Task-Aware Variations in Robot Motion

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

***Abstract*— Social robots can benefit from motion variance because non-repetitive gestures will be more natural and intuitive for human partners. We introduce a new approach for synthesizing variance, both with and without constraints, using a stochastic process. Based on optimal control theory and operational space control, our method can generate an infinite number of variations in real-time that resemble the kinematic and dynamic characteristics from the single input motion sequence. We also introduce a stochastic method to generate smooth but nondeterministic transitions between arbitrary motion variants. Furthermore, we quantitatively evaluate task-aware variance against random white torque noise, operational space control, style-based inverse kinematics, and retargeted human motion to prove that task-aware variance generates human-like motion. Finally, we demonstrate the ability of task-aware variance to maintain velocity and time-dependent features that exist in the input motion.**

I. Introduction

We are interested in the problem of motion generation for social robots. When a robot is meant to interact with human partners, as part of its functional goal, we would like the robot's motions to be natural and intuitive for the human partner. Several works posit that human-like qualities in robots will facilitate a more effective interaction [1], [2], [3]. Thus we aim to create human-like motion in robots.

In this paper we address one aspect of human-like motion—variance. Human motion is the result of a huge number of variables (personality, age, gender, culture, environment, intents, etc.). This makes repetitive motion highly unlikely. Thus, for a social robot, the ability to easily produce variants of its exemplar motions or gestures, will be important for generating life-like and natural behavior.

The simplest way to generate variance is injecting white noise into the joint angle trajectories for each degree-of-freedom (DOF). However, applying this technique to robots creates problems. The robot will appear to shake, and these unsmooth trajectories may damage actuators. But we can fix this by applying noise in the torque domain. Another problem is that purposeful motion is disrupted by noise. This issue requires algorithms that induce variations in motion, while respecting the "task" the robot is performing.

We propose an algorithm, task-aware variance (TAV), that generates, in real-time, an infinite number of motion variants from a single robot motion trajectory. We define a task as a set of kinematic constraints and a cost function that measures motion execution (e.g. desirability of joint position, joint velocity, torque usage). To enforce kinematic constraints we employ operational space control (OSC), but OSC alone is not sufficient to produce natural, varied motion. The key insight of our task-aware variance formulation is that an optimal control solution can be used to approximate space curvature locally and the shape of the value function yields insight into the control policy's tolerance to perturbations. Task-aware variance takes advantage of the optimal control solution to select directions in the task space in which injecting noise will minimally disrupt the task.

By combining OSC and TAV, we create variations that are similar in dynamics while still respecting kinematic constraints. For example, we demonstrate motion variance in an "I don't know" gesture created in the null space of a cup holding task. Additionally, task-aware variance easily handles other interesting constraints, like velocity and time constraints. We exemplify this with variance in a synchronized dance task and a motion-speech synchronization task.

In addition to demonstration results, we also evaluate the quality of task-aware variants. We compare against motions from random torque-space white noise, operational space control, style-based inverse kinematics (SIK) and human motion projected onto the robot architecture. The results show task-aware variance is significantly more human-like than two alternatives, and of similar 'human-likeness' to SIK.

II. Related Work

Our work is related to the more general area of human-like motion generation for robots. In some works, natural variance arises from interaction with the environment and preparatory action [4]. And there are many examples of learning human-like motion control policies via demonstrations [5], [6], or observation [7]. Our work focuses on making motion more human-like through variability, and is complementary to these works.

Motion variance can be produced by dependence upon a database [8], [9]. Where, when the system commands a specific motion, one from a set of pre-annotated gestures is selected randomly. In most applications, there is a tradeoff between the richness of variations and the size of dataset.

In general, unless environmental constraints (e.g., obstacles) induce variability, robots have repetitive motion. When motion variance is seen, it comes either via professional animators or motion capture data (e.g., [10]), for which generating new exemplars is costly.

The field of computer animation has explored creating varied motion, for example, by building models of style from

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 and A.L. Thomaz are with the School of Interactive Computing, Georgia Institute of Technology, 85 5th St. NE, TSRB 317, Atlanta, GA, 30308 USA (karenliu@cc.gatech.edu, athomaz@cc.gatech.edu)

human motion capture data for subsequent interpolation to generate new motions [11]. Or, given an input sequence, introducing random noise in the joint trajectories [12], [13]. These methods can be computed efficiently in real-time applications, but the results depend on careful parameter tuning for the stochastic models. Additionally, these methods cannot produce variations for purposeful motion, where constraints must be preserved. Other techniques that rely upon direct manipulation of trajectories are not applicable because they require manual intervention, which affects online application [14], [15], [16].

III. Overview

We develop an algorithm to produce variations of an input motion while maintaining the robot's task. Our approach induces biased noise in the torque domain based on both kinematic constraints and a cost function derived from the input motion. Inspired by optimal control theory and OSC, our algorithm first adds the biased torque to the optimal control torques according to a defined cost function, and then projects the "corrupted" torque to the null space of kinematic constraints. The resultant torque preserves the characteristics of the input motion encoded in the cost function and kinematic constraints, but stochastically produces motion that is visually different from the input.

Given a sequence of poses $\mathbf{q}_t,\ 0 \leq t \leq N$ that constitute a motion, we construct a reference state trajectory $\bar{\mathbf{x}}$, and its corresponding reference control trajectory $\bar{\mathbf{u}}$. The state trajectory consists of both $\mathbf{q}_t$ and $\dot{\mathbf{q}}_t$, while the control trajectory $\bar{\mathbf{u}}$ consists of joint torques computed from an inverse dynamics process. Kinematic constraints to be satisfied $\mathbf{C}$ can also be specified. The core of our algorithm computes a time-varying multivariate Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_t)$, constraint projection matrices $\mathbf{P}_t$, and a feedback control policy, $\Delta\mathbf{u}_t = -\mathbf{K}_t\Delta\mathbf{x}_t$. The characteristics of the input motion are represented by the Gaussian and the feedback policy (Sec. IV), while the kinematic constraints are preserved by the projection matrices (Sec. V).

Variation for an input motion is generated online by applying the following operations at each time step.

1) Draw random Gaussian sample: $\Delta\mathbf{x}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_t)$
2) Compute the corresponding control force via the feedback control policy: $\Delta\mathbf{u}_t = -\mathbf{K}_t\Delta\mathbf{x}_t$
3) Project the control force to enforce kinematic constraints: $\Delta\mathbf{u}_t^* = \mathbf{P}_t\Delta\mathbf{u}_t$
4) Apply $\bar{\mathbf{u}}_t + \Delta\mathbf{u}_t^*$ as the current control force

Our approach can integrate with techniques that organize multiple motions into graphs or state-machines. As a practical issue, we describe a method to generate nondeterministic transitions between arbitrary motion sequences (Sec. VI).

IV. Task-aware Variance

Given an input reference state and control trajectory, $\bar{\mathbf{x}}_t$ and $\bar{\mathbf{u}}_t$, respectively, $t = 1 \cdots N$, we formulate an optimization that tracks the reference trajectory. $\bar{\mathbf{x}}_t \in \Re^{2n}$ contains the joint angles and velocity at frame t while $\bar{\mathbf{u}}_t \in \Re^m$ contains the joint torques. The task can be viewed as minimizing the state and control deviation from the reference trajectory, subject to discrete-time dynamic equations.

$$\begin{aligned} \min_{\mathbf{x},\mathbf{u}} \quad & \frac{1}{2}\|\mathbf{x}_N - \bar{\mathbf{x}}_N\|^2_{\mathbf{S}_N} + \\ & \sum_{t=0}^{N-1} \frac{1}{2}(\|\mathbf{x}_t - \bar{\mathbf{x}}_t\|^2_{\mathbf{Q}_t} + \|\mathbf{u}_t - \bar{\mathbf{u}}_t\|^2_{\mathbf{R}_t}) \\ \text{subject to} \quad & \mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t) \end{aligned} \tag{1}$$

where $\mathbf{S}_N$, $\mathbf{Q}_t$, and $\mathbf{R}_t$ are positive semidefinite matrices that indicate the weight between different objective terms. (Throughout, we use the shorthand $\|\mathbf{x}\|^2_{\mathbf{Y}} = \mathbf{x}^T\mathbf{Y}\mathbf{x}$.) We will discuss how these matrices are determined shortly. For an optimal control problem, it is convenient to define an optimal value function, $v(\mathbf{x}_t)$ measuring the minimal total cost of the trajectory from state $\mathbf{x}_t$. It can be written recursively:

$$v(\mathbf{x}_t) = \min_{\mathbf{u}} \frac{1}{2}(\|\mathbf{x}_t - \bar{\mathbf{x}}_t\|^2_{\mathbf{Q}_t} + \|\mathbf{u}_t - \bar{\mathbf{u}}_t\|^2_{\mathbf{R}_t}) + v(\mathbf{x}_{t+1}) \tag{2}$$

This function is the key to Bellman's optimality principle. If we can evaluate the value function, we can readily define an optimal policy that maps a state to an optimal action: $\Pi : X \rightarrow U(X)$.

While our work is not on optimal control, we use the optimal value function to derive probabilistic models for generating motion variance. Our key insight is that the shape of this value function reveals information about the tolerance of the control policy to perturbations. With this, we can choose a perturbation that causes minimal disruption to the task while inducing visible variation to the reference motion.

However, the optimal value function is usually very difficult to solve for a nonlinear problem. One exception is the linear-quadratic-regulator (LQR) which has a linear dynamic equation and a quadratic cost function. A common practice in approximating a nonlinear dynamic tracking problem is to linearize the dynamic equation around the reference trajectory and substitute the variables with the deviation from the reference, $\Delta\mathbf{x}$ and $\Delta\mathbf{u}$. The original optimization can be reformulated to:

$$\begin{aligned} \min_{\Delta\mathbf{x},\Delta\mathbf{u}} \quad & \frac{1}{2}\|\Delta\mathbf{x}_N\|^2_{\mathbf{S}_N} + \sum_{t=0}^{N-1} \frac{1}{2}(\|\Delta\mathbf{x}_t\|^2_{\mathbf{Q}_t} + \|\Delta\mathbf{u}_t\|^2_{\mathbf{R}_t}) \\ \text{subject to} \quad & \Delta\mathbf{x}_{t+1} = \mathbf{A}_t\Delta\mathbf{x}_t + \mathbf{B}_t\Delta\mathbf{u}_t \end{aligned} \tag{3}$$

where $\mathbf{A}_t = \frac{\partial f}{\partial \mathbf{x}}|_{\bar{\mathbf{x}}_t,\bar{\mathbf{u}}_t}$ and $\mathbf{B}_t = \frac{\partial f}{\partial \mathbf{u}}|_{\bar{\mathbf{x}}_t,\bar{\mathbf{u}}_t}$. The primary reason to approximate our problem with a time-varying LQR formulation is that the optimal value function can be represented in quadratic form with time-varying Hessians.

$$v(\Delta\mathbf{x}_t) = \frac{1}{2}\|\Delta\mathbf{x}_t\|^2_{\mathbf{S}_t} \tag{4}$$

where the Hessian matrix $\mathbf{S}_t$ is a symmetric matrix. At time step t, the optimal value function is a quadratic function centered at the minimal point $\bar{\mathbf{x}}_t$. Therefore, the gradient of the optimal value function at $\bar{\mathbf{x}}_t$ vanishes, while the

Hessian is symmetric, positive semidefinite, and measures the curvatures along each direction in the state domain. A deviation from $\bar{\mathbf{x}}_t$ along a direction with high curvature causes large penalty in the objective function and is considered inconsistent with the task. For example, the perturbation in the direction of the first eigenvector (the largest eigenvalue) of the Hessian induces the largest total cost of tracking the reference trajectory. We use this bias to induce more noise in the dimensions consistent with the tracking task. We draw random samples from a zero-mean Gaussian with a time-varying covariance matrix defined as the inverse of the Hessian: $\mathcal{N}(\mathbf{0}, \mathbf{S}_t^{-1})$. The matrices $\mathbf{S}_t$ can be efficiently computed by the Riccati equation, which exploits backward recursive relations starting from the weight matrix at the last frame $\mathbf{S}_N$. We omit the subscript t on $\mathbf{A}$, $\mathbf{B}$, $\mathbf{Q}$, and $\mathbf{R}$ for clarity (detailed derivation in [17]).

$$\mathbf{S}_t = \mathbf{Q} + \mathbf{A}^T\mathbf{S}_{t+1}\mathbf{A} - \mathbf{A}^T\mathbf{S}_{t+1}\mathbf{B}(\mathbf{R} + \mathbf{B}^T\mathbf{S}_{t+1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{S}_{t+1}\mathbf{A} \tag{5}$$

The random sample, $\Delta\mathbf{x}_t$, drawn from the Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{S}_t^{-1})$ indicates deviation from the reference state trajectory $\bar{\mathbf{x}}_t$. Directly applying this "task-aware" deviation to joint trajectories will cause vibration. Instead, our algorithm induces noise in control space via the feedback control policy derived from LQR: $\Delta\mathbf{u}_t = -\mathbf{K}_t\Delta\mathbf{x}_t$. In our discrete-time, finite-horizon formulation, the feedback gain matrix $\mathbf{K}_t$ is a $m \times 2n$ time-varying matrix computed in closed-form:

$$\mathbf{K}_t = (\mathbf{R} + \mathbf{B}^T\mathbf{S}_{t+1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{S}_{t+1}\mathbf{A} \tag{6}$$

a) Singular Hessians: Occasionally, the Hessians of the optimal value function become singular. In this case, we apply singular value decomposition on the Hessian to obtain a set of orthogonal eigenvectors $\mathbf{E}$ and eigenvalues $\sigma_1 \cdots \sigma_n$ (because $\mathbf{S}_t$ is always symmetric). For each eigenvector $\mathbf{e}_i$, we define a one-dimensional Gaussian with zero mean and a variance inversely proportional to the corresponding eigenvalue: $N_i(0, \frac{1}{\sigma_i})$. For those eigenvectors with zero eigenvalue, we simply set the variance to a chosen maximal value (e.g., the largest eigenvalue of the covariance matrix in the entire sequence). The final sample $\Delta\mathbf{x}_t$ is a weighted sum of eigenvectors: $\Delta\mathbf{x}_t = \sum_i w_i\mathbf{e}_i$, where w_i is a random number drawn from N_i.

b) Determine weight matrices: The cost weight matrices, $\mathbf{S}_N$, $\mathbf{Q}_t$ $\mathbf{R}_t$, can be selected manually based on prior knowledge of the input motion and control. Intuitively, when a joint or actuator is unimportant we would like to assign a small value to the corresponding diagonal entry in these matrices. Likewise, when two joints are moving in synchrony, we might want to give them similar weights. $\mathbf{Q}$ and $\mathbf{R}$ in theory can vary over time, but most practical controllers hold $\mathbf{Q}$ and $\mathbf{R}$ fixed to simplify the design process. Still, when the robot has a large number of DOFs, tuning these weights manually can be difficult.

We propose a method to automatically determine the cost weights based on coordination in the reference trajectory. We apply Principal Component Analysis (PCA) on the reference motion $\bar{\mathbf{x}}$ and on the reference control $\bar{\mathbf{u}}$ to obtain respective sets of eigenvectors $\mathbf{E}$ and eigenvalues $\mathbf{\Sigma}$ (in diagonal matrix form). The weight matrix for motion can be computed by $\mathbf{Q} = \mathbf{E}\mathbf{\Sigma}\mathbf{E}^T$. By multiplying $\Delta\mathbf{x}$ on both sides of $\mathbf{Q}$, we effectively transform the $\Delta\mathbf{x}$ into eigenspace, scaled by the eigenvalues. As a result, $\mathbf{Q}$ preserves the coordination of joints in the reference motion, scaled by their importance. $\mathbf{R}$ can be computed in the same way. In our implementation, we set $\mathbf{S}_N$ equal to $\mathbf{Q}$.

V. Kinematic constraints

In addition to maintaining characteristics of the input motion, we also want variance that adheres to kinematic constraints. To do this we use an approach first introduced for controlling robots to achieve simultaneous tasks [18].

At each iteration, we define a projection matrix $\mathbf{P}_t$ that maps the variation in torque, $\Delta\mathbf{u}_t$, to the appropriate control torque that does not interfere with the given kinematic constraint. Suppose a constraint is specified to enforce a point $\mathbf{p}$ on the hand to be at a fixed coordinate, we can express the Jacobian of the constraint by $\mathbf{J}_t = \frac{\partial\mathbf{p}}{\partial\mathbf{q}_t}$. Intuitively, the Jacobian maps the Cartesian force required to maintain $\mathbf{p}$ to a joint torque: $\tau = \mathbf{J}_t^T\mathbf{f}$. Following Khatib's formulation, we define a projection matrix $\mathbf{P}_t$ as follows:

$$\mathbf{P}_t = \mathbf{I} - \mathbf{J}_t^T\bar{\mathbf{J}}_t^T \tag{7}$$

where $\bar{\mathbf{J}}_t$ is one of the many pseudo inverse matrices of $\mathbf{J}$. We use the "dynamically consistent generalized inverse":

$$\bar{\mathbf{J}}_t^T = \mathbf{\Lambda}_t\mathbf{J}_t\mathbf{M}_t^{-1} \tag{8}$$

where $\mathbf{\Lambda}_t$ and $\mathbf{M}_t$ are the current inertia matrix in Cartesian space and in joint space. When we apply the projection matrix $\mathbf{P}_t$ to a torque vector, it removes the component in the space spanned by the columns of $\bar{\mathbf{J}}_t$, where the variation will directly affect the satisfaction of the constraint. Consequently, the final torque variation $\Delta\mathbf{u}_t^* = \mathbf{P}_t\Delta\mathbf{u}_t$ applied to the robot will maintain the kinematic constraints. Our algorithm can achieve a variety of tasks by setting kinematic constraints in Cartesian space or in joint space. For example, holding a cup, or pointing at an object.

VI. Nondeterministic transitions

In this section, we describe an algorithm that enables our technique to transition between multiple motion sequences. We demonstrate that it is possible to transition to the next desired motion from a wide range of states. Further, to make the transition reflect natural variance as well, we stochastically select a transition-to pose online. We call it *nondeterministic transition* because the transition motion to the same sequence is different each time.

Once the next motion is selected, our algorithm selects a transition-to pose $\mathbf{x}_0^*$ via a stochastic process, such that the robot does not always predictably transition to the first frame of the next motion. This can be viewed as a task-aware deviation from the first frame of the next motion $\bar{\mathbf{x}}_0$.

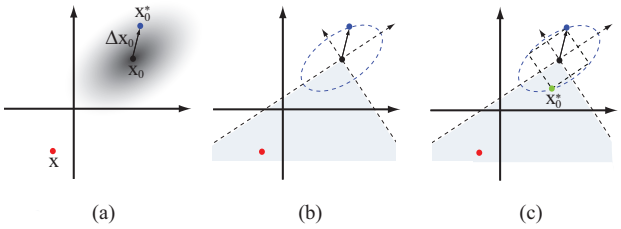

Fig. 1. (a): Setting the transition-to pose to $\mathbf{x}_0^* = \bar{\mathbf{x}}_0 + \Delta\mathbf{x}_0$, can generate an awkward transition when $\mathbf{x}_0^*$ is further away from $\mathbf{x}$ than from $\bar{\mathbf{x}}_0$. (b): Poses with the same likelihood as $\bar{\mathbf{x}}_0 + \Delta\mathbf{x}_0$ form a hyper-ellipsoid. (c): $\Delta\mathbf{x}_0$ defines a hypercube aligned with the eigenvectors of the covariance matrix. We pick $\mathbf{x}_0^*$ as the corner that lies in the quadrant of $\mathbf{x}$.

We reuse the Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{S}_0^{-1})$ computed for the next motion to get a random sample $\Delta\mathbf{x}_0$. If we directly set the transition-to pose to $\mathbf{x}_0^* = \bar{\mathbf{x}}_0 + \Delta\mathbf{x}_0$, it could generate an awkward transition when $\mathbf{x}_0^*$ is further away from the current pose $\mathbf{x}$ than from $\bar{\mathbf{x}}_0$ (Fig. 1 (a)).

To overcome this issue, we account for the current state $\mathbf{x}$ when selecting the transition-to pose $\mathbf{x}_0^*$. Because $\Delta\mathbf{x}_0$ is drawn from a symmetric distribution, poses with the same likelihood form a hyper-ellipsoid. To bias $\mathbf{x}_0^*$ toward the $\mathbf{x}$, we want to pick a state from this hyper-ellipsoid that lies in the same quadrant in the coordinates defined by the eigenvectors of the covariant matrix $\mathbf{S}_0^{-1}$ (Fig. 1 (b)). To speed up computation, we use $\Delta\mathbf{x}_0$ to define a hypercube aligned with the eigenvectors. We pick $\mathbf{x}_0^*$ to be the corner within the same quadrant as $\mathbf{x}$ (Fig. 1 (c)).

After determining the transition-to pose, we use a spline interpolation and PID-tracking to move the robot to this pose from the current pose. This works well because the transition-to pose is both consistent with the next task and biased toward the current pose.

VII. Alternative Approaches for Comparison

In the remainder of this paper, we analyze results of TAV. In order to do so, we consider three alternative algorithms.

A. Random White Torque Space Noise (RWTSN)

RWTSN is a naïve variance approach that generates smooth motion by allowing unbiased torque noise to alter trajectories without intelligent shaping. Results show that purposeful motion is distorted by this random noise.

B. Operational Space Control + RWTSN

In this approach, step two in Sec. III is replaced by $\Delta\mathbf{u}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and steps three and four of the TAV algorithm remain unchanged. For point constraints and partial posture constraints, white noise torques projected into the null space of the robot architecture creates one of two outcomes: task disruption or lack of naturalness. This is particularly clear in the video accompanying this paper. [1]

[1] http://www.cc.gatech.edu/social-machines/projects

C. Style-based Inverse Kinematics

We implemented a basic form of style-based inverse kinematics, in which human motion capture data is represented in a low dimensional space. Motion variants are generated by learning probability density functions of the data and interpolating in the low-dimensional space [19]. The disadvantages of SIK are that variance is limited by the database and the variants generated may appear unnatural when interpolating between drastically different exemplars.

VIII. Research Platform

The platform for this research is an upper-torso humanoid robot, Simon (Figure 2). Each arm has seven DOFs, three at the shoulder, one at the elbow, and three at the wrist. Each hand has four DOFs. The torso has two DOFs, with an additional uncontrollable joint in the torso fore/aft direction. It has three DOFs for the eyes, two per ear, and four for the neck. The robot operates on a dedicated ethercat network coupled with a real-time PC operating at 1kHz.

IX. Analysis of Resulting Motion

Our results demonstrate that the task-aware variance approach is able to produce variance in unconstrained gestures, motions with physical task constraints, and is able to maintain time-dependent and velocity features of a task.

A. Variance in Unconstrained Gestures

In the unconstrained case, we demonstrate the capability of our algorithm through two common social robot gestures: waving and an "I don't know" gesture. A task with no constraint has $\mathbf{P}_t = I$ in step 3 of Sec. III.

To characterize TAV generated variation, we measure variance from the original motion of 12 generated sequences:

$$\frac{1}{K}\frac{1}{N}\sum_{i=1}^{K}(\bar{\mathbf{q}} - \mathbf{q}_i)^T(\bar{\mathbf{q}} - \mathbf{q}_i) \tag{9}$$

where $\mathbf{q}_i$ is the trajectory of a particular DOF from variant i, $\bar{\mathbf{q}}$ is the original motion, K is the number of variants, and N is the number of time samples in the motion.

For this analysis, only the left arm DOFs are evaluated. This is valid since we use a left hand waving gesture, and the input "I don't know" gesture is symmetric. Thus the left arm is sufficient to characterize the output of a variance-generating algorithm with these inputs. Table I shows that variance is created in different DOFs for different motions. This shows that TAV produces motions different from the original, but it does not show that variants are different from one another. To verify this, we recompute Equation 9, replacing the original motion with a variant from the set, which gives Equation 10.

$$\frac{1}{K^2}\frac{1}{N}\sum_{i=1}^{K}\sum_{j=1}^{K}(\mathbf{q}_j - \mathbf{q}_i)^T(\mathbf{q}_j - \mathbf{q}_i) \tag{10}$$

The results in Table II show that on average the variants generated by TAV are different from each other, especially in the "I don't know" gesture.

TABLE I

VARIANCE FROM THE ORIGINAL MOTION (IN SQUARE DEGREES). AVERAGES TAKEN OVER FIRST 12 TAV GENERATED MOTIONS.

DOF	"Waving"	"I don't know"
shoulder X	364	974
shoulder Z	170	751
shoulder Y	1198	1238
elbow X	185	5993
wrist Y	649	5354
wrist X	664	1742
wrist Z	177	834

TABLE II

VARIANCE BETWEEN TAV-GENERATED VARIANTS (IN SQ. DEGREES). AVERAGES TAKEN OVER THE FIRST 12 TAV GENERATED MOTIONS.

DOF	"Waving"	"I don't know"
shoulder X	294	3740
shoulder Z	90	1078
shoulder Y	2143	2902
elbow X	1627	2846
wrist Y	704	643
wrist X	1470	235
wrist Z	510	202

TAV is capable of generating infinite variants of a motion in the absence of constraints. The accompanying video shows the original motions and three variants generated for both the waving and "I don't know" gestures. An example of the range of variance produced is shown in Figure 2.

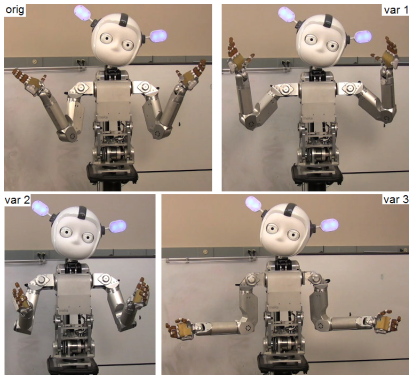

Fig. 2. "I don't know" motion. Upper-left is original motion. Other three are the inflection point in three variants generated by TAV.

B. Producing Variance with Constraints

We also demonstrate that our algorithm can preserve kinematic constraints by applying both TAV and OSC. In the example shown in the accompanying video, the robot's left hand is constrained with an orientation and position constraint so that a cup of water will not spill.

To measure task-aware variance in the presence of constraints, we again adapt the standard formula for variance. We analyze the DOFs on the left arm that hold the cup, because the variance of these DOFs provides more insight into how constraints affect the task space. We first apply OSC to constrain the original motion and use it as the mean to compute the variance (Eq. 9). Results are in Table III.

TABLE III

LEFT ARM JOINT ANGLE VARIANCE (IN SQ. DEGREES), FROM THE CONSTRAINED ORIGINAL AND THE OTHER VARIANTS. AVERAGES USE THE FIRST 12 TAV GENERATED MOTIONS FOR "I DON'T KNOW".

DOF	w.r.t. orig	between variants
shoulder X	116	354
shoulder Z	271	133
shoulder Y	298	599
elbow X	581	846
wrist Y	560	1132
wrist X	22	189
wrist Z	44	124

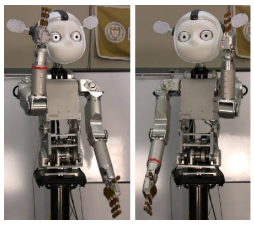

Fig. 3. Representative poses of the two concatenated dancing input motions.

As expected, physical constraints reduce variance overall (Table III). This is especially true closer to the constrained DOF, which for the cup constraint is the wrist for the x and y degrees-of-freedom. This effect is also evident in the calculations that reference the other variants (i.e. the right-hand side of Table III). However, note that average variance between variants is still large for the left arm DOFs. The video is the best example of these effects, but from our analysis we can conclude that TAV produces variance in constrained motion, if a null space for the constraints exists.

C. Maintains Velocity and Time-Dependent Task Features

If the task-aware variance technique creates variance from an exemplar that is already human-like, naturalness is maintained due to the cost function in the optimal control solution. This unique framework provides a simple way to maintain time-dependent task features by modulating the cost function weights. A time-dependent feature enforces an arbitrary equation on joint position or velocity at a particular time. For example, a time-dependent feature could enforce the velocity of the left hand at frame 10. We demonstrate with two examples: dancing and motion-speech synchronization.

1) Dancing: Although there are many ways to define the task of dancing, we define it as moving the body rhythmically to the same beat as the original motion. The specific task goal is to maintain the same joint velocity as the input motion at the moments when the joint velocity is zero. This is an example of time-dependent and velocity features of a task. As a result, our algorithm produces variants that synchronize with the original dancing motion. Additionally, we make eye gaze a part of the dancing task, a time-varying eye direction constraint, that is enforced by OSC.

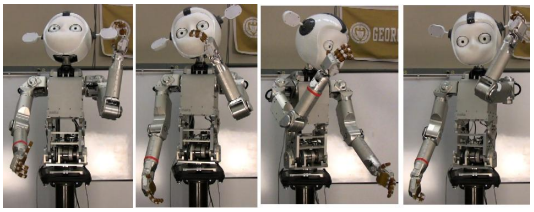

Fig. 4. Representative poses from the varied dancing produced by TAV. These poses are taken at the time-dependent feature points during the task.

In our implementation, two input dance moves are concatenated together. In the first primitive motion, one arm moves up as the other moves down and the robot turns to the side. The other motion is a similar, but all DOFs move in the opposite direction. Figure 3 shows two representative poses from the primitives. When concatenated, they create a cyclic, repetitive dance sequence. For our task, the features we want to maintain are the zero velocity points, when one primitive transitions to the other. This is time-dependent because it occurs only twice during one cycle of the dance.

To handle time-dependent task features with TAV, the penalty against violating all important features increases closer to the point in time when those features must be maintained. In the dancing task, synchronization of original and variant motions can be achieved by introducing very high weights for all DOFs zero velocity points. The reason that TAV can preserve velocity and time-dependent task features is because the optimal control policy accounts for the cost of the entire trajectory; the cost of a future state will affect the action of the current state. On the other hand, OSC does not consider the long-term goal of the motion when enforcing a constraint. If we set a constraint that enforces zero joint velocity at a future frame, OSC will stop the joint motion abruptly at that frame without anticipation.

The result shown in the accompanying video demonstrates that the task-aware dancing changes velocity in sync with the primitives from which it was generated. Results for TAV are shown in simulation with the eye gaze constraint, and on hardware without the eye gaze constraint. Contrast the varied poses in Figure 4 against the input motions shown in figure 3 to see the variance induced by TAV.

2) Timing of Gestures to Speech: As our second illustrating example, we chose one that is a common occurrence for social robots: synchronizing speech to gesture. Although there are many ways to define the task of synchronizing motion to speech, we define the task as ensuring an emphasized word in a phrase is spoken at the same instant in time as the zero velocity point in an accompanying gesture. We demonstrate this with an "I don't know" gesture. Our goal is to prevent the added variance from disrupting the synchronized points such that the same speech can be synchronized with varying gestures.

For comparison, we implemented a basic version of style-based inverse kinematics, a computer animation technique which builds a low-dimensional model from human input motions, and interpolates the space of motions to generate new motions. The exemplars produced through this approach are realistic because they are based on human input data. However, interpolation does not guarantee that the task features such as zero velocity points will align in different exemplars unless preprocessing of the input data occurs to warp the timing of all motions.

Similar to the dancing example, TAV can maintain the synchronized gestures by setting high cost for deviation at the moments of emphasized words. Unlike interpolation, TAV only requires one representative exemplar in order to create infinite variants, which removes the tedious process of warping and aligning a database of motions.

The accompanying video shows speech and gesture synchronized according to emphasized words. For example, "I don't know" versus "I don't know". With task-aware variance, the gesture can be varied autonomously, but the timing of the zero-velocity point still remains deterministic and predictable, thereby allowing synchronization of speech and varied motion simultaneously.

X. Evaluation

In the previous section we characterized the types of motion variance that TAV can produce. In this section we want to prove that TAV produces resultant motions that are as human-like as the input motion. In other words, that it does not corrupt human-like characteristics of the motion.

A. Experimental Design

In order to prove that task-aware variance produces human-like variants, we need an objective, comparative evaluation against human motion. We collected a data set of "waving" and "I don't know" gestures from 24 different humans with motion capture equipment. We then projected each of these 48 motions onto the robot architecture using an optimization that calculates Simon's joint angle values as a function of time from the 28 upper-body position constraints collected from human motion capture markers and a similar set of constraints positioned on the robot body. The optimal mapping allows for proportional scaling the overall size of Simon based on a human subject's size. This procedure creates a motion trajectory that the robot can execute [20].

This data set of 48 human motions projected onto the robot hardware is the `human-like` data set, a ground truth for human-like motion. We compare motions generated from our technique and three alternative variance inducing algorithms, yielding the following four test cases:

- Task-Aware Variance (TAV),
- Random white torque space noise (RWTSN),
- Operational space control+random torque noise. Random torque noise is projected onto the robot body after applying a two time-varying Cartesian end-effector point constraints (one per hand) calculated from the average of all the human motion-capture data.
- Style-based inverse kinematics (SIK). The implementation details for style-based IK can be found in [19].

For each of the four test cases we generate 20 variants of both "waving" and "I don't know" gestures (40 total).

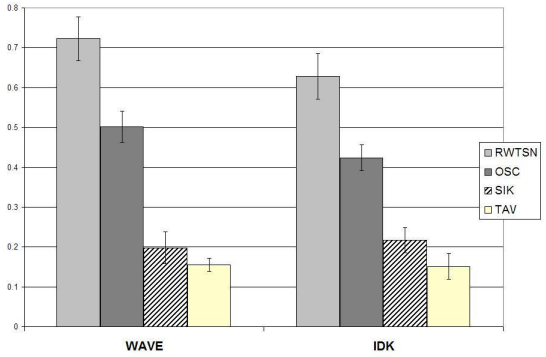

Fig. 5. Average distance to nearest neighbor in human-like set. Techniques described in Sec. VII. T-tests results for "wave" and "I don't know" gestures. TAV is statistically different from RWTSN and OSC in both cases. TAV and SIK are not significantly different in this measure.

TABLE IV

AVG. DISTANCE OF RESULTANT MOTIONS FROM THE FOUR VARIANCE TECHNIQUES TO ITS NEAREST NEIGHBOR IN THE HUMAN DATA. RESULTS ARE AN AVERAGE OF 20 VARIANTS; VALUES ARE JOINT ANGLES EUCLIDEAN DISTANCES IN RADIANS.

	RWTSN	OSC	SIK	TAV
Waving Mean	0.723	0.502	0.198	0.155
Waving (SD)	(0.246)	(0.176)	(0.178)	(0.074)
I don't know Mean	0.628	0.424	0.217	0.151
I don't know (SD)	(0.254)	(0.146)	(0.142)	(0.144)

We then use nearest-neighbor, with a Euclidean distance metric in joint angle space, to find the minimum distance between the technique-generated variant and a trajectory in the `human-like` set. We assume that distance to a `human-like` trajectory is a good metric for human-like quality of a given motion. For this analysis, all techniques were constrained to robot joint angle limits, i.e., torque noise was not applied to any DOF which would force it to exceed a joint angle limit. Additionally, certain DOFs (eyes, eyelids, and ears) were excluded from the analysis because they are not measurable with the motion capture equipment.

B. Task-Aware Variance Creates Human-like Motion

Table IV shows results for "waving" and "I don't know" gestures. The means are averages for all times during the trajectory, for all DOFs, over all 20 motion variants. For the style-based IK technique, cross validation was performed ten times using random sets of 12 test exemplars and 12 training exemplars. Table IV shows the cross validation averages.

Table IV shows that for two common social robot gestures, OSC and random torque noise are less human-like than task-aware variance and style-based IK. As shown in Figure 5, paired t-tests showed statistical significance between all techniques ($p<0.01$), except TAV and SIK. This suggests that TAV provides the variance benefits of a high-quality technique like style-based IK, without all the preprocessing steps necessary for model synthesis. Unlike SIK, which requires dozens of input exemplars, TAV produces motion without training a model, using one exemplar.

XI. CONCLUSION

We aim to create human-like motion in robots. In this paper we address the problem of creating variability in gestures for a robot. We present task-aware variance, an autonomous algorithm capable of generating an infinite number of human-like motion variants both with and without constraints, in real-time using only one exemplar. We also demonstrate the capability of task-aware variance to handle time-dependent and velocity constraints such as synchronization. Finally, we show that our technique creates more human-like motion variants than two other algorithms.

REFERENCES

[1] M. Riley *et al.*, "Methods for motion generation and interaction with a humanoid robot: Case studies of dancing and catching." *AAAI and CMU Workshop on Interactive Robotics and Entertainment*, April 2000.

[2] S. Kopp and I. Wachsmuth, "A knowledge-based approach for lifelike gesture animation." in *ECAI 2000 - Proceedings of the 14th European Conference on Artificial Intelligence*. IOS Press, 2000, pp. 663–667.

[3] J. Cassell, "Embodied conversational agents: Representation and intelligence in user interfaces." *AI Magazine.*, vol. 22, no. 4, pp. 67–84, 2001.

[4] L. Y. Chang, S. S. Srinivasa, and N. S. Pollard, "Planning pre-grasp manipulation for transport tasks," *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2010.

[5] J. Butterfield, O. C. Jenkins, D. Sobel, and J. Schwertfeger, "Modeling aspects of Theory of Mind with Markov Random Fields," *International Journal of Social Robotics*, vol. 1, no. 1, pp. 41–51, Jan 2009.

[6] A. Billard and S. Schaal, "Robust learning of arm trajectories through human demonstration," in *Proceedings 2001 IEEE/RSJ Intl. Conf. Intelligent Robots and Systems*, October 2001, pp. 734–739.

[7] L. D. O. C. Kulic, D. and Y. Nakamura, "Incremental learning of full body motion primitives for humanoid robots," *IEEE-RAS International Conference on Humanoid Robots*, 2008.

[8] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *Intl Journal of Robotics Research*, vol. 20, no. 5, p. 378400, 2001.

[9] K. J. Yamane, K. and J. Hodgins, "Synthesizing animations of human manipulation tasks," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 532–539, 2004.

[10] T. Asfour and R. Dillmann, "Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem," *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS 2003)*, pp. 1407–1412, October 2003.

[11] W. Ma *et al.*, "Modeling style and variation in human motion," *The ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA 2010)*, 2010.

[12] K. Perlin, "Real time responsive animation with personality," *IEEE Transactions on Visualization and Computer Graphics*, vol. I, no. 1, July 1995.

[13] A. Egges, T. Molet, and N. Magnenat-Thalmann, "Personalised real-time idle motion synthesis," in *Pacific Graphics*, 2004, pp. 121–130.

[14] Z. Popović and A. Witkin, "Physically based motion transformation," in *SIGGRAPH*, Aug. 1999, pp. 11–20.

[15] A. Witkin and Z. Popović, "Motion warping," in *SIGGRAPH*, Aug. 1995.

[16] C. K. Liu, A. Hertzmann, and Z. Popović, "Learning physics-based motion style with nonlinear inverse optimization," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 24, no. 3, pp. 1071–1081, July 2005.

[17] F. L. Lewis and V. L. Syrmos, *Optimal Control*. Wiley-Interscience, 1995.

[18] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 1, pp. 43–53, Feb 1987.

[19] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović, "Style-based inverse kinematics," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 23, no. 3, pp. 522–531, July 2004.

[20] J. Yang *et al.*, "Capturing and analyzing of human motion for designing humanoid motion," in *International Conference on Information Acquisition*, June/July 2005, pp. 332–337.