applied sciences

MDPI

Article

3D Optimal Control Using an Intraoperative Motion Planner for a Curvature-Controllable Steerable Bevel-Tip Needle

Binxiang Xu [1], Seong Young Ko [2,*] and Chen Zhou [3,*]

[1] College of Mechanical and Electrical Engineering, Shaoxing University, Shaoxing 312000, China; bxu@usx.edu.cn
[2] Department of Mechanical Engineering, Chonnam National University, Gwangju 61186, Republic of Korea
[3] College of Mechanical and Electrical Engineering, Wenzhou University, Wenzhou 325035, China
* Correspondence: sko@jnu.ac.kr (S.Y.K.); chzhou@wzu.edu.cn (C.Z.)

Abstract: Robotic needle steering has become a topic of interest in intervention surgery. Yet, this surgical procedure poses challenges due to external disturbances and tissue movement. To address these challenges, several novel steering algorithms have been developed to guide the needle precisely from the entry point to the target point. However, some of these algorithms may cause additional trauma to patients. In this paper, we present a 3D optimal control algorithm for a curvature-controllable steerable (CCS) needle, aiming to achieve effective operations with minimal trauma. We derive a kinematics without duty cycle control strategy (needle shaft spin), propose a novel intraoperative motion planner for path replanning, and design a full-state feedback controller for accurate path tracking. A dynamic environment was simulated, and the optimal controller showed a better result (0.01 ± 0.01 mm) than the case (3.86 ± 1.32 mm) using a full-state feedback controller. The demonstration indicates that the optimal control system can safely, effectively, and accurately steer the needle to the target point in a dynamic environment.

Keywords: 3D optimal control; intraoperative motion planner; trajectory following; needle steering

check for updates

Citation: Xu, B.; Ko, S.Y.; Zhou, C. 3D Optimal Control Using an Intraoperative Motion Planner for a Curvature-Controllable Steerable Bevel-Tip Needle. *Appl. Sci.* **2024**, *14*, 8917. https://doi.org/10.3390/app14198917

Academic Editor: George Mylonas

Received: 27 August 2024
Revised: 19 September 2024
Accepted: 28 September 2024
Published: 3 October 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Steerable needles represent a new technology in intervention surgery; they can follow a curvilinear trajectory to safely avoid critical organs, while reaching a lesion inside a patient's body [1]. Steerable needles have been widely used for drug delivery, biopsy, and brachytherapy [2–4]. Thus far, needle-steering systems have encountered several challenges, such as the complex anatomical structure, disturbances resulting from tissue deformation and the movement of important organs, and trauma caused by the steering methodology.

According to the statistics reported by Zhang et al., needle-steering techniques can be classified into six categories: needle tip actuation by applying an external moment at the needle base [5], a bevel-tip needle [6] that is deflected by tissue reaction forces, a steerable needle composed of concentric tubes [7], a needle with a tendon-driven tip [8], and a magnetic needle controlled using an electromagnetic system [9]. However, the above steering techniques have issues, such as insufficient driving force, tissue damage, and large dimensions. To overcome the aforementioned shortcomings, we proposed a curvature-controllable steerable (CCS) bevel-tip needle in a previous work [10]. The curvature of the CCS needle was mainly controlled by means of stylet translation, as illustrated in Figure 1, and the trajectory-following capability of the CCS needle was preliminarily tested in [11]. Yet, the steering system has to partially use the duty cycle algorithm [12] (a control strategy which periodically spins the needle shaft to adjust the trajectory curvature for a conventional bevel-tip needle), such motion causes additional damage to surrounding tissues. In addition, in the trajectory-following test, the movement of important organs and the target point were not considered.

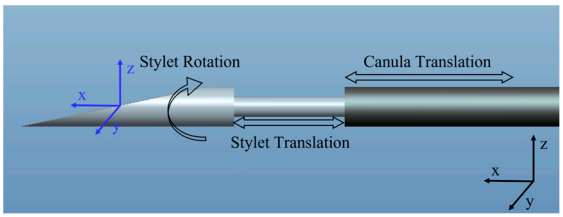


Figure 1. Schematic design of the CCS bevel-tip needle, with a tip coordinate system (in blue) and a world coordinate system (in black).

Motion planning involves finding feasible trajectories from the insertion point to the target point, while avoiding any collisions with obstacles [13]. The motion-planning task has been investigated extensively by many researchers and it can be a good solution to the existing issues affecting CCS needle-steering systems. Bano et al. [14] represented the trajectory of a needle by using a curvature polynomial and identified its coefficients by using a gradient-based optimization method. In [15], an evaluation was conducted in a gelatin phantom using a programmable needle-steering system, the result showed this algorithm is only suitable for offline path planning. Pinzi et al. [16] integrated the adaptive fractal tree algorithm with optimized geometric Hermite curves, and it was found that the resulting algorithm could plan trajectories under both curvature and heading constraints. However, the above algorithms did not consider the dynamic nature of the operating environment, and they can be used only for preoperative path planning. For online path replanning, Likhachev et al. [17] proposed an algorithm based on A* that could use information from a previous path to perform iterative path optimization. Niyaz et al. [18] developed a lifelong planning A* algorithm for replanning the path of a concentric tube robot; this algorithm improved working efficiency by avoiding the recalculation of the entire graph. The abovementioned graph-based methods are relatively simple, but they require finer environment discretization in three-dimensional (3D) space, which increases the computation time. Alterovitz et al. [19] formulated the planning problem as a Markov decision process based on an efficient discretization of the state space, with optimal steering actions computed using dynamic programming. Though motion uncertainty was considered, the study focused only on planar needle steering.

Recently, neural networks have been used as a planning method for robotic systems [20]. In the context of steerable needle systems, Segato et al. [21] implemented the GA3C algorithm for building steerable needle path planning models. Compared to other methods, this neural network-based path planning model can be generalized easily for different anatomies. In [22], this model was extended using an inverse reinforcement learning-based approach to ensure that it could react to dynamic environments. Although the effectiveness of the underlying neural network was demonstrated, the accuracy of this model relied heavily on the training data, and normally these data are not adequately accurate for use in clinical settings.

In terms of planners based on random sampling, Hong et al. [23] used a rapidly exploring random tree star (RRT*) algorithm for 3D path planning. This algorithm considered both the kinematic constraints of a magnetic steerable needle and the anatomical constraints of patients. Zhao et al. [24] proposed a GHRG-RRTs algorithm, consisting of a greedy heuristic strategy and a reachability-guided strategy, to be applied to an active flexible needle. Favaro et al. [25] used the RRT algorithm for path planning as well, where they redefined the random searching space by using kinematics and employed an evolutionary optimization procedure for trajectory optimization. However, these three algorithms did not consider the dynamism of the environment. For intraoperative path replanning, three types of replanning strategies were introduced in [6], but the simulation results were not promising. Owing to the dynamic nature of the operating environment, an online path was deformed by an extended bubble bending method in [26], and the deformed path was smoothed using a convex optimization method. This method required discretization

of the path into a succession of 3D points, but as the number of bubbles increased, the computation time increased.

When the information on a planned trajectory is available, a real-time feedback controller is required to track the path accurately. Jeong et al. [27] applied the classic proportional–integral–derivative (PID) control algorithm to steer a magnetic guidewire, but with a PID controller, the tuning control gain is problematic. Patil et al. [28] presented a linear quadratic Gaussian (LQG) controller for bevel-tip needle control in 3D space, but its dynamic response to external disturbances was not appropriate. Secoli et al. [29] developed a nonlinear dynamic feedback controller to track the 3D path of a programmable needle. Hong et al. [30] proposed a nonlinear error equation with the proper control gains; the time derivative of the Lyapunov-like function proved to be negative semi-definite. Donder et al. [31] adopted a nonlinear guidance law as the high-level controller for tracking the path of a steerable needle and proposed an active disturbance rejection control scheme to manage needle–tissue interactions. The designs of the aforementioned nonlinear systems are rather complex. Ko et al. [32] implemented a full-state feedback controller with an approximately linearized control system, but they focused only on following planar trajectories.

In the present work, an optimal control system consisting of an intraoperative motion planner and a feedback controller is proposed for steering a CCS needle in 3D space. The contributions of this work are as follows: the curvature of the planned trajectory is within a specific range, where the duty cycle control strategy proposed in [11] can be avoided, and the safety of intervention surgery can be improved. Compared to the conventional RRT algorithm, the proposed planning algorithm has fewer nodes and a lower cost. In physical terms, it requires a smaller rational angle. Rather than devising a completely new trajectory, the intraoperative motion planner uses the information on the best trajectory from the last round and optimizes it iteratively. The high similarity between the existing and iteratively optimized trajectories can reduce the computational cost of the algorithm. Furthermore, instead of compensating for the position error by using the feedback controller alone, the position error can be reduced by replanning a trajectory that reduces robot manipulation.

The remainder of this paper is organized as follows: Section 2 describes the kinematics modeling of a CCS needle, without using the duty cycle control strategy. Section 3 presents the algorithms for trajectory planning in the static and dynamic environments, as well as a cost function for path optimization. The presentation of the chained form of the kinematics and feedforward and full-state feedback controller designs are introduced in Section 4. Section 5 describes the simulation environment, a comparison of the motion planner performance, the gain tuning procedure, the optimal controller performance evaluation, and presents a discussion of our results. In Section 6, we summarize this study and present an outline for future work.

2. Kinematics Analysis

To improve the safety of intervention surgery, the duty cycle control strategy proposed in a previous work [11] to adjust the trajectory curvature is not used in the present study. As depicted in Figure 1, the trajectory curvature is controlled by means of stylet translation alone, the insertion orientation is controlled by means of stylet rotation, and the insertion depth is controlled by means of cannulation translation.

Under the follow-the-leader assumption, which means that the needle shaft follows the trajectory created by the tip, the 3D kinematics of the CCS needle is formulated, as illustrated in Figure 2. The world frame is marked in black color at the insertion point, and the black line indicates the needle's trajectory in 3D space. Euler angle representation is adopted to describe the orientation of the tip frame T with respect to the world frame W, and a rotation matrix is used to express the tip rotations about the z and y axes of the world frame, as follows:

$$^{W}_{T}R = R_z(\gamma)R_y(-\beta), \quad (1)$$

where γ and β are rotation angles. Define v_x as the insertion velocity at the needle base and the linear velocity vector of the kinematics is $v_l = \begin{bmatrix} v_x & 0 & 0 \end{bmatrix}$. The linear velocity of the needle tip $\dot{p}$ can be computed as:

$$\dot{p} = {}^{W}_{T}R v_l. \tag{2}$$

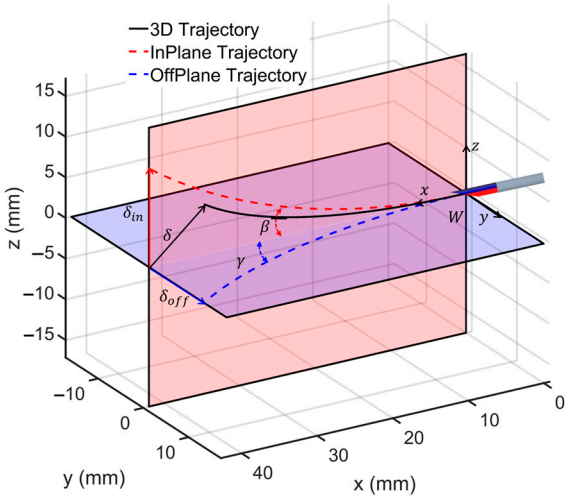


Figure 2. Kinematic model of a CCS needle; δ_{in} and δ_{off} are the projection of the steering offset δ in the *InPlane* and *OffPlane* region, respectively.

As illustrated in Figure 2, the xz plane marked with the transparent red color is defined as the *InPlane*, and the xy plane marked with the transparent blue color is defined as the *OffPlane*. The dashed red and blue lines represent projections of the 3D trajectory with respect to the *InPlane* and *OffPlane*, respectively. The assumption that the instantaneous curvature ρ is proportional to the steering offset δ is adapted from our planar kinematics findings [10]. Therefore, the instantaneous curvatures of the *InPlane* and *OffPlane* trajectories are as follows:

$$\begin{aligned} \rho_{in} &= k\delta_{in} \\ \rho_{off} &= k\delta_{off}, \end{aligned} \tag{3}$$

where δ_{in}, δ_{off}, and k denote the *InPlane* steering offset, the *OffPlane* steering offset, and the coefficient related to tissue property, which should be calibrated experimentally. The angular velocity vector of the kinematics is $\omega = \begin{bmatrix} 0 & \omega_y & \omega_z \end{bmatrix}$, where ω_y and ω_z are:

$$\begin{aligned} \omega_y &= \dot{\delta}_{in} \\ \omega_z &= \dot{\delta}_{off}. \end{aligned} \tag{4}$$

Therefore, the changing rate of the rotation angles can be expressed as follows:

$$\begin{aligned} \dot{\beta} &= \omega_y s = k\delta_{in} v_x \\ \dot{\gamma} &= \omega_z s = k\delta_{off} v_x, \end{aligned} \tag{5}$$

where s denotes the insertion depth.

By combining Equations (2), (4), and (5), the kinematics of the CCS needle can be expressed as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\beta} \\ \dot{\gamma} \\ \dot{\delta}_{in} \\ \dot{\delta}_{off} \end{bmatrix} = \begin{bmatrix} cos\gamma cos\beta \\ sin\gamma cos\beta \\ sin\beta \\ k\delta_{in} \\ k\delta_{off} \\ 0 \\ 0 \end{bmatrix} v_x + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \omega_y + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \omega_z, \tag{6}$$

where the stylet rotation (as shown in Figure 1) angle of the CCS needle θ and composite steering offset δ (stylet translation) can be computed from the kinematics as:

$$\begin{aligned} \theta &= atan2\left(-\delta_{off}, \delta_{in}\right) \\ \delta &= \sqrt{{\delta_{in}}^2 + {\delta_{off}}^2}. \end{aligned} \tag{7}$$

The 3D kinematics of the CCS needle expressed in (6) is nonlinear, and its general form is as follows:

$$\dot{q} = G(q)v \tag{8}$$

where q is the vector of the CCS needle's generalized coordinates $\begin{bmatrix} x & y & z & \beta & \gamma & \delta_{in} & \delta_{off} \end{bmatrix}^T$, v is the vector of input velocities $\begin{bmatrix} v_x & \omega_y & \omega_z \end{bmatrix}^T$, and the columns $g_i (i = 1,2,3)$ of the matrix $G(q)$ denote vector fields [33].

3. The 3D Motion-Planning Algorithm

Motion planning is one of the topics related to needle steering, and it entails finding feasible trajectories from an entry point to a target point, while avoiding important organs. The motion planner proposed herein can not only find a feasible trajectory, but can also reduce tissue damage and robotic manipulation. A motion planning algorithm adapted from the goal-biased RRT* algorithm is proposed herein.

3.1. Kinematic Constrains and Cost Function

The tip of a CCS needle is asymmetrical and, therefore, its insertion trajectory consists of several circular segments, as illustrated in Figure 3a. CCS needle robots are a nonholonomic system. Therefore, the needle insertion trajectory should be continuous and smooth, and adjacent segments should have a common tangential direction at their intersection point. In addition, the curvature of each circular segment depends on the control strategy and mechanical properties of the tissue. Given that use of the duty cycle control strategy is prohibited, linear segments cannot be generated. The minimal and maximal curvatures of the resulting circular segments are $\rho_{min} = k\delta_{min} > 0$ and $\rho_{max} = k\delta_{max}$, respectively. Where δ_{min} and δ_{max} denote the minimal and maximal stylet translations (steering offset), respectively.

The objective of the motion planner is to generate a feasible and optimal trajectory from among several options. The goodness of the generated trajectory can be evaluated from the perspectives of safety and trauma reduction, among other factors. Herein, each segment S generated by the motion planner is evaluated in terms of its cost function, as follows:

$$c(S) = c_1 f_L + c_2 f_R + c_3 f_C, \tag{9}$$

where f_L, f_R, and f_C denotes the relative segment length, needle rotation angle, and clearance of the obstacle, respectively. Moreover, $c_i (i = 1,2,3)$ denote the weights of the three aforementioned optimization criteria, and their sum is equal to one.

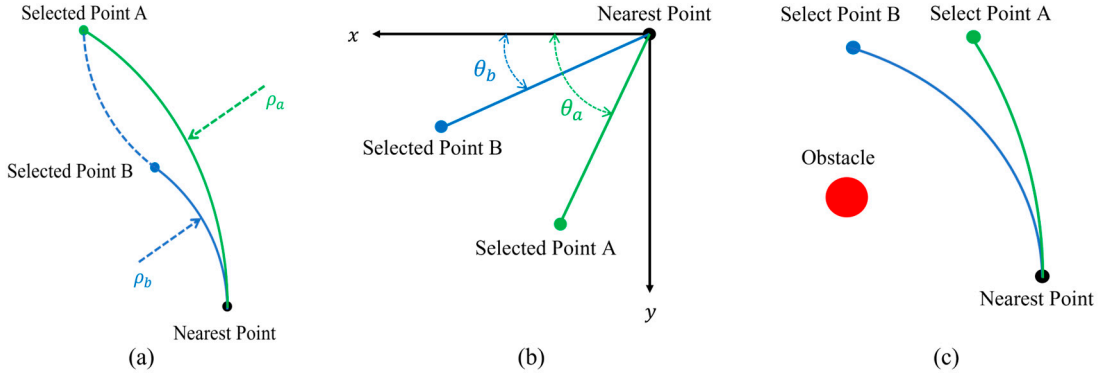


Figure 3. (**a**) The insertion trajectories displayed in the front view; (**b**) the rotation angles of the green segment and the blue segment, respectively; (**c**) the proximity of the obstacle to the blue segment and the green segment, respectively.

A short trajectory length can reduce the trauma caused to patients. Therefore, the first optimization term is the segment length. In Figure 3a, the absolute length of the blue segment is shorter than that of the green segment, and the curvature of the blue segment ρ_b is larger than that of the green segment ρ_a. For each trajectory, the insertion depth from the start point to the target point is fixed. The blue segment requires another dashed segment to reach the same insertion depth as the green segment, and their total absolute length is longer than that of the green segment. Therefore, the relative segment length f_L is adopted instead of the absolute segment length, as follows:

$$f_L = \frac{\rho_i}{\rho_{min}}, \tag{10}$$

where ρ_i is the curvature of segment i. The second optimization term is the rotation angle of the needle shaft:

$$f_R = \theta_i, \tag{11}$$

The needle needs to rotate from the end point of the last segment to the start point of the new segment. As illustrated in Figure 3b, the blue segment is better than the green segment because $\theta_b < \theta_a$, which means that the damage cause by the needle rotation in the blue segment is less than that in the green segment. The third optimization term is related to segment safety. As illustrated in Figure 3c, the green segment is superior to the blue segment because the blue segment is considerably closer to the obstacle than the green segment, and this proximity is defined as follows:

$$f_c = \begin{cases} 2^{\frac{C_d}{d_{ij}}} & d_{ij} < 10 \\ 0 & d_{ij} \geq 10 \end{cases} \tag{12}$$

where C_d is a constant, and d_{ij} is the distance between the i_{th} point of segment j to the center of an obstacle.

3.2. The 3D Intraoperative Motion Planner

The computational cost of a 3D motion planner is substantially higher than that of a planar motion planner. To design a 3D intraoperative motion planner, frequency is among the key parameters that should be considered. We propose a modified RRT*-based motion planner, owing to its high trajectory planning efficiency. The superiority of the modified motion planner will be proved by comparing it with other motion planners. A static motion-planning algorithm is used to generate multiple candidates, and it selects the best trajectory as one of the inputs to the intraoperative motion planner. The outline of the static motion-planning algorithm is shown in Algorithm 1.

The three inputs into the static motion-planning algorithm are the position and tangential orientation of the insertion point q_{ins}, the target point q_{tar}, and set of obstacles Q_{obs}. The output of the algorithm is the best trajectory evaluated among several options, by using the cost function expressed in Equation (9).

The static motion planner is revised from a goal-biased RRT* algorithm [34]. First, a tree t_i is initialized and the environmental map E is constructed. Then, a point q_{samp} is sampled using the *RandomSample* function to emphasize the purpose of target searching. This sampling point is either the target point or a random point. Later, the nearest point q_{near} in the tree t_i is searched for and the feasibility of the segment starting from the nearest point to the sampling point is checked. The *ConstrainsFree* function checks for this feasibility in terms of the kinematics constraints (bounded curvature, insertion angle, and so forth) and environmental constraints (obstacle avoidance). If the constraint conditions are not satisfied, the algorithm repeats the procedures mentioned in lines 3~11. Otherwise, it calculates the kinematic input u.

Algorithm 1: Static Motion Planning Algorithm
$Input: \{q_{ins}, q_{tar}, Q_{obs}\}$ $Output: \{t_{best}\}$

1: $Q_{envir} = ObsAndTarInit(q_{ins}, q_{tar}, Q_{obs})$;
2: $for(i = 1;\ i < ValidTreeNum;\ i++)$
3: $t_i = InitializeTree()$;
4: $E = EnvionmentConstruion(Q_{envir})$;
5: $for(j = 1;\ j \le maxIteration;\ j++)$
6: $if\left(\left\|q_{new} - q_{goal}\right\| < d\right)$
7: $T_{valid}.AddTree(t_i)$;
8: $break$;
9: end
10: $q_{samp} = RandomSample(P_{threshold})$;
11: $q_{near} = NearestNode(q_{samp}, t_i)$;
12: $if\left(ConstrainsFree(q_{near},\ q_{samp}, E) == true\right)$
13: $u = InputComputation(q_{near},\ q_{samp})$;
14: $else$
15: $continue$;
16: end
17: $q_{new} = VarSegLen(q_{near},\ q_{samp}, u,\ s, direct)$;
18: $Q_{neighbor} = FindNeighbor(q_{new},\ t_i,\ r)$;
19: $t_i = OptNewNodePar\left(q_{new},\ Q_{neighbor}, t_i\right)$;
20: $t_i = OptNeighborPar\left(q_{new},\ Q_{neighbor},\ t_i\right)$;
21: $t_i.AddNode(q_{new})$;
22: $t_i.AddSegment(q_{near}, q_{new}, u)$;
23: end
24: $t_i.CostComputation()$;
25: end
26: $t_{best} = SelectBestTrajectory(T_{valid})$;
27: $return\ t_{best}$

Function 1: $RandomSample()$
$Input: \{P_{threshold}\}$ $Output: \{q_{samp}\}$

1: $p = rand(0 \sim 1)$;
2: $if(p < P_{threshold})$
3: $q_{samp} = q_{goal}$;
4: $direct = true$;
5: $else$
6: $q_{samp} = RandomNode()$;
7: $direct = false$;
8: end
9: $return\ q_{samp}$

Function 2: $VarSegLen()$
$Input: \{q_{samp}, q_{near}, u, s, direct\}$ $Output: \{q_{new}\}$

```
if (direct == true)
    q_new = q_samp;
else
    q_new = NewNode(q_near, u, s);
end
return q_new
```

Unlike the conventional RRT algorithm that truncates the feasible segment with a constant arc length s, the proposed algorithm uses the *VarSegLen* function to identify the new point q_{new} that will be added to the tree. As expressed in the *VarSegLen* function, if the Boolean value *direct* is false, the new point is the end point of the segment that is truncated by the *NewNode* function with a predefined arc length s, as illustrated in Figure 4a. Otherwise, the new point is the sampling point. In this algorithm, the Boolean value is true only when the sampling point is the target point. Therefore, the *VarSegLen* function can directly generate the segment from the nearest point to the target point. Given that the tree requires fewer sampling points (nodes) and segments, this function reduces the needle shaft rotation, thereby improving the steering safety. The superiority of this function is demonstrated in the experimental section. The remaining functions *OptNewNodePar* and *OptNeighborPar* are employed to optimize the parent point of the new point by using the neighboring points and to optimize the parent points of the neighboring points by using the new point, respectively. The last step entails selecting the best candidate among the set of valid trajectories T_{valid}.

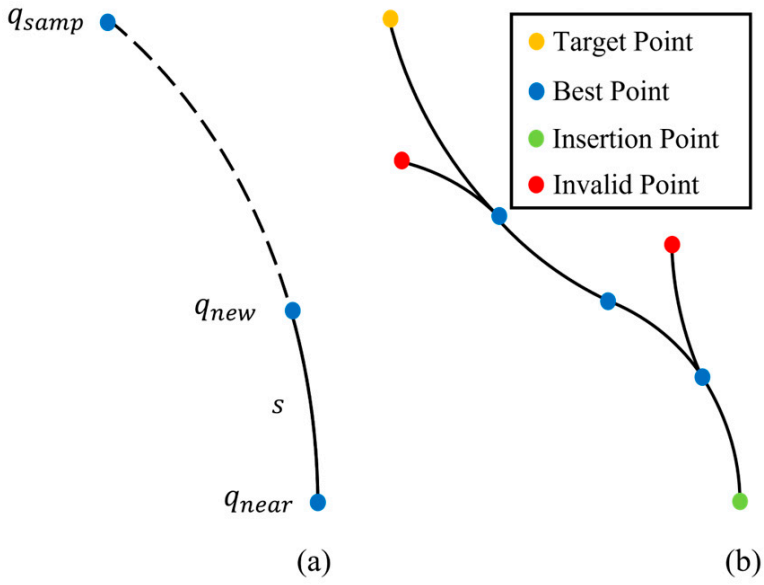


Figure 4. (**a**) A segment connecting the nearest point with the sampling point is truncated with an arc length s; (**b**) a trajectory consisting of several best point.

The output is the best trajectory t_{best}, and it is used as the input $t_{prebest}$ in the next invocation of the intraoperative motion-planning algorithm, as shown in Algorithm 2. To reduce the trajectory search time, while using the information generated in the previous invocation, the *BestNode* function is used to select the set of best nodes $Q_{bestnodes}$ (except the insertion point, target points, or invalid points, as depicted in Figure 4b in the best trajectory. Because physiological movements of organs are considered in this algorithm, the target point and obstacles should be updated to construct the environment E in each iteration. The *ValidTreeNum* is set to three, and the *DirectTrajctory* function first checks whether a direct

trajectory from the insertion point q_{ins} to the target point q_{tar} exists. If the trajectory housing the best nodes is not found and the tree index number i is smaller than the threshold number *thresnum*, the program executes the *BestSelection* function. To ensure that the tree grows toward the target, the *AvailableNodes* function selects the nodes $Q_{AvailNodes}$ whose depth (z-axis) is greater than that of the latest point in the tree t_i. If there is a feasible node within the set $Q_{AvailNodes}$, the sampling point q_{samp} is q_{best}. Otherwise, the sampling point is sampled randomly.

Therefore, the second valid tree denotes the trajectory that uses the information obtained from $t_{prebest}$. The remaining codes, identical to those in the static motion planning algorithm, and the last step of the algorithm, involves finding the best trajectory t_{best} among the above three trajectories.

Algorithm 2: Intraoperative Motion Planning Algorithm

$Input: \left\{q_{ins}, q_{tar}, Q_{obs}, t_{prebest}\right\}$ $Output: \{t_{best}\}$

$Q_{bestnodes} = BestNode\left(t_{prebest}\right)$
$for(i = 1;\ i < ValidTreeNum;\ i++)$
  $t_i = InitializeTree();$
  $Q_{environ} = ObsAndTarUpdate(q_{ins}, q_{tar}, Q_{obs});$
  $E = EnvionmentConstruion(Q_{environ});$
  $if(i == 1)$
    $DirectTrajectory(t_i);$
    $continue;$
  end
  $for(j = 1;\ j \le maxIteration;\ j++)$
    $if\left(\left\|q_{new} - q_{goal}\right\| < d\right)$
      $T_{valid}.AddTree(t_i);$
      $break;$
    end
    $if(i < thresnum\ \ \&\&\ bestexist == false)$
      $q_{samp} = BestSelection(Q_{bestnodes}, t_i, E);$
    $else$
      $q_{samp} = RandomSample(P_{threshold});$
    end
    $q_{near} = NearestNode(q_{samp}, t_i);$
    $if(ConstrainsFree(q_{near},\ q_{samp}, E) == true)$
      $u = InputComputation(q_{near},\ q_{samp});$
    $else$
      $continue;$
    end
    $q_{new} = VarSegLen(q_{near},\ q_{samp}, u,\ s, direct);$
    $Q_{neighbor} = FindNeighbor(q_{new},\ t_i,\ r);$
    $t_i = OptNewNodePar\left(q_{new},\ Q_{neighbor}, t_i\right);$
    $t_i = OptNeighborPar\left(q_{new},\ Q_{neighbor},\ t_i\right);$
    $t_i.AddNode(q_{new});$
    $t_i.AddSegment(q_{near}, q_{new}, u);$
  end
  $t_i.CostComputation();$
end
$t_{best} = SelectBestTrajectory(T_{valid})$
$return\ t_{best}$

Function 3: $BestSelection()$
$Input: \{Q_{bestnodes}, t_i, E\}$ $Output: \{q_{samp}\}$

$Q_{AvaiNodes} = AvailableNodes(t_i, Q_{BestNodes})$
$bestexist = false;$
$for(i = 1; i < size(Q_{AvaiNodes}); i++)$
  $q_{best} = Q_{AvaiNodes}(i)$
  $q_{near} = NearestNode(q_{best}, t_i);$
  $if(ConstrainsFree(q_{near}, q_{best}, E) == true)$
    $q_{samp} = q_{best};$
    $direct = true;$
    $bestexist = true;$
    $break;$
  end
end
$if(bestexist == false)$
  $RandomSample(P_{threshold});$
end
$return\ q_{samp}$

4. Feedback Controller Design

During update time, a feedback controller can accurately track the proximal segment of the best trajectory provided by the motion planner and feed the end point of the proximal segment to the motion planner as the insertion point for the next update round. The above procedures are repeated until the needle reaches the target. A feedback controller is required to complete this task.

4.1. Chained Form Presentation

The kinematics in Equation (6) indicates that the steering system is nonholonomic. In [32], a method was introduced to convert a nonholonomic system into a chained form system, for which the design of the feedforward and feedback controllers are straightforward. The kinematics can be transformed into the two chained forms with three new inputs (u_1, u_2, u_3) and seven states $(\xi_1, \xi_2, \xi_3, \xi_4, \xi_5, \xi_6, \xi_7)$, as follows:

$$\begin{aligned} \dot{\xi}_1 = u_1 \quad \dot{\xi}_2 = u_2 \qquad \dot{\xi}_5 = u_3 \\ \dot{\xi}_3 = \xi_2 u_1 \qquad \dot{\xi}_6 = \xi_5 u_1 \\ \dot{\xi}_4 = \xi_3 u_1 \qquad \dot{\xi}_7 = \xi_6 u_1. \end{aligned} \tag{13}$$

By choosing $\xi_1 = x$, $\xi_4 = y$, and $\xi_7 = z$, the kinematics can be expressed as follows:

$$\xi = \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \\ \xi_4 \\ \xi_5 \\ \xi_6 \\ \xi_7 \end{bmatrix} = M(q) = \begin{bmatrix} x \\ k\delta_{off} sec^3\gamma sec\beta \\ tan\gamma \\ y \\ k\delta_{off} sec^2\gamma tan\gamma sec\beta tan\beta + k\delta_{in} sec^2\gamma sec^3\beta \\ sec\gamma tan\beta \\ z \end{bmatrix}, \tag{14}$$

where M is a function that converts the original state q into the chained form state ξ. Once the desired trajectory is given, the control input u can be computed using the chained form state. The original input v can be computed as follows:

$$
\begin{aligned}
v &= \begin{bmatrix} N_{11} & N_{12} & N_{13} \\ N_{21} & N_{22} & N_{23} \\ N_{31} & N_{32} & N_{33} \end{bmatrix} u \\
N_{11} &= sec\gamma sec\beta \\
N_{12} &= 0 \\
N_{13} &= 0 \\
N_{21} &= -3k{\delta_{off}}^2 sec\gamma tan\gamma sec\beta - k\delta_{off}\delta_{in} sec\gamma sec\beta tan\beta \\
N_{22} &= \tfrac{1}{k} cos^3\gamma cos\beta \\
N_{23} &= 0 \\
N_{31} &= k{\delta_{off}}^2 sec\gamma tan^2\gamma sin\beta - k{\delta_{off}}^2 sec^3\gamma sin\beta \\
&\quad -3k\delta_{off}\delta_{in} sec\gamma tan\gamma sec\beta - 3k{\delta_{in}}^2 sec\gamma sec\beta tan\beta \\
N_{32} &= -\tfrac{1}{k} cos^2\gamma sin\gamma cos^2\beta sin\beta \\
N_{33} &= \tfrac{1}{k} cos^2\gamma cos^3\beta.
\end{aligned} \tag{15}
$$

4.2. Controller Design

The position, velocity, and acceleration information of the desired trajectory, generated by the intraoperative motion planner, are available. The states of the CCS needle can be computed using $L(x_d, y_d, z_d)$, as follows:

$$
\begin{aligned}
\xi_{d1} &= x_d \\
\xi_{d2} &= \left(\dot{x}_d\ddot{y}_d - \ddot{x}_d\dot{y}_d\right) / \dddot{x}_d^3 \\
\xi_{d3} &= \dot{y}_d / \dot{x}_d \\
\xi_{d4} &= y_d \\
\xi_{d5} &= \left(\dot{x}_d\ddot{z}_d - \ddot{x}_d\dot{z}_d\right) / \dddot{x}_d^3 \\
\xi_{d6} &= \dot{z}_d / \dot{x}_d.
\end{aligned} \tag{16}
$$

The desired input u_d into the system can be derived as:

$$
\begin{aligned}
u_{d1} &= \dot{\xi}_{d1} = \dot{x}_d \\
u_{d2} &= \dot{\xi}_{d2} = \frac{\dot{x}_d^2\dddot{y}_d - \dot{x}_d\ddddot{x}_d\dot{y}_d - 3\dot{x}_d\ddot{x}_d\ddot{y}_d + 3\dot{x}_d^2\dot{y}_d}{\dot{x}_d^4} \\
u_{d3} &= \dot{\xi}_{d3} = \frac{\dot{x}_d^2\dddot{z}_d - \dot{x}_d\ddddot{x}_d\dot{z}_d - 3\dot{x}_d\ddot{x}_d\ddot{z}_d + 3\dot{x}_d^2\dot{z}_d}{\dot{x}_d^4}.
\end{aligned} \tag{17}
$$

However, the needle will deviate from the desired trajectory owing to several factors, such as the external disturbance and tissue inhomogeneity. The system errors can be defined as follows:

$$
\begin{aligned}
\dot{e}_1 &= \dot{\xi}_{d1} - \dot{\xi}_1 = u_{d1} - u_1 = e_{u1} \\
\dot{e}_2 &= \dot{\xi}_{d2} - \dot{\xi}_2 = u_{d2} - u_2 = e_{u2} \\
\dot{e}_3 &= \xi_{d2}u_{d1} - \xi_2 u_1 = e_2 u_{d1} + \xi_2 e_{u1} \\
\dot{e}_4 &= \xi_{d3}u_{d1} - \xi_3 u_1 = e_3 u_{d1} + \xi_3 e_{u1} \\
\dot{e}_5 &= \dot{\xi}_{d5} - \dot{\xi}_5 = u_{d3} - u_3 = e_{u3} \\
\dot{e}_6 &= \xi_{d5}u_{d1} - \xi_5 u_1 = e_5 u_{d1} + \xi_5 e_{u1} \\
\dot{e}_7 &= \xi_{d6}u_{d1} - \xi_6 u_1 = e_6 u_{d1} + \xi_6 e_{u1},
\end{aligned} \tag{18}
$$

where u_i and e_{ui} denote the input and compensated input, respectively. The linearization method mentioned in [32] is adopted herein by setting $\xi_{d2} = \xi_2$, $\xi_{d3} = \xi_3$, $\xi_{d5} = \xi_5$, and

$\xi_{d6} = \xi_6$. The abovementioned equation can be presented as the following state space equation:

$$\begin{aligned} \dot{e} &= Ae + Be_u \\ A &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & u_{d1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & u_{d1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & u_{d1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & u_{d1} & 0 \end{bmatrix} \\ e &= \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{bmatrix}, \; B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \xi_{d2} & 0 & 0 \\ \xi_{d3} & 0 & 0 \\ 0 & 0 & 1 \\ \xi_{d5} & 0 & 0 \\ \xi_{d6} & 0 & 0 \end{bmatrix}, \; e = \begin{bmatrix} e_{u1} \\ e_{u2} \\ e_{u3} \end{bmatrix}. \end{aligned} \tag{19}$$

To compensate for the system error, the full-state feedback control algorithm is employed, and its input is set as follows:

$$\begin{aligned} e_{u1} &= k_1 e_1 \\ e_{u2} &= k_2 e_2 + \tfrac{k_3}{u_{d1}} e_3 + \tfrac{k_4}{u_{d1}^2} e_4 \\ e_{u3} &= k_5 e_5 + \tfrac{k_6}{u_{d1}} e_6 + \tfrac{k_7}{u_{d1}^2} e_7. \end{aligned} \tag{20}$$

Upon substituting Equation (20) into (19), the system matrix changes from A to A_{fs}, and the eigenvalue of the feedback system is:

$$\det\left(\lambda I - A_{fs}\right) = (\lambda + k_1)\left(\lambda^3 + k_2\lambda^2 + k_3\lambda + k_7\right)\left(\lambda^3 + k_2\lambda^2 + k_3\lambda + k_7\right). \tag{21}$$

These eigenvalues can be manipulated easily by tuning the control gains $[k_1, k_2, k_3, k_4, k_5, k_6, k_7]$. The input into the chained system is:

$$u = u_d + e_u. \tag{22}$$

A block diagram of the optimal control system for steering a CCS needle is depicted in Figure 5. The overall system consists of a motion planning algorithm, a high-level controller, and a low-level controller.

In the preoperative stage, the static motion planner selects the best trajectory as the input to the intraoperative motion planner. In the intraoperative stage, the motion planner optimizes the best trajectory based on the updated environmental information, and the optimized best trajectory is fed into the high-level controller and the intraoperative motion planner. Once the reference trajectory and the measured states are available, the desired input is computed using Equation (17), and the compensated input is computed using Equation (20). Equation (15) is used to compute the original input to the kinematics. The *Joint Velocity Conversion* function converts the kinematic input into joint input by using Equation (7), and a proportional–derivative (PD) controller is used to guarantee accuracy in the joint space.

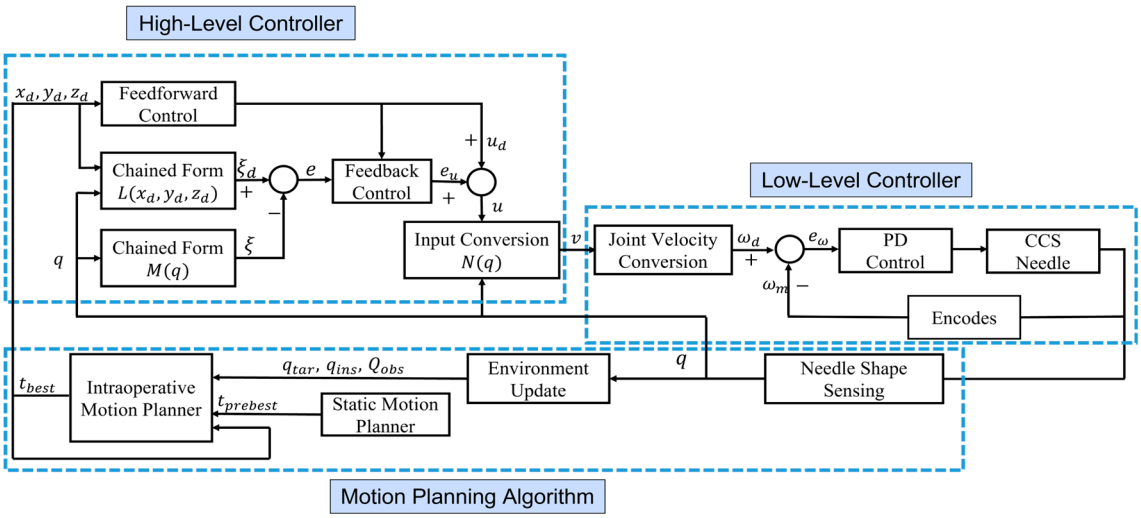


Figure 5. A block diagram of the optimal CCS needle-steering system.

5. Simulation Results

The simulation system used to evaluate the optimal control system was built in the MATLAB software environment, on a PC equipped with a 2.3 GHz 8-core Intel i7 CPU. The working environment was modeled in a workspace measuring $200 \times 200 \times 200$ mm^3. Five spherical obstacles, with diameters of $10 \sim 15$ mm, were set in this environment. The initial target was placed at a depth of 185 mm and the initial insertion point was placed at the origin of the coordinate system. The weight coefficients of the cost function were set to $c_1 = 0.1$, $c_2 = 0.5$, and $c_3 = 0.4$.

5.1. Motion Planner Comparison

To evaluate the performance of the static motion planner proposed herein, we compared the underlying static algorithm to the conventional goal-biased RRT* algorithm, and the difference between these two algorithms was the *VarSegLen* function. In the conventional goal-biased RRT* algorithm, the new point is the end point of a truncated segment, with a predefined arc length. This is identical to the case in which the Boolean value *direct* is false in the proposed static motion-planning algorithm; otherwise, the new point is the target point. For a fair comparison, the parameters of these two algorithms were identical, and they will be optimized later.

The simulation was carried out with the task of generating 100 trajectories and the test was repeated five times to increase the reliability of the comparison results. The results are depicted in Figure 6 and the performance was evaluated in terms of the average node number, the average trajectory costs, and the total time required to complete the task. The green bars denote the results of the conventional goal-biased RRT* algorithm and the yellow bars denote the results of the proposed static motion-planning algorithm. According to these results, the trajectory generated by the static motion-planning algorithm consists of 3.73 nodes, which is almost half the number of nodes generated by the conventional algorithm. The smaller the number of nodes in a trajectory, the smaller the angle the needle requires to rotate. In terms of cost, the comparison results indicate that the algorithm proposed herein can generate a much more optimized trajectory than that generated by the conventional algorithm. The results of the total time required indicate the algorithm proposed herein is computationally efficient. In addition, the static motion-planning algorithm is superior as a sampling-based planning algorithm, and the performance of the intraoperative algorithm obtained by modifying the static algorithm is evaluated later in this work.

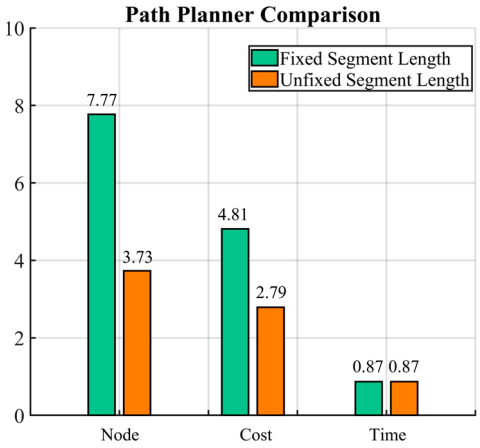


Figure 6. The results of the comparison between the static planning algorithms.

5.2. Parameter Optimization

To enhance the performance of the needle-steering system, the parameters of the optimal controller have to be optimized. The optimal controller consists of a motion planner and a feedback controller, both of which should be tuned; *maxIteration* is a parameter of the motion planner. An optimal trajectory may not be found with a small *maxIteration* value, it may also waste time if the feasible trajectory does not exist when this value is too big. A test was performed to identify 100 trajectories and optimize the maxIteration value based on the results, which are presented in Figure 7. The success rate refers to the probability of finding an optimal trajectory, the cost time refers to the time required to find 100 trajectories (including available and unavailable trajectories), and the mean iterations refers to the average number of iterations required to find an optimal trajectory. Herein, the *maxIteration* was set to 40 because the success rate was almost saturated, while the cost time and mean iteration were at acceptable levels. The $P_{threshold}$ is another parameter that has to be optimized, and to maintain the expansion ability of random numbers in unknown spaces, this value should not be large. A similar test was conducted to optimize this value, and based on the performance results, the $P_{threshold}$ was set to 15%. The remaining parameters, namely the initial insertion position $\begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}$ and initial insertion angle $\begin{bmatrix} \beta_0 & \gamma_0 \end{bmatrix}$, are listed in Table 1.

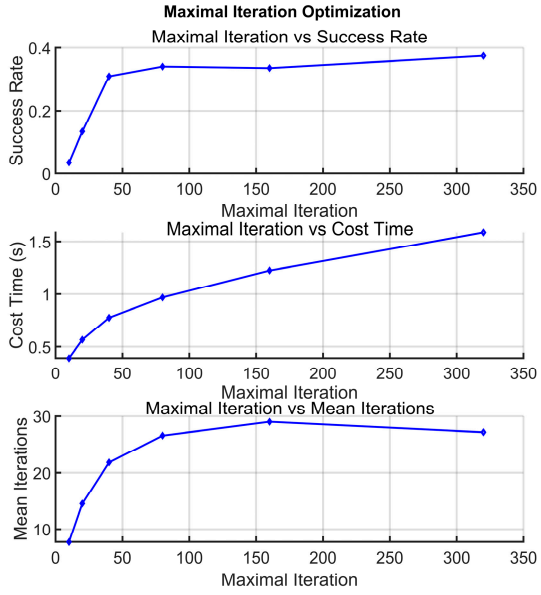


Figure 7. Optimization of the *maxIteration* parameter.

Table 1. Parameters used for the simulation.

Parameters		Values
Max. iterations	$maxIteration$	40
Goal bias	$P_{threshold}$	15%
Insertion position	$[x_0 \quad y_0 \quad z_0]$	[0 mm 0 mm 0 mm]
Insertion angle	$[\beta_0 \quad \gamma_0]$	$[0^\circ \quad 0^\circ]$
Desired insertion velocity	v_{d1}	1 mm/s
Steering coefficient	k	3×10^{-3} mm^{-2}
Initial steering offset	$[\delta_{in0} \quad \delta_{off0}]$	[0 mm 0 mm]
Control gains	$\begin{bmatrix} k_1 & & \\ k_2 & k_3 & k_4 \\ k_5 & k_6 & k_7 \end{bmatrix}$	$\begin{bmatrix} 0.1 & & \\ 0.8 & 0.2 & 0.021 \\ 0.8 & 0.2 & 0.021 \end{bmatrix}$

A feedback controller is needed to track the path generated by the motion planner. A simulation system for the feedback controller was built for tuning the control gains. The simulation sequence was identical to that of the block diagram depicted in Figure 5, except that it excluded the motion planning part. The control gains k_i decide the root placement of the feedback system and the gains should be set properly, such that the root of the control system is on the right side of plane s and the tracking error is convergent. The gains were tuned by trial and error, and the optimal values $[k_1, k_2, k_3, k_4, k_5, k_6, k_7]$ were tuned to $[0.1, 0.8, 0.2, 0.02, 0.8, 0.2, 0.02]$. The remaining parameters for the control system, such as the steering coefficient k and the initial steering offset $[\delta_{in0} \quad \delta_{off0}]$, are listed in Table 1.

5.3. Evaluation of the Optimal Control Algorithm

In the preoperative stage, when the positions of the initial insertion point, initial target point, and obstacle points are known, the static motion-planning algorithm generates 200 trajectories, as depicted in Figure 8, and selects the best candidate among them.

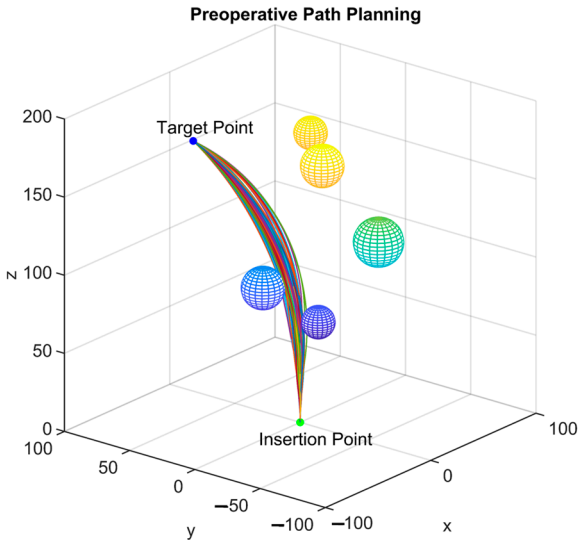


Figure 8. The feasible trajectories generated by the preoperative motion-planning algorithm.

In dynamic environments, the relative motion (with respect to the initial position) of the obstacles $\Delta Pobs$ and target $\Delta Ptar$ are defined using sinusoidal functions, as follows:

$$\begin{aligned} \vec{P}(\theta_1,\ \theta_2) &= \begin{bmatrix} cos\theta_1 cos\theta_2 & sin\theta_1 cos\theta_2 & -sin\theta_2 \end{bmatrix}^T \\ \Delta Pobs &= 8\sin\left(\tfrac{\pi}{25}t\right)\vec{P}, \\ \Delta Ptar &= 5\sin\left(\tfrac{\pi}{300}t\right)\vec{P} \end{aligned} \tag{23}$$

where θ_1 and θ_2 are randomly updated during the steering procedure, thus $\vec{P}$ is a unit vector that represents the motion direction. The amplitudes of the sinusoidal functions are set as 8 and 5 for the obstacles and the target, respectively, such values are sufficiently large compared with the CCS needle diameter (0.432 mm). The frequency of the sinusoidal functions are 50 Hz and 600 Hz, which are sufficiently high since the insertion velocity is 1 mm/s. Through a tradeoff between the optimal extent and loop frequency, the *ValidTreeNum* is set to 3. Based on the updated information, the intraoperative motion planner generates three trajectories and selects the best one among them. The feedback controller tracks the best trajectory and provides the current point to the intraoperative motion-planning algorithm. The trajectory is updated with a resolution of 1 mm, which is adequate considering that the insertion velocity is 1 mm/s.

To emphasize the superiority of the proposed intraoperative motion planner, simulations were conducted by considering two case scenarios. Case I involves the use of the trajectory-following controller alone, and Case II involves the use of the trajectory-following controller in conjunction with the intraoperative motion planner. Each case was tested using three different target points and each test was repeated thrice to increase the credibility of the results. In both cases, an external disturbance with a magnitude of 3 mm offset was provided at the insertion depths of 61 mm and 122 mm, respectively. The simulation results are presented in Table 2, where the performance of these two cases are compared in terms of the mean, standard deviation, maximal tip position, and target errors. The data in the rows for Points 1~3 are the average values of the three repeated tests and the last row presents the overall results. According to the overall results, the case without the intraoperative motion planner yielded a path-tracking accuracy of 3.861 $\pm$ 1.322 mm and an average target error of 4.331 mm. The case with the intraoperative motion yielded a path-tracking accuracy of 0.005 $\pm$ 0.013 mm and an average target error of 0.093 mm.

Table 2. Trajectory-following results of the cases with/without intraoperative motion planning algorithm.

Target Point	Trajectory-Following Accuracy P_{error} (mm)							
	Preoperative Path Tracking				Intraoperative Path Tracking			
	Mean	STD	Max	Target	Mean	STD	Max	Target
Point 1	4.379	1.470	1.062	4.864	0.005	0.015	0.106	0.106
Point 2	3.709	1.281	0.865	3.998	0.005	0.014	0.104	0.104
Point 3	3.495	1.215	0.846	4.131	0.004	0.009	0.068	0.068
Overall Results	3.861	1.322	0.924	4.331	0.005	0.013	0.093	0.093

According to the comparison results, the optimal control algorithm was considerably superior to the algorithm without the intraoperative controller. This is graphically explained in Figure 9, where the red, green, and blue points denote the insertion point, initial target point, and final target point, respectively. As depicted in Figure 9a, the feedback controller can minimize the position error caused by external disturbances. However, it cannot update the desired path and steer the needle to the final target point. In Figure 9b, when the intraoperative motion planner is involved, the motion planner is able to update the intraoperative path from the point of deviation resulting from external disturbances to the target point. Therefore, the position error can not only be compensated for by the feedback controller, but can also be reduced by updating the intraoperative path using the motion planner. Moreover, the final target point can be reached using the motion planner incorporated into the optimal controller. This means that the steering system requires less control input to overcome the error, which reduces the strain on the surrounding tissues.

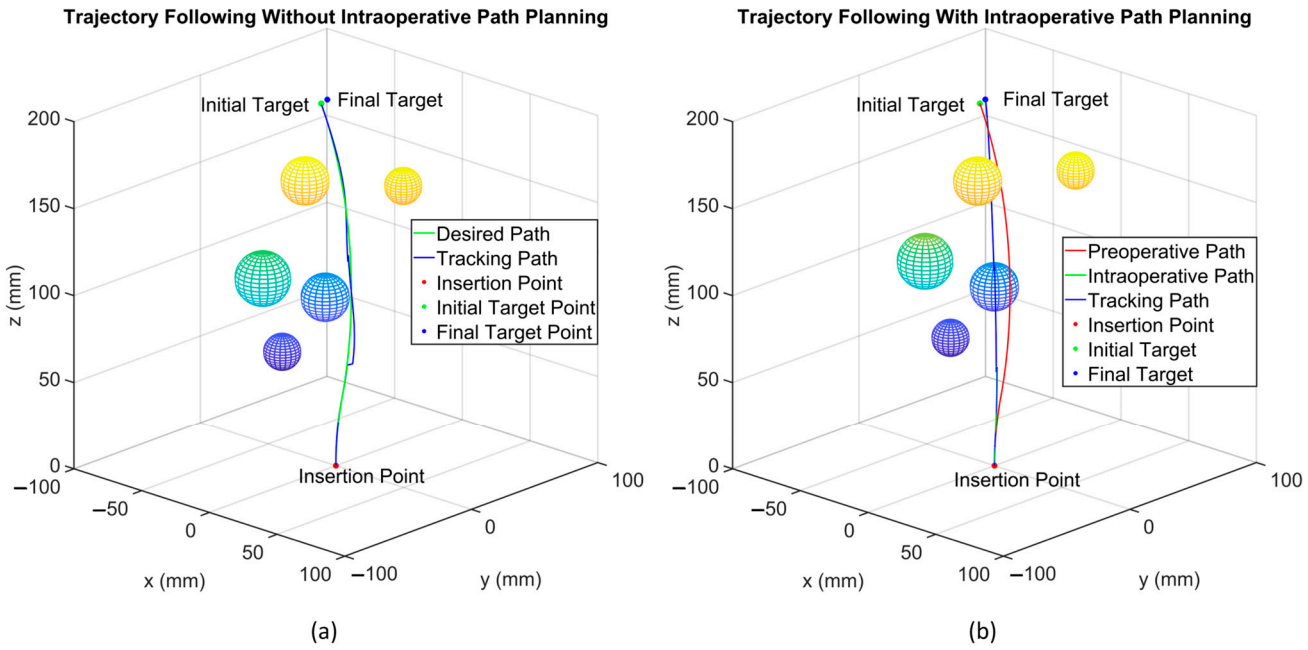


Figure 9. (**a**) Trajectory-following result without intraoperative path planning algorithm. (**b**) Trajectory-following result with intraoperative path planning algorithm.

The above evaluation indicates that the optimal control algorithm, which consists of a full-state space controller and an intraoperative motion planner, not only has superior positioning accuracy in dynamic environments, but is also considerably safer for patients than the conventional needle-steering algorithm.

6. Conclusions

Herein, we presented our research on the optimal control of a CCS needle. Based on the concept of projected steering offset, a 3D kinematic model of a CCS needle was formulated using the Euler angle presentation. A static motion planner was derived from the goal-biased RRT* algorithm, an optimization function was defined for improving steering safety and efficiency, and the best trajectory was selected using the optimization function in the preoperative stage. In addition, an intraoperative motion-planning algorithm was proposed to update the optimal trajectory in dynamic environments. To compute the control input directly, the kinematics was converted into chain form, and a state space controller was used to accurately track the desired trajectory. A simulation system was built to demonstrate the superiority of the optimal control algorithm, the parameters of the motion planner and state space controller were optimized, and a steering test was conducted in regard to a dynamic system with moving obstacles. The resulting target points indicated that the proposed optimal controller was adequate for steering the CCS needle.

Although the simulation results are promising, an experimental demonstration of the optimal control algorithm for CCS needle steering is underway. A real-time ultrasound-sensing module can be used in real settings; several brief experiments can be conducted with the acquisition of the feedback signal from an ultrasound device. Torsional deformation along the needle shaft was ignored in this work, and a compensation algorithm is required to improve the shaft rotation accuracy. The modeling of dynamic environments needs to be explored further. The environment relevant to a specific intervention surgery will be built using the scanning data on the tissue at the lesion site. After ex vivo demonstration of the steering algorithm, the steering system will be implemented in a clinical application.

Author Contributions: Conceptualization, C.Z. and S.Y.K.; methodology, B.X.; software, B.X.; validation, B.X.; formal analysis, B.X.; writing—original draft preparation, B.X.; writing—review and editing, B.X. and S.Y.K.; supervision, C.Z. and S.Y.K.; funding acquisition, B.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the "Pioneer" and "Leading Goose" R&D Program of Zhejiang (Grant No. 2024C03038).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: The authors would like to thank the medical robotics and intelligent control laboratory, Chonnam National University in Korea, for their support in terms of language correction.

Conflicts of Interest: The authors declare that there are no conflicts of interest.

References

1. Fu, M.; Solovey, K.; Salzman, O.; Alterovitz, R. Toward certifiable motion planning for medical steerable needles. *Robot. Sci. Syst.* **2021**. [CrossRef]
2. Wan, G.; Wei, Z.; Gardi, L.; Downey, D.B.; Fenster, A. Brachytherapy needle deflection evaluation and correction. *Med. Phys.* **2008**, *26*, 1261–1268. [CrossRef] [PubMed]
3. Prausnitz, M.R.; Langer, R. Transdermal drug delivery. *Nat. Biotechnol.* **2008**, *26*, 1261–1268. [CrossRef] [PubMed]
4. Zhao, Y.; Shen, Y.; Bernard, A.; Cachard, C.; Liebgott, H. Evaluation and comparison of current biopsy needle localization and tracking methods using 3D ultrasound. *Ultrasonics* **2017**, *26*, 206–220. [CrossRef] [PubMed]
5. Glozman, D.; Shoham, M. Flexible needle steering and optimal trajectory planning for percutaneous therapies. *Lect. Notes Comput. Sci.* **2004**, *3217*, 137–144.
6. Zhao, Y.J.; Wu, W.Q.; Zhang, Y.D.; Wang, R.X.; Peng, J.C.; Yu, Y. 3D Dynamic motion planning for robot-assisted cannula flexible needle insertion into soft tissue. *Int. J. Adv. Robot. Syst.* **2016**, *13*, 121. [CrossRef]
7. Sears, P.; Dupont, P. Inverse Kinematics of Concentric Tube Steerable Needles. In Proceedings of the International Conference on Robotics and Automation, Rome, Italy, 10–14 April 2007.
8. Berg, N.; Dankelman, J.; Dobbelsteen, J. Design of an actively controlled steerable needle with tendon actuation and FBG-based shape sensing. *Med. Eng. Phys.* **2015**, *37*, 617–622. [CrossRef] [PubMed]
9. Petruska, A.J.; Ruetz, F.; Hong, A.; Regli, L.; Surucu, O.; Zemmar, A.; Nelson, B.J. Magnetic Needle Guidance for Neurosurgery: Initial Design and Proof of Concept. In Proceedings of the IEEE International Conference on Robotics and Automation, Stockholm, Sweden, 16–21 May 2016.
10. Xu, B.; Zhou, C.; Ko, S.Y. Closed-loop planar fuzzy control system for a curvature-controllable steerable bevel-tip needle. *Int. J. Control. Autom. Syst.* **2018**, *16*, 2421–2431. [CrossRef]
11. Xu, B.; Ko, S.Y. 3D feedback control using fuzzy logic for a curvature-controllable steerable bevel-tip needle. *Mechatronics* **2020**, *68*, 102368. [CrossRef]
12. Wood, N.; Shahrour, K.; Ost, M.; Riviere, C. Needle Steering System Using Duty-Cycled Rotation for Percutaneous Kidney Access. In Proceedings of the 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology, Buenos Aires, Argentina, 31 August–4 September 2010.
13. Wu, K.; Li, B.; Zhang, Y.; Dai, X. Review of research on path planning and control methods of flexible steerable needle puncture robot. *Comput. Assist. Surg.* **2022**, *27*, 91–112. [CrossRef]
14. Bano, S.; Ko, S.Y.; y Baena, F.R. Smooth Path Planning for a Biologically-Inspired Neurosurgical Probe. In Proceedings of the 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, San Diego, CA, USA, 28 August–1 September 2012.
15. Ko, S.Y.; y Baena, F.R. Toward a miniaturized needle steering system with path planning for obstacle avoidance. *IEEE Trans. Biomed. Eng.* **2013**, *60*, 910–917. [PubMed]
16. Pinzi, M.; Galvan, S.; y Baena, F.R. The adaptive hermite fractal tree (AHFT): A novel surgical 3D path planning approach with curvature and heading constraints. *Int. J. Comput. Assist. Radiol. Surg.* **2019**, *14*, 659–670. [CrossRef] [PubMed]
17. Likhachev, M.; Ferguson, D.I.; Gordon, G.J.; Stentz, A.; Thrun, S. Anytime Dynamic a*: An Anytime, Replanning Algorithm. In Proceedings of the Fifteenth International Conference on International Conference on Automated Planning and Scheduling, Monterey, CA, USA, 5–10 June 2005.
18. Niyaz, S.; Kuntz, A.; Salzman, O.; Alterovitz, R.; Srinivasa, S. Following Surgical Trajectories with Concentric Tube Robots via Nearest-Neighbor Graphs. In Proceedings of the 2018 International Symposium on Experimental Robotics, Buenos Aires, Argentina, 5–8 November 2018.

19. Alteroxitz, R.; Branicky, M.; Goldberg, K. Motion Planning Under Uncertainty for Image-guided Medical Needle Steering. *Int. J. Robot. Res.* **2008**, *27*, 1361–1374. [CrossRef]
20. Panov, A.I.; Yakovlev, K.S.; Suvorov, R. Grid path planning with deep reinforcement learning: Preliminary results. *Procedia Comput. Sci.* **2018**, *123*, 347–353. [CrossRef]
21. Segato, A.; Sestini, L.; Castellano, A.; Momi, E.D. GA3C Reinforcement Learning for Surgical Steerable Catheter Path Planning. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation, Paris, France, 31 May–31 August 2020.
22. Segato, A.; Marzo, M.D.; Zucchelli, S.; Galvan, S.; Secoli, R.; Momi, E.D. Inverse Reinforcement Learning Intra-Operative Path Planning for Steerable Needle. *IEEE Trans. Biomed. Eng.* **2022**, *69*, 1995–2005. [CrossRef]
23. Hong, A.; Boehler, Q.; Moser, R.; Zemmar, A.; Stieglitz, L.; Nelson, B.J. 3D path planning for flexible needle steering in neurosurgery. *Int. J. Comput. Assist. Radiol. Surg.* **2019**, *15*, e1998. [CrossRef] [PubMed]
24. Zhao, Y.J.; Konh, B.; Honarvar, M.; Joseph, F.; Podder, T.K.; Hutapea, P.; Dicker, A.P.; Yu, Y. 3D motion planning for robot-assisted active flexible needle based on rapidly-exploring random trees. *J. Autom. Control. Eng.* **2015**, *3*, 360–367. [CrossRef]
25. Favaro, A.; Segato, A.; Muretti, F.; Momi, E.D. An Evolutionary-Optimized Surgical Path Planner for a Programmable Bevel-Tip Needle. *IEEE Trans. Robot.* **2021**, *37*, 1039–1050. [CrossRef]
26. Pinzi, M.; Watts, T.; Secoli, R.; Galvan, S.; y Baena, F.R. Path Replanning for Orientation-Constrained Needle Steering. *IEEE Trans. Biomed. Eng.* **2021**, *68*, 1459–1466. [CrossRef]
27. Jeong, S.; Choi, H.; Go, G.; Lee, C.; Ko, S.Y.; Park, J.-O.; Park, S. Feasibility study on magnetically steerable guidewire device for percutaneous coronary intervention. *Int. J. Control. Autom. Syst.* **2017**, *15*, 473–479. [CrossRef]
28. Van Den Berg, J.; Patil, S.; Alterovitz, R.; Abbeel, P.; Goldberg, K. LQG-based Planning, Sensing, and Control of Steerable Needles. In *Algorithmic Foundations of Robotics IX*; Hsu, D., Isler, V., Latombe, J.C., Lin, M.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2010.
29. Seoli, R.; y Baena, F.R. Closed-loop 3D Motion Modeling and Control of a Steerable Needle for Soft Tissue Surgery. In Proceedings of the International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 5831–5836.
30. Hong, A.; Petruska, A.J.; Zemmar, A.; Nelson, B.J. Magnetic control of a flexible needle in neurosurgery. *IEEE Trans. Biomed. Eng.* **2021**, *68*, 616–627. [CrossRef] [PubMed]
31. Donder, A.; y Baena, F.R. 3-D Path-Following Control for Steerable Needles with Fiber Bragg Gratings in Multi-Core Fibers. *IEEE Trans. Biomed. Eng.* **2023**, *70*, 1072–1085. [CrossRef] [PubMed]
32. Ko, S.Y.; Frasson, L.; y Baena, F.R. Closed-loop planar motion control of a steerable probe with a "programmable bevel", inspired by nature. *IEEE Trans. Robot.* **2011**, *27*, 970–983. [CrossRef]
33. Luca, A.; De Oriolo, G.; Samson, C. Feedback Control of a Nonholonomic Car-like Robot. In *Motion Planning and Control*; Laumond, J.P., Ed.; Springer: Berlin/Heidelberg, Germany, 1998; pp. 172–253.
34. Path Planning—Improving RRT Algorithm. Available online: https://zhuanlan.zhihu.com/p/51087819 (accessed on 27 August 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.