The Fourth IEEE RAS/EMBS International Conference
on Biomedical Robotics and Biomechatronics
Roma, Italy. June 24-27, 2012

Risk-Based Path Planning for a Steerable Flexible Probe for Neurosurgical Intervention

Chiara Caborni, Seong Young Ko, Elena De Momi, Giancarlo Ferrigno and Ferdinando Rodriguez y Baena

***Abstract*—Minimally invasive surgery has become increasingly common in neurosurgical intervention. Recently, various types of flexible needles have been developed to reach deep targets while avoiding important anatomical regions. In a planning stage, it is required to compute a curvilinear trajectory from an entry point on the skull of the patient to the target lesion. We present a path planner that provides a feasible path for a steerable flexible probe which takes into account its mechanical properties and soft tissue morphology surrounding the target to produce a "path of least risk" for the patient. To meet such requirements, a well known Rapidly-exploring Random Trees (RRT)-based method is adopted since it is capable of producing a curvature-constrained path in complex brain risk maps. To alleviate the sub-optimality of RRTs, a multiple growth of trees that ensures several solutions is introduced so that the clinician can evaluate and select a path that minimizes the cost of different weighted competing objectives, such as the length of the path, the clearance from vessels or nervous bundles and, most importantly, the accumulated risk in crossing different brain regions. Simulations using the risk map of a brain's coronal slice and experimental tests using the path planner integrated with the flexible probe, showed that the approach is promising and that the main objectives of the planning method have been achieved.**

***Index Terms*—Path Planning, Non-holonomic Systems, Medical Robotics, Flexible Probe, Rapidly-exploring Random Trees, Curvature Constraint, Brain Risk Map**

I. Introduction

KEY hole neurosurgery has become the standard in many procedures such as biopsy, placement of electrodes for deep brain stimulation and cancer treatments, since it can reduce damage to the patient. Rigid linear probes can reach deep targets accurately, but cannot always avoid very important anatomical brain regions. The recent introduction of flexible needles overcomes this problem. Several prototypes have been developed: a relatively stiff needle, flexed by a moment applied by an external base [1], a thin bevel-tip flexible needle [2], a nested cannula system that includes a set of pre-bent concentric tubes [3] and a multi-segment steerable flexible probe STING [4].

A computational method is needed to plan the path of the needle to accurately reach the target with the minimum damage to the patient. To use these flexible needles, the planning algorithm should be able to find the curvilinear path that safely reaches the target, such as a tumour or a lesion, minimizing a cost function associated with the amount of penetrated tissue i.e. the damage to the patient.

This work presents a path planner for flexible probes, with a focus on the neurosurgical flexible probe currently under development at Imperial College London, UK [4]. Since the brain consists of a number of regions, functionality and risk level of which differ, the safety, in terms of the total length of the path, the accumulated risk along the path, and the clearance from dangerous areas, needs to be considered in the path planning phase. In addition, the constraint on the maximum curvature, arising from the unique mechanism of the motion of the flexible probe, needs to be considered. This work, therefore, describes a path planner able to satisfy the kinematic constraints of the probe while minimizing risk to the patient.

II. Research Background

A. Bio-inspired Flexible Probe

The steerable flexible probe codenamed STING is capable of accessing deep lesions in soft tissues while avoiding critical structures such as vessels, motor or sensory areas, by allowing the surgeon to take a "roundabout" route to the target. STING consists of four independent segments which interlock together through a dovetail mechanism which allows sliding between the parts [4]. The probe can then be steered along any predefined trajectory within a plane by means of the so-called programmable bevel as shown in Fig. 1: the approach angle of the probe tip is found to be a function of the offset between segments, with a larger offset resulting in a tighter curve [5].

The requirement to constrain the minimum radius of curvature on the path, r_{min}, is caused by mechanical limits of the flexible probe (i.e. the maximum offset between the segments). The relationship between the curvature and the steering offset was established by performing a set of experiments with a 12 mm outer diameter, scaled-up prototype. Based on these results, the curvature ρ has been assumed to

This work was supported by the EU-FP7 Project ROBOCAST (FP7-ICT-215190) and has also received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement n° [258642-STING].

C. Caborni, E. De Momi, G. Ferrigno are with the Department of Biomedical Engineering, Politecnico di Milano, 20133, Italy, (e-mail: chiara.caborni@mail.polimi.it, elena.demomi@polimi.it and giancarlo.ferrigno@polimi.it).

S. Y. Ko was with the Department of Mechanical Engineering, Imperial College London, SW7 2AZ, UK, and is now with the School of Mechanical Systems Engineering, Chonnam National University, Gwangju, 500-757, South Korea, (e-mail: sko@chonnam.ac.kr.)

F. Rodriguez y Baena is with the Department of Mechanical Engineering, Imperial College London, SW7 2AZ, UK (corresponding author, phone: +44-(0)20-7594-7046; fax: +44-(0)20-7594-1472; e-mail: f.rodriguez@imperial.ac.uk).

978-1-4577-1200-5/12/$26.00 ©2012 IEEE

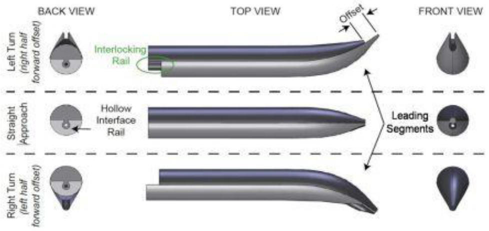


Figure 1. Schematic of the programmable bevel tip concept. An offset of two parts of the flexible probe determines the steering amount and direction of the tip. The offset is proportional to the curvature of the path [5].

be proportional to the steering offset d with a coefficient κ = 0.000185 mm^{-2}, i.e. $\rho = \kappa \cdot d$.

B. Curvilinear Path Planning

The robot's position is represented by a point in 2D-space; the path planning solution is represented by the definition of a feasible trajectory for the robot to reach the target from a starting point.

Various approaches, originally developed for the path planning of mobile robots, have been implemented to find a path for the aforementioned needles: these include potential fields [6], deterministic sampling-based searches such as A* [7], level-set-based methods such as Fast Marching [8] and probabilistic sampling based searches such as Rapidly-exploring Random Trees (RRTs) [7]. However A* cannot generate smooth paths, and potential fields methods and most probabilistic and deterministic methods cannot deal with the non-holonomicity of flexible needles.

Two approaches that can deal with constraints on the curvature of the path have been identified in literature: a deterministic one, a Fast Marching algorithm (FM) proposed by Pètres [9], and a probabilistic one, an RRT-based method, proposed by Patil and Alterovitz [10].

The FM algorithm finds the minimum-cost path from the entry to the target point. FM has been used for path planning of underwater autonomous robots [9]. Although Cohen and Kimmel [11] suggested a relationship between the minimum radius of curvature of the extracted path and the input map's features, the minimum theoretical bound of the relationship is too conservative to estimate the real curvature of the final path.

Contrary to this, RRT-based methods can generate the final path, the curvature of which is limited less conservatively and the computation time of which is less. Xu et al. [12] utilized RRT-based methods to find the path for a bevel-tipped flexible probe, which includes arcs of fixed radius of curvature. Following this, Patil et al. [10] utilized a modified version of the basic RRTs method, named Reachability-Guided RRTs (RG-RRTs), in order to speed up the search and to improve convergence. They introduced the concept that the curvature of the needle is variable but bounded and that the randomly sampled position needs to be reachable from a pre-identified configuration. The time with which the RRTs algorithm computes a solution is dependent on the environment, as well as on the randomness of the search.

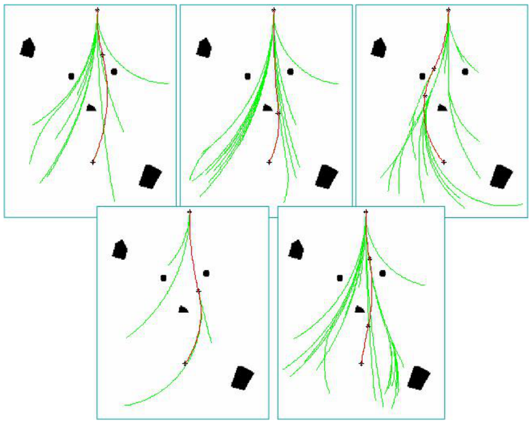


Figure 2. Several outputs of the same planning problem: each trial produces different RRTs. The tree is depicted in green while the final path is in red. The intermediate black star-shaped points indicate the nodes of RRTs, between which the arcs of different radius are connected and compose the path.

Since the algorithm has a probabilistic nature, given the same planning problem, the same RRTs routine can produce different results, as shown in Fig. 2. This sub-optimal characteristic of RRT-based path planning methods provides an opportunity for further optimisation, especially in neurosurgery, where the optimum path is highly influenced by the requirements associated to patient safety. Among the candidate solutions, all of which have already met the required constraints, the final path can be chosen based on a cost function. In addition, the surgeon can evaluate and select the best trajectory in terms of the length of the path, risk accumulated in crossing different regions and clearance from vessels or nervous bundles. A multiple growth of RRTs is implemented here in order to achieve this purpose.

III. Curvature Constrained Path Planning considering Risk Brain Maps

Because the RRT-based algorithm is shown to alleviate the heavy computation in the path planning [7] and proved to be efficient in constraining the minimum radius of curvature of the generated trajectory [10], this method is chosen to compute the path for our neurosurgical flexible probe. Since the current control algorithm for the probe has been implemented in two dimensional (2D) space, the methods proposed in this paper are also contrained to 2D.

A. RG-RRTs methods

This section explains RG-RRTs based path planning in brief [10]. The main difference of RG-RRTs with respect to conventional RRTs-based search [7] is the strategy employed to sample random points and connect nodes of the tree. As a result, the tree expands by adding arcs of circles with bounded curvature ($\rho < 1/r_{min}$).

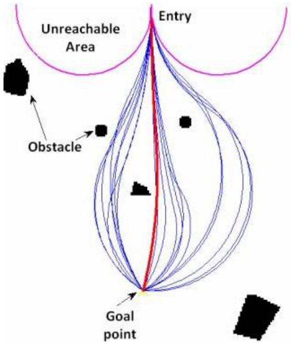


Figure 3. Semi-optimal results of RG-RRT algorithm.

Furthermore, in contrast to standard RRTs, with RG-RRTs, the sensitivity to the distance metric used to compute the next candidate node in the tree is improved. The random point, p_{rand}, sampled over the space, becomes directly the next candidate node in the tree, while in standard RRTs the next node determines only the direction in which the next candidate is going to lie. Therefore, when p_{rand} is sampled over the space, it is checked to see if it is reachable (i.e. tangentially connectable with an arc of bounded curvature) from at least one of the configurations q_{old}:(x_n, y_n, θ_n) already stored in the tree. In other words, the random point is discarded if, after converting it into the local frame $\{L\}$ of q_{old}, it does not belong to the region defined locally by:

$$y_r \geq \sqrt{2r_{min}\,|x_r| - x_r^2} \tag{1}$$

where r_{min} is the minimum radius constraint for probe steering and (x_r, y_r) is the random point's position in the $\{L\}$ frame. In Fig. 3, the magenta lines display the unreachable region in the first iteration step, while the blue lines display the final paths once RRTs-based searching was performed 20 times.

If the random point lies in the reachable region of the configuration q_{old} defined in (1), an arc connecting q_{old} and p_{rand} needs to be computed and stored as an edge in the tree. The arc's exact parameters $C : (r, \phi)$ include the angle ϕ subtended by the arc and the arc radius r. By expressing p_{rand} in the local frame $\{L\}$ of q_{old}, the parameters can be computed as follows:

$\phi = \pi - 2 \cdot sin^{-1}(\frac{^{\{L\}}x_{rand}}{d}) \equiv \pi - 2 \cdot \varepsilon$

$r = d \cdot \frac{sin(\varepsilon)}{sin(\phi)}$

where $d = |\overrightarrow{p}_{rand} - \overrightarrow{p}_{old}|$.

Combining the orientation $^{O}\theta_n$ of q_{old} with ϕ, it is possible to find the orientation of the probe's tip, $^{O}\theta_{new}$, in the new configuration $q_{new} : (^{O}x_{rand}, ^{O}y_{rand}, ^{O}\theta_{new})$ in the global frame $\{O\}$.

B. Risk-Based Optimization

The path found is sub-optimal because it is only one of the possible solutions for the given planning problem, based on the random nature of samples aquisition.

To improve upon this sub-optimal performance of the method, a multiple search strategy has been implemented and a criterion to choose the best path while accomplishing initial safety parameters α, β, γ, has been defined as following:

$$Cost_i = \alpha \cdot \frac{\lambda_i}{max(\lambda)} - \beta \cdot \frac{\Upsilon_i}{max(\Upsilon)} + \gamma \cdot \frac{\Delta_i}{max(\Delta)} \tag{2}$$

where $Cost_i$ refers to the ith-path and the normalised weights α, β and γ, which vary from 0 to 1 and for which their sum is 1. Three important criteria to compute the cost function are chosen: the overall length of the path λ_i, the clearance from no-go areas Υ_i and the accumulated risk along the path, $\triangle_i$. In neurosurgery, the length of the path needs to be minimized to reduce the amount of brain tissue to be traversed, clearance from obstacles needs to be maximised to reduce the possibility of the probe intersecting a no-go area due to unexpected errors and finally, the accumulated risk needs to be minimized to identify the path which reaches the target at minimum risk to the patient (i.e. while traversing "low risk" areas as opposed to "higher risk areas" when possible). Thus, the best path is the one with the minimum cost, $min(Cost)$. The clearance from vessels and no-go areas that are optional and patient-specific, is a hard constraint since any trajectory crossing these areas will be automatically discarded. The accumulated risk along the path is rather a soft constraint in the sense that the trajectory segments are just weighted according to the risk associated to the areas it intersects. The last two components of the cost function have thus been considered separately.

The multiple growth, which can provide a number of trees no larger than the input parameter $MaxTrees$, has been developed and its workflow is reported in Algorithm 1. Once each tree is initialized with the entry configuration q_{init}, i.e. the entry point and the orientation of the probe's tip, a random point over the free configuration space C_{free} is sampled according to the routine *Random_Free_State(goal_bias)*. This routine also checks for collisions: on a binary image or a risk-labelled brain map (e.g. vessels are "high risk", white matter is "low risk"), this is achieved by checking if the pixel in question is outside any no-go area dilated by a safe margin, as explained in Sec. IV. A goal bias strategy is adopted in the sampling procedure as it speeds up the search and improves the path's convergence to the goal [13]: is to select the goal position, p_{goal}, as the random candidate position on 20% of occasions, and for the remaining 80% allowing it to be selected randomly. In line 8, the random configuration, q_{rand}, is checked to see whether it is reachable from at least one configuration, q, among the available trees, which are defined as the trees that have not been completely searched yet. If q_{rand} is not reachable from any configuration of any tree, a new random point is sampled until at least one configuration in any tree can reach this point. Among the configurations q of each tree, from which the random point is reachable, the closest configuration $q_{near}(j)$, in terms of the standard Euclidean distance metric, is selected for each tree by the routine q_{near}*(j)=NearestNeighbor()* in line 15. Then, an ascending sorting operation is performed using $q_{near}(j)$ based on the distances ($q_{near}(j)$, q_{rand}) to select the nearest absolute point q_{near} to be connected toward the random one. The routine *ValidEdge()* in line 22 then checks whether or not the arc connecting the two configurations is safely far from the obstacles; this operation is repeated for $q_{near}(j)$ of each tree in the sorted list until the random point can be connected without crossing the no-go area. At this point, q_{rand} becomes q_{new} and the arc connecting q_{near} and q_{new} is added to the tree. Each tree stores the valid configurations and the information about their parent configuration in a data structure which allows to find the final path by backtracking from the target to the start point using proper arcs. The sampling and connecting operations are repeated until the maximum number of iterations is reached, or until all trees reach the target. A field $Tree.done$ is set to TRUE when one tree reaches the target, then a backtracking algorithm can build the path from the goal to the start configuration, q_{start}. A structure of, at most, $MaxTrees$ number of trees is returned as the solution of the algorithm. All of them may converge to a solution, but some may not if the maximum number of iterations was reached.

It is observed that at times, some of the trees are almost complete and others just started. This phenomenon comes from the ascending order operation that sorts trees in order to expand the closest point toward the random sample. When q_{near} of several trees have the same distance from q_{rand}, especially at the beginning, the first tree is always chosen. This has been easily solved by implementing a sorting function that gives priority to the tree which has fewest nodes.

Algoritmo 1 Multiple RRTrees Growth : (qinit, qgoal, MaxTrees)

```
Trees: initialize_MaxTrees(qinit)
 for iter=1:MaxIter
  Qreach=zeros(length(availableTrees));
  while (Qreach(any Tree)=TRUE) do
   prand: Random_Free_State(goal_bias);
   for i=1:availableTrees
         for all q in the Tree(i)
          if Reachable(prand,q,minRadius)
           Qreach(Tree(i))=TRUE
          end if
         end for
   end for
  end while
  for j=1:length(Q_reach)
   qnear(j)=NearestNeighbor(Qreach(j),prand,Tree(Qreach(j)));
   qnew(j)=SolveParameter(qnear(j),prand);
  end for
  [index]=ascending_sort(distance(qnear,prand));
  k=0; ans_ValidEdge=FALSE;
  while (ans_ValidEdge==FALSE && availableTrees>0)
   k=k+1
    ans_ValidEdge=ValidEdge(qnear(index(k)),qnew(index(k)));
  end while
  if ans_ValidEdge
   Tree(index(k)).addNode(qnew(index(k)))
   Tree(index(k)).addEdge(qnew(index(k)),qnear(index(k)))
   if qnew(index(k))==qgoal
         Tree(index(k)).done=TRUE;
   end if
   if all trees reach the goal (i.e. Trees.done==TRUE)
         break
   end if
  end if
 end for
return Trees
```

Table I
SETTING OF INPUT PARAMETERS FOR SIMULATIONS AND EACH EXPERIMENT.

Parameter		Values in simulation	Values in experiments
Max Curvature (mm^{-1})		0.025	0.00708
Path Sampling Interval (mm)		1	1
Probe Thickness (mm)		4	4.5
Control Margin (mm)		-	2
Max Iterations Number		6000	500
N^o of Multiple Trees		20	5
Goal Bias		0.2	0.2
Optimization weights α/β/γ	Case 1	1/0/0	0.5/0/0.5
	Case 2	0/1/0	
	Case 3	0/0/1	
Desired Speed (mm/sec)		-	1

IV. Simulation on Risk Brain Map

The algorithm is implemented in MATLAB 7.6.0 (®2008a, MathWorks inc.), and is executed on an Intel® Core^{TM} 2 Duo CPU T5870 @ 2.00GHz processor.

In order to prove the concept of a risk-based trajectory planner for brain surgery, the main brain structures of a segmented anonymous Magnetic Resonance Imaging (MRI)-based dataset were arbitrarily classified into six categories according to the risk associated with the insertion of the probe into each. The chosen categories were defined as "Avoid", "Dangerous", "Warning", "Careful", "Common", "Accessible" with each class represented by a unique grey value on the image, white being impenetrable while black meaning fully accessible. "Avoid" and "Dangerous" areas and optional user-defined restricted regions, such as patient specific constraints, are set to "no-go" areas that the probe must avoid. The no-go areas are dilated by the outer radius of the probe and an additional margin accounting for any uncertainty arising from inaccuracies in the probe's control.

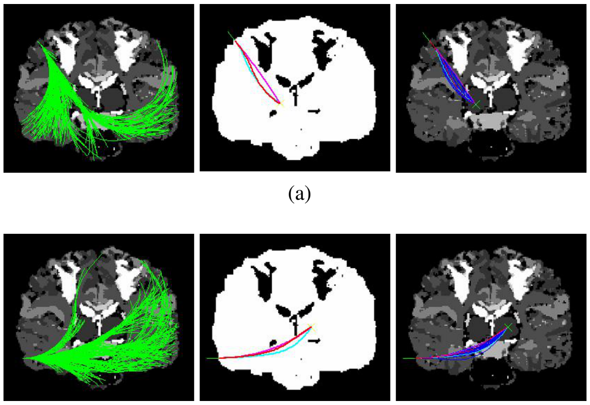
Path planning simulations were performed on various brain images. The reported examples show the results on a grey scale risk map (134×153 pixels) of a coronal section of the brain. Two sets of inputs for the entry configuration and the target position were used:

1) Entry: [29mm, 30mm, 50°] and target: [65mm, 80mm]
2) Entry: [16mm, 100mm, 0°] and target: [90mm, 75mm]

where the edge of 1 pixel was assumed to be 1 mm. Other input parameters are reported in Table I.

Since the environment is complex, the maximum number of iterations was set to 6,000 and the minimum radius of curvature to 40 mm. A higher number of possible iterations gives a greater probability of finding solutions. By reducing the minimum achievable radius, the probability of avoiding obstacles and dangerous areas is increased.

To extract the final paths, the set of arcs composing a solution are then converted into a set of configurations (x, y, θ), with a sampling interval set to 1 mm, where (x, y) is the set of 2D coordinates the probe tip's position and θ the orientation of the probe's tip with respect to the local frame of the entry configuration.


(a)

(b)

Figure 4. Two example problems (a) and (b) on a coronal brain slice, solved by multiple RRTs searching. Starting from the left, the RRTs search in green on the risk map, the white free region with the selected paths superimposed and the generated paths on the risk map.

For each set of inputs, the two images on the left of Fig. 4 show that all of the resulting RRTs as green lines on the brain risk map. The middle figures represent no-go areas in black and the paths obtained with three extreme weights (α, β, γ) selections in different colours. On the right-hand picture, all of the generated paths are depicted in blue. The minimum radius constraint is always satisfied with the minimum radius measured as 43 mm.

The minimum, maximum, root-mean-square, and average computation times of 50 trials for the first environment in Fig. 4a measured t_{min}=20sec, t_{max}=42sec, t_{RMS}=27.92sec, and t_{mean}=27.45sec respectively. For Fig. 4b, they measured t_{min}=19.97sec, t_{max}=40.7sec, t_{RMS}=28.38sec, and t_{mean}=27.94sec. In both cases the success rate in finding 20 paths was 100%.

From Fig. 4 it is possible to see the effects of weight set (α, β, γ) on the final paths. When α is set to 1, the shortest path, represented in magenta, is selected; when $\beta = 1$ the path with the largest clearance from no-go areas is selected, as depicted in cyan, while for $\gamma = 1$ the path in red is the least risky one in term of the accumulated risk values. Despite the fact that an increase in the number of trees requires more computation time, a larger population size improves the chances of obtaining a near optimum solution.

V. In Vitro Validation

A. Integration with flexible probe

To control the flexible probe and to interactively set the parameters of the path, a high level controller for the flexible probe (HLC-FP) was implemented. The HLC-FP consists of a trajectory planner, a graphical user interface (GUI) and a CORBA interface to communicate with the other components of a robotic suite, as shown in Fig. 5. A low level controller module, which considers the kinematic model of STING and provides the optimal inputs to the probe actuator box, is used to implement path following along the optimum path. It was implemented through Labview (National Instruments inc.) and embedded into a CompactRIO motion controller (National Instruments Inc.) [5].

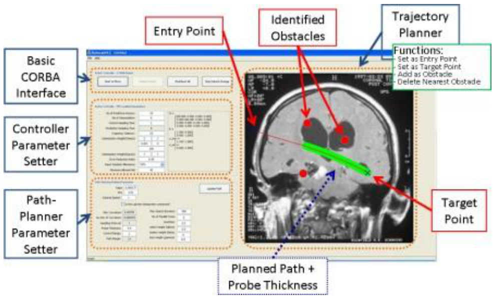


Figure 5. Graphic User Interface for Path Planning.

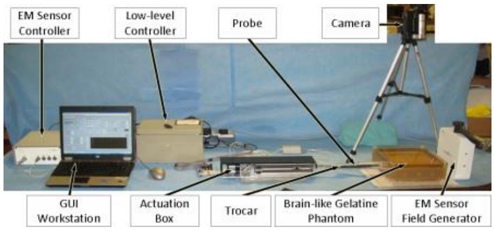


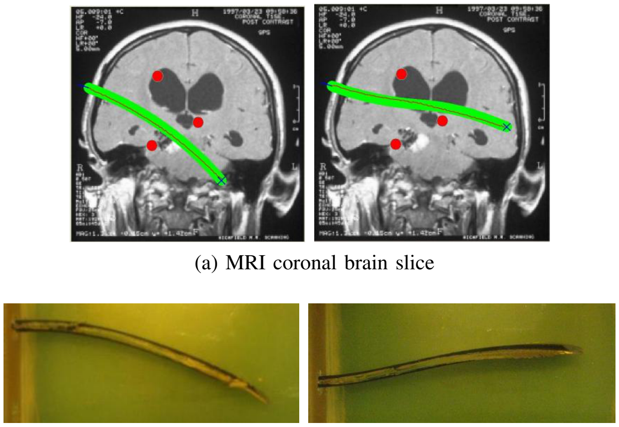
Figure 6. Experimental setup for the integrated system with the flexible probe.

The trajectory planner helps the operator (i.e. the operating surgeon) to determine the entry and target points and any identified obstacles. The flexible probe is then controlled to follow the optimal path generated by the path planner according to the modified brain map environment. In order to integrate the path planner developed in a Matlab environment into HLC-FP, it was converted into a C++ dynamic-link library linked into the HLC-FP at run time. The path planner communicates with the low level controller module using a TCP/IP-based network protocols.

B. Experiments in Gelatine

Figure 6 shows the experimental setup used to test the performance of the closed-loop trajectory controller which is integrated with the developed path planner, the flexible probe prototype and a gelatine sample. The scaled-up two-part prototype, 9mm in outer diameter, was used and a brain gelatine phantom of 6 wt. % was prepared according to the literature. The position of the tip of the probe was measured by a 5 degrees-of-freedom (DOF) electromagnetic sensor (Aurora, Northern Digital inc.). The forward speed was set to 1mm/s. Parameters for the path planner and constants for each test are reported in Table I. The maximum curvature was chosen based on the experiments using a real prototype.

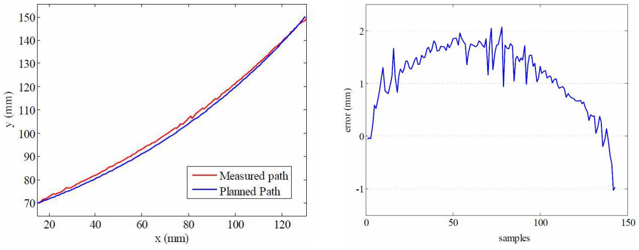
Figure 7a depicts the MRI image available via the intra-operative GUI alongside the picture showing the real behaviour of the probe; the green area shows the ideal path with the thickness of the probe, while the red curve represents the real position of the EM sensor on the tip of the probe penetrating the gelatine sample. Examples of a single bend



(a) MRI coronal brain slice

(b) Probe performance into the gelatin sample

Figure 7. Examples of tests.



(a)

(b)

Figure 8. Tracking results of the flexible probe.

trajectory and of a double bend trajectory are given in Fig. 7b.

Figure 8a compares the real path and the desired one in a single bend experiment: the desired path computed by the path planner is represented in blue and the measured path in red. Figure 8b shows the displacement error of the real tip position with respect to the planned trajectory: the error is measured as the distance between the measured point and the corresponding closest point on the planned path. Results in a single bend test show an error e_{RMS} = 1.31mm and a mean error e_{mean} = 1.15mm, with a standard deviation e_{std} = 0.63mm. The maximum error is constrained within 2mm of the plan, i.e. the control margin, to account for the uncertainty of the probe's movement due to its interaction with the surrounding tissue.

VI. Conclusion and Future work

This work describes our early progress on a curvilinear path planning algorithm for a steerable flexible probe designed for neurosurgical intervention. The maximum constraint on the path's curvature is explicitly considered by adopting a Reachablity-Guided Rapidly-exploring Random Trees approach, where the actual value is based on the mechanical properties of the designed probe. The implementation of multiple solution allows the clinician to select a high quality plan based on an several optimization criteria, which is generally preferable to having a deterministic solution which cannot be modified. In addition, the path planner makes use of a labelled image dataset to incorporate risk into the optimization process, thus enabling the safest path which meets the contraints to be selected. The path planning algorithm was integrated with the scaled up prototype of a steerable flexible probe and experiments within a brain-like gelatine phantom showed that the probe could follow the planned path with acceptable accuracy. Even though greater curvatures were tested in simulation, the method was able to produce a feasable path even for the relatively large minimum radius constraint imposed by the current probe prototype. Since the probe is currently under development, we plan to test the planner with a more flexible prototype in the future.

Further developments can be directed to finding a trajectory that takes into account a continuous and constrained change of the curvature, which is needed for the probe to steer with predefined speed. In addition, a significant optimization of the algorithm's implementation would be needed to achieve sub-second computation time, a requirement which would become mandatory for the probe to work in a dynamic environment.

References

[1] D. Minhas, J. A. Engh, M. M. Fenske, and C. Riviere, "Modeling of needle steering via duty-cycled spinning," in *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 2756–2759, August 2007.

[2] R. Webster III, N. J. Cowan, G. Chirikjian, and A. Okamura, "Nonholonomic modeling of needle steering," *Int. J. Rob. Res.*, vol. 25, pp. 509–525, May 2006.

[3] K. Trovato and A. Popovic, "Collision-free 6d non-holonomic planning for nested cannulas," *Medical Imaging: Visualization, Image-Guided Procedures and Modeling, Proc of SPIE*, 2009.

[4] L. Frasson, S. Y. Ko, A. Turner, T. Parittotokkaporn, J. F. Vincent, and F. Rodriguez y Baena, "STING: a soft-tissue intervention and neurosurgical guide to access deep brain lesions through curved trajectories," *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, vol. 224, pp. 775–788, 2010.

[5] S. Y. Ko, L. Frasson, and F. Rodriguez y Baena, "Closed-loop planar motion control of a steerable probe with a "programmable bevel" inspired by nature.," *IEEE Transactions on Robotics*, vol. 27, pp. 970–983, October 2011.

[6] C. I. Connolly, J. B. Burns, and R. Weiss, "Path planning using laplace's equation," in *In Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, pp. 2102–2106, 1990.

[7] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[8] J. Sethian, *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[9] C. Petrès, *Trajectory Planning for Autonomous Underwater Vehicles*. PhD thesis, Ocean Systems Laboratory, Heriot-Watt University, Department of Computing and Electrical Engineering, October 2007.

[10] S. Patil and R. Alterovitz, "Interactive motion planning for steerable needles in 3D environments with obstacles," in *International Conference on Biomedical Robotics and Biomechatronics*, University of Tokio, Japan, September 26-29 2010.

[11] L. Cohen and R. Kimmel, "Global minimum for active contour models: A minimal path approach," *Int. J. Comput. Vis.*, vol. 24, no. 1, pp. 57–78, 1997.

[12] J. Xu, V. Duindam, R. Alterovitz, and K. Goldberg, "Motion planning for steerable needles in 3d environments with obstacles using rapidly exploring random trees and backchaining," in *Proc. IEEE Int. Conf. Automation Science and Engineering (CASE)*, pp. 41–46, Aug. 2008.

[13] S. M. Lavalle and J. J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, pp. 293–308, 2000.