Stylized Motion Generalization Through Adaptation of Velocity Profiles

Michael J. Gielniak, C. Karen Liu, and Andrea L. Thomaz

Abstract— **Stylized motion is prevalent in the field of Human-Robot Interaction (HRI). Robot designers typically hand craft or work with professional animators to design behaviors for a robot that will be communicative or life-like when interacting with a human partner. A challenge is to apply this stylized trajectory in varied contexts (e.g. performing a stylized gesture with different end-effector constraints). The goal of this research is to create useful, task-based motion with variance that spans the reachable space of the robot and satisfies constraints, while preserving the "style" of the original motion. We claim the appropriate representation for adapting and generalizing a trajectory is not in Cartesian or joint angle space, but rather in joint velocity space, which allows for unspecified initial conditions to be supplied by interaction with the dynamic environment. The benefit of this representation is that a single trajectory can be extended to accomplish similar tasks in the world given constraints in the environment. We present quantitative data using a continuity metric to prove that, given a stylized initial trajectory, we can create smoother generalized motion than with traditional techniques such as cyclic-coordinate descent.**

I. Introduction

In designing autonomous robots to be deployed in human environments, we recognize that these environments are inherently dynamic and unpredictable. Achieving autonomy in such an environment will require adaptation—the ability to modulate motion to conform to constraints, such as positions of end-effectors [1], [2].

Ideally we want a solution that does not require a large set of basis trajectories from which to generalize a new motion. Additionally, we are interested in the scenario where an algorithm or a professional animator produces a *single* exemplar motion for the robot, from which we automatically generate a new motion that preserves the "style" inherent in the exemplar while satisfying some interactive task constraints (e.g., end-effector location of a robot arm).

Stylized motion is prevalent in the field of Human-Robot Interaction (HRI). Robot designers typically hand craft or work with professional animators to design behaviors for a robot that will be communicative or life-like when interacting with a human partner. For example, waving or nodding gestures for a receptionist robot. Our goal in this work is two-fold:

M.J. Gielniak is with Department of Electrical & Computer Engineering, Georgia Institute of Technology, 777 Atlantic Dr. NW, Atlanta, GA, 30332 USA (mgielniak3@mail.gatech.edu)

C.K. Liu is with the School of Interactive Computing, Georgia Institute of Technology, 85 5th St. NE, TSRB 317, Atlanta, GA, 30308 USA (karenliu@cc.gatech.edu)

A.L. Thomaz is with the Robotics & Intelligent Machines Center, Georgia Institute of Technology, 801 Atlantic Dr. NW, Rm. 256 CCB, Atlanta, GA, 30332 USA (athomaz@cc.gatech.edu)

1) We would like to utilize stylized motion designed by an animator to create life-like robot behaviors for HRI, but minimize the number of exemplar motions needed to broadly apply this motion. In this paper we present a technique that achieves stylized motion generalization from one exemplar.
2) Additionally, we would like to use stylized motion even when there are dynamic task constraints. We present two task constraint examples with our style-preserving generalization technique. In one, the robot has a stylized pick-and-place behavior that can be generalized to any target location. In simulation, we show that a single a stylized baseball swing motion can be generalized to hit a ball pitched anywhere.

Motion adaptation in HRI poses unique challenges that have not been fully addressed in previous methods. Unlike many methods in computer animation that leverage a large database of existing motion sequences, the resultant motion must be autonomously-produced in response to real-time constraints without the need for manual intervention, including manual tuning of parameters. Second, unlike previous methods in robotics that focus only on functionality, in HRI applications it is desirable that the adapted motion be similar to the representative motion, preserving style, continuity, and smoothness.

In this paper, we propose a computationally efficient, real-time method that overcomes these challenges by exploiting the representation of velocity profiles and reusing the information within an existing motion to adapt motion to satisfy interactive kinematic constraints in the world. We describe the technique in detail, and compare results to cyclic-coordinate descent (CCD) for an adaptive pick-and-place behavior modulated based on real-world 3-D object location. Given the qualitative nature of our goals, we demonstrate our results, and we use a continuity metric to show that our technique produces smoother motion than CCD while satisfying constraints.

II. Design Problems for Adaptive Stylized Motion

The goal of this research is to identify a generalizable motion representation and establish a technique for motion generation so that motion can be adapted to satisfy world constraints smoothly in real-time. The approach should be insensitive to initial conditions and preserve style with minimal amounts of data. In prior work, the necessary pieces for a generalizable motion representation have not been integrated with a technique to satisfy constraints and preserve style—our work fulfills this need.

A. Adaptation

Interpolation (a.k.a. blending, morphing, warping, sampling, or deforming) is a frequent choice to satisfy kinematic constraints, especially with simple constraints known in advance, such as end-effector position. The representation can be motion curves, trajectories, joint angles, sets of static poses, shapes, photos, surface meshes, functions, or even video images. However, interpolation is known to lack coherence to more realistic poses or results in loss of the motion "aesthetic." Complexity of the blending approach increases with number of constraints [3]. Interpolation can produce unexpected results, and these techniques require more than just one trajectory, which can be time consuming to develop.

B. Preserving Style

An excellent way to preserve the style or induce desirable variance into a motion is to use training data to learn a model [4], function [5], or distribution [6], [7] of a particular motion in advance. But, such techniques require large amounts of input data, and are not easily extensible to other motions without considerable training.

Style or emotional content can be retargeted from one motion to another, either via a motion induced due to perturbation or pre-defined trajectories using techniques such as filtering and scaling in the frequency domain and leveraging differences between multiple high and low dimensional motion trajectories [8], [9], [10]. However, these techniques to extract and manipulate the information content of a trajectory or a set of trajectories must be augmented with a method to satisfy constraints.

C. The Representation for Motion

Often motion trajectories are specified as a representation of end effector position over time. Then techniques such as motion blending, analytical IK, Jacobian inverse, Jacobian transpose, CCD, gradient methods, least-squares methods, pseudo-inverse, and neural nets are used to identify directions to travel to satisfy point constraints and/or create trajectories in real-time for motion. However, these techniques can sacrifice continuity in favor of the more important goal of getting the end-effector to a certain point in Cartesian space. Also, when the the point constraints are specified outside of the reach of of the robot, oscillation can occur [11].

Others have recognized the importance of a gradient-based representation to achieve insensitivity to variations in the position and scale of related motions [12]. However, their representation remained in Euclidean space, where transformation necessary to apply commands to actuators would be subject to redundancy and multiple solutions.

We claim that the appropriate motion representation for style-preserving generalization is the velocity profile. Since actuators are commanded with joint angle positions in most hardware, the initial condition to any velocity profile remains unspecified and a free variable. Therefore, this representation allows command of a scaled velocity profile to begin from any desired current state, thereby making it an ideal representation for stylized motion. Storage of trajectories as positions in joint angle space are specific to one set of states in the space. Storage of trajectories as positions in end-effector space potentially span a larger region of the state space due to redundancy, but the tradeoff is that an end-effector trajectory cannot be directly applied to the actuators on most robots. The velocity trajectory combines the extensibility necessary to span a large region of the state space with the simplicity of being one linear operation away from direct command as positions to actuators.

III. Research Platform

The platform for this research is an upper-torso humanoid robot we call SIMON (Fig. 1). It has sixteen controllable DOFs on the body and four controllable DOFs on each hand. Each arm has seven degrees of freedom (three at the shoulder, one at the elbow, and three at the wrist) and the torso has two DOFs, with one additional uncontrollable slave joint in the torso fore/aft direction. The head DOFs remained unactuated to produce the results in this paper. The robot operates on a dedicated ethercat network coupled a real-time PC operating at a frequency of 1kHz. To maintain the highest level possible of joint angle position accuracy, the hardware is controlled with PID gains of very high magnitude. The physics-based, dynamic model of the robot hardware used to present some results in this paper is shown in Fig. 1.

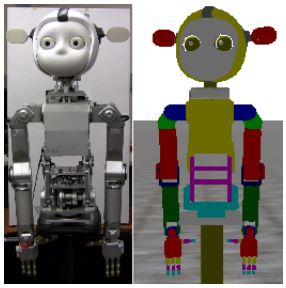

Fig. 1. Hardware (left) and simulation (right) of SIMON.

IV. Generalized Motion Algorithm

The algorithm consists of two steps:

1) Kinematic search. Given any kinematic constraint, such as a point constraint in the Cartesian space, a set of joint angles that satisfy the kinematic constraint are computed in real-time.
2) Velocity modulation. Velocity trajectories of each DOF are modulated so that the trajectory intercepts the joint angle constraint at the appropriate instant in time.

A. Kinematic Search

Computing a set of joint angles to satisfy a point constraint is a common problem in robotics and computer animation and can be solved by many existing methods, such as any inverse kinematics techniques. The strength of our algorithm, TSFKS, does not lie in its novelty, but rather its simplicity, efficiency, and robustness when applied to real-world applications. Empirical evidence from the lab shows that it works

consistently and remarkably well for producing poses that appear "natural."

The two-step forward kinematics search begins with identification of the DOF limits in joint angle space to bound the search for a final configuration of joint angles, which always exists for robot degrees of freedom that have hardware joint angle limits. In the worst case, the DOF limits are set to be the joint angle limits of the hardware.

For certain tasks, the kinematic constraint will be defined only in a localized area, the trajectory will be implemented only in a subspace of the total reachable space by all robot degrees of freedom, or a particular DOF will have a closed form solution. In addition, to reduce computation time, we exclude DOFs that have minimal influence on the kinematic constraint from the kinematic search, such as left-arm DOFs for a right-arm reaching task or wrist DOFs for end-effector constraint. Less significant DOFs appear in fewer terms in the kinematic equations relating joint angles to Cartesian space. Constraining these DOFs to the midpoint of their range introduces negligible, acceptable error into the final solution. In these instances, reducing the joint angle space of possible solutions, allows TSFKS to converge much more quickly.

TSFKS runs in two steps. First, a coarse search reduces the joint angle space, and then a refined search locates the joint angles that satisfy the constraint. During each search, the reduced joint angle space needs to be quantized by either the coarse and fine angle increments. The fine adjustment determines the final error between the end-effector and the constraint in the world. Different angle increments can be used for different joints.

The valid joint angle range is sampled for each DOF using Eq. (1). If a coarse angle increment is, for example, 20% of the search range for a particular DOF, then only four discrete values are required for that DOF in the coarse search.

$$\Theta(i) = \Theta_{min} + \frac{i}{i_{max}} \times (\Theta_{max} - \Theta_{min}) \qquad (1)$$

where,
Θ_{min} = minimum joint angle value in DOF range
Θ_{max} = maximum joint angle value in DOF range
$\Theta(i)$ = discrete value for sample *i*
i = 1, 2, ... up to $i_{max} - 1$
i_{max} = number of increments that will fit in a joint angle range

By performing forward kinematics on all possible unique permutations of discrete values, the set of DOF values that yields minimum Euclidean distance between the constraint in the world and the constraint on the robot is kept, denoted as S. For the refined searched, another set of forward kinematics calculations is performed using permutations derived from Eq. (1), where Θ_{min} and Θ_{max} are constrained to the value returned from S plus or minus one half the coarse angle increment.

Furthermore, we limit the search to leave a buffer of 10-15% of full range away from the hardware limits to insure the integrity of the trajectory produced in the velocity modulation step.

B. Velocity Modulation

After kinematic search, the set of joint angles that satisfy the kinematic constraint is known. The set of joint angles at the point when the trajectory was initiated is used as the initial conditions for the velocity trajectory. If the trajectory is sampled at equidistant time instances, the sum of all joint angles up to the time instance of the kinematic constraint yields a scalar multiplier, given by Eq. (2), which can be computed for each DOF.

$$scalar = \frac{(\Psi(t_{pc}) - \Psi(t_0))}{(\Theta(t_{pc}) - \Theta(t_0))} \qquad (2)$$

where,
$\Theta(t_{pc})$ = joint angle value from original trajectory at the point in time where the point constraint must be satisfied
$\Theta(t_0)$ = initial joint angle value at the beginning of the unmodified trajectory
$\Psi(t_0)$ = actual joint angle value of the robot hardware when the stylized algorithm is initially called
$\Psi(t_{pc})$ = joint angle value calculated from the kinematic search step

Eq. (2) yields one solution per DOF, which is used to multiply the corresponding velocity trajectory for each DOF in the motion. Along with the initial condition of each position trajectory for each DOF in the original motion, this set of scaled velocity trajectories is used to create the new set of position trajectories, which causes the motion to intercept the constraint.

We define a cyclic trajectory to be any trajectory that has a net zero value for the denominator of Eq. (2), such as a constant position trajectory. For such trajectories, Eq. (2) does not produce a viable scalar multiplier for a particular DOF. One alternative to resolve this unique situation is to select another non-cyclic trajectory.

While no such trajectories are used in our particular implementation, several strategies could be used to deal with cyclic trajectories. The problem can be resolved by dividing the original trajectory into time steps of positive increments of the joint angle and time steps where the joint angle value decreases. By scaling time steps in the trajectory where joint angle value increases by a different scalar than the negative increments, cyclic trajectories can be easily accommodated. Alternatively, we can select a different point in time during the trajectory to satisfy the constraint, or use frequency modulation instead of amplitude modulation.

V. Results

We first present comparative results with a typical action trajectory in robot tasks: moving objects. The robot is provided with point constraints in the world in a sequential order, and the robot transitions between the point constraints either with the baseline approach (which is CCD) or our style-preserving technique. In both techniques, the robot derives

the joint angles necessary to reach these point constraints with its end-effector autonomously. The point constraint in 3-D coordinates is provided via the robot vision system. Three point constraints are used to create a complete motion cycle: home (starting) position, 3-D location to pick up the object, and 3-D bin location to drop off the object. The object or the bin are moved to show how the trajectories change as point constraints vary for both techniques. Additionally we provide a simulated baseball example using our style-preserving technique with time varying constraints. These comparisons are most obvious and evident from the video submission accompanying this paper.

A. Baseline: CCD

Cyclic coordinate descent searches for the joint angles to satisfy an end-effector position by traversing a kinematic hierarchy starting from one extreme, optimizing joint angles with respect to end-effector location as much as possible one-by-one. Iteration continues over the chain until satisfying some stopping criteria, such as within epsilon distance of the target end-effector location [13].

Since CCD uses single DOF optimizations, joint usage is unbalanced and certain joints can dominate the resultant motion trajectory. Also, the result of CCD is dependent upon the initial conditions of the posture, often requiring multiple applications to get desirable results. Thus, CCD can result in trajectories or motions that appear awkward or unusual. CCD can also drastically change poses for a small end-effector change. This sensitivity is best exhibited on optimizations that converge to a set of joint angles near the limits of one or more joint angles [14].

CCD is selected as the baseline method for comparison due to fast computation time and very quick convergence in the case of small changes in constraints differences. Unlike other techniques, CCD does not require a Jacobian pseudo-inverse and can compute configurations for kinematic hierarchies of arbitrarily large sizes. We also require comparison against a technique that changes performance with different predefined data. CCD fulfills this requirement because it is a search algorithm, which produces a trajectory that is dependent upon initial and final configurations.

B. Smoothness Metric

Since smoothness of a function is determined by the order of the derivative that remains continuous, ideal smoothness is infinitely differentiable. However, actuator commands are discrete values, thereby forcing the trajectory to be a piecewise linear approximation of the desired trajectory. If the interpolation method must remain linear between sample points, approximations to smooth functions by piecewise linear functions (PLF) are improved by sampling faster and adding more intermediate points [15]. Thus, a metric for comparing smoothness of piecewise linear functions sampled at constant time intervals is presented by Eq. (3). Normalization in Eq. (3) is not required based upon magnitude of the joint angles because it is a relative computation, comparing the two tangents at a particular point (one from the right and one from the left). Trajectories that execute at different rates are accommodated by the time difference in the denominator of Eq. (3).

$$sn = \frac{1}{(N-2)\Delta t} \sum_{i=2}^{N-1} |x(i+1) + x(i-1) - 2x(i)| \quad (3)$$

where,
$x(i)$ = discrete value of the PLF at time index i
N = number of discrete points in the PLF
i = time index
Δt = time between samples in PLF, $x(t)$

A larger value of sn indicates that on average, a PLF has more unequal tangents in and out of the sample points, indicating a worse approximation to a smooth function or in other words, a less smooth function. For the results presented, sn is used as the metric to compare the continuity of the trajectories produced by our technique and CCD.

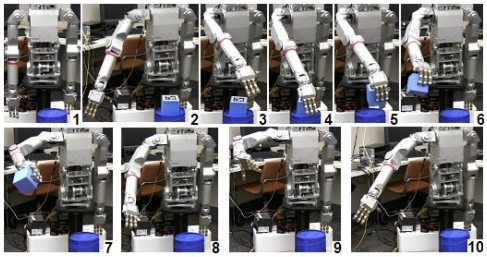

Fig. 2. Trajectory to lift and move a box using CCD.

C. Grasping Example: End-effector Constraints

In Fig. 2, the CCD-generated trajectory begins from the starting position with the arms at the sides of the robot. However, since CCD-generated trajectories are sensitive to initial conditions, in fairness to both algorithms, we shall only compare the regions of the trajectories between box and bin, which constitute identical starting and final positions for both algorithms. Use of this stable, predefined data is what creates equitable comparison between CCD and the stylized technique. As shown in the seventh keyframe of Fig. 2, after lifting the box, the right arm awkwardly uses the elbow and drops the box from the side of its hand into the bin.

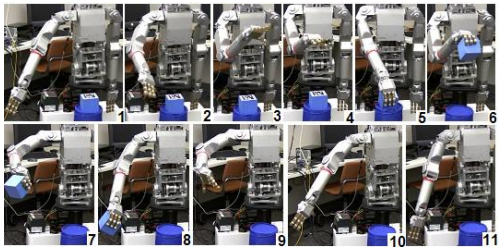

Fig. 3. Trajectory to lift and move a box using CCD.

In Fig. 3, the box location was altered only by a small amount and noticeable changes to the CCD-produced trajectory result. The most perceptible change is in the sixth

keyframes, where the hand moves a considerable distance above the box after grasping the box. Additionally, CCD produces abrupt velocity changes, which are not noticeable from the keyframes but are obvious in the video accompanying this paper.

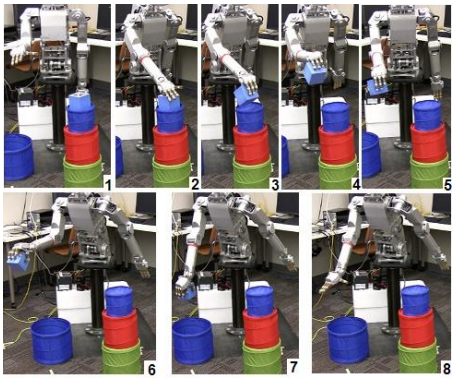

Fig. 4. Trajectory to lift and move a box using our style-preserving technique.

Fig. 4 shows our technique used to modulate the trajectory according to the kinematic constraints. To demonstrate the power of this technique, the particular trajectory selected was one designed by a professional animator, yielding a stylized grasping motion with ideal smoothness and continuity. Use of such a trajectory shows that the algorithm is able to maintain the high quality of the one original motion while satisfying the constraints.

In Fig. 4, the stylized trajectory begins with the right arm bent at the elbow because this was part of the original motion. However, as previously mentioned, trajectory comparison will be restricted between common initial and final configuration, for fairness. Fig. 4 and Fig. 5, which show slightly different box locations, demonstrate that the elbow does not move unnecessarily high in either case, the torso does not exhibit any sudden changes in velocity, and the torso does not bend too far forward. The robot moves to its joint angle limit in the sixth keyframe in Fig. 4 and the fifth keyframe in Fig. 5 to show that the algorithm works well targeting point constraints outside the bounds of the reachable space without oscillation. Compared to CCD, our style-preserving technique drops the box into the bin without an awkward sideways approach to the bin.

The motivation of this work is to preserve continuity of a trajectory and re-use the information content contained in a trajectory to satisfy constraints. Preservation of continuity is essential because typical "natural" motions are characterized by continuity in jerk, acceleration, and velocity [16]. Introduction of discontinuities into the trajectory or one of its derivatives by a motion modification algorithm is undesirable, unless the discontinuities are intentionally introduced for reasons such as holds to draw attention.

We use the smoothness metric in Eq. (3) to compare our technique to CCD. The visible range of the robot was discretized into 83 constraints (i.e. target end-effector locations) that spanned the range of the visible and reachable space of the hardware. The actual hardware trajectories were

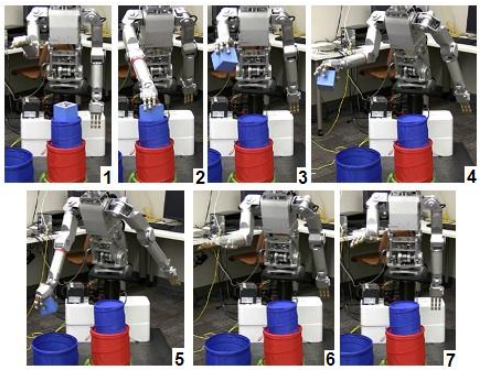

Fig. 5. Trajectory to lift and move a box using our style-preserving technique.

TABLE I

COMPARISON OF THE CONTINUITY METRIC FOR CCD AND OUR STYLE-PRESERVING TECHNIQUE (SN $\times$ 10000)

DOF	CCD	Stylized
Torso Y	2.55	1.12
Torso X	2.39	0.36
Shoulder X	0.78	0.64
Shoulder Z	1.34	0.51
Shoulder Y	0.53	0.19
Elbow X	1.27	1.25
Wrist X	0.34	0.32
Wrist Z	0.27	0.21

recorded for the eight DOFs that participate in the one-armed motion during the random tests. The results for the average of all 83 trajectories are shown in Table I for each DOF. Any DOFs excluded from this table did not have the trajectory modulated for either technique. For all the trajectories created by our style-preserving technique, the sn is lower on average for all DOFs, indicating that robot motion using our style-preserving technique produces motion that is smoother overall when compared to CCD.

D. Baseball Example: Time Varying Constraints

Our technique is also useful in tasks that have time varying constraints. We demonstrate this with simulated robot baseball, where the ball location presents a time varying constraint to a stylized robot swing trajectory. The search space for the constraint is very limited to a small range of Euclidean space because the robot should only swing the bat if the ball is in the strike zone. TSFKS converges very quickly with high accuracy for this constrained search space due to the coarse and fine angle increments being set to very limited ranges. The bat becomes an extension of the kinematic structure at a ninety degree angle to the hand. Both arms are additionally constrained to trajectories dictated by two end-effector constraints which are separated by a constant offset. The radius of the bat is small and therefore, the bat can be represented by a line of finite extent projected onto the trajectory of the pitched ball.

Currently, our technique has only been tested in simulation for robot baseball for straight pitches, where the constant

velocity of the pitch and direction of the ball are known at the moment of release. Assuming the robot is equipped with the sensing capabilities to determine pitch velocity and trajectory (which are separate research problems), figs. 6 and 7 demonstrate the appearance of the swing trajectory after modulation based upon the pitched baseball. figs. 6 and 7 exhibit similar style of the swing despite the batting stance on the opposite side of the plate. Fig. 8 shows a series of time instants when the point constraint is satisfied.

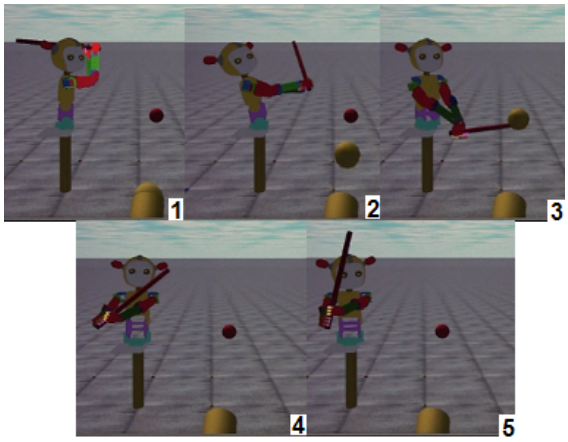

Fig. 6. SIMON simulation using our style-preserving technique to modulate the swing trajectory based upon ball location.

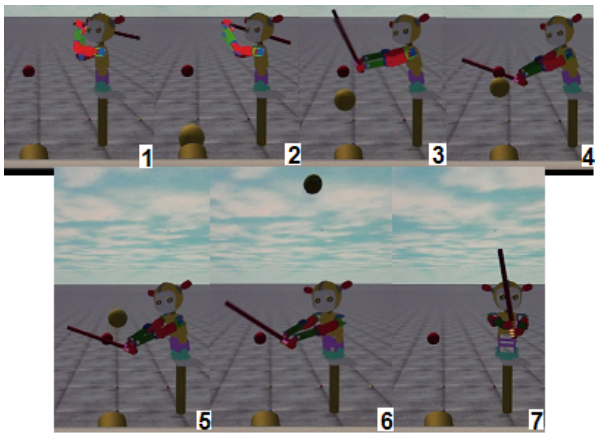

Fig. 7. SIMON simulation using our style-preserving technique to modulate the swing trajectory based upon ball location.

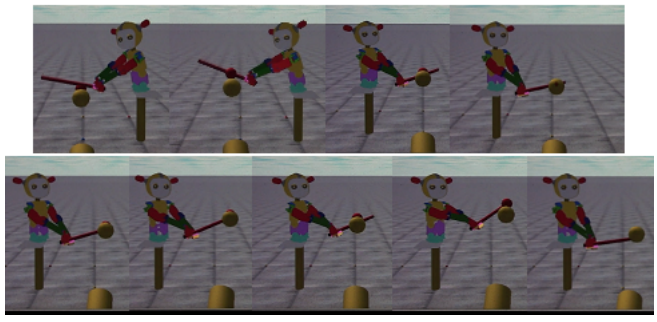

Fig. 8. Frames from different trajectories of the modulated baseball swing showing that the constraint can be satisfied.

VI. Conclusion

In this work, we have shown that an appropriate representation for adapting and generalizing a stylized trajectory is not in Cartesian or joint angle space, but rather in joint velocity space with initial conditions supplied by the interactive environment. This allows the motions to be more readily generalizable and easily applied to actuators without redundancy in the specification. The benefit of this representation is that only one trajectory must be given and it can be extended to accomplish similar tasks in the world by learning only environment constraints. We create useful, task based motion with variance that spans the reachable space of the robot, while satisfying kinematic constraints.

Our representation and technique create motion that is more smooth than other completely autonomous techniques such as CCD, provided that the given instance of the single trajectory being modulated possesses an inherent smooth quality. Unlike CCD, our technique preserves the noticeable *style* of the trajectory while satisfying the constraints, eliminating unexpected or sudden velocity changes, awkward motions, or drastic changes in motion due to subtle changes in end-effector location.

References

[1] A. Sulejmanpasic, and J. Popovic, "Adaptation of performed ballistic motion," in *ACM Transactions on Graphics*, vol. 24, no. 1, pp. 165-179, Jan. 2005.

[2] A. Witkin, and Z. Popovic, "Motion warping," *Proc. SIGGRAPH 95*, Computer Graphics Proceedings, pp. 105-108, 1995.

[3] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[4] A.D. Wilson, and A.F. Bobick, "Parametric hidden markov models for gesture recognition," *IEEE Transactions PAMI*, vol. 21, no. 9, pp. 884-900, Sept. 1999.

[5] K. Perlin, "Real time responsive animation with personality," *IEEE Trans. Visualization and Computer Graphics*, vol. 1, no. 1, pp. 5-15, March 1995.

[6] M. Brand, and A. Hertzmann, "Style machines," *Proc. SIGGRAPH'00*, pp. 183-192, July 2000.

[7] K. Grochow, S.L. Martin, A. Hertzmann, and Z. Popovic, "Style-based inverse kinematics," *Intl. Conf. on Computer Graphics and Interactive Techniques*, pp. 522-531, Los Angeles, CA, 2004.

[8] A. Bruderlin, and L. Williams, "Motion signal processing," *Proc. SIGGRAPH 95*, Computer Graphics Proceedings, pp. 97-104, 1995.

[9] Y. Ye, and C.K. Liu, "Animating responsive characters with dynamic constraints in near-unactuated coordinates," *ACM Trans. On Graphics*, vol. 27, no. 5, article 112, Dec. 2008.

[10] E. Hsu, K. Pulli, and Z. Popovic, "Style translation for human motion," in *Intl. Conf. Computer Graphics and Interactive Techniques*, pp. 1082-1089, Los Angeles, CA, 2005.

[11] S.R. Buss. (2004, April 17) "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," [Online Technical Report], Available: http://math.ucsd.edu/ sbuss/ResearchWeb/ikmethods/iksurvey.pdf

[12] O.C. Jenkins, M.J. Mataric, and S. Weber, "Primitive-based movement classification for humanoid imitation," *Proc. First IEEE-RAS Intl. Conf. Humanoid Robotics*, Cambridge, MA, Sept. 2000.

[13] L. Wang, and C. Chen, "A Combined Optimization Method for Solving the Inverse Kinematics Problem of Mechanical Manipulators, *IEEE Trans. On Robotics and Applications*, vol. 7, no. 4, pp. 489-499, 1991.

[14] J.O. Kim, B.R. Lee, C.H. Chung, J. Hwang, and W. Lee, "The inductive inverse kinematics algorithm to manipulate the posture of an articulated body" *ICCS 2003*, pp. 305-313, 2003.

[15] A. Wouwer, P. Sauscez, and W.E. Schiesser, *Adaptive Method of Lines*, CRC Press, Chapman & Hall, 2001.

[16] J. Laczko, S. Jaric, J. Tihanyi, V.M. Zatsiorsky, and M.L. Latash, "Components of the end-effector jerk during voluntary arm movements," *Journal Applied Biomechanics*, vol. 16, pp. 14-25, 2000.