Optimal Parameter Identification for Discrete Mechanical Systems with Application to Flexible Object Manipulation

T. M. Caldwell and D. Coleman and N. Correll

***Abstract*—**

I. Introduction

The goal of this works is to use a robotic arm to identify the behavior of a flexible object through touch only. The object under consideration for this paper is a flexible loop. We assume the robot has already rigidly grasped the object at one end and that it is clamped at the other. The robot then bends, twists, and stretches the loop. During the manipulation, the robot measures the arm's joint torques and joint angles. With this information, it is possible to back out mechanical properties of the loop in order to generate an accurate model for future control and manipulation. In this paper, the manner in which we model the loop is so that the underlying mechanics of the loop are the same as the robot arm. Furthermore, we provide an optimal control approach for calculating properties of the model that best matches the behavior of the physical loop.

There are many methods to model and simulate flexible objects [9], [10]. A common approach is to model the object as a lattice or collection of links of masses and springs [13], [14], [9]. This approach has been used to simulate linear object like strings, hair, and electrical cables for which the model is a series of masses linked together with springs. Our approach is similar with the primary difference that the loop connects back onto itself. This connection restricts the loops movement and is handled using holonomic constraints.

We approximate the loop as a ring of n_l links. Adjacent links connect together through spherical joints with torsional springs. We assume each link is rigid and as such the vast theory of rigid body mechanics is relevant [12]. Furthermore, since the robot arm is a series of joined rigid links—albeit with actuated joints—the full system, robot arm, and loop can be modeled together. As such, planning and control can be done in the configuration space, instead of the end effector or object space.

We use Rethink Robotics' Baxter [3] robot to both manipulate and measure the loop. Each of Baxter's arms have 7 degrees of freedom. The arms are designed for compliance since each joint has series elastic actuators that allows for force sensing and control. An illustration of Baxter manipulating a six link approximation of a loop is in Figure 1

A good representation of a flexible object is not enough to accurately model it. We also need the model simulation to be consistent with the physical behavior of the loop. We decided not to apply Euler integration or another low order integrator

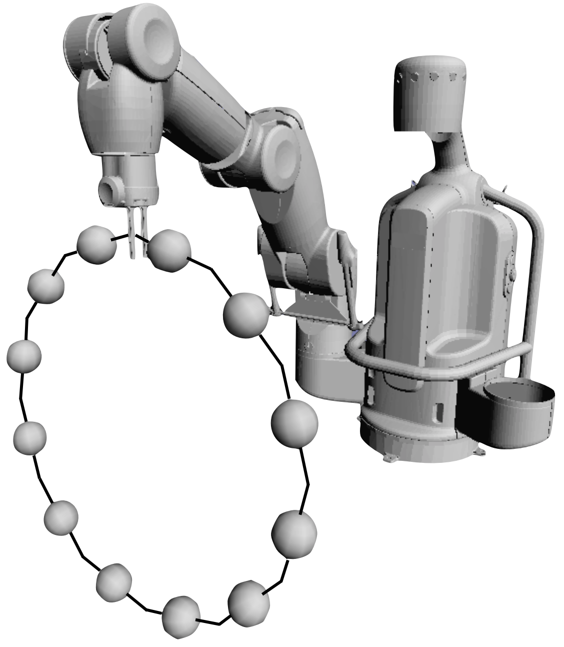

Fig. 1. Illustration of Rethink Robotics' Baxter manipulating the approximated loop.

to the continuous dynamics, as is the case in [13], because such integrators can introduce significant energy errors. At worst these errors will destabilize the integration and at best compromise the model's energy dissipation [5]. Instead we decided to use *variational integrators*. Variational integrators can be used to describe discrete-time equations of motion of a mechanical system. They are designed from the least action principle and have good properties that match known physical phenomenon like preserving—or nearly preserving—conservation of energy and momentum [5], [11].

Furthermore, variational integrators elegantly handle holonomic constraints. Holonomic constraints are specified as $h(q) = 0$, where q is the system's configuration. They are used to constrain positions and orientations. For the example, we use holonomic constraints to "close the loop"—i.e. to constrain the link at one end of the loop to the other end. In simulating continuous dynamics, holonomic constraints are handled with equivalent constraints on the acceleration that creep due to numerical integration error. In comparison, variational integrators apply holonomic constraints directly and do not have this issue (see [5] for a discussion).

Due to recent work by Johnson and Murphey [5], [6],

This material is .....

it is possible to efficiently simulate mechanical systems using variational integrators in generalized coordinates. They provide a framework using a tree representation and caching that not only makes for efficient simulation—especially for articulated rigid bodies—but also efficient model-based calculations like linearizations about a trajectory. In optimal controls, linearizations are needed for gradient calculations like the gradient calculation presented in this paper.

With a model and simulation that accurately predicts the motion of a flexible object, unknown quantities of its behavior, referred to as system parameters, can be identified. These system parameters may be lengths, masses, or damping coefficients. For the example, we assume the loop's stiffness is unknown. As such, the goal is to obtain the spring constants of the torsional springs at the loop model's joints. In this paper, this is done using an optimal control approach.

The parameter identification optimization problem is set up as a discrete-time Bolza problem. For the loop example, the cost function is a summation of the joint errors between the simulated joint angles, for a given set of parameters, and the experimentally-measured joint angles. The joint error directly correlates to the difference between the simulated loop displacement and the measured loop displacement, at least at the manipulator. Alternatively, the cost can be given by a maximum likelihood estimate for which the cost is lower for parameters that correspond to the simulation that is most consistent with the measurement (see [4]).

The paper's contributions are twofold. First, it provides a discrete-time adjoint-based gradient calculation for optimal parameter estimation. Second, it formulates physical parameter identification of flexible objects with variational integrators for greater confidence in the accuracy of the model's simulation.

This paper is organized as follows: Section II reviews continuous and discrete mechanical systems. It also introduces the variational integrator and provides linearization calculations. Section III discusses the parameter identification optimization problem as well as provides an adjoint-based gradient calculation. Section IV provides the details of the physical experiment using Rethink Robotics' Baxter, the model of the loop and the simulation set up. In addition, it applies the parameter identification approach to estimate stiffness properties of the rubber loop model.

II. Mechanical Systems

A comparison of continuous time and discrete time mechanical systems is presented. The mechanical system depends on n_a system parameters from the parameter space $\mathcal{A} \in \mathbb{R}^m$. The mechanical system has n_q generalized coordinates $q \in \mathbb{R}^{n_q}$. For continuous time, q varies the continuous variable t—e.g. $q(t)$—for t in the interval $[0, t_f]$, $t_f > 0$. Likewise, for the discrete representation, q varies with the discrete variable k—e.g. q_k—for k in the set $\mathcal{K} := \{0, 1, \ldots, k_f\}$, $k_f > 0$. Each discrete time k pairs with a continuous time, labeled t_k, where $t_0 = 0$, $t_{k_f} = t_f$ and $t_k < t_{k+1}$. This section presents the continuous and discrete dynamics dependent on the parameters $a \in \mathcal{A}$. The continuous dynamics are provided for comparison with the discrete system, in which the paper results are given.

A. Continuous Mechanical System

We review continuous mechanical systems for reference with discrete mechanical systems.

A mechanical system's evolution is given by the path of least action. The system's action is

$$S = \int_0^{t_f} L(q(\tau), \dot{q}(\tau), a) d\tau$$

where $L(q, \dot{q}, a) := KE(q, \dot{q}, a) - V(q, a)$, the system Lagrangian, is the difference of the system's kinetic energy, KE, with its potential energy, V. For this paper, we assume that both kinetic and potential energies depend on system parameters $a \in \mathcal{A}$. Possible parameters of L are lengths, spring constants, and masses.

As is common in mechanical systems, we wish to include external forces, $F_c(q, \dot{q}, a, t)$. This term is the total external forcing in the generalized coordinates. We also assume dependence on parameters $a \in \mathcal{A}$. For example, likely parameter candidates showing in F_c are damping coefficients. By including the additional external force term, F_c, to the action, the Lagrange d'Alembert principle finds that the continuous dynamics of the mechanical system are given by the forced Euler-Lagrange equations **cite**:

$$\frac{d}{dt}\frac{\partial}{\partial \dot{q}} L(q, \dot{q}, a) - \frac{\partial}{\partial q} L(q, \dot{q}, a) = F_c(q, \dot{q}, a, t).$$

Holonomic constraints can be enforced as external forces using Lagrange multipliers. The n_h holonomic constraint equations are given in the form $h(q, a) = [h_1, \ldots, h_{n_h}]^T(q, a) = 0$. With the addition of the constraint, the forced Euler-Lagrange equations are **cite**:

$$\begin{gathered}\tfrac{d}{dt}\tfrac{\partial}{\partial \dot{q}} L(q, \dot{q}, a) - \tfrac{\partial}{\partial q} L(q, \dot{q}, a) = F_c(q, \dot{q}, a, t) + \tfrac{\partial}{\partial q} h^T(q, a)\lambda \\ \tfrac{\partial^2}{\partial q^2} h(q, a) \circ (\dot{q}, \dot{q}) + \tfrac{\partial}{\partial q} h(q, a)\ddot{q} = 0\end{gathered} \tag{1}$$

where $\lambda(t) \in \mathbb{R}^{n_h}$ are Lagrange multipliers. For fixed parameters a, the system's evolution is integrated from Eq.(1) for q, $\dot{q}$ and λ. It is worth noting that holonomic constraints are not enforced directly but instead the acceleration is constrained. During integration, numerical error will violate the constraint and $h(q, a) \neq 0$.

Equation 1 can be transformed into first-order state space equations. Define the continuous state as $x = [q, \dot{q}]^T$. The state equations, dependent on the parameters $a \in \mathcal{A}$, are $\dot{x}(t) = f(x(t), \lambda(t), a, t)$ where $\ddot{q}$ is specified by the forced Euler-Lagrange equations. In the state space representation, gradient and Hessian calculations with respect to parameters are given for parameter optimization in [7].

B. Discrete Mechanical System

The discrete mechanical system is an approximation of its continuous counterpart. For an initial configuration $q(0)$ and velocity $\dot{q}(0)$, the continuous configuration $q([0, t_f])$ is integrated from the forced Euler-Lagrange equations, Eq.(1). The discrete analog to the forced Euler-Lagrange equations instead calculates the sequence $q_k \approx q(t_k)$ using a variational integrator approach **cite**.

The discrete Lagrangian, labelled L_d, is an approximation of the action over a short time interval. Instead of a velocity term, the discrete Lagrangian is defined by the current and next configurations, q_k and q_{k+1}:

$$L_d(q_k, q_{k+1}, a) \approx \int_{t_k}^{t_{k+1}} L(q(\tau), \dot{q}(\tau), a) d\tau. \tag{2}$$

The integration can be approximated with a quadrature like midpoint or trapezoidal rules. Refer to [5] for details of using midpoint rule, which we use in the example.

Similarly, external forcing is included by approximating F_c by the discrete left and right forces $F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1})$ and $F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1})$ using a quadrature. In addition, the n_h holonomic constraints $h(q_k, a)$ can be enforced with the n_h Lagrange multipliers λ_k. With the discrete Lagrangian, discrete forces and holonomic constraints, [5], finds that the forced discrete Euler-Lagrange equations are:

$$\begin{cases} D_2 L_d(q_{k-1}, q_k, a) + F_d^+(q_{k-1}, q_k, a, t_{k-1}, t_k) \\ \quad + D_1 L_d(q_k, q_{k+1}, a) + F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1}) \\ \quad - D_1 h^T(q_k, a)\lambda_k = 0. \\ h(q_{k+1}, a) = 0 \end{cases} \tag{3}$$

These equations should be viewed as an implicit function on q_{k+1}. For example, given consecutive configurations q_0 and q_1, the next configuration q_2 is found with a root solving operation on Eq.(3). Following, q_3 is obtained from q_1 and q_2 and so forth. Whereas the continuous mechanical system is solved using integration, the discrete mechanical system is solved through recursive calls to root finding Eq.(3).

As in [5], define p_k as

$$p_k := D_2 L_d(q_{k-1}, q_k, a) + F_d^+(q_{k-1}, q_k, a, t_{k-1}, t_k). \tag{4}$$

Without external forcing, p_k is the conserved momentum. With forcing, though, the physical interpretation for p_k is less clear and is defined simply for computational convenience. This convenience comes from defining the discrete state as $x_k := [q_k, p_k]^T$ which has a one-step mapping:

$$x_{k+1} = f(x_k, \lambda_k, a, t_k) := \begin{cases} p_k + D_1 L_d(q_k, q_{k+1}, a) + F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1}) \\ \quad - D_1 h^T(q_k, a)\lambda_k = 0 \\ h(q_{k+1}, a) = 0 \\ p_{k+1} = D_2 L_d(q_k, q_{k+1}, a) + F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1}). \end{cases} \tag{5}$$

This equation is the state equation for the discrete mechanical system. The function $f(x_k, a, t_k)$ is implicit, but, according to the Implicit Function Theorem, it exists when

$$M_{k+1} := D_2 D_1 L_d(q_k, q_{k+1}, a) + D_2 F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1})$$

is nonsingular. By assuming nonsingularity, even though $f(x_k, a, t_k)$ is implicit, the linearization around x_{k+1}—i.e. $\frac{\partial x_{k+1}}{\partial x_k}$—is explicit. Letting $dx_k = [dq_k, dp_k]^T$, be the differential of x_k and da be the differential of a, the linearization of $f(x_k, a, t_k)$ is:

$$\begin{bmatrix} dq_{k+1} \\ dp_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial q_{k+1}}{\partial q_k} & \frac{\partial q_{k+1}}{\partial p_k} \\ \frac{\partial p_{k+1}}{\partial q_k} & \frac{\partial p_{k+1}}{\partial p_k} \end{bmatrix}}_{A_k} \begin{bmatrix} dq_k \\ dp_k \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{\partial q_{k+1}}{\partial a} \\ \frac{\partial p_{k+1}}{\partial a} \end{bmatrix}}_{B_k} da \tag{6}$$

The calculations for the linearization term A_k is given in [6] and duplicated here for reference:

$$\frac{\partial q_{k+1}}{\partial q_k} = -M_{k+1}^{-1}[D_1^2 L_d(q_k, q_{k+1}, a) + D_1 F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1}) - D_1^2 h^T(q_k, a)\lambda_k - D_1 h^T(q_k, a)\frac{\partial \lambda_k}{\partial q_k}] \tag{7a}$$

$$\frac{\partial q_{k+1}}{\partial p_k} = -M_{k+1}^{-1} \tag{7b}$$

$$\frac{\partial p_{k+1}}{\partial q_k} = [D_2^2 L_d(q_k, q_{k+1}, a) + D_2 F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1})]\frac{\partial q_{k+1}}{\partial q_k} + D_1 D_2 L_d(q_k, q_{k+1}, a) + D_1 F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1}) \tag{7c}$$

$$\frac{\partial p_{k+1}}{\partial p_k} = [D_2^2 L_d(q_k, q_{k+1}, a) + D_2 F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1})]\frac{\partial q_{k+1}}{\partial p_k} \tag{7d}$$

where $\frac{\partial \lambda}{\partial q_k}$ can be found in [6]. Notice the calculations for Eqs (7c) and (7d) rely on the calculations for Eqs (7a) and (7b) respectively. The B_k term is given by chain rule:

$$\frac{\partial q_{k+1}}{\partial a} = \frac{\partial q_{k+1}}{\partial p_k}\frac{\partial p_k}{\partial a} + \frac{\partial q_{k+1}}{\partial q_k}\frac{\partial q_k}{\partial a} + M_{k+1}^{-1}[D_1 D_2 h^T(q_k, a)\lambda_k - D_3 D_1 L_d(q_k, q_{k+1}, a) - D_3 F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1})] \tag{8a}$$

$$\frac{\partial p_{k+1}}{\partial a} = [D_2^2 L_d(q_k, q_{k+1}, a) + D_2 F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1})]\frac{\partial q_{k+1}}{\partial a} + [D_1 D_2 L_d(q_k, q_{k+1}, a) + D_1 F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1})]\frac{\partial q_k}{\partial a} + D_3 D_2 L_d(q_k, q_{k+1}, a) + D_3 F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1}) \tag{8b}$$

The term B_k depends on A_k and the previous term B_{k-1}. The linearization of Eq.(5) is needed for calculations of the parameter identification gradient.

III. Parameter Optimization

The goal of parameter optimization is to calculate the system parameters $a \in \mathcal{A}$ that minimize a cost functional. For the continuous problem, the cost functional is the integral of a running cost $\ell(x(t), t, a)$ plus a terminal cost $m(x(t_f), a)$:

Problem 1 (Continuous System Parameter Optimization): Calculate the parameters $a \in \mathcal{A}$ which solves:

$$\min_{a \in \mathcal{A}} \left[J(a) := \int_0^{t_f} \ell(x(t), a) dt + m(x(t_f), a) \right]$$

constrained to $\dot{x}(t) = f(x(t), a, t)$.

The gradient and Hessian of the continuous cost is given in [7]

For the discrete problem, it is reasonable to choose a discrete cost function that approximates the continuous cost—i.e. $\ell_d(x_{k-1}, x_k, a) \approx \int_{t_{k-1}}^{t_k} \ell(x(\tau), a) d\tau$ and $m_d(x_{k_f}, a) \approx m(x(t_f), a)$. Alternatively, ℓ_d and m_d can be designed directly without first choosing an underlying continuous cost. The discrete parameter optimization problem is as follows:

Problem 2 (Discrete System Parameter Optimization): Calculate the parameters $a \in \mathcal{A}$ which solves:

$$\min_{a \in \mathcal{A}} \Big[J_d(a) := \sum_{k=1}^{k_f} \ell_d(x_k, a) + m_d(x_{k_f}, a) \Big]$$

constrained to $x_{k+1} = f(x_k, a, t_k)$, Eq.(5).

In optimal control theory, it is common practice to solve optimization problems using iterative methods. Iterative optimization methods repeatedly reduce the cost by stepping in a descending direction until a local optimum is found. Commonly, the step direction and step size is calculated using local derivative information [2], [8], which is the case for this paper. In the next section, we provide an adjoint-based calculation for the gradient of the cost function with respect to the parameters.

A. Discrete System Parameter Gradient

The gradient of the cost function given in problem 2 is provided in the following lemma.

Lemma 1: Suppose $L_d(q_k, q_{k+1}, a)$, $F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1})$, $F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1})$, and $h(q_k, a)$ are $\mathcal{C}^2$ with respect to q_k, q_{k+1} and a. Take A_k and B_k from Eq.(6) and assume M_k is always nonsingular. Then,

$$DJ_d(a) = \sum_{k=1}^{k_f} \lambda_k B_{k-1} + D_2\ell_d(x_k, a) + D_2 m_d(x_{k_f}, a) \quad (9)$$

where λ_k is the solution to the backward one-step mapping

$$\lambda_k = \lambda_{k+1} A_k + D_1 \ell_d(x_k, a) \quad (10)$$

starting from $\lambda_{k_f} = D_1\ell(x_{k_f}, a) + D_1 m_d(x_{k_f}, a)$.

Proof: The derivative of the cost in the direction $\theta \in \mathbb{R}^{n_a}$ is

$$\begin{aligned} DJ_d(a)\theta := &\textstyle\sum_{k=1}^{k_f} D_1\ell_d(x_k, a)\frac{\partial x_k}{\partial \theta} + D_2\ell_d(x_k, a)\theta \\ &+ D_1 m_d(x_{k_f}, a)\frac{\partial x_{k_f}}{\partial \theta} + D_2 m_d(x_{k_f}, a)\theta. \end{aligned} \quad (11)$$

Label $z_k := \frac{\partial x_k}{\partial \theta}$ for convenience. Also for convenience, label

$$H := \sum_{k=1}^{k_f} D_1\ell_d(x_k, a) z_k + D_1 m_d(x_{k_f}, a) z_{k_f} \quad (12)$$

The linearized state, z_k, is the solution to the linearized state equation, Eq.(6). In other words,

$$z_{k+1} = A_k z_k + B_k \theta$$

starting from $z_0 = 0$. The linearized state's solution depends on the discrete state transition matrix defined as

$$\Phi(k_2, k_1) := \prod_{j=1}^{k_2-k_1} A_{k_2-j} = A_{k_2-1}A_{k_2-2}\cdots A_{k_1}$$

for integers $k_2 > k_1$ and where $\Phi(k_1, k_1) := I$, the identity matrix. Recalling $z_0 = 0$, the linearized state's solution is

$$z_k = \Phi(k, 0)z_0 + \sum_{s=0}^{k-1} \Phi(k, s+1)B_s\theta = \sum_{s=0}^{k-1} \Phi(k, s+1)B_s\theta$$

for $k = 1, \ldots, k_f$. Plugging z_k into H, Eq.(12), H becomes

$$\begin{aligned} H = &\textstyle\sum_{k=1}^{k_f} D_1\ell_d(x_k, a)\sum_{s=0}^{k-1}\Phi(k, s+1)B_s\theta \\ &+ D_1 m_d(x_{k_f}, a)\textstyle\sum_{s=0}^{k_f-1}\Phi(k_f, s+1)B_s\theta \\ = &\textstyle\sum_{k=1}^{k_f}\sum_{s=0}^{k-1} D_1\ell_d(x_k, a)\Phi(k, s+1)B_s\theta \\ &+ \textstyle\sum_{s=0}^{k_f-1} D_1 m_d(x_{k_f}, a)\Phi(k_f, s+1)B_s\theta \end{aligned}$$

Switch the order of the double sum.

$$\begin{aligned} H = &\textstyle\sum_{s=0}^{k_f-1}\sum_{k=s+1}^{k_f} D_1\ell_d(x_k, a)\Phi(k, s+1)B_s\theta \\ &+ D_1 m_d(x_{k_f}, a)\textstyle\sum_{s=0}^{k_f-1}\Phi(k_f, s+1)B_s\theta \\ = &\textstyle\sum_{s=0}^{k_f-1}\Big[\sum_{k=s+1}^{k_f} D_1\ell_d(x_k, a)\Phi(k, s+1) \\ &+ D_1 m_d(x_{k_f}, a)\Phi(k_f, s+1)\Big]B_s\theta \end{aligned}$$

Set λ_{s+1} as the resulting covector in the brackets so that

$$H = \sum_{s=0}^{k_f-1} \lambda_{s+1} B_s \theta = \sum_{k=1}^{k_f} \lambda_k B_{k-1}\theta.$$

The efficient calculation for λ_k is given in Eq.(10). Plugging H into Eq.(11), we find

$$DJ(a)\theta = \Big[\sum_{k=1}^{k_f} \lambda_k B_{k-1} + D_2\ell_d(x_k, a) + D_2 m_d(x_{k_f}, a)\Big]\theta$$

and the proof is finished. ■

IV. Experiment

The goal is to identify the model parameters of a flexible loop that best match its physical behavior. We use Rethink Robotics' Baxter robot to both manipulate and measure the loop. Furthermore, we simulate this interaction using a discrete model of the loop. The simulated arm joint angles and torques depend on the parameters set for a simulation. As such, we can find the parameters which correspond to simulated joint angles that best match the measured joint angles. For this experiment, the unknown parameters describe the stiffness of the loop.

The tools for simulating Baxter manipulating a loop are given in Section II while the optimization technique for best matching the simulation to the measured is in Section III. The following provides the experimental setup, the model and simulation of Baxter manipulating the loop, and the results of optimally identifying loop parameters.

A. Experimental Setup

The experiment is set up as follows. Baxter's arm is positioned similarly to that in Figure 1. The top of a rubber loop is placed in Baxter's gripper and the bottom is clamped to the table. The loop is initially positioned vertically. The loop has a radius of 0.355 meters and a mass of 0.132 kilograms. The arm is run through a recorded motion that is designed to stretch, bend, and twist the tube. The arm's joint angles and torques are captured every 100Hz, which we compile into the two vectors b_{meas} and T_{meas}.

B. Model

We model Baxter and the loop with the same mechanics. We model both as a series of rigid links connected by rotational joints. For Baxter, these joints are 7 series elastic actuators. The dimensions, inertia, and other information concerning Baxter's arm can be obtained at `https://github.com/RethinkRobotics`.

We use a discrete model of the loop. The loop is composed of 12 rigid links wherein the connection between each rigid link is a spherical joint. A single joint is shown in Figure 2. As seen in the figure, the orientation of the new link is given by the angles θ_i and ψ_i. These rotations are described on a frame that is rotated about the $X-axis$ by $5\pi/6$ radians so that when each $\psi_i = \theta_i = 0$, the loop approximation is a regular dodecagon. These angles, the θ_i and ψ_i, along with Baxter's joint angles make up the system configuration q.

We represent Baxter grasping the loop using a tree structure of transformations. This tree structure is illustrated in Figure 3a). Starting from the base of Baxter, the arm is specified with successive rotation and translation transformations. At Baxter's end effector, labeled H, the description branches, splitting the loop into two sides, marked with ℓ and r for left and right. The two branches meet at the base of the loop, where the loop is clamped. For arbitrary Baxter and loop joint angles, q, there is no guarantee that the two ends of the loop meet at the clamping location in space. When they do, we say that the system configuration satisfies the holonomic constraint $h(q) = 0$. There are 12 total constraints—i.e. $h(q) \in \mathbb{R}^{12}$—6 constraints for each side of the loop transformation description in order to constrain the orientation and position of frames N_ℓ and N_r to the clamp location. Figure 3 shows three different configurations q that satisfy the constraints.

We model the stiffness of the loop with torsional springs on each loop configuration variable—i.e. for each θ_i and ψ_i. Due to the uniformity of the loop, we assume that all of the springs on the θ_i configuration variables have the same spring constant κ_θ. Likewise, each of the springs on ψ_i configuration variables have same spring constant κ_ψ. The goal of this section is to identify these spring constant and as such we set $a = [\kappa_\theta, \kappa_\psi]^T$.

C. Simulation

In order to simulate the robot arm and loop, we turn to Section II which reviews variational integrators. The continuous state is given by q and its time derivatives—i.e. $x(t) = [q(t), \dot{q}(t)]^T$. These are the robot and loop joint angles and associated angular velocities. The system's discrete time state is $x_k = [q_k, p_k]^T$ where p_k is defined in Eq.(4). The continuous dynamics are given by the constrained, forced Euler Lagrange equations, Eq.(1), which depends on the system Lagrangian $L(q, \dot{q}, a)$, external forces $F_c(q, \dot{q}, a, t)$ and holonomic constraints $h(q, a)$. Techniques to derive these formulas for rigid bodies are well understood [12].

From the continuous dynamics, it is straightforward to obtain the discrete dynamics, which are given in Eq.(3). The discrete dynamics depend on the discrete system Lagrangian $L_d(q_k, q_{k+1}, a)$, discrete external left and right forces, $F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1})$ and $F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1})$, and holonomic constraints $h(q_k, a)$.

For simulation, we chose a constant time step of $\Delta_t = 0.01$ seconds. This time step matches the broadcast frequency of Baxter. Further, we decided to approximate L_d from L using midpoint rule (see Eq.(13)):

$$L_d(q_k, q_{k+1}, a) = \Delta_t L\left(\frac{q_{k+1} + q_k}{2}, \frac{q_{k+1} - q_k}{\Delta_t}, a\right). \quad (13)$$

Similarly, using midpoint rule, we approximate F_c by F_d^- and F_d^+ where

$$\begin{cases} F_d^-(q_k, q_{k+1}, a, t_k, t_{k+1}) = \\ \qquad\qquad \Delta_t F_c\left(\frac{q_{k+1}+q_k}{2}, \frac{q_{k+1}-q_k}{\Delta_t}, a, \frac{t_{k+1}+t_k}{2}\right) \\ F_d^+(q_k, q_{k+1}, a, t_k, t_{k+1}) = 0. \end{cases} \quad (14)$$

It is a simple process to translate the continuous system to the discrete variational integrator one. Finally, in order to simulate the dynamics using the one-step mapping in Eq.(5), we need certain partial derivatives of the Lagrangian, external forces and constraints with respect to configuration variables, which can be found in [5].

The external forces F_c are applied at the configuration variables. For Baxter manipulating the loop, the only external forces are those applied at the 7 configuration variables that define Baxter's arm. Label those configuration variables as b. Due to model and sensor error, it is not reasonable to expect that the simulation will provide meaningful results, let alone be stable by setting F_c to the unfiltered joint torques measured from Baxter. Instead, we decided to filter them using a proportional feedback loop as so:

$$F_c(t) = T_{meas}(t) - K(t)(b(t) - b_{meas}(t)),$$

where b_{meas} and T_{meas} are Baxter's measured joint angles and torques as mentioned in Section IV-A and $K(t_k)$ is a feedback gain. When $K(t)$ is chosen correctly, the simulation is stable. We chose $K(t)$ from a finite horizon LQR which calculates an optimal feedback gain from a quadratic cost function [1].

Aside: We used the software tool TREP [5] which simulates articulated rigid bodies using midpoint variational integrators. It additionally provides partial derivative calculations that we need for the system linearization, Eqs.(7) and (8).

D. Linearization

The linearization of the discrete equations of motion is given by matrices A_k and B_k in Eqs.(7) and (8). We need the linearization for the gradient calculation, Lemma 1, in order to perform a gradient-based descent algorithm like steepest descent for parameter identification. Partial derivatives of L_d and F^+ with respect to q_k and p_k can be obtained from [6]. In this section, we only concern ourselves with the partials that depend on the paramaters $a = [\kappa_\theta, \kappa_\psi]^T$, which are not included in [6]

For the loop example, we need to calculate $D_3 D_1 L_d(q_k, q_{k+1}, a)$ and $D_3 D_1 L_d(q_k, q_{k+1}, a)$ for

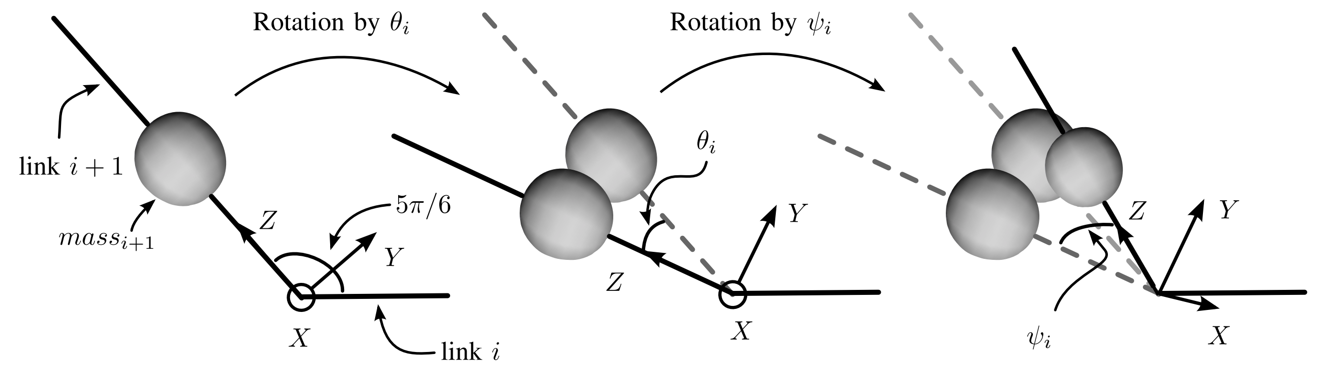

Fig. 2. Illustration of the spherical joint i connecting links i and $i+1$. The joint is a rotation of θ_i radians about the X-axis followed by a rotation of ψ_i radians about the Y-axis.

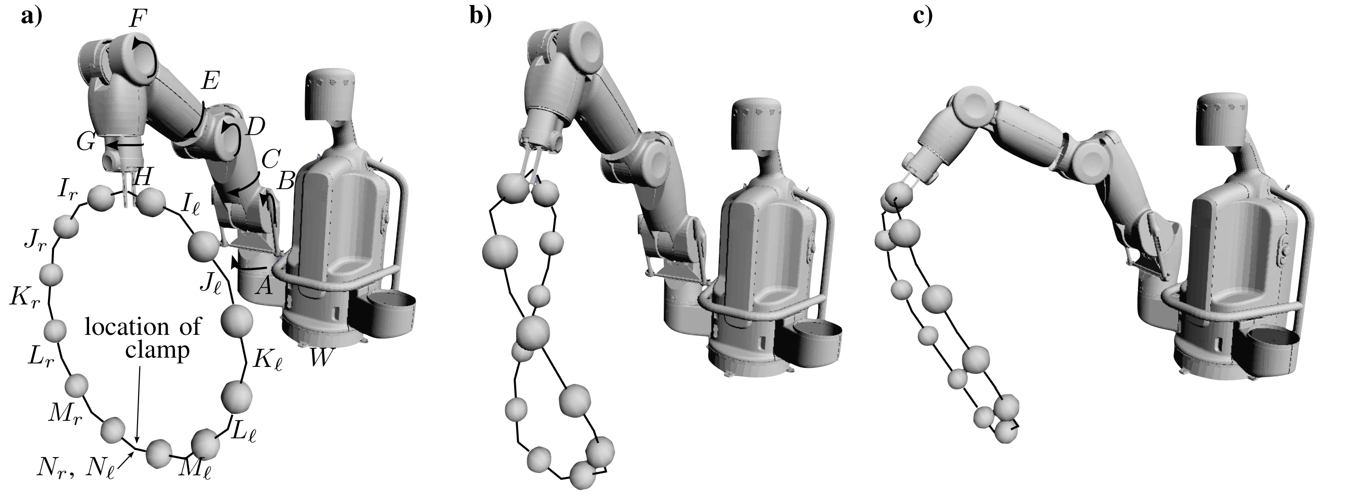

Fig. 3. Three distinct configurations. **a)** The frames for Baxter and the loop in their initial configuration. **b)** Baxter "twisting" the loop. **c)** Baxter "bending" the loop.

$a = [\kappa_\theta, \kappa_\psi]$. Note that the potential energy of the system can be written as:

$$V(q,a) = V_\theta(q,\kappa_\theta) + V_\psi(q,\kappa_\psi) + V_g(q)$$

where V_θ, V_ψ and V_g are the potential energies due to the spring torques on the $theta_i$ configuration variables, the spring torques on the ψ_i configuration variables, and gravity, respectively. Label I_θ and I_ψ index the θ_i and ψ_i configuration variables in q respectively. The potential energy due to the θ_i torsional springs is $V_\theta(q,\kappa_\theta) = \sum_{i\in I_\theta} \frac{1}{2}\kappa_\theta \theta_i^2$. Approximating for the discrete time potential energy—see Eq.(13)—we find that [1]

$$V_{\theta,d}(q_k, q_{k+1}, \kappa_\theta) = \sum_{i\in I_\theta} \frac{\Delta_t}{2}\kappa_\theta\left(\frac{q_{i,k+1}+q_{i,k}}{2}\right)^2.$$

Taking the needed partial derivatives to calculate $D_3D_1L_d$, the i^{th} element of $D_3D_1V_{\theta,d}$ is

$$D_3D_1V_{\theta,d}(q_k,q_{k+1},\kappa_\theta)_i = \begin{cases} \frac{\Delta_t}{4}(q_{i,k+1}+q_{i,k}), & i \in I_\theta \\ 0 & \text{else.} \end{cases}$$

[1]Here, we (poorly) index the i^{th} term of q_k as $q_{i,k}$.

Similarly, for the needed partial derivatives of $V_{\psi,d}$,

$$D_3D_1V_{\psi,d}(q_k,q_{k+1},\kappa_\psi)_i = \begin{cases} \frac{\Delta_t}{4}(q_{i,k+1}+q_{i,k}), & i \in I_\psi \\ 0 & \text{else.} \end{cases}$$

Since the kinetic energy does not depend on $a = [k_\theta, k_\psi]$,

$$\begin{aligned} &D_3D_1L_d(q_k,q_{k+1},a) = \\ &\quad -[D_3D_1V_{\theta,d}(q_k,q_{k+1},\kappa_\theta), D_3D_1V_{\psi,d}(q_k,q_{k+1},\kappa_\psi)]. \end{aligned}$$

Repeating the derivation for $D_3D_2L_d(q_k,q_{k+1},a)$ we find that $D_3D_2L_d(q_k,q_{k+1},a) = D_3D_1L_d(q_k,q_{k+1},a)$

Furthermore, we need to calculate $D_1h(q_k,a)$, $D_1^2h(q_k,a)$ and $D_1D2h(q_k,a)$, the last of which is 0 since the constraints do not depend on the parameters. These partial derivatives of h are given simply by chain rule and depend on the first and second partial derivatives of the transformations from the world frame W to frames N_r and N_ℓ (refer to Figure 3 for the frames). These partial derivatives can be found in [6].

E. Optimal Paramater Identification

With the linearization of the discrete state equations, Eq.(5), we can calculate the gradient of a cost from Lemma 1 in a steepest descent algorithm to identify the model parameters $a = [\kappa_\theta, \kappa_\psi]^T$. The experiment is set up as in Section IV-A. We program Baxter's arm to stretch, bend and twist the loop. The joint torques to displace the loop are recorded in T_{meas} while joint angles are recorded in b_{meas}. For fixed parameters a, the system can be simulated as discussed in Section II using T_{meas}. We wish to find the simulated joint angles for Baxter, labelled b—recall b are the configuration variables of q that describe Baxter's arm—that best match b_{meas}.

This matching is measured by the cost J_d. We choose J_d to be quadratic on the error of b from b_{meas}. Set $\epsilon_k := b_k - b_{meas}(t_k)$. The cost J_d is defined by the running cost $\ell_d(q_k, a)$ and the terminal cost $m_d(q_{k_f}, a)$. We set them to

$$\ell_d(q_k, a) = \epsilon_k^T \epsilon_k \text{ and } m_d(q_{k_f}, a) = \epsilon_{k_f}^T \epsilon_{k_f}$$

The cost measures the quality of the parameter estimate. We seed the steepest descent algorithm with an initial guess of $a = [10, 10]^T$. At each iteration of the descent, an Armijo line search updates the parameters by choosing a distance to step in the direction of the negative gradient. We used Armijo parameters $\alpha = \beta = 0.4$ [2]. After $XXXXXXXX$ iterations, the algorithm terminated with gradient norm $DJ_d(a) = XXXXXX$. The parameters were identified as $[XXXXXX, XXXXXX]^T$. The convergence is shown in XXXXX.

V. Discussion

Through the proposed parameter identification procedure, we calculated the model parameters that best match physical phenomena. This process is important since the improved model can make for better object manipulation. However, it is unclear, especially in the presented experiment, what this matching tells us about the object's physical properties. Certainly, measuring at a single contact point provides little insight into the interior stress and strain of the rubber loop.

Furthermore, we assumed the loop has uniform stiffness. Without this assumption, multiple experiments would be needed at different contact points to identify the non-uniformity. Also, objects with more complex geometries would require additional experiments. For instance, grasping and manipulating a single leaf of a tomato plant would provide little insight for modeling the stiffness of the full plant. Multiple experiments is a topic of future work.

VI. Conclusion

References

[1] B. D. O. Anderson and J. B. Moore. *Optimal Control: Linear Quadratic Methods*. Dover Publications, INC, 1990.

[2] L. Armijo. Minimization of functions having lipschitz continuous first-partial derivatives. *Pacific Journal of Mathematics*, 16:1–3, 1966.

[3] E. Guizzo and E. Ackerman. How rethink robotics built its new baxter robot worker. *IEEE Spectrum*, 2011.

[4] B. Houska, F. Logist, M. Diehl, and J. Van Impe. A tutorial on numerical methods for state and parameter estimation in nonlinear dynamic systems. In *Identification for Automotive Systems*, pages 67–88. Springer, 2012.

[5] E. R. Johnson and T. D. Murphey. Scalable variational integrators for constrained mechanical systems in generalized coordinates. *IEEE Transactions on Robotics*, 25(6):1249–1261, 2009.

[6] E. R. Johnson and T. D. Murphey. Linearizations for mechanical systems in generalized coordinates. *American Control Conference*, pages 629–633, 2010.

[7] L. M. Johnson and T. D. Murphey. Simultaneous optimal estimation of mode transition times and parameters applied to simple traction models. *IEEE Transactions on Robotics*, 29(6):1496–1503, 2013.

[8] C. T. Kelley. *Iterative Methods for Optimization*. Society for Industrial and Applied Mathematics, 1999.

[9] F. F. Khalil and P. Payeur. Dexterous robotic manipulation of deformable objects with multi-sensory feedback-a review. *Robot Manipulators, Trends and Development, In-Teh (Eds)*, pages 587 – 621, 2010.

[10] J. Lang, D. K. Pai, and R. J. Woodham. Acquisition of elastic models for interactive simulation. *The International Journal of Robotics Research*, 21(8):713–733, 2002.

[11] J. E. Marsden and M. West. Discrete mechanics and variational integrators. *Acta Numerica*, 10(1):357–514, 2001.

[12] R. M. Murray, Z. Li, and S. S. Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[13] K. S. M. Sahari, C. H. Min, and Y. C. Hou. Dynamic modeling of string for robotics application. In *Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS)*, pages 774–779. IEEE, 2012.

[14] H. Wakamatsu and K. Takahashiand S. Hirai. Dynamic modeling of linear object deformation based on differential geometry coordinates. *International Conference on Robotics and Automation*, pages 1028–1033, 2005.