# 3D Robust Online Motion Planning for Steerable Needles in Dynamic Workspaces Using Duty-Cycled Rotation

**Mariana C. Bernardes · Bruno V. Adorno · Geovany A. Borges · Philippe Poignet**

Received: 22 February 2013 / Revised: 14 November 2013 / Accepted: 10 December 2013 / Published online: 3 January 2014

**Abstract** This work presents a closed-loop strategy for 3D online motion planning of beveled steerable needles using duty-cycled rotation. The algorithm first selects an entry point that minimizes a multi-criteria cost function and then combines an RRT-based path planner with an intraoperative replanning algorithm and workspace feedback information to constantly update the needle inputs and adjust the trajectory. Simulations in a workspace based on a typical prostate needle steering scenario show that the algorithm is robust against disturbances and model uncertainties and can provide online trajectories to avoid obstacles even under the presence of physiological motion.

**Keywords** Medical robotics · Needle steering · Motion planning

## 1 Introduction

Many medical procedures involve percutaneous diagnosis and local therapies that require the insertion of a needle deep into soft tissue to reach a target, and depend on precise tip positioning for effectiveness (Abolhassani et al. 2007). Needle deflection and tissue deformation are the most important factors that affect needle insertion accuracy and require great expertise from the surgeon to compensate for their effects. In addition, the procedure target may be located in a region surrounded by important organs, nerves, or vessels that must be avoided.

Beveled flexible needles capable of being steered during their insertion into soft tissue (Webster et al. 2006) have been designed to allow complex trajectories and expand the applicability of percutaneous procedures to areas of difficult access, which could not be reached by conventional rigid needles without causing excessive, injurious pressure on tissue.

This type of needle can be modeled as a kinematic system with nonholonomic constraints. As a consequence, motion planning is a complex task and its difficulty increases under the presence of uncertainties due to errors in tip positioning, needle modeling, tissue inhomogeneity and deformation. Robot-assisted needle steering has the potential to overcome such complications and has been an area of active research in the past decade. In this context, medical imaging can be used not only for motion planning but also for the control of robot-assisted medical procedures, providing information related to tissue properties, target displacement, and needle tip position (Abolhassani et al. 2007).

Many 3D motion planning methods for beveled steerable needles have been already proposed in the literature. The first work was that of Park et al. (2005), who introduced a diffusion-based approach that considers obstacle-free 3D environments. Later, Duindam et al. (2010) used explicit geometric inverse kinematics for 2D and 3D needle motion planning. Xu et al. (2008) were the first to apply rapidly exploring random tree (RRT)-based methods to steerable needle planning and more recently, Lobaton et al. (2011)

**Electronic supplementary material** The online version of this article (doi:10.1007/s40313-013-0104-4) contains supplementary material, which is available to authorized users.

M. C. Bernardes (✉) · G. A. Borges
Universidade de Brasília, LARA, Caixa Postal 4386,
Brasília, DF 70919-970, Brazil
e-mail: bernardes@lara.unb.br

B. V. Adorno
Universidade Federal de Minas Gerais, Av. Antônio Carlos 6627,
Belo Horizonte, MG 31270-010, Brazil

P. Poignet
Université Montpellier 2, LIRMM, 161 Rue Ada,
34090 Montpellier, France

presented a sampling-based method for planning trajectories with multiple goals. However, none of these methods deal with motion uncertainty caused by modeling approximations, tissue deformation, and other interaction forces that may cause the needle to greatly deviate from its planned path.

One possibility to overcome this limitation is to model such uncertainties and consider their effects during preoperative path planning. Alterovitz et al. (2005) used a finite element mesh (FEM) to compute soft tissue deformations combined with numerical optimization to find a locally optimal initial configuration and insertion distance. A finite element method has also been used by Vancamberg et al. (2010, 2011) to minimize the final error of a RRT solution in a breast biopsy application whereas Patil et al. (2011) used FEM meshes combined with a sampling-based algorithm to plan in highly deformable environments. But the efficiency of these strategies depends a lot on the quality of the mesh simulation and how accurately it represents the real tissue.

Instead of simulating a tissue mesh, Alterovitz et al. (2008) considered uncertainty in needle motion by formulating the planning problem as a Markov Decision Process, using a discretization of the state space and a Stochastic Roadmap (Alterovitz and Goldberg 2007). Both these approaches generate a lookup table using dynamic programming that allows for instantaneous image-guided control for the steerable needle in a static environment. However, if we consider a dynamic environment that presents tissue and anatomical structures displacement due to patient motion or breathing, the use of precomputed paths may not be appropriate. van den Berg et al. (2010) used an LQG-based approach for planning and control of beveled needle insertions. Their technique minimizes the probability of obstacles intersection and deals with noisy sensor measurements, but target and obstacles displacement due to patient motion are not considered.

Another alternative is to apply a single-query planning method that is fast enough to be used intraoperatively—that is, during the medical procedure—in order to replan the trajectory from imaging feedback information. Hauser et al. (2009) were the first to propose a control-loop policy for needle steering in deformable tissue. But again, only obstacle-free environments were considered. Patil and Alterovitz (2010) finally proposed a fast RRT-based algorithm with obstacle avoidance but, although it presented potential for intraoperative use, its performance in closed-loop was not evaluated. In a previous work (Bernardes et al. 2011) we have proposed a 2D planning method that was combined with an intraoperative replanning strategy for systematically correcting the needle trajectory. This algorithm has been improved and extended to the 3D case in the present paper.

### 1.1 Contributions and Organization of the Article

The main contribution of this work was the proposal of an intraoperative strategy for robust 3D needle steering that applies fast replanning during the insertion procedure to compensate for system uncertainties like tissue deformation, tissue inhomogeneity, positioning errors, and other modeling approximations. Different from the previous approaches, our method is not only robust to disturbances, but can also deal with changes in the obstacles and target positions caused by patient and physiological motion during the procedure.

Another contribution was the development of an application-specific path planning algorithm with input sampling strategy to obtain improved performance and efficiency. It explores the duty-cycled rotation strategy (Minhas et al. 2007) to achieve variable curvature paths and consequently achieve a motion planning performance compatible with intraoperative use.

A third contribution was the development of an Entry Point Planner that provides an initial configuration for the needle insertion procedure, which maximizes safety and minimizes length of the generated trajectories based on a multi-criteria cost function.

This article is organized as follows. Section 2 describes the steerable needle nonholonomic model and the principle of the duty-cycle insertion strategy. Then, in Sect. 3 the motion planning problem is detailed, where first we present the direct extension of our previous algorithm, and then we introduce a new improved solution. Also, a derived strategy for selecting feasible insertion entry points is proposed. Section 4 presents the replanning strategy whereas in Sect. 5 we expose the results obtained and the evaluation of our method. Last, in Sect. 6 we discuss its use possibilities, the next steps in development, and the future works.

## 2 Steerable Needle Kinematic Model

The natural behavior of a beveled steerable needle when it is pushed forward is to bend in the direction of its sharpened tip, following an arc of approximately constant curvature $\kappa_{\max} = 1/r$. The natural curvature is related to the geometrical and mechanical properties of the beveled needle and tissue. So, for a given needle–tissue set, $r$ is constant. As demonstrated by Webster et al. (2006), the kinematic model for this kind of needle can be approximated by that of a nonholonomic unicycle vehicle with the following constraints:

$$\begin{aligned} v_y &= v_z = \omega_y = 0 \\ \omega_z &= v_x \kappa_{\max}. \end{aligned} \tag{1}$$

The two control inputs $v_x$ and $\omega_x$, which are, respectively, the needle's insertion and rotation velocities along its shaft

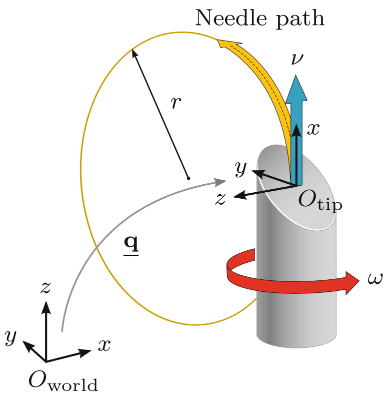


**Fig. 1** Needle coordinate system and control inputs

as illustrated in Fig. 1, are referred from now on as $\nu$ and $\omega$ for simplicity.

The configuration $\underline{\mathbf{q}}$ of the needle tip can be described in 3D by a rigid transformation from $O_{\mathrm{world}}$ to $O_{\mathrm{tip}}$ as shown in Fig. 1. A position is represented by a quaternion $\mathbf{p} = x\hat{\imath}+y\hat{\jmath}+z\hat{k}$ whereas a configuration—that is, a set of position and orientation—is represented by a dual quaternion $\underline{\mathbf{q}} = \mathbf{r} + \epsilon\frac{1}{2}\mathbf{pr}$, with $\mathbf{r}$ and $\mathbf{p}$ being, respectively, the rotation and translation quaternions, and $\epsilon$ being Clifford's dual unit (see Appendix for more details).

A sequence of rigid motions can be represented by a sequence of dual quaternion multiplications. As a consequence, we can obtain a discrete time implementation of the needle kinematic model as

$$\underline{\mathbf{q}}_{k+1} = \underline{\mathbf{q}}_k \, \underline{\mathbf{q}}_\delta, \tag{2}$$

where $\underline{\mathbf{q}}_\delta$ represents the incremental movement during a period $T_\delta$ and is given by

$$\underline{\mathbf{q}}_\delta = \mathbf{r}_\delta + \epsilon\frac{1}{2}\mathbf{p}_\delta\mathbf{r}_\delta. \tag{3}$$

From the nonholonomic constraints (1), we have that the needle rotation is a consequence of the velocity $\boldsymbol{\omega} = \omega\hat{\imath} + \nu\kappa_{\max}\hat{k}$, which is the resultant of $\omega \triangleq \omega_x$ and $\omega_z$ (the quaternionic units $\hat{\imath}$ and $\hat{k}$ represent the $x$ and $z$ axis, respectively). On the other hand, the needle translation is a consequence of $\nu \triangleq \nu_x$ only. Hence, the rotation $\mathbf{r}_\delta$ and translation $\mathbf{p}_\delta$ are given by

$$\begin{aligned} \mathbf{r}_\delta &= \cos\left(\frac{\phi_\delta}{2}\right) + \frac{\sin\left(\frac{\phi_\delta}{2}\right)}{\sqrt{\omega^2+\nu^2\kappa_{\max}^2}}(\omega\hat{\imath} + \nu\kappa_{\max}\hat{k}), \\ \mathbf{p}_\delta &= \nu T_\delta \hat{\imath}, \end{aligned} \tag{4}$$

where $\phi_\delta = \sqrt{\omega^2+\nu^2\kappa_{\max}^2}T_\delta$.

If we combine rotation and insertion movements using a duty-cycling strategy (Minhas et al. 2007), the resultant needle path can achieve different curvature values. The duty-cycling strategy combines periods $T_{\mathrm{ins}}$ of pure insertion with periods $T_{\mathrm{rot}}$ of simultaneous insertion and rotation, so that any curvature ranging from the needle natural curvature to a pure straight trajectory can be achieved.

The duty-cycle DC is defined as the ratio of $T_{\mathrm{rot}}$ to the cycle period $T$,

$$\mathrm{DC} = \frac{T_{\mathrm{rot}}}{T} = \frac{T_{\mathrm{rot}}}{T_{\mathrm{rot}} + T_{\mathrm{ins}}}, \tag{5}$$

and it has an affine relationship to the needle curvature

$$\kappa = \kappa_{\max}(1 - \mathrm{DC}), \tag{6}$$

where $\kappa$ is the effective curvature and $\kappa_{\max}$ is the needle natural curvature, when no rotation is applied.

Hence, given that $\kappa_{\max}$ is a known system parameter, a needle trajectory with curvature $\kappa$ can be achieved by combining two incremental movements: $\underline{\mathbf{q}}_{\mathrm{rot}}$, which represents the movement during simultaneous rotation and insertion ($\omega \triangleq \omega_{\mathrm{ref}}$, $T_\delta \triangleq T_{\mathrm{rot}}$), and $\underline{\mathbf{q}}_{\mathrm{ins}}$ which is the movement during the insertion-only period ($\omega \triangleq 0$, $T_\delta \triangleq T_{\mathrm{ins}}$), both executed according to (4). This can be easily implemented by choosing fixed values for $\omega_{\mathrm{ref}}$ and for the rotation period $T_{\mathrm{rot}}$, both defined a priori. Since $\kappa$ corresponds to the curvature of the arc related to the desired motion, the period of pure insertion $T_{\mathrm{ins}}$ is then obtained from (6) and (5). If the needle is moved by a fixed insertion distance $\triangle s$ at each cycle, the insertion velocity is variable and given by

$$\nu = \frac{\triangle s}{T_{\mathrm{DC}}}, \tag{7}$$

where $T_{\mathrm{DC}} = T_{\mathrm{rot}} + T_{\mathrm{ins}}$.

*Remark 1* It is important to note that the procedure used to obtain the curvature $\kappa$, which is necessary to calculate the $DC$ value in (6) and is related to the arc of the desired motion, will be introduced later in Sect. 3.2.1.

## 3 3D Arc-Based Planners

The needle insertion environment is a bounded 3D workspace and the size and location of the obstacles and the desired target are defined by the surgeon. Also, the parameter $\kappa_{\max}$ for the given needle–tissue combination is considered to be known. In practice, it is usually identified offline from a sequence of multiple insertions or estimated online from visual feedback of the needle trajectory inside tissue.

Approximating the needle movement to that of a 3D unicycle model (Webster et al. 2006), the desired needle path is a combination of circular arcs. The final extremity of each arc $\mathcal{A}$ should correspond to the next arc's initial extremity, not only in position but also in orientation, so we have $C^1$ continuity. In prostate brachytherapy, the final orientation of radioactive seeds may influence the treatment final result. But if we consider percutaneous applications like some types of biopsy, tumor ablation, and anesthesia, the needle orientation at the target may not be a strong problem requirement. Thus, we propose it to be relaxed and used as an extra degree of freedom to obtain such orientation continuity.

### 3.1 Arc-RRT Planner

The objective of a Arc-RRT planner is to find a sequence of duty-cycle signals parameterized on insertion length that is capable of taking the needle from its initial configuration $\underline{\mathbf{q}}_{\text{init}}$ to a final position $\mathbf{p}_{\text{goal}}$ while respecting the nonholonomic constraints and avoiding the obstacles. This sequence must be obtained in a computational speed compatible with intraoperative use, so that the planned path can be adjusted to changes in the workspace and the unexpected behavior of the needle. Our method is based on the classic RRT approach (LaValle and Kuffner 1999)—a tree $\mathcal{T}$ is constructed with its root in $\underline{\mathbf{q}}_{\text{init}}$ and continuously expanded until the goal point $\mathbf{p}_{\text{goal}}$ can be connected to it.

In a previous work (Bernardes et al. 2011), we proposed a first algorithm, the Arc-RRT with point sampling, whose extension to 3D environments is depicted in Algorithm 1. It expands the tree by randomly sampling collision-free points and using them to grow branches and explore the workspace. In this strategy, for each point $\mathbf{p}_{\text{rand}}$ randomly sampled from the workspace, we define a set $\mathcal{Q}_{\text{reachable}}$ of all nodes in $\mathcal{T}$ from which $\mathbf{p}_{\text{rand}}$ can be reached. A node is considered reachable if the arc that connects it to $\mathbf{p}_{\text{rand}}$ does not intersect any obstacles and if its curvature respects the needle maximum value $\kappa_{\max}$. Then, the nearest node from $\mathcal{Q}_{\text{reachable}}$, namely $\underline{\mathbf{q}}_{\text{near}}$, is added to $\mathcal{T}$. The process is repeated until the target can be connected to the tree or until a maximum number of generated samples is reached. The extension of our previous 2D method to a 3D environment incidentally resulted in an algorithm that resembles the one presented in Patil and Alterovitz (2010).

However, after further testing it, we have noticed that this algorithm was not fast enough, especially in cases with reduced needle steerability. In more challenging situations, we have obtained low success rates and long processing times which are not desirable for intraoperative use (see Sect. 5.1 for statistical analysis).

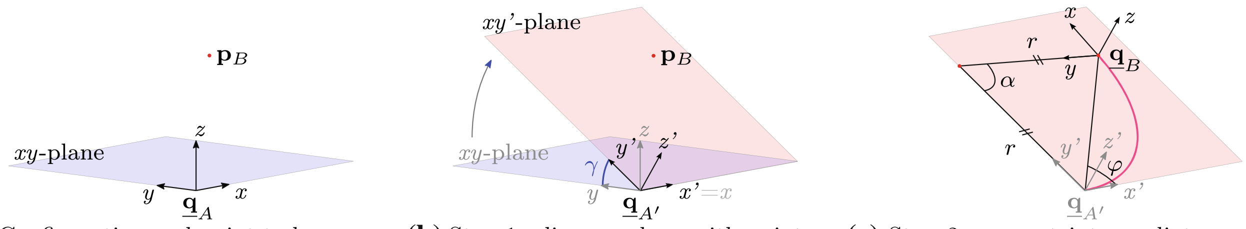
In order to improve the performance of Algorithm 1, in the present work we propose a new sampling strategy and the addition of new functions which result in the Arc-RRT with input sampling, depicted in Algorithm 2 and from now on referred simply as Arc-RRT. The initialization of the algorithm is the same as the previous method, with the construc-

**Algorithm 1** Arc-RRT with point sampling

```
ARC_RRT ($\underline{\mathbf{q}}_{\text{init}}, \mathbf{p}_{\text{goal}}$)
$\mathcal{T} \leftarrow$ INIT_TREE($\underline{\mathbf{q}}_{\text{init}}$)
while $\mathcal{T} \cap \mathbf{p}_{\text{goal}} = \emptyset$ do
    $\mathbf{p}_{\text{rand}} \leftarrow$ RANDOM_POINT()
    $\underline{\mathbf{q}}_{\text{new}} \leftarrow$ RRT_CONNECT_NEAREST($\mathcal{T}, \mathbf{p}_{\text{rand}}$)
    $\underline{\mathbf{q}}_{\text{goal}} \leftarrow$ RRT_CONNECT($\mathcal{T}, \underline{\mathbf{q}}_{\text{new}}, \mathbf{p}_{\text{goal}}$)
end while
$\mathcal{P} \leftarrow$ SEARCH_GRAPH($\mathcal{T}, \underline{\mathbf{q}}_{\text{init}}, \underline{\mathbf{q}}_{\text{goal}}$)
return $\mathcal{P}$

RRT_CONNECT_NEAREST ($\mathcal{T}, \mathbf{p}$)
for all $\underline{\mathbf{q}}_i \in \mathcal{T}$ do
    $\mathcal{A}_i \leftarrow$ GET_ARC($\underline{\mathbf{q}}_i, \mathbf{p}$)
    if COLLISION_FREE($\mathcal{A}_i$) and $\mathcal{A}_i.\kappa < \kappa_{\max}$ then
        add $\mathcal{A}_i$ to $\mathcal{Q}_{\text{reachable}}$
    end if
end for
if $\mathcal{Q}_{\text{reachable}} \neq \emptyset$ then
    $\mathcal{A}_{\text{near}} \leftarrow$ GET_NEAREST($\mathbf{p}, \mathcal{Q}_{\text{reachable}}$)
    $\mathcal{T}$.add_vertex($\mathcal{A}_{\text{near}}.\underline{\mathbf{q}}_B$)
    $\mathcal{T}$.add_edge($\mathcal{A}.\underline{\mathbf{q}}_A, \mathcal{A}.\underline{\mathbf{q}}_B, \mathcal{A}.\kappa$)
    return $\mathcal{A}.\underline{\mathbf{q}}_B$
end if
return NULL

RRT_CONNECT ($\mathcal{T}, \underline{\mathbf{q}}, \mathbf{p}$)
if $\underline{\mathbf{q}} \neq$ NULL then
    $\mathcal{A} \leftarrow$ GET_ARC($\underline{\mathbf{q}}, \mathbf{p}$)
    if COLLISION_FREE($\mathcal{A}$) and $\mathcal{A}.\kappa < \kappa_{\max}$ then
        $\mathcal{T}$.add_vertex($\mathcal{A}.\underline{\mathbf{q}}_B$)
        $\mathcal{T}$.add_edge($\mathcal{A}.\underline{\mathbf{q}}_A, \mathcal{A}.\underline{\mathbf{q}}_B$)
        return $\mathcal{A}.\underline{\mathbf{q}}_B$
    end if
end if
return NULL
```

**Algorithm 2** Arc-RRT with input sampling

```
ARC_RRT ($\underline{\mathbf{q}}_{\text{init}}, \mathbf{p}_{\text{goal}}$)
$\mathcal{T} \leftarrow$ INIT_TREE($\underline{\mathbf{q}}_{\text{init}}$)
while $\mathcal{T} \cap \mathbf{p}_{\text{goal}} = \emptyset$ do
    $\mathbf{p}_{\text{rand}} \leftarrow$ RANDOM_POINT()
    $\underline{\mathbf{q}}_{\text{new}} \leftarrow$ RRT_EXTEND($\mathcal{T}, \mathbf{p}_{\text{rand}}$)
    $\underline{\mathbf{q}}_{\text{goal}} \leftarrow$ RRT_CONNECT($\mathcal{T}, \underline{\mathbf{q}}_{\text{new}}, \mathbf{p}_{\text{goal}}$)
end while
$\mathcal{P} \leftarrow$ SEARCH_GRAPH($\mathcal{T}, \underline{\mathbf{q}}_{\text{init}}, \underline{\mathbf{q}}_{\text{goal}}$)
return $\mathcal{P}$

RRT_EXTEND ($\mathcal{T}, \mathbf{p}$)
$\underline{\mathbf{q}}_{\text{near}} \leftarrow$ NEAREST_NEIGHBOR($\mathcal{T}, \mathbf{p}$)
$u_{\text{rand}} \leftarrow$ RANDOM_INPUT()
$\mathcal{A} \leftarrow$ APPLY_INPUT($\underline{\mathbf{q}}_{\text{near}}, u_{\text{rand}}$)
if COLLISION_FREE($\mathcal{A}$) then
    $\mathcal{T}$.add_vertex($\mathcal{A}.\underline{\mathbf{q}}_B$)
    $\mathcal{T}$.add_edge($\mathcal{A}.\underline{\mathbf{q}}_A, \mathcal{A}.\underline{\mathbf{q}}_B$)
    return $\mathcal{A}.\underline{\mathbf{q}}_B$
end if
return NULL

RRT_CONNECT ($\mathcal{T}, \underline{\mathbf{q}}, \mathbf{p}$)
if $\underline{\mathbf{q}} \neq$ NULL then
    $\mathcal{A} \leftarrow$ GET_ARC($\underline{\mathbf{q}}, \mathbf{p}$)
    if COLLISION_FREE($\mathcal{A}$) and $\mathcal{A}.\kappa < \kappa_{\max}$ then
        $\mathcal{T}$.add_vertex($\mathcal{A}.\underline{\mathbf{q}}_B$)
        $\mathcal{T}$.add_edge($\mathcal{A}.\underline{\mathbf{q}}_A, \mathcal{A}.\underline{\mathbf{q}}_B$)
        return $\mathcal{A}.\underline{\mathbf{q}}_B$
    end if
end if
return NULL
```



**(a)** Configuration and point to be connected.

**(b)** Step 1: align $xy$-plane with point.

**(c)** Step 2: connect intermediate configuration and point with a 3D arc.

**Fig. 2** GET_ARC function connecting a configuration $\underline{\mathbf{q}}_A$ to a point $\mathbf{p}_B$

tion of a tree $\mathcal{T}$ rooted in $\underline{\mathbf{q}}_{\text{init}}$. Then, a random point $\mathbf{p}_{\text{rand}}$ is also sampled from the workspace, but instead of trying to connect it to the tree, we select its nearest neighbor $\underline{\mathbf{q}}_{\text{near}}$ to be extended by applying a sampled input signal $u_{\text{rand}}$. The procedure is repeated until the target $\mathbf{p}_{\text{goal}}$ can be connected to the tree.

The input sampling algorithm has two main advantages when compared to the previous approach. First, each iteration is computed faster since the collision and reachability check is performed for only one node at each iteration, while it was tested for all nodes present in the tree in the point sampling version. Also, the input sampling algorithm presented less rejection of samples, being capable of finding a solution in fewer iterations. Consequently, our new algorithm presents a much better performance when compared to the previous one, as we can observe from the simulation results in Sect. 5.1.

## 3.2 Arc-Based Local Planning

The needle path curvature $\kappa$ is directly related to the duty-cycle sequence of its inputs $\omega$ and $\nu$ as described in Sect. (2). Consequently, in the Arc-RRT planner, we use explicit geometry to describe the needle motion as a sequence of arcs which are connected to expand the tree. This expansion process is based on two local planning functions described as follows.

### *3.2.1 Function GET_ARC*

The duty-cycling technique used to insert the needle allows it to move with different arc curvatures in its *xy*-plane. Consequently, to take its tip from a configuration $\underline{\mathbf{q}}_A$ to a point $\mathbf{p}_B$ in a 3D workspace, the point $\mathbf{p}_B$ must lie in the needle's *xy*-plane. When it is not the case, the needle must be previously rotated by an angle $\gamma$ along its shaft until the *xy*-plane is aligned with $\mathbf{p}_B$. This rotation results in a new tip configuration $\underline{\mathbf{q}}_{A'}$ which can be connected to $\mathbf{p}_B$ by an arc $\mathcal{A}$ which lies in the new *xy*'-plane (see Fig. 2).

Thus, the sequence of movements that will take the needle tip from a given configuration $\underline{\mathbf{q}}_A$ to a given point $\mathbf{p}_B$ is defined by two steps presented as follows.

*Step 1:* In order to obtain the intermediate configuration $\underline{\mathbf{q}}_{A'}$, we must calculate the angle $\gamma$ . First, we write $\mathbf{p}_B$ in $\underline{\mathbf{q}}_A$ frame as $\mathbf{p}_B^A = \mathbf{r}_A^*(\mathbf{p}_B - \mathbf{p}_A)\mathbf{r}_A$, where $\mathbf{r}_A$ and $\mathbf{p}_A$ satisfy $\underline{\mathbf{q}}_A = \mathbf{r}_A + \epsilon\frac{1}{2}\mathbf{p}_A\mathbf{r}_A$.

From the point coordinates $\mathbf{p}_B^A = (a\hat{\imath} + b\hat{\jmath} + c\hat{k})$, the angle $\gamma$ between planes *xy* and *xy'* is given by

$$\gamma = \arctan\left(\frac{c}{b}\right). \tag{8}$$

Note that angles are represented in the interval $(-\pi, \pi]$.

The intermediate configuration $\underline{\mathbf{q}}_{A'}$ is obtained by rotating $\underline{\mathbf{q}}_A$ of an angle $\gamma$ around the *x*-axis

$$\underline{\mathbf{q}}_{A'} = \underline{\mathbf{q}}_A \underline{\mathbf{q}}_\gamma, \tag{9}$$

with $\underline{\mathbf{q}}_\gamma = \cos(\frac{\gamma}{2}) + \sin(\frac{\gamma}{2})\hat{\imath}$.

*Step 2:* From a given start configuration $\underline{\mathbf{q}}_{A'}$ and a final point $\mathbf{p}_B$, it is possible to uniquely define an arc that connects them. The arc parameters can be geometrically obtained with a few trigonometric calculations.

The final point $\mathbf{p}_B$ written in the frame of $\underline{\mathbf{q}}_{A'}$ is given by $\mathbf{p}_B^{A'} = \mathbf{r}_{A'}^*(\mathbf{p}_B - \mathbf{p}_{A'})\mathbf{r}_{A'}$.

Being that $\mathbf{p}_B^{A'} = (a'\hat{\imath} + b'\hat{\jmath})$, the signed bearing angle $\varphi$ of the arc that connects $\underline{\mathbf{q}}_{A'}$ to $\mathbf{p}_B^{A'}$ is given by

$$\varphi = \arctan(\frac{b'}{a'}), \tag{10}$$

and the arc curvature is given by

$$\kappa = \frac{1}{r} = \frac{2\sin(\varphi)}{\sqrt{a'^2 + b'^2}}. \tag{11}$$

The final configuration $\underline{\mathbf{q}}_B$ is given by a sequence of dual quaternion multiplications

$$\underline{\mathbf{q}}_B = \underline{\mathbf{q}}_{A'} \underline{\mathbf{q}}_\alpha, \tag{12}$$

where $\underline{\mathbf{q}}_\alpha = \mathbf{r}_\alpha + \epsilon\frac{1}{2}\mathbf{p}_B^{A'}\mathbf{r}_\alpha$, with $\mathbf{r}_\alpha = \cos(\frac{\alpha}{2}) + \sin(\frac{\alpha}{2})\hat{k}$ and $\alpha = 2\varphi$.

### 3.2.2 Function APPLY_INPUT

Instead of connecting the configuration $\underline{\mathbf{q}}_A$ to a given point, one might want to expand it with the concatenation of a given 3D arc segment. This arc is obtained from the random input $u_{\text{rand}} = (\gamma_{\text{rand}}, \varphi_{\text{rand}}, \kappa_{\text{rand}})$,[1] whose parameters are related to the arc plane, bearing angle and curvature, respectively. Hence, once arc parameters $\gamma$, $\varphi$, and $\kappa$ are known, they are used to determine a corresponding final configuration $\underline{\mathbf{q}}_B$.

First, we apply $\gamma$ to find $\underline{\mathbf{q}}_{A'}$ as given in (9). Then, from the curvature $\kappa$, we can determine the position of the arc center as $\mathbf{p}_C = \mathbf{r}_{A'}\mathbf{p}_C^{A'}\mathbf{r}_{A'}^{*} + \mathbf{p}_{A'}$, where $\mathbf{p}_C^{A'} = \frac{1}{\kappa}\hat{\jmath}$ is the position of the center with respect to the frame of $\underline{\mathbf{q}}_{A'}$.

At last, from the bearing angle $\varphi$, we can get the arc sector angle $\alpha = 2\varphi$. The final point position is given by

$$\mathbf{p}_B = \mathbf{r}_{A'}\mathbf{p}_B^{c}\mathbf{r}_{A'}^{*} + \mathbf{p}_C, \tag{13}$$

being $\mathbf{p}_B^{c} = r\sin(\alpha)\hat{\imath} - r\cos(\alpha)\hat{\jmath}$ its position with respect to the arc center. Finally, the final configuration $\underline{\mathbf{q}}_B$ is

$$\underline{\mathbf{q}}_B = \mathbf{r}_B + \epsilon\frac{1}{2}\mathbf{r}_B\mathbf{q}_B, \tag{14}$$

with $\mathbf{r}_B = \mathbf{r}_{A'}\mathbf{r}_\alpha$ and $\mathbf{r}_\alpha = \cos(\frac{\alpha}{2}) + \sin(\frac{\alpha}{2})\hat{k}$.

### 3.3 Entry Point Planner

The objective of the Entry Point Planner (EP Planner) is to find a feasible starting configuration for the needle insertion procedure. It is especially useful when the target is in a region of difficult access and the starting configuration requirements may be relaxed. In this case, instead of having a defined $\underline{\mathbf{q}}_{\text{start}}$, we only constrain the entry point to a region of interest from where the insertion can be performed. Consequently, we increase the chances of finding a feasible path, even for the cases where the procedure target is in a difficult configuration (see Sect. 5.3 for example).

In order to search for a feasible entry point, the planner discretizes the entry region and runs the Arc-RRT algorithm using each one of the points as a starting point candidate. For each candidate that successfully returns a path $\mathcal{P}$, we calculate a cost function $\mathcal{C}(\mathcal{P})$ and the lowest cost path is selected. For the cases where there is no constraint for the insertion entry angle, the region discretization can be adapted to also include a discretization of the possible initial orientations.

We have chosen a cost function that takes in consideration two factors conflicting with each other Mittal and Deb (2007): the total path length and the collision risk. The objective of having a short insertion clashes with the desire of going as far as possible from obstacles. The closer to obstacles the needle passes, the higher are the chances of collision. Hence, we define the two functions

$$f_{\text{risk}} = \sum_{i=1}^{N}\left(\frac{d_{\text{safe}}}{d_{i,\min}}\right)^2 \quad \text{and} \quad f_{\text{length}} = \sum_{i=1}^{K}\left|\frac{2\varphi_i}{\kappa_i}\right|, \tag{15}$$

where $N$ is the number of discretization points of the trajectory, $d_{i,\min}$ is the minimum distance from point $i$ to obstacles, $d_{\text{safe}}$ is the minimum safety distance the needle should have from the obstacles, $K$ is the number of arcs in the trajectory, $\varphi_i$ is the bearing angle from the $i$-th arc, and $\kappa_i$ is its correspondent curvature.

The final cost $\mathcal{C} = \beta f_{\text{risk}} + (1-\beta) f_{\text{length}}$ is a weighted sum of both functions, with $\beta$ being a weighting factor. If the given region is reasonably large, we may adapt the algorithm in order to make it faster. For this, we make a coarse initial discretization of the entry region and after we have found the lowest cost path, we perform a finer discretization around the chosen point to check for a better solution.

## 4 Intraoperative Replanning

Using the affine relationship (6) between duty-cycle and arc curvature, we obtain the sequence of DC inputs that will take the needle from its insertion point to the goal while following the desired path. However, tissue deformation and inhomogeneity, imprecision of the unicycle model, and uncertainties in the initial configuration may deflect the needle from the planned trajectory, leading to a possible collision with an important organ or misplacement of the needle tip at the end of the insertion task.

**Algorithm 3** Intraoperative Replanning

```
INTRAOPERATIVE_REPLANNING($\underline{\mathbf{q}}_{\text{tip\_now}}$, $\mathbf{p}_{\text{goa]\_now}}$, $\mathcal{P}$)
$\mathcal{P}_{\text{now}} \leftarrow$ UPDATE_PATH_EXTREMITIES($\underline{\mathbf{q}}_{\text{tip\_now}}$, $\mathbf{p}_{\text{goal\_now}}$, $\mathcal{P}$)
for all $\mathcal{A}_i \in \mathcal{P}_{\text{now}}$ do
    $\mathcal{A}_i \leftarrow$ GET_ARC($\mathcal{A}_i.\underline{\mathbf{q}}_A$, $\mathcal{A}_i.\mathbf{p}_B$)
    if !COLLISION_FREE($\mathcal{A}_i$) or $\mathcal{A}_i.\kappa > \kappa_{\max}$ then
        return ARC_RRT($\underline{\mathbf{q}}_{\text{tip\_now}}$, $\mathbf{p}_{\text{goal\_now}}$)
    end if
end for
return $\mathcal{P}_{\text{now}}$
```

To avoid this, we propose a replanning strategy which should be executed at each cycle to systematically correct the trajectory until the needle tip is sufficiently close to the target (see Algorithm 3). We assume that the current information about the workspace is provided by a 3D medical imaging

[1] Although we use here the expression "input," one should notice that we do not refer to the needle actuation inputs $\omega$ and $\nu$, but to pure geometrical parameters used to expand the RRT tree. The insertion and velocity actuation inputs are actually a consequence of the planned path formed by the concatenation of arcs obtained from such geometrical parameters.

system. The workspace feedback is used to update the path by considering the needle's current configuration $\underline{\mathbf{q}}_{\mathrm{tip\_now}}$ as the new starting position for the path, and the target's current position $\mathbf{p}_{\mathrm{goal\_now}}$ as the new goal. Then, the GET_ARC function (see Sect. 3.2.1) is run to adjust all the arcs from $\mathcal{P}$, recalculating the new curvatures and the final orientations. If a collision is detected in the updated arcs, or if the new curvature does not respect the maximum limit, the complete RRT planner is run again to find a new feasible trajectory.

The update of the path is extremely fast since it uses only the geometric relations defined in Sect. (3.2) to satisfy the system nonholonomic constraints. Combined with the good performance of the Arc-RRT with input sampling, and considering that needle insertion procedures normally occur at small insertion and rotation velocities due to safety restrictions, the replanning algorithm can be easily used intraoperatively.

In addition, since the Arc-RRT algorithm converges much faster than the insertion procedure speed, the remaining time between cycles can be used to accumulate executions of the planner. Then, the same cost function from the EP Planner could be used to choose the best path between the multiple solutions found. This way, the planned trajectory gets closer to the optimal path without the need of using any expensive numerical optimization in favor of intraoperative use.

It is important to note that the needle insertion procedure is performed at very low velocities due to safety requirements of medical interventions. This results in a relatively slow dynamics, so that the intraoperative replanning does not need to be performed in hard real time. However, although the replanning is usually not performed in real time, the device used to control the insertion procedure is usually real time controlled at the low level.

## 5 Results

Our tests were conducted in the simulated 3D environment with obstacles depicted in Fig. 3 with all units in centimeters. It is based on a typical scene for prostate needle insertion and consists of a cubical region with coordinates $(-5, 5) \times (-5, 5) \times (-5, 5)$ and six unit-radius spheres, centered at $[-1, 0, 0]$, $[3.5, 0, 1.5]$, $[2.5, 0, 2.9]$, $[5, 0, 2]$, $[0.5, 1.4, 0.3]$, and $[0.5, -1.4, 0.3]$ to represent obstacles around the prostate such as the pubic arch, the urethra, and the penile bulb (Xu et al. 2008). The two blue rectangles represent regions of interest for possible entry points and targets. Simulations were run in a PC with Intel Core i7 2.93 GHz, 2.9 GB memory and Ubuntu 10.10 operating system. To evaluate the performance of the proposed planners, we performed three different sets of simulations.

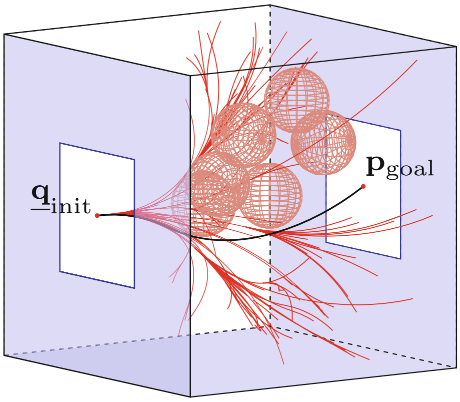

$\underline{\mathbf{q}}_{\mathrm{init}}$
$\mathbf{p}_{\mathrm{goal}}$

**Fig. 3** 3D scenario for simulations based on Xu et al. (2008). The *spheres* represent obstacles, the *rectangles* are the regions of interest, the *red lines* compose the tree constructed by an execution of Arc-RRT with input sampling, and the *black line* is the path found (Color figure online)

### 5.1 Comparison of Algorithms with Point and Input Sampling

The first set of simulations compares the performance of the Arc-RRT with point and input sampling with respect to computation time, success rate, and insertion length. We specified a needle with 4 cm of minimum radius of curvature.

*Simulation 1* First, we compared the computational time needed by each algorithm to build a tree with the same amount of nodes. The initial configuration was set to the position $(x, y, z) = (-5, 0, 0)$ and aligned with the world frame (i.e, $\mathbf{p}_{\mathrm{init}} = -5\hat{\imath}$ and $\mathbf{r}_{\mathrm{init}} = 1$), and for each algorithm, a tree was created and expanded until it reached 500 nodes. This process was repeated for 1,000 trials and the obtained average computation time in milliseconds is 648.58 ± 93.45 for Algorithm 1 (point sampling) and 159.63 ± 90.75 for Algorithm 2 (input sampling). We can see that the input sampling strategy expands the tree much faster than the point sampling one. The main cause of such difference is because for each iteration, the point sampling algorithm has to check the reachability of all the nodes in the tree, while the input sampling method performs the check for only one node.

*Simulation 2* Second, we compared the ability of each algorithm to find a solution for the same planning situation. In this test, the maximum number of samples for constructing the RRT was set to 1,000 and, for each trial, an initial configuration and a goal point were randomly picked with an uniform distribution from the regions of interest. Each set of $\underline{\mathbf{q}}_{\mathrm{init}}$ and $\mathbf{p}_{\mathrm{goal}}$ was tested in both algorithms. Table 1 presents the results for 1,000 trials, indicating a better performance of Algorithm 2 (input sampling), which had a higher rate of success and lower processing time when compared to Algorithm 1 (point sampling). From the data, we observe that the

**Table 1** Performance of algorithms 1 and 2

| | Algorithm 1<br>point sampling | Algorithm 2<br>input sampling |
|---|---|---|
| Number of trials | 1,000 | 1,000 |
| Number of successes | 768 | 951 |
| Number of samples | 224 ± 198 | 79 ± 132 |
| CPU time (ms) | 33.19 ± 39.46 | 3.46 ± 7.59 |
| Path length (cm) | 10.55 ± 3.92 | 10.07 ± 1.64 |

input sampling strategy is capable of finding a solution with fewer samples when compared to point sampling.

Thus, by changing from point to input sampling, we obtained a new path planner with higher success rate, and more than 9 times faster than the previous one, without increase in the average insertion length. This indicates that the use of the new algorithm is advantageous to an intraoperative planner.

### 5.2 Performance of the Arc-RRT with Input Sampling

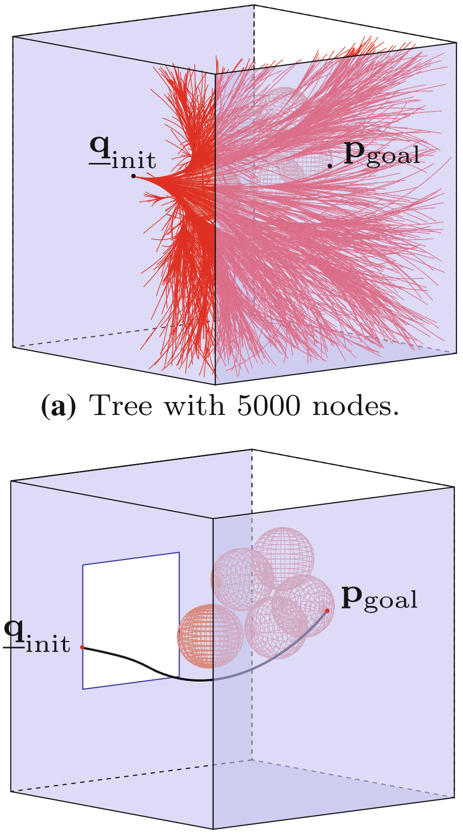
The next simulation evaluates the influence of the needle steerability in the performance of the Arc-RRT with input sampling (Algorithm 2). For each trial, an initial configuration and a target point are randomly picked from an uniform distribution. The maximum number of nodes for the RRT is 2500. For the kinematic model of the needle, we specified three different minimum radii of curvature (4, 5, and 6 cm). Table 2 presents the obtained results for 1,000 trials. Feasible paths were found in most of the trials, with a success rate higher than 95 % in the worst case. However, we can notice that the performance of the planner is strongly dependent on needle steerability. As we raise the minimum radius of curvature, the harder it gets to find the feasible paths, until the limit where there is no possible solution for a given combination of start configuration and target position. In such cases, an alternative is to relax the needle initial configuration requirement and use the EP Planner to find a feasible path within a region of interest, as illustrated by the next simulation.

### 5.3 Performance of the Entry Point Planner

An especially difficult combination for the given workspace (see Fig. 3) is the initial configuration at position $\mathbf{p}_{\text{init}} = -5\hat{\imath}$ and $\mathbf{r}_{\text{init}} = 1$, and $\mathbf{p}_{\text{goal}} = 5\hat{\imath}$, for which no solution could be found with $r = 6$ cm. Figure 4 shows one of the exploration trees for this configuration with 5,000 nodes.

**Table 2** Performance of algorithm 2 (input sampling)

| | $r = 4$ | $r = 5$ | $r = 6$ |
|---|---|---|---|
| Number of trials | 1,000 | 1,000 | 1,000 |
| Number of successes | 1,000 | 982 | 953 |
| Number of samples | 220 ± 99 | 459 ± 209 | 627 ± 329 |
| CPU time (ms) | 6.14 ± 23.40 | 19.83 ± 66.33 | 35.40 ± 89.99 |


$\underline{\mathbf{q}}_{\text{init}}$
$\mathbf{p}_{\text{goal}}$
$\underline{\mathbf{q}}_{\text{init}}$
$\mathbf{p}_{\text{goal}}$

**(a)** Tree with 5000 nodes.

**(b)** Solution from EP Planner.

**Fig. 4** Scenario of Simulation C with a difficult combination of $\underline{\mathbf{q}}_{\text{init}}$ and $\mathbf{p}_{\text{goal}}$ and lower needle steerability

**Table 3** Performance of the EP planner

| | $r = 6$ |
|---|---|
| Number of trials | 100 |
| Number of successes | 100 |
| Number of samples | 17798 ± 1390 |
| CPU time (s) | 13.17 ± 1.66 |

Then, we used the EP Planner to search the region of interest for a new feasible start configuration. Table 3 presents the simulation results. In total, 100 trials were performed and all of them were able to find a path that connects to $\mathbf{p}_{\text{goal}}$, confirming the interest in using the EP Planner for finding a better start configuration whenever the planning requirements may be relaxed.

### 5.4 Needle Insertion Under Disturbances

The next simulation evaluates the intraoperative replanning strategy and its capability of driving the needle to the desired target even with the presence of model uncertainties and disturbances such as tissue deformation, tissue inhomogeneity, and needle tip positioning errors which are inherent to medical imaging and sensor measurements.[2]

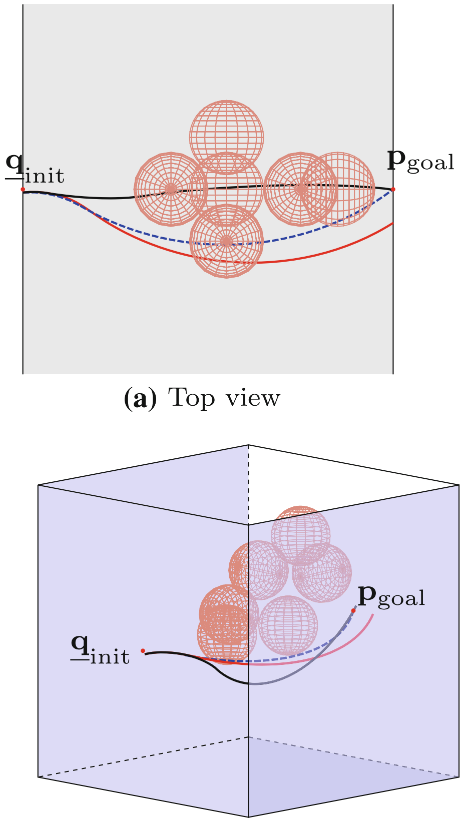
In simulation, these effects were modeled as white noises added to the measured tip configuration and to the applied needle rotation. Such induced errors follow a normal distribution N $\sim (\mu, \sigma)$ with $\mu = 0$ and $\sigma = 1$ mm to the tip position, and $\mu = 0$ and $\sigma = 0.01$ rad to the orientation. The same perturbations were also included to the tip initial position and orientation, while the actual natural curvature was simulated with a value 25 % bigger than that expected by the planning algorithm. The workspace configuration, $\underline{\mathbf{q}}_{\text{init}}$ and $\mathbf{p}_{\text{goal}}$ were the same from Simulation 5.3.

In a preoperative step, the Arc-RRT with input sampling was used to obtain a feasible path given the initial tip configuration, target position, and workspace. The correspondent sequence of duty-cycles parametrized in path length was directly calculated from the arc's curvatures, and applied to the simulated needle. The needle tip was simulated using the discrete model from (4), with natural radius of curvature of 4 cm, rotational velocity of 2 Hz (i.e., $T_{\text{rot}} = 0.5$ s), and insertion distance per cycle $\triangle s = 1$ mm.

At this point, it is important to recall that, in order to calculate the needle input signals for each desired arc curvature $\kappa$, we use (6) to obtain $DC$. Using (5), and assuming a fixed $T_{\text{rot}}$, we obtain $T_{\text{ins}}$. This way, considering a constant insertion distance per cycle, we calculate $v$ from (7). Since we also define a constant $\omega_{\text{ref}}$, both values are used in (4) to obtain (2).

At each simulation cycle, our replanning strategy was used to systematically correct the trajectory along the insertion procedure according to the current measured tip configuration. Figure 5 shows an example of the replanning in action. The initial planned trajectory is shown in dashed blue and it would lead the needle tip to the desired target if it were not for the induced disturbances.

Without the replanning action, the effects of such disturbances cause the needle to follow the trajectory in red, with a final error of 8.3 mm to the target location. Instead, if the replanning algorithm is used, as the needle insertion advances, the planned trajectory is updated in closed-loop. The constant online replanning resulted in the path depicted in solid black, with a final error of only 0.6 mm. The same simulation was repeated several times resulting in a final mean error of 0.66 mm and standard deviation of 0.23 mm in 10 trials, a much better accuracy than the open-loop insertion.



**(a)** Top view

**(b)** Isometric view

**Fig. 5** Performance of the intraoperative replanning. In *dashed lines*, the initial planned path; in *red* the trajectory obtained in open-loop and in *black* the trajectory with replanning. See provided Online Resource 1 for other insertion examples (Color figure online)

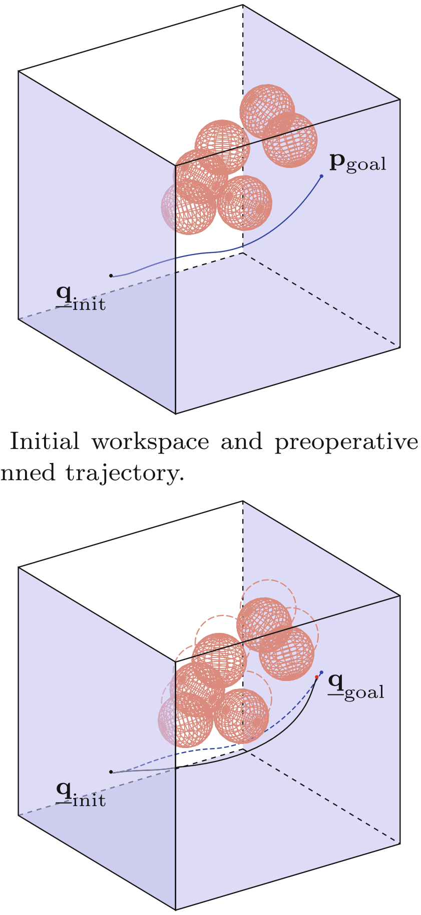
### 5.5 Needle Insertion in a Dynamic Environment

The last simulation evaluates the effects of changes in the workspace during needle steering and the ability of our intraoperative replanning strategy to compensate for these changes. When we consider the presence of patient and physiological motion between preoperative planning and treatment phases, the result can greatly deviate from the expected one. In simulation, these changes in the insertion scene were modeled as periodic sinusoidal motions applied to the obstacles and to the desired target positions with a period of 5 s and 40 s, respectively. The displacement amplitude was 2 mm in all directions for the obstacles, whereas for the target it was 3 mm for $y$ and $z$ directions.[3]

The initial tip configuration and goal position were $\mathbf{p}_{\text{init}} = -5\hat{\imath} - 1\hat{\jmath} - 1\hat{k}$ and $\mathbf{r}_{\text{init}} = \cos(\pi/6) - \sin(\pi/6)\hat{k}$, which were obtained from the EP Planner algorithm, while the initial workspace and simulation parameters were the same from Simulation 5.4, including the disturbances added to the system.

[2] For videos, see the first part of Online Resource 1.

[3] For videos, see the second part of Online Resource 1.


$\mathbf{p}_{\text{goal}}$
$\underline{\mathbf{q}}_{\text{init}}$
$\underline{\mathbf{q}}_{\text{goal}}$
$\underline{\mathbf{q}}_{\text{init}}$

**(a)** Initial workspace and preoperative planned trajectory.

**(b)** Final workspace and trajectory resultant of systematic intraoperative replanning.

**Fig. 6** Example of needle simulation with dynamic workspace and intraoperative trajectory replanning. The *dashed lines* represent the original positions of obstacles and the initial planned path; in *solid black* the trajectory with replanning. The *blue* (*right*) and *red* (*left*) *dots* indicate the initial and final target positions, respectively. See provided Online Resource 1 for other insertion examples (Color figure online)

Similar to the previous simulation, the Arc-RRT algorithm was used to obtain a feasible trajectory, and as the insertion moves on, the planned trajectory was updated by the Intraoperative Replanning to adjust according to the changes observed in the obstacles and target positions.

This simulation with dynamic environment was repeated for 10 trials and the needle was able to avoid the obstacles while reaching the moving target with a final mean error of 0.83 mm and standard deviation of 0.47 mm, which is considered to be a good precision for the majority of percutaneous procedures (Fichtinger et al. 2008). One of the trials is depicted in Fig. 6, where we can observe the replanning in action. The initially planned trajectory is shown in Fig. 6a (solid blue line) and if it was not for the workspace motion, it would lead the needle tip to the desired target while avoiding the obstacles.

However, if we consider the displacement of the scene objects as shown in Fig. 6b, the initial trajectory (dashed blue line) would cause the needle to collide with an obstacle and miss the current target position, which has also moved during the insertion. Instead, because the replanning algorithm was used, as the needle insertion advanced the planned trajectory was updated in closed-loop. The constant online replanning resulted in the path depicted in solid black, with a final error of only 0.33 mm.

## 6 Conclusions

In this work we proposed a closed-loop strategy for 3D motion planning of steerable needles using duty-cycling. It combines an RRT-based path planner with an intraoperative algorithm and workspace feedback information to constantly update the needle inputs and adjust the trajectory.

The new Arc-RRT planner has shown to have a good success rate while being fast enough to be used in an intraoperative system. We also presented a variation to use it as a preoperative planner for calculation of feasible insertion entry points that maximize safety and minimize path length. Even though the convergence of RRT-based algorithms is not assured in finite time, the RRT is proved to be probabilistic complete, meaning that if we give it enough time to search for the solution and if the solution exists, it will be found (Kuffner and LaValle 2000). In practice, what we observe is that the Arc-RRT converges much before the next insertion cycle due to its high success rate and fast execution when compared to the insertion procedure speed. As a consequence, we may use the remaining time between cycles to accumulate executions of the Arc-RRT algorithm and use the same cost function from the EP Planner to choose the best path so the planned trajectory gets closer to the optimal without the need of using any expensive numerical optimization in favor of intraoperative use.

The proposed replanning strategy made the insertion procedure more robust to system uncertainties such as tissue deformation, errors in position, inhomogeneity, and modeling approximations. The simulation results showed that, even under perturbations, the needle was able to reach the target with a satisfactory precision while an open-loop strategy would fail to avoid the obstacles and to arrive at the desired position. Simulations in a dynamic workspace also suggest that the replanning could also be used to compensate for small magnitude physiological movement if the tissue–needle combination presents good steerability.

The next step is to evaluate the proposal on tissue phantoms and robotic hardware, which we have partially implemented for the 2D case in Bernardes et al. (2012). Possible future works include the investigation of motion planners that combine the duty-cycling strategy with smoother trajectories such as splines, intraoperative estimation of needle curvature, and prediction of the obstacles displacement.

## 7 Online Resource

Online Resource 1 provides a video showing simulations of the needle insertion under the effect of disturbances, as described in Sect. 5.4. During the insertion, we use the proposed strategy for intraoperative replanning to update the desired needle trajectory according to the current needle tip configuration. As a consequence, the needle converges to the target, despite the model uncertainties and positioning errors added to the simulation. In the second part, the video shows simulations of the needle insertion in dynamic workspaces, as described in Sect. 5.5. Even with small displacements of obstacles and target, the needle is able to reach its final destination.

**Acknowledgments** This work was supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), and by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

## 8 Appendix

A quaternion **h** consists of a real component plus an imaginary part composed of three quaternionic units $\hat{\imath}, \hat{\jmath}, \hat{k}$; that is, $\mathbf{h} = a + b\hat{\imath} + c\hat{\jmath} + d\hat{k}$, where $a, b, c, d \in \mathbb{R}$, $\hat{\imath}^2 = \hat{\jmath}^2 = \hat{k}^2 = -1$, and $\hat{\imath}\hat{\jmath}\hat{k} = -1$. Its conjugate is given by $\mathbf{h}^* = a - b\hat{\imath} - c\hat{\jmath} - d\hat{k}$.

A rotation composed of a rotation angle $\phi$ around the axis $\mathbf{n} = n_x\hat{\imath} + n_y\hat{\jmath} + n_z\hat{k}$ is given by the quaternion $\mathbf{r} = \cos(\phi/2) + \sin(\phi/2)\mathbf{n}$.

A translation **p** is represented by a pure quaternion; that is, a quaternion where the real part is equal to zero. Thus, $\mathbf{p} = p_x\hat{\imath} + p_y\hat{\jmath} + p_z\hat{k}$.

The rigid motion is then completely represented by the dual quaternion $\underline{\mathbf{q}} = \mathbf{r} + \epsilon\frac{1}{2}\mathbf{pr}$, where $\epsilon$ is Clifford's dual unit, which is nilpotent; that is, $\epsilon \neq 0$ but $\epsilon^2 = 0$.

## References

Abolhassani, N., Patel, R., & Moallem, M. (2007). Needle insertion into soft tissue: A survey. *Medical Engineering and Physics*, *29*, 413–431.

Alterovitz, R., & Goldberg, K. (June 2007). The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty. In *Proceedings of the Robotics: Science and Systems Conference*, Atlanta, GA, pp. 1–8.

Alterovitz, R., Branicky, M., & Goldberg, K. (2008). Motion planning under uncertainty for image-guided medical needle steering. *The International Journal of Robotics Research*, *2*(11–12), 1361–1374.

Alterovitz, R., Goldberg, K., & Okamura, A. (2005). Planning for steerable bevel-tip needle insertion through 2D soft tissue with obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1640–1645.

Bernardes, M. C., Adorno, B. V., Poignet, P., & Borges, G. A. (2012). Semi-automatic needle steering system with robotic manipulator. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1595–1600.

Bernardes, M. C., Adorno, B. V., Poignet, P., Zemiti, N., & Borges, G. A. (2011). Adaptive path planning for steerable needles using duty-cycling. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2545–2550.

Duindam, V., Alterovitz, R., Sastry, S., & Goldberg, K. (2010). Three-dimensional motion planning algorithms for steerable needles using inverse kinematics. *The International Journal of Robotics Research*, *29*(7), 789–800.

Fichtinger, G., Kazanzides, P., Okamura, A. M., Hager, G. D., Whitcomb, L. L., & Taylor, R. H. (2008). Surgical and Interventional Robotics: Part II. In *IEEE Robotics and Automation Magazine*, pp. 94–102.

Hauser, K., Alterovitz, R., Chentanez, N., Okamura, A., & Goldberg, K. (June 2009). Feedback control for steering needles through 3D deformable tissue using helical paths. In *Proceedings of the Robotics: Science and Systems Conference*, Seattle, WA, pp. 1–8.

Kuffner, J. J., & LaValle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 995–1001.

LaValle, S. M., & Kuffner, J. J. (1999). Randomized kinodynamic planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 473–479.

Lobaton, E., Zhang, J., Patil, S., & Alterovitz, R. (2011). Planning curvature-constrained paths to multiple goals using circle Sampling. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1463–1469.

Minhas, D., Engh, J., Fenske, M., & Riviere, C. (2007). Modeling of needle steering via duty-cycled spinning. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 2756–2759.

Mittal, S., & Deb, K. (2007). Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 3195–3202.

Park, W., Kim, J., Zhou, Y., Cowan, N. J., Okamura, A., & Chirikjian, G. (2005). Diffusion-based motion planning for a nonholonomic flexible needle model. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 4611–4616.

Patil, S., & Alterovitz, R. (2010). Interactive motion planning for steerable needles in 3D environments with obstacles. In *Proceedings of the IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, pp. 893–899.

Patil, S., van den Berg, J. P., & Alterovitz, R. (2011). Motion planning under uncertainty in highly deformable environments. In *Proceedings of the Robotics: Science and Systems Conference*.

van den Berg, J. P., Patil, S., & Alterovitz, R. (2010). LQG-based planning, sensing, and control of steerable needles. In *Proceedings of the International Workshop on the Algorithmic Foundations of Robotics*, pp. 373–389.

Vancamberg, L., Sahbani, A., Muller, S., & Morel, G. (2010). Needle path planning for digital breast tomosynthesis biopsy. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2062–2067.

Vancamberg, L., Sahbani, A., Muller, S., & Morel, G. (2011). Needle path planning for digital breast tomosynthesis biopsy using a heterogeneous model. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 5749–5755.

Webster III, R. J., Kim, J., Cowan N. J., Chirikjian, G., & Okamura, A. (2006). Nonholonomic modeling of needle steering. *The International Journal of Robotics Research*, *25*(5–6), 509–525.

Xu, J., Duindam, V., Alterovitz, R., & Goldberg, K. (2008). Motion planning for steerable needles in 3D environments with obstacles using rapidly-exploring Random Trees and backchaining. In *Proceedings of the IEEE International Conference on Automation Science and Engineering*, pp. 41–46.