appear in the corpus.

2.1 BODY POSTURE REPRESENTATION

We represent a human body posture by specifying the position of each of the 23 main joints with a *quaternion*, a standard representation in rigid-body mechanics that dates back to Hamilton[9]. 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 $\vec{u}$. Thus, a body-position symbol is quite complicated: 23 descriptors (`pelvis`, `right-wrist`, etc.), 92 numbers (four for each joint), and a variety of information about the position and orientation of the center of mass.

Joint orientations are, in reality, continuous variables, but computational complexity requires that they be discretized in our algorithms. Specifically, each joint λ can take on a finite number M^λ of allowed orientations[4]. 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: $s_\lambda = r$ such that $\|b_\lambda - r\| \leq \|b_\lambda - q\|$ for all $q, r \in Q^\lambda$ where $\|x - y\|$ is the Euclidean distance[5] between the quaternions x and y. We can find r in $\log(M^\lambda)$ time using K-D trees[8] to represent the Q^λ sets. The procedure described in this paragraph is analogous to "snapping" objects to a grid in computer drawing applications.

Deriving a successful discretization of joint states was unexpectedly difficult. Simply discretizing the quaternion variable values — that is, classifying all positions between, say, (`right-wrist`, `1`, `1`, `0`, `1`) and (`right-wrist`, `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

---

[4]In practice, $M^\lambda < 400$.

[5]One of the main advantages of quaternions is that they can be treated as 4-vectors in the standard norm and transformation operations.

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.

2.2   REPRESENTATION OF A MOVEMENT CORPUS

2.2.1   Joint Transition Graphs

A transition graph is a weighted-directed graph that captures the transition probabilities in a symbol sequence. In general, each vertex v in such a graph represents a symbol and each weighted edge (v, u) reflects the probability that the symbol associated with vertex u follows the symbol associated with vertex v. For the purposes of analyzing a human movement corpus, we build one transition graph for each joint, using the corpus to identify orientations that the joint assumes and to estimate the corresponding transition probabilities. Vertices in this kind of graph represent particular discretized joint orientations, and edges correspond to the movement of the joint from one orientation to another.

The transition graph construction procedure is fairly straightforward. We first transform every body position in the corpus to a discretized position, as described in the previous section, 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, each of which corresponds to exactly one quaternion in Q^λ. For convenience, we will refer to vertices in G^λ by the corresponding quaternions in Q^λ. 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 assign a weight to this edge that models the "unlikeliness" with which such a transition occurs in the corpus. This measure of unlikeliness is related to $P(q \rightarrow r)$, the probability that joint λ moves from the quaternion $q \in Q^\lambda$ to the quaternion $r \in Q^\lambda$, per the following expression for the weight of edge $(q, r) \in G^\lambda$:

$$w_{q,r}^\lambda \;=\; -log(P(q \rightarrow r)) \;=\; -log(P(r|q))$$

$$\approx log(C(q)) \;-\; log(C(q,r))$$

where $C(q)$ is the number of times joint λ assumed an orientation approximated by q and $C(q,r)$ is the number of times that the ordered pair (q,r) occurred. Larger weights correspond to transitions that are less likely to occur[6].

Figure 3 shows a transition graph for the hips that was constructed in this fashion from a corpus of 38 short ballet sequences totaling 1720 positions. In the interests of clarity, edge weights and isolated vertices have been omitted

---

[6]Given this formulation, saying that two vertices are disconnected is synonymous with saying that two are connected by an edge with infinite weight.
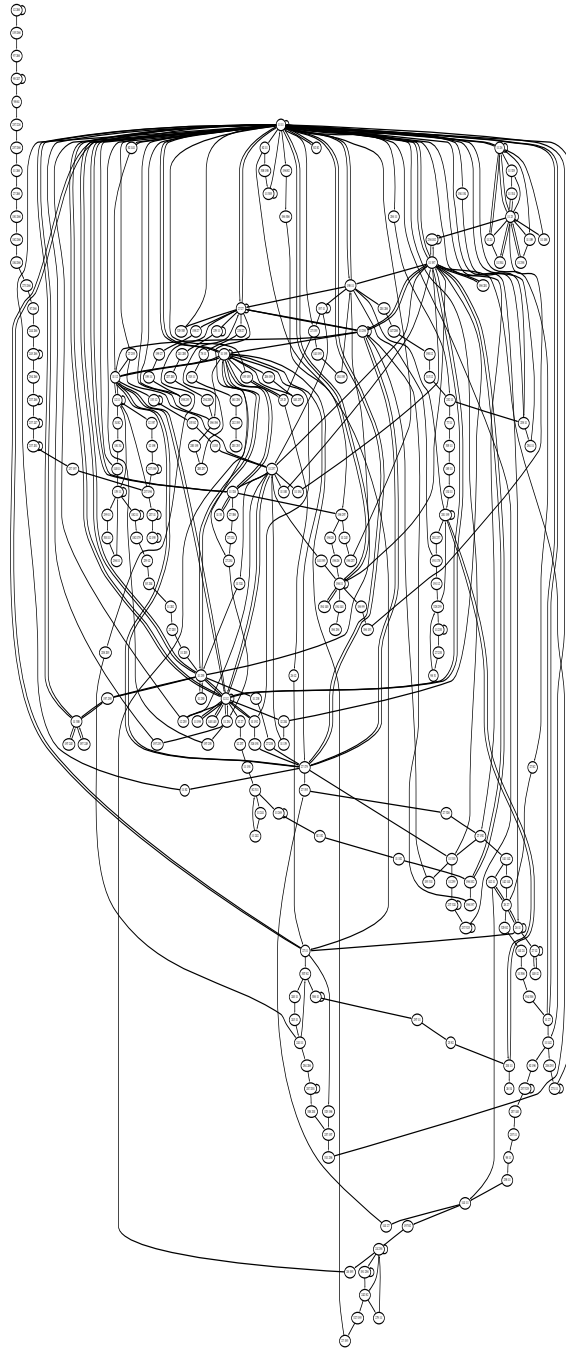
Figure 3: A transition graph that represents the movement patterns of the hips in a small corpus of 38 short ballet sequences. The numbers in each state identify the discretized position of the joint. Edge weights and isolated vertices have been omitted in the interests of clarity.

7

from this figure. The intricate patterns in these dance progressions are reflected by the complex topology of the graph.

2.2.2 Coordinating Joint Movements

A joint transition graph represents the behavior of a joint *in isolation*. This information, alone, cannot capture the physical constraints that govern the coordination of the joints in the body. For example, if the shoulder is in its resting position with the palm facing the thigh, the elbow can bend nearly 180 degrees, but if the shoulder is turned 90 degrees on its long axis (until the palm faces backwards), the elbow can only bend about five degrees before the hand collides with the leg. In order to construct sensible interpolation sequences, we need a simple and efficient model of this type of joint coordination.

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, but doing so engenders a combinatorial explosion in the search space. There are sensible ways to reduce the complexity of the problem, however; to a first approximation, a joint is not influenced by *every* other joint in the body. The position of the wrist, for instance, strongly affects the position of the fingers but has little effect on the toes. We put this simplifying assumption into effect by using an *influence diagram*[11] that reflects the structure and physics of the human body to explicitly represent the relationships of the joints to one another. As shown in figure 4(b), the nodes (joints) in the tree only affect the position of their immediate children. The pelvis is the root of this tree; three branches
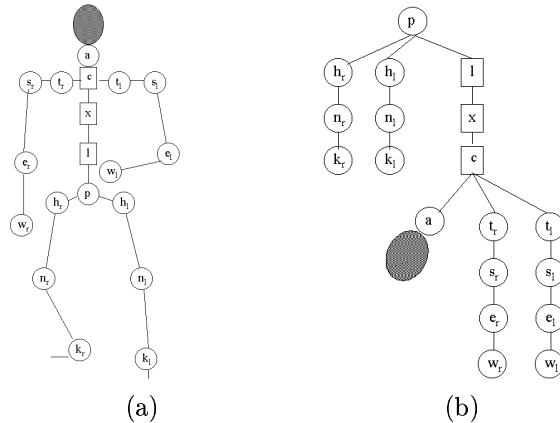

(a)   (b)

Figure 4: An influence diagram that explicitly represents the coordination of joints of the human body. Part (a) depicts the body and part (b) shows the interjoint 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. Without this simplifying assumption, the search space for this problem is intractable.

8

lead from this root to nodes corresponding to the right hip, the left hip, and the lower spine[7]. Each hip joint is the parent node to a knee, and so on. We assign a conditional probability distribution, estimated from the corpus, 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 of quaternions $q \in Q^\lambda, r \in Q^\mu$.

2.3   A JOINT-WISE INTERPOLATION ALGORITHM

Given a pair of discretized body postures $(\vec{s}, \vec{t})$ and a set of 23 transition graphs (one for each joint), we can use a memory-bounded A* search strategy[12] to find an interpolation subsequence that moves smoothly between $\vec{s}$ and $\vec{t}$. In general, 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* — 23-vectors of discretized quaternions that represent full body positions. To generate successors of a body state $\vec{s}$, we first use the transition graphs to find successors for each *joint state* s_λ independently, and then take all combinations (cross product) across the joints to obtain the list of body-state successors. From this list, we can filter out the disallowed body positions using the influence diagram and the probability distribution of parent-child pairs. The 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.

The score assigned to a body state $\vec{u}$ has two parts:

1. the cost of the path from the initial state $\vec{s}$ to $\vec{u}$

2. an estimate of the distance between $\vec{u}$ and the goal state $\vec{t}$

The cost of the path starting at $\vec{s}$ and ending at $\vec{u}$ is simply the sum of the costs of the transitions taken in the path. Furthermore, since each body movement is composed of a group of joint movements, we can compute the cost of one body-state transition by summing the weights over the edges traversed by the joints. To make this concrete, suppose we are trying to find an interpolating path between the body states $\vec{s}$ and $\vec{t}$. At some point in the search, we reach the body-state $\vec{u}$ and must assign the path from $\vec{s}$ to $\vec{u}$ a score. If we write the path from $\vec{s}$ to $\vec{u}$ as $\vec{s} \to \vec{u} = (\vec{x}^1 = \vec{s}, \vec{x}^2, \cdots, \vec{x}^{z-1}, \vec{x}^z = \vec{u})$, we can express the

---

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

cost of such a path as

$$g(\vec{s} \to \vec{u}) = \sum_{i=1}^{z-1} \sum_{\lambda} w^{\lambda}_{x_{\lambda}^{i}, x_{\lambda}^{i+1}}$$

The heuristic part of the score, $h(\vec{u})$, estimates how far $\vec{u}$ is from the goal state $\vec{t}$. $h(\vec{u})$ is calculated by summing the weights of the shortest paths from u_{λ} to t_{λ}, $u_{\lambda}, t_{\lambda} \in G^{\lambda}$ over all the joints. We obtain these shortest path weights using Dijkstra's single-source shortest path algorithm[7], implemented as described in [4]. The final score assigned to body-state $\vec{u}$ is then $f(\vec{s} \to \vec{u}) = g(\vec{s} \to \vec{u}) + h(\vec{u})$.

At the time of this writing, we have only done extensive testing on a greedy search strategy that ignores the cost of paths and scores nodes in the search based solely on the estimated distance between them and the goal (i.e., $f(\vec{s} \to \vec{u}) = h(\vec{u})$). In the following section, we describe the implications of this strategy and suggest how different A* scoring functions are likely to affect the interpolation sequences. We are also working on incorporating more information about 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; accomplishing this will require wide-ranging adaptations to the basic A* algorithm and perhaps even a wholly different approach. Finally, we are also in the process of testing how different influence diagram topologies affect the interpolation algorithm's ability to select good postures during the search. (For example, to model and enforce the symmetry of the body, we could combine left and right counterparts into one node.)

3   RESULTS AND EVALUATION

The "goal" of choreography is aesthetic appeal, so it is difficult to analyze the results of this work using standard scientific methods[8]. However, there are some standard rules, procedures, and patterns in certain dance and martial arts genres that can be used to evaluate the interpolation sequences generated by the corpus-based techniques described in the previous sections. The evaluation described in this section is a highly condensed transcript of a dozen one- to two-hour sessions, wherein expert dancers — primarily Professor David Capps of the Department of Theater and Dance at the University of Colorado, an accomplished dancer and choreographer whose works have appeared on stages around the world, and Nadia Rojasadame, a student in that department and the composer of the adagio used to generate the variations shown in figures 1 and 2 — went through the results frame by frame, answering and then discussing the following questions:

- Does this posture transition look reasonable?

---

[8]The very notion of objective, quantifyable evaluation elicited much consternation and mirth — along with some offense — from our expert dance consultants.
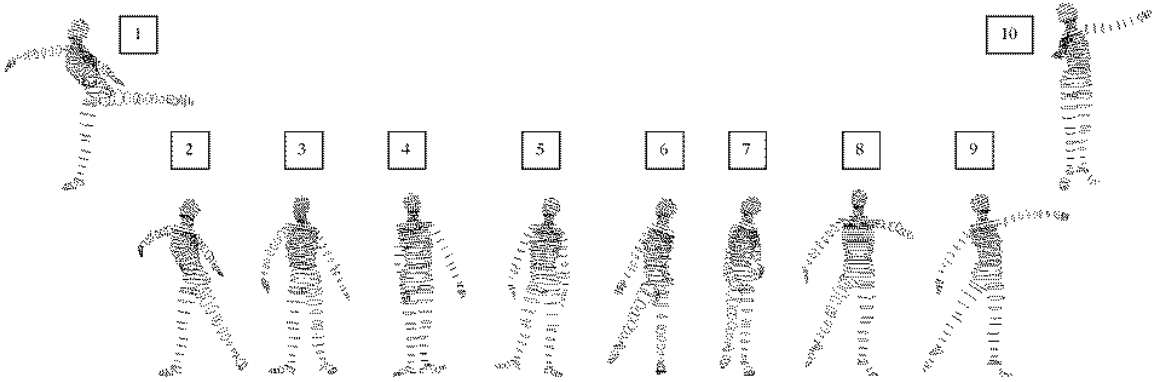
Figure 5: An interpolation sequence computed by the corpus-based techniques described in the previous section. The starting and ending positions passed as input to the interpolation procedure are shown at the top left and top right, respectively; the eight frames below them were computed by the interpolator.

- If so, why and how?

- If not, why and how? What would *you* do instead? How many poses would you assume in doing so?

In order to make this process less subjective, we are developing a formal evaluation protocol, consisting of several subsequences and a series of scored questions about the flow of the movement therein, to be administered to groups of University of Colorado dance students.

Figure 5 shows a movement sequence that the learning and search algorithms described in the previous sections produced when given the task of interpolating between the fifth and sixth frames of the ballet sequence in figure 2. The search strategy was a simple greedy approach — an A* score $f(\vec{s} \to \vec{u}) = h(\vec{u})$ that only factored in the distance to the goal — and the corpus included 38 short ballets. The starting and ending body postures (top left and top right in figure 5, labeled $\boxed{1}$ and $\boxed{10}$, respectively) are quite different; note the facing of the dancer and the weight distribution on the feet, for example. The eight-move interpolation sequence computed by the interpolator moves between those positions in a very natural way. Its first move, for instance, is to lower the left leg, a natural strategy if one is going to change one's facing and end up on two feet. The following move is a simple weight shift (frames $\boxed{4}$ and $\boxed{5}$), in preparation for a lift of the right leg. This lift, which is not strictly necessary to move from the fifth frame to the tenth, is an innovation that the program inserted because of the observed patterns in the corpus; it reflects the fact that ballet dancers rarely spin with *both* feet flat on the ground. Perhaps the most interesting thing about this interpolation sequence, from a balletic standpoint,

is the relévé[9] that the interpolation procedure inserted between frames $\boxed{6}$ and $\boxed{10}$. Many relévés appear in the corpus, but none of them are associated with upper body positions that resemble the one that appears in this sequence. Our algorithm has invented a physically *and stylistically* appropriate way to move the dancer between the specified positions. The interpolation sequence in figure 5 includes a variety of other stylistically consistent innovations as well; consider, for example, the uplifted chest and chin in frames $\boxed{7}$ and $\boxed{9}$ — posture elements that are quintessential ballet style. Recall that these postures were not simply pasted in verbatim from the corpus; they were synthesized *joint by joint* using the transition graphs and influence-diagram directed A* search, and their fit to the genre is strong evidence of the success of the methods described in the previous section.

The original ballet sequence from which the snapshot in figure 1 was drawn contained 68 frames, and the chaotic shuffling scheme introduced 23 abrupt transitions into the variation (e.g., frames 5 → 6 of figure 2). In eleven of those 23 cases — including the one depicted in figure 5 — our interpolation scheme was successful in interpolating smoothly between the two moves that framed the gap. The interpolation subsequences so constructed, which ranged in length from two to 60 frames, included a variety of stylistically consistent and often innovative sequences; among other things, the interpolation algorithms used relévés, pliés and fifth-position rests in highly appropriate ways — and all with no hard coding. From a subjective artistic standpoint, the results have some room for improvement; there are still five somewhat-awkward transitions in the 185 total frames of the 11 interpolation sequences. A less-subjective way to evaluate the success of this scheme is to compare the length of these interpolation sequences to the distance between the corresponding postures *in the original piece*, which is presumably a good metric for how long it would take a human to move from one to the other. For the most part, the interpolated sequences were shorter than or the same length[10] as the number of frames separating the corresponding positions in the original piece, which indicates that the search strategies are working well. mpeg movies of this adagio sequence and its chaotic variation — both with and without interpolation— are available on the web[11].

This example brings out two significant failure modes of this approach. The algorithms cannot find interpolation subsequences between body positions that occur in reversed temporal order — e.g., places where the chaotic shuffler has forced a jump backwards in time, inserting a move into the variation that appeared earlier in the original piece. Secondly, the algorithms sometimes introduce relatively long paths between positions that appear very similar; in one such instance, where the task was a simple 90-degree rotation of the right shoulder around the long axis of the arm, the algorithm constructed an 65-move sequence that involved much leg and trunk movement. Both of these problems

---

[9] A relévé, which consists of lifting up on one's toes, is a stylistically required component of a direction shift in ballet.

[10] Five were shorter (77% average), four were the same length, and two were somewhat (150% and 110%) longer.

[11] www.cs.colorado.edu/~lizb/chaotic-dance.html

are the result of limited corpus size and corresponding patterns in the joint transition graphs. These graphs are far from being connected, so some joint orientations are not reachable from others. Even when they *are* connected, the search may have to wander all over the graph to find a path between two given vertices. In a large, rich corpus, the graphs would be highly connected, giving the search algorithms more leeway. In the existing corpora, however, the paucity of edges constrains them to very narrow (and long) search paths that can translate to stilted, idiosyncratic movement sequences. This is an unavoidable problem in this application, unfortunately; the dance world has not yet embraced the notion of computer animation, so the availability of animated dances is quite limited.

Long, linear vertex chains like the ones at the top left of figure 3 are introduced into the joint transition graph when one animation in the corpus progresses through orientations that do not occur in other animations. The directionality in these chains makes it impossible for the search to move "upstream," which is the cause of the first failure mode described in the previous paragraph. We could fix this problem, artificially, by introducing reverse edges into the graphs in some kinesiologically and stylistically justifiable way. For every transition $\vec{s} \to \vec{t}$ seen in the corpus, for example, we could introduce an edge from $\vec{t}_\lambda$ to $\vec{s}_\lambda$ for every joint λ. The implicit assumption here is that *it is always possible to reverse the motion of a joint*[12]. Thus, at the expense of destroying some of the accuracy with which the original approach modeled the temporal asymmetry of the genre, we could force the graphs to be connected. We are currently investigating what probabilities to use on these reverse edges. Artificially introduced reverse transitions would not solve the second problem; chains — even bidirectional chains — tend to lengthen interpolated sequences. One solution to this problem is to add more examples to the corpus to enrich its connectivity. If more examples are hard to come by, another (artificial) solution is to perform a coarser discretization to minimize the number of possible states a joint can assume. We are currently experimenting with different discretization resolutions to simultaneously minimize the number of nodes and maximize the statistical information content of the transition graphs.

The greedy A* search strategy is reflected by "inefficiencies" in the interpolation sequences — places where the dancer appears to be headed towards the goal state, but then moves away. For example, one of the interpolation goals in figure 5 is to change the facing almost 180 degrees, from left to right. By the fourth frame, the dancer has turned to the right, but in the fifth frame s/he has turned back to the left again, which is part of what necessitates the relevé sequence between frames $\boxed{6}$ and $\boxed{7}$. We are in the process of testing different search strategies and analyzing the results; instead of choosing the state that is closest to the goal, for instance, we are incorporating the path weights up to the current point in the solution as part of the scoring function. This should allow the search algorithm to find shorter, more-direct sequences. Finally, note that

---

[12]This makes sense for classical ballet, but not modern dance; motion in the former tends to be "circular" in space, whereas in the latter, one often moves a limb out and back along the same path.

some search strategies — e.g., always taking the highest-probability branch — can be a significant source of cliché.

In order to explore the effects of joint coordination, we removed the influence diagram and ran simple, uncoordinated A* search to find paths between positions. The resulting sequences were extremely interesting. To the layman's eye, they look jerky and unappealing, so we expected negative comments about them from the experts. However, it seems that an uncoordinated path through a classical ballet corpus is a very good way to generate *modern* dance sequences, and the results were inventive and appealing: "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 the ballet motif.

The interpolation procedure is fairly rapid. Applying greedy search to the 23 abrupt-transition pairs in the 68-frame variation, for instance, required[13] 280 seconds on an HP9000/735 workstation running HP-UX v10.20 for a corpus containing 1720 ballet postures. A more-complex scoring function will obviously require longer run time. Preliminary runs of non-greedy A*, for example, required 500 seconds to perform the same task and yielded similar results, in terms of quality, sequence length, etc. The complexity also increases with corpus size; the same (non-greedy A*) task on an augmented corpus of 5000 postures — the 1720 original frames plus 3280 non-ballet sequences — required 3620 seconds. The chaotic shuffling procedure is also fast: for a 1000-position movement sequence, the chaotic shuffling procedure required 18 seconds on the same workstation, while a 9000-move sequence required 156 seconds.

4   CONCLUSION

By applying techniques from graph theory, artificial intelligence, and statistics to a corpus of movement sequences from a particular genre, the interpolation methods described in this paper automatically construct interpolation sequences that move from one specified body posture to another in a *physically and stylistically coherent* fashion. These tactics can be used to smooth abrupt transitions that result from subsequence reordering, a common creative mechanism in modern choreography that can be emulated mathematically by using chaotic dynamics to generate variations.

Evaluating the results of this work is necessarily somewhat subjective. We have shown animations of a variety of different chaotic variations to hundreds of people, including dozens of dancers and martial artists, both with and without smoothing of the abrupt transitions. We have also worked in depth with several expert dancers in order to evaluate those interpolation sequences sensibly. The consensus is that the chaotic variations with smoothed transitions 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 nontrivial accomplishment. A previous attempt to use mathematics to generate

---

[13]This will obviously depend on the positions involved.

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, *primarily because of the awkwardness at the transition points*[14].

Many of the techniques used here, as well as others on which we are currently working, were inspired by solutions to similar problems that arise in computational linguistics (e.g., learning a grammar from a corpus and then using it to construct meaningful sentences). For example, one can view the transition graphs in section 2.2.1 and figure 3 as first-order Markov chains, where a single chain represents the probabilistic behavior of each joint in the body.

The objective of this research project was to tailor generic strategies for a specific high-dimensional search problem in an unusual and demanding domain. The results could be extended to other domains where the genre of sequence is important, such as speech recognition (e.g., filling in missing parts of a signal) or text. Finally, the implementation of these algorithms allows for arbitrary body topologies, so we are by no means limited to *human* motion sequences — though one would, of course, have to adapt the quaternion-based symbol set and the influence diagram to the topology of the limbs and joints that are involved.

Acknowledgements

References

[1] E. Bradley and J. Stuart. Using chaos to generate choreographic variations. In *Proceedings of the Fourth Experimental Chaos Conference*, pages 451–456, 1997.

[2] E. Bradley and J. Stuart. Using chaos to generate variations on movement sequences. *Chaos*, 8:800–807, 1998.

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

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

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

---

[14]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."[3].

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

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

[8] 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.

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

[10] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. Animating human athletics. In *Proceedings of SIGGRAPH*, 1995.

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

[12] P. Winston. *Artificial Intelligence*. Addison Wesley, Redwood City CA, 1992. Third Edition.

[13] W. L. Wooten. *Simulation of Leaping, Tumbling, Landing, and Balancing*. PhD thesis, Georgia Institute of Technology, 1998.