Tendon-Driven Control of Biomechanical and Robotic Systems: A Path Integral Reinforcement Learning Approach.

Eric Rombokas, Evangelos Theodorou, Mark Malhotra, Emo Todorov and Yoky Matsuoka

Abstract— **We apply path integral reinforcement learning to a biomechanically accurate dynamics model of the index finger and then to the Anatomically Correct Testbed (ACT) robotic hand. We illustrate the applicability of Policy Improvement with Path Integrals (PI^2) to parameterized and non-parameterized control policies. This method is based on sampling variations in control, executing them in the real world, and minimizing a cost function on the resulting performance. Iteratively improving the control policy based on real-world performance requires no direct modeling of tendon network nonlinearities and contact transitions, allowing improved task performance.**

I. INTRODUCTION

We demonstrate control and learning of tendon driven bio-mechanical and robotic systems, under difficult-to-model real-world conditions. This work is part of a bigger project in which the ultimate goal is twofold. First, we aim to use biologically inspired control and design principles to improve the state of the art robot control and design. Secondly, we seek to better understand the underlying computational principles of neural and bio-mechanical systems.

Although there have been a number of studies of neural motor control and robotics, there still remains much progress to be made in bridging the gap between these two areas. Most studies are limited to applications of control algorithms to simulation, due to the difficulty of interfacing with real-world robotic hardware and biological motor systems. One of the main reasons for this discrepancy between simulations and real systems is that tendon-driven systems are very complex. In tendon driven systems, torque around the joints is created through a network of tendons attached to the links. These tendons produce only positive force since they must pull and not push. Nonlinearities due to friction and control constraints contribute to the complexity of the underlying robotic dynamics. Control and reinforcement learning algorithms which perform well in simulation may not perform well on real robotic systems due to factors like dependence on accurate models and the "Curse of Dimensionality."

A promising strategy for overcoming the complexity of robotic tendon driven control is to avoid modeling it directly, but instead to learn a set of controls which achieves success by actually trying them. Beginning with a single example or demonstration, reinforcement learning can be applied by iteratively minimizing a cost function on the outcome of sample trials. This strategy, then, is to explore variations of control, observe the outcome of using that control, and revise the controller accordingly.

Recent work on path integral reinforcement learning [1] has demonstrated the robustness and scalability of the method to robotic control in high dimensional state spaces. The iterative version of path integral control, the so-called Policy Improvement with Path Integrals (PI^2), has been applied for learning and control with torque-driven robotic systems. PI^2 may be classified as model based, semi-model based or model free depending on how the learning problem is formulated. This is useful for learning control applications with complex robotic systems, for which modeling of the underlying dynamics and contact phenomena is very difficult.

One of the main ingredients of the application of PI^2 in previous work has been the use of nonlinear point attractors, called Dynamic Movement primitives (DMPs). DMPs were used to parameterize trajectories for the case of planning or gains for the case of gain scheduling and applications of variable stiffness control. In this work we go one step further by applying (PI^2) to tendon-driven hand systems. In particular we demonstrate the use of PI^2 on an accurate biomechanical model of the index finger, and go on to apply PI^2 to the Anatomically Correct Testbed (ACT) robotic hand for the task of sliding a switch. As we show, PI^2 is flexible because 1) it may be extended to tendon-driven systems and 2) its use does not rely on policy parameterizations, though it can accommodate them if desired. With very small algorithmic changes, PI^2 can be used to either directly compute control commands $\mathbf{u}(\mathbf{x},t)$ or learn parameters $\boldsymbol{\theta}$ which, when projected onto basis functions, represent desired trajectories or control gains: $\mathbf{u}(\mathbf{x},t) = \Phi(\mathbf{x},t)^T\boldsymbol{\theta}$.

In Sections II-III we review the control framework and parameterization. Section IV describes the tendon-driven systems, first in simulation and then on the ACT robotic hand. The experimental conditions and results are described in Section V.

II. Path Integral Control

In this section we review the path integral control framework [1], [2]. We consider the following stochastic optimal control problem with the cost function under minimization given by the mathematical expression:

E. Rombokas, M. Malhotra, and Y. Matsuoka are with the Neurobotics Laboratory, Computer Science & Engineering, University of Washington, Seattle, USA, {rombokas,malhotra,yoky}@cs.washington.edu

E.A. Theodorou is Postdoctoral Research Associate with the Department of Computer Science and Engineering, University of Washington, Seattle, USA etheodor@cs.washington.edu

E. Todorov is with the Movement Control Laboratory, Computer Science & Engineering, University of Washington, Seattle, USA, todorov@cs.washington.edu

$$V(\mathbf{x}) = \min_{\mathbf{u}} J(\mathbf{x}, \mathbf{u}) = \min_{\mathbf{u}} \int_{to}^{t_N} \boldsymbol{\mathcal{L}}(\mathbf{x}, \mathbf{u}, t) dt \tag{1}$$

subject to the nonlinear stochastic dynamics:

$$\mathbf{dx} = \mathbf{F}(\mathbf{x}, \mathbf{u})dt + \mathbf{B}(\mathbf{x})d\mathbf{w} \tag{2}$$

with $\mathbf{x} \in \Re^{n\times 1}$ denoting the state of the system, $\mathbf{u} \in \Re^{p\times 1}$ the control vector and $d\mathbf{w} \in \Re^{p\times 1}$ Brownian noise. The function $\mathbf{F}(\mathbf{x}, \mathbf{u})$ is a nonlinear function of the state $\mathbf{x}$ and affine in controls $\mathbf{u}$ and therefore is defined as $\mathbf{F}(\mathbf{x}, \mathbf{u}) = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}$. The matrix $\mathbf{G}(\mathbf{x}) \in \Re^{n\times p}$ is the control matrix, $\mathbf{B}(\mathbf{x}) \in \Re^{n\times p}$ is the diffusion matrix and $\mathbf{f}(\mathbf{x}) \in \Re^{n\times 1}$ are the passive dynamics. The cost function $J(\mathbf{x}, \mathbf{u})$ is a function of states and controls. Under the optimal controls $\mathbf{u}^*$ the cost function is equal to the value function $V(\mathbf{x})$. The term $\boldsymbol{\mathcal{L}}$(**x**,**u**,t) is the immediate cost and it is expressed as:

$$\boldsymbol{\mathcal{L}}(\mathbf{x}, \mathbf{u}, t) = q_0(\mathbf{x}, t) + q_1(\mathbf{x}, t)\mathbf{u} + \frac{1}{2}\mathbf{u}^T\mathbf{R}\mathbf{u} \tag{3}$$

The immediate cost has three terms[1], the first $q_0(\mathbf{x}_t, t)$ is an arbitrary state-dependent cost, the second term depends on states and controls and the third is the control cost with $\mathbf{R} > 0$ the corresponding weight. The stochastic HJB equation [3], [4] associated with this stochastic optimal control problem is expressed as follows:

$$-\partial_t V = \min_{\mathbf{u}} \left(\boldsymbol{\mathcal{L}} + (\nabla_{\mathbf{x}} V)^T \mathbf{F} + \frac{1}{2} tr\left((\nabla_{\mathbf{xx}} V) \mathbf{G}\mathbf{G}^T \right) \right) \tag{4}$$

To find the minimum, the cost function (3) is inserted into (4) and the gradient of the expression inside the parenthesis is taken with respect to controls $\mathbf{u}$ and set to zero. The corresponding optimal control is given by the equation:

$$\mathbf{u}(\mathbf{x}_t) = -\mathbf{R}^{-1} \Bigg(q_1(\mathbf{x}, t) + \mathbf{G}(\mathbf{x})^T \nabla_{\mathbf{x}} V(\mathbf{x}, t) \Bigg) \tag{5}$$

substitution of the optimal controls into the stochastic HJB (4) results in the following nonlinear, second-order PDE:

$$\begin{aligned} -\partial_t V = \tilde{q} + (\nabla_{\mathbf{x}} V)^T \tilde{\mathbf{f}} - \frac{1}{2} (\nabla_{\mathbf{x}} V)^T \mathbf{G}\mathbf{R}^{-1}\mathbf{G}^T (\nabla_{\mathbf{x}} V) \\ + \frac{1}{2} tr\left((\nabla_{\mathbf{xx}} V) \mathbf{B}\mathbf{B}^T \right) \end{aligned}$$

with $\tilde{q}(\mathbf{x}, t)$ and $\tilde{f}(\mathbf{x}, t)$ defined as $\tilde{q}(\mathbf{x}, t) = q_0(\mathbf{x}, t) - \frac{1}{2} q_1(\mathbf{x}, t)^T \mathbf{R}^{-1} q_1(\mathbf{x}, t)$ and $\tilde{f}(\mathbf{x}, t) = \mathbf{f}(\mathbf{x}, t) - \mathbf{G}(\mathbf{x}, t)\mathbf{R}^{-1} q_1(\mathbf{x}, t)$ and the boundary condition $V(\mathbf{x}_{t_N}) = \phi(\mathbf{x}_{t_N})$. Solving the PDE above, especially for high dimensional dynamical systems remains one of the main challenges in nonlinear optimal control theory. To transform the PDE above into a linear one, we use an exponential transformation of the value function $V = -\lambda \log \Psi$. By inserting the logarithmic transformation and the derivatives of the value function as well as considering the assumption $\lambda \mathbf{G}(\mathbf{x})\mathbf{R}^{-1}\mathbf{G}(\mathbf{x})^T = \mathbf{B}(\mathbf{x})\mathbf{B}(\mathbf{x})^T = \mathbf{\Sigma}$ the resulting PDE is formulated as follows:

$$-\partial_t \Psi = -\frac{1}{\lambda} \tilde{q} \Psi + \tilde{f}^T (\nabla_{\mathbf{x}} \Psi) + \frac{1}{2} tr\left((\nabla_{\mathbf{xx}} \Psi) \mathbf{\Sigma} \right) \tag{6}$$

with boundary condition: $\Psi_{t_N} = \exp\left(-\frac{1}{\lambda}\phi_{t_N}\right)$. Application of the Feynman-Kac lemma to the Chapman-Kolmogorov PDE (6) yields its solution in form of an expectation over system trajectories

$$\Psi\left(\mathbf{x}_{t_i}\right) = E_{\boldsymbol{\tau}_i}\left(e^{-\int_{t_i}^{t_N} \frac{1}{\lambda} q(\mathbf{x}) dt} \Psi(\mathbf{x}_{t_N}) \right) \tag{7}$$

on sample paths $\boldsymbol{\tau}_i = (\mathbf{x}_i, ..., \mathbf{x}_{t_N})$ generated with the forward sampling of the diffusion equation $\mathbf{dx} = \tilde{f}(\mathbf{x}_t)dt + \mathbf{B}(\mathbf{x})d\mathbf{w}$. Thus, the Feynman-Kac lemma is crucial to transforming the stochastic optimal control problem into a problem of approximating a path integral. In discrete time, the solution to 7 is approximated by:

$$\begin{aligned} \Psi(\mathbf{x}_{t_i}) = \lim_{dt\to 0} \int P\Bigg(\mathbf{x}_N, t_N | \mathbf{x}_i, t_i \Bigg) \\ \times \exp\left[\frac{-\left(\phi_{t_N} + \sum_{j=i}^{N-1} q_{t_j} dt \right)}{\lambda} \right] d\mathbf{x}_N \end{aligned} \tag{8}$$

where the probability $P\Bigg(\mathbf{x}_N, t_N | \mathbf{x}_i, t_i \Bigg)$ has the form of path integral. After approximating the exponentiated value function $\Psi(\mathbf{x}, t)$, the optimal controls can be recovered:

$$\mathbf{u}_{PI}(\mathbf{x}) = \mathbf{R}^{-1} \Bigg(q_1(\mathbf{x}, t) + \lambda \mathbf{G}(\mathbf{x})^T \frac{\nabla_{\mathbf{x}} \Psi(\mathbf{x}, t)}{\Psi(\mathbf{x}, t)} \Bigg) \tag{9}$$

where the subscript PI stands for Path Integral. When constraints in control are considered $\mathbf{u}_{\min} \preceq \mathbf{u}_{PI}(\mathbf{x}) \preceq \mathbf{u}_{\max}$ the optimal control is expressed as:

$$\mathbf{u}_{CPI}(\mathbf{x}) = \max \Bigg(\mathbf{u}_{\min}, \min \Bigg(\mathbf{u}_{PI}(\mathbf{x}), \mathbf{u}_{\max} \Bigg) \Bigg)$$

The subscript CPI stands for Constrained Path Integral. The $\min$ and $\max$ operators need to be applied element-wise. In [1], [2] it has been shown that the path integral optimal control takes the form:

$$\mathbf{u}_{PI}(\mathbf{x}_{t_i}) = \lim_{dt\to 0} \int P\left(\boldsymbol{\tau}_i \right) d\mathbf{w}_{t_i} \tag{10}$$

with $\boldsymbol{\tau}_i$ is a trajectory in state space starting from $\mathbf{x}_{t_i}$ and ending in $\mathbf{x}_{t_N}$, therefore $\boldsymbol{\tau}_i = (\mathbf{x}_{t_i}, ..., \mathbf{x}_{t_N})$. The probability $P\left(\boldsymbol{\tau}_i\right)$ is defined as

$$P\left(\boldsymbol{\tau}_i\right) = \frac{e^{-\frac{1}{\lambda}\tilde{S}(\boldsymbol{\tau}_i)}}{\int e^{-\frac{1}{\lambda}\tilde{S}(\boldsymbol{\tau}_i)} d\boldsymbol{\tau}_i} \tag{11}$$

In the iterative version of path integral control framework, $d\mathbf{w}$ can be thought as variations $\delta\mathbf{u}$ in the controls $\mathbf{u}$. An alternative formulation exists when control policies are parameterized as $\mathbf{u}(\mathbf{x}, t) = \mathbf{\Phi}(\mathbf{x}, t)^T \boldsymbol{\theta}$. In these cases, the parameter $\boldsymbol{\theta}$ plays the role of controls while $d\mathbf{w}$ can be thought of as variations $\delta\boldsymbol{\theta}$ in the parameters $\boldsymbol{\theta}$ of the parameterized policy $\mathbf{u}(\mathbf{x}, t)$. Table I illustrates PI^2 when

[1]The aforementioned immediate cost has the additional second term and in that sense is more general than costs where only the first and third terms are considered.

TABLE I: Policy Improvements with path integrals PI^2-I.

- **Given**:
  - An immediate state dependent cost function $q(\mathbf{x}_t)$
  - The control weight $\tilde{\mathbf{R}} \propto \mathbf{\Sigma}^{-1}$
- **Repeat** until convergence of the trajectory cost R:
  - Create K roll-outs of the system from the same start state $\mathbf{x}_0$ using stochastic parameters $\mathbf{u} + \delta\mathbf{u}_s$ at every time step
  - **For** $k = 1...K$, compute costs and weights:
    - $S(\boldsymbol{\tau}_i) = \phi_{t_N} + \sum_{j=i}^{N-1}\left(q_{t_j} + \delta\mathbf{u}_s\ \tilde{\mathbf{R}}\ \delta\mathbf{u}_s\right)dt$
    - $P\left(\boldsymbol{\tau}_{i,k}\right) = \frac{e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}_{i,k})}}{\sum_{k=1}^{K}[e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}_{i,k})}]}$
  - **For** $i = 1...(N-1)$, compute:
    - $\delta\mathbf{u}(\mathbf{x}_{t_i}) = \sum_{k=1}^{K} P\left(\boldsymbol{\tau}_{i,k}\right)\ \delta\mathbf{u}_s(t_i, k)$
  - **Update** $\mathbf{u} \leftarrow \max\left(\mathbf{u}_{\min}, \min\left(\mathbf{u} + \delta\mathbf{u}, \mathbf{u}_{\max}\right)\right)$

TABLE II: Policy Improvements with path integrals PI^2-II.

- **Given**:
  - An immediate state dependent cost function $q(\mathbf{x}_t)$
  - The control weight $\tilde{\mathbf{R}} \propto \mathbf{\Sigma}^{-1}$
- **Repeat** until convergence of the trajectory cost R:
  - Create K roll-outs of the system from the same start state $\mathbf{x}_0$ using stochastic parameters $\boldsymbol{\theta} + \delta\boldsymbol{\theta}_s$ at every time step
  - **For** $k = 1...K$, compute costs and weights:
    - $S(\boldsymbol{\tau}_i) = \phi_{t_N} + \sum_{j=i}^{N-1}\left(q_{t_j} + \delta\boldsymbol{\theta}_s\ \tilde{\mathbf{R}}\ \delta\boldsymbol{\theta}_s\right)dt$
    - $P\left(\boldsymbol{\tau}_{i,k}\right) = \frac{e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}_{i,k})}}{\sum_{k=1}^{K}[e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}_{i,k})}]}$
  - **For** $i = 1...(N-1)$, compute:
    - $\delta\boldsymbol{\theta}(\mathbf{x}_{t_i}) = \sum_{k=1}^{K} P\left(\boldsymbol{\tau}_{i,k}\right)\ \delta\boldsymbol{\theta}_s(t_i, k)$
  - **Time averaging**
    - $\delta\boldsymbol{\theta} = \sum_{i}^{N-1} w_i \delta\boldsymbol{\theta}(\mathbf{x}_{t_i})$
  - **Update** $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \delta\boldsymbol{\theta}$

it is applied to constrained optimal control problems. Table II illustrates PI^2 for the case where parameterized policies are used. The main difference is that in PI^2-I there is no time averaging of the control strategy changes as in PI^2-II. In addition, in the last step of PI^2-I the controls are updated such that constraints are not violated.

The assumption of Path integral control framework $\lambda\mathbf{G}(\mathbf{x})\mathbf{R}^{-1}\mathbf{G}(\mathbf{x})^T = \mathbf{B}(\mathbf{x})\mathbf{B}(\mathbf{x})^T = \mathbf{\Sigma}$ establishes a relationship between control cost and variance of noise. Essentially, high variance results in low control cost and therefore increased control authority. Depending on the levels of noise, the connection between noise and control authority may result in noisy control commands. This characteristic may be desirable when applying path integral control to biomechanical and neuromuscular models, in order to match observed noisy controls. For reinforcement learning applications to robotic systems, however, it may be preferable to use smooth control commands. In that case, nonlinear point attractors [5] offer a low dimensional parameterization of trajectories and control gains. This parameterization reduces the search space and also has a smoothing effect on the control commands. In the next section we review nonlinear point attractors and provide their mathematical formulations.

III. Dynamic Movement Primitives: Nonlinear Point Attractors with adjustable attractor landscape

The nonlinear point attractor consists of two sets of differential equations, the canonical and transformation system which are coupled through a nonlinearity [5]. The canonical system is formulated as $\frac{1}{\tau}\dot{x}_t = -\alpha x_t$. That is a first - order linear dynamical system for which, starting from some arbitrarily chosen initial state x_0 , e.g., $x_0 = 1$, the state x converges monotonically to zero. x can be conceived of as a phase variable, where $x = 1$ would indicate the start of the time evolution, and x close to zero means that the goal g (see below) has essentially been achieved. The transformation system consist of the following two differential equations:

$$\begin{aligned} \tau\dot{z} &= \alpha_z\beta_z\left(\left(g + \frac{f}{\alpha_z\beta_z}\right) - y\right) - \alpha_z z \qquad (12) \\ \tau\dot{y} &= z \end{aligned}$$

These 3 differential equations code a learnable point attractor for a movement from y_{t_0} to the goal g, where $\boldsymbol{\theta}$ determines the shape of the attractor. $y_t, \dot{y}_t$ denote the position and velocity of the trajectory. α_z, β_z, τ are time constants. The nonlinear coupling or forcing term f is defined as:

$$f(x) = \frac{\sum_{i=1}^{N} K\left(x_t, c_i\right)\theta_i x_t}{\sum_{i=1}^{N} K\left(x_t, c_i\right)}(g - y_0) = \mathbf{\Phi}_P(x)^T\boldsymbol{\theta} \qquad (13)$$

The basis functions $K\left(x_t, c_i\right)$ are defined as $K\left(x_t, c_i\right) = \exp\left(-0.5h_j(x_t - c_j)^2\right)$ with bandwith h_j and center c_j of the Gaussian kernels – for more details see [5]. The full dynamics of the point attractor have the form of $d\mathbf{x} = \alpha(\mathbf{x})dt + \mathbf{C}(\mathbf{x})\mathbf{u}dt$ where the state $\mathbf{x}$ is specified as $\mathbf{x} = (y, z)$ while the controls are specified as $\mathbf{u} = \boldsymbol{\theta} = (\theta_1, ..., \theta_p)^T$. Thus $\alpha(\mathbf{x})$ and $\mathbf{C}(\mathbf{x})$ are specified as follows:

$$\alpha(\mathbf{x}) = \begin{pmatrix} z \\ \alpha_z\beta_z(g - y) - \alpha_z z \end{pmatrix} \qquad (14)$$

$$C(\mathbf{x}) = \begin{pmatrix} 0 \\ \mathbf{\Phi}_P(x)^T \end{pmatrix} \qquad (15)$$

The representation above is advantageous as it guarantees that the attractor progresses towards the goal while remaining linear in the parameters $\boldsymbol{\theta}$. By varying $\boldsymbol{\theta}$, the shape of the trajectory changes while the goal state g and initial state y_{t_0} remain fixed. These properties facilitate learning [6].

IV. Tendon-driven systems

In this section we describe two tendon-driven systems used in our work. The first is a dynamical model of the human index finger, and the second is the ACT robotic hand.

A. Index Finger Biomechanics

The skeleton of the human index finger consists of 3 joints connected with 3 rigid links. The two joints (proximal interphalangeal (PIP) and the distal interphalangeal (DIP)) are described as hinge joints that can generate both flexion-extension. The metacarpophalangeal joint (MCP) is a saddle joint and it can generated flexion-extension as well as abduction-adduction. Fingers have at least 6 muscles, and the index finger is controlled by 7. Starting with the flexors, the index finger has the Flexor Digitorum Profundus (FDS) and the Flexor Digitorum Superficialis (FDP). The the Radial Interosseous (RI) acts on the MCP joint. Lastly, the extensor mechanism acts on all three joints. It is an interconnected network of tendons driven by two extensors Extensor Communis (EC) and the Extensor Indicis (EI), and the Ulnar Interosseous (UI) and Lumbrical (LU).

The full model of the index finger is given by:

$$\ddot{\mathbf{q}} = -\mathbf{I}(\mathbf{q})^{-1}\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{B}\dot{\mathbf{q}} + \mathbf{I}(\mathbf{q})^{-1}\mathbf{T} \quad (16)$$

$$\mathbf{T} = \mathbf{M}(\mathbf{q}) \cdot \mathbf{F} \quad (17)$$

$$\dot{\mathbf{F}} = -\frac{1}{\tau}(\mathbf{F} - \mathbf{G}\mathbf{u}) \quad (18)$$

$$\mathbf{u}_{max} > \mathbf{u} > \mathbf{u}_{min} \quad (19)$$

where $\mathbf{I} \in \Re^{6\times6}$ is the inertial matrix and $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) \in \Re_{6\times1}$ is matrix of Coriolis and centripetal forces and $\mathbf{B} \in \Re^{3\times3}$ is the joint friction matrix. The matrix $\mathbf{M} \in \Re^{3\times7}$ is the moment-arm matrix specified in [7], $\mathbf{T} \in \Re^{3\times1}$ is the torque vector, $\mathbf{F} \in \Re^{7\times1}$ is the force in Nt applied on the tendons and $\mathbf{u}$ is the control vector in units of muscle stress Nt/cm^2. Equation (18) is used to model delays in the generation of tensions on the tendons. The matrix $\mathbf{G}$ is determined by the PCSA parameter [7] of each individual muscle- tendon $\mathbf{G} = \mathbf{Diag}(4.10, 3.65, 1.12, 1.39, 0.36, 4.16, 1.60)\,cm^2$. The control constraints are specified as $\mathbf{u}_{min} = 0$ and the maximum muscle stress $\mathbf{u}_{max} = 35Nt/cm^2$.

For our simulations we have excluded the abduction-adduction movement at MCP joint, so we examine the tendon length and velocity profiles necessary for producing planar movements. The quantities $\mathbf{q}$ and $\dot{\mathbf{q}}$ are vectors of dimensionality $\mathbf{q} \in \Re^{3\times1}$,$\dot{\mathbf{q}} \in \Re^{3\times1}$ defined as $\mathbf{q} = (q_1, q_2, q_3)$ and $\dot{\mathbf{q}} = (\dot{q}_1, \dot{q}_2, \dot{q}_3)$. The inertia $\mathbf{I}(\mathbf{q})$ term of the forward dynamics are given in the appendix.

B. The ACT robotic hand

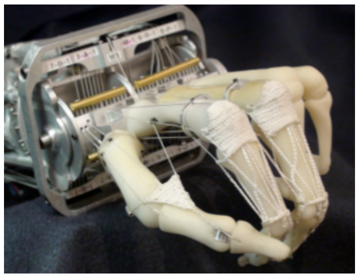


Fig. 1: Anatomically Correct Testbed-ACT hand.

The Anatomically Correct Testbed (ACT) robotic hand mimics the interactions among muscle excursions and joint movements produced by the bone and tendon geometries of the human hand. This mimicry results in a robotic system sharing the redundancies and nonlinearities of the biological hand [8] [9].

The ACT hand uses 24 motor-driven tendons to control a thumb, index finger, middle finger, and wrist. Each segment of these fingers is machined using human bone data, and is accurate in surface shape, mass, and center-of-gravity to the human equivalent. The extensor mechanisms are webs of tendons on the dorsal side of the fingers, and are crucial for emulating dynamic human behavior [10]. As each tendon is pulled by a motor, it is routed through attachment points mimicking human tendon sheaths and following the contours of the bones. Since these bone shapes are complicated surfaces, the effective moment arm the tendon exerts on the joint varies with joint angle [11]. The hand may optionally include a silicon rubber skin on its palmar surfaces. The ACT motors are controlled at 200 Hz using real-time RTAI Linux, and have encoders with a resolution of 230 nm, allowing for precise control and sensing of tendon length.

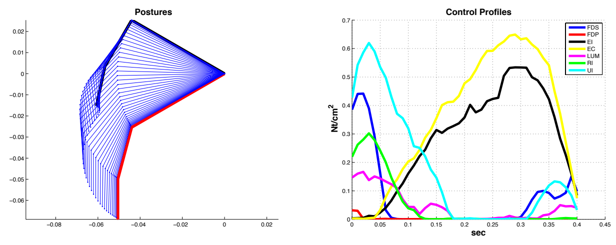
V. Results

A. Biomechanical model: learning to tap

We apply PI^2-I on the biomechanical model of the index finger presented in Section IV-A. The task is to move the finger from an initial posture to final posture. In this work there is no pre-specified trajectory incorporated in the cost function, but there is a constraint in the terminal finger position and velocity. Consequently, there is a terminal cost that is a function of the desired position and velocity states, and it is only the control cost that is accumulated over the time horizon of the movement. In mathematical terms the objective function is expressed as follows:

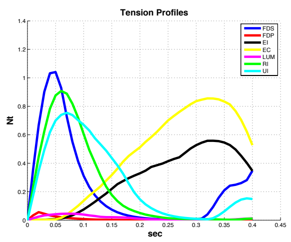
$$J = (\mathbf{q} - \mathbf{q}^*)^T Q_p (\mathbf{q} - \mathbf{q}^*) + \dot{\mathbf{q}}^T Q_v \dot{\mathbf{q}} + \int \mathbf{u}^T R \mathbf{u} dt \quad (20)$$

with $Q_p = 1000 \times I_{3\times3}$, $Q_v = 10 \times I_{3\times3}$ and $R = 250 \times I_{3\times3}$. The desired target posture and desired velocity are defined as $\mathbf{q}^* = (7\pi/6, \pi/4, \pi/12)$ and $\dot{\mathbf{q}}^* = (0, 0, 0)$, while the time horizon is $T = 420ms$.

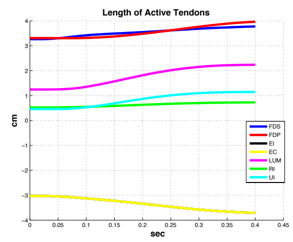


(a) (b)

Fig. 2: Sequence of postures and control profiles for FDS(blue), FDP(red), EI(black), EC(yellow), LUM (cyan), RI(green) and UI(magenta).

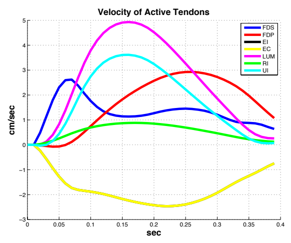


(a)

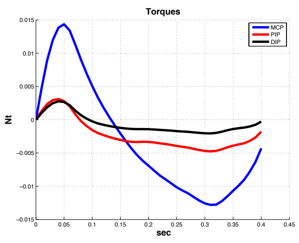


(b)

Fig. 3: Tension profiles and length of FDS(blue), FDP(red), EI(black), EC(yellow), LUM (cyan), RI(green) and UI(magenta).



(a)



(b)

Fig. 4: Velocity profiles for FDS(blue), FDP(red), EI(black), EC(yellow), LUM (cyan), RI(green) and UI(magenta) and joint torques.

The results are shown in Figures 2-4. Figure 2a illustrates the sequence of postures. Figure 2a presents the control profiles required for the finger to perform the tapping movement. The controls are in units of stress Nt/cm^2. Characteristically, the tendons FDS, UI, RI and LUM are activated during the acceleration phase of movement, while the extensor tendons EC and EI are involved in the second, deceleration, phase of the movement. The same synchronization among tendons is shown in Figure 3a that illustrates the tension profiles in units of Nt. The only difference with respect to stress profiles is that the tension applied to the LUM is small relative to FDS, UI, and RI. This observation agrees with studies of the index finger [7] showing that LUM is the weakest tendon.

Tendon excursions are illustrated in Figure 3b. All tendons besides EC and EI are acting as flexors since they are moving inwards (towards the muscle) and therefore their length increases. EC and EI act as extensors since they are moving outward and towards the finger tip. The result of this motion is that their lengths decrease. Figure 4 illustrates the tendon velocities and torques generates at the MCP, PIP and DIP joints.

The application of PI^2-I on the constrained biomechanical model of the finger reveals the efficiency of the method when applied to constrained nonlinear stochastic dynamics. PI^2 is a sampling based method. In contrast to other trajectory optimizers, the efficiency of PI^2 is not affected by the existence of control constraints. In fact, constraints in control reduce the sampling space and improve performance.

B. *ACT Hand: Sliding a switch*

1) Experimental setup: The second experiment is a switch-sliding task using real-world hardware. Before each attempt, the index finger began in an extended position, hovering over the switch in the air (Figure 5). The task consisted of first making contact with the switch, and then sliding it down, using mostly flexion of the MCP joint, though this requirement is implicit in the switch movement performance.

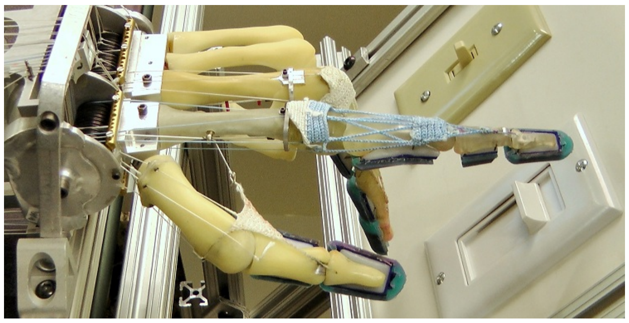


Fig. 5: Experimental setup. The finger begins extended, not touching the switch, and must perform a contact transition.

For the simulation experiment, tendon tension profiles were learned directly, but PI^2 may also be applied to learn other control formulations. For switch-sliding using the ACT hand, we learned controls which drove a nonlinear point attractor system (Section III). The point attractors for all tendons share the same canonical system. The point attractor outputs smooth target trajectories of tendon lengths for a lower-level PID controller. The dynamics of the environment, together with the control-induced dynamics of the hand and the dynamics of the point attractor, may be combined into an augmented plant [12]. In this way, the learning framework encounters the lumped dynamics of robotic manipulator in the context of the task. Figure 6 provides an overview.

A single example of task completion was demonstrated by a human moving the ACT finger through the motion of pushing the switch. The tendon excursions produced by this externally-powered example grossly resemble those required for the robot to complete the task, but simply replaying them using the PID controller does not necessarily result in successful task completion. Firstly, during demonstration the tendons are not loaded, which changes the configuration of the tendon network in comparison to when it is actively moving. Secondly, and more importantly, the tendon trajectories encountered during a demonstration do not impart any information about the necessary torques required to accommodate the dynamics of the task. For instance, at the beginning of the task, the finger must transition from moving through air freely, to contacting and pushing the switch. A PID controller following a reference trajectory has no way of anticipating this contact transition, and therefore will fail to initially strike the switch with enough force to produce the desired motion. The nonlinear point attractor provides a

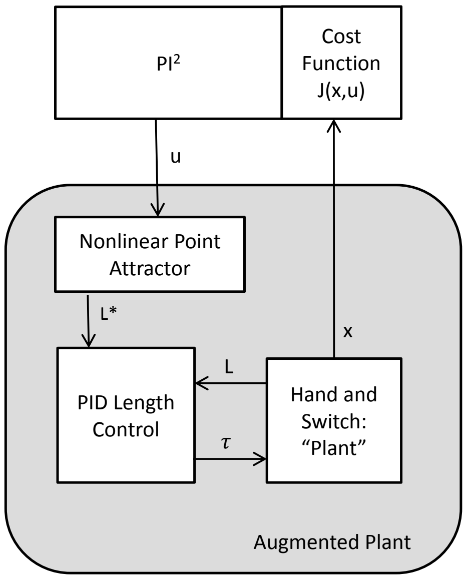


Fig. 6: System overview. A proportional-integral-derivative (PID) controller outputs torques τ to attain the target tendon lengths L^* generated by the nonlinear point attractor. The actual tendon lengths L, and state of the switch x, constitute the sensory feedback observed by the system. The PI^2 framework finds controls u, which minimize the cost for the augmented plant (all components within the shaded box).

means for generating smooth reference trajectories based on the demonstration but modulated by the learned controls u.

Controls take the form $u = \delta\boldsymbol{\theta}$, the change in the parameter $\boldsymbol{\theta}$ determining the shape of the attractor trajectory (see Section III). Each revision of the control parameter $\boldsymbol{\theta}$ we refer to as a *trial*. A sample trajectory is queried from the system by sampling $\delta\boldsymbol{\theta}$, and actually performing a switch-slide using the resulting $\boldsymbol{\theta}$. We refer to one of these exploratory executions of the task as a *rollout*. To revise $\boldsymbol{\theta}$ at the end of a trial, each sampled control strategy is weighted according to the cost encountered by the corresponding rollout (Table II). The results reported here use $\sigma = 30$ for sampling. The smaller this exploration variance is, the more similar rollouts are, so the magnitute of σ should depend on the natural stochasticity of the plant, though here it is set by hand. Convergence is qualitatively insensitive to the exact value of σ, and has been confirmed for σ as low as 10 and high as 50. Each trial consists of fifteen rollouts, and after every third trial, performance is evaluated by executing three exploration-free rollouts ($\sigma = 0$). The cost-to-go function for a rollout having duration T had the following form :

$$C_t = q_{terminal}(x_T) + \sum_t^T q(x_t) + \mathbf{u_t}^T \mathbf{R} \mathbf{u_t} \tag{21}$$

In this cost function, x_t is the location of the switch at time t. $q(x_t)$ is the cost weighting on the switch state, with $q_{terminal}(x_T)$ referring to the terminal cost at the end of the rollout. $\mathbf{R}$ is the cost weighting for controls. Results reported here are for $q_{terminal} = 300, T = 300, q = 1, \mathbf{R} = .3333\mathbb{I}$.

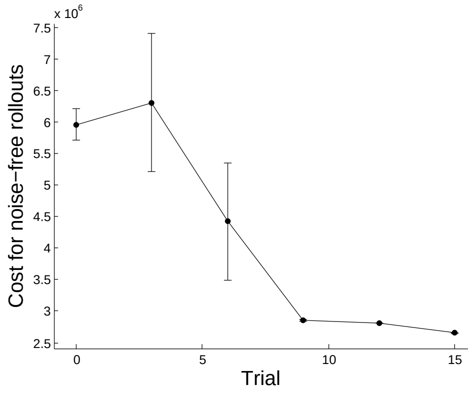


Fig. 7: Sum cost for revisions of the control parameter $\boldsymbol{\theta}$. Each trial consists of a revision based on fifteen rollouts. On every third revision, three exploration-free rollouts were evaluated, each using identical controls, to evaluate learning progress. The bars indicate standard deviation for those three rollouts.

2) ACT Hand Experimental Results: Performance is improved, with decreasing costs as trials progress resulting in the switch being moved further in less time. The sum cost-to-go results for every third trial, begining with trial 0, the "before learning" performance, are reported in Figure 7. Before learning, the system is able to move the switch only a small amount, 0.7 cm, but after 10 trials the switch is pushed to the end of its range (2.75 cm).

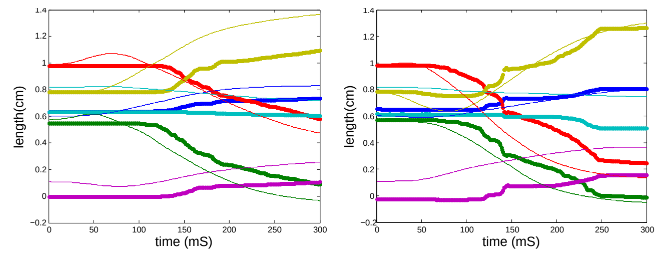
Learning effects on the trajectories of the flexors are most pronounced. Consider the change in reference trajectory for FDS (Figure 8, the red lines beginning near 1cm). Before learning, the reference trajectory undergoes extension before flexing, but after learning it simply flexes, and more aggresively. The dynamics of the underlying PID controller dictate that reference and actual trajectories must differ in order to exert forces on the switch. Tendons may not push, so only differences in the negative direction contribute significantly to forces in the system.

Contact with the switch occurs near 150 mS both before and after learning, but after learning the contact is more vigorous, resulting in greater switch displacement until the end of the range is met near 250 mS, for the post-learning example (Figure 8b).

VI. DISCUSSION

Animals are capable of impressive feats of motor control in novel and uncertain environments, and even major changes to their own bodies due to growth, fatigue, and injury. Through embodied experience of using their bodies to interact with the world, they learn strategies for dealing with the complexities of the sensorimotor landscape. We hope to bring robots closer to this ability by using the world as its own model [13], and emphasize the importance of moving beyond simulation into the complex and uncertain real world.

In this work we perform reinforcement learning in tendon-driven systems in simulation as well as a real robotic system.



(a) (b)

Fig. 8: Lengths before (a) and after (b) learning the switch-pushing task. The bold lines are the actual tendon lengths recorded, and the thin lines are the reference trajectories selected by the learning algorithm. Tendon trajectories displayed are Palmar Interosseus (Blue), FDP (Green), FDS (Red), LUM (Aqua), EI (Purple), and RI (Yellow). Figure (b) corresponds to 15 revisions of the control parameter $\boldsymbol{\theta}$.

PI^2 is a sampling based method in which variations in control are generated, actually run, and then updated using scores according to the cost of the outcome. We show that this can improve the performance of a real-world task despite the complexity of the underlying dynamics, using no models and only sensors of tendon length and switch position.

The successes and limitations of these experiments suggest a number of next steps. For instance, control variations (e.g. $\delta\boldsymbol{\theta}$ for the ACT experiment) were sampled from a Guassian distribution having spherical covariance, but this sampling strategy may be shaped according to observed costs or plant characteristics. Alternatively, incorporatation of sensory feedback for use in the cost function or feedback control would allow for a variety of improvements, such as gain scheduling and variable stiffness control.

VII. Appendix

In this section we provide the parameters of the inertia, coriolis and centripetal forces matrices. More precisely, the elements of the inertia matrix are expressed as follows:

$$\begin{aligned} I_{11} &= I_{31} + \mu_1 + \mu_2 + 2\mu_4 \cos\theta_2 \\ I_{21} &= I_{22} + \mu_4 \cos\theta_2 + \mu_6 \cos\left(\theta_2 + \theta_2\right) \\ I_{22} &= I_{33} + \mu_2 + 2\mu_5 \cos\theta_3 \\ I_{31} &= I_{32} + \mu_6 \cos\left(\theta_3 + \theta_3\right) \\ I_{33} &= \mu_3 \end{aligned}$$

The coriolis and centripetal forces $\mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ are:

$$\begin{aligned} C_1 = {} & \mu_4 \sin\theta_2 \left[-\dot{\theta}_2 \left(2\dot{\theta}_1 + \dot{\theta}_2\right)\right] \\ & + \mu_5 \sin\theta_3 \left[-\dot{\theta}_3 \left(2\dot{\theta}_1 + 2\dot{\theta}_2 + \dot{\theta}_3\right)\right] \\ & - \mu_6 \sin\left(\theta_2 + \theta_3\right) \left(\dot{\theta}_2 + \dot{\theta}_3\right) \\ & \times \left(2\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3\right) \end{aligned}$$

$$\begin{aligned} C_2 = {} & \mu_5 \sin\theta_2 \dot{\theta}_1^2 \\ & - \mu_5 \sin\theta_3 \left[\dot{\theta}_3 \left(2\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3\right)\right] \\ & + \mu_6 \sin\left(\theta_2 + \theta_3\right) \dot{\theta}_1^{\,2} \end{aligned}$$

$$C_3 = \mu_5 \sin\theta_3 \left(\dot{\theta}_1 + \dot{\theta}_2\right) + \mu_6 \sin\left(\dot{\theta}_2 + \dot{\theta}_3\right) \dot{\theta}_1^2$$

The terms μ_1, μ_2, μ_3 are functions of the masses $(m_1, m_2, m_3) = (0.05, 0.04, 0.03)\, Kgr$ and the lengths $(l_1, l_2, l_3) = (0.0508, 0.0254, 0.01905)\, m$ of the 3 bones of the index finger. They are specified as $mu_1 = (m_1 + m_2 + m_3)\,, \mu_1 = (m_1 + m_2 + m_3)\, l_1^2, \mu_3 = m_3 l_3^2, \mu_4 = (m_2 + m_3)\, l_1 l_2, \mu_5 = m_3 l_2 l_3$ and $\mu_6 = m_3 l_1 l_3$.

References

[1] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral approach to reinforcement learning. *Journal of Machine Learning Research*, (11):3137–3181, 2010.

[2] E.. Theodorou. *Iterative Path Integral Stochastic Optimal Control: Theory and Applications to Motor Control.* PhD thesis, university of southern California, May 2011.

[3] Robert F. Stengel. *Optimal control and estimation.* Dover books on advanced mathematics. Dover Publications, New York, 1994.

[4] W. H. Fleming and H. Mete Soner. *Controlled Markov processes and viscosity solutions.* Applications of mathematics. Springer, New York, 2nd edition, 2006.

[5] A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1547–1554. Cambridge, MA: MIT Press, 2003.

[6] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Netw*, 21(4):682–97, 2008.

[7] F J Valero-Cuevas, F E Zajac, and C G Burgar. Large index-fingertip forces are produced by subject-independent patterns of muscle excitation. *Journal of Biomechanics*, 31(8):693–703, 1998.

[8] V. Weghe, M. Rogers, M. Weissert, and Y. Matsuoka. The ACT hand: design of the skeletal structure. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 4, pages 3375–3379. IEEE, 2004.

[9] A.D. Deshpande, J. Ko, D. Fox, and Y. Matsuoka. Anatomically correct testbed hand control: muscle and joint control strategies. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4416–4422. IEEE, 2009.

[10] D.D. Wilkinson, M.V. Weghe, and Y. Matsuoka. An extensor mechanism for an anatomical robotic hand. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 238–243. IEEE, 2003.

[11] A.D. Deshpande, R. Balasubramanian, R. Lin, B. Dellon, and Y. Matsuoka. Understanding variable moment arms for the index finger MCP joints through the ACT hand. In *Biomedical Robotics and Biomechatronics, 2008. BioRob 2008. 2nd IEEE RAS & EMBS International Conference on*, pages 776–782. IEEE, 2009.

[12] E. Todorov, W. Li, and X. Pan. From task parameters to motor synergies: A hierarchical framework for approximately optimal control of redundant manipulators. In *Journal of robotic systems*, volume 22, pages 691–710, 2005.

[13] R.A. Brooks. Intelligence without representation. *Artificial intelligence*, 47(1-3):139–159, 1991.