Exploration adaptive RRT*

Shounak Das
Department of Mechanical and Aerospace Engineering
West Virginia University
Morgantown, USA
sd0111@mix.wvu.edu

***Abstract*—This paper discusses a variant of the very popular sampling based planning algorithm RRT*(Rapidly exploring Random Trees)[2] which encourages the rover to explore new regions. While the original version minimizes the cost of the path by re-wiring the tree, this algorithm compromises between the cost of path (length of path in this paper) and information gained. This is done by adding an information penalty term to the cost of each node. The graph framework defined in [5] is used for running this algorithm. This paper also discusses a method with which the rover can stop exploring at any location and plan a shortest path to goal very quickly.**

***Index Terms*—path planning, RRT, information, gaussian process**

I. Introduction

With more planetary missions planned than ever before and rovers exploring on different planets, path planning has become one of the most essential components for a successful mission. A common scenario for planetary rovers is to plan a path from its current location to base station such that it moves through some regions along the way, which might have some important geological features. Finding the shortest path to the base station is not enough. It needs to sacrifice length of path for information gain such that its length of path remains below a limit. This limit can be the maximum distance the rover can travel with one full battery. A good pre-planned path is vital for this kind of exploration.

II. Problem Definition

Before going in to the details of the algorithm, we discuss the problem statement of this discussion. Suppose we have sent a planetary rover to some planet where it resides now. It needs a pre-planned path to follow from its current location(start) to its base station(goal). Note, we will not be going into the replanning part of this problem. It just needs an initial path. Another spacecraft orbiting the planet has sent locations(red) which is of interest to scientists, that must be visited by this rover on its way to the base station.

Previous works have put forward planning algorithms for information gathering using sampling-based methods [1][6][7] and information theory. Here, information is represented in a simpler way with *information score*. These are scalar values at every point in the map and range within $[0, 5]$. 0 represents a point with no interesting features(eg. geological features) and the 5 is a point of highest interest. Figure 1 shows a discrete map with the the start and goal points in green and information landmarks in red. But this prior map also has uncertainties which can be incorporated to convert this into continuous information map using Gaussian Processes(GP)[3]. Here Matern Kernel has been chosen as the covariance function and inference is done using the GPML Toolbox[4]. This is shown in 2 which shows a continous map with a resolution of $0.01{\times}0.01$. Using this definition, the information score of a location x is expressed as $I(x)$.

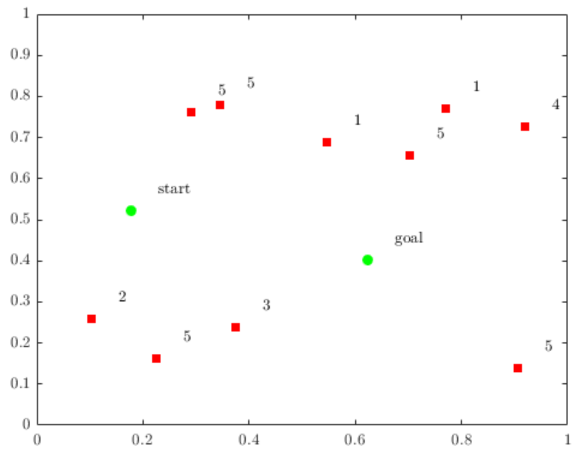


Fig. 1. Map with start, goal and locations of interest

III. Algorithm

The Informative RRT* described in 1 is very similar to the original RRT* described in [2]. So the same functions like `steer`, `SampleFree`, `ObstacleFree`, `CollisionFree` are used. The only difference is the selection of the best node to connect the newly sampled point to the tree and the rewiring part. On close comparison with the original algorithm, it can be seen that the cost updating line has an extra term called the `infPen` function. To understand this, we explain the fundamental change that has been made to the problem setup. The original algorithm expresses `Cost` of a node of a tree as the length of the path from the root of the tree (the start location) to that node. Here this term is modified to be the sum of length of path from the start and the cumulative information penalty along the same path. The *information penalty* at a point on the map is defined as

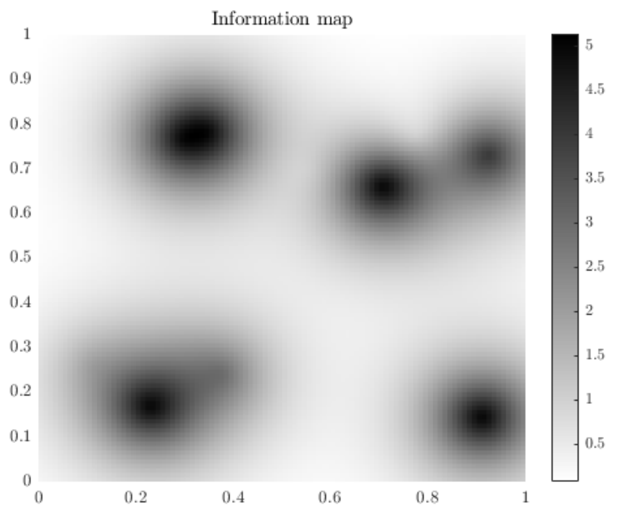


Fig. 2. Smoothed information map (I) based on prior map in 1 using GP

$$infPen(x) = max(I) - I(x) \ .$$

This means that high information zones will get less penalty than the less information zones. Thus the path will tend to go through high information regions but since the total cost also has length component, the paths will tend to compromise between the length and information gain along it. After a certain number of iterations, we select the leaf node which has least sum of the length and information penalty. Since the penalty term is always positive, the cost function remains additive and thus the tree structure is maintained resulting in convergence like RRT*. The time and space complexity remains same as RRT* since the added information penalty term is a query from a pre-computed I matrix which is constant time.

Algorithm 1: Adaptive RRT*

```
$V \leftarrow \{x_{init}\}$;
$E \leftarrow \emptyset$;
for $i = 1...n$ do
    $x_{rand} \leftarrow SampleFree$;
    $x_{nearest} \leftarrow Nearest(G = (V, E), x_{rand})$;
    $x_{new} \leftarrow steer(x_{nearest}, x_{rand})$;
    if $ObstacleFree(x_{nearest}, x_{new})$ then
        $X_{near} = Near(G = (V, E), x_{new}, min\{\gamma_{RRT*} (log(card(V)/card(V))^{\frac{1}{d}}, \eta\}$;
        $V \leftarrow V \cup \{x_{new}\}$;
        $x_{min} \leftarrow x_{nearest}$;
        $c_{min} \leftarrow Cost(x_{nearest}) + c(Line(x_{nearest}, x_{new})) + \boldsymbol{infPen}(x_{new})$;
        foreach $x_{near} \in X_{near}$ do
            if $CollisionFree(x_{near}, x_{new}) \wedge Cost(x_{near}) + c(Line(x_{near}, x_{new})) + \boldsymbol{infPen}(x_{new}) < c_{min}$ then
                $x_{min} \leftarrow x_{near}$;
                $c_{min} = Cost(x_{near}) + c(Line(x_{nearest}, x_{new})) + \boldsymbol{infPen}(x_{new})$;
        $E \leftarrow E \cup \{(x_{min}, x_{new})\}$ ;
        foreach $x_{near} \in X_{near}$ do
            if $CollisionFree(x_{near}, x_{new}) \wedge Cost(x_{new}) + c(Line(x_{new}, x_{near})) + \boldsymbol{infPen}(x_{near}) < Cost(x_{near})$ then
                $x_{parent} \leftarrow Parent(x_{near})$;
                $E \leftarrow (E \setminus \{x_{parent}, x_{near}\}) \cup \{x_{new}, x_{near}\}$;
return $G = (V, E)$;
```

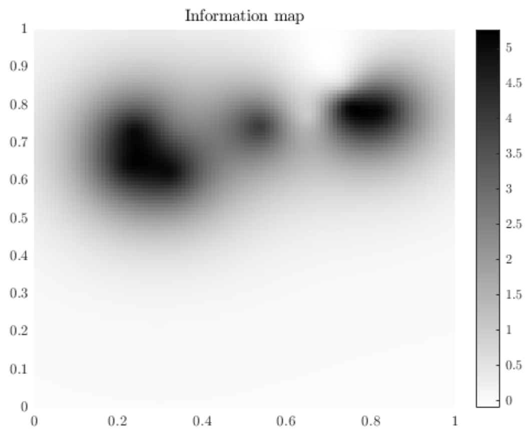
IV. Switching mode RRT* planning

The adaptive RRT* algorithm described here produces a path that is a balance between exploration and least cost. However, we might want the rover to stop exploration and move to base station after some time. In that case, it cannot follow this path anymore. This section discusses how the *exploration* mode of the rover can be quickly changed to *return-to-base-station* mode in an online manner.

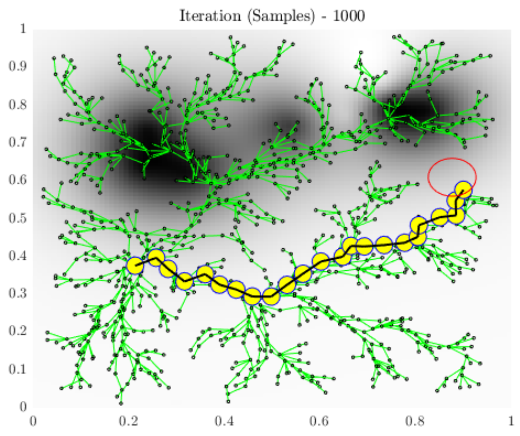
First we grow 2 trees starting from the goal, one following the standard RRT* algorithm and another following the adaptive RRT* in 1. Let the standard tree and the adaptive tree be represented by T_{normal} and $T_{adaptive}$ respectively. T_{normal} produces shortest path, suppose P_{normal} and as discussed in the previous section $T_{adaptive}$ gives the path, $P_{adaptive}$, which has the least sum of path length and information penalty. These paths are represented as sets of waypoints, which the rover uses for global planning. The rover starts off in exploration mode by following $P_{adaptive}$. After some time , it decides to end its *exploration* mode and go to base station as quickly as possible. Now it should transfer to T_{normal} and will find a path to its root i.e goal. Let the location where it decides to stop exploration be called the *switching point*. The node in T_{normal} which is closest to this *switching point* is selected as the starting point and follows the tree to the goal. Hence, this method describes a method using two different trees which allows it to switch from *exploration* mode to *return-to-base-station* mode.

V. Simulation Results

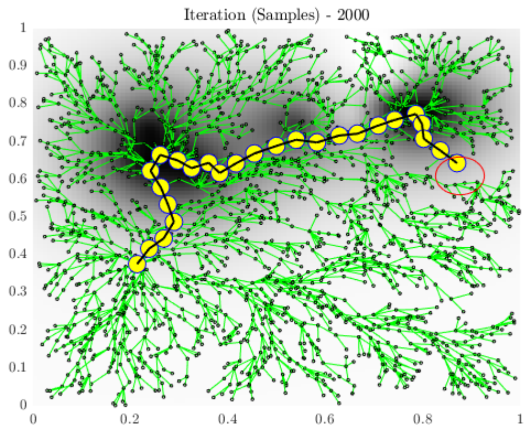
Algorithm 1 is implemented in Matlab using the Robotics Toolbox [5]. Figure 3 shows results of after running it with increasing sample sizes in a 2D environment with no obstacles and a prior information map given in 3(a). As the number of nodes increase, it gradually finds more informative paths but does not sacrifice length completely. The path after 5000 samples(Figure 3(f)) looks like a stretched version of shortest path that passes through the interesting regions.



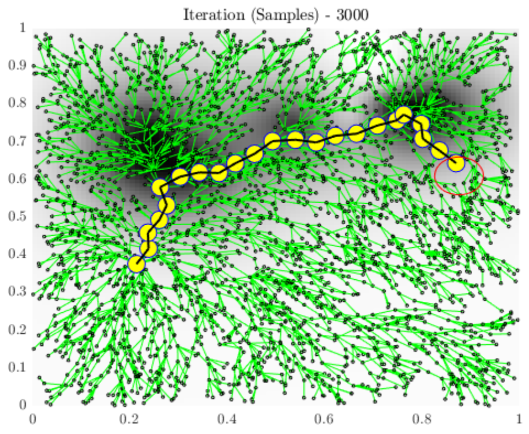
(a) Iterations- 0 ; Length- 0 ; Information score- 0 ;



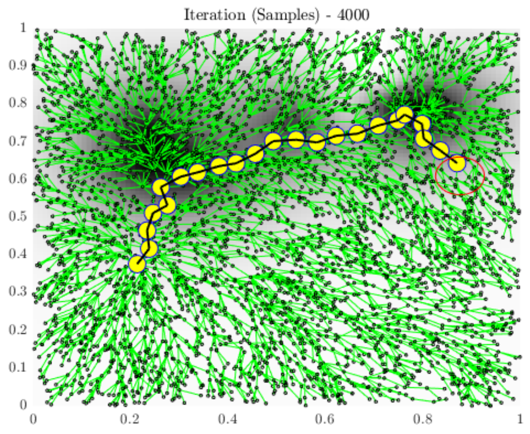
(b) Iterations- 1000 ; Length- 1.2747 ; Information score- 6.2878 ;



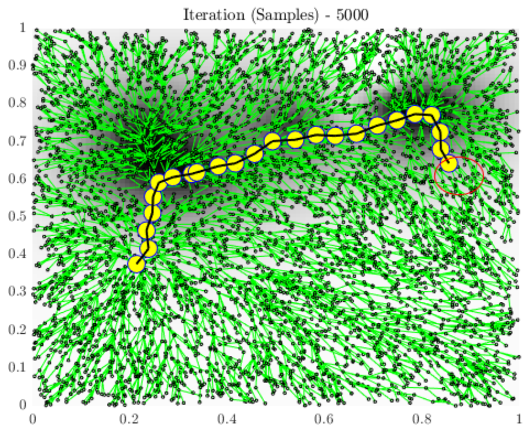
(c) Iterations- 2000 ; Length- 1.7039 ; Information score- 81.1073 ;



(d) Iterations- 3000 ; Length- 1.2747 ; Information score- 67.4307 ;



(e) Iterations- 4000; Length- 1.2747 ; Information score- 67.7974 ;



(f) Iterations- 5000 ; Length- 1.4716 ; Information score- 73.8061 ;

Fig. 3. Evolution of best compromised path with increasing iterations

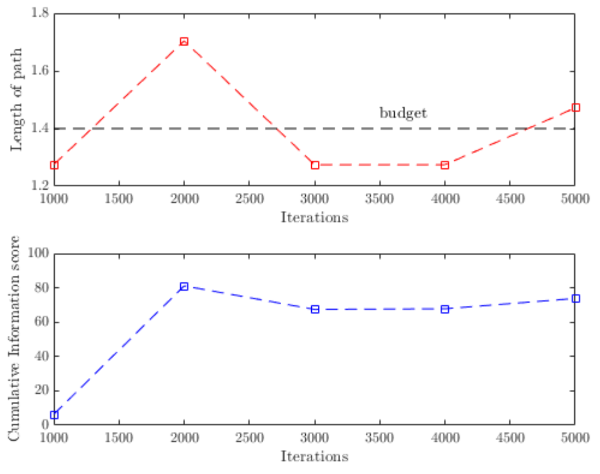


Fig. 4. Change of path length and information score with iterations

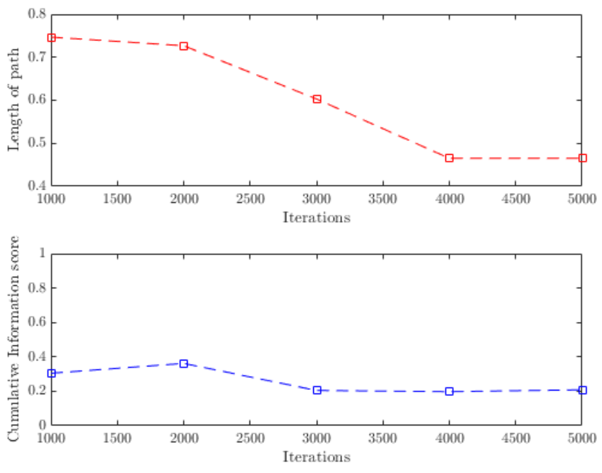


Fig. 5. Change of path length and information score with iterations when information zones are far away

It also might be the case that the path length has a limit which should not be crossed, i.e a budget. Figure 4 shows how the path length and the information scores along the best path changes with iterations. The horizontal line shows the budget that must be met while planning. From this figure we can select the best path among all the paths that lie below this line has the highest information score.

However, if the information zones are far away from the start and goal, then the solution paths will not move towards that region since the information penalty and path length that builds up in this high information path is far more than the low information lesser length paths, which makes sense practically. This is shown in the figures 7 and 5. In this scenario, it essentially works like the original RRT* algorithm, with improvement in path lengths while the information scores almost remain unchanged.

The switching mode method has been implemented in simulation as well using 2 trees each with 5000 nodes. The gray regions denote the exploration zones. The green path is the shortest path with standard RRT* and the yellow path is explorative path obtained with algorithm 1. In figure 6, the switching point is shown with the arrow. The switched mode path, shown in cyan overlaps with the informative path till the switching point is reached and then it changes to the shortest path.

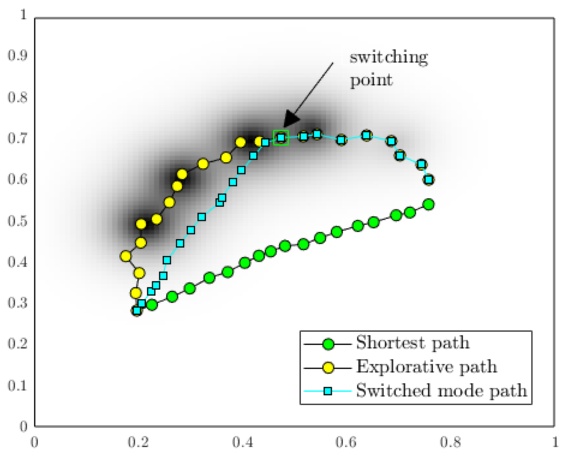


Fig. 6. Rover switching paths from exploration mode to shortest path mode

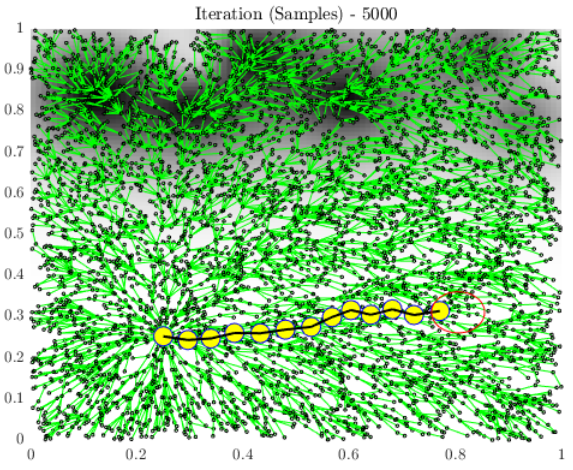


Fig. 7. Best path with 5000 iterations when information zones are far away

All simulations have been done without any obstacles, but will work the same with obstacles, with the samples only drawn from the free regions.

VI. Discussion and Future Work

This work demonstrates the results of adding a penalty term to the usual distance cost function in RRT* to make it explore information rich areas. The resulting path turns out to be a compromise between the shortest path and a long path with high information score. The advantage of this method is its simplicity. Future work will look into penalties which are a function of the node's distances from start and goal as well as online versions. Analysis with mathematically rigorous description of information in the form of entropy and mutual information also needs to be investigated.

Acknowledgment

I would like to thank Dr. Guilherme Pereira for continued support and guidance in MAE593A class, for which this article was written.

References

[1] Hollinger, Geoffrey A., and Gaurav S. Sukhatme. "Sampling-based Motion Planning for Robotic Information Gathering." Robotics: Science and Systems. Vol. 3. No. 5. 2013.

[2] Karaman, Sertac, and Emilio Frazzoli. "Sampling-based algorithms for optimal motion planning." The international journal of robotics research 30.7 (2011): 846-894.

[3] Rasmussen, Carl Edward. "Gaussian processes in machine learning." Summer School on Machine Learning. Springer, Berlin, Heidelberg, 2003.

[4] Rasmussen, Carl Edward, and Hannes Nickisch. "Gaussian processes for machine learning (GPML) toolbox." The Journal of Machine Learning Research 11 (2010): 3011-3015.

[5] Corke, Peter I. "A robotics toolbox for MATLAB." IEEE Robotics Automation Magazine 3.1 (1996): 24-32.

[6] Viseras, Alberto, Dmitriy Shutin, and Luis Merino. "Online information gathering using sampling-based planners and GPs: An information theoretic approach." 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2017.

[7] Lim, Zhan Wei, David Hsu, and Wee Sun Lee. "Adaptive informative path planning in metric spaces." The International Journal of Robotics Research 35.5 (2016): 585-598.