# Autonomous Steering of Concentric Tube Robots via Nonlinear Model Predictive Control

**Citation for published version:**
Khadem, SM, O'Neill, J, Mitros, Z, Da Cruz, L & Bergeles, C 2020, 'Autonomous Steering of Concentric Tube Robots via Nonlinear Model Predictive Control', *IEEE Transactions on Robotics*. https://doi.org/10.1109/TRO.2020.2991651

**Digital Object Identifier (DOI):**
10.1109/TRO.2020.2991651

**Link:**
Link to publication record in Edinburgh Research Explorer

**Document Version:**
Peer reviewed version

**Published In:**
IEEE Transactions on Robotics

# Autonomous Steering of Concentric Tube Robots via Nonlinear Model Predictive Control

Mohsen Khadem, John O'Neill, Zisos Mitros, Lyndon da Cruz*, Christos Bergeles*

***Abstract*—This paper presents a Model Predictive Controller (MPC) developed for the autonomous steering of concentric tube robots (CTRs). State-of-the-art CTR control relies on differential kinematics developed by local linearization of the CTR's mechanics model and cannot explicitly handle constraints on robot's joint limits or unstable configurations commonly known as snapping points. The proposed nonlinear MPC explicitly considers constraints on the robot configuration space (*i.e.* joint limits) and the robot's workspace (*i.e.* mixed boundary conditions on robot curvature). Additionally, the MPC calculates control decisions by optimizing the model-based predictions of future robot configurations. This way, it avoids configurations it cannot recover from, joint limits, singular configurations and snapping. The proposed controller is evaluated via simulations and experimental studies with a variety of trajectories of increasing complexity. Simulation results demonstrate the capability of MPC to avoid singularities while satisfying robot mechanical constraints. Experimental results demonstrate that our solution enables following of trajectories unattainable by state-of-the-art controllers with mean error corresponding to $1\%$ of robot arclength.**

***Index Terms*—Model Predictive Control, Continuum Surgical Robots.**

## I. Introduction

A concentric tube robot (CTR) is a continuum robot comprising a series of precurved elastic tubes. Each tube can be axially translated and rotated with respect to the others to control the robot shape and tip pose. CTRs can traverse confined spaces and manipulate objects in complex environments, similar to complementary surgical continuum robots. Deployment of CTRs inside the body requires precise control of robot motion.

The kinematic model that is widely used to control the CTR was derived in [1], [2]. The model estimates the robot shape as a function of tubes' base translation, rotation, and known external forces, via a set of differential equations with boundary conditions split between the base and tip of the robot. Iterative numerical approaches such as shooting methods can solve the boundary value problem (BVP) and estimate the robot shape [1], [2].

Dupont *et al.* used Fourier Series to approximate the inverse kinematics of the robot for open-loop control [1]. Gilbert *et al.* proposed a follow-the-leader approach by estimating deployment sequences that would lead the robot body to follow the position of its tip [3]. Using these approaches, CTRs have been successfully employed with open loop control in several clinical scenarios [4], [5], in which an operator directly controls the robot tip. However, in general, the challenges in introducing the robot up to the location of interest are purposefully kept minimal. To enable automatic deployment of CTRs through complex tortuous anatomy, accurate closed loop control of the CTR's end-effector is essential. Then, assuming follow the leader architectures [3], [6] and safe intraluminal deployment, the shape will also conform to anatomical constraints.

Due to the complexity of the BVP model of CTR, calculating the inverse kinematics of the robot requires tailored computationally

This research was supported by an ERC Starting Grant [714562].

M. Khadem (Corresponding author: mohsen.khadem@ed.ac.uk) is now with the School of Informatics, University of Edinburgh, UK. J. O'Neill, Z. Mitros, and C. Bergeles are with the School of Biomedical Engineering and Imaging Sciences, King's College London, London, UK. L. da Cruz is with Moorfields Eye Hospital, London, UK. * denotes equal contribution.

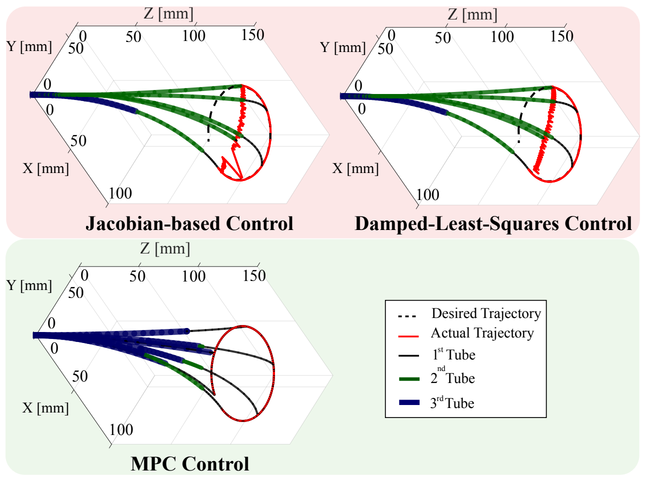


Fig. 1. Comparison of the designed controller with the state-of-the-art.

efficient methodologies. A straightforward approach is to approximate the Jacobian of the robot by finite differences on the solution of the BVP problem, *e.g.* [7]. Rucker *et al.* [8] proposed a numerical approach to increase the computational efficiency of Jacobian estimation for a CTR under external loading. Xu *et al.* [9] proposed real-time estimation of the robot Jacobian by discretizing the robot into small sublinks. The Jacobian was used for closed-loop control of the CTR's tip. Wu *et al.* [10] presented a model-less method for real-time estimation of the robot Jacobian via visual servoing. The Jacobian was then used to control the position of the CTR. Kudryavtsev *et al.* [11] developed a CTR with an embedded camera at the robot's tip. They used the linear approximation of the CTR Jacobian and visual feedback for real-time steering.

Most recent research in CTRs has focused on their open-loop steering [1], [3]. State-of-the-art closed-loop control strategies rely on linear approximation of robot inverse kinematics near equilibriums [9], [12] but neglect the system's non-linearities, potentially leading to instability and reduced controller accuracy. Recently, researchers proposed application of dynamic active constraints that could improving the safety and stability also of CTRs [13].

It should be noted that the CTR's workspace is constrained by the tubes' pre-curvature, range of linear translation, and nonholonomic path-dependent constraints on robot motion in terms of tubes' end curvatures. These constraints make the tip trajectory and joint values dependent on their traversed path [14], [15]. Therefore, control methods that only use contemporary information may lead the robot to instabilities or configurations that it cannot recover from (see Fig. 1).

In this paper, we design a novel controller for steering of CTR's end-effector. Unlike current CTR controllers [9], [11] that employ the local robot Jacobian, the Model Predictive Controller (MPC) uses future predictions of robot behaviour to handle explicit constraints on the robot state and control inputs, therefore being able to steer the robot away from unstable configurations commonly known as "snapping" points. Snapping is a mechanical instability in CTRs caused by the release of the elastic potential energy that is accumulated due to tube bending and twisting [1].

Fig. 1 shows a comparison of the MPC controller proposed in this paper with state-of-the-art Jacobian-based controllers. The simulation results show that local controllers (*e.g.* Jacobian-based or damped-least-squares inverse kinematics) blindly steer the robot in configurations it cannot recover from, while the proposed MPC incorporates future knowledge to follow optimal paths. The proposed controller

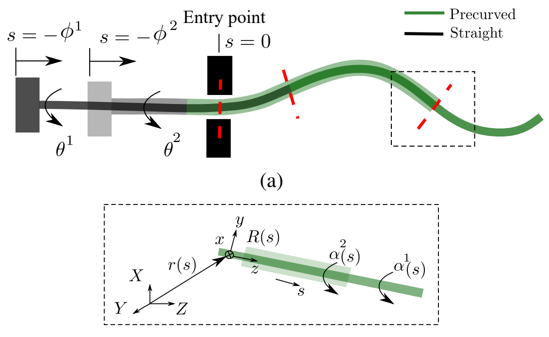


Fig. 2. Illustration of a CTR. Tubes are grasped at their proximal ends. The actuation variables $\theta^i(t)$ and $\phi^i(t)$ denote the proximal base rotation, and translation of the $i$-th tube, respectively. Each tube comprises a straight and a curved part. $\alpha^i(s)$ denotes angular displacement of tube $i$ at arclength $s$. Arc length is the length of the robot measured from the entry point, *i.e.*, $s = 0$.

fills a gap between the locally intelligent and rapid Jacobian-based controllers, and the globally intelligent but computation-heavy path-planners [7], [16]. Additionally, the proposed MPC is designed such that it only requires the solution of the CTR model as an initial value problem (IVP) rather than a BVP. The controller includes the mixed boundary conditions as constraints on the optimal solution. Implementing only the solution of the IVP problem improves computational efficiency of control strategies. We make our software available online[1].

## II. Review of CTR Mechanics Model

CTR robot behaviour is captured well by quasi-static models [1], [2]. Each concentric tube is modelled as a deformable curve with a Bishop frame attached to every point along its arclength, with the $z$-axis of the frame remaining tangent to the curve. The configuration of the robot can be defined using a unique set of 3D centroids, $\boldsymbol{r}(s,t) : [0,\ell] \times [0,\infty] \rightarrow \mathbb{R}^3 \times [0,\infty]$, and a family of orthogonal transformations, $\mathbf{R}(s,t) \;:\; [0,\ell] \times [0,\infty] \;\rightarrow\; SO(3) \times [0,\infty]$. The spatial evolution of the curvature of the robot can be defined as $\boldsymbol{u}(s,t) = (\mathbf{R}^T(s,t)\mathbf{R}'(s,t))^\vee$, where the $^\vee$ operator converts a skew-symmetric cross-product matrix to a vector in $\mathbb{R}^3$. Now, assuming the tubes are made of linear elastic isotropic materials without pre-twist, and following the approach introduced in [1], [2], we can derive the constitutive equations for calculating the instantaneous curvature of the tubes and the overall robot shape.

First, the robot is separated into transition points at which the continuity of shape and internal moment is enforced. Each segment contains up to $N$ tubes. Example transition points are indicated by dashed lines in Fig. 2. Next, we consider that the final deformed curve of all tubes at a given time $t$ must be equal to the curve of the innermost tube, *i.e.*, $\boldsymbol{r}^i(s,t) = \boldsymbol{r}^1(s,t)$, as the inner most one has the maximum extension (following the guidelines introduced in [1]). We use $\alpha^i(s,t)$ to parametrize the tubes' twist angle around $z$ axis, *i.e.*, $\mathbf{R}^i(s,t) = \mathbf{R}^1(s,t)\mathbf{R}_z(\alpha^i(s,t))$, where $\mathbf{R}_z$ denotes a rotation around the $z$ axis of the $i$-th tube due to twist. Finally, based on these assumptions, tube curvatures are calculated by

$$\boldsymbol{r}^{1'}(s,t) = \mathbf{R}^1(s,t)e3, \tag{1a}$$

$$\mathbf{R}^{1'}(s,t) = \mathbf{R}^1(s,t)\hat{\mathbf{u}}^1(s,t), \tag{1b}$$

$$u_n^i(s,t) = \left(\sum_{j=1}^{N}\mathbf{K}^j\right)^{-1}\mathbf{R}_z{}^T(\alpha^i(s,t))\left(\sum_{j=1}^{N}\mathbf{R}_z(\alpha^j(s,t))\mathbf{K}^j\,\boldsymbol{U}^j\right)\Bigg|_{n=1,2} \tag{1c}$$

$$u_3^{i\,\prime}(s,t) = \frac{E^iI^i}{G^iJ^i}(u_1^i(s,t)U_2^i - u_2^i(s,t)U_1^i), \tag{1d}$$

$$\alpha^{i\,\prime}(s,t) = u_3^i(s,t) - u_3^1(s,t), \tag{1e}$$

where superscript $i = 1, \cdots, N$ denotes the $i$-th tube, with $i = 1$ corresponding to the innermost tube; subscript $n = 1, 2, 3$ denotes the $n$-th element of a vector; $\boldsymbol{e}3 = [0,\ 0,\ 1]^T$ is the unit vector aligned with the z-axis of the global coordinate frame; $\boldsymbol{U}^i$ denotes the precurvature of each tube in its reference configuration; $\alpha'^i = u_3^i$ denotes the angle of twist about the local $z$-axis with respect to the global frame; $\mathbf{K}^i = \mathrm{diag}(E^iI^i, E^iI^i, G^iJ^i)$ is the stiffness matrix for tube $i$; $E$ is the tube's Young's modulus; $I$ is the second moment of inertia; $G$ is the shear modulus; $J$ is the polar moment of inertia.

In the absence of external forces/torques, the boundary conditions are specified in terms of tubes curvatures and actuators values by

$$\boldsymbol{r}^1(0,t) = [0\ 0\ 0]^T, \tag{2a}$$
$$\mathbf{R}^1(0,t) = \mathbf{R}_z(\theta^1(t) - \phi^1(t)u_3^1(0,t)), \tag{2b}$$
$$\alpha^i(0,t) = \theta^i(t) - \phi^i(t)u_3^i(0,t), \tag{2c}$$
$$u_3^i(\ell^i + \phi^i(t), t) = U_3^i, \tag{2d}$$

The boundary conditions given in (2) define $\boldsymbol{r}$, $\mathbf{R}$, and $\alpha$ at the base of the robot, and curvatures along $z$ direction, $\boldsymbol{u}_3^i$, at the end of the tubes, thus forming a boundary value problem. The mixture of boundary values imposes nonholonomic constraints on the motion of the robot. These constraints are discussed in the following section.

Here, we demonstrate that although the CTR model in (1) is quasi-static, it has a time-dependent behaviour. First, it is assumed that at a given time $t = t_k$, time-dependent variables are constant and the equations are solved in spatial domain (with respect to $s$). Next, the time-dependent variables are updated (*i.e.* $\theta^i(t)$, $\phi^i(t)$), and new transition points for separating the tubes are defined for the next sampling time, $t = t_{k+1}$. Finally, the equations are solved again in the spatial domain with the updated time-dependent variables. This process is repeated to calculate the robot's motion in the time domain.

To solve (1) in the spatial domain, a boundary value problem must be solved as the curvatures of the tubes at the robot base along the $z$ direction are unknown. Commonly, a shooting method is used to estimate the tubes base curvature $u_3^i(0,t_k)$ at a given time $t_k$ as a function of the known boundary conditions at the robot tip $u_3^i(\ell^i + \phi^i(t), t_k)$, thus satisfying the tip boundary conditions. Based on (2d), satisfying the boundary conditions requires that the twist curvature of tubes tip is constant at all times, *i.e.*

$$\frac{\partial u_3^i(\ell^i + \phi^i(t), t)}{\partial t} = 0. \tag{3}$$

Here, we show that solving the BVP, which leads to (3), imposes nonholonomic constraints on the motion of the robot. The time variation of $\mathbf{R}^1(s,t)$ during robot motion has the form of

$$\dot{\mathbf{R}}^1(s,t) = \mathbf{R}^1(s,t)\hat{\boldsymbol{\omega}}^1(s,t), \tag{4}$$

where $\hat{\boldsymbol{\omega}}^1(s,t)$ is the first tube's angular velocity. Equating the mixed partial derivatives of $\mathbf{R}^1(s,t)$ by taking the derivative of (1b) with respect to $t$ and derivative of (4) with respect to $s$ gives

$$\begin{gathered}\mathbf{R}^{1'}(s,t)\hat{\boldsymbol{\omega}}^1(s,t) + \mathbf{R}^1(s,t)\hat{\boldsymbol{\omega}}^{1'}(s,t) = \\ \dot{\mathbf{R}}^1(s,t)\hat{\boldsymbol{u}}^1(s,t) + \mathbf{R}^1(s,t)\dot{\hat{\boldsymbol{u}}}^1(s,t)\end{gathered} \tag{5}$$

[1]http://www.github.com/RViMLab/TRO2020-CTR-Model-Predictive-Control

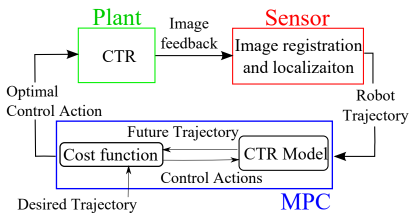


Fig. 3. Block diagram of MPC control loop.

Substituting from (1b) and (4), pre-multiplying by $\mathbf{R}^{1^T}(s,t)$, applying the identity $[\widehat{a \times b}] = \hat{a}\,\hat{b} - \hat{b}\,\hat{a}$, and taking the $\{.\}^\vee$ operator on the entire equation yields

$$\boldsymbol{u}^1(s,t) \times \boldsymbol{\omega}^1(s,t) + \boldsymbol{\omega}^{1'}(s,t) = \dot{\boldsymbol{u}}^1(s,t). \tag{6}$$

Equating the time derivative of the 1st tube end curvature using the third vector component of (6) and (3) gives

$$\begin{aligned} &u_2^1(\ell^1+\phi^1,t)\;\omega_1^1(\ell^1+\phi^1,t) - \\ &u_1^1(\ell^1+\phi^1,t)\;\omega_2^1(\ell^1+\phi^1,t) + \omega_3^{1'}(\ell^1+\phi^1,t) = 0 \end{aligned} \tag{7}$$

Equation (7) denotes the kinematic constraints on the motion of the robot end-effector due to robot tip boundary condition. It is nonholonomic and does not reduce the Degrees of Freedom of the robot. However, it reduces the robot's accessibility, because the motion of the robot is confined to a particular surface defined by (7). The nonholonomic constraint implies dependence of CTR configuration to its path history, as shown experimentally in [14]. Therefore, controllers that take decisions based on contemporary information may lead the CTR to configurations that it cannot recover from. This issue is tackled by the proposed controller.

## III. Control System Design

We develop an MPC that can safely and precisely steer a CTR along predefined trajectories with low computational overheads and respect of workspace boundaries, joint constraints, and path-dependency. Model predictive control theory [17] has never so far been investigated in the context of CTR. The controller's goal is to steer the robot tip along a desired trajectory. Fig. 3 depicts a block diagram of the developed closed-loop control algorithm, comprising:

1) A kinematic model of a CTR that predicts the robot trajectory given the robot tubes' rotation and translation as inputs.
2) A stereo camera system that estimates the robot's tip position.
3) The nonlinear model predictive controller (MPC) that manipulates the robot inputs to steer it along a desired trajectory.

The MPC controller calculates control decisions that iteratively minimise a cost function representing an error in desired versus current robot state parameters. We implement the robot's input saturations and boundary conditions of the kinematic models as equality, and inequality, constraints on the cost function, respectively. We now give a detailed mathematical description of the MPC scheme.

### A. Model Predictive Control

The controller will be designed to track a time-varying trajectory, $\boldsymbol{x}_d(t)$. This trajectory is essentially a trajectory in state-space, and can therefore include any parameter that can be considered robot state (*e.g.* stiffness, position, orientation, energy). For practical reasons we refer to $\boldsymbol{x}_d(t)$ as a trajectory in 3D Cartesian space. Without loss of generality, the CTR end-effector is assumed to be the tip of the most inner tube, and is denoted by $\boldsymbol{x}(t) = \int_0^{\ell^1+\phi^1(t)} \boldsymbol{r}^{1'}(s,t)\mathrm{d}s$. First, let's summarize the solution of the CTR model given in (1) as

$$\boldsymbol{x} = f(t, \boldsymbol{y}(s,t)), \tag{8a}$$

$$\boldsymbol{y} = g(s,t,\boldsymbol{y}(s,t),\boldsymbol{q}(t)), \tag{8b}$$

where $\boldsymbol{y} = [\boldsymbol{u}^1(s,t)...\boldsymbol{u}^N(s,t)]$, and the actuator value vector $\boldsymbol{q}(t)$ consists of the rotations and translations of each tube, $\theta^i(t)$ and $\phi^i(t)$, as shown in Fig. 2. We note that despite the quasi-static nature of CTR model, $t$ is included in (8) to reflect the path dependency of the CTR trajectory due to nonholonomic constraints.

To cast the kinematics problem into an IVP problem, the framework initially assumes that the curvatures of the tubes at the entry point $\boldsymbol{y}(0,t)$ are known. Then, (8) is the solution of (1) as an IVP. In later steps, these values are updated through an optimization process to respect BVP constraints. Also, it is assumed that there exists a control value $\boldsymbol{q}^*(t)$ that satisfies $\boldsymbol{x}^* = f(t,\boldsymbol{y}^*)$ and $\boldsymbol{y}^* = g(s,t,\boldsymbol{y}^*,\boldsymbol{q}^*)$. Here, $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ are the solution of (8) as an IVP with known $\boldsymbol{y}(0,t)$ as initial boundary conditions.

The proposed controller uses the CTR model to predict and optimize the future behaviour of the system. The idea of the MPC is that given the last measurement of CTR end-effector position available at each sampled time instant, we optimize the predicted behaviour of the system based on the CTR model, over a finite time horizon $\Gamma = t_0, ..., t_k, ..., t_{K-1}$ with $K \geqslant 2$. Only the first element of the predicted optimal input sequence is used as a feedback control value for the next sampling interval. Subsequently, the horizon is shifted one step forward and a new optimization problem is formulated and solved. This way the controller utilises information about the future trajectory to select the proper control inputs at the present. This is a key feature of MPC that helps the controller avoid singularities in robot workspace and robot configurations that it cannot recover from, while satisfying constraints on robot motion.

The MPC calculates the control decisions based on iterative optimization of the quadratic cost function

$$\mathcal{L}(\boldsymbol{x}_p,\boldsymbol{q}) = (\boldsymbol{x}_p(t) - \boldsymbol{x}_d(t))\,\mathbf{W}\,(\boldsymbol{x}_p(t) - \boldsymbol{x}_d(t))^T, \tag{9}$$

where $\mathbf{W}$ is a weighting matrix, $\boldsymbol{x}_d(t)$ the desired trajectory and $\boldsymbol{x}_p(t)$ is the predicted future trajectory given an initial value of control input $\boldsymbol{q}_0$ and a control sequence $\boldsymbol{q}(t)$. The predicted trajectory $\boldsymbol{x}_p(t)$ is calculated iteratively based on (8) as

$$\boldsymbol{y}(t_{k+1}) = g(\boldsymbol{y}(t_k), \boldsymbol{q}(t_k)), \tag{10}$$

$$\boldsymbol{x}_p(t_{k+1}) = f(\boldsymbol{y}(t_{k+1})). \tag{11}$$

Given the cost function in (9) and a prediction horizon length $K$, the nonlinear MPC scheme is formulated as Algorithm 1. The appropriate prediction horizon for the MPC is estimated with experimentation to balance predictive capabilities (extended horizons preferred) with computational cost (reduced horizons preferred).

**Algorithm 1:** Nonlinear MPC algorithm at each sampling time $t_k$

```
1 Measure the state of the system $x(t_k)$.
2 Set $x(t_0) = x(t_k)$.
3 Obtain the optimal control sequence $\boldsymbol{q}^*$ by solving:
    Minimize: $\mathcal{J}(x(t_0), q) := \int_0^{t_{K-1}} \mathcal{L}(\boldsymbol{x}_p, \boldsymbol{q})dt$
    with respect to: $\boldsymbol{q}$
    subject to: $\begin{cases} \boldsymbol{y}(t_{k+1}) = g(\boldsymbol{y}(t_k), \boldsymbol{q}(t_k)), \\ \boldsymbol{x}_p(t_{k+1}) = f(\boldsymbol{y}(t_{k+1})). \end{cases}$
4 Define the new feedback value $q(x(t_k)) := \boldsymbol{q}^*$ and use this
    control value in the next sampling period $t_{k+1}$.
```

We now present a numerical discretization approach to: (1) account for the effects of distal boundary conditions, (2) include inequality constraints on control inputs, and (3) solve the nonlinear optimal control problem within the MPC algorithm.

## *B. Recursive Discretization of the MPC*

We have so far neglected the boundary conditions at the end of the tubes in (2d) in the kinematic model used in the MPC. Here, we propose a discretization approach that allows implementation of the boundary conditions. It also transforms the MPC problem (Algorithm 1) into a standard nonlinear optimization problem, which can be solved via well-known robust nonlinear optimization algorithms. A standard nonlinear optimization problem has the form of

$$\begin{aligned}&\text{minimize: } \boldsymbol{F}(\boldsymbol{z})\\&\text{with respect to: } \boldsymbol{z}\\&\text{subject to: } \mathcal{E}(\boldsymbol{z})=0,\ \mathcal{I}(\boldsymbol{z})>0\end{aligned} \tag{12}$$

where $\mathcal{E}$ and $\mathcal{I}$ denote the equality and inequality constraints.

A common approach for transforming the MPC problem to (12) is the full discretization and incorporation of the robot model as constraints [18]. This method is straightforward but computationally inefficient as the optimization variables will include both the system states (*e.g.* curvatures along the length of the robot) and control inputs. Another approach is to decouple the control system from the optimization problem by recursively estimating the optimal solution $\boldsymbol{x}_p(t_{k+1})$ in (11) using the model of (8) outside of the optimization problem. This way, only the control inputs are optimization variables. However, because we are solving the IVP problem and neglecting the boundary conditions at the end of the tubes given in (2c), this method would result in significant error. To overcome these problems, we implement a method known as shooting discretization [18].

The idea of shooting discretization, not to be confused with the shooting methodology employed to solve the BVP in, *e.g.* [1], is to include in the problem selected components of the CTR states (*i.e.* initial curvature of the tubes), and kinematics (*i.e.* end curvature of the tubes), as independent optimization variables, and constraints, respectively. Proceeding this way, we provide useful information from the model to the optimizer throughout the optimization, and include the boundary conditions at the robot tip. The selected independent optimization variables are the curvatures of the tubes along $z$ direction at the entry point, *i.e.* $\boldsymbol{\psi}(0,t)=[\boldsymbol{u}_3^1(0,t)...\boldsymbol{u}_3^N(0,t)]$. Thus, the new set of optimization variables $z$ are

$$\boldsymbol{z}:=\{\boldsymbol{q}(t_0),\boldsymbol{\psi}(0,t_0),...,\boldsymbol{q}(t_{K-1}),\boldsymbol{\psi}(0,t_{K-1})\}, \tag{13}$$

where $\boldsymbol{\psi}(0,t)$ are the unknown initial values of curvature. Afterwards, we impose a set of constraints on the robot curvature (end point/boundary) to ensure the optimal solution satisfies the boundary conditions. The constraints on the control inputs and the kinematics of the model used in the optimization are

$$\mathcal{E}(\boldsymbol{z}):=\begin{cases} u_3^i(\ell^i+\phi^i,t_k)-U_3^i \quad (i=1...N, k=1...K)\\ \boldsymbol{x}_p(t_0)-\boldsymbol{x}(t_0)\end{cases} \tag{14a}$$

$$\mathcal{I}(\boldsymbol{z}):=\begin{cases} -\phi^N-\epsilon\\ -\phi^i+\phi^{i+1}-\epsilon \quad (i=1...N-1)\\ \phi^N-\phi_{max}\end{cases} \tag{14b}$$

The equality constraints, $\mathcal{E}(\boldsymbol{z})$, are the desired boundary conditions at the distal end of the CTR's tubes. By satisfying these constraints throughout the optimization, the optimal solution of the IVP converges to the BVP. The term in the second row of $\mathcal{E}(\boldsymbol{z})$ ensures the optimal solution begins from the current position of the robot. Without this term, trajectory continuity through time would not be respected. The inequality constraints, $\mathcal{I}(\boldsymbol{z})$, are the limits on the translational motion of the CTR's tubes. The desired linear offset between the tubes' base is $\epsilon$, representing transmission/collar couplings, while $\phi_{max}$ is the maximum allowed distance of the innermost tube from the entry point, $s=0$. Implementing $\mathcal{I}(\boldsymbol{z})$ in the optimization ensures that the solution satisfies the mechanical limits of the robot and that the base of the tubes will not collide with each other or pass the entry point. It has also been shown that keeping the tubes' bases far from collision results in higher manipulability [19] and reduces the possibility of instability [6].

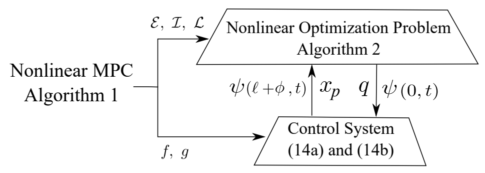


Fig. 4. Data communication between the elements of the control hierarchy.

We can now define the new nonlinear MPC algorithm as standard nonlinear optimization (12) in Algorithm 2.

**Algorithm 2:** Discrete nonlinear MPC algorithm at each sampling time $t_k$

```
1 Measure the state of the system $x(t_k)$.
2 Set $x(t_0)=x(t_k)$.
3 Obtain the optimal control sequence $\boldsymbol{q}^*$ by solving:
    Minimize: $\boldsymbol{F}(z):=\sum_{i=0}^{K-1}\mathcal{L}_i(t_i,\boldsymbol{x}_p,\boldsymbol{q})$
    w.r.t.: $\boldsymbol{z}:=\{\boldsymbol{q}(t_0),\boldsymbol{\psi}(0,t_0),...,\boldsymbol{q}(t_{K-1}),\boldsymbol{\psi}(0,t_{K-1})\}$
    subject to: $\begin{cases}\mathcal{E}(z)=0,\\ \mathcal{I}(z)>0\end{cases}$
4 Define the new feedback value $q(x(t_k)):=\boldsymbol{q}^*$ and use this
    control value in the next sampling period $t_{k+1}$.
```

Fig. 4 shows the hierarchy within the control system based on Algorithm 2. A numerical solver tackles the underlying differential equation of the CTR model of (1) and (8) as IVP for given control sequences, $\boldsymbol{q}$. These sequences correspond to values required by the nonlinear optimization solver in Algorithm 2. The interaction between these two components consists of sending control sequences, $\boldsymbol{q}$, and initial boundary conditions, $\boldsymbol{\psi}(0,t)$, from the nonlinear optimal problem solver to the solver of the CTR model, which in turn sends computed state sequences, $\boldsymbol{x}_p$, boundary values at the distal end of robot, $\boldsymbol{\psi}(\ell+\phi,t)=[\boldsymbol{u}_3^1(\ell^1+\phi^1,t)...\boldsymbol{u}_3^N(\ell^N+\phi^N,t)]$, back to the nonlinear optimal problem solver. Next, a numerical solution to the nonlinear optimal problem listed within Algorithm 2 is presented.

## *C. Solution of the MPC*

An interior-point method [20], efficient at handling equality and inequality constraints via appropriate implementation of *slack variables*, solves the nonlinear optimal control problem listed within Algorithm 2. In the interior-point method, a sequence of optimization variables, *i.e.* $\boldsymbol{z}$, that always belong to the feasible set of optimization variables that satisfy the constraints, is generated. For generating this sequence, in each iteration the full set of inequality constraints is used via slack variables $\gamma$ that transfer inequality constraints to active equality constraints with a non-negativity constraint on the slack variable. To this end, the optimization problem is reformulated:

$$\begin{aligned}&\text{minimize: } F(\boldsymbol{z})\\&\text{with respect to: } \boldsymbol{z},\ \boldsymbol{\gamma}\\&\text{subject to: } \mathcal{E}(\boldsymbol{z})=0,\ \mathcal{I}(\boldsymbol{z})-\boldsymbol{\gamma}=0,\ \text{and } \boldsymbol{\gamma}\geqslant 0.\end{aligned} \tag{15}$$

At this point, to include the effects of constraints, we replace the cost function in Algorithm 2 with the Lagrangian

$$L=F(z)-\boldsymbol{\lambda}_1^T\mathcal{E}(z)-\boldsymbol{\lambda}_2^T(\mathcal{I}(z)-\boldsymbol{\gamma}), \tag{16}$$

where $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$ are Lagrange multipliers.

Now, a first-order necessary optimality condition commonly known as KKT (Karush-Kuhn-Tucker) can be applied to the Lagrangian to solve the optimization problem. Let's denote by $\Delta$ the gradient with respect to $\boldsymbol{z}$. Then, assuming that $\Delta_z(\mathcal{E})$ and $\Delta_z(\mathcal{I})$ are linearly independent, KKT provides a set of necessary conditions for a solution in nonlinear programming to be optimal. The minimisation problem listed within Algorithm 2 indeed exhibits this property because the constraints given in (14a) and (14b) are linearly independent. The KKT conditions for (16) can be written as $\boldsymbol{\kappa} = 0$, where $\boldsymbol{\kappa}$ is

$$\boldsymbol{\kappa} = \begin{bmatrix} \Delta_z F(\boldsymbol{z}) - \Delta_z \mathcal{E}(\boldsymbol{z})^T \boldsymbol{\lambda}_1 - \Delta_z(\mathcal{I}(\boldsymbol{z}))^T \boldsymbol{\lambda}_2 \\ \boldsymbol{\lambda}_2 - \mu \ \mathrm{diag}(\boldsymbol{\gamma})^{-1} e \\ \mathcal{E}(\boldsymbol{z}) \\ \mathcal{I}(\boldsymbol{z}) - \boldsymbol{\gamma} \end{bmatrix} \tag{17}$$

with $\boldsymbol{\gamma} > 0$, $\boldsymbol{\lambda}_2 > 0$, $e = [1, 1, \ldots, 1]^T$, and $\mu$ is an initially positive integer that acts as a perturbation parameter added to the equations. The interior point method consists of solving the perturbed KKT conditions ($\boldsymbol{\kappa} = 0$) for a sequence of positive $\mu$ that converges to zero [20]. The strict positivity of $\mu$ in each iteration forces the slack variable $\gamma$ and the multiplier $\lambda_2$ to be positive, and therefore the solution to stay away from the boundary of the feasible set of solutions defined by the constraints. When $\mu \to 0$, the optimal solution of (16) will be independent of slack variables while satisfying the inequality constraints. Newton's method to solve (17) gives

$$\begin{bmatrix} \Delta^2_{zz} L + \sigma_1 \mathbf{I} & \mathbf{0} & \Delta_z \mathcal{E}^T & \Delta_z \mathcal{I}^T \\ \mathbf{0} & \Sigma & \mathbf{0} & -\mathbf{I} \\ \Delta_z \mathcal{E} & \mathbf{0} & -\sigma_2 \mathbf{I} & \mathbf{0} \\ \Delta_z \mathcal{I} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix} \overbrace{\begin{bmatrix} \delta z \\ \delta\gamma \\ -\delta\lambda_1 \\ -\delta\lambda_2 \end{bmatrix}}^{\delta} = \boldsymbol{\kappa} \tag{18}$$

where $\Sigma = \mathrm{diag}(\gamma)^{-1} \mathrm{diag}(\boldsymbol{\lambda}_2)$, and $\mathbf{I}$, and $\mathbf{0}$, are the identity, and zero matrices of appropriate dimensions, respectively. Variables $\sigma_1 > 0$ and $\sigma_2 > 0$ are damping factors introduced to ensure the positive definiteness of the Hessian matrix $\Delta^2_{zz} L$, and to avoid rank deficiency of $\Delta_z \mathcal{E}$, respectively. By solving (18), the step sizes for estimation optimization variables $\delta$ can be obtained. Finally, the MPC problem is solved following Algorithm 3.

**Algorithm 3:** Solver of nonlinear optimal problem

```
1  Suppose $(\boldsymbol{z},\ \boldsymbol{\gamma}) := (\boldsymbol{z}_0,\ \boldsymbol{\gamma}_0)$ (arbitrary) and set $m := 0$.
2  Compute multipliers $\lambda_{1_0}$ and $\lambda_{2_0}$ from (17) and define
     parameters $\mu_0 > 0$ and $\varepsilon \in (0, 1)$.
3  do
4      do
5          Compute search direction $\delta$ via (18),
6          Find $\Theta := \max\{\Theta \in (0,1] \mid \gamma + \Theta\delta\gamma > 0\}$,
7          Set $\boldsymbol{z}_{m+1} := \boldsymbol{z}_m + \Theta\delta\boldsymbol{z}$,
           Set $\boldsymbol{\gamma}_{m+1} := \boldsymbol{\gamma}_m + \Theta\delta\boldsymbol{\gamma}$,
           Set $\boldsymbol{\lambda}_{1_{m+1}} := \boldsymbol{\lambda}_{1_m} + \delta\boldsymbol{\lambda}_1$,
           Set $\boldsymbol{\lambda}_{2_{m+1}} := \boldsymbol{\lambda}_{2_m} + \delta\boldsymbol{\lambda}_2$,
8          Set $\mu_{m+1} := \mu_m$,
           Set $m := m + 1$.
           Compute $\mathcal{C} := ||\boldsymbol{\kappa}||_\infty$.
9      while $\mathcal{C} > \mu_m$;
10     Choose $\mu_m \in (0, \varepsilon\mu_m)$.
11 while $\mathcal{C} > \epsilon_{Tol}$;
```

In Algorithm 3, $\Theta$ is estimated to ensure slack variables neither become zero nor reach their lower bounds too fast. The error function $\mathcal{C}$ is the infinity norm of $\boldsymbol{\kappa}$, and is implemented as termination criterion and measure of convergence and fulfillment of KKT optimality condition. Algorithm 3 can be used to solve the MPC problem at each sampling time. In the next section, simulations are performed to evaluate the proposed MPC controller.

## IV. MPC Evaluation via Simulations

Simulations compare MPC performance with Jacobian-based controllers, which to the best of authors knowledge, are the most common type of controller for CTRs [8], [9], [21]. The Jacobian can be used with the following closed-loop control law to steer the CTR [22]:

$$\dot{\boldsymbol{q}}_d = \mathbf{J}^\dagger[\dot{\boldsymbol{x}}_d + \mathbf{K}_p(\boldsymbol{x}_d - \boldsymbol{x})], \tag{19}$$

where $\mathbf{J}^\dagger$ is the pseudo-inverse of the robot Jacobian, $\mathbf{K}_p$ is a symmetric positive definite matrix, $\boldsymbol{q}_d$ is the control input, and $\boldsymbol{x}_d$ is the desired robot trajectory in its task space. An effective strategy that allows Jacobian-based control of robots in the neighborhood of kinematic singularities is the damped least-squares technique. The method improves inverse kinematics when the Jacobian is ill-conditioned and close to a singularity by adding a damping factor [22]. In this case, the pseudo-inverse of the robot Jacobian in (19) can be replaced by $(\mathbf{J}^T\mathbf{J} + \Lambda^2\mathbf{I})^{-1}\mathbf{J}^T$, where $\Lambda$ is the damping factor. Here, we compare the performance of the MPC controller to one that uses the control law of (19) with and without damping.

The controllers are tested on a perturbed CTR model with $\pm 10\%$ uncertainty in the values of the tubes' Young and shear moduli with respect to the nominal ones. The selected robot is purposefully unstable within its workspace to test the performance of the controller in avoiding instabilities. The CTR design parameters are shown in Table I. For consistency, these parameters are identical to those used in our experimental setup in Sec. V. Simulations show that local controllers (*e.g.* Jacobian-based) blindly steer the robot to configurations it cannot recover from, while the proposed MPC incorporates future knowledge to follow optimal paths.

We performed a simulation study to select the appropriate prediction horizon. We incremented the horizon of the controller from 2 to 10 and used it to track several linear and circular trajectories. Increasing the prediction horizon to higher than 5 did not significantly improve the performance of the controller, while it precluded our desired sampling time of 150 ms on the robot workstation (non real-time performance). Therefore, we decided to use an MPC with prediction horizon of 5 that satisfies both the performance and real-time criteria. Results for prediction horizons of 2 and 5 are shown in Fig. 5(a) and Fig. 5(b). The controller parameters used in the simulations were $\mu = 0.1$, $\sigma_1 = \sigma_2 = 0.1$, and $\mathcal{C} = 10^{-7}$. $\mathbf{K}$ in (19), and $\Lambda$, were selected as $2\mathbf{I}$, and 0.45, respectively, as these values were found to achieve the minimum tracking error. The simulation results for tracking a linear trajectory are shown in Fig. 5(a). The mean error of the Jacobian-based controller, and MPC controllers with prediction horizons of 2, and 5, are 0.203, 0.123, and 0.080 mm, respectively. The damped least-squares method had the same results as the Jacobian-based controller. All controllers satisfied the robot boundary conditions at its tip with a minimum accuracy of $5 \times 10^{-4}\mathrm{m}^{-1}$. Overall, the MPC with expanded horizon reduced the trajectory-following error by 80%.

In the second simulation study, a circular trajectory near the limits of the robot workspace was selected. Results of the robot tip trajectory in task space are shown in Fig. 5(b) and Fig. 5(c), while the evolution of the tubes' base position and base rotation values is shown in Fig. 6. Both Jacobian-based controllers initially follow the desired trajectory at the cost of failing the joints' constraints as seen in Fig. 6. In Fig. 6, snapping manifests when tube 3 is fully retracted and tube 2 is inserted up to its limit. At the same time, the difference between the rotation angle of tube 1 and tube 2 is close to $180^\circ$, which leads to an

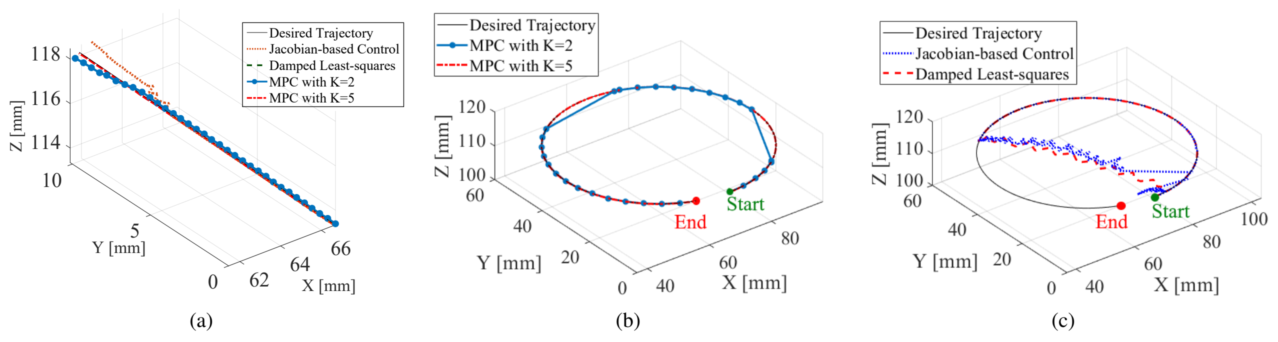


Fig. 5. Comparison between Jacobian-based controllers, MPC controller with horizon of 2, and MPC controller with horizon of 5. (a) Linear trajectory tracking, (b) MPC controller for circular trajectory tracking, and (c) Jacobian-based controllers for circular trajectory tracking.

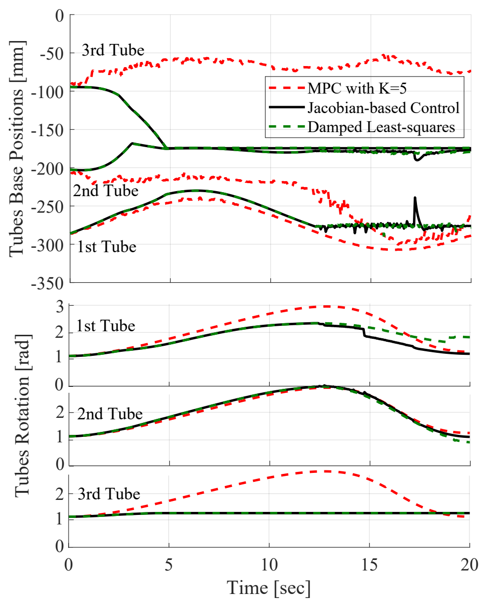


Fig. 6. Comparison between position of the base of robot's tubes and rotation of the tubes for Jacobian-based controllers and MPC controller.

TABLE I
PHYSICAL PARAMETERS FOR CTR'S TUBES.

| | Inner Radius [mm] | Outer Radius [mm] | Straight Length []mm] | Curved Length [mm] | $U$ [m$^{-1}$] | $E$ [GPa] | $G$ [GPa] |
|---|---|---|---|---|---|---|---|
| 1st Tube | 0.35 | 0.55 | 431 | 103 | 21.3 | 10.25 | 18.79 |
| 2nd Tube | 0.7 | 0.9 | 330 | 113 | 13.1 | 68.6 | 11.53 |
| 3rd Tube | 1 | 1.2 | 174 | 134 | 3.5 | 16.96 | 14.25 |

almost straight robot (see Fig. 1), instability, and subsequent snapping to a position of less energy. The damped least-squares controller slightly outperforms the simple Jacobian-based controller (*e.g.* see T =17 sec). The MPC with the short prediction horizon of 2 is also unable to find the optimal solution near the robot singularities, and simply skips those areas. The MPC with longer prediction horizon, however, not only satisfies joint constraints (see Fig. 6) but also steers the CTR on the desired trajectory and avoids workspace singularities.

## V. Experimental Evaluation and Discussion

Fig. 7 shows the experimental setup. Commonly, electromagnetic trackers are used to track the robot tip. Here, we simulated a scenario in which the same measurement is acquired using a calibrated stereo rig comprising two Logitech HD Pro C922 webcams. The cameras were running at 1080p resolution and 30 frames per second. As identified through calibration using on average 30 views of a circular calibration pattern, a single pixel corresponded to $0.2 \times 0.2$ mm on the image plane. A spherical coloured marker was fixed at the tip of the CTR. The marker was detected by color thresholding, and its triangulated 3D position was used as the MPC input.

Manual backbone segmentation established the base and shape of the CTR relative to the aligned calibration grid. Matching backbone points were selected in both images, and then triangulated to provide the 3D point cloud. The extracted 3D backbones were used to calibrate for the CTR model parameters, namely, Young's and shear moduli of the tubes. The parameters were identified by fitting the kinematic model given in (1) to the shape of the robot estimated via the cameras at 25 different configurations. The identified parameters of the model are given in Table I, and are the same as those used in the simulations. Tube diameters were selected to ensure smooth telescopic translation of the tubes with minimal friction. The maximum error of the model in predicting the CTR tip position was 1.74 cm, corresponding to 9% of robot's length, which is inline with [2], where the model was proposed.

The identified model was implemented in the MPC to steer the

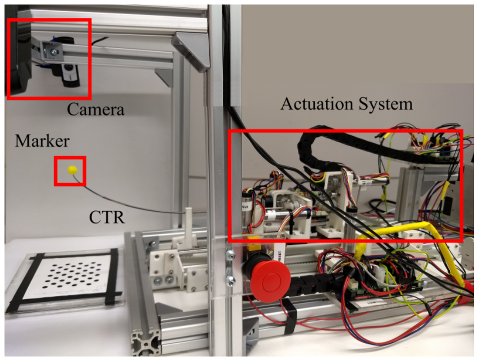
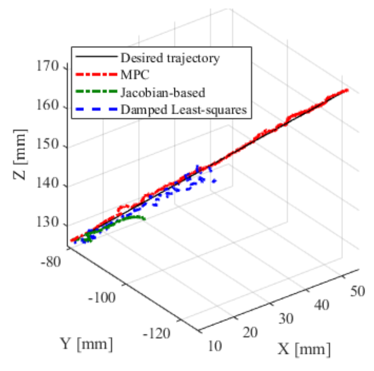


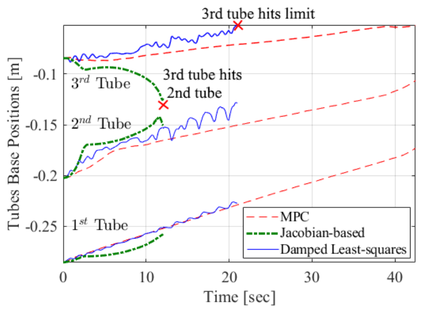
Fig. 7. The CTR comprises 3 precurved Nitinol tubes. The robot uses 6 motors (Maxon, CH), connected to pulleys and timing belts for rotating and translating the tubes. Motor position controllers (EPOS4 Compact 50/5 CAN, Maxon, CH) are connected to a PC using a CAN-to-USB interface (Kvaser Inc., CA, USA). A stereo rig tracks the robot tip in 3D.

TABLE II
RESULTS OF 3D POINT TRACKING.

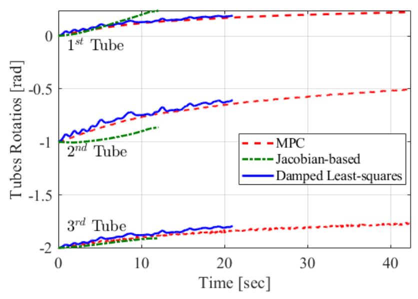
| $e_{mean}$ [mm] | $PO$ | $t_r$ [sec] | $t_s$ [sec] | $\sigma$ [mm] |
|---|---|---|---|---|
| 0.5 | 7% | 0.64 | 0.85 | 0.70 |



(a)

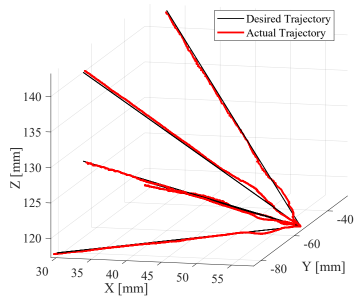


(b)

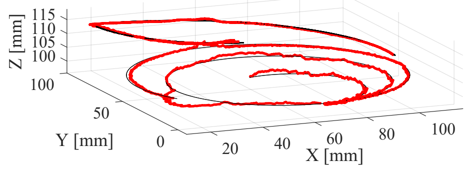


(c)

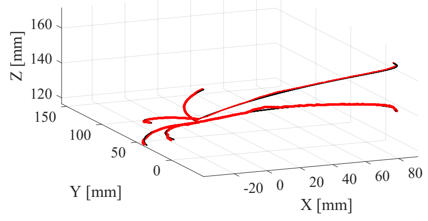
Fig. 8. Experimental comparison of the MPC against a controller with damped least-squares inverse kinematics. (a) Desired trajectory and actual trajectory. (b) The position base of robot's tubes, violation of joints' limits are shown with a red cross. (c) The angular position of base of robot's tubes.



(a)



(b)



(c)

Fig. 9. Representative experimental results for trajectory tracking. (a) 2D and 3D linear trajectory tracking. (b), (c) 2D and 3D circular trajectory tracking.

TABLE III
RESULTS OF TRAJECTORY TRACKING EXPERIMENTS.

| | Linear trajectory | Circular trajectory |
|---|---|---|
| $e_{x\ \max}$[mm] | 2.9 | 14 |
| RMSE$_x$[mm] | 1.2 | 1.7 |
| $e_{u\ \max}$ [m$^{-1}$] | $1.6\times10^{-3}$ | $1.4\times10^{-3}$ |
| RMSE$_u$ $10^{-4}$ | $0.8\times10^{-3}$ | $2.2\times10^{-3}$ |

TABLE IV
COMPARISON OF JACOBIAN-BASED AND MPC CONTROLLERS.

| | MPC | Jacobian-based | Damped least-squares |
|---|---|---|---|
| RMSE[mm] | 0.8 | 5.3 | 4.1 |
| $\sigma$ [mm] | 0.5 | 3.1 | 2.4 |

CTR. Other parameters of the controller were practically identified as $\mu = 0.1$, $\sigma_1 = \sigma_2 = 0.1$, and $\mathcal{C} = 10^{-7}$. An Intel Core i7 (2.93 GHz) machine was used to solve the optimal control problem at sampling time of 150 ms. The prediction horizon of the MPC, $K$, was equal to 5, as per our simulations. To minimise the effect of delays caused by image processing, two computers connected via UDP communication were used to control the robot. One tracked the robot tip using ROS (Linux OS), and one run the control algorithm using Matlab Simulink Real-time (Windows OS). The measured round-trip time in the control-loop was 300 ms (*i.e.* twice the sampling time).

We evaluated the performance of the controller in reaching 3D points, and following linear and circular trajectories. These are elemental trajectories that can make up any complex trajectory in 3D space, and provide quantitative insights on controller performance.

In the first experiment, we evaluated the performance of the MPC in tracking single points across the robot workspace (step responses). We performed 20 (random) point tracking experiments, each repeated 4 times, in which the robot is tasked to move the chosen point in 3D space. Mean error $e_{mean}$, percentage overshoot $PO$, response time $t_r$, settling time $t_s$, and standard deviation $\sigma$ for the experiments are reported in Table. II. The mean error of the MPC as a percentage of robot length surpasses the state-of-the-art (0.03% vs 0.05% in [10] and 0.09% in [9]; note that our three-tube robot is unstable, while the compared controllers were evaluated on a two-tube stable robot.

Next, we compared the performance of the controller with implemented controllers that use the inverse-Jacobian and damped least-squares inverse kinematics. The CTR was steered on a 3D trajectory near its joint limits; 10 trials were performed. A comparison between the MPC and damped least-squares inverse kinematics is shown in Fig. 8. It can be seen that the damped least-squares controller is unable to track the desired trajectory, while the joint constraints defined in (14a) are violated (the 3rd tube reaches the entry point). However, the proposed MPC controller respects the joint limits until it reaches the robot's workspace boundaries. The limit is shown by a red cross in Fig. 8(b). The mean error and standard deviation of the controllers are compared in Table IV.

In the next set of experiments, we used the MPC to steer the CTR on 2D and 3D linear trajectories, and 2D and 3D circular trajectories. The trajectories were randomly selected across the robot workspace with the constraint to be near instabilities and workspace boundaries; 15 trials were performed for each scenario. Representative experimental results for 2D circular trajectories, 3D circular trajectories, and linear trajectories are shown in Fig. 9. The Jacobian-based controllers (both with and without damped least squares) failed to perform circular trajectory tracking. Results of all

$$\mathrm{RMSE_u} = \sqrt{\frac{\sum_{i=1}^{n}\left(\left\|\left[U_3^1,\ \ U_3^2,\ \ U_3^3\right]^T\right\| - \left\|\left[u_3^1(\ell^1+\phi^1,t_n),\ \ u_3^2(\ell^2+\phi^2,t_n),\ \ u_3^3(\ell^3+\phi^3,t_n)\right]^T\right\|\right)^2}{n}} \tag{20}$$

the experiments are summarized in Table III. Maximum of norm of error for trajectory tracking $e_{x\,\max}$, the root-mean-squared tracking error RMSE, maximum of norm of error for equality constraints in (14a) $e_{u\,\max}$, and root-mean-squared error of equality constraints RMSE$_u$ are listed. The root-mean-squared error is calculated as RMSE$_x$ $= \sqrt{\frac{\sum_{i=1}^{n}(\|\hat{x}\|_i - \|x\|_i)^2}{n}}$, and is used as a measure of the differences between the desired value, $\hat{x}$, and the actual value of the parameters experimentally observed, *i.e.* $x$, for $n$ data points. $e_{u\,\max}$ is calculated as $\max\{\Sigma_{i=1}^{3}\left\|u_3^i(\ell^i+\phi^i,t_k) - U_3^i\right\|\}$. RMSE of curvature is calculated as shown in (20) and is used as a measure of the differences between the desired value of end curvature that satisfies the constraints in (14a), and the actual value of the end curvature obtained by the MPC for $n$ data points.

The CTR is able to follow the trajectories with a maximum RMSE of $1.7\,\mathrm{mm}$,corresponding to $1\%$ of robot arclength. The maximum tip error occurs for one of the circular trajectories. Errors are due to the nonlinearity of CTR's kinematics compared to classic robots and implicit constraints on the motion of the robot as discussed in Sec. I and II. The MPC reduces the errors and leads the robot tip back to the desired trajectory despite locally exhibiting its maximum error. Moreover, the controller satisfies the equality constraints on robot curvature, *i.e.* boundary conditions, with maximum error of $2.2 \times 10^{-3}\,\mathrm{m}^{-1}$.

## VI. Concluding Remarks

In this paper, we introduced a tailored Model Predictive Controller (MPC) for autonomous steering of Concentric Tube Robots (CTR). The MPC approach was selected as it explicitly considers the constraints on the robot's workspace and boundary conditions. More importantly, by considering a time horizon, the MPC makes informed control decisions that steer the CTR away from configurations that it cannot recover from, *e.g.* joint limits, instabilities, singularities. The MPC calculates control actions based on iterative optimization of the predictions of the kinematic model of CTR. Simulations and experiments show that the proposed controller outperforms local Jacobian-based controllers by a large margin. Ongoing work focuses on improving computational efficiency.

## References

[1] P. E. Dupont, J. Lock, B. Itkowitz, and E. Butler, "Design and control of concentric-tube robots," *IEEE Trans. Robotics*, vol. 26, no. 2, pp. 209–225, 2010.

[2] D. C. Rucker, B. A. Jones, and R. J. Webster, "A geometrically exact model for externally loaded concentric-tube continuum robots," *IEEE Trans. Robotics*, vol. 26, no. 5, pp. 769–780, 2010.

[3] H. B. Gilbert, J. Neimat, and R. J. Webster, "Concentric tube robots as steerable needles: Achieving follow-the-leader deployment," *IEEE Trans. Robotics*, vol. 31, no. 2, pp. 246–258, 2015.

[4] A. H. Gosline, , N. V. Vasilyev, E. J. Butler *et al.*, "Percutaneous intracardiac beating-heart surgery using metal mems tissue approximation tools," *Int. J. Robotics Research*, vol. 31, no. 9, pp. 1081–1093, 2012.

[5] C. R. Mitchell, R. J. Hendrick, R. J. Webster, and S. D. Herrell, "Toward Improving Transurethral Prostate Surgery: Development and Initial Experiments with a Prototype Concentric Tube Robotic Platform," *J. Endourology*, vol. 30, no. 6, pp. 692–696, 2016.

[6] C. Bergeles, A. H. Gosline, N. V. Vasilyev, P. J. Codd, P. J. del Nido, and P. E. Dupont, "Concentric tube robot design and optimization based on task and anatomical constraints," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 67–84, 2015.

[7] K. Leibrandt, C. Bergeles, and G. Yang, "Concentric tube robots: Rapid, stable path-planning and guidance for surgical use," *IEEE Robotics & Automation Magazine*, vol. 24, no. 2, pp. 42–53, 2017.

[8] D. C. Rucker and R. J. Webster, "Computing jacobians and compliance matrices for externally loaded continuum robots," in *IEEE Int. Conf. Robotics and Automation*, 2011, pp. 945–950.

[9] R. Xu, A. Asadian, S. F. Atashzar, and R. V. Patel, "Real-time trajectory tracking for externally loaded concentric-tube robots," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 4374–4379.

[10] K. Wu, G. Zhu, L. Wu, W. Gao, S. Song, C. M. Lim, and H. Ren, "Safety-enhanced model-free visual servoing for continuum tubular robots through singularity avoidance in confined environments," *IEEE Access*, vol. 7, pp. 21 539–21 558, 2019.

[11] A. V. Kudryavtsev, M. T. Chikhaoui, A. Liadov, P. Rougeot, F. Spindler, K. Rabenorosoa, J. Burgner-Kahrs, B. Tamadazte, and N. Andreff, "Eye-in-hand visual servoing of concentric tube robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2315–2321, 2018.

[12] M. Khadem, J. O'Neill, Z. Mitros, L. d. Cruz, and C. Bergeles, "Autonomous steering of concentric tube robots for enhanced force/velocity manipulability," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 2197–2204.

[13] M. M. Marinho, B. V. Adorno, K. Harada, and M. Mitsuishi, "Dynamic active constraints for surgical robots using vector-field inequalities," *IEEE Trans. Robotics*, vol. 35, no. 5, pp. 1166–1185, 2019.

[14] J. Ha, G. Fagogenis, and P. Dupont, "Effect of path history on concentric tube robot model calibration," *Hamlyn Symp. Medical Robotics*, 2017.

[15] M. Khadem, J. O'Neill, Z. Mitros, L. d. Cruz, and C. Bergeles, "Autonomous steering of concentric tube robots for enhanced force/velocity manipulability," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2019, pp. 2197–2204.

[16] J. Ichnowski and R. Alterovitz, "Motion planning templates: A motion planning framework for robots with low-power cpus," in *IEEE Int. Conf. Robotics and Automation*, 2019, pp. 612–618.

[17] R. R. Negenborn and J. M. Maestre, "Distributed model predictive control: An overview and roadmap of future research opportunities," *IEEE Control Systems Magazine*, vol. 34, no. 4, pp. 87–97, 2014.

[18] H. Bock and K. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems*," *IFAC Proceedings Volumes*, vol. 17, no. 2, pp. 1603 – 1608, 1984.

[19] M. Khadem, L. Da Cruz, and C. Bergeles, "Force/velocity manipulability analysis for 3d continuum robots," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2018, pp. 4920–4926.

[20] J. Nocedal and S. Wright, *Interior-Point Methods for Nonlinear Programming*. New York, NY: Springer New York, 2006, pp. 563–597.

[21] J. Burgner, D. C. Rucker, H. B. Gilbert, P. J. Swaney, P. T. Russell, K. D. Weaver, and R. J. Webster, "A telerobotic system for transnasal surgery," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 3, pp. 996–1006, 2014.

[22] S. Chiaverini, B. Siciliano, and O. Egeland, "Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator," *IEEE Trans. Control Systems Technology*, vol. 2, no. 2, pp. 123–134, 1994.