Bayesian Neural Network Modeling and Hierarchical MPC for a Tendon-Driven Surgical Robot With Uncertainty Minimization

Francesco Cursi, *Graduate Student Member, IEEE*, Valerio Modugno, Leonardo Lanari, *Member, IEEE*, Giuseppe Oriolo, *Fellow, IEEE*, and Petar Kormushev

***Abstract*—In order to guarantee precision and safety in robotic surgery, accurate models of the robot and proper control strategies are needed. Bayesian Neural Networks (BNN) are capable of learning complex models and provide information about the uncertainties of the learned system. Model Predictive Control (MPC) is a reliable control strategy to ensure optimality and satisfaction of safety constraints. In this work we propose the use of BNN to build the highly nonlinear kinematic and dynamic models of a tendon-driven surgical robot, and exploit the information about the epistemic uncertainties by means of a Hierarchical MPC (Hi-MPC) control strategy. Simulation and real world experiments show that the method is capable of ensuring accurate tip positioning, while satisfying imposed safety bounds on the kinematics and dynamics of the robot.**

***Index Terms*—Medical robots and systems, model learning for control, robot safety, tendon/wire mechanism.**

I. Introduction

ACCURACY and precision are of uttermost importance to ensure safety in many robotic applications, especially in minimally invasive surgery, where little (or preferably no) damage should be caused to the patient's body. Moreover, effective control strategies are needed to guarantee the safe execution of surgical tasks, yet, two main challenges need to be faced: robot modeling and robot control.

Different types of mechanical transmissions have been used in the design of surgical robots, with the vast majority being tendon-driven [1]. Due to the model uncertainties and the highly complex nonlinearities in tendon-driven systems, researchers employed machine learning approaches [2] such as Artificial Neural Networks [3], Gaussian Mixture Regression, K-nearest Neighbour Regression, and Extreme Machine Learning [4] to learn the system's model.

With regards to robot control, Model Predictive Control (MPC) is shown to be an approximation to the optimal feedback control [5] and proved to be an effective strategy in dealing with nonlinear systems under constraints [6]. MPC can be effectively employed both for regulation and tracking tasks where the desired trajectory is known or at least can be defined within the prediction horizon (i.e. for reference paths that change during the task execution). Free-space tracking is a first essential step towards more complicated tasks such as not-tying, suturing, or tumor resection Even though the surgical environment is weakly structured, it is still possible to estimate its evolution in time, for instance, by tracking tissue deformation [7], and thus still have some reference for tracking.

In this work, we focus on extending our recent approach [6] to controlling the Micro-IGES tendon-driven surgical robot in multiple ways. Firstly, a more precise kinematic model for Cartesian trajectory tracking has been included. Learning a more accurate model for the system kinematics and dynamics increases the MPC performances [8] and for this purpose, we use Bayesian Neural Networks (BNN). BNN have been chosen because, similarly to feedforward neural networks, they allow building complex mappings but are less prone to overfitting and, most importantly, return information about the model uncertainty. Secondly, the information of the models' uncertainties is exploited to increase the reliability of the controller by means of prioritized optimal control [9] to solve two different tasks: accurately tracking a desired path and minimizing the models' epistemic errors. Therefore, the proposed control strategy consists of a Hierarchical Model Predictive Control (Hi-MPC), with a primary MPC for path tracking, and a secondary MPC for the minimization of the uncertainties in the model.

The paper is structured as follows. Section II introduces the BNN for system modeling, presents the Micro-IGES robotic surgical tool (Fig. 1), and shows how BNN are employed to build kinematic and dynamic models. Section III describes the Hi-MPC approach for trajectory tracking and uncertainty minimization, under the imposed kinematic and dynamic constraints. In Section IV results on a simulation environment and on the real robot are shown. Lastly, conclusions are drawn in Section V.

Manuscript received October 15, 2020; accepted February 12, 2021. Date of publication February 25, 2021; date of current version March 17, 2021. This letter was recommended for publication by Associate Editor C. Rucker and Editor P. Valdastri upon evaluation of the Reviewers' comments. This work was supported by the EPSRC Project EP/P012779/1.) *(Corresponding author: Francesco Cursi.)*

Francesco Cursi is with the Hamlyn Centre, Imperial College London, Exhibition Road, London, U.K., and also with Robot Intelligence Lab, Imperial College London, London, U.K. (e-mail: cursifrancesco@gmail.com).

Valerio Modugno, Leonardo Lanari, and Giuseppe Oriolo are with the Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, 00185 Roma, Italy (e-mail: modugno@diag.uniroma1.it; lanari@diag.uniroma1.it; oriolo@diag.uniroma1.it).

Petar Kormushev is with Robot Intelligence Lab, Imperial College London, Exhibition Road, London, U.K. (e-mail: p.kormushev@imperial.ac.uk).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3062339, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3062339

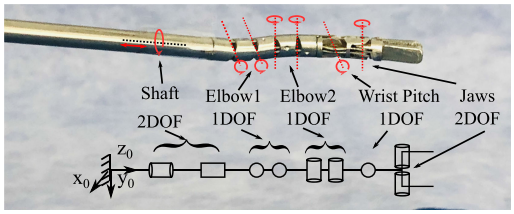

Fig. 1. Micro-IGES surgical robotic tool and its kinematic model.

II. Robotic System Modelling

In this section, an overview of the Micro-IGES surgical robotic tool, describing its kinematic and dynamic models, and Bayesian Neural Networks for modeling are introduced.

A. Micro-IGES Surgical Robotic Tool

The Micro-IGES [10] (Fig. 1) is a surgical robotic tool, composed of a rigid shaft and a flexible section. The shaft provides the roll and translation Degrees of Freedom (DOFs). The articulated end, instead, consists of 2 elbows for pitch and yaw, with each elbow made of a pair of coupled joints, a 1 DOF revolute joint for the wrist pitch, and the jaws. The jaws provide two more DOFs: one for the wrist yaw and one for the gripper's opening/closing. Each joint of the articulated part is driven by an antagonistic pair of tendons, with each pair being connected to the corresponding driving capstan at the proximal drive unit. The coupling of the two pairs of joints of the elbows occurs at the driving unit: the two capstans that drive the two serial joints for each DOF of the elbow (pitch and yaw) are coupled by a series of gears with a 1:2 ratio. Due to the current setup, in this work, the translation DOF cannot be used, therefore only 5 independent DOFs are considered (Roll, Elbow 1, Elbow 2, Wrist Pitch, Wrist Yaw). In order to control the Wrist Yaw DOF, with a null gripping angle, the two jaws need to move equally (their motion is not independent). Due to the lack of distal sensors, the joints cannot be directly controlled, whereas the motors can be. Since the robot is position controlled, the vector of controllable motor positions is here defined by $\boldsymbol{\theta} = [\theta_R \quad \theta_{e1} \quad \theta_{e2} \quad \theta_W \quad \theta_{j1}]$, where θ_{j1} is the motor command of one jaw.

B. Robot Kinematic and Dynamic Model

The nonlinearities in tendon transmission make the mathematical derivation of the system kinematics and dynamics tedious. Because of the generally nonlinear motor-to-joint (and joint-to-motor) algebraic mapping $\boldsymbol{q} = \boldsymbol{q}(\boldsymbol{\theta})$, being $\boldsymbol{\theta}$ the vector of motor positions and $\boldsymbol{q}$ the vector of joint positions, the kinematics and dynamics, in absence of external interactions, can be rewritten in terms of the controllable motor commands as:

$$\begin{aligned} \boldsymbol{P} &= \boldsymbol{P}_q(\boldsymbol{q}) = \boldsymbol{P}_q(\boldsymbol{q}(\boldsymbol{\theta})) = \boldsymbol{P}(\boldsymbol{\theta}) \\ \boldsymbol{\tau}_{mot} &= \boldsymbol{L}(\boldsymbol{q})^T \boldsymbol{\tau}_j(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \boldsymbol{\Gamma}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \ddot{\boldsymbol{\theta}}) \end{aligned} \tag{1}$$

with $\boldsymbol{P} \in \mathbb{R}^3$ being the Cartesian end-effector position, $\boldsymbol{P}_q(\boldsymbol{q})$ the mapping from joint-to-tip position, $\boldsymbol{P}(\boldsymbol{\theta})$ the consequent motor-to-tip mapping, $\boldsymbol{\tau}_{mot}$ the vector of motor torques and $\boldsymbol{\tau}_j$ the vector of joint torques, which is typically known for articulated robots, and the matrix $\boldsymbol{L}$, with $L_{i,j} = \frac{\partial q_i}{\partial \theta_j}$, the motor-to-joint differential matrix [6]. $\boldsymbol{\Gamma}$ takes into consideration all the dynamic effects, including the nonlinearities in the tendon transmission.

Given the geometry of the robot and the motor-to-joint mapping, an approximated kinematic model $\boldsymbol{P}_{app}$ can be built [11]. BNN are then used in this work to correct the current robot model, by learning the positioning error $\boldsymbol{P}_{err}$ between the actual tip position and the approximated model's one, and to build the mapping from motor positions to tip position $\boldsymbol{\theta} \rightarrow \tilde{\boldsymbol{P}}$, with $\tilde{\boldsymbol{P}} = \boldsymbol{P}_{app} + \boldsymbol{P}_{err}$.

With regards to the dynamic model, due to the small accelerations usually required in robotic surgery, in this work we assume that the motor torques only depend on the motor positions and velocities, meaning $\boldsymbol{\tau}_{mot} = \boldsymbol{\tau}_{mot}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$, and, therefore, with BNN we learn the mapping $[\boldsymbol{\theta} \quad \dot{\boldsymbol{\theta}}] \rightarrow \tilde{\boldsymbol{\Gamma}}$. In this work $\tilde{\boldsymbol{P}}$ and $\tilde{\boldsymbol{\Gamma}}$ are the BNN output estimates.

C. Bayesian Neural Networks for Modelling

Similarly to Feedforward Artificial Neural Networks (ANN), BNN are capable of representing complicated behaviours, without the need of knowing any mathematical or physical model. ANN, however, are highly influenced by outliers in the dataset [12], suffer from overfitting, and it is difficult to control their complexity [13]. BNN, instead, allow including uncertainties in the model, by adding priors to the weights of the network. BNN use probability distributions to quantify uncertainty in inferences and output a probability distribution expressing the belief on how likely the different predictions are [13].

Given a dataset of D observations $\mathcal{D} = \{\tilde{y}_1, \ldots, \tilde{y}_D\}$ and the corresponding input vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_D$, the resulting neural network model is defined as $\hat{y} = y(\boldsymbol{x}, \boldsymbol{w})$, where $\boldsymbol{w}$ is the vector of network's weights and biases. In the Bayesian learning, a Gaussian conditional distribution is assigned to each data point $p(\tilde{y}|\boldsymbol{x}, \boldsymbol{w}, \beta) = \mathcal{N}(\tilde{y}|y(\boldsymbol{x}, \boldsymbol{w}), \beta^{-1})$, with the output of the network as mean and inverse variance β. Assuming each point independently, the overall distribution on the dataset is computed as $p(\mathcal{D}|\boldsymbol{w}, \beta) = \prod_{d=1}^{D} \mathcal{N}(\tilde{y}_d|y(\boldsymbol{x}_d, \boldsymbol{w}), \beta^{-1})$. A prior distribution $p(\boldsymbol{w}|\alpha) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \alpha^{-1}\boldsymbol{I})$, with zero mean and inverse variance α, is also assigned to the network's weights. The posterior distribution on the weights is then:

$$p(\boldsymbol{w}|\mathcal{D}, \alpha, \beta) \propto p(\mathcal{D}|\boldsymbol{w}, \beta) p(\boldsymbol{w}|\alpha) \tag{2}$$

which is non Gaussian, due to the nonlinear network model $y(\boldsymbol{x}, \boldsymbol{w})$. However, a Gaussian approximation can be found by using different techniques, such as Laplace approximation or Variational Inference [14]. In this work, we adopted the latter one, which allows identifying the Gaussian approximation $q(\boldsymbol{w}|\mathcal{D}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}^*, \boldsymbol{A}^{-1})$, with the optimal weights $\boldsymbol{w}^*$ as mean and $\boldsymbol{A}$ as inverse covariance matrix, utilizing Kullback-Leibler divergence. This is performed offline by employing the Dense Flipout estimator [15] for each network layer.

Once the Gaussian approximation for the weights' posterior is computed, the predictive marginal likelihood on the data can be obtained as $p(\tilde{y}|\boldsymbol{x}, \mathcal{D}) = \int p(\tilde{y}|\boldsymbol{x}, \boldsymbol{w}, \beta) q(\boldsymbol{w}|\mathcal{D}) d\boldsymbol{w}$. However, the analytical solution is intractable due to the nonlinearities

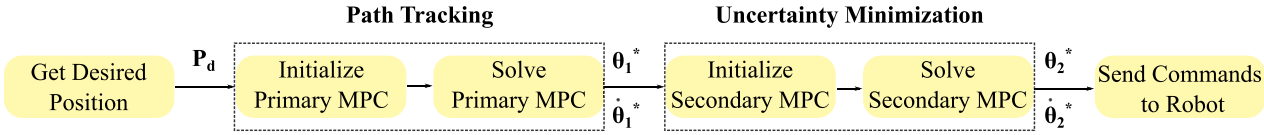

Fig. 2. The Hierarchical MPC (Hi-MPC) controller for path tracking and model's uncertainty minimization.

in the network mapping. By linearizing the network mapping around the optimal weights $\boldsymbol{w}^*$, the predictive distribution on the network output can be computed analytically as [14]:

$$p(\tilde{y}|\boldsymbol{x}, \mathcal{D}, \alpha, \beta) = \mathcal{N}(\tilde{y}|y(\boldsymbol{x}, \boldsymbol{w}^*), \sigma^2(\boldsymbol{x}))$$
$$\text{and } \sigma^2 = \beta^{-1} + \boldsymbol{g}^T \boldsymbol{A}^{-1} \boldsymbol{g} \tag{3}$$

where $\boldsymbol{g}$ is the gradient of the network for the weights. The output of the BNN is therefore a Gaussian distribution with the network model $y(\boldsymbol{x}, \boldsymbol{w}^*)$ as mean, and σ^2 as variance.

The kinematic and dynamic models learned through BNN will therefore be identified by:

$$\tilde{\boldsymbol{P}} = \tilde{\boldsymbol{P}}(\boldsymbol{\theta}), \text{ and } \boldsymbol{\sigma}_P^2 = \boldsymbol{\sigma}_P^2(\boldsymbol{\theta}) \tag{4a}$$

$$\tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\Gamma}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}), \text{ and } \boldsymbol{\sigma}_\tau^2 = \boldsymbol{\sigma}_\tau^2(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}). \tag{4b}$$

To build the models by BNN, Tensorflow Probability framework was used. For both models, $swish$ activation functions [16] are used, which combine both benefits of $relu$ and $sigmoid$, and guarantee continuity in the derivatives.

III. Control Method

In this section we describe the proposed approach for solving the optimal control problem through Hi-MPC.

A. Hierarchical Model Predictive Control

Model predictive control consists in formulating an optimal problem with a cost function to be minimized over a finite prediction horizon, with respect to the control inputs. Once a solution is found, only the first control action is executed and the procedure is repeated by shifting the horizon forward.

In case of redundant systems, strict prioritized motion tasks can be specified and solved while avoiding conflicts with each other. In this work we adopt a strict hierarchical prioritization to ensure that the primary task will always be executed, while avoiding the need of manually updating the weights whenever a new motion is specified (which instead affects soft task prioritization schemes [17]).

In a surgical environment precisely tracking a trajectory is fundamental to guarantee the safety of the patient and reduce traumas. Yet, the tracking accuracy depends on the precision and reliability of the kinematic model. Thanks to the use of BNN, it is possible to have an estimate of the uncertainties in the model and to control the robot to move within safe regions [18]. In order to do that, the additional task of minimizing the kinematic model uncertainties can be added and solved as a lower priority task for the path tracking. The information on the uncertainties in the dynamic model, instead, is used to restrict the torque bounds. For solving a stack of prioritized tasks employing Hi-MPC, two MPC problems are solved: the primary MPC to track a desired path, and the secondary one to guarantee that the robot moves in reliable state-space regions. The hierarchical control structure allows for the exploitation of self-motions to reconfigure the robot, without affecting the tracking accuracy. Fig. 2 shows the flow chart for the Hi-MPC controller. In order to solve the nonlinear MPCs problem, the fast nonlinear MPC ACADO toolkit [19] was used. It is a symbolic solver, which allows C++ code for fast optimized nonlinear MPC control to be generated.

1) Primary MPC for Path Tracking: Since the control interface of the Micro-IGES robot accepts desired positions, in this work we formulate the tracking problem in terms of a cost function minimizing the error between the desired Cartesian 3D position and the current one.

To ensure safety conditions, constraints on joint positions, velocities, and torques can be imposed. Due to these limitations, the desired Cartesian position may not be achieved, leading to a deformation of the followed path. To reduce this issue, a scaling factor s can be introduced [20]. This scaling factor allows reducing the desired rate of change of the path (which equates to a dilation of the execution time), without modifying the path itself.

The optimal control problem is then formulated in terms of the robot's states $\boldsymbol{\theta}$ and controls $\dot{\boldsymbol{\theta}}$ as:

$$\min_{\dot{\boldsymbol{\theta}}_0, \ldots \dot{\boldsymbol{\theta}}_N s_1 \ldots s_N \boldsymbol{\rho}_1 \ldots \boldsymbol{\rho}_N} \frac{1}{2} \sum_{k=1}^{N} ||\tilde{\boldsymbol{P}}_k(\boldsymbol{\theta}_k) - \boldsymbol{P}_{d,k} - s_k \Delta \boldsymbol{P}_{d,k}||^2_{\boldsymbol{W}_p}$$
$$+ W_s(1 - s_k)^2 + ||\tilde{\boldsymbol{\Gamma}}_k||^2_{\boldsymbol{W}_t} + ||\dot{\boldsymbol{\theta}}_k||^2_{\boldsymbol{W}_v} + ||\boldsymbol{\rho}_k||^2_{\boldsymbol{W}_\rho} \tag{5a}$$

$$\text{s.t.} \quad \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{f}(\boldsymbol{\theta}_k, \dot{\boldsymbol{\theta}}_k) \tag{5b}$$

$$\boldsymbol{\theta}_m \leq \boldsymbol{\theta}_k \leq \boldsymbol{\theta}_M \tag{5c}$$

$$\dot{\boldsymbol{\theta}}_m \leq \dot{\boldsymbol{\theta}}_k \leq \dot{\boldsymbol{\theta}}_M \tag{5d}$$

$$\boldsymbol{\tau}_{mot,m} - \boldsymbol{\rho}_k \leq \tilde{\boldsymbol{\Gamma}}_k(\boldsymbol{\theta}_k, \dot{\boldsymbol{\theta}}_k) - 3\boldsymbol{\sigma}_{\tau,k}(\boldsymbol{\theta}_k, \dot{\boldsymbol{\theta}}_k) \tag{5e}$$

$$\tilde{\boldsymbol{\Gamma}}_k(\boldsymbol{\theta}_k, \dot{\boldsymbol{\theta}}_k) + 3\boldsymbol{\sigma}_{\tau,k}(\boldsymbol{\theta}_k, \dot{\boldsymbol{\theta}}_k) \leq \boldsymbol{\tau}_{mot,M} + \boldsymbol{\rho}_k \tag{5f}$$

$$0 \leq s_k \leq 1 \tag{5g}$$

$$\boldsymbol{0} \leq \boldsymbol{\rho}_k \tag{5h}$$

where N is the number of timesteps in the prediction horizon, $\boldsymbol{P}_d, \Delta \boldsymbol{P}_d$ the desired tip position and tip position variation, $\tilde{\boldsymbol{P}}$ and $\tilde{\boldsymbol{\Gamma}}$ the learned kinematic (4a) and dynamic models (4b), the subscripts m, M indicate lower and upper bounds, and $\boldsymbol{W}_p, W_s, \boldsymbol{W}_t, \boldsymbol{W}_v, \boldsymbol{W}_\rho$ are the weights for each component of the cost function. $\boldsymbol{\rho}$ are slack variables, used to avoid possible infeasibilities. For the system's state evolution (5b), a simple integral model is used such that $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \dot{\boldsymbol{\theta}}_k \Delta t$, with Δt being the sampling time. This formulation of the primary MPC is fundamentally the same as in our previous work [6], with

the addition of the kinematic model for control and the uncertainties in the dynamics. These uncertainties are added in the torque constraints (5e) and (5f) to guarantee that the maximum and minimum expected torques (with a probability of 99.7%) reside within the imposed motor torque bounds ($\boldsymbol{\tau}_{mot,m(M)}$). Similarly as in [6], because of the large number of operations ACADO needs to perform to formulate the whole MPC in a symbolic form, to reduce the amount of time needed for the code generation, a linear approximation of the dynamic model is implemented.

2) Secondary MPC for Uncertainty Minimization: The secondary MPC is employed to exploit the redundancies of the robot and move it within reliable state-space regions, closer to those where the data were collected for learning the model. However, the secondary task must not deteriorate the results of the primary, with consequent failure in tracking the desired path. Given the optimal states $\boldsymbol{\theta}_1^*$ and controls $\dot{\boldsymbol{\theta}}_1^*$ from the primary MPC for the whole prediction horizon, the secondary MPC is formulated as:

$$\min_{\dot{\boldsymbol{\theta}}_0,\dots\dot{\boldsymbol{\theta}}_N\boldsymbol{\rho}_1\dots\boldsymbol{\rho}_N} \quad \frac{1}{2}\sum_{k=1}^{N} ||\boldsymbol{\sigma}_P^2(\boldsymbol{\theta}_k,\dot{\boldsymbol{\theta}}_k)||^2_{\boldsymbol{W}_{\sigma_p}} + ||\tilde{\boldsymbol{\Gamma}}_k||^2_{\boldsymbol{W}_t} + ||\dot{\boldsymbol{\theta}}_k||^2_{\boldsymbol{W}_v} + ||\boldsymbol{\rho}_k||^2_{\boldsymbol{W}_\rho} \tag{6a}$$

$$\text{s.t.} \quad \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{f}(\boldsymbol{\theta}_k,\dot{\boldsymbol{\theta}}_k) \tag{6b}$$

$$\boldsymbol{\theta}_m \le \boldsymbol{\theta}_k \le \boldsymbol{\theta}_M \tag{6c}$$

$$\dot{\boldsymbol{\theta}}_m \le \dot{\boldsymbol{\theta}}_k \le \dot{\boldsymbol{\theta}}_M \tag{6d}$$

$$\boldsymbol{\tau}_{mot,m} - \boldsymbol{\rho}_k \le \tilde{\boldsymbol{\Gamma}}_k(\boldsymbol{\theta}_k,\dot{\boldsymbol{\theta}}_k) - 3\boldsymbol{\sigma}_{\tau,k}(\boldsymbol{\theta}_k,\dot{\boldsymbol{\theta}}_k) \tag{6e}$$

$$\tilde{\boldsymbol{\Gamma}}_k(\boldsymbol{\theta}_k,\dot{\boldsymbol{\theta}}_k) + 3\boldsymbol{\sigma}_{\tau,k}(\boldsymbol{\theta}_k,\dot{\boldsymbol{\theta}}_k) \le \boldsymbol{\tau}_{mot,M} + \boldsymbol{\rho}_k \tag{6f}$$

$$\boldsymbol{P}(\boldsymbol{\theta}_k) = \boldsymbol{P}(\boldsymbol{\theta}_{1,k}^*) \tag{6g}$$

$$\mathbf{0} \le \boldsymbol{\rho}_k. \tag{6h}$$

The optimal solution from the primary MPC is used to initialize the secondary one. The same bounds on the motor positions, velocities, and torques as in the primary MPC are imposed. Also, the constraint (6g) is enforced in order to not deteriorate the solution of the primary MPC, and thus guarantee the path tracking. A different strategy could be employed by using the projector onto the null space of the primary task. However, the inclusion of the Jacobian matrix of the primary task into the second one resulted to be too computationally demanding. Finally, the optimal solutions of the secondary MPC are then commanded to the robot.

The analytical derivation of the variance of the kinematic model using BNN, as described in II-C allows exploiting the capabilities of ACADO solver, without the need for any approximation on the uncertainties of the model.

IV. Results

In this section, the results for the micro-IGES modeling and control are shown, by performing tests both in simulation and on the real robot. In particular, with the simulation results, we empirically validate our Hi-MPC and we show on the real robot that our controller outperforms other classical control approaches.

TABLE I
Root Mean Squared Errors (RMSE) Between the Simulated Robot Models And: I) BNN Kinematic Model; Ib) BNN Dynamic Model

(a) RMSE (mm) for the simulated kinematic model

	x	y	z
Train set	0.108	0.148	0.108
Test set	0.109	0.145	0.108

(b) RMSE (Nm) for the simulated dynamic model

	Roll	Elbow 1	Elbow 2	Pitch	Yaw
Train set	0.037	0.046	0.050	0.023	0.022
Test set	0.037	0.050	0.050	0.024	0.025

A. Simulation Environment

For the simulated environment, V-REP [21] is employed. Similarly to the real case scenario, an initial approximated kinematic model is built. The initial model is constructed by means of the Denavit-Hartenberg convention, given the articulated structure of the robot, but by imposing wrong links' lengths, with respect to the actual simulated robot. The link lengths errors were set to $[2,\ 2,\ 3,\ 3,\ 4,\ 3,\ 2.3]$ mm. With regards to the dynamics, larger masses and inertias of each link are considered to guarantee numerical stability of the physics engine. To map the motor values to the joint values, the same motor-to-joint mapping as in [11] is used.

1) Simulated Robot Models: During the training to build the kinematic and dynamic models, the robot is commanded in position to follow three circular paths of 13 mm, 15 mm, 17 mm in the x, y components at different times, and the z oscillating sinusoidally with an amplitude of 3 mm. In this work, a limited space exploration was performed since our focus was only to test the controller's capabilities, not the generalization of the learning method.

The approximated initial model provides the motor commands, which are then used to drive both the accurate simulated model and the approximated one. In the meantime, the expected tip position of the approximated model, the actual tip position of the accurate simulated model, and the robot's torques are collected from the simulator. Gaussian noise is then added to all the measurements. BNN are employed to correct the kinematic model and learn the dynamic model, as discussed in II-B and II-C. The variance β^{-1} of the probability distribution of the data in (3) is the same as the variance of the added noise. The data from the trajectories were stacked together into 11 500 data points, which were then point-wise split into a training set (70%) and a test set (30%). For the kinematic modeling, three different networks are used, one for each Cartesian position, each one with one hidden layer of 20 nodes. The input of each network is the 5-dimensional vector of motor values (II-A). With regards to the dynamic model, 5 independent networks are employed, with the 10-dimensional vector of motor positions and velocities as input, each single torque component as output, and two hidden layers with 20 and 10 nodes respectively. Table I reports the

TABLE II

SIMULATION RESULTS FOR THE TRAJECTORY TRACKING USING ONLY THE PRIMARY MPC (MPC 1) AND HI-MPC. $\dot{\boldsymbol{\theta}}_{m(M)}$ $\boldsymbol{\tau}_{mot,m(M)}$ ARE THE MINIMUM AND MAXIMUM BOUNDS ON THE MOTOR VELOCITIES AND TORQUES, WHICH ARE SET EQUAL FOR EACH MOTOR. IIA REPORTS THE DESIRED EXECUTION TIME T AND THE ACTUAL EXECUTION TIME T_{act}, THE AVERAGE MOTION SCALING FACTOR $\bar{s}$, AND THE MEAN AND MAXIMUM ABSOLUTE POSITIONING ERRORS $|\bar{\boldsymbol{\epsilon}}_P|, |\boldsymbol{\epsilon}_P|_{\text{MAX}}$. IIB REPORTS THE MAXIMUM ABSOLUTE VALUES OF THE MOTOR VELOCITIES $|\dot{\boldsymbol{\theta}}|_{\text{MAX}}$ AND TORQUES $|\dot{\boldsymbol{\tau}}|_{\text{MAX}}$, WITH THE RED VALUES INDICATING VIOLATIONS OF THE BOUNDS

(a) Mean and maximum tracking errors

Shape	Control	T(s)	T_{act}(s)	$\bar{s}$	$\lvert\bar{\epsilon}_P\rvert$(mm)			$\lvert\epsilon_P\rvert_{max}$(mm)		
					x	y	z	x	y	z
Circle	MPC 1	60	63.7	0.94	0.29	0.31	0.78	3.20	4.01	5.10
	Hi – MPC	60	64.1	0.93	0.36	0.34	0.89	3.20	1.50	2.70
Square	MPC 1	75	13.2	0.98	1.20	7.00	1.50	14.50	9.20	4.20
	Hi – MPC	75	78.0	0.96	0.70	0.40	2.10	1.80	1.50	4.20

(b) Maximum absolute motor velocities and torques. The red values are beyond the bounds.

Shape	Control	$\dot{\theta}_{m(M)}$(rad/s)	$\lvert\dot{\theta}\rvert_{max}$(rad/s)					$\tau_{m(M)}$(Nm)	$\lvert\tau\rvert_{max}$(Nm)				
			Roll	Elbow 1	Elbow 2	Pitch	Yaw		Roll	Elbow 1	Elbow 2	Pitch	Yaw
Circle	MPC 1	-100 (100)	42.70	48.40	25.20	45.90	52.50	-10 (10)	1.17	76.60	33.90	17.70	0.50
	Hi – MPC	-100 (100)	3.70	5.30	7.90	7.02	11.29	-10 (10)	0.52	4.14	3.63	3.31	0.07
Square	MPC 1	-100(100)	36.50	72.45	100	100	100	-10 (10)	5.96	63.9	94.20	55.60	5.60
	Hi – MPC	-100(100)	1.94	1.55	1.60	1.43	2.61	-10(10)	0.22	1.17	0.09	0.22	0.01

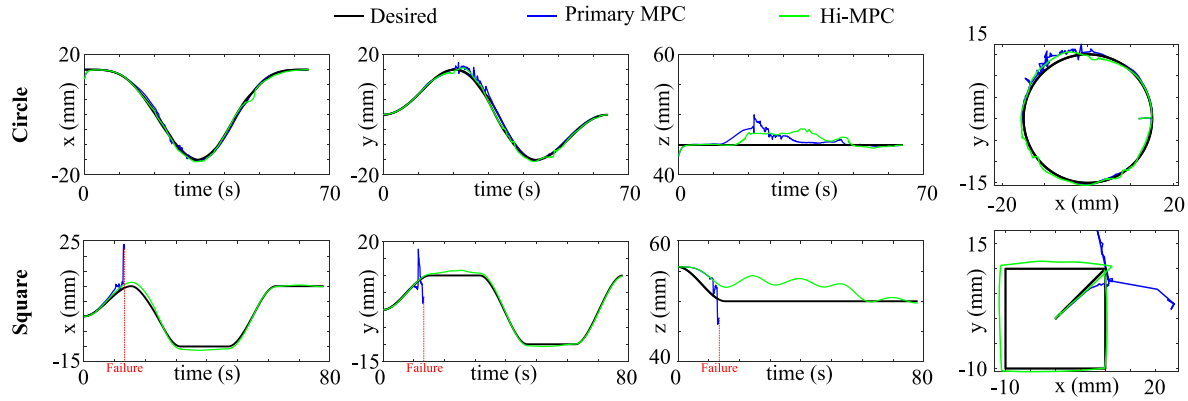

Fig. 3. Results for the simulated trajectory tracking, showing the desired path (black) and the actual tip position when applying only the primary MPC (blue) and Hi-MPC strategy (green). On the square path the primary MPC fails.

Root Mean Squared Errors (RMSE) between the BNN models outputs and the actual Cartesian tip position and motor torques measured from the simulator.

2) Validation of the Hi-MPC: A validation of the proposed Hi-MPC approach and a comparison with the non-hierarchical primary MPC (5) only for path tracking are here shown, roughly simulating a tumor resection task. The robot end-effector is required to follow two different trajectories: a circular path of 15 mm in radius in x, y, while keeping the z fixed at a 45 mm, and a square path on the x, y plane (which was not part of the training set), with each side 20 mm long, and the z fixed at 50 mm. A quintic polynomial-time law is assigned for the circle and for each side of the square. Moreover, in tracking the square, the system starts from the straight home configuration. For all the tests, the sampling time has been set to 200 ms, 10 steps are used for the prediction horizon of both MPCs and the motor positions are bounded between $\pm [280 \quad 55 \quad 47 \quad 74 \quad 74]$ rad. The bounds on the motor positions are a consequence of the motor-to-joint mapping and of the gearbox. As a matter of fact, due to the routing of the tendons around the capstan, the motors may need to complete more than one full turn in order for the joints to reach their limits. The bounds on the torques, instead, are much higher than those on the real system due to the larger mass and inertia values that had to be used to overcome simulation issues. With regards to the sampling time, such a high value is set to make the simulation close to the real case, as described in the following section. Moreover, the weights in the cost function of the MPCs have been heuristically tuned for each test and they are kept the same in both controllers (single primary MPC and Hi-MPC). For tracking the circle the weights have been set to $\boldsymbol{W}_P = 10^7 \boldsymbol{I_3}, W_s = 10^1, \boldsymbol{W}_t = 10^1 \boldsymbol{I_5}, \boldsymbol{W}_v = 10^{-1} \boldsymbol{I_5}$, $\boldsymbol{W}_\rho = 10^8 \boldsymbol{I_5}, \boldsymbol{W}_{\sigma_P} = 10^{10} \boldsymbol{I_3}$, with $\boldsymbol{I_n}$ the identity matrix and n the matrix dimension. For the other tests, the weights have not been reported for conciseness since they are in the same order of magnitude.

Table II and Fig. 3 show the results for the control of the simulated robot when employing only the primary MPC (MPC 1) and the Hi-MPC strategy for additional minimization of the model uncertainties. Due to the imposed bounds, the motion needs to be scaled, resulting in a longer execution time than desired. The proposed Hi-MPC approach clearly outperforms the control with only the primary MPC. In fact, the minimization of the model uncertainties ensures more accuracy in the models, yielding safer and more accurate tracking. Even though the primary MPC achieves smaller mean errors on the circle path, it does not ensure the satisfaction of the imposed bounds.

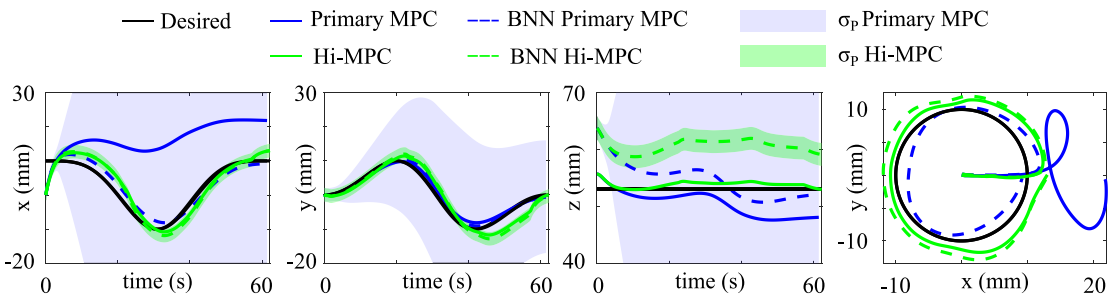

Fig. 4. Simulation tracking test to show how the model uncertainties affect the controller's performances. The black line is the desired path; blue color is for the primary MPC and green for Hi-MPC; the full lines are the actual tip positions, and the dashed lines are the BNN expected outputs. The shaded areas indicated the uncertainty corresponding to the BNN input joint-space configurations associated to each Cartesian position and it is equal to one standard deviation $\boldsymbol{\sigma}_P$ (light blue for primary MPC and green for Hi-MPC).

Moreover, the primary MPC completely fails the tracking of the square path. With Hi-MPC, instead, the minimization of the model uncertainties guarantees that the bounds are all satisfied and small tracking errors are achieved.

To better show how the model uncertainties affect the controller, an additional test to track a 10 mm circle, starting from the home straight configuration, was performed. As shown in Fig. 4, when employing only the primary MPC the BNN prediction becomes inaccurate, due to the controller driving the robot to highly uncertain state-space regions. Because of the increasing inaccuracies, the BNN output diverges from the actual tip position, especially along the x direction, resulting in improper trajectory tracking. With Hi-MPC, instead, the variance is kept low, yielding better tracking and smaller error between the BNN expected output and the actual tip position. Since Hi-MPC drives the robot into more reliable state-space regions, also the uncertainty on the predicted torques $\boldsymbol{\sigma}_\tau$ is reduced: the mean norm of the torque standard deviation is 18.76 Nm and 0.61 Nm for primary MPC and Hi-MPC respectively.

B. Real Robot Experiments

Since the Hi-MPC architecture displayed better performance in the simulation setting we considered only this solution for the experiments on the real robot. In this section, we test our method on the Micro-IGES robot and we provide comparative results for the Cartesian tracking error. For more details about the mechanical structure and control interface of the robot, please refer to II-A.

1) Robot Models: In order to collect data for modeling the Micro-IGES robot, the setup shown in Fig. 5 was adopted, with one electromagnetic (EM) tracker on the shaft's end (after the roll joint) as reference frame and one on the tip. Ideally, the reference frame would be placed at the beginning of the kinematic chain. However, the acquisition range is not enough to track both sensors simultaneously in that configuration, so the roll DOF is fixed in our experiments and the control variables set to $\boldsymbol{\theta} = [\theta_{e1} \quad \theta_{e2} \quad \theta_W \quad \theta_{j1}]$.

The training data points are collected using the same trajectories as in the simulated experiment. In order to avoid damages in the system, however, the z component is not controlled. The initial approximated model [11] is utilized to compute the motor values to command to the robot. For each motor command, the corresponding tip position, by means of the EM trackers, and the motor torques, read directly from the motors, are collected. Due to the low frequency of acquisition of the EM trackers of 5 Hz, 2700 data points were collected.

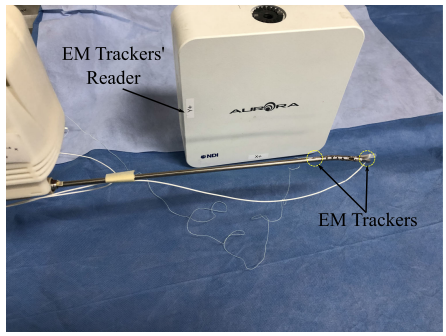

Fig. 5. The experimental setup for data acquisition.

TABLE III
ROOT MEAN SQUARED ERRORS (RMSE) ON THE REAL ROBOT BETWEEN: III) BNN KINEMATIC MODEL AND EM TRACKER MEASUREMENTS; III) BNN DYNAMIC MODEL AND MOTOR TORQUES

(a) RMSE (mm) for the simulated kinematic model

	x	y	z
Train set	1.127	1.010	0.458
Test set	1.114	1.034	0.469

(b) RMSE (10^{-3}Nm) for the simulated dynamic model

	Elbow 1	Elbow 2	Pitch	Yaw
Train set	0.077	0.045	0.055	1.414
Test set	0.264	0.198	0.122	0.200

BNN are employed to learn the kinematic error between the approximated initial model and the measurements and to map the motor values to the motor torques. The same training-test split (70-30) as in the simulated experiment is utilized. Three neural networks for the kinematic modeling are used, each one with a 4-dimensional input of the motor positions, one hidden layer of 20 nodes, and one-dimensional output. For the dynamic modeling, 4 networks are used with the input being 8-dimensional vector of motor positions and velocities, two hidden layers of 30 and 10 nodes, and one-dimensional output.

TABLE IV
REAL WORLD RESULTS FOR THE TRAJECTORY TRACKING USING HI-MPC. $\dot{\boldsymbol{\theta}}_{m(M)}$, $\boldsymbol{\tau}_{mot,m(M)}$ ARE THE MINIMUM AND MAXIMUM BOUNDS ON THE MOTOR VELOCITIES AND TORQUES, WHICH ARE SET EQUAL FOR EACH MOTOR. IV REPORTS THE DESIRED EXECUTION TIME T AND THE ACTUAL EXECUTION TIME T_{act}, THE AVERAGE MOTION SCALING FACTOR $\bar{s}$, AND THE MEAN AND MAXIMUM ABSOLUTE POSITIONING ERRORS $|\bar{\boldsymbol{\epsilon}}_P|$, $|\boldsymbol{\epsilon}_P|_{\text{MAX}}$. IV REPORTS THE MAXIMUM ABSOLUTE VALUES OF THE MOTOR VELOCITIES $|\dot{\boldsymbol{\theta}}|_{\text{MAX}}$ AND TORQUES $|\dot{\boldsymbol{\tau}}|_{\text{MAX}}$

(a) Mean and maximum tracking errors

Shape	T(s)	$\mathbf{T_{act}}$(s)	$\bar{\mathbf{s}}$	$\lvert\bar{\epsilon}_\mathbf{P}\rvert$(mm)			$\lvert\epsilon_\mathbf{P}\rvert_\mathbf{max}$(mm)		
				x	y	z	x	y	z
Circle	60	89.7	0.85	0.82	0.55	1.21	5.28	3.87	3.57
Square	75	101.2	0.89	1.14	1.07	1.40	5.07	5.07	3.22

(b) Maximum absolute motor velocities and torques. The red values are beyond the bounds.

Shape	$\dot{\theta}_{\mathbf{m(M)}}$(rad/s)	$\lvert\dot{\theta}\rvert_\mathbf{max}$(rad/s)				$\tau_{\mathbf{m(M)}} \cdot \mathbf{10^{-3}}$(Nm)	$\lvert\tau\rvert_\mathbf{max} \cdot \mathbf{10^{-3}}$(Nm)			
		Elbow 1	Elbow 2	Pitch	Yaw		Elbow 1	Elbow 2	Pitch	Yaw
Circle	-100 (100)	3.20	4.81	3.17	6.81	-4.3 (4.3)	0.44	0.45	0.53	0.30
Square	-100 (100)	4.14	5.70	3.40	4.98	-4.3 (4.3)	0.47	0.46	0.52	0.34

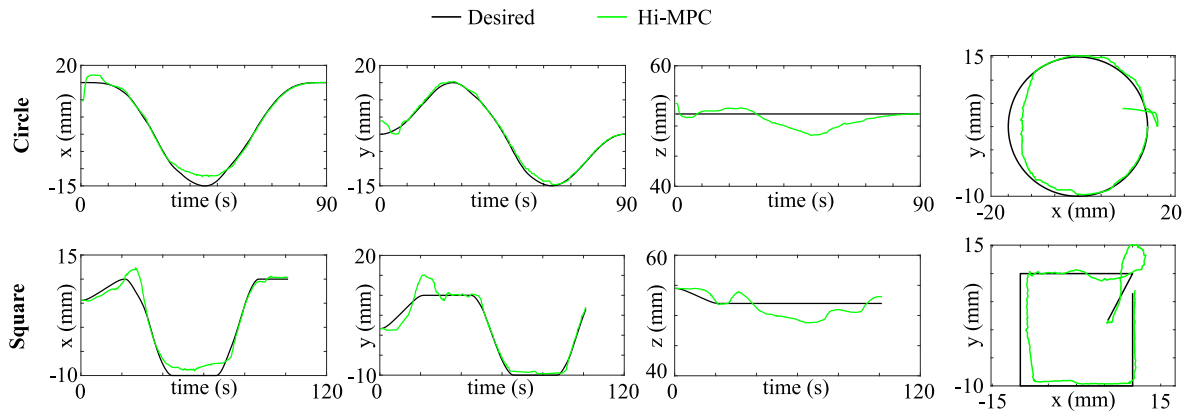

Fig. 6. Results for the trajectory tracking on the real system, showing the desired path (black) and the actual tip position when applying Hi-MPC strategy (green).

Also in the real scenario, the BNN manage to compensate for the kinematic modeling errors and build a satisfactory dynamic model, as shown Table III.

2) Path Tracking Results: In the real scenario, the robot was required to follow the same shapes as in the simulation, but for both the circle and the square the z was fixed at 52 mm. The same motor position bounds as in the simulation are imposed, and the torque bounds are set to the real motor torques limits of $4.3 \cdot 10^{-3}$ Nm. Regarding the control strategy, 10 time steps were used for the MPCs, with a sampling time of 200 ms, due to the acquisition frequency of the EM trackers. The weights are set to $\boldsymbol{W}_P = 10^7 \boldsymbol{I_3}$, $W_s = 10^1$, $\boldsymbol{W}_t = 10^0 \boldsymbol{I_4}$, $\boldsymbol{W}_v = 10^0 \boldsymbol{I_4}$, $\boldsymbol{W}_\rho = 10^8 \boldsymbol{I_4}$, $\boldsymbol{W}_{\sigma_P} = 10^{10} \boldsymbol{I_3}$ for both paths.

Table IV and Figure 6 report the results for tracking in the real case scenario. On both shapes, the bounds are always satisfied and, in order to guarantee them, the motion is slowed down, yielding longer execution times. On the circular path, good tracking accuracy is achieved, with larger errors at the beginning due to an inaccurate initial positioning. On the square path, instead, the accuracy is less, with some overshooting when reaching the first corner.

During the execution of the experiments on the real system, we noticed two main sources of errors. Firstly, the backlash and hysteresis effects on the robots are still relevant. Due to an initial manual straightening to reach the home configuration, the possible positioning inaccuracies and slacking or elongation of the tendons would lead to less accurate tracking or failure in the control. The backlash and hysteresis effects are also the sources of errors in tracking the desired shapes in the negative x direction. Secondly, the inaccuracies in the lower level controller do not allow the motors to perfectly reach the desired motor commands, with a consequent inaccurate tip positioning.

TABLE V
MEAN AND MAXIMUM ABSOLUTE TIP POSITIONING ERRORS FOR THE HI-MPC AND CLASSICAL PSEUDOINVERSE APPROACHES. THE VALUES DO NOT CONSIDER THE INITIAL POSITIONING ERRORS

Control	$\lvert\bar{\epsilon}_\mathbf{P}\rvert$(mm)			$\lvert\epsilon_\mathbf{P}\rvert_\mathbf{max}$(mm)		
	x	y	z	x	y	z
Classical	1.91	1.21	0.77	6.28	4.66	3.29
Hi − MPC	0.82	0.55	1.21	3.13	1.67	3.57

3) Comparison With a Classical Controller: We finally compared our method with a pseudo-inverse kinematic controller that is largely employed in the control of tendon driven system [11], [22], [23]. Table V and Fig. 7 show the experimental results obtained using both methods. The comparison is meant to show only the tracking accuracy, as the traditional approach does not include bounds. Despite the initial positioning errors, the proposed Hi-MPC outperforms the traditional approach, even though slightly larger errors on the z component occur. Due to the constraints and the scaling factor, a slower motion is achieved with Hi-MPC.

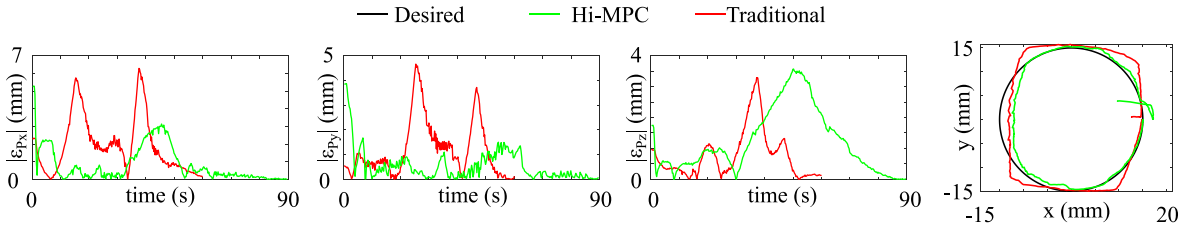

Fig. 7. Absolute tip positioning errors (left three plots) and Cartesian path for the Hi-MPC (green) and the traditional pseudoinverse (red) approaches.

V. Conclusions

In conclusion, in this work we presented an approach to model the kinematics and dynamics of a highly nonlinear robotic system by means of BNN and control it with a Hierarchical MPC (Hi-MPC) approach, under safety constraints in the kinematics and dynamics. BNN allow having an estimate of the uncertainties in the model and the Hi-MPC exploits this information to increase the reliability of the models, thus leading to safer and more accurate control. This is important in application scenarios like minimally invasive surgery, where high motion accuracy and safety must be guaranteed in order to limit patients' traumas.

Future work will focus on improving the effectiveness of the proposed method to include hysteresis and backlash in the models, include contact forces to perform the tasks when interacting with the surrounding environment, and we will further investigate the role of uncertainty reduction with additional experiments.

References

[1] H. M. Le, T. N. Do, and S. J. Phee, "A survey on actuators-driven surgical robots," *Sensors Actuators, A: Phys.*, vol. 247, no. 8, pp. 323–354, Aug. 2016.

[2] T. G. Thuruthel, Y. Ansari, E. Falotico, and C. Laschi, "Control strategies for soft robotic manipulators: A survey," *Soft Robotics*, vol. 5, no. 2, pp. 149–163, Apr. 2018.

[3] F. Cursi, G. P. Mylonas, and P. Kormushev, "Adaptive kinematic modelling for multiobjective control of a redundant surgical robotic tool," *Robot.*, vol. 9, no. 3, Aug. 2020, Art. no. 68.

[4] W. Xu, J. Chen, H. Y. Lau, and H. Ren, "Data-driven methods towards learning the highly nonlinear inverse kinematics of tendon-driven surgical manipulators," *Int. J. Med. Robot. Comput. Assist. Surg.*, vol. 13, no. 3, Sep. 2017, Art. no. e1774.

[5] A. L. Dontchev, I. V. Kolmanovsky, M. I. Krastanov, V. M. Veliov, and P. T. Vuong, "Approximating optimal finite horizon feedback by model predictive control," *Syst. Control Lett.*, vol. 139, May 2020, Art. no. 104666.

[6] F. Cursi, V. Modugno, and P. Kormushev, "Model predictive control for a tendon-driven surgical robot with safety constraints in kinematics and dynamics," in *Proc. Int. Conf. Intelli. Robot. Syst.*, 2020, pp. 7653–7660.

[7] F. Zhong, Y. Wang, Z. Wang, and Y. H. Liu, "Dual-arm robotic needle insertion with active tissue deformation for autonomous suturing," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 2669–2676, Jul. 2019.

[8] M. Capotondi, G. Turrisi, C. Gaz, V. Modugno, G. Oriolo, and A. De Luca, "An online learning procedure for feedback linearization control without torque measurements," in *Proc. Conf. Robot Learn., (ser. Proc. Mach. Learn. Research)*, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds., vol. 100, pp. 1359–1368, Mar. 2020.

[9] A. Del Prete, F. Romano, L. Natale, G. Metta, G. Sandini, and F. Nori, "Prioritized optimal control," in *Proc. - IEEE Int. Conf. Robot. Automat.* , 2014, pp. 2540–2545.

[10] J. Shang *et al.*, "A single-port robotic system for transanal microsurgery-design and validation," *IEEE Robot. Automat. Lett.*, vol. 2, no. 3, pp. 1510–1517, Jul. 2017.

[11] K. Leibrandt *et al.*, "Effective manipulation in confined spaces of highly articulated robotic instruments for single access surgery," *IEEE Robot. Automat. Lett.*, vol. 2, no. 3, pp. 1704–1711, Jul. 2017.

[12] F. Cursi and G.-Z. Yang, "A novel approach for outlier detection and robust sensory data model learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4250–4257.

[13] A. Vehtari and J. Lampinen, "Bayesian neural networks: Case studies in industrial applications," in *Soft Comput. Ind. Appl.*, 2000, pp. 415–424.

[14] C. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.

[15] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, "Flipout: Efficient pseudo-independent weight perturbations on mini-batches," in *Proc. 6th Int. Conf. Learn. Representations, - Conf. Track Proc. Int. Conf. Learn. Representations*, March 2018.

[16] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," in *Proc. 6th Int. Conf. Learn. Representations, ICLR 2018 - Workshop Track Proc.*, Oct. 2017.

[17] V. Modugno, U. Chervet, G. Oriolo, and S. Ivaldi, "Learning soft task priorities for safe control of humanoid robots with constrained stochastic optimization," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robot.*, 2016, pp. 101–108.

[18] Y. Yun and A. D. Deshpande, "Control in the reliable region of a statistical model with gaussian process regression,"in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, 2014, pp. 654–660.

[19] B. Houska, H. J. Ferreau, and M. Diehl, "ACADO toolkit-an open-source framework for automatic control and dynamic optimization," *Optimal Control Appl. Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[20] M. Faroni, M. Beschi, N. Pedrocchi, and A. Visioli, "Predictive inverse kinematics for redundant manipulators with task scaling and kinematic constraints," *IEEE Trans. Robot.*, vol. 35, no. 1, pp. 278–285, Feb. 2019.

[21] E. Rohmer, S. P. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *IEEE Int. Conf. Intell. Robots Syst.*, 2013, pp. 1321–1326.

[22] H. Sadeghian, F. Zokaei, and S. Hadian Jazi, "Constrained kinematic control in minimally invasive robotic surgery subject to remote center of motion constraint," *J. Intell. Robot. Syst.*, vol. 95, no. 3–4, pp. 901–913, 2019.

[23] P. Berthet-Rayne, K. Leibrandt, G. Gras, P. Fraisse, A. Crosnier, and G.-Z. Yang, "Inverse kinematics control methods for redundant snakelike robot teleoperation during minimally invasive surgery," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 2501–2508, Jul. 2018.