A Deep Reinforcement Learning Approach for Inverse Kinematics of Concentric Tube Robots

K. Iyengar, G. Dwyer, D. Stoyanov

Surgical Robot Vision Group, University College of London
keshav.iyengar.17@ucl.ac.uk

INTRODUCTION

Minimally invasive surgical procedures are performed with a small incision for surgical instruments to pass through to reach the stage of the procedure. Dexterous surgical instruments like concentric tube robots (CTR) are needed to steer along sensitive structures within the body and achieve minimal damage by employing a remote centre of motion (RCM) at the incision point. A CTR is a continuum robot composed of multiple telescopic, concentric, pre-curved, super-elastic tubes that can be axially translated and rotated at their base relative to each other [1]. The bending is derived from the elastic tube interactions with neighboring tubes, allowing for high dexterity while maintaining a small footprint. Along with the CTR, which is known as the distal configuration, there are outer degrees of freedom (DOF) that fix the CTR's base to a remote center of motion known as the proximal configuration. Kinematic modelling of such systems is non-trivial due to the complex interaction of individual tubes with neighboring tubes that form unique bending curves. Previously, traditional iterative approaches have been used to moderate success, but challenges include model complexity and a reliance on material constants. Previous work on introducing model-free solution include solutions with feed-forward neural networks [5]. Although these solutions are accurate, the cost of data sampling is high as all training, validation and testing is done on a physical CTR.

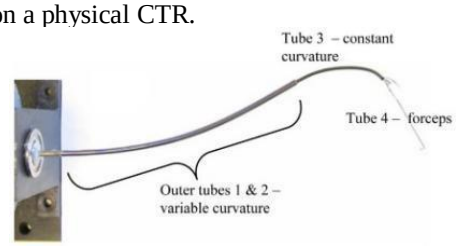

Figure. 1. Distal configuration of 3 tubes [1].

This paper presents a deep reinforcement learning approach to solving the inverse kinematics of CTRs. The approach known as deep deterministic policy gradient (DDPG) [2] has shown promising results in high dimensional, continuous control problems such as humanoid robot control [3] and RC car drifting [4]. DDPG is an off-policy, actor-critic based algorithm that uses experience replay. Fully-connected neural networks are used to model the actor and critic.

With this work, the authors investigate using DDPG for CTRs in simulation for 3 configurations, distal, proximal and full. The distal configuration consists of a rotational and prismatic DOF per tube and the proximal configuration has 3 rotational and 1 prismatic DOF to maintain a remote centre of motion.

MATERIALS AND METHODS

State, Action and Reward Definitions

The state formulation consists of the joint states, q, desired position, $X_{desired}$, and achieved position, $X_{achieved}$. The proximal configuration has three rotational DOFs, ψ, ϕ and θ about the *x*, *y* and *z* axis and a prismatic DOF, *r*, about the *z* axis. The distal configuration has a rotational DOF, γ_i, about the z-axis and prismatic DOF, l_i , about the z-axis for each tube *i* of a *n* tube CTR. The state *s* is defined in equations 1, 2, 3.

(1) $s = [q \quad X_{achieved} \quad X_{desired}]$

(2) $q_{distal} = [\gamma_1 \quad l_1 \quad \dots \quad \gamma_n \quad l_n]$

(3) $q_{proximal} = [\psi \quad \phi \quad \theta \quad r]$

(4) $q_{distal} = [\psi \quad \phi \quad \theta \quad r \quad \gamma_1 \quad l_1 \quad \dots \quad \gamma_n \quad l_n]$

The action is defined as the change in joint state, q in equation 4.

(5) $a = \Delta q$

A novel reward function was formulated specific to CTRs. α and β are normalization constants for equal weighting of the norm-distance error term and change in joint states term. α is the multiplicative inverse of the longest normal distance between two points in the achievable workspace. In most cases, the minimum and maximum joint state values provide the two points with the largest normal distance. β is the multiplicative inverse of the number of active joints depending on the configuration.

(6) $$r = -\alpha|X_{achieved} - X_{desired} - \beta\sqrt{\sum_i^n \left(\frac{\Delta q_i}{q_{i,max} - q_{i,min}}\right)^2}$$

Agent Model

The exploration strategy chosen was zero-mean Gaussian noise where the variance of the noise decreases with each time step *t*, proportional to a decay period, given the starting variance $\sigma_{max,}$ and final variance σ_{min}.

(7) $$\sigma_t = \sigma_{max} - (\sigma_{max} - \sigma_{min}) \times \min\left(1, \frac{t}{T}\right)$$

The fully-connected neural network architectures of the actor and critic (including target networks) have 2 hidden layers. The number of neurons at each hidden layer differs based on the configuration of the CTR as show in table 1.

Configuration	Hidden Layer 1	Hidden Layer 2
Distal	50	10
Proximal	100	50
Full	200	100

Table. 1. Hidden Layer Configurations.

Environment Model

The forward kinematics model utilized the dominating stiffness model [1] for CTRs which assumes that the bending stiffness of one tube is much larger than the neighboring tube, resulting in the neighboring tube conforming to the curvature of the stiffer tube. This model was used to generate achievable 3D goal positions for the end effector and track the current position of the end effector in simulation. Given k is the curvature of all tubes and l_{tip} is the length of the tip of the end effector the transformations from the remote center of motion to the end effector (l_{tip}) were formulated.

(8) $$T_{shaft}^{RCM}(\omega, \phi, \theta, r) = T_{RotX}(\psi)T_{RotY}(\phi)T_{RotZ}(\theta)T_{TransZ}(r)$$

(9) $$\mathrm{T_n^{shaft}}(\gamma_1 \quad l_1 \quad \dots \quad \gamma_n \quad l_n) = \prod_{i=1}^{n} \mathrm{T_{RotZ}}(\gamma_i)\mathrm{T_{TransX}}\left(\frac{1-\cos(\mathrm{kl_i})}{\mathrm{k}}\right)\mathrm{T_{TransZ}}\left(\frac{1-\sin(\mathrm{kl_i})}{\mathrm{k}}\right)\mathrm{T_{RotY}}(\mathrm{kl_i})$$

(10) $$\mathrm{T_{l_{tip}}^{RCM}} = \mathrm{T_{shaft}^{RCM}}(\omega, \phi, \theta, \mathrm{r})\mathrm{T_n^{shaft}}(\gamma_1 \quad l_1 \quad \dots \quad \gamma_n \quad l_n)$$

RESULTS

For the distal configuration with n = 1, after every 200 episodes, a 100 episode rollout was performed. The rollout gave an accuracy measure based on the number times the desired goal was reached. For the other configurations, the desired goal was never within the tolerance, so this measure was not possible. After 3000 episodes, the distal configuration rollout gave an average accuracy of 18.3%, a minimum accuracy of 13.0% and a maximum accuracy 23.3% done over 3 seeds with the error plot shown in figure 2.

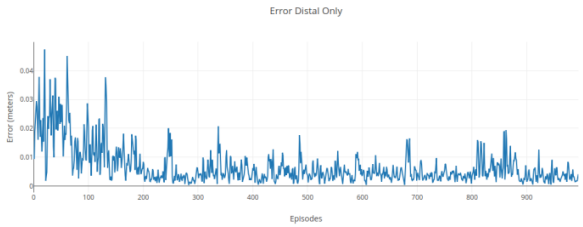

Figure. 2. Error for distal configuration.

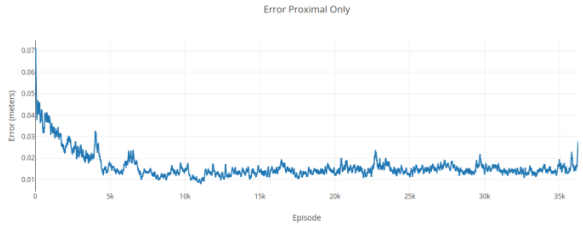

Figure. 3. Error for proximal configuration.

For the proximal configuration n = 1, after 10000 episodes, the goal tolerance was not achieved. Over 3 seeds, the average error was 0.004 meters, minimum error was 0.001 meters and maximum error was 0.005 meters. For the full configuration n = 1, after 10000 episodes, the goal tolerance was not achieved. Over 3 seeds, the average error was 0.010 meters, minimum error was 0.005 meters and maximum error was 0.013 meters.

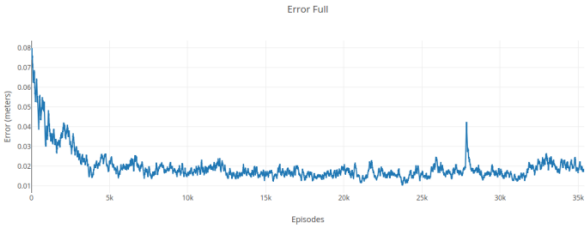

Figure. 4. Error for full configuration.

CONCLUSION AND DISCUSSION

Figure 2, 3 and 4 all demonstrate learning and optimization is occurring which is promising for future work in continuum robot control applications. Note however, the results are no where accurate to either te iterative Jacobian methods or the feed-forward neural network approach [5]. Because of the low accuracy, a physical robot was not used for experimentation. Three main domains of interest to improve the accuracy of DDPG are the exploration strategy, reward function formulation and transfer learning. First, without fully exploring the solution space, DDPG cannot find a good policy, better exploration strategies such as parameter noise exploration can be investigated. Second, the reward function can be improved upon by better representing the requirements of the solution mathematically. Last, DDPG must learn with no prior knowledge of the model resulting in long convergence. Contextualizing the learning by incrementally advancing the complexity using transfer learning could help convergence times as shown in autonomous RC car drifting [4]. In the future, a more complex model should be used along with this simple model and a physical CTR to perform deep learning with forward and reverse transfer learning. Although no expermintation was done, the results are a first step to introduce a data efficent deep learning approach for CTRs.

REFERENCES

[1] P. Sears and P. Dupont, “A Steerable Needle Technology Using Curved Concentric Tubes,” in 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.

[2] D. Silver et al, “Mastering the game of go with deep neural networks and tree search,” Nature, vol. 529, pp. 484–489, January 2016.

[3] S. Phaniteja, Dewangan, et al, “A deep reinforcement learning approach for dynamically stable inverse kinematics of humanoid robots,” 2018.

[4] M. Cutler and J. P. How, “Autonomous drifting using simulation-aided reinforcement learning,” in 2016 IEEE International Conference on Robotics and Automation (ICRA), IEEE, May 2016.

[5] Grassmann, Reinhard et al. (2018). Learning the Forward and Inverse Kinematics of a 6-DOF Concentric Tube Continuum Robot in SE(3). 5125-5132. 10.1109/IROS.2018.8594451.