A Parallelized Iterative Algorithm for Real-Time Simulation of Long Flexible Cable Manipulation

Jeongmin Lee, Minji Lee, Jaemin Yoon and Dongjun Lee

***Abstract*— We propose a novel real-time physically-accurate simulator for long flexible cable manipulation. We first discretize the cable into multiple rigid link segments, each with complementarity-based contact model and inter-segment compliant coupling; and partition the cable into a number of subsystems, each composed with a number of consecutive links. We then formulate the inter-subsystem consistency constraint as a certain analytical condition among the inter-subsystem coupling and the contact impulses; and solve each subsystem dynamics in parallel with the contact model together with this consistency condition in an iterative manner, achieving both the speed and the accuracy of the simulation. A novel post-regulation scheme is also proposed to further speed up the simulation. Experimental validation/demonstration are also performed to show the theory.**

I. Introduction

As the hardware and software technologies of robotics are rapidly advancing and, in some cases, even quickly maturing, many attempts are being recently reported and witnessed to bring in robots into real industrial processes and applications. Among numerous examples of this industrial robotic application, one with an importance for many industrial sectors and domains is the long flexible cable manipulation: e.g., electrical wire handling [1], automotive wire-harness assembly, fabrication with hot wire cutting [2], and even probe navigation and knot tying for surgery. See also [3].

Properly manipulating this long flexible cable for the robotic applications as stated above is challenging, since the cable, due to its length and flexibility, can itself assume fairly complex geometric shape with very large degree-of-freedom (DOF), yet, the actuation DOF of the robots are typically only few. This is further complicated by the cable-environment/object contacts possibly arbitrarily spread throughout the long cable with their occurrences in turn depending on the cable behavior, thus, typically neither predictable *a prior* (e.g., many iterations of very large-DOF cable model simulation with contacts necessary) nor measurable with affordable-enough sensor arrays (e.g., number of individual sensors too large or sensing itself infeasible (e.g., very thin cable, hot cutting wire)).

Due to these reasons, a model-based control approach for the long flexible cable manipulation is difficult to attain if not impossible in most cases, as it requires the followings: 1) a real-time computable control model with very large DOF and complex contact situation; and 2) an affordable sensor system with very large number of proprioceptive and contact sensors for feedback control. The discontinuity of the dynamics stemming from the contacts further substantially complicates the design of conventional model-based control (e.g., control primitive design and their switching strategy). An promising alternative to this is the recently burgeoning data-driven or machine learning approaches, particularly those based on reinforcement learning (RL [4]). This technique of RL necessitates fast and accurate simulators, as it typically requires vast amount of data, that is often too expensive and time-consuming to obtain via real physical experiments, while also completely physically-inaccurate data, however many or augmented/randomized they are, would likely fail for the sim-to-real (e.g., [5]) of the learning control policy into the reality.

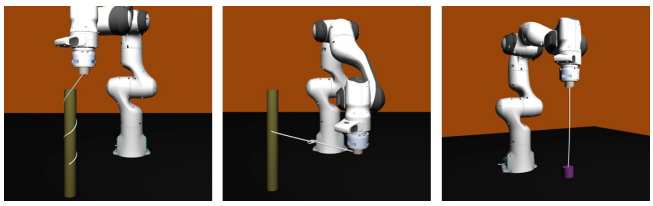

Fig. 1: Various long flexible cable manipulation simulation: winding, knot tying, and lifting from left to right.

In this paper, we propose a novel real-time experimentally-validated simulator for long flexible cable manipulation which is applicable for various scenarios. We first discretize a long flexible cable into multiple rigid cylindrical-link segments, each with contact model in the form of nonlinear complementarity problem (NCP [6]) and inter-segment compliant coupling based on the Cosserat rod theory [7] incorporating extension, shear, bending and torsion between the links. We also adopt our recently proposed passive midpoint integration (PMI [8]) as the basis of our simulator, as it is stable against wide range of simulation parameters (e.g., very stiff constraint, very light link etc.) due to its peculiar discrete-time passivity property.

Directly solving this large number of rigid segments with contacts simultaneously, however, is in general time-consuming, and to circumvent this, we render our simulation framework to be parallelized and iterative. More precisely, we partition the long cable into a number of subsystems, each consisting of a number of the consecutive rigid-link segments. We then solve the dynamics of each subsystem in parallel while formulating their inter-subsystem mechanical consistency as an analytical condition among the inter-subsystem coupling impulses and the contact impulses; and iteratively enforce this consistency condition along with the dynamics of each subsystem with the contact model, thereby, simultaneously achieving both the computation speed (i.e.,

This research was supported by the Industrial Strategic Technology Development Program (20001045) of the Ministry of Trade, Industry & Energy (MOTIE) of Korea.

The authors are with the Department of Mechanical & Aerospace Engineering and IAMD, Seoul National University, Seoul, Republic of Korea. {ljmlgh,mingg8,yjm5181,djlee}@snu.ac.kr. Corresponding author: Dongjun Lee.

real-time) and the structural/dynamic consistency among the segments/subsystems of the long flexible cable simulation. A novel post-regulation scheme of the projected Gauss-Seidel (PGS) solver [9] for the contact NCP is also proposed to prevent its ill-behaving and, consequently, quicker iteration convergence and faster simulation time. The accuracy of the proposed simulator is then validated and demonstrated against real experimental results.

This fast and accurate simulation of a long flexible cable has not been incorporated in any of well-known robot simulators (e.g., Vortex [10], Bullet [11]). In fact, this lack of proper simulation for contact-rich flexible cables or, more generally, deformable objects, has been pointed out in [12] as the key obstacle for their learning-based control development. Recently, the SOFA simulator [13] has been developed for deformable/soft objects simulation for robotic applications with real-time speed. However, from its being based on data-driven model reduction with each reduced model corresponding to each contact mode, it is more suitable for the case where the number of contact modes is not large [14] and not applicable for the case of long flexible cable simulation, where the number of possible contact modes is too many to be captured by a reasonably many reduced models (e.g., winding, self-collisions). To our knowledge, our simulator proposed in this paper is the very first real-time physically-correct and experimentally-validated simulation framework for the long flexible cable manipulation.

The rest of the paper is organized as follows. Sec. II presents some preliminary materials used in the ensuing developments. Sec. III describes the dynamics of the discretized cable in PMI and Sec. IV proposes the main result, i.e., our parallelized and iterative solver for the long flexible cable simulation. Sec. V presents validation and demonstration of our simulator with real experimental results. Sec. VI summarizes the paper with some comments on future research directions.

II. Preliminary

A. Cosserat Rod Model

In order to simulate flexible movement of the cable while reflecting the material properties, we adopt the Cosserat rod model [7] which describes one dimensional rod by the centerline and the local reference frame. Using continuous parameter s, strains along the centerline are calculated as

$$\Gamma(s) = \delta r(s) - d_3(s)$$
$$\Omega(s) = 2\bar{q}^*(s)\frac{\partial}{\partial s}\bar{q}(s)$$

where $r : s \to \mathbb{R}^3$ is the centerline function, d_i $(i = 1 \sim 3)$ is the local reference frame, $\Gamma : s \to \mathbb{R}^3$ is the extension and the shear strain, $\Omega : s \to \mathbb{R}^3$ is the bending and the torsion strain, $\bar{q}(\cdot) = (q_0(\cdot), \vec{q}(\cdot))$ is the mapping to quaternion, and $\bar{q}^*$ denotes conjugate of $\bar{q}$. Dynamics of the Cosserat rod model can be derived as partial differential equation from and can be numerically solved [15]. However, since the complementarity condition resulting from the contact situation is neither explicitly determined nor can be differentiated, it is difficult to simulate a operation scenario by continuous formulation. Thus, we apply spatial discretization to the rod model.

B. Passive Midpoint Integration

We adopt passive midpoint integration (PMI) [8] for dynamic integration scheme which is derived to enforce passivity property of the mechanical systems in discrete-time domain. By considering energy relation directly, PMI induces superior energy-preserving behavior than other implicit schemes without additional iteration step. For more details, see [8]. The maximal coordinate expression of the PMI is given by:

$$\begin{aligned} M\frac{V_{k+1} - V_k}{T_k} + C\hat{V}_k + B\hat{V}_k &= -d\psi_k^T + F_k + J_{ct}^T\lambda_{ct} \\ \hat{V}_k &= \frac{V_k + V_{k+1}}{2} \end{aligned} \tag{1}$$

where $M, C, B \in \mathbb{R}^{6n\times 6n}$ is the mass, Coriolis, damping matrix with n bodies, T_k is the time step size, $V_k = [v_k; \omega_k] \in \mathbb{R}^{6n}$ with v_k and w_k are linear and angular velocity, $\hat{V}_k \in \mathbb{R}^{6n}$ is the representative velocity, $F_k \in \mathbb{R}^{6n}$ is the external force at the k-th step, and $J_{ct} \in \mathbb{R}^{3n_c\times 6n}, \lambda_{ct} \in \mathbb{R}^{3n_c}$ is the contact Jacobian and impulse while n_c is the number of contact. Also here, $d\psi_k^T$ is the potential action which can be derived from

$$d\psi_k^T\hat{V}_kT_k = \frac{d\psi_k}{dx_k}\hat{V}_kT_k + \frac{1}{2}\hat{V}_k^T\frac{d^2\psi_k}{dx_k^2}\hat{V}_kT_k^2 \tag{2}$$

where ψ_k is a potential energy at the k-th step. The right hand side of (2) is in fact the second-order approximation of $\psi_{k+1} - \psi_k$.

III. Dynamics of Discretized Cable

A. Segmentation and Discrete Strain Energy

To consider all general motion of the cable (i.e., extension, shear, bending, torsion), we model flexible cable as n serially articulated 6-DOF rigid cylindrical-link. Each j-th link segment is expressed by the global position, orientation, frame vector which denoted as p_j, R_j, d_j, and the motion of the cable can be described as relative behavior of the consecutive segments:

$$\begin{aligned} e_{1,j} &= d_j - R_j^T(p_{j+1} - R_{j+1}d_{j+1} - p_j) \\ e_{2,j} &= \text{Vec}(\bar{q}^*(R_j)\cdot\bar{q}(R_{j+1})) \end{aligned}$$

where $e_{1,j} \in \mathbb{R}^3$ is the constraint error for the extension, shear, $e_{2,j} \in \mathbb{R}^3$ is the constraint error for the bending, torsion derived from finite difference [16], $q(\cdot)$ maps SO3 to quaternion and Vec$(\cdot)$ represents the vector part of the quaternion. Then the discrete strain energies are given by

$$\psi_e = \frac{1}{2}{e_1}^T[K_1]e_1 + \frac{1}{2}e_2^T[K_2]e_2 = \frac{1}{2}e^T[K_e]e \tag{3}$$

where ψ_e is total strain energy, $e_m = [e_{m,1}; \cdots ; e_{m,n}] \in \mathbb{R}^{3n}$, $[K_m] = \text{diag}(K_m, \cdots, K_m) \in \mathbb{R}^{3n\times 3n}$ for $m = 1, 2$, $e = [e_1; e_2] \in \mathbb{R}^{6n}$ and $[K_e] = \text{diag}(K_e, \cdots, K_e) \in \mathbb{R}^{6n\times 6n}$. Here, gain matrix K_1, K_2 are derived from the Cosserat model:

$$K_1 = \text{diag}(GA/L, GA/L, EA/L)$$
$$K_2 = \text{diag}(EI/L, EI/L, GJ/L)$$

where E, G is the Young's modulus, torsion modulus of the cable, A, L is a cross section area and a length of

each discretized segment. In this paper, we use discrete element as cylinder so geometric parameters are defined as $A = \pi r^2, I = 0.25\pi r^4, J = 0.5\pi r^4$ where r is the radius of cylinder.

B. Dynamics Formulation of Discretized Cable

Based on the discrete strain energy of the total cable (3), the potential action for the PMI formulation (2) can be written as

$$d\psi_e^T = J_e^T[K_e](e + \frac{1}{2}J_e\hat{V}_k T_k) = J_e^T\lambda_e \tag{4}$$

where $\lambda_e \in \mathbb{R}^{6n}$ is the potential action impulse, $J_e \in \mathbb{R}^{6n\times 6n}$ is the Jacobian matrix which maps V_k to $\frac{de}{dt}$ obtained by derivative on Lie groups. Here, outer product approximation of Hessian (i.e., $\nabla^2\psi_e \approx J_e^T[K_e]J_e$) is used and by this approximation, we can obtain a linear relation between λ_e and $\hat{V}$ which will further be used in subsystem solver in Sec. IV. By applying (4) to (1), we can simplify the dynamic equation as

$$\begin{aligned} &A_k\hat{V}_k = B_k + J_{ct}^T\lambda_{ct} \\ &A_k = \frac{2M}{T_k} + C + B + \frac{T_k}{2}J_e^T[K_e]J_eT_k \in \mathbb{R}^{6n\times 6n} \\ &B_k = -J_e^T[K_e]e + F_k \in \mathbb{R}^{6n} \end{aligned} \tag{5}$$

Here, A_k is invertible from the positive definite property.

C. Direct Solver

To solve $\hat{V}$ with λ_{ct} directly from (5), contact form equation is derived as

$$J_{ct}\hat{V}_k = J_{ct}A_k^{-1}J_{ct}^T\lambda_{ct} + J_{ct}A_k^{-1}B_k \tag{6}$$

with NCP conditions:

$$\begin{aligned} &0 \le \lambda_{ct_c}^n \perp J_{ct_c}^n\hat{V} + \epsilon_c \ge 0 \\ &\lambda_{ct_c}^t \in FC(\lambda_{ct_c}^n) \\ &\textbf{if } \lambda_{ct_c}^n = 0 \textbf{ then } \left\|\lambda_{ct_c}^t\right\| = 0 \textbf{ (open)} \\ &\textbf{else if } \lambda_{ct_c}^t \in FC_I(\lambda_{ct_c}^n) \textbf{ then } \left\|J_{ct,c}^t\hat{V}\right\| = 0 \textbf{ (stick)} \\ &\textbf{else } \lambda_{ct_c}^t / \left\|\lambda_{ct_c}^t\right\| = -J_{ct_c}^t\hat{V} / \left\|J_{ct_c}^t\hat{V}\right\| \textbf{ (slip)} \end{aligned} \tag{7}$$

for $^\forall c = 1 \sim n_c$ where n, t denotes the normal and the tangential components of the contact, ϵ is for penetration compensation term and FC stands for the friction cone set with subscript I the inside, not including the boundary. Conditions in (7) can be relaxed as several way (e.g., penalty-based [17], bounded linearized model [18], etc.), however for accurate dry-frictional behavior, we have taken all the conditions that the physical contact should have. From (6) and (7), λ_{ct} can be obtained by widely-used projected Gauss-Seidel (PGS) method [9] and then $\hat{V}$ can be directly calculated from (5).

However, applying direct solve to elastic object manipulation scenario is in many case intractable since we encounter a large size matrix calculation (e.g., Delassus operator $J_{ct}A_k^{-1}J_{ct}^T$) for every time step while dealing with large sized coupled contact problem (6) which takes considerable time.

IV. Parallelized Iterative Solver for Flexible Cable Simulation

In Sec. IV, we propose the novel subsystem-based parallelized iterative solver to speed up solving dynamic formulation in Sec. III-B. We subdivide the whole system, formulate the dynamics of each subsystem in parallel, and integrate the coupling and parallelized contact solver through iteration. Since our strategy is based on direct solve for each subsystem, it is better in terms of accuracy than other iteration-based methods and is less affected by the number of iterations or time step size.

A. Generalized Contact Model

In most simulation, Coulomb friction model is widely used for its simplicity and convexity. However, it is known that Coulomb friction model does not fit well in polymers which usually constitute flexible cables [19]. Therefore, in this paper, we adopt generalized friction model:

$$\| \lambda_{ct_c}^t \| \le \kappa|\lambda_{ct_c}^n|^N \tag{8}$$

for $^\forall c = 1 \sim n_c$ where κ, N depends on the material. While Coulomb friction model use $N = 1$, N for polymer usually exists in the range $0.67 \sim 1$ [20]. In this case, FC has non-conical shape, which means certain schemes based on conical shape [21], [22] are not applicable. Thus, to solve NCP in generalized contact model, we slightly modified friction cone projection step of PGS algorithm based on (8).

B. Subsystem Modeling

Each subsystem consists of a set of consecutive segments as shown in Fig. 2. Since dynamic coupling exists only via the connection between the two subsystems, the dynamics of each i-th subsystem is given by:

$$A_i\hat{V}_i = B_i + J_{cp_i^i}^T\lambda_{cp_i} + J_{cp_{i-1}^i}^T\lambda_{cp_{i-1}} + J_{ct_i}^T\lambda_{ct_i} \tag{9}$$

where A_i, B_i are the subsystem dynamic matrix, λ_{cp_i} is the coupling impulse between i-th and $(i+1)$-th subsystem, $J_{cp_i^{i'}}$ is the Jacobian for coupling λ_{cp_i} corresponding to i'-th subsystem. Here we omit notation for k-th time step for simplification, however all the components of (9) are varies according to the time step. Despite A_i, B_i are computed based on (5), since we exclude terms relate to λ_{cp} and λ_{ct} in B_i, each subsystem dynamic matrix only contain its 'internal' properties. Therefore, we can obtain all of the component (i.e., A_i,B_i,$J_{cp,i}$,$J_{ct,i}$) independently for each subsystem.

At the same time, we can use coupling action between adjacent subsystem from (4) which can be written as

$$\lambda_{cp_i} = -K_{cp}e_{cp_i} - \frac{1}{2}K_{cp}(J_{cp_i^i}\hat{V}_i - J_{cp_i^{i+1}}\hat{V}_{i+1})T_k \tag{10}$$

where $K_{cp} = K_e \in \mathbb{R}^6$ is the coupling gain matrix. By substituting (9) into (10) for all coupling indices, linear relation between the entire coupling impulse λ_{cp} and the entire contact impulse λ_{ct} can be obtained:

$$\begin{aligned} &S\lambda_{cp} = -C\lambda_{ct} + U \\ &\lambda_{cp} = [\lambda_{cp_1}; \cdots ; \lambda_{cp_{n_s-1}}] \in \mathbb{R}^{6(n_s-1)} \\ &\lambda_{ct} = [\lambda_{ct_{i1}}; \cdots ; \lambda_{ct_{in_c}}] \in \mathbb{R}^{3n_c} \end{aligned} \tag{11}$$

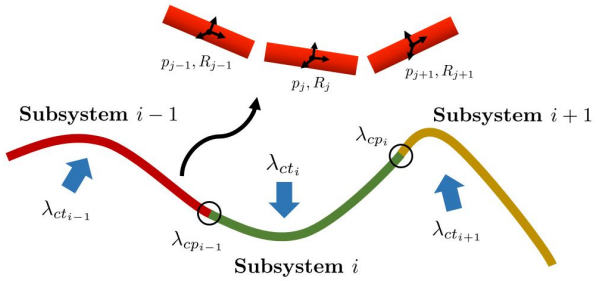

Fig. 2: Discretization structure of the cable: subsystems (i-index) and segments (j-index).

where n_s is the number of subsystem, ic is the node index for c-th contact and $S \in \mathbb{R}^{6(n_s-1)\times 6(n_s-1)}, C \in \mathbb{R}^{6(n_s-1)\times 3n_c}, U \in \mathbb{R}^{6(n_s-1)}$ is derived from substitution. Here, detailed derivation is omitted for the page limit but the form of S can be expressed as follow:

$$S = \begin{bmatrix} S_1^1 & S_1^2 & & \\ S_2^1 & \ddots & \ddots & \\ & \ddots & \ddots & S_{n_s-2}^{n_s-1} \\ & & S_{n_s-1}^{n_s-2} & S_{n_s-1}^{n_s-1} \end{bmatrix} \tag{12}$$

$$S_i^i = \frac{2}{T_k} K_e^{-1} + J_{cp_i^i} A_i^{-1} J_{cp_i^i}^T + J_{cp_i^{i+1}} A_{i+1}^{-1} J_{cp_i^{i+1}}^T$$
$$S_i^j = J_{cp_i^i} A_i^{-1} J_{cp_i^j}^T$$

If there is no contact, coupling impulse can be obtained by solving $\lambda_{cp} = -S^{-1}U$ which is similar to Schur-complement based scheme [23] but with a primal-based formulation [24] for each subsystem. When we take a large number of subsystem, linear solve of S becomes heavy and in multi-contact situation, solving NCP (7) cannot be basically decoupled with (11). Due to this difficulty, λ_{cp} and λ_{ct} cannot be solved at the same time and all contact situations must be resolved at once (i.e., not parallelizable).

C. Parallel Contact Solver

To embrace multi-contact situation in our solver, we utilize iteration scheme. We first modify (11) as

$$(S_d + S_l)\lambda_{cp}^l = -S_u \lambda_{cp}^{l-1} - C\lambda_{ct}^l + U \tag{13}$$

where λ^l denotes impulse at the l-th iteration step, S_d, S_l, S_u is the block diagonal, the strictly block lower triangular, and the strictly block upper triangular part of the S. Solving (13) is one step block Gauss-Seidel iteration which only need inversion of S_i^i which can be easily computed in parallel. Also, to obtain λ_{ct}^l before solving (13), we also revise (9) for each subsystem as

$$A_i \hat{V}_i = B_i + J_{cp_i^i}^T \lambda_{cp_i}^{l-1} + J_{cp_{i-1}^i}^T \lambda_{cp_{i-1}}^{l-1} + J_{ct_i}^T \lambda_{ct_i}^{l-1} \tag{14}$$

Projecting (14) to contact space, contact formulation of each subsystem can be written as

$$J_{ct,i}\hat{V}_i = J_{ct_i} A_i^{-1} J_{ct_i}^T \lambda_{ct_i}^l + f(\lambda_{cp_{i-1}}^{l-1}, \lambda_{cp_i}^{l-1})$$
$$\textbf{proj. to NCP conditions as (7)} \tag{15}$$

where f is the simplified notation of linear mapping. Here, since we know λ_{cp}^{l-1} from previous iteration step, (15) of each i-th subsystem is completely decoupled which means parallel computing can be applied. For each subsystem, single iteration of PGS algorithm described in Sec. IV-A which updates $\lambda_{ct_i}^{l-1}$ to $\lambda_{ct_i}^l$ is performed in parallel. Although single iteration is used here, we find that reliable contact behavior can be obtained because it is repeated continuously in the l-loop (i.e., contact solver loop). After obtaining $\lambda_{ct_i}^l$, λ_{cp}^l can be updated through (13).

In the strategy as above, matrix used in the l-loop (i.e., sub-size Delassus operator $J_{ct,i}A_i^{-1}J_{ct,i}^T$, inverse of S_i^i) can be pre-computed in the outside of the loop. Therefore, the process contained in the inner loop becomes very light. Also we use warm start that initialize $\lambda_{cp}, \lambda_{ct}$ as previous simulation step value. This scheme is similar to [25] called iterative constraint anticipation (ICA) that it block splits A_k and iterates between V_{k+1} and λ_{ct}. However, our algorithm does not have to calculate full size of A_k which demands large size matrix multiplication and uses λ_{cp} rather than V_{k+1} for iteration, so the handling size is smaller.

Despite the l-loop process adopt Gauss-Seidel method, convergence of the algorithm cannot easily guaranteed since it is unified with PGS iteration which induces non-linear relation between λ_{cp} and λ_{ct}. In order to achieve benefit in terms of convergence, we proceeded following PGS post-regulation:

$$\textbf{if } (\parallel C(\lambda_{ct}^l - \lambda_{ct}^{l-1}) \parallel \geq \xi\alpha^l) \textbf{ then}$$
$$\textbf{do } \lambda_{ct}^l = \lambda_{ct}^{l-1} \tag{16}$$

where $\xi > 0, 0 < \alpha < 1$ is the hyperparameter which bounds deviation of λ_{ct} from previous step. By restricting bouncing deviation of λ_{ct}, this post-regulation ensures 'noise' in the Gauss-Seidel iteration to be bounded. Although (16) is a condition that cannot be easily interpreted physically, we confirm through simulation results that it is sufficiently valid. Also based on this, convergence of λ_{cp} in our l-loop process can be shown as Theorem 1. Pseudo-code of the whole algorithm is specified in Algorithm 1.

Lemma 1 $\parallel (S_d + S_l)^{-1} S_u \parallel < 1$ *can be ensured by taking small enough* T_k.

Proof: Considering (12), we can write as

$$S_{dl} = \frac{2}{T_k}[K_e^{-1}] + S_{dl}^{'} = F^{-1} + S_{dl}^{'}$$

where $S_{dl} = S_d + S_l$, $F = \frac{T_k}{2}[K_e]$ since K_e is invertible and $S_{dl}^{'}$ is the residual part of the matrix which is independent of T_k. Then such inequality can be hold:

$$\begin{aligned} \parallel S_{dl}^{-1} S_u \parallel &\leq \parallel (F^{-1} + S_{dl}^{'})^{-1} \parallel\parallel S_u \parallel \\ &\leq \parallel F \parallel\parallel (I + FS_{dl}^{'})^{-1} \parallel\parallel S_u \parallel \end{aligned}$$

Since S_u is also independent of T_k, by reducing the value of T_k, we can only reduce $\parallel F \parallel$ directly. Applying Lemma 2.3.3. in [26], we can get

$$\parallel F \parallel\parallel (I + FS_{dl}^{'})^{-1} \parallel\parallel S_u \parallel \leq \frac{\parallel F \parallel\parallel S_u \parallel}{1 - \parallel F \parallel\parallel S_{dl}^{'} \parallel}$$

and by reducing $\parallel F \parallel$, right side of the inequality can be less than 1, which showing that given lemma hold. ■

Algorithm 1 Subsystem-based Parallelized Iterative Solver

while simulation loop **do**
  for each subsystem i **do** in parallel
    Compute $A_i, A_i^{-1}, B_i, J_{cp_i}$
    Collision detection with J_{ct_i}
  end for
  Compute S, C, U to construct (11)
  for each subsystem i **do** in parallel
    Compute $J_{ct_i} A_i^{-1} J_{ct_i}^T$
    Compute inverse of S_i^i
  end for
  Initialize λ_{cp},λ_{ct} as previous simul. loop value
  while contact loop **do**
    $l \leftarrow l + 1$
    for each subsystem i **do** in parallel
      Compute $\lambda_{ct_i}^l$ from single PGS iter (15)
    end for
    PGS post-regulation using (16)
    Block Gauss-Seidel update using (13)
    if $\| \lambda_{cp}^l - \lambda_{cp}^{l-1} \| < \epsilon$ or $l > l_{max}$ **then**
      break
    end if
  end while
  Update each subsystem using (9) in parallel
end while

Theorem 1 *Convergence of λ_{cp} in Algorithm 1 can be ensured by taking small enough T_k.*

Proof: Let us define as $\| S_{dl}^{-1} S_u \| = \gamma$. By Lemma 1, $\gamma < 1$ holds for $^{\exists}T_k$. Then from (13),(16), we can get

$$\begin{aligned}\Lambda_l &= \| -S_{dl}^{-1} S_u (\lambda_{cp}^{l-1} - \lambda_{cp}^{l-2}) - S_{dl}^{-1} C (\lambda_{ct}^{l} - \lambda_{ct}^{l-1}) \| \\ &\leq \gamma \Lambda_{l-1} + \xi' \alpha^l\end{aligned}$$

where $\Lambda_l = \| \lambda_{cp}^l - \lambda_{cp}^{l-1} \|$, $\xi' = \| S_{dl}^{-1} \| \xi$. By repeating such inequality, following is derived:

$$\Lambda_l \leq \gamma^{l-1} \Lambda_1 + \xi' \alpha (\sum_{i=0}^{l-1} \alpha^i \gamma^{l-i-1})$$

By taking β that satisfies $\alpha, \gamma < \beta < 1$, we can finally get

$$\Lambda_l < \gamma^{l-1} \Lambda_1 + \xi' l \beta^l$$

which directly shows Λ_l converges to 0 as $l \to \infty$. ■
Although sufficiently small T_k is demanded for our proof, we observe that convergence is still possible even for large $T_k (\sim 0.02s)$ for typical simulation.

D. Additional Details and Expansions

1) Collision detection: Since we model each segment in a cylindrical shape, the environment and self-collision composed of primitive shapes can be easily implemented with some hashing algorithms. Contact with more complex figures (e.g., mesh) has not been implemented yet, but we believe it can be included without difficulty using algorithm like [27].

2) Self-collision: In the case of self-collision (e.g., knot-tying), knowing all the coupling impulse between subsystems does not imply that all subsystem dynamics can be calculated in a decoupled state. To deal with this, we utilize an adaptive subsystem division scheme that binds the segments where self-contact occurred to the same subsystem.

3) Coupling with manipulator dynamics: Manipulator dynamics expressed in generalized coordinates can be conveniently included in our framework by considering the manipulator dynamics as a single subsystem with Jacobian mapping of joint-to-Cartesian coordinate. In this paper, we use the passivity-based Lagrangian dynamics integration method proposed in [8].

4) Non-linearized dynamics: Although we have developed expressions on PMI based linearized dynamics so far, our method can also be used for other projected Newton type dynamics [28] integration. By constructing multiple linearized dynamics (i.e., Newton iterations) at each time step, simulation stability and accuracy can be further increased.

V. Simulation and Experiment

A. Parameter Identification

To identify the physical parameters (i.e., Young's modulus E, Poisson's ratio ν, frictional attribute κ and N), we develop simple, efficient way which is based on Capstan equation [19]. The equation describes relation between the friction and the tension of the cable and can be written as

$$T = T_0 e^{\mu \theta}$$

where T_0, T is the tension applied in one end and the minimum tension that must be applied to the other end to move the cable, μ is the friction coefficient, θ is the winded angle. However, since this relation is derived from the assumption that the cable does not extend, we adopt modified Capstan equation (MCE) proposed in [19] which considers extension with nonlinear friction (8) and represent as ordinary differential equation (ODE) form. For full equation and derivation, see [19].

We conduct experiment to collect (T_0, T) for various θ, T_0 conditions: put a mass on one end of the cable, wind it around a cylinder, and gradually increase the tension of the other end using robot arm (FRANKA EMIKA Panda) impedance control [29] while tension is measured using force/torque (F/T) sensor (ATI Gamma). After collecting sufficient number of data, the parameters that best match the theoretical results are calculated. Here, we use MATLAB ode45 solver to solve MCE numerically and conduct non-linear optimization using MATLAB fmincon solver. As a result, we obtain the values as $E = 4.462$ MPa, $\nu = 0.5012, \kappa = 1.296, N = 0.7616$.

B. Simulation

Using simulation scheme described above, we simulate cable winding operation around cylinder using robot arm. The task is designed to satisfy the constant desired pitch while extending the cable. The parameters obtained from Sec. V-A are used, and radius, length of the cable is $0.002m, 0.4m$. We also use $40 \sim 160$ as segments number, $8 \sim 20$ as subsystem number, 10ms as time step and 100 as max iteration number. We use Intel Core i5-7500 CPU 3.40GHz (Quad-Core),

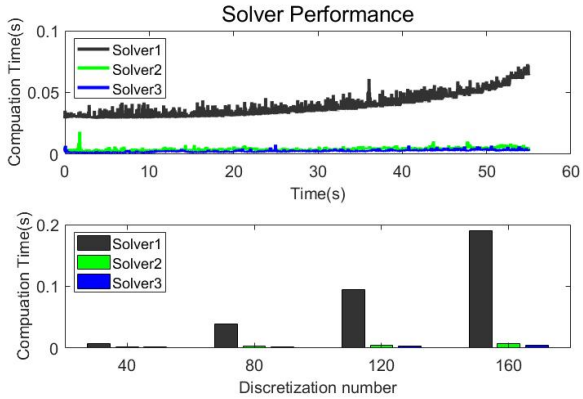

Fig. 3: Solver performance comparison in simulation. Computation speed over simulation time flows(upper) and average computation time per step along discretization number(lower).

(a) 80 segments discretization

	Avg. time(ms)	Avg. iter	Max. iter
Solver 1	38.515	-	-
Solver 2	3.4606	60.550	2107 / 5500
Solver 3	2.5928	31.762	0 / 5500

(b) 160 segments discretization

	Avg. time(ms)	Avg. iter	Max. iter
Solver 1	189.99	-	-
Solver 2	7.2411	71.2535	3151 / 5500
Solver 3	4.6810	30.5344	0 / 5500

TABLE I: Solver Performance Comparison

OpenGL as rendering tool, C++ Eigen as matrix computation library and C++ OpenMP as parallelization library.

We compare three solvers: 1) classical direct solver in Sec. III-C (Solver1) 2) our solver without PGS post-regulation (Solver2) 3) our solver with PGS post-regulation (Solver3) and the result is depicted in Fig. 3 and Table I. The results clearly shows that our method is superior in computation efficiency compared to the direct solver. Computation time is able to speed up to $20 \sim 40$ times and this performance gap increases along discretization number which means our algorithm has preferable scalability. Note that even for 160 segments (960 DOF), our simulation scheme can be performed in real-time. It is also demonstrated that our post-regulation strategy affects to the decrease in iteration number and computation time, while ensuring convergence so that the maximum number of iterations is not reached. We also find that the decrease in speed as increasing number of contacts is smaller in our solver than in direct solver.

C. Comparison with Experiment

To verify the consistency of our simulation with reality, experiment on the same task in the real environment is performed. For performance evaluation, we measure the force applied to the end effector of the robot arm during winding operation task using F/T sensor. Here, we utilize admittance control [29] with very high gain and use velocity control as low level control for exact tracking of the desired trajectory. Experiment snapshot is depicted in Fig. 1. To compare with our solver, we also implement the extended position based dynamics (XPBD) solver [30] with material

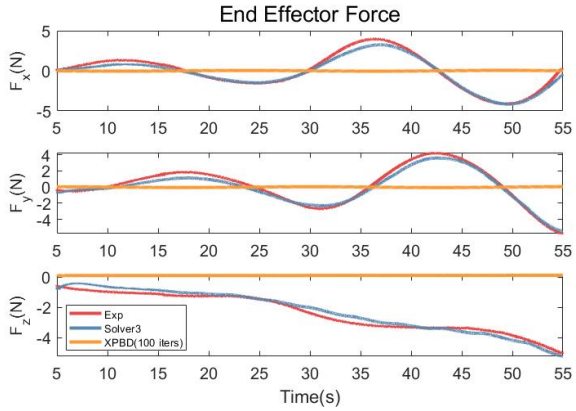

Fig. 4: End effector force comparison while winding operation.

Fig. 5: Configuration after conducting winding process. Experiment, Solver3, and the XPBD solver from left to right.

parameters which is widely used in the deformable object simulation in the field of graphics.

As a result shown in Fig. 4 and Fig. 5, our simulation result fits well with the experimental force result (RMS error x : 0.3519N, y : 0.3981N, z : 0.2465N) and also tracks desired pitch well, while the XPBD result shows unreliable performance as the cable is not extended but drooped. This is because 1) the XPBD's iteration does not sufficiently converge along strong gain to follow the desired trajectory (even with 100 iterations for all steps); 2) loss of accuracy by modeling contact with constraint potential. However, it is validated that iteration and regulation schemes applied in our subsystem-based parallel solver well maintain the physical conditions.

This implies that simulator for visually plausible results do not always correspond to physical accuracy and suggest that methods with both speed and accuracy, like our solver, are important in areas such as robot manipulation that require physical correct behavior.

VI. Conclusion

In this paper, we present a novel real-time simulation method of long flexible cable manipulation with physical accuracy. We model the cable as articulated rigid link segments, each with complementarity-based contact model and inter-segment compliant coupling based on analytical theory. To accelerate the simulation of this large size, complicated model, we subdivide the entire system and solve them in parallel using our novel Gauss-Seidel based iteration for inter-subsystem consistency and post-regulation for convergence speed up. Both simulation and experiment is performed for verification of the proposed algorithm. Some future works can be discussed as follow: 1) expansion to general deformable object; 2) manipulation performance improvement based on reinforcement learning in simulation environment.

REFERENCES

[1] Y. She, S. Wang, S. Dong, N. Sunil, A. Rodriguez, and E. Adelson. Cable manipulation with a tactile-reactive gripper. In *Robotics: Science and Systems*, 2020.

[2] S. Duenser, R. Poranne, B. Thomaszewski, and S. Coros. Robocut: hot-wire cutting with robot-controlled flexible rods. *ACM Transactions on Graphics*, 37(4), 2020.

[3] J. Sanchez, J. Corrales, B. Bouzgarrou, and Y. Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: A survey. *International Journal of Robotics Research*, 37(7):688–716, 2018.

[4] Andrychowicz OM., Baker B., Chociej M., and et al. Learning dexterous in-hand manipulation. *International Journal of Robotics Research*, 39(1):3–20, 2020.

[5] D. Son, H. Yang, and D. J. Lee. Sim-to-real transfer of bolting tasks with tight tolerance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 9056–9063, 2020.

[6] C. Horak and C. Trinkle. On the similarities and differences among contact models in robot simulation. *IEEE Robotics and Automation Letters*, 4(2):493–499, 2019.

[7] M.B. Rubin. *Cosserat Theories: Shells, Rods and Points*. Springer, 2013.

[8] M. Kim, Y. Lee, Y. Lee, and D. J. Lee. Haptic rendering and interactive simulation using passive midpoint integration. *International Journal of Robotics Research*, 36(12):1341–1362, 2017.

[9] F. Jourdan, P. Alart, and M. Jean. A gauss-seidel like algorithm to solve frictional contact problems. *Computer Methods in Applied Mechanics and Engineering*, 15(1-2):31–47, 1998.

[10] *Vortex studio*. https://www.cm-labs.com/vortex-studio/.

[11] *Bullet physics engine*. https://pybullet.org/wordpress/.

[12] J. Matas, S. James, and A. J. Davison. Sim-to-real reinforcement learning for deformable object manipulation. pages 734–743, 2018.

[13] O. Goury and C. Duriez. Fast, generic, and reliable control and simulation of soft robots using model order reduction. *IEEE Transactions on Robotics*, 34(6):1565–1576, 2018.

[14] R. Jangir, G. Alenyà, and C. Torras. Dynamic cloth manipulation with deep reinforcement learning. In *IEEE International Conference on Robotics and Automation*, pages 4630–4636, 2020.

[15] J. Till, V. Aloi, and C. Rucker. Real-time dynamics of soft and continuum robots based on cosserat rod models. *International Journal of Robotics Research*, 38(6):723–746, 2019.

[16] J. Spillmann and M. Teschner. Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 63–72, 2007.

[17] K. Yamane and Y. Nakamura. Stable penalty-based model of frictional contacts. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1904–1909, 2006.

[18] K. Erleben. Velocity-based shock propagation for multibody dynamics animation. *ACM Transactions on Graphics*, 26(2):2222–2234, 2007.

[19] J. Jung, N. Pana, and T. Kang. Generalized capstan problem: Bending rigidity, nonlinear friction, and extensibility effect. *Tribology International*, 41(6):524–534, 2008.

[20] X. Gao, L. Wang, and X. Hao. An improved capstan equation including power-law friction and bending rigidity for high performance yarn. *Mechanism and Machine Theory*, 90:84–94, 2015.

[21] J. Hwangbo, J. Lee, and M. Hutter. Per-contact iteration method for solving contact dynamics. *IEEE Robotics and Automation Letters*, 3(2):895–902, 2018.

[22] A. Tasora and M. Anitescu. A matrix-free cone complementarity approach for solving large-scale, nonsmooth, rigid body dynamics. *Computer Methods in Applied Mechanics and Engineering*, 200(5):439–453, 2011.

[23] A. Peiret, S. Andrews, J. Kövecses, P. G. Kry, and M. Teichmann. Schur complement-based substructuring of stiff multibody systems with contact. *ACM Transactions on Graphics*, 38(5):493–499, 2019.

[24] M. Macklin, K. Erleben, M. Müller, S. Chentanez, N.and Jeschke, and T. Y. Kim. Primal/dual descent methods for dynamics. *Computer Graphics Forum*, 39(8):89–100, 2020.

[25] A. Otaduy, R. Tamstorf, D. Steinemann, and M. Gross. Implicit contact handling for deformable objects. *Computer Graphics Forum*, 28(2):559–568, 2009.

[26] G. H. Golob and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 2013.

[27] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M. P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino. Collision detection for deformable objects. *Computer Graphics Forum*, 24(1):61–81, 2005.

[28] J. Teran, E. Sifakis, G. Irving, and R. Fedkiw. Robust quasistatic finite elements and flesh simulation. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 181–190, 2005.

[29] M. W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot modeling and control*. Wiley, 2006.

[30] M. Macklin, M. Müller, and N. Chentanez. Xpbd: position-based simulation of compliant constrained dynamics. In *International Conference on Motion in Games*, pages 49–54, 2016.