경로 계획을위한 샘플링 기반 알고리즘 조사

Vi Van Vo*, 염상길, 추현승

성균관대학교 소프트웨어대학

e-mail : {vovanvi, sanggil12, choo}@skku.edu

Survey of Sampling-Based Algorithms for Path Planning

Vi Van Vo, Sanggil Yeoum, HuynSeung Choo
The College of software, Sungkyunkwan University

Abstract

Sampling-based algorithms are one of the most commonly approaches which give good results in robot path planning with many degree of freedom. So that many proposed methods as well as their improvement based on these approaches have been proposed. The purpose of this paper is to survey some current algorithms using for path planning, the original proposed methods as well as their improvement. Some advantages and disadvantages of these algorithms will be also mentioned, how the improved version of the proposed methods overcome the original proposed methods' drawback.

Keywords: Path planning, sampling-based algorithms, rapidly-exploring random tree, probabilistic roadmaps

1. Introduction

Nowadays, path planning is one of the most important feature integrated in robots such as underwater robots, self-driving cars [1], unmanned aerial vehicles (UAV) [2,3], micro air vehicles [4, 5], vacuum cleaning robot [6]. In robotics, path planning is a fundamental research area [7-9] where robots automatically move from the initial point to the target point by themselves without colliding with any obstacles during the process. In path planning problems, there are many methods have been proposed to solve it. These methods somehow have some similar mechanisms, based on that we group some of them into sampling-based algorithms category due to the fact that in these methods the planning occurs by sampling the configuration space.

The paper is organized as follow. Section 2 discusses some algorithms which are classified in sampling-based algorithms. We will conclude our discussion and outline some directions for future research in the section 3.

2. Sampling-Based Algorithms

Sampling-based methods sample the configuration space as a set of nodes, or cells or other forms, then capture the connectivity of this environment or search to find feasible path or optimal path from the initial point to the target point. To do this, these kind of methods need to know the information of the configuration space beforehand. These randomized methods can provide fast solutions for difficult problems but they do not guarantee that they can find a solution in all cases. There are many algorithms can be classified as sampling-based methods, such as Rapidly-Exploring Random Tree (RRT) and its improved versions, Probabilistic Roadmaps (PRM) and its improved versions, Voronoi [10, 11] and Artificial Potential Field algorithms [12]. These methods are also put in sampling-based algorithms because they need to know the whole information of the configuration space, including obstacles and free area, beforehand. However, in scope of this paper we only show two commonly methods and their improvement in the sampling-based algorithms as demonstrated in the Figure 1.

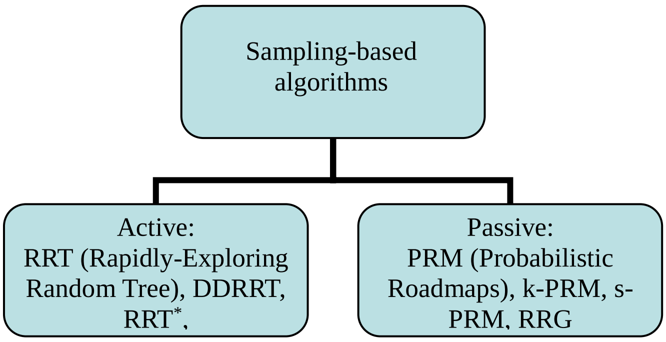

Sampling-based algorithms
Active: RRT (Rapidly-Exploring Random Tree), DDRRT, RRT*,
Passive: PRM (Probabilistic Roadmaps), k-PRM, s-PRM, RRG

Figure 1: Elements of sampling-based algorithms

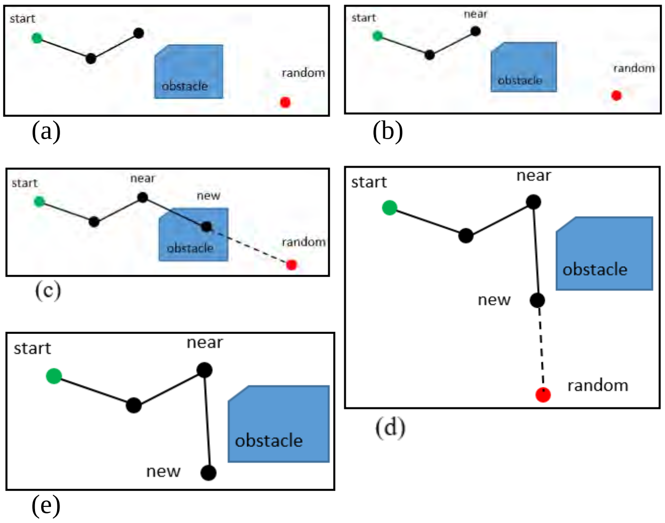
In the sampling-based algorithms, the algorithms can be divided into two parts [13]: Active methods and Passive methods. To give some details of the elements of sampling-based methods, the paper analyses one by one method in each part.

2.1. Active methods

Active methods consist of algorithms which can achieve the best feasible path from initial point to the goal by processing procedure itself.

2.1.1. Rapidly-Exploring Random Tree (RRT) [14, 19]: Given an area with obstacles, a starting point and a goal, step-size which defines the distance we can go in one single step, number of randomized nodes. The purpose is to find a feasible path from the starting point to the goal. In particular, the algorithm works as follow:

- The search starts from q_{start}
- A random node q_{random} will be generated, as shown in figure 2(a). If the random node is in the collision-free area, do the next step. If the random node is in collision with the obstacle, discard the q_{random}, start a new iteration.
- A nearest node q_{near} with the random node q_{random} will be determined, as shown in Figure 2(b)
- From q_{near}, a connection will go in the direction of the q_{random} in a step-size distance.
- If the new node q_{new} and the path from the q_{near} to the q_{new} is in a collision with the obstacle, discard them, as shown in Figure 2(c). Start a new iteration. If they are in collision-free area, then add them to the tree, as shown in Figure 2(d) and remove the random node q_{random}, as shown in Figure 2(e).
- The search will finish when $q_{new} = q_{goal}$.



(a) (b) (c) (d) (e)

Figure 2. The procedure of RRT algorithm.

The RRT uniformly samples nodes based on Monte Carlo, this makes the bias explored region which will increase with the time even though the advantage of RRT is to find a path from an initial node to the goal. When the environments are cluttered, the RRT algorithm will consume much more time to make the search.

2.1.2. Dynamic Domain RRT: [15] The DDRRT introduces a n-dimensional sphere of certain radius r at the center to represent the reachable region. At each iteration if the distance of new random node to the nearest node is within radius r, the random node will be chosen; otherwise it will be discarded. Then try to connect the nearest node and the random node; if the extension is succeed, then set $r = \infty$; otherwise set $r = R$, where R is the boundary radius of dynamic domain. The DDRRT can solve the Voronoi bias problem of original RRT; it can ensure fast exploration. But the path is generated by this algorithm will not be optimal.

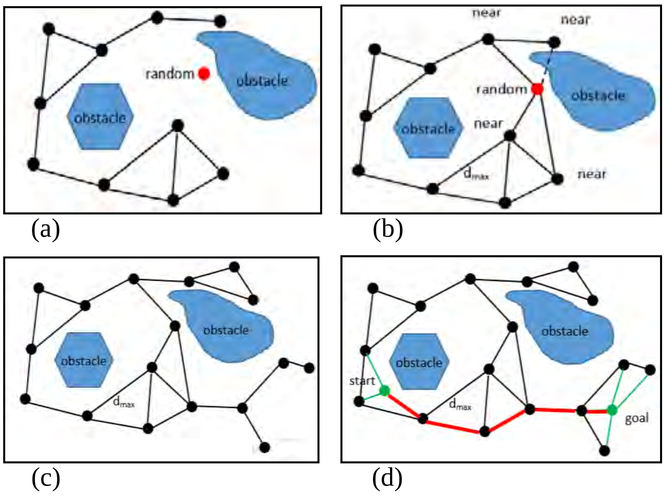
2.1.3. RRT-Star (RRT*): [16] RRT* finds the nearest state and the neighbor states of a new random node, then adds the nearest node to the tree and the minimal cost connection. After that, the RRT* tries to discard the larger cost connections from the new random node to neighbor nodes. After the pruning and reconnecting, the tree is updated, it becomes more dense and compact. In the end, we can obtain the tree with overall minimal cost. However, there are some drawbacks of this algorithm, it cannot work to generate multipath and the time consumption increases

2.2. Passive methods

Passive methods include algorithms which can pick up the best feasible path to the goal by combining with another search algorithms because they can only generate the set of paths connecting from initial point to the goal.

2.2.1. Probabilistic Roadmaps (PRM) [17, 18] Given an area with a starting point, a goal, a maximum distance d_{max} which define the maximum distance we can go in one single step, number of randomized nodes. The aim of the method is to combine with another algorithm to find the best feasible path between starting point and the goal.

- In the learning phase, build the roadmap
  - A random node q_{random} is sampled. If q_{random} is in obstacle, then discard it. Otherwise, add it to the roadmap, as displayed in Figure 3(a).
  - Find all the neighbor nodes with the random node q_{random} within a specific range d_{max}
  - Then connect them together if the path is in collision-free area. Otherwise, do not connect them, as shown in the Figure 3(b).
  - So on and so forth, the process is going on until all a certain number of nodes has been sampled, as shown in Figure 3(c).
- In the query phase, connect the starting node and the goal to the roadmap. A graph search algorithm, like Dijkstra algorithm is used to find the shortest path of these two nodes, as shown in the Figure 3(d).



(a) (b) (c) (d)

Figure 3. The procedure of PRM algorithm.

The PRM algorithm can be applied to systems with lots of degrees of freedom. However it may take a long time to generate a sufficient number of samples. Also, it is possible to have a situation where the algorithm would fail to find a path. If there are not enough number of sample nodes, then the graph is disconnected, when the starting point and the goal are in different graph, then the algorithm will not find the path to connect them.

3. Conclusion

There are many algorithms have been proposed to improve the path planning efficiency and the quality of the planning. In this paper, we analyze some basic and the most common algorithms in sampling-based algorithms which have established some success to solve path planning problem in robotic. The sampling-based algorithms can be classified in to active methods and passive methods, in which active methods can find the optimal path by itself, whereas the passive methods need to combine other searching methods to find the best feasible path from the starting point to the goal. These methods have some advantages and disadvantages themselves. Nowadays, there are more and more proposed methods which improve the strategies of the path planning not only in robotics but also in other fields [20]. In the future work, we will study and update more algorithms for our work to see how these methods can be applied in real life scenarios.

ACKNOWLEDGMENT

본 논문은 기초연구사업 (NRF-2010-0020210)과 과학기술정보통신부 및 정보통신기획평가원의 Grand ICT 연구센터지원사업 (IITP-2019-2015-0-00742)의 연구결과로 수행되었음.

References

[1] G. INC, Google self-driving car project, https://www.google.com/selfdrivingcar/.

[2] H.Himawan Triharminto, A. Prabuwono, T. Adji,N. Setiawan, and N. Chong, UAV Dynamic Path Planning for Intercepting of a Moving Target: A Review, vol. 376 of Book Section 18, Springer, Berlin, Germany, 2013.

[3] F. Yan, Y. Zhuang, and J. Xiao, “3D PRM based real-time path planning for UAV in complex environment,” in Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO ’12), vol. 6, pp. 1135–1140, Guangzhou, China, December 2012.

[4] F. Yan, Y.-S. Liu, and J.-Z. Xiao, “Path planning in complex 3D environments using a probabilistic roadmap method,” International Journal of Automation and Computing, vol. 10, no. 6, pp. 525–533, 2013.

[5] F. Schler, 3d path planning for autonomous aerial vehicles in constrained spaces [Ph.D. thesis], Department of Electronic Systems, Faculty of Engineering and Science, Aalborg University, Aalborg, Denmark, 2012.

[6] iRobot INC, “Vacuum cleaning robot,” http://www.irobot.com/For-the-Home/Vacuum-Cleaning/Roomba.aspx.

[7] H. M. Choset, Principles of Robot Motion: Theory, Algorithms, and Implementations, MIT Press, 2005.

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

[9] J.-C. Latombe, Robot Motion Planning, vol. 124, Kluwer Academic, Norwell, Mass, USA, 1991.

[10] V. A. Luchnikov, N. N. Medvedev, L. Oger, and J.-P. Troadec,” Voronoi-Delaunay analysis of voids in systems of nonspherical particles,” Physical Review E, vol. 59, no. 6, pp. 7205–7212, 1999.

[11] M. I. Shamos and D. Hoey, “Closest-point problems,” in Proceedings of the 16th Annual Symposium on Foundations of Computer Science, pp. 151–162, Berkeley, Calif, USA, 1975.

[12] O. Khatib, Real-Time Obstacle Avoidance for Manipulators and Mobile Robots, Book Section 29, Springer, New York, NY, USA, 1990.

[13] Liang Yang, Juntong Qi, Dalei Song, Jizhong Xiao, Jianda Han, and Yong Xia, “Survey of Robot 3D Path Planning Algorithms,” Journal of Control Science and Engineering, vol. 2016, Article ID 7426913, 22 pages, 2016.

[14] S. M. LaValle, “Rapidly-exploring random trees a new tool for path planning,” Tech. Rep. 98-11, Computer Science Department, Iowa StateUniversity, Ames, Iowa, USA.

[15] A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle, “Dynamic domain RRTs: efficient exploration by controlling the sampling domain,” in Proceedings of the IEEE International Conference on Robotics and Automation, pp. 3856–3861, Barcelona, Spain, April 2005.

[16] S. Karaman and E. Frazzoli, “Optimal kinodynamic motion planning using incremental sampling-based methods,” in Proceedings of the 49th IEEE Conference on Decision and Control (CDC ’10), pp. 7681–7687, Atlanta, Ga, USA, December 2010.

[17] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high dimensional configuration spaces,” IEEE Transactions on Robotics and Automation, vol. 12, no. 4, pp. 566–580, 1996.

[18] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” International Journal of Robotics Research, vol. 30, no. 7, pp. 846–894, 2011.

[19] M. Lavalle, Steven & Kuffner, James. (2000). Rapidly-Exploring Random Trees: Progress and Prospects. Algorithmic and computational robotics: New directions.

[20] J.C. Latombe, “Motion planning: A journey of robots molecules, digital actors, and others artifacts”, Int. J. Robot. Res., vol. 18, no. 11, pp. 1119-1128, Nov. 1999