Automated Derivation of Behavior Vocabularies for Autonomous Humanoid Motion

Odest Chadwicke Jenkins[*]
cjenkins@usc.edu

Maja J Matarić
mataric@usc.edu

Interaction Lab
Center for Robotics and Embedded Systems
Department of Computer Science
University of Southern California
941 W. 37th Place
Los Angeles, CA 90089-0781

ABSTRACT

In this paper we address the problem of automatically deriving vocabularies of motion modules from human motion data, taking advantage of the underlying spatio-temporal structure in motion. We approach this problem with a data-driven methodology for modularizing a motion stream (or time-series of human motion) into a vocabulary of parameterized primitive motion modules and a set of meta-level behaviors characterizing extended combinations of the primitives. Central to this methodology is the discovery of spatio-temporal structure in a motion stream. We estimate this structure by extending an existing nonlinear dimension reduction technique, Isomap, to handle motion data with spatial and temporal dependencies. The motion vocabularies derived by our methodology provide a substrate of autonomous behavior and can be used in a variety of applications. We demonstrate the utility of derived vocabularies for the application of synthesizing new humanoid motion that is structurally similar to the original demonstrated motion.

Categories and Subject Descriptors

I.2.9 [**Artificial Intelligence**]: Robotics; I.2.6 [**Artificial Intelligence**]: Learning; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*; I.5.3 [**Pattern Recognition**]: Clustering

General Terms

Algorithms,Design,Performance

---

[*]Student author

Keywords

autonomous humanoid agents, humanoid robotics, spectral dimension reduction, motion vocabularies, motion primitives, kinematic motion segmentation

1. INTRODUCTION

In our view of creating autonomous humanoid agents, the ability to produce autonomous control relies on a solid foundation of basic "skills". These skills represent the primitive-level capabilities of the agents and form a *primitive behavior repertoire* [14]. Regardless of the control architecture (behavior-based, planning, hybrid, reactive), a representative primitive behavior repertoire is useful for producing autonomous motion output.

This viewpoint begs the question of how one determines an appropriate primitive behavior repertoire. Typically, primitive behaviors are chosen manually based on domain knowledge of the desired classes of motion or domain specific heuristics. Manual estimation of a primitive repertoire, however, can be subject to decision errors, including design, parameterization, and implementation errors. These errors can lead to problems with scalability to new or modified behaviors and interference between behaviors. Furthermore, the manual effort required to construct and maintain a primitive behavior repertoire can be costly.

In this paper, we present an alternative method (Figure 1) for constructing a primitive behavior repertoire, or a *motion vocabulary*, thorugh learning from demonstration. More specifically, we extract a motion vocabulary using the underlying spatio-temporal structure from a stream of human motion data. The stream is expected to be of motion that is not explicitly directed or scripted, but is indicative of the types of motion to be represented by the repertoire. We envision the use of our methodology with motion capture mechanisms, such as those developed by Measurand Inc. [9], Chu et al. [5], and Mikic et al. [15], which are suited for capturing extended-duration motion (over the course of hours). Such motion would contain a variety of activities, from natural or more directed ones, structured by some underlying spatio-temporal representation. By extracting this structure, our methodology derives vocabulary modules, with each module representing a set of motion with a common theme or meaning (e.g., punch, jab, reach).

We address several issues involved in building motion vocabularies. First, we address the derivation of structure from human motion using dimension reduction, similar to Fod et al. [7]. Most existing dimension reduction techniques assume spatial data which have no temporal order. However, we seek to take advantage of both spatial and temporal dependencies in movement data. Tenenbaum et al. [21] alluded to use of temporal order within the context of Isomap, a method for nonlinear dimension reduction. Our approach extends Isomap to extract spatio-temporal structure (i.e., data having nonlinear spatial structure with temporal dependencies). With additional clustering and interpolation mechanisms, we derive parameterized *primitive motion modules*. These primitives are similar to "verbs" in the manually-derived *Verbs and Adverbs* vocabularies [17]. Drawing an analogy to linguistic grammars, primitive motion modules could be considered terminals. Further dimension reduction iterations allow for the derivation of *meta-level behavior modules*. These behavior modules extend the existing vocabulary to represent more complex motion that is present in the demonstration motion through sequencing of the primitive modules. Behavior modules are similar to the "verb graphs" in Verb and Adverbs and could be considered non-terminals in a grammar.

We believe there are many applications where our derived vocabularies can facilitate autonomous control systems for humanoids, such as imitation of humans. For this paper, we demonstrate this potential for the application of motion synthesis. Using only a derived meta-level behavior, our vocabulary can synthesize non-smooth motion at interactive speed.

2. RELATED WORK

Behaviors for an agent or a robot typically express control at the motor, skill, or task level. Control of an the agent at the motor level acts by prescribing commands directly to the system's actuators. At the skill level, behaviors express the capabilities of the agent as a set of modules. Each module provides the ability to control the agent to perform non goal-directed actions. Skills are models expressed as parameterized programs for motor level controllers, without the ability to strategize about objectives or encode sematics about the world. Task level behaviors are programs of skills or motor commands directed towards achieving an agent's goals specified with respect to its environment. Behaviors derived by our approach are skill-level behaviors that can be converted into torque commands for a physical humanoid and/or used as a substrate for task-level humanoid control.

Our approach is closest in its aims and methodologies to two previous approaches for constructing skill-level motion vocabularies, *Verbs and Adverbs [17]* and *Motion Texture [13]*. Both of these have desirable properties in that their vocabularies can synthesize motion at run-time without user supervision. Both use a two-level approach, in which a primitive-level is used for motion generation and a meta-level is used for transitioning between primitives.

However, each of these approaches has shortcomings for automatically deriving vocabularies with observable meaning. In our approach, we aim to derive vocabularies that are structurally similar to those of Verbs and Adverbs. Verbs and Adverbs vocabularies are manually constructed by a skilled user and benefit from human intuition. We will be trading off these semantically intuitive primitive modules for
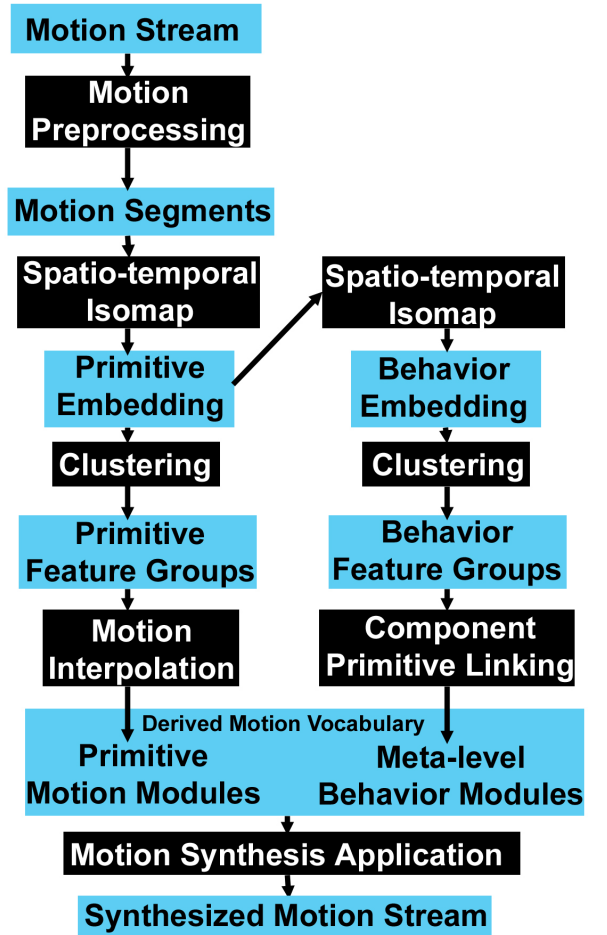

Figure 1: Flowchart of approach. The input to the system is a motion stream, which is segmented (using one of several approaches). Dimension reduction, clustering, and interpolation are applied to the segments to derive primitives motion modules. Using the initial embedding, another iteration of dimension reduction and clustering is applied to find behavior feature groups. Meta-level behaviors are formed by determining component primitives from a behavior unit and linking those with derived transition probabilities. The resulting primitive and behavior vocabulary is used to synthesize novel motion streams.

the significant amounts of training, time, and effort saved by automated derivation. Furthermore, Verbs and Adverbs requires *a priori* knowledge of the the necessary verbs and their connectivity, which is a potential source of vocabulary problems and not required in automated derivation.

Similar to Motion Textures, our aim is to break down an extended stream of human motion data into a set of primitive modules and a transitioning mechanism. However, the guiding principle of Motion Texture is to preserve the dynamics of motion to allow for synthesis of similar motion. In contrast, the aim of our vocabulary derivation methodology is to produce primitive behaviors that have an ob-

servable theme or meaning. This difference can be seen in the segmentation of the demonstration motion into intervals. In Motion Textures, motion segments are learned so they can be accurately reproduced by a linear dynamical system, within some error threshold. In our approach, motion segmentation incorporates domain knowledge by using some heuristic criteria in an automated routine, thus decoupling the definition of a "motion segment" from the internal learning machinery. While automatically determining the appropriate segmentation is an open problem, we trade a linearly optimal segmentation for segments with an understandable meaning containing potentially nonlinear dynamics.

Kovar et al. [11], Lee et al. [12] and Arikan and Forsyth [1] have also presented work for building directed graphs from motion capture. These methods, however, are more specific to motion synthesis based on user constraints rather than providing a foundation for control architectures. Brand and Hertzmann [4] developed a method for separating human motion data into stylistic and structural components using an extension of Hidden Markov Models. This method assumes the motion is specific to a single class of behavior with stylistic variations.

Wolpert and Kawato [22] have proposed an approach for learning multiple paired forward and inverse modules for motor control under various contexts. Our focus in this work is modularization of kinematic motion rather than learning inverse dynamics. Ijspeert et al. [8] have presented an approach for learning nonlinear dynamical systems with attractor properties from motion capture. Their approach provides a useful mechanism for humanoid control robust to perturbations, but only for a single class of motion.

Projects, such as those by Rickel et al. [16] and Kallmann et al. [10], have been working towards building architectures for believable humanoid agents for virtual environments. As a part of those efforts, human figure animation using behavior vocabularies is a necessary component for creating such agents. Additionally, work by Bentivegna et al. [2, 3] assumes the presence of a set of vocabulary of primitives for performing high-level task-oriented robot control. The behavior vocabularies used in such projects require significant manual attention to create and maintain. We envision our vocabulary derivation methodology providing skill-level behaviors useful to those task-level approaches. Thus, the amount of necessary manual effort for skills can be eliminated or significantly reduced.

3. SPATIO-TEMPORAL DIMENSION RE-DUCTION

Central to our motion derivation methodology is the ability to transform a set of motion data so that its underlying structure can be estimated. Our approach is to use dimension reduction to extract structure-indicative features. Each extracted feature represents a group of motions with the same underlying theme and can then be used to construct a primitive motion module realizing the theme. We use joint angle data as input, and consider a motion segment as a data point in a space of joint angle trajectories with dimensionality equal to the product of the number of frames in the segment and the number of degrees of freedom in the kinematics of the humanoid data source. The dimension reduction mechanism transforms the motion data into a space in which the extraction of features can be performed.

Recently, Fod et al. [7] used *Principal Components Analysis (PCA)* to derive movement primitives from human arm motion. While the reduction of dimension using PCA is useful, the extraction of features from the linear PCA-embedded subspace is unintuitive. The derived principal components (PCs) are not convenient features, as they do not have a meaningful interpretation at least in part because the input data are non-linear. Alternatively, features could be constructed by clustering the PCA-dimension-reduced data. However, without an *a priori* specification for the number of clusters, this approach also produces unintuitive features. Furthermore, the best result from clustering provides simply a discretization of the space of joint angle trajectories. This spatial limitation restricts our ability to extract features that vary across a wide volume in the space of joint angle trajectories, potentially overlapping with other features.

Several nonlinear dimension reduction techniques, including Isomap [21], Locally Linear Embedding [18], and Kernel PCA [19], address the linear limitation of PCA. However, these approaches perform a *spatial* nonlinear dimension reduction. Consequently, our attempts to apply these approaches to arm motion data resulted in the same feature extraction problems as described for PCA, except with fewer and more appropriate extracted features. In the remainder of this section, we describe our extension of Isomap for spatio-temporal data, such as kinematic motion.

Human motion data have a meaningful temporal ordering that can be utilized for feature extraction. In our approach, we use long streams of motion as input, which we then segment (automatically or manually), retaining the natural temporal sequence in which the segments occur. We extend the Isomap algorithm to incorporate this temporal structure in the embedding process. Figure 2 illustrates the differences between PCA, Isomap, and our spatio-temporal extension of Isomap. This figure illustrates three embeddings of sequentially-ordered trajectories of a point in 3D following an "S-curve". The PCA embedding simply rotates the data points, providing no greater intuition about the spatial or temporal structure of the S-curve data. The Isomap embedding unravels the spatial structure of the S-curve removing the "S" nonlinearity, producing the flattened data indicative of the 2-manifold structure of the S-curve. However, the model that has generated the data are a 1-manifold with an S-curve nonlinearity and multiple translated instances. Spatio-temporal Isomap produces an embedding indicative of this 1-manifold structure. This embedding both unravels the S-curve nonlinearity and collapses corresponding points from multiple instances of the S-curve to a single point.

In the remainder of this section, we describe our extension of the Isomap algorithm for spatio-temporal dimension reduction to allow for the extraction of meaningful features. For simplicity, we will assume that the data are always mean-centered (refer to [19] for feature space centering).
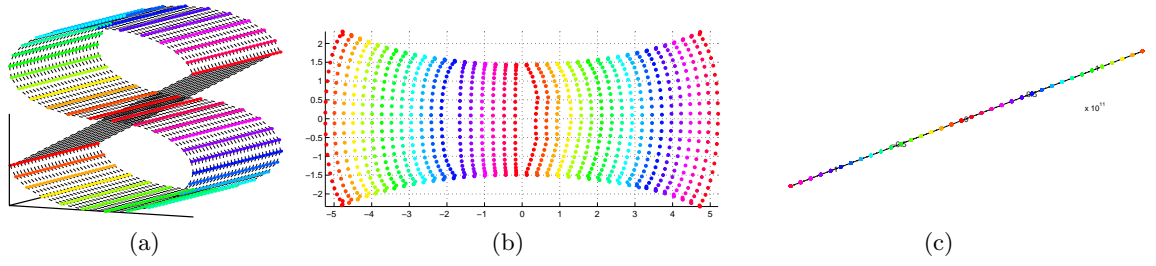
(a)   (b)   (c)

Figure 2: "S"-curve trajectory example motivated by Roweis and Saul. (a) Plot of 3D trajectory of a point moving along an S-Curve with its temporal order specified by the dashed line. The point traverses the S and returns to its initial position along the "S", translated slightly off of the previous S. (b) The embedding produced by spatial Isomap removes the spatial nonlinearity of the S. (c) The embedding produced by spatio-temporal Isomap removes the spatial nonlinearity and collapses the multiple traversals of the S to a line.

The procedure for *spatial Isomap* is as follows:

1. Determine a local neighborhood of nearby points for each point in the data set (through k-nearest neighbors or an epsilon radius).

    (a) Set the distance D_{ij} between point i and a neighboring point j based on a chosen distance metric (e.g., spatial Euclidean distance); set $D_{ij} = \infty$ if j is not a neighbor of i.

2. Compute all-pairs shortest paths for the D matrix (using Dijkstra's algorithm)

3. Construct d-dimensional embedding by an eigenvalue decomposition of D, given d

The intuition behind the Isomap algorithm is that an eigenvalue decomposition is performed on a *feature space similarity matrix* D instead of an *input space covariance matrix* C (as in PCA). Each element of the input space covariance matrix C_{ij} specifies the *correlation* of input dimension i to input dimension j. The result from the eigenvalue decomposition of C produces linear principal component vectors in the input space that are the axes of an ellipse fitting the data points. Algorithms for Isomap and Kernel PCA use the same basic structure as PCA, except the operation is performed in *feature space*. The feature space is a higher dimensional space in which a linear operation can be performed that corresponds to a nonlinear operation in the input space. The caveat is that we cannot transform the data directly to feature space. For performing PCA in feature space, however, we only require the dot-product (or *similarity*) between every pair of data points in feature space. By replacing the covariance matrix C with the similarity matrix D, we fit an ellipsoid to our data in feature space that produce nonlinear PCs in the input space.

Spatio-temporal Isomap is performed in the same manner as spatial Isomap, except an additional step is introduced to account for data with temporal dependencies. Spatial Isomap uses geodesic distances between each data pair to produce each entry in D, computed as shortest path distances from local neighborhoods. Constructing D in this manner is suitable for data with only spatial characteristics (i.e., independently sampled from the same underlying distribution). If the data have temporal dependency (i.e., a sequential ordering), spatial similarity alone will not accurately reflect the actual structure of the data. We experimented with incorporation of temporal dependencies by the adjustment of the similarity matrix D, through *Weighted Temporal Neighbors (WTN)*, or a different distance metric, through a *phase-space distance metric*. The phase-space distance metric was implemented as the Euclidean distance between two data points with concatenated spatial and velocity information. A comparison of the two methods showed that WTN provided more meaningful embeddings for deriving motion vocabularies.

3.1 Weighted Temporal Neighbors

In spatial Isomap, neighborhoods local to each point x_i are formed by the spatially closest points to x_i. In WTN, these spatial neighbors and adjacent temporal neighbors, points x_{i-1} and x_{i+1}, form local neighborhoods. By including adjacent temporal neighbors, our aim is to introduce a first-order Markov dependency into the resulting embedding. Furthermore, a single connected component can be realized in the D matrix and, thus, include all of the data points in the embedding. We use the constant c_{atn} to regulate the distance in the D matrix between a point and its adjacent temporal neighbors.

WTN also modifies the D matrix based on Common Temporal Neighbors (CTN). We define two data points, t_x and t_y, as *common temporal neighbors (ctn)* if $t_y \in nbhd(t_x)$ and $t_{y+1} \in nbhd(t_{x+1})$, where $nbhd(t_x)$ is the spatial neighborhood of t_x, t_{x+1} and t_{y+1} are data points temporally adjacent to t_x and t_y, respectively. CTN are used to identify points in the local spatial neighborhood that are more likely to be grouped in the same feature. We use a constant c_{ctn} to specify how much to reduce the distance in D between two CTN. By providing a significant distance reduction between two CTN, we ensure that these two points will be *proximal* in the resulting embedding. Two points that are not CTN, but are linked by CTN, will also be proximal in the embedding. We define a set of points in which all pairs in the set are connected by a path of CTN as a *CTN connected component*. Points belonging to a single CTN connected component will be proximal in the embedding. Points not in this CTN connected component will be relatively distal. Thus, CTN connected components will be separable in the embedding through simple clustering.

4. DERIVING BEHAVIORS FROM KINEMATIC MOTION

We present a fully automated approach for iteratively applying spatio-temporal Isomap to human motion data to produce embeddings from which features representing primitive motion modules and meta-level behaviors can be extracted. The first iteration of embedding yields clusterable groups of motion, called *primitive feature groups*. An interpolation mechanism is combined with each primitive feature group to form a *parameterized primitive motion module* capable of producing new motion representative of the theme of the primitive. Spatio-temporal Isomap is reapplied to the first embedding to yield more clusterable groups of motion, called *behavior feature groups*. From a behavior feature group, a meta-level behavior is formed to encapsulate its component primitives and link them with transition probabilities.

4.1 Motion Pre-processing

The first step in the derivation of primitive and behavior modules is the segmentation of a single motion stream into a set of motion segments. The motion streams consisted of human upper-body motion with 27 degrees of freedom (DOFs), with each stream containing performances of various reaching, dancing, and fighting activities. The streams were segmented manually, for ground truth, and also by using *Kinematic Centroid Segmentation (KCS)*. KCS segments the motion of a kinematic substructure (e.g., an arm) based on the motion of a centroid feature that is the average of a set of Cartesian features along the arm. KCS determines motion segment boundaries in a greedy fashion using the following procedure:

1. Set current segment to the first frame

2. Compute distance between centroid at current segment boundary and centroid at every subsequent frame

3. Find first local maximum in centroid distance function

   (a) Traverse frames with a moving window until current frame exceeds a distance threshold and is the maximum value in the window

4. Place new segment at the found local maximum, go to step 2

KCS was applied to each kinematic substructure (left arm, right arm, and torso) independently and proximal segment boundaries were merged. For both segmentation methods, the resulting segments were normalized to a fixed number of frames using cubic spline interpolation.

4.2 Extracting Primitive Motion Modules

Spatio-temporal Isomap was performed on the set of sequentially ordered motion segments using WTN for temporal adjustment of the pairwise distance matrix D. For extracting primitive feature groups, we aim to collapse motion segments which belong to the same CTN connected component into proximity in the resulting embedding. This can be achieved with a significantly large value assigned to c_{ctn}. However, c_{atn} cannot be set to a single constant because the adjacent temporal neighbor of a motion segment may or may not be included in the same primitive feature group. We set c_{atn} to a negative value as a flag for spatio-temporal Isomap to set the distance between adjacent temporal neighbors as their spatial distance.

The resulting embedding produces linearly separable groups that can be extracted automatically by clustering. The clustering method is implemented based on the one-dimensional "sweep-and-prune" technique [6] for detecting overlapping axis-aligned bounding boxes. This clustering method does not require the number of clusters to be specified *a priori*, but rather a separating distance for distinguishing intervals of cluster projections along each dimension. Once clustering is applied, each cluster is considered a primitive feature group.

Next, each primitive feature group is generalized to a primitive module using interpolation. Similar to Verbs and Adverbs [17], we use the set of motion segments in each feature group as *exemplars*. New motions that are variations on the theme of the feature group can be produced by interpolating between the feature group exemplars. To produce new motion variations, an interpolation mechanism would use the correspondence between the data in the input and reduced space. This mechanism maps a selected location in the reduced space to a location in input space, representing a joint angle trajectory. A variety of interpolation mechanisms could be used. We chose Shepard's interpolation [20] because of the simplicity of its implementation. The locations from the embedding are used for interpolation coordinates, although these locations could be refined manually (as in the Verbs and Adverbs approach) or automatically (potentially through spatial Isomap) to improve interpolation results.

4.3 Extracting Meta-level Behaviors

Using the first embedding, we perform another iteration of spatio-temporal Isomap to extract behavior feature groups. From the first embedding, motion segments from the same primitive feature group were proximal. In the second embedding, we aim to collapse primitives typically performed in sequence into behavior features. Consequently, we must collapse motion segments from the corresponding primitive feature groups in this sequence into the same behavior feature group. For the collapsing to take place, we set c_{ctn} to a large constant and c_{atn} to a constant large enough to collapse primitive features but small enough that our existing primitives do not decompose. Regardless of choices for c_{ctn} and c_{atn}, an appropriate embedding will result; however, some tuning of these parameters is necessary to yield more intuitive results. We then perform bounding-box clustering on this embedding to find behavior feature groups.

We now describe how we generalize behavior features into meta-level behaviors by automatically determining component primitives and transition probabilities. Each meta-level behavior is capable of determining valid transitions between its component primitives. A behavior feature group only specifies its member motion segments. By associating primitive and behavior features with common motion segments, we can determine what primitives are components of a certain meta-level behavior. By counting the number of motion segment transitions that occur from a certain component primitive to other component primitives, the transition probabilities from this primitive are established with respect to the specific meta-level behavior.
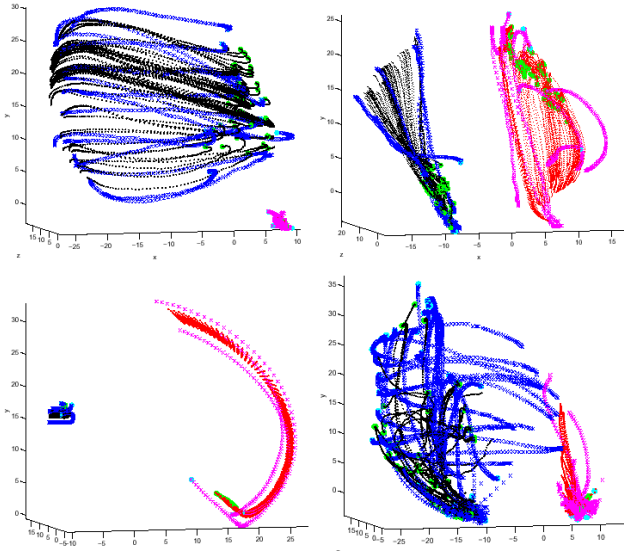
Figure 3: Each plot shows hand trajectories in Cartesian space for motion segments grouped into a primitive feature groups (right hand in dark bold marks, left hand in light bold marks). The primitives shown were found for (a) waving an arm across the body, (b) dancing "the monkey", (c) punching, (d) a merged action. Merged actions result from motion segments inappropriately merged into a single CTN connected component. Hand trajectories are also shown for interpolated motions (right hand in dark marks, left hand in light marks).

For the purposes of continuity, we consider a transition from one primitive to another to be valid if no large "hops" in joint space occur in the resulting motion. More specifically, a transition between two primitives should not require an excessive and instantaneous change in the posture of the humanoid. In order to determine valid transitions, we densely sampled each primitive module for interpolated motions. When contemplating a transition to a component primitive, the meta-level behavior can examine the interpolated motions for this primitive to determine if such a transition would require a hop in joint space that exceeds some threshold.

5. MOTION SYNTHESIS

As an example of the usefulness of the vocabularies we automatically derived, we implemented a mechanism to synthesize a stream of human motion from a user-selected meta-level behavior. An initial primitive is selected by the user or randomly, from the component primitives of the selected behavior. An interpolated motion segment, sampled from the current primitive, is the initial piece in a synthesized motion stream. Given the currently selected primitive and interpolated motion, we collect a set of *append candidates* from the interpolated motion segments of the other component primitives. Append candidates are motion segments that would provide a valid transition if appended to the end of the current synthesized motion stream. Transition validity is enforced with a threshold on the joint-space distance

between the last frame of the current synthesized motion stream and the first frame of the interpolated motion segment. The append candidates are weighted by the transition probabilities from the current primitive and the primitive that produced the candidate segment. The current synthesized motion stream is updated by appending a randomly selected append candidate, considering the candidate weightings. Using the current primitive and synthesized stream, the process of selecting append candidates and updating the stream is repeated until no append candidates are available or a stopping condition is reached.

The motion synthesis application was implemented in Matlab. The motion synthesizer produced motion that was output to a file in the Biovision BVH motion capture format. The synthesizer was able to output 500 frames of motion at 30 Hz in less than 10 seconds. We believe that the synthesis can be faster and attribute any lack of speed to a basic Matlab implementation performing significant file I/O.

While usable, the proposed motion synthesis mechanism remains a very naive means of demonstrate the utility of the derived vocabulary. We have begun development of motion synthesis and motion classification mechanisms that better utilize the derived vocabulary. These mechanisms treat a primitive module as a velocity field and use it as a nonlinear dynamical system to perform prediction or update.

6. RESULTS AND DISCUSSION

Using the implementation of our method, we derived vocabularies of primitive motion modules and meta-level behaviors for two different streams of motion. The first stream contained motion of a human performing various activities, including several types of punching, dancing, arm waving, semaphores, and circular hand movements. The second stream contained only two-arm reaching motions to various set positions. Each stream used 27 DOFs to describe the upper body of the performer and contained 22,549 and 9,145 frames, respectively, taken at 30 Hz. The streams were segmented using KCS and segments were time-normalized to 100 frames. We derived 56 primitives and 14 behaviors for the first stream and 2 primitives and 2 behaviors for the second stream. The results for the first stream are shown in Figure 4. The vocabulary derived from the second stream was as expected; the two primitives, "reach out" and "return to idle", formed one reaching behavior. The second behavior was irrelevant because it contained only a single transient segment. The derived vocabulary from the first stream also produced desirable results, including distinct behaviors for the circular, punching, and some of the dancing behaviors and no distinct behavior for the semaphores, which had no meaningful motion pattern. However, the arm waving behavior merged with several dancing behaviors due to their similarity. Each derived primitive was sampled for 200 new interpolated instances. Our motion synthesis mechanism was used to sequence new motion streams for each derived behavior. Each stream synthesized plausible motion for the structure of each behavior. Excerpts from these streams for a few behaviors are annotated in Figure 4. Additional results and movies are available at http://robotics.usc.edu/~cjenkins/motionmodules.

There is a distinct trade-off in exchanging the convenience of our automated approach for the elegance of a manual approach. The common sense and skill of a human animator or programmer allow for intuitive semantic expressiveness to be

applied to motion. A significant cost, however, is incurred in terms of time, effort, and training. By using our automated approach, we reduce the cost of manual intervention in exchange for useful motion modules with an observable, but not explicitly stated, meaning. Even used with unsophisticated techniques for clustering, interpolation, and synthesis, our derived vocabularies were able to produce plausible motion, limiting manual effort to observing the types of motion produced by each derived behavior.

7. CONCLUSION

We have described an approach for deriving vocabularies consisting of primitive motion modules and meta-level behavior modules from streams of human motion data. We were able to derive these vocabularies based on embeddings produced by our extension of Isomap for spatio-temporal data. Using these derived motion vocabularies, we demonstrated the usefulness of our approach with respect to synthesizing new human motion. Our vocabulary derivation and motion synthesis procedures required little manual effort and intervention in producing useful results.

8. ACKNOWLEDGMENTS

9. REFERENCES

[1] O. Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Transactions on Graphics (TOG)*, 21(3):483–490, 2002.

[2] D. C. Bentivegna and C. G. Atkeson. Learning from observation using primitives. In *IEEE International Conference on Robotics and Automation*, pages 1988–1993, Seoul, Korea, May 2001.

[3] D. C. Bentivegna, A. Ude, C. G. Atkeson, and G. Cheng. Humanoid robot learning and game playing using pc-based vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2449–2454, Lausanne, Switzerland, October 2002.

[4] M. Brand and A. Hertzmann. Style machines. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 183–192. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, July 2000. ISBN 1-58113-208-5.

[5] C.-W. Chu, O. C. Jenkins, and M. J. Matarić. Markerless kinematic model and motion capture from volume sequences. In *To appear in the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, Madison, Wisconsin, USA, June 2003.

[6] J. D. Cohen, M. C. Lin, D. Manocha, and M. K. Ponamgi. I-COLLIDE: An interactive and exact collision detection system for large-scale environments. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 189–196, 218, 1995.

[7] A. Fod, M. Matarić, and O. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12(1):39–54, January 2002.

[8] A. J. Ijspeert, J. Nakanishi, and S. Schaal. Trajectory formation for imitation with nonlinear dynamical systems. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2001)*, pages 752–757, Maui, Hawaii, USA, 2001.

[9] M. Inc. http://www.measurand.com.

[10] M. Kallmann, J.-S. Monzani, A. Caicedo, and D. Thalmann. Ace: A platform for real time simulation of virtual human agents. In *11th Eurographics Workshop on Animation and Simulation*, Interlaken, Switzerland, August 2000.

[11] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. *ACM Transactions on Graphics (TOG)*, 21(3):473–482, 2002.

[12] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics (TOG)*, 21(3):491–500, 2002.

[13] Y. Li, T. Wang, and H.-Y. Shum. Motion texture: a two-level statistical model for character motion synthesis. *ACM Transactions on Graphics (TOG)*, 21(3):465–472, 2002.

[14] M. J. Matarić. Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics. In C. Nehaniv and K. Dautenhahn, editors, *Imitation in Animals and Artifacts*, pages 392–422. MIT Press, 2002.

[15] I. Mikić, M. Trivedi, E. Hunter, and P. Cosman. Articulated body posture estimation from multi-camera voxel data. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 455–460, Kauai, HI, USA, December 2001.

[16] J. Rickel, S. Marsella, J. Gratch, R. Hill, D. Traum, and W. Swartout. Toward a new generation of virtual humans for interactive experiences. *IEEE Intelligent Systems*, 17(4):32–38, July/August 2002.

[17] C. Rose, M. F. Cohen, and B. Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics & Applications*, 18(5):32–40, September - October 1998. ISSN 0272-1716.

[18] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[19] B. Scholkopf, A. J. Smola, and K.-R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[20] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the ACM national conference*, pages 517–524. ACM Press, 1968.

[21] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[22] D. M. Wolpert and M. Kawato. Multiple paired forward and inverse models for motor control. *Neural Networks*, 11(7-8):1317–1329, 1998.
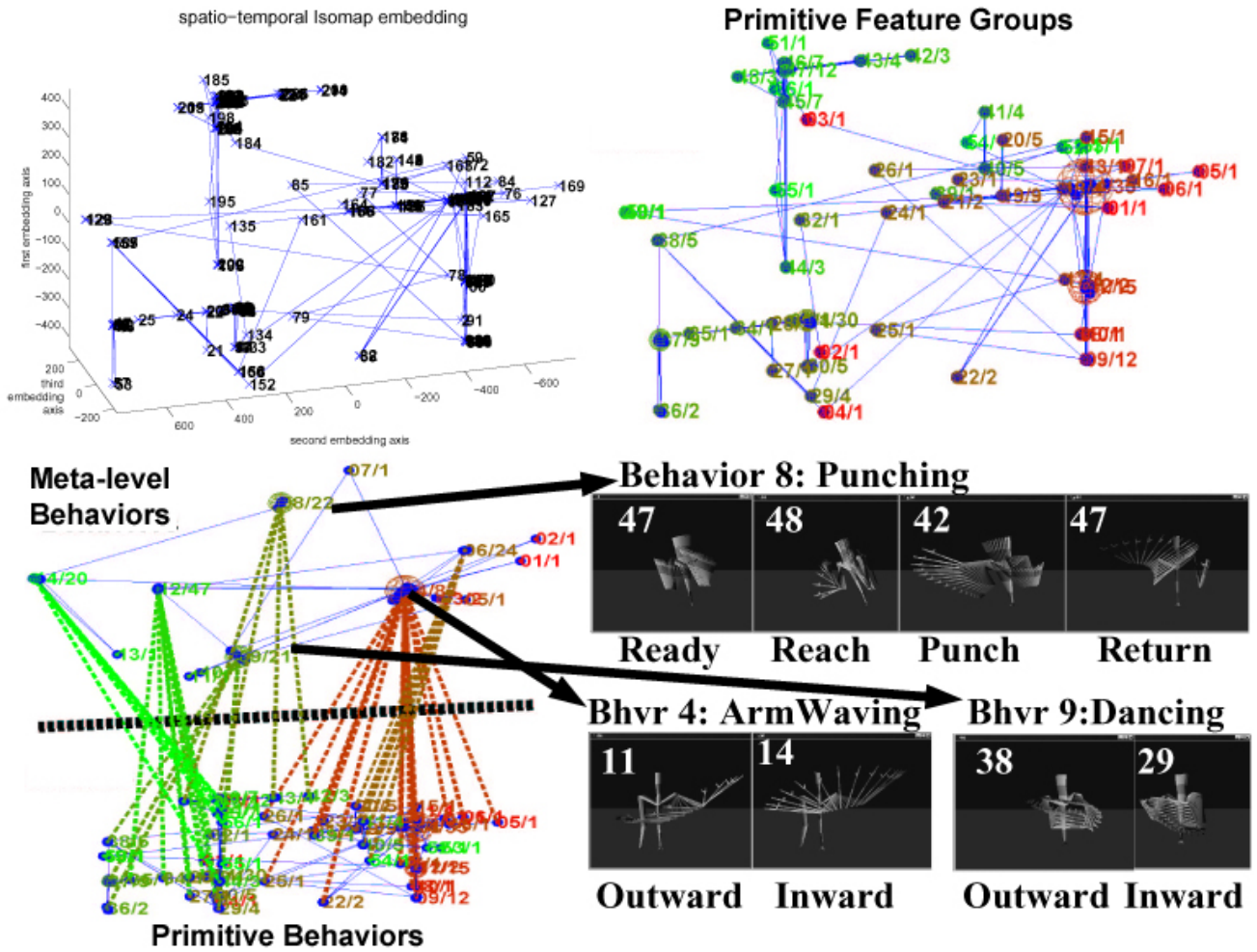
Figure 4: (top left) Embedding of first motion stream using Spatio-temporal Isomap. Each segment in the embedding is marked with an "X" and a number indicating its temporal position. Lines are drawn between temporally adjacent segments. (top right) Derived primitive feature groups produced through clustering. Each cluster is marked with a bounding sphere. (bottom left) The derived behavior units placed above primitives units. Dashed lines are drawn between a behavior unit and its component primitives. (bottom right) Motion streams synthesized from three behaviors (left-arm punching, arm waving across the body, and the "cabbage patch" dance), with each image showing a segment of the stream produced by a specified primitive.
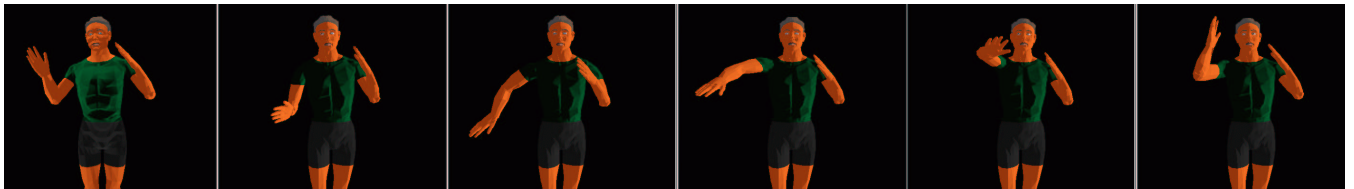

Figure 5: Snapshots of our 20 DOF dynamically simulated humanoid robot platform, Adonis, performing motion synthesized from a derived punching behavior using a PD controller. The motion begins in a fighting posture, the left arm is drawn back, the punch is performed and, finally, the left arm returns to the fighting posture.