

PROFICIENCY BASED PLANNER FOR SAFE PATH PLANNING AND APPLICATIONS IN SURGICAL TRAINING

Shubham Jain
Minsik Hong
Jerzy W. Rozenblit

Department of Electrical and Computer Engineering
University of Arizona
1230 East Speedway Boulevard, Tucson, AZ, USA
{shubhamjain,mshong}@email.arizona.edu, jr@ece.arizona.edu

ABSTRACT

Instrument navigation is a fundamental task in laparoscopic surgical training. To assist a trainee in this task, suggested path needs to be collision free and should maintain sufficient distance, or *clearance* from the obstacle so that the possibility of collision is minimized. Clearance highly depends on the ability of a trainee to perform the task. In this paper, we propose a new algorithm for suggesting a path for a navigation task, that considers *Proficiency Level* of the trainee and provides a path according to how well a trainee can perform. The suggested approach is based on Probabilistic Roadmap Planners (PRM) and focuses on certain configuration spaces where most planners may fail to find a path while guaranteeing path's clearance. The new method provides a way to compensate for clearance constraints in regions wherever such compensation is necessary. Finally, the simulation in a surgical trainer demonstrates the effectiveness of the method.

Keywords: safe path planning, laparoscopic training, probabilistic roadmap planners, path repairing

1 INTRODUCTION

Path planning is one of the most researched areas in the field of robotics. Autonomous systems, either mobile robots or manipulators, essentially need a mechanism to generate a path for various activities like exploration, human-robot interaction, surgical operations, etc. Although, “*safe path*” is a subjective term that depends on the problem definition, as a baseline for safety, a path generated for any task should essentially be collision free in the configuration space.

According to the problem definition, researchers consider safety in various forms. Lacevic and Rocco (2010b) use the concept of *Danger Fields* described in Lacevic and Rocco (2010a) as the evaluation of safety. They focus on environments where humans and robots work together to ensure safety from a collision. Phillips and Likhachev (2011) introduce the concept of wait state for the mobile wheeled robots in the environment with dynamic obstacles where the robot waits by interpreting obstacle movement and move after the path is clear. Do et al. (2013) proposed a path planner for creating a safe margin, which essentially uses the concept of Support Vector Machines (SVMs) to generate a path for autonomous vehicles maintaining equidistant clearance from the obstacle or the boundary while traveling, that is, creating a path that tries to remain in the central region of the available free space. Though innovative, they only considered the environments where boundaries are symmetric, the approach is likely to create some unnecessary deviations in a path for asymmetric passages. Other approaches (Silver 2005;

Narayanan, Phillips, and Likhachev 2012; Van Den Berg, Ferguson, and Kuffner 2006) assumed the precomputation of the regions with high probability of collision with moving obstacles. They considered time as an additional dimensionality so path planning can be done according to the dynamics of obstacles in time and hence, avoiding passing through a region when there is a high probability of the presence of the obstacle. In general, safeness differs with problem definition, but fundamentally, safeness is introduced when there is incomplete knowledge of the environment, or imperfect robot dynamics. Hence the main aim for path planning is to avoid the collision in the presence of uncertainty and for this, it is important to keep a safe distance between a robot and the obstacles in the environment.

This problem of collision is more pronounced in the scenarios where there is incomplete information of the environment, or where humans are involved in motion and control of the robot. A perfect example where humans are responsible for instrument movement manipulation is Minimal Invasive Surgery (MIS), where a human surgeon has to operate using a few inches long and thin instruments, which are inserted using small incisions in the body. Various challenges in the procedure, described in Gallagher and O’Sullivan (2012) include minimal haptic feedback, single camera view with limited depth perception making hand-eye coordination difficult, limited flexibility of the instruments and most important, the instrument’s counter-intuitive movements. There are numerous tasks a surgeon has to perform in the operating room using a laparoscopic instrument, namely, suturing, object transfer, grasping, instrument navigation, etc. Moreover, with human factors involved, some imperfections are unavoidable. For example, a new MIS trainee might not be comfortable navigating instrument smoothly for the challenges described in MIS, until he or she practices on simulation trainers multiple times.

Imperfection in our context means that the task under consideration is not performed perfectly. We assume that by providing sufficient visual guidance for better depth perception and a precomputed recommended path for instrument navigation task, a trainee can learn and perform the task in optimal time. Moreover, we consider that the suggested instrument movements would be safe enough to avoid collisions with the environment if navigation is done perfectly. However, in the initial phases of training, due to inexperience with the instrument, a novice trainee can deviate from the suggested path, which introduces imperfection in task performance. In this paper, we focus on the instrument navigation task considering the *safety* in path planning based on *Proficiency Level* of the trainee and describe how different training scenarios can be generated in a given environment to help a trainee overcome the challenges inherent in the use of a laparoscopic instrument. It is important to note that collisions with the obstacles should always be avoided since they might have adverse effects, like internal bleeding.

Along with providing the importance of a *safe path*, we investigate a special case where general path planners, as described in LaValle (2006), may fail to provide a safe path and propose a new method that modifies a popular sampling-based path planner called Probabilistic Roadmap Planner (Kavraki et al. 1996) to solve this problem. The proposed method provides a path which is termed as *safe* in all possible regions. This paper is arranged as follows: the following section introduces the problem definition, then the path planning problem is formulated and a new approach is defined, after that, we present the simulation results performed using the simplified instrument configuration of the Computer-Assisted Surgical Trainer (CAST) (Rozenblit et al. 2014). Finally, we end with a conclusion and discussion of the future research scope.

2 PROBLEM DEFINITION

The major challenges of MIS are the difficulty of accurate instrument movement, which might, in the actual operating room lead to injuries such as internal bleeding if an instrument collides with the organs. Surgeons get better in the task of instrument navigation with the experience of performing surgery in the operating room. However, novice trainees have to practice a lot in simulated environments before they move to actual surgery in the operating room. No simulation is perfect for training, but when compared to most of the laparoscopic trainers (Ordóñez et al. 2007; Pham et al. 2005; Stylopoulos et al. 2004; Jaber 2010), CAST provides visual guidance (Wagner and Rozenblit 2017) for navigation that might be helpful

for the trainees to learn to operate and have better hand-eye coordination. CAST visualizes the recommended path for an instrument navigation task, but it does not consider trainee's *Proficiency Level* for path generation.

Consider a training scenario, where the recommended path for an instrument navigation task passes close to the obstacle but due to inexperience, there is a high possibility that a novice trainee would hit the obstacle while performing the task. Intuitively, it is better to have a path planner that considers trainee's *Proficiency Level* and generates tasks where paths are sufficiently distant from the obstacle, so that the possibility that trainee hits the obstacle decreases. Therefore, we model our safety criteria as the reasonable distance of the path from the obstacle; thus, collision possibility is minimized. The concept of Cumulative Danger Fields (Lacevic and Rocco 2010a) describes danger field at a point as a function of the position of a point and a velocity vector. For the purpose of this paper, we are only interested in the position of the point. By analysis in Lacevic and Rocco (2010a) it can be said that the farther a point is from the obstacle, the lower is the danger field. Moreover, *Proficiency Level* depends on how perfectly a trainee performs the instrument navigation task (Riojas et al. 2011), for instance, an experienced surgeon tends to deviate less from the suggested path as compared to a novice trainee and hence it can be said that the former has higher proficiency level than the latter.

However, there may be certain regions in the environment, where if the path is not allowed to pass close to the obstacle, most path planners may fail to return a path. Although, there is a *risk* while navigating through these regions, compensating safety just for these regions guarantees a recommended path, which is safe except when navigating close to obstacles that cannot be avoided. We provide a method to get a path in such scenarios. In the next section, we provide the stated method. We also show how it can be incorporated in CAST to enhance its capability of taking proficiency level of the trainee into account while creating training scenarios.

3 KEY ALGORITHM

3.1 Problem Formulation

In this sub-section, we briefly formulate the path planning problem and introduce some important terms and definitions that will be used throughout the paper. Let C be the configuration space, where, $C \in \mathbb{R}^d$, $d \in \mathbb{N}, d \geq 2$. The obstacle space is defined by C_{obs} and the obstacle-free region is defined by C_{free} , where $C_{obs} \subset C$, $C_{free} = C \setminus C_{obs}$ and $C = C_{obs} \cup C_{free}$. For sampling-based path planning algorithms, a path is found between the initial and goal states by sampling points in C_{free} , connecting them and finding a feasible set of connected samples that form a connection between the initial and goal points. The discrete set of samples is represented by X , where each sample $x_i \in X, i \in \mathbb{N}$ is sampled in the free region, C_{free} . A typical path planner is based on generating x_i iteratively and connecting them by *Edges* to find a path, with two or more nodes connected by edges. The sampling and connection procedures differ by application. Let $P[p_0, p_1, \dots, p_j, \dots, p_{n-1}] \rightarrow \mathbb{R}^d$ represent a path, where $p_j \in P, j \in [0, n-1]$, $j, n \in \mathbb{N}$, where p_j and $p_{j+1}, j \in \{0, n-2\}$, are connected by edges. Let x_{init} and x_{goal} be the initial and goal, where $x_{init}, x_{goal} \in X$, respectively. Moreover, for a path, $P, p_0 = x_{init}$ and $p_{n-1} = x_{goal}$.

Let us say that a path generated by the planner needs to maintain a minimum distance, $d_{thresh}, d_{thresh} \geq 0$, from the obstacle. For our purpose d_{thresh} depends on the *Proficiency Level* of the trainee; higher proficiency means less d_{thresh} . The value of $\square_{\square h} \square_{\square h}$ is application dependent parameter that is selected based on domain and geometric scale of the environment and d_{thresh} can be modeled simply by considering a *Virtual Boundary* of thickness d_{thresh} around an obstacle as shown in Figure 1(a). The *Euclidian Distance*, d of x_i from the nearest point of the nearest obstacle is termed *clearance*. Ideally, when connections are made between samples, edges should not pass through this virtual boundary and an alternate path should be found (Figure 1(a)). However, it is common to have environments where these virtual boundaries may block the path construction by separating the regions containing start and goal as

shown in Figure 1(b), we call these as *Critical Regions*. In these environments, most planners will fail to return the path if it is to be guaranteed outside the virtual boundaries. Since a path planner should provide a path, whenever one exists, it is better to allow planning through a virtual boundary (Figure 1(c)) just for *Critical Regions* and using original virtual boundaries for the rest of the environment.

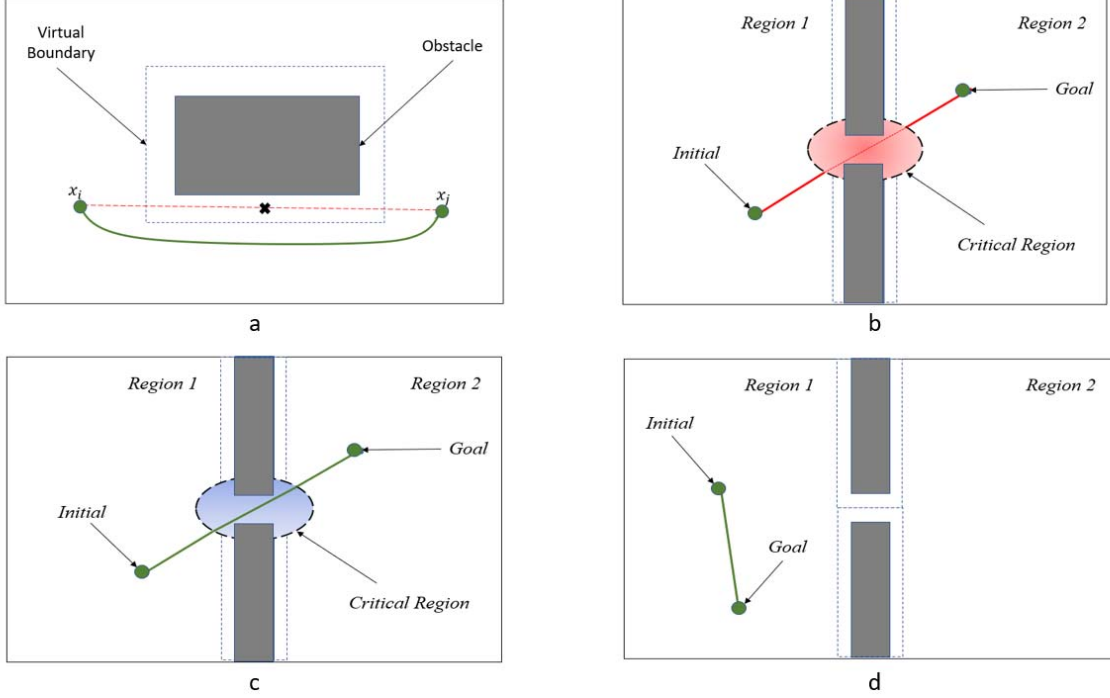


Figure 1: Path planning examples considering virtual boundaries and critical regions.

Additionally, note that depending on the locations of the initial and goal, *Critical Regions* might or might not be important, for example, if both are in the same region (Figure 1(d)), either Region 1 or Region 2, then there is no need to care for the *Critical Regions*. Moreover, scanning the whole environment to identify critical regions is an expensive process and would not be beneficial in some cases as shown in Figure 1(d). We now discuss the path planning to solve this problem.

3.2 Basic Path Planning

To the best of our knowledge, there exist no planner which solves the problem described above in Figure 1(b). To solve this problem, we define a sampling space according to clearance. Therefore, based on the virtual boundary, C_{free} can be divided into C_{safe} (space outside the virtual boundary) and C_{risk} (space inside the virtual boundary), $C_{safe} \cup C_{risk} = C_{free}$ and $C_{safe} \cap C_{risk} = \phi$. Therefore, x_i that lies in C_{safe} is added to X_{safe} , where $X_{safe} \subset X$ and those in C_{risk} are added to X_{risk} , where $X_{risk} \subset X$, $X_{safe} \cup X_{risk} = X$ and $X_{safe} \cap X_{risk} = \phi$. For the sake of explanation, let us call these sets samples from the *Safe Region* and the *Risk Region*, respectively. The main contribution of this paper is to provide a path planning method to generate a path in C_{safe} except for the regions where C_{risk} cannot be avoided.

There are various sampling-based algorithms for path planning but a popular choice for environments where it is required to maintain a fixed graph, the Probabilistic Roadmap (PRM) Planner (Kavraki et al. 1996) is a good choice. The reason for using a graph-based approach will be clear by the end of this section. PRM constructs a graph of samples with information of the edges and connected components, like a typical graph. The importance of connected components lies in the fact that a path between initial and goal is possible only if they lie in the same connected components space. For example, Figure 1(b)

has the initial and goal in different connected components (region 1 and region 2, respectively), while in Figure 1(d) they are in same one, that is Region 1. We would use PRM terminology of defining samples as *milestones* and graphs as *roadmaps*. Next, we present basic PRM in the original form and then propose a modification that can help find the proposed path.

Basic PRM is a simplified version of the planner described in Kavraki et al. (1996). This paper describes path planning in two phases, *Learning Phase* and *Query Phase*. The first phase is the *Learning Phase* (Algorithm 1) where a *Roadmap*, R , is constructed using milestones, $\forall x_i \in X$, connected with edges sampled from C_{free} . A set of nodes, N , which contains all milestones is maintained. Selection Strategy is a milestone selection procedure which provides a sorted list of Nearest Neighbors, N_c , from N based on distance d from newly added milestone. A new milestone x_i attempts to connect to every n milestones, $n \in N_c$, and successful valid direct connection adds an edge to the existing set of edges, E and update the R 's connected component. Update of R 's connected component is an important step since the new milestone can combine two or more connected components into one, which increases the probability of having the initial and goal points in the same connected component, an essential condition for finding a path. *Learning Phase* procedure provides R , which helps execute the second phase, that is, the *Query Phase* (Algorithm 2). This procedure adds x_{init} and x_{goal} to the roadmap and tries to find a connecting path, P . Since a path is essentially a set of milestones with a connection between them using edges, x_{init} and x_{goal} should lie in the same connected component to guarantee a path.

Algorithm 1: Learning phase pseudocode.

Procedure Roadmap :	Procedure addmilestone (c, R)
$N \leftarrow \phi$	$N_c \leftarrow \text{selectionStrategy}(c, R)$
$E \leftarrow \phi$	$N \leftarrow N \cup c$
$R \leftarrow (N, E)$	<i>foreach</i> $n \in N_c$ <i>sorted by increasing</i> $d(c, n)$
<i>loop</i>	<i>if</i> (validConnection (c, n))
$x_i \leftarrow \text{sample}(C_{free})$	$E \leftarrow E \cup \{(c, n)\}$
$R \leftarrow \text{addmilestone}(x_i, R)$	update R 's Connected Component
<i>return</i> R	<i>return</i> R

Algorithm 2: Query phase pseudocode.

Procedure Query (x_{init}, x_{goal}, R)
$P \leftarrow \phi$
$R \leftarrow \text{addmilestone}(x_{init}, R)$
$R \leftarrow \text{addmilestone}(x_{goal}, R)$
<i>if</i> (sameComponent (x_{init}, x_{goal}))
$P \leftarrow \text{getPath}((x_{init}, x_{goal}))$
<i>return</i> P

Additionally, analysis done by Karaman and Frazzoli (2011) and Yang et al. (2016), divides PRM's selection strategy into three major forms, namely K, S, and K-S. It should be noted that in every strategy, a set N_c , which is sorted based on increasing d from the newly added milestone provides a set of milestones to which the new milestone attempts to connect. K-strategy selects the nearest k , $k \in \mathbb{N}$, milestones for the set N_c . Although this strategy has an advantage of adopting enough milestones to ensure smoothness by adjusting the parameter k , it has a bias towards the denser region of sampling space. Moreover, since there is no bound on the length of the edge, distant connections are possible, which provides limited information about the intermediate space configuration of the path. This information might be useful, like in our case. The S-strategy uses a parameter s , $s \in \mathbb{R}$, which defines a distance bound inside which milestones are selected for N_c . This overcomes some shortcomings of K-

strategy, by choosing all milestones within a specified radius and limiting the maximum distance between two different milestones which ensures more information about the path, but this comes with reduced smoothness and increased computation time if the sampled space is dense (since there can be many milestones when space becomes dense). Finally, K-S-strategy is an adaptable S-strategy, which has the flexibility of selecting k and s as described above, helping to tune smoothness, guaranteeing all direction connections, reducing the computational burden and providing more information about the path. Selections of the value of parameters k and s are application dependent.

We are interested in having more information about the path, thus K-S-strategy is a good choice. The reasons would become clear towards the end of the Section 3. Now, we describe a method to enhance the path so that the planner can provide a path in the situations described in Section 3.1.

3.3 Modified Planner

Given an environment, a planner should provide a recommended path, but if it does not meet the specified criteria for clearance, then few portions of the generated path might be “repaired” to get a safer path. Hence, in this sub-section, we propose a modified PRM (Algorithm 3) to meet such requirements. For this, in addition to developing one roadmap, R , as in basic PRM where milestones are sampled from C_{free} , we maintain an additional roadmap R_{safe} , where milestones are sampled only from C_{safe} . First, x_{init} and x_{goal} are checked if they lie in the valid space, C_{safe} . If so, then the two mentioned Roadmaps are constructed. Let us assume that x_{init} and x_{goal} are in the *Safe Region*. After adding x_{init} and x_{goal} to R , and a *Query* is made in R , which returns a path, if one exists (Figure 2(a)). The path may pass through *Safe Region* and/or *Risk Region*, so we need to inspect the path if its portion is inside the risk region. For this, we should have sufficient information about the path, ideally, the milestones constructing the path and the C_{free} should be infinite, $n \rightarrow \infty$. This is computationally impractical, to simplify, we use the K-S-strategy, which can tune k and s parameters for adjusting the computation time for Roadmap construction and maintain sufficient information about the path. Moreover, because of finite samples at a given time, sampling algorithms, like PRM, are conditionally Probabilistic Complete and Asymptotically Optimal (Karaman and Frazzoli 2011). The path generated at given query and configuration is considered the best path, although it might not be intuitively shortest. The path P is cached and if all milestones in the path are in R_{safe} , then the path is already safe and the algorithm returns this path, otherwise procedure PathRepair repairs P according to desired clearance.

Algorithm 3: Modified planner pseudocode.

Procedure **ModifiedPlanner**: ($x_{init}, x_{goal}, C_{free}, C_{safe}$)

$R, R_{safe} \leftarrow \phi$

if (not (**Valid**(x_{init}) or **Valid**(x_{goal})))

 return *invalidConfiguration*

loop

$x_i \leftarrow \text{sample}(C_{free})$

 if (x_i in C_{safe})

$R_{safe} \leftarrow \text{addmilestone}(x_i, R_{safe})$

$R \leftarrow \text{addmilestone}(x_i, R)$

$P \leftarrow \text{query}(x_{init}, x_{goal}, R)$

if (P exists)

 if (not $\forall p_j \in P$ in R_{safe})

$P \leftarrow \text{pathRepair}(P, R_{safe})$

 return P

return *noPath*

Procedure PathRepair (Algorithm 4) operates as follows: first, the milestones constituting the path are marked invalid if they are not in R_{safe} , as shown in red in Figure 2(b). The second step involves initiating the first valid milestone as start and next as goal (Figure 2(b)), then traversing through all the valid milestones to get the last vertex on the path in the same connected component, and update this as the goal (Figure 2(c)). After that *Query* between the new initial (p_l) and new goal (p_m) states provides a path in *Safe Region*, since both the initial and goal points are in the same component, a path is guaranteed. After getting a sub-path, the section of the original path from the new initial to new goal is replaced with the newly generated path (Figure 2(d)). In the next iteration, the milestone next to the goal is taken as the next initial milestone and the procedure is repeated until it reaches the end milestone in the path (Figure 2(e)). In the end, a repaired path is returned (Figure 2(f)). It is important to note that during path repair, a query is generated by only using the milestones in the safe region, the reconstructed sub-section of the path is guaranteed to be in the safe region.

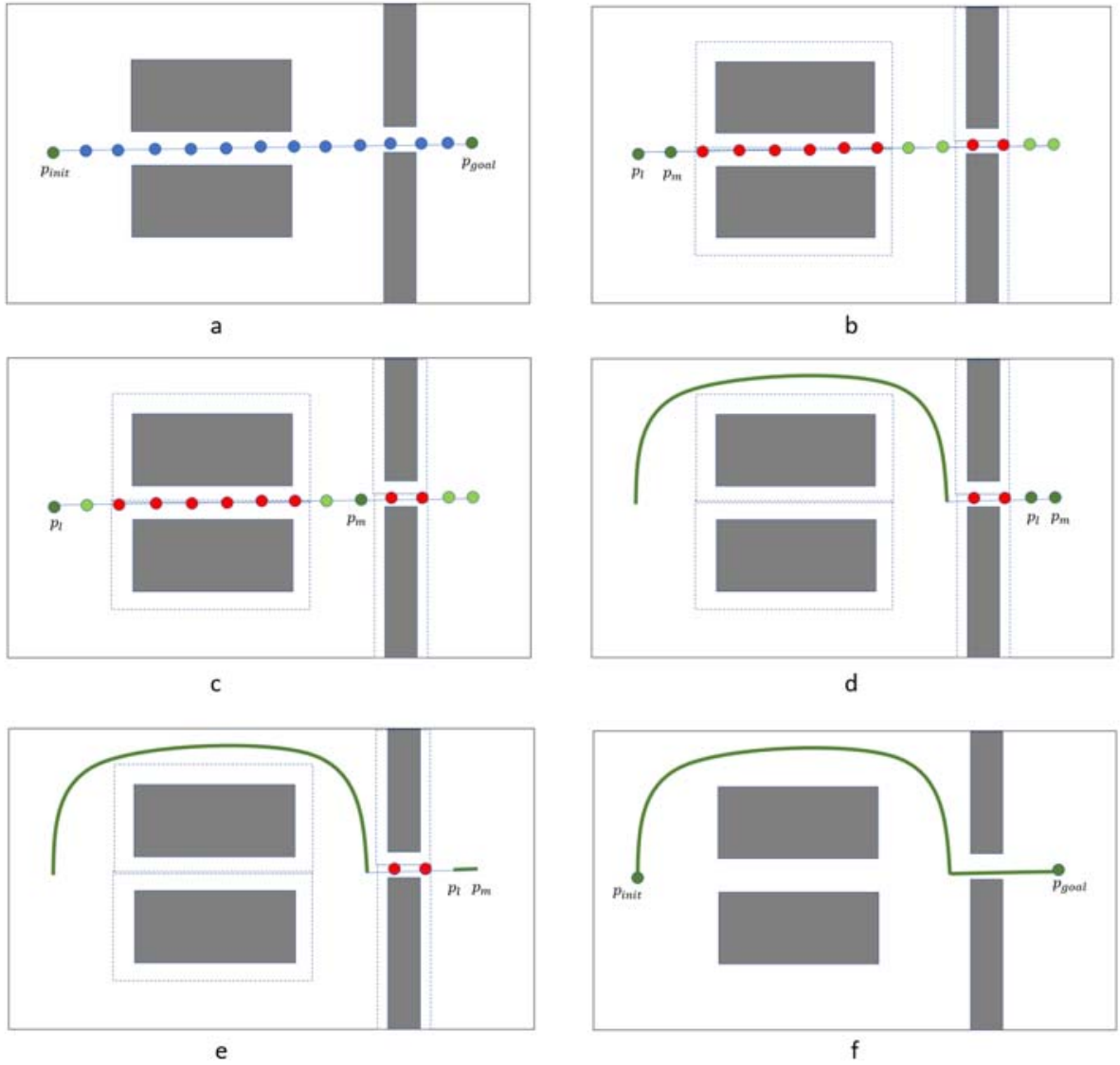


Figure 2: Illustration for the algorithm steps.

Algorithm 4: Path repair pseudocode.

```

Procedure PathRepair( $P, R_{safe}$ ):
 $I \leftarrow \forall p_i \in P \text{ not in } R_{safe}$            // where  $I$  is a set of invalid milestones
 $l \leftarrow 0$ 
while( $l < n - 1$ )
    while( $p_l \in I$ )
         $l++$ 
     $q \leftarrow l$ 
    while( $q < n - 1$ )
         $q++$ 
        if( $p_q \notin I$ )
            if(sameComponent( $p_l, p_q$ ))
                 $m \leftarrow q$ 
             $P'_{l \rightarrow m} \leftarrow \text{query}(p_l, p_m, R_{safe})$ 
            if( $P'_{l \rightarrow m} \neq \phi$ )
                 $P_{l \rightarrow m} \leftarrow P'_{l \rightarrow m}$ 
            update  $l, n$ 
return  $P$ 

```

Moreover, since the path cannot be repaired when checked against *Critical Regions*, it intuitively ignores the clearance constraint in these regions. Thus, as we are interested to repair portion of path that lies in one region, where regions are separated by either the actual obstacle or virtual boundaries, PRM uses concept of connected components to mark different regions (Figure 1(c)), whereas approaches not based on connected components, like Rapidly-Exploring Random Trees (Vinet and Zhedanov 2011) cannot distinguish between separated regions, hence, our choice of using PRM is a reasonable to make.

4 SIMULATION RESULTS

4.1 Experimental Setup

Our experimental setup uses CAST configuration, that has two laparoscopic instruments, left and right, mounted on two fixtures, each having a gimbal. To mimic a trocar in laparoscopic surgery, the gimbal center provides four degrees of freedom, namely, insertion, pitch, yaw, and roll, all centered around a single-entry point which corresponds to the incision. For the sake of simplicity of demonstrating the algorithm in action, we consider three degrees of freedom: insertion, pitch, and yaw. The instrument navigation task in CAST requires a trainee to follow a recommended path for task efficiency and collision minimization. The recommended path is generated based on the proficiency level of the trainee. The simulations are performed using Open Motion Planning Library (Şucan, Moll, and Kavraki 2012) and MATLAB is used for the visualization of path and environment. The simulated environment, shown in Figure 3(b) is carefully selected to show the intuitive distinctions in the paths for different proficiency levels. The environment is a 3D model with the actual constraints on pitch, yaw, and insertion as in CAST. For demonstrating results, only the right instrument is used and, we consider three proficiency levels of the trainee, let us call them Beginner, Intermediate, and Expert, where beginner tends to have maximum deviation from the path navigation task, therefore maximum d_{thresh} , while an expert is expected to perform the task almost perfectly, hence minimum d_{thresh} . In other words, a beginner is provided most virtual boundary and for expert, there is no virtual boundary, that is, the path in entire C_{free} is acceptable for an expert.

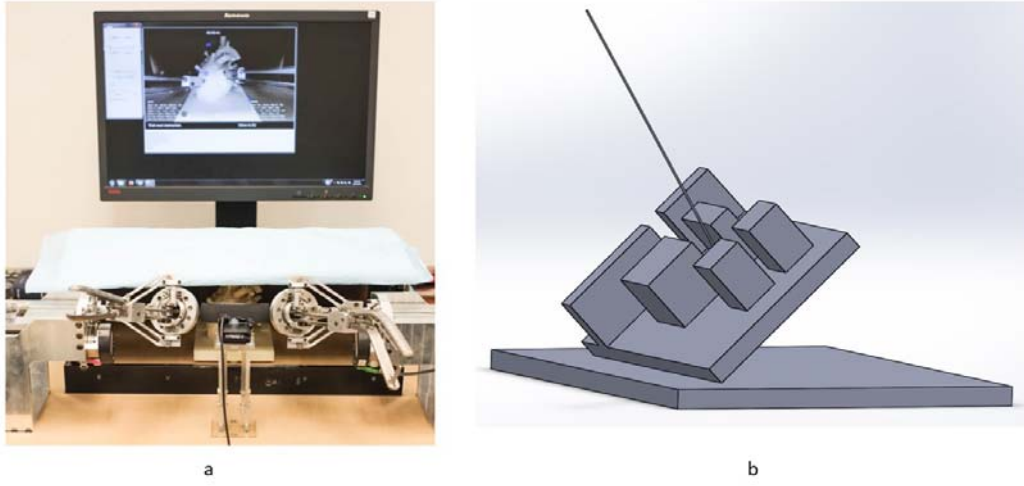


Figure 3: (a) CAST, and (b) model used for simulation.

4.2 Results

Figure 4 shows the results performed in the environment ($150 \times 100 \times 40 \text{ mm}$) shown in Figure 3(b). As described, the value of d_{thresh} depends on application and geometric scale of the environment, hence for demonstrating the proof-of-concept, value of d_{thresh} is selected according to environment dimensions, though other values can be selected. Path length is selected as optimization criteria. Figure 4 shows the simulation results performed for three levels of proficiency.

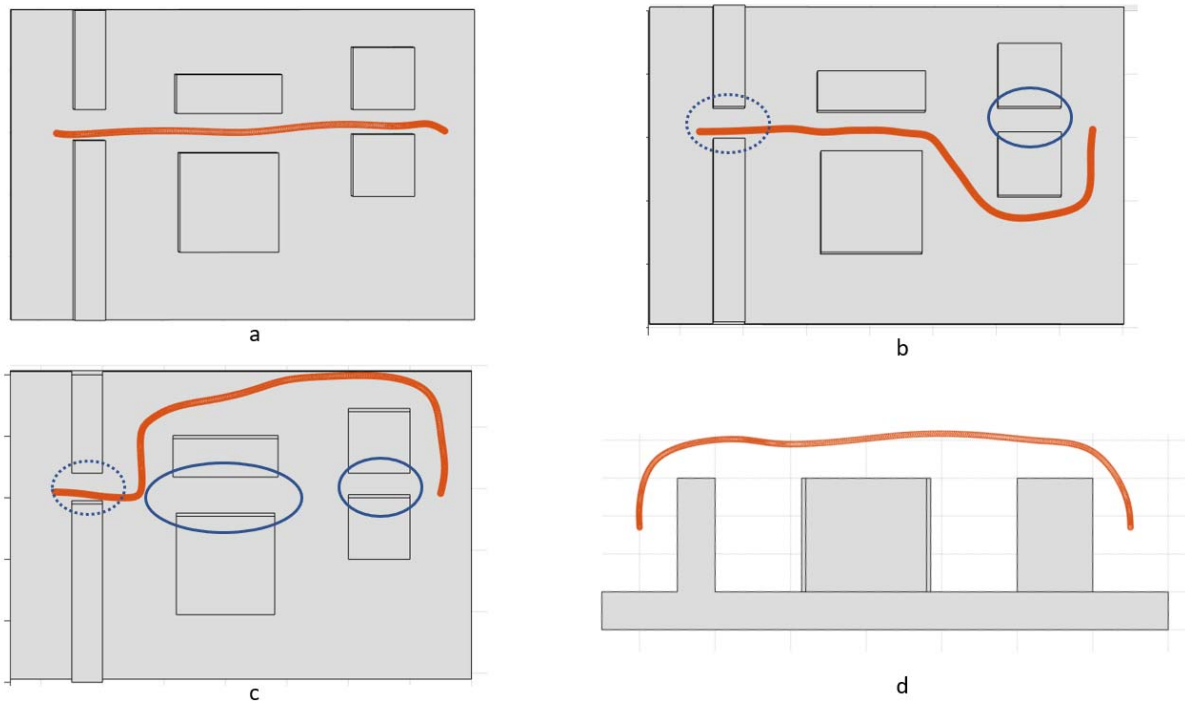


Figure 4: Simulation results: solid oval - region where detour is possible, dashed oval - critical region.

Figure 4(a) illustrates a recommended path for an expert; it considers no virtual boundary ($d_{thresh} = 0$ mm), because of no expected deviation. The path shown in Figure 4(b) is suggested for an intermediate trainee, hence it considers a virtual boundary ($d_{thresh} = 4$ mm) around the obstacle. Though not depicted in the figure, d_{thresh} creates a virtual boundary that blocks the passage between the right-most (solid oval marking) and the left-most pair (dashed oval markings) of obstacles. The traditional planners in this case would fail because of no possible detour due to blockage of the left-most region, which is a critical region. However, due to possibility of detour around the right-most pair of obstacles, the proposed planner finds an alternative path and allow planning through the left-most region as there was no alternate path. Thus, the path repairs for the regions where detour is possible but allow passing through the critical regions. Figure 4(c) demonstrates the same concept but with increased thickness of the virtual boundaries ($d_{thresh} = 8$ mm) blocking the passage between all three pair of obstacles. Therefore, like the previous case, it detours where an alternate path is possible. These simulations are performed by considering C_{free} in a bounding box ($150 \times 100 \times 40$ mm) fixed to the environment's dimensions, but for the beginner, if we consider extending the dimensions (i.e., bounding box change to $(150 \times 100 \times 80$ mm)), a path that guarantees d_{thresh} is obtained due to planning entirely in safe region as shown in figure 4(d). This path is like any planner with guaranteed d_{thresh} would have provided. Hence, this demonstrates that the proposed method adapts well to the traditional planner if the space allows entire planning in safe region, otherwise, it provides repaired path as shown in simulations.

5 CONCLUSION AND DISCUSSION

In this paper, we introduced a new algorithm based on PRM. The new method takes the proficiency level of a trainee in laparoscopic surgery to provide an instrument navigation task given a training environment. Since an expert is expected to perform the navigation task better than a beginner, the algorithm provides an alternative path for the beginner, so that the probability of collision with the obstacles is reduced. The proposed algorithm provides a path which avoids entering risk regions in the free space except the critical regions, as these are necessary to navigate to compute a path for the navigation task. In other words, the proposed method allows planning through critical regions by ignoring the distance bounds in these regions and not in rest of the space.

However, there are a few important things to notice. First, it should be noted that since the modification is based on repairing a path when it lies inside the Risk Regions, therefore, there should be at least two milestones in the safe region for the repair to take place. In other words, if the path completely lies in the risk region, a repaired path cannot be computed, and the algorithm provides a path which is completely inside the risk region. The algorithm in this case, that is the worst-case, works like a traditional planner without taking virtual boundaries into consideration. Alternatively, if the path completely lies in the safe region, for example, the configuration shown in Figure 1(d), then no repair is required, and the path is suited for all proficiency levels. Hence, it is difficult to annotate the path as an expert or beginner path, as path may or may not always lie completely in the safe region for a particular query. Moreover, in Section 3.3, we assumed that the initial and goal milestones should be inside the safe region, but this assumption can be removed by projecting the initial and/or goal state if they are inside the risk region to the safe region using an incremental sphere around them and selecting the nearest sample which lies inside the safe region. The projected and original samples can be connected for the final path.

We used this method specifically for medical applications in Minimally Invasive Surgery, but it is equally applicable to other systems involving path planning problem for navigation. For instance, the method can be used to plan a path for autonomous quadrotor when it is required to maintain a certain distance from the obstacle, but due to the presence of some unavoidable passages, like *Critical Regions*, it might fail. However, it must compute a path which maintains a required distance except while passing through critical regions. Moreover, *Proficiency Level* can be modeled according to the application. Finally, the present research suggests further improvement. Due to the use of the K-S strategy, the quality of the path is not sufficiently smooth, therefore for future work, we will focus on the quality of the path. Moreover,

we consider virtual boundaries to construct the safe regions, but some paths, due to *Visibility* property of PRM, may cross virtual boundaries around the corners, making a small section of the path occasionally closer than expected. Additionally, for perfectly avoiding the virtual boundaries, infinite milestones would be required which is not practically possible. Therefore, exploring how to avoid such scenarios is another research scope. Moreover, a human study would be important to verify if the proposed method would actually be helpful to assist laparoscopic trainees in overcoming difficulties of MIS surgery.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant Number 1622589 “Computer Guided Laparoscopy Training”. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

- Berg, J. V. D., D. Ferguson, and J. Kuffner. 2006. “Anytime Path Planning and Replanning in Dynamic Environments.” In *Proceedings - IEEE International Conference on Robotics and Automation*, 2366–71. doi:10.1109/ROBOT.2006.1642056.
- Do, Q. H., H. T. N. Nejad, K. Yoneda, S. Ryohei, and S. Mita. 2013. “Vehicle Path Planning with Maximizing Safe Margin for Driving Using Lagrange Multipliers.” In *IEEE Intelligent Vehicles Symposium, Proceedings*, 171–76. doi:10.1109/IVS.2013.6629466.
- Gallagher, A. G., and G. C. O’Sullivan. 2012. *Fundamentals of Surgical Simulation: Principles and Practice*. Springer Science & Business Media. doi:10.1007/978-0-85729-763-1.
- Jaber, N. 2010. “The Basket Trainer: A Homemade Laparoscopic Trainer Attainable to Every Resident.” *Journal of Minimal Access Surgery* 6 (1): 3–5. doi:10.4103/0972-9941.62525.
- Karaman, S., and E. Frazzoli. 2011. “Sampling-Based Algorithms for Optimal Motion Planning.” *The International Journal of Robotics Research* 30 (7): 846–94. doi:10.1177/0278364911406761.
- Kavraki, L. E., P. Švestka, J. C. Latombe, and M. H. Overmars. 1996. “Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces.” *IEEE Transactions on Robotics and Automation* 12 (4): 566–80. doi:10.1109/70.508439.
- Lacevic, B., and P. Rocco. 2010a. “Kinetostatic Danger Field - A Novel Safety Assessment for Human-Robot Interaction.” In *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 2169–74. doi:10.1109/IROS.2010.5649124.
- Lacevic, B., and P. Rocco. 2010b. “Towards a Complete Safe Path Planning for Robotic Manipulators.” In *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 5366–71. doi:10.1109/IROS.2010.5650945.
- LaValle, S. M. 2006. *Planning Algorithms*. Cambridge University Press.
- Narayanan, V., M. Phillips, and M. Likhachev. 2012. “Anytime Safe Interval Path Planning for Dynamic Environments.” In *IEEE International Conference on Intelligent Robots and Systems*, 4708–15.
- Ordóñez, P., P. Kodeswaran, V. Korolev, W. Li, O. Walavalkar, B. Elgamil, A. Joshi, T. Finin, Y. Yesha, and I. George. 2007. “A Ubiquitous Context-Aware Environment for Surgical Training.” In *Proceedings of the 4th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, MobiQuitous 2007*, 1–6. doi:10.1109/MOBIQ.2007.4451029.
- Pham, T., Roland L., K. A. Benson, R.W. Webster, and A.G. Gallagher 2005. “Smart Tutor: A Pilot

- Study of a Novel Adaptive Simulation Environment.” In *Studies in Health Technology and Informatics*, 385–89.
- Phillips, M., and M. Likhachev. 2011. “SIPP: Safe Interval Path Planning for Dynamic Environments Goal.” In *Proc. of the Int. Conf. on Robotics and Automation (ICRA)*, 5628–35.
- Riojas, M., C. Feng, A. Hamilton, and J. Rozenblit. 2011. “Knowledge Elicitation for Performance Assessment in a Computerized Surgical Training System.” *Applied Soft Computing Journal* 11 (4): 3697–3708. doi:10.1016/j.asoc.2011.01.041.
- Rozenblit, J. W., C. Feng, M. Riojas, L. Napalkova, A. J. Hamilton, M. Hong, P. Berthet-Rayne, et al. 2014. “The Computer Assisted Surgical Trainer: Design, Models, and Implementation.” In *Summer Computer Simulation Conference*, 211–20. doi:10.1145/2254556.2254608.
- Silver, D. 2005. “Cooperative Pathfinding.” In *Proceedings of the First Artificial Intelligence and Interactive Digital Entertainment Conference*, 117–22. doi:10.1016/j.ydbio.2008.06.039.
- Stylopoulos, N., S. Cotin, S. K. Maithel, M. Ottensmeyer, P. G. Jackson, R. S. Bardsley, P. F. Neumann, D. W. Rattner, and S. L. Dawson. 2004. “Computer-Enhanced Laparoscopic Training System (CELTS): Bridging the Gap.” *Surgical Endoscopy* 18 (5): 782–89.
- Şucan, I. A., M. Moll, and L. Kavraki. 2012. “The Open Motion Planning Library.” *IEEE Robotics and Automation Magazine* 19 (4): 72–82. doi:10.1109/MRA.2012.2205651.
- Vinet, L., and A. Zhedanov. 2011. “Rapidly-Exploring Random Trees: A New Tool for Path Planning.” *Journal of Physics A: Mathematical and Theoretical*. doi:10.1135.1853.
- Wagner, A., and J. W. Rozenblit. 2017. “AUGMENTED REALITY VISUAL GUIDANCE FOR SPATIAL PERCEPTION IN THE COMPUTER ASSISTED SURGICAL TRAINER.” In *Proc. of the 2017 Spring Simulation Conference, Modeling and Simulation in Medicine (SpringSim-MSM 2017)*, 855–66.
- Yang, L., J. Qi, D. Song, J. Xiao, J. Han, and Y. Xia. 2016. “Survey of Robot 3D Path Planning Algorithms.” *Journal of Control Science and Engineering*, 1–22. doi:10.1155/2016/7426913.

AUTHOR BIOGRAPHIES

SHUBHAM JAIN is a master’s student in Electrical and Computer Engineering Department at the University of Arizona. He currently works in Model Based Design Laboratory on Robotic Path Planning and applications of path planning in surgical instruments. He received his B.S. in Electronics and Instrumentation Engineering from Uttar Pradesh Technical University, Lucknow, India. His email address is shubhamjain@email.arizona.edu.

MINSIK HONG is a Ph. D. candidate at the University of Arizona. He received a Master of Science degree in Electrical and Computer Engineering from POSTECH, Republic of Korea. His research interests are robotics, control system, fuzzy theory, and modeling and simulation for medical devices. His email address is mshong@email.arizona.edu.

JERZY W. ROZENBLIT is University Distinguished Professor, Raymond J. Oglethorpe Endowed Chair in the Electrical and Computer Engineering (ECE) Department, with a joint appointment as Professor of Surgery in the College of Medicine at the University of Arizona. During his tenure at the University of Arizona, he established the Model-Based Design Laboratory with major projects in design and analysis of complex, computer-based systems, hardware/software codesign, and simulation modeling. He presently serves as Director of the Life-Critical Computing Systems Initiative, a research enterprise intended to improve the reliability and safety of technology in healthcare and life-critical applications. His email address is jr@ece.arizona.edu.