Optimizing Design Parameters for Sets of Concentric Tube Robots using Sampling-based Motion Planning

Cenk Baykal, Luis G. Torres, and Ron Alterovitz

Abstract—Concentric tube robots are tentacle-like medical robots that can bend around anatomical obstacles to access hard-to-reach clinical targets. The component tubes of these robots can be swapped prior to performing a task in order to customize the robot’s behavior and reachable workspace. Optimizing a robot’s design by appropriately selecting tube parameters can improve the robot’s effectiveness on a procedure- and patient-specific basis. In this paper, we present an algorithm that generates sets of concentric tube robot designs that can collectively maximize the reachable percentage of a given goal region in the human body. Our algorithm combines a search in the design space of a concentric tube robot using a global optimization method with a sampling-based motion planner in the robot’s configuration space in order to find sets of designs that enable motions to goal regions while avoiding contact with anatomical obstacles. We demonstrate the effectiveness of our algorithm in a simulated scenario based on lung anatomy.

I. INTRODUCTION

Concentric tube robots are tentacle-like medical robots that can potentially enable safer minimally invasive interventions at many sites in the human body, including the lungs, the skull base, and the heart [1]. These robots are composed of nested nitinol tubes that each are precurved, typically with a straight segment followed by a curved segment. To perform a task, the robot axially rotates and translates each tube relative to one another, causing the entire device’s shape to change. Concentric tube robots act like shape-changing robotic needles that can curve around anatomical obstacles (e.g., bones, blood vessels, critical nerves, etc.) to reach clinical targets not easily accessed using traditional straight medical instruments.

The curvilinear shapes achievable by concentric tube robots are highly dependent on the component tubes’ physical specifications. The design of the concentric tubes, including the tubes’ lengths and precurvatures, affects the robot’s workspace and the space of the robot’s attainable shapes. Consequently, the design of the concentric tubes determines the set of clinical targets that the robot can safely reach.

Even with the shape-changing capabilities of a concentric tube robot (as shown in Fig. 2), due to kinematic constraints a single design is often not capable of reaching all targets in a physician-specified goal region. Fortunately, concentric tube robots can be built to facilitate fast swapping of tubes of varying physical specifications; selecting tubes for a particular task could maximize the robot’s efficacy during the procedure. In this paper we introduce a new algorithm to efficiently compute a set of designs for a concentric tube robot, such that this set of designs can be sequentially swapped into the robot to access as much of the goal region as feasibly possible while avoiding anatomical obstacles. Fig. 1 shows that two designs collectively can reach more targets than one design.

The methods we propose could be used to create designs for classes of procedures or on a patient-specific basis. Prior
to a procedure, physicians typically obtain a CT scan or MRI of the relevant anatomy, and we can use these volumetric images to segment (either manually or via automatic segmentation software) the goal region as well as anatomical obstacles that must be avoided. Unfortunately, the complex kinematics of concentric tube robots makes it difficult to assess if a given design of concentric tubes can safely access a given target while avoiding anatomical obstacles. As the device’s tip moves, the shape of the shaft of the device may change substantially, and this shape change must be considered to ensure obstacle avoidance. We previously addressed the challenge of computing a single design to reach a finite set of points by using a sampling-based motion planning method that explicitly considered the shape of the entire device en route to a target point [2]. In this paper we build on the prior approach and introduce algorithms that efficiently find a set of designs that can reach as many targets as possible inside a specified goal region while avoiding obstacles along the entire shaft of the robot.

We consider two variants of the problem: (1) finding a set of designs of a fixed size that maximizes the percentage of a goal region that can be reached and (2) minimizing the size of a set of designs that reaches a desired percentage of a goal region. The optimization should be fast, especially for patient-specific design optimization, so as to minimize the required time between patient imaging and task performance. Our algorithm interleaves a search in the concentric tube robot’s design space (i.e., the lengths and precurvatures of the robot’s component tubes) with a motion planner in the robot’s configuration space (i.e., the rotations and translations of the robot’s tubes). We use Adaptive Simulated Annealing (ASA) with parallel computing to accelerate the design space search and the Rapidly-exploring Random Trees (RRT) motion planning algorithm to quickly evaluate the goal region reachability of candidate designs. We demonstrate the effectiveness of our approach in finding sets of designs for concentric tubes using anatomy-inspired scenarios.

II. RELATED WORK

Computing multiple designs for a concentric tube robot can be seen as a generalization of the problem of finding one design that best performs a task. Several approaches have previously been developed for the single design problem. Bergeles et al. proposed a powerful optimization framework for generating robot tube designs that can reach sets of points subject to anatomical constraints, and then applied this method to brain and heart surgery scenarios [3]. They achieve computational tractability by (1) reducing the motion planning problem to finding individual configurations that can reach each specified task point, and (2) using a simpler and faster kinematic model for the general optimization, and then refining the solution using a more accurate (but slower) kinematic model. Although this method works well for a variety of cases, the assumptions that enable computational tractability can sometimes yield suboptimal solutions [2]. This can happen because the method does not explicitly consider the entire robot deployment to the target site [2].

Ha et al. presented a method for designing concentric tube robots while maximizing device stability [4]. The method complements this paper’s approach to computing sets of concentric tube robot designs.

Burgner et al. addressed the problem of finding a concentric tube robot design that maximized the reachable region of points in the sella of the human skull, subject to physical constraints imposed by the bones in the skull [5]. They achieved this by performing a nonlinear optimization over the design space of the robot; they quantify how much of the sella is reachable under a given design by computing the forward kinematics over a grid on the robot’s configuration space. Their approach is well-suited for the neurosurgical scenario in question, but can be subject to the same suboptimal solutions as the work by Bergeles et al. due to not considering the entire robot deployment to the surgical task site.

We take an alternative approach that explicitly considers the entire robot deployment, i.e., the complete motion that the robot has to undertake, to reach the target site. Due to tube interactions, the robot’s tip during deployment will likely not trace the shape of the concentric tubes at the final configuration. In this paper we extend previous work that considered full deployment [2] to the case of computing sets of designs. We also use an improved search strategy that enables faster convergence to higher quality solutions.

Designing a concentric tube robot in advance to perform a particular task requires accurate kinematic modeling. Kinematic modeling of concentric tube robots has rapidly
progressed in accuracy and sophistication [6], [7], [8], [9], [10], [11]. In this paper we used a mechanics-based model developed by Rucker et al. [12].

Our approach depends on the ability to determine the positions in the anatomy that are safely reachable by a concentric tube robot. Burgner-Kahrs et al. developed a method to characterize the workspace of concentric tube robots [13]. We instead take a motion planning approach in order to find points that are reachable by robot motions that avoid contact with anatomical obstacles. Prior work in motion planning for concentric tube robots includes planners using simplified [14], [15] and mechanics-based [16], [17] kinematic models. In this paper we aim to accurately approximate the robot’s reachability, so we use an accurate kinematic model combined with a motion planning algorithm.

The problem of optimizing robot design has been addressed in previous work for serial manipulators. Prior work has used genetic algorithms to optimize the structure of robot’s component tubes that are selected based methods [24]. We explore an alternative approach that interval analysis [22], geometric methods [23], and grid-based methods [24]. We explore an alternative approach that can handle the complex kinematics of concentric tube robots.

III. Problem Definition

A concentric tube robot design \( d \) is the set of physical parameters of the robot’s component tubes that are selected and fixed before performing a clinical task. Specifically, we describe each tube’s design with the following 3 parameters.

- \( L_i \): length of tube \( i \)’s straight section
- \( L_i^* \): length of tube \( i \)’s pre-curved section
- \( \kappa_i \): curvature of tube \( i \)’s pre-curved section

Therefore, a concentric tube robot composed of \( n \) tubes has a design space \( D \) with \( 3n \) parameters, i.e., \( D \subset \mathbb{R}^{3n} \).

During operation of a concentric tube robot, each tube can be independently axially rotated and translated, meaning that an \( n \)-tube robot’s configuration is a \( 2n \)-dimensional vector \( q \). The robot’s configuration space is \( Q \subset (S^1)^n \times \mathbb{R}^n \).

We represent the shape of a concentric tube robot during operation as a 3D space curve that depends on (1) the robot’s configuration \( q \) and (2) the robot’s design \( d \). We therefore denote the robot’s shape as a function \( x(q,d,s) : Q \times D \times [0,1] \rightarrow \mathbb{R}^3 \), where \( x(q,d) \) is a space curve parameterized by \( s \). The positions of the robot’s insertion point and end-effector correspond to \( s = 0 \) and \( s = 1 \), respectively. We compute \( x \) using an accurate mechanics-based model of concentric tube interactions [12].

Safe operation of a concentric tube robot requires that we avoid collisions between the robot’s shaft and anatomical obstacles such as bones, blood vessels, and sensitive tissue. We define the anatomical obstacles \( O \subset \mathbb{R}^3 \) as all 3D points in space that should never intersect with the robot’s shape \( x \). We can determine \( O \) via manual or automatic segmentation of the patient’s preoperative medical imaging [25]. We denote the collision-free subset of robot configurations as \( Q^{\text{free}}_d \), which depends on the robot’s current design \( d \) since the robot’s design affects the robot’s shape \( x \) at any configuration.

We wish to find concentric tube robot designs that can access targets by performing collision-free motion. We consider a motion plan \( \Pi = (q_1, \ldots, q_r) \) to be collision-free if the continuous motion to each subsequent configuration \( q_i \) is all collision-free (i.e., free of intersection with \( O \)).

We focus on finding concentric tube robot designs that allow for collision-free motions so the robot’s tip reaches as many points as possible in a goal region \( G \subset \mathbb{R}^3 \), which is identified in medical images by physicians in a manner similar to obstacles. We emphasize that this goal region is different from typical motion planning problems since we want to reach as many points as possible in the goal region rather than finding a single collision-free motion to any point in the goal region.

We denote the set of points that a concentric tube robot can reach by collision-free motions as \( W(d) \subset \mathbb{R}^3 \) (i.e., the reachable workspace). We can therefore quantify the quality of a given design by the percentage of \( G \) that lies in the robot’s reachable workspace \( W \). For computational feasibility, we discretize the goal region \( G \) into a countable and finite set of voxels \( V \) (i.e., cells in a 3D grid). We can therefore compute the reachable goal percentage of a given design \( d \) as \( |\text{VoxelsReachable}(d)|/|V| \), where \( \text{VoxelsReachable}(d) \) is the set of goal voxels in \( V \) within the reachable workspace of a robot of design \( d \) in an environment with obstacles \( O \). When computing \( \text{VoxelsReachable} \), we emphasize that we must consider the entire sequence of motions executed to reach a goal voxel from the robot’s starting configuration.

We first consider finding a set of robot designs whose union of reachable workspaces covers as much of the goal region as possible. The reachable goal percentage \( r \) of a given design set \( S \) is

\[
 r(S) = \frac{|\bigcup_{d \in S} \text{VoxelsReachable}(d)|}{|V|}. \tag{1}
\]

Finally, we wish to find a set \( S^* \) of robot designs that is minimal (in cardinality) but with a reachable goal percentage greater than a physician-specified threshold \( r_{\text{threshold}} \):

\[
 S^* = \arg\min_{S \subset 2^D} |S|, \quad \text{s.t.} \quad r(S^*) > r_{\text{threshold}}, \tag{2}
\]

where \( 2^D \) is the set of all possible sets of designs.

IV. Methods

We optimize a set of robot designs by interleaving a guided sampling-based search in the robot’s design space with a sampling-based motion planner in the robot’s configuration space. For the design space, we use a global optimization algorithm called Adaptive Simulated Annealing (ASA) [26]. For motion planning in the configuration space, we use the Rapidly-exploring Random Tree (RRT) [27] algorithm. We use ASA to sample a group of robot designs, and then we use RRT to evaluate this group’s reachable goal percentage. We iterate on this process to find a set of designs that can collectively reach a maximal percentage of the goal region or to find a design set of minimal cardinality.
A. Computing Reachable Goal Percentage

According to Eq. 1, in order to evaluate the reachable goal percentage $r$ of a set of designs $S$, we need to compute $\text{VoxelsReachable}(d)$ for each design $d$ in the set. Checking whether a given voxel can be reached by a collision-free motion is equivalent to solving the motion planning problem, which is known to be PSPACE-hard [28]. This implies that, in order to generate solutions in a feasible amount of time, we must accept approximate solutions. We therefore use a probabilistic, sampling-based motion planning algorithm, RRT [27], that can quickly compute an approximation of the robot’s reachable workspace.

RRT incrementally builds a tree of robot configurations that can be reached by collision-free motions from a given start configuration under a given design $d$. After a given number $t$ of iterations of RRT, we iterate over each configuration $q$ in the tree to check which goal voxels can be reached from these configurations. In this way we compute an approximation of $\text{VoxelsReachable}(d_i)$ for each $d_i$ in a given design set $S$, and then we use Eq. 1 to compute the design set’s approximate reachable goal percentage $\hat{r}_i(S)$. We use the $t$ in $\hat{r}_i(S)$ to denote that this approximation was generated using $t$ iterations of the RRT algorithm. We note that the nature of RRT’s reachable workspace approximation is such that we never overestimate the design set’s true reachable goal percentage, i.e., $\hat{r}_i(S) \leq r_i(S)$. RRT also provides probabilistic completeness, a useful property in which the longer we execute the RRT algorithm, the more likely it is to find a collision-free motion to a given target (if a feasible motion plan exists). This implies that, as we increase the iterations $t$ of RRT, the probability of our approximation $\hat{r}_i(S)$ being equal to the true $r_i(S)$ approaches 100%. For a given design set $S$, we compute each $\text{VoxelsReachable}(d_i)$ in parallel for a considerable computational speedup. We compute the reachability of a set of designs $S$ by computing the union of all $\text{VoxelsReachable}(d_i)$ for all $d_i \in S$.

B. Finding a Design Set of Fixed Size

In this section we describe how we find a set of designs of fixed size (i.e., $|S| = m$) that collectively maximize the reachable goal percentage. We utilize the method in the previous section for computing the goal reachability of a set of designs. We provide pseudocode in Alg. 1 and Alg. 2.

For a design set of fixed size, the space of possible sets of designs is $D^m$. We will refer to members of the set $D^m$ as states. For an $n$-tube concentric tube robot, this problem is a $3nm$-dimensional search for a design set (i.e., a state) with maximal goal reachability. Due to the high dimensionality of the search space, we opt for a stochastic approach based on the adaptive simulated annealing (ASA) algorithm. We use ASA because it is a global optimization algorithm that can escape local optima during the search for better design sets.

ASA is always centered on a “current” state $S_{\text{current}}$ in the search space. At the beginning, ASA tends to sample states far away from $S_{\text{current}}$ in order to adequately explore the space. As ASA progresses, it tends to sample states nearer and nearer to $S_{\text{current}}$ in order to make local refinements. ASA controls this sampling variance using a temperature parameter $T$ that decreases with each iteration of ASA. Whenever ASA samples a state $S_{\text{sample}}$ with a lower cost than that of $S_{\text{current}}$, ASA updates $S_{\text{current}}$ to be equal to this new sample $S_{\text{sample}}$. Additionally, if the cost of the new sample is higher than that of the current state, ASA might still update to the new sample with an acceptance probability that decreases over time (also controlled by the temperature $T$). This potential to take steps of increasing costs allows ASA to escape local minima in state space.

In our method, a state $S \in D^m$ has a low cost if it has a high reachable goal percentage, which we approximate with $\hat{r}_i(S)$ as described in Sec. IV-A. Computing $\hat{r}_i(S)$ requires that we specify $t$, i.e., the number of iterations of RRT to use for the approximation. We cannot know in advance how many iterations of RRT it will take to compute an adequate approximation of a design’s goal reachability, so we set this number of iterations $t$ to an initial value $t_{\text{start}}$ and increase it by $t_{\text{increase}}$ after every design set we consider. This enables us to more quickly (but more coarsely) evaluate many design sets at the beginning, and then we evaluate at a slower rate with higher accuracy as the algorithm progresses. This behavior is analogous to ASA’s decreases in sampling variance and acceptance probability over time.

As mentioned in Sec. IV-A, for efficiency we compute $\hat{r}_i(S)$ by parallelizing the computations of $\text{VoxelsReachable}(d_i)$ for all $d_i \in S$ across multiple processor cores. However, we often have more processor cores than designs in a design set, i.e., $c > m$ for $c$ processor cores and $m$ designs per design set. This leaves $c - m$ cores that are free for additional computation. In order to make use of all our cores, at each iteration of ASA we actually sample a design set $S'$ of size $c$ and evaluate each design’s reachable goal voxels. We then iterate over all $\binom{c}{m}$ subsets of $S'$ of size $m$ to find the set of designs that collectively yield the highest reachable goal percentage $\hat{r}_i(S)$. This subset iteration step is completely dominated in computation time by the evaluation of each design’s reachable goal percentage, so this method effectively enables us to sample design sets of higher quality with no extra computation time due to parallelization.

C. Finding a Design Set of Minimal Size

In Sec. IV-B we described how we find a set of designs of fixed size that collectively maximize the reachable goal percentage. We now focus on finding a design set of minimal cardinality with a reachable goal percentage greater than a user-specified threshold $r_{\text{threshold}}$ (shown in Alg. 3).

We begin by invoking the fixed size algorithm (Alg. 2) with a user-specified maximum set size $m_{\text{max}}$ and terminating after the threshold $r_{\text{threshold}}$ is reached. Once a design set $S'$ of size $m_{\text{max}}$ has been found that can reach a percentage of the goal greater than $r_{\text{threshold}}$, we iterate by invoking Alg. 2 using a design set of size $m_{\text{max}} - 1$. In order to speed up the algorithm, we initialize subsequent iterations of Alg. 2 with the set $S$, the best subset of $S'$ of size $m_{\text{max}} - 1$. We iterate until we run out of the time allotted for the optimization.
A. Maximizing Reachability of a Design Set of Fixed Size

We first show how the reachable goal percentage of a set of designs is affected by the size of the design set. We considered fixed set sizes \(m\) of 1, 2, 4, and 6. For each value of \(m\), we executed Alg. 2 to find a design set of size \(m\) that maximizes the reachable goal percentage. For each trial we recorded how the solutions’ reachable goal percentage progressed over an allotted time of 3 hours, and we averaged the results of 20 trials for each value of \(m\). Results are shown in Fig. 3.

In the time allotted, design sets of larger size were found to reach a larger percentage of the goal region by our design algorithm, with design sets of sizes 1, 2, 4, and 6 being found to reach approximately 70%, 84%, 94%, and 97% of the goal region, respectively. This demonstrates the benefit of considering collections of designs in order to enable a wider variety of possible clinical procedures. Also, the marginal difference in reachable goal percentage between using 4 and...
6 designs highlights the diminishing returns of adding more designs to the set.

B. Minimal Design Set with Sufficient Goal Reachability

We next evaluated the ability of Alg. 3 to generate a robot design set of minimal cardinality that can reach at least \( r_{\text{threshold}} = 95\% \) of the goal region. We used a maximum design set size of \( m_{\text{max}} = 12 \). We executed 20 trials of our algorithm, with 3 hours of computation time per trial.

We show the average minimum set size found by our algorithm over time in Fig. 4. We note that we did not begin averaging results until all trials had found their first design set with a sufficient reachable goal percentage \( r_{\text{threshold}} \), which occurred at 116 minutes. The figure shows that, over time up to 3 hours, our algorithm progressively finds smaller and smaller sets of robot designs that can still reach a sufficient percentage of the goal region.

C. Benchmarking Variations on Algorithm

We next compare our method with different approaches to design optimization. We compare our algorithm for fixed-size design sets against two variants that borrow some elements of the design algorithm presented by Burgner et al. [5], which we note was developed for different anatomical scenarios.

- **NM + G**: We use the Nelder-Mead optimization algorithm instead of ASA for generating new designs to consider. To evaluate the reachable goal percentage of a design, we do not use motion planning; we instead consider a goal voxel reachable if there exists a single collision-free robot configuration where the tip lies inside the goal voxel. We compute the reachable goal voxels of a design by discretizing the robot’s configuration space into a grid and iterating over each point on the grid.

- **NM + MP**: We use the Nelder-Mead optimization algorithm to sample new design sets instead of ASA.

We use motion planning to compute the reachable goal percentage of design sets.

We compare the above approaches against our full method, which we denote as “ASA + MP”. We executed each algorithm on the lung scenario with a fixed design set size of 2. Since the “NP + G” variant does not ensure that goal voxels are reachable by entirely collision-free motions when considering full deployment, we estimated the reachable goal voxels of designs generated by this variant by executing 200,000 RRT iterations on each design returned (and we did not count this verification step in the timing results).

We executed 20 trials of each approach and averaged their reachable goal percentage over time to generate the results in Fig. 5. The results demonstrate that (1) using motion planning as part of the optimization to determine a set of designs’ reachable goal percentage and (2) using a global optimization method like ASA enable us to more quickly compute sets of concentric tube robot designs that can reach larger portions of the goal region without colliding with anatomical obstacles during deployment.

VI. CONCLUSIONS

We presented algorithms for computing sets of concentric tube robot designs that can reach physician-specified goal regions while avoiding contact with anatomical obstacles. We focused on (1) finding a set of designs of a fixed size that maximizes the percentage of a goal region that can be reached and (2) minimizing the size of a set of designs that reaches a desired percentage of a goal region. Our approach interleaves a global stochastic search over the space of robot design sets with a sampling-based motion planner in the robot’s configuration space.

In future work we plan to improve our algorithms to bring them closer to clinical applicability. Thus far we have focused on static environments, and we plan to extend...
our approach to consider tissue deformations and dynamic obstacles. We also plan to consider new optimization metrics, including metrics that consider tissue damage and uncertainty. Furthermore, we plan to evaluate the effectiveness of our design algorithms in physical experiments using tissue phantoms and to assess the benefits of design optimization in a variety of medical scenarios.

ACKNOWLEDGMENT

The authors thank C. Bowen for his insights regarding the visualization component of the design optimization software and thank collaborators in the research group of R. J. Webster III for general discussions about the problem.

REFERENCES


