УДК 004.9

Фатеев Алексей Александрович, магистр, МГТУ “Станкин”, г. Москва

РАЗРАБОТКА АЛГОРИТМА ПЛАНИРОВАНИЯ ПЕРЕМЕЩЕНИЯ НА ОСНОВЕ МЕТОДА RRT*-SMART

Аннотация

В статье рассматривается задача разработки эффективного алгоритма планирования перемещения мобильных роботов на основе метода RRT-SMART. Базовый алгоритм RRT (Rapidly-exploring Random Tree) широко применяется для решения задач планирования траекторий в многомерных конфигурационных пространствах, однако его модификация RRT обеспечивает асимптотическую оптимальность, а расширение RRT*-SMART дополнительно ускоряет сходимость к оптимальному решению за счёт использования эвристических стратегий оптимизации и перепланирования. В работе анализируются особенности реализации алгоритма, исследуются его основные этапы: построение случайного дерева поиска, процедура переподключения вершин и использование механизма «умного» укорачивания траектории (path smart shortening). Предлагаемый подход позволяет существенно повысить качество формируемых траекторий при сохранении приемлемой вычислительной сложности, что делает его пригодным для применения во встроенных системах управления мобильными роботами.

Annotation

Abstract: This paper considers the problem of developing an efficient motion planning algorithm for mobile robots based on the RRT-*SMART method. The basic*

RRT (Rapidly-exploring Random Tree) algorithm is widely used for solving trajectory planning problems in high-dimensional configuration spaces. Its RRT modification provides asymptotic optimality, while the RRT*-SMART extension further accelerates convergence to the optimal solution through the use of heuristic optimization and replanning strategies. The paper analyzes the implementation features of the algorithm and examines its main stages: building a random search tree, vertex rewiring procedure, and the use of path smart shortening mechanism. The proposed approach significantly improves the quality of generated trajectories while maintaining acceptable computational complexity, making it suitable for use in embedded control systems of mobile robots. The results of the work can be used in the development of autonomous navigation systems in static and dynamically changing environments.*

Ключевые слова: *планирование перемещения, RRT*-SMART, мобильная робототехника, оптимизация траектории, асимптотическая оптимальность, случайные деревья поиска, автономная навигация.*

Keywords: *motion planning, RRT*-SMART, mobile robotics, trajectory optimization, asymptotic optimality, rapidly-exploring random trees, autonomous navigation.*

За последние четыре десятилетия было предложено множество алгоритмов планирования траекторий, включая геометрические, сеточные, методы потенциальных полей, нейронных сетей, генетические и методы на основе случайной выборки. Каждый из этих алгоритмов имеет свои преимущества и недостатки с точки зрения временной и пространственной сложности, а также оптимальности построенного пути.

Алгоритм быстро растущих случайных деревьев (Rapidly-exploring Random Tree, RRT) является одним из самых быстрых при нахождении

первого же пути, однако он не обеспечивает асимптотической оптимальности. Значительный прорыв произошёл с появлением алгоритма RRT. Данный алгоритм является расширением базового RRT и отличается тем, что быстро находит начальный путь, а затем непрерывно оптимизирует его по мере увеличения количества выборок. Благодаря этому RRT наряду с вероятностной полнотой обеспечивает асимптотическую оптимальность. Дальнейшим развитием данного подхода стал алгоритм RRT-SMART, который дополнительно ускоряет сходимость к оптимальному решению за счёт использования эвристических стратегий оптимизации и механизма «умного» укорачивания траектории.

Перед разбором алгоритма RRT*-Smart, следует изучить алгоритм RRT*. Пусть Q определят конфигурационное пространство, в котором q_{obs} - область препятствий, $q_{obs} = \frac{Q}{q_{obstacle}}$ - область без препятствий, а q_{goal} - область цели. RRT* работает, чтобы найти входной сигнал $u{:}\,[0{:}\,T] \in U$, который дает допустимый путь $x(t) \in q_{free}$, который начинается с $x(0) \in q{-}initial$ до $x(T) \in q{-}goal$, следуя системным ограничениям. При нахождении этого решения RRT* поддерживает дерево $T = (V, E)$ вершин V, отобранных из пространства свободных от препятствий состояний Q_{free}, и ребер E, которые соединяют эти вершины вместе. Этот алгоритм использует набор функций, которые объясняются ниже:

Sampling: случайным образом выбирает состояние $q_{rand} \in Q_{free}$ из конфигурационного пространства без препятствий.

Distance: функция возвращает стоимость пути между двумя состояниями, предполагая, что область между ними свободна от препятствий.

Nearest Neighbor: функция Nearest(T, q_{rand}) возвращает ближайшую вершину из $T = (V, E)$ до q_{rand} в терминах стоимости, определяемой функцией расстояния.

Steer: функция Steer(q_{rand}, $q_{nearest}$) решает для входного сигнала управления $u[0, T]$, который управляет системой от $x(0) = q_{rand}$ до $x(T) = q_{nearest}$ вдоль пути $x: [0, T] \rightarrow Q$, давая q_{new} на расстоянии Δd от $q_{nearest}$ в сторону q_{rand}, где Δd - это приращение расстояния.

Collision Check: функция Obstaclefree(x) определяет, находится ли путь $x: [0, T]$ в области без препятствий Q_{free} для всех $t = 0$ до $t = T$.

Near-by Vertices: функция Near(T, q_{rand}, n) возвращает соседние вершины, которые находятся в шаре объема $(\beta(\log_n n))$ вокруг q_{rand}, где β - константа.

Insert node: функция Insertnode(q_{parent}, q_{new}, T) добавляет узел q_{new} в V в дереве $T = (V, E)$ и соединяет его с уже существующей вершиной q_{parent} в качестве родительской, и добавляет это ребро в E. Стоимость назначается q_{new}, которая равна стоимости его родителя плюс евклидова стоимость, возвращаемая функцией Distance между $cnew$ и его родителем $cparent$.

Rewire: функция Rewire(T, q_{near}, q_{min}, q_{new}) проверяет, меньше ли стоимость до вершин в q_{near} через q_{new} по сравнению с их старыми затратами. Если это так для определенной вершины, ее родитель q_{parent} изменяется на q_{new}.

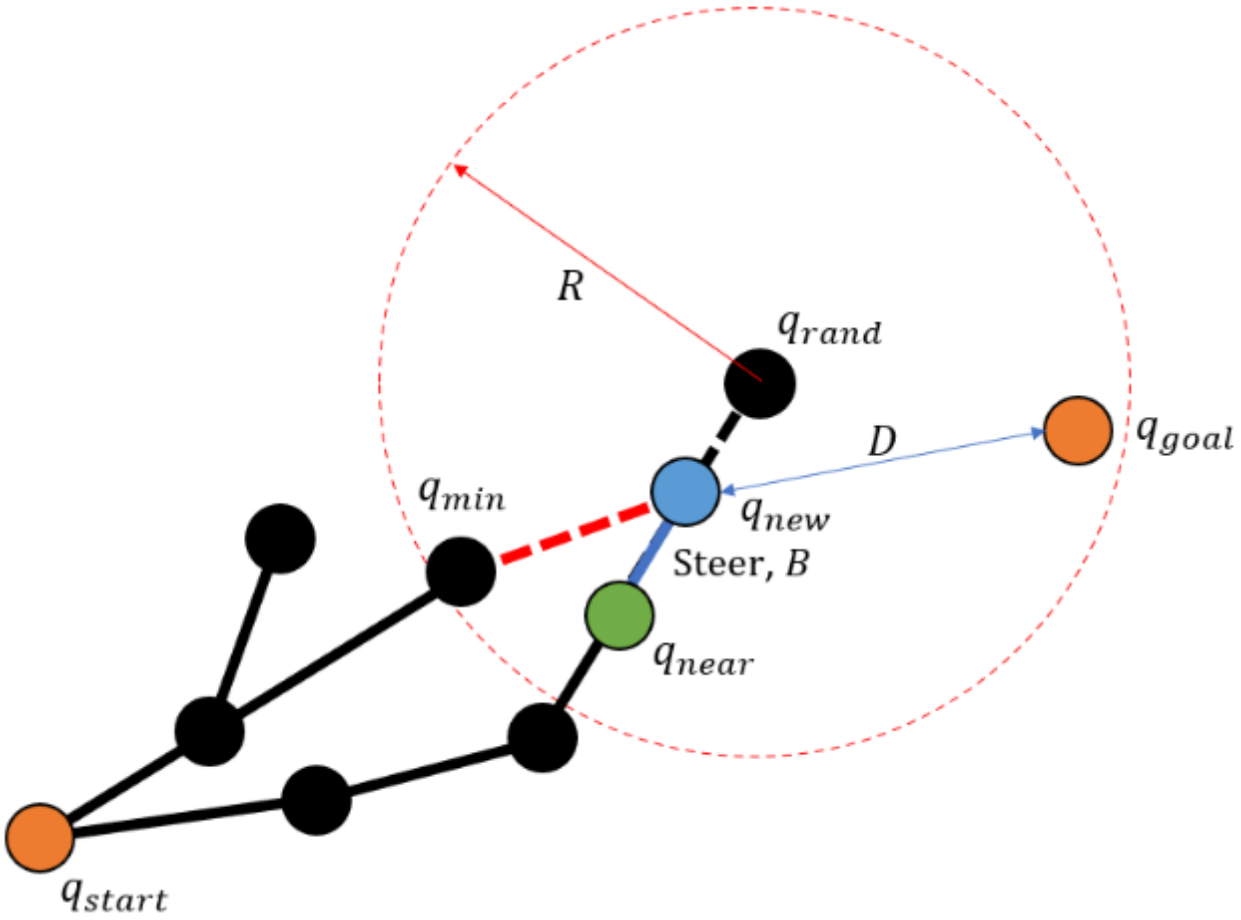

Рисунок 1. Процесс построения дерева методом RRT*

RRT*-Smart представляет два новых ключевых изменения: Intelligent Sampling и Path Optimization. Изначально, RRT*-Smart работает так же, как и RRT*. Как только найден первый путь, он затем оптимизирует этот путь путем соединения напрямую видимых узлов. Этот оптимизированный путь дает точки смещения для Intelligent Sampling. Этот процесс продолжается по мере прогресса алгоритма и непрерывной быстрой оптимизации пути. Всякий раз, когда находится более короткий путь, смещение перемещается в сторону нового пути.

Path Optimization. Как только RRT* предоставляет начальный путь, узлы в этом пути $x: [q_{init}, q_{goal}] \rightarrow Q$, которые видны друг другу, соединяются напрямую. Начинается итерационный процесс от q_{goal} и движется к q_{init}, проверяя прямые соединения с последовательными родителями каждого узла, пока не нарушится условие отсутствия столкновений. В конце этого процесса

больше не остается напрямую соединяемых узлов. Таким образом, путь оптимизируется на основе концепции треугольного неравенства, как показано на рис. 31. Согласно треугольному неравенству: $c < a + b$.

При каждой проверке видимости между двумя узлами требуется проверка на отсутствие столкновений. Для этой цели мы использовали интерполяцию, которая не требует явных сведений о препятствиях, как это необходимо в других методах проверки столкновений, например, проверяя точки пересечения. Таким образом, этот метод интерполяции для проверки на отсутствие столкновений не зависит от формы препятствий и является менее затратным с вычислительной точки зрения.

Количество узлов, присутствующих в этом пути, теперь значительно меньше по сравнению с исходным путем, найденным RRT*. Эти узлы называются маяками $c_{beacons}$, которые образуют основу для интеллектуального сэмплирования.

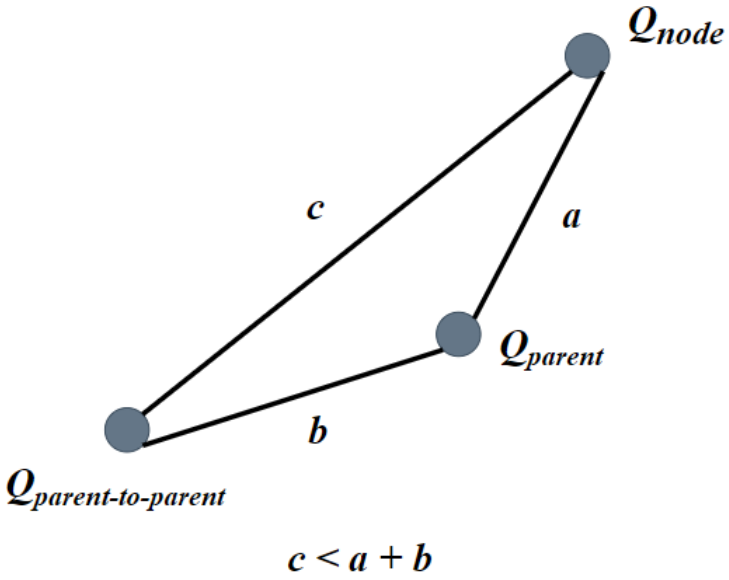

Рисунок 2. Оптимизация пути на основе треугольного неравенства

Intelligent Sampling. Идея интеллектуального сэмплирования заключается в приближении к оптимальности путем генерации узлов как можно ближе к вершинам препятствий, следуя идее техники видимости графа. Однако техника видимости графа требует сложного моделирования

окружающей среды и явных сведений о препятствиях. Более того, они могут не найти решение в средах, содержащих препятствия с сложной геометрией (вогнутые, полигональные, круговые и т.д.).

Как только найден начальный путь, интеллектуальное сэмплирование начинается с определенного количества сэмплов, которые непосредственно порождаются в сфере радиуса $R_{beacons}$ с центром в $c_{beacons}$. Причина, по которой сэмплирование смещено в сторону этих маяков, заключается в том, что эти маяки дают подсказку относительно положения вершин препятствий (или периферии в случае круговых препятствий). Следовательно, эти маяки должны быть окружены максимальным количеством узлов для оптимизации пути на этих поворотах. Таким образом, достигается оптимальность за значительно меньшее количество итераций по сравнению с RRT*, который достигает близкого к оптимальному решению только при приближении сэмплов к бесконечности.

По мере итерации алгоритма, каждый раз, когда находится новый путь RRT* с меньшей стоимостью по сравнению с предыдущим путем, вычисляется оптимизированный путь. Стоимость этого оптимизированного пути сравнивается с предыдущим оптимизированным путем. Если стоимость меньше, генерируются новые маяки $q_{beacons}$, и таким образом формируются новые точки смещения. Эти новые сформированные маяки ближе к вершинам. Этот процесс продолжается до тех пор, пока не будут завершены требуемые итерации.

Список литературы

1. LaValle S.M. Rapidly-exploring random trees: A new tool for path planning // Computer Science Dept, Iowa State University, Tech. Rep. TR: 98–11, 1998.

2. Karaman S., Frazzoli E. Sampling-based algorithms for optimal motion planning // The International Journal of Robotics Research. 2011. Vol. 30. No. 7. P. 846–894. DOI: 10.1177/0278364911400001.

3. Garcia I., How J.P. Improving the efficiency of rapidly-exploring random trees using a potential function planner // Proceedings of the 44th IEEE Conference on Decision and Control and the European Control Conference. Seville: IEEE, 2005. P. 7965–7970. DOI: 10.1109/CDC.2005.1582308.

4. Kuffner J., LaValle S.M. RRT-connect: An efficient approach to single-query path planning // Proceedings of the 2000 IEEE International Conference on Robotics and Automation. San Francisco: IEEE, 2000. Vol. 2. P. 995–1001. DOI: 10.1109/ROBOT.2000.844730.

5. Kavraki L.E., Svestka P., Latombe J.C., Overmars M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces // IEEE Transactions on Robotics and Automation. 1996. Vol. 12. No. 4. P. 566–580. DOI: 10.1109/70.508439.

6. Hsu D., Latombe J.C., Motwani R. Path planning in expansive configuration spaces // International Journal of Computational Geometry and Applications. 1999. Vol. 9. No. 4/5. P. 495–512. DOI: 10.1142/S0218195999000289.

7. LaValle S.M., Kuffner J.J. Randomized kinodynamic planning // The International Journal of Robotics Research. 2001. Vol. 20. No. 5. P. 378–400. DOI: 10.1177/02783640122067453.

8. Sucan I., Kavraki L.E. A sampling-based tree planner for systems with complex dynamics // IEEE Transactions on Robotics. 2012. Vol. 28. No. 1. P. 116–131. DOI: 10.1109/TRO.2011.2160466.

9. LaValle S.M., Kuffner J.J. Rapidly exploring random trees: Progress and prospects // Algorithmic and Computational Robotics: New Directions. Natick: A K Peters, 2000. P. 293–308.

10. Karaman S., Walter M.R., Perez A., Frazzoli E., Teller S. Anytime motion planning using the RRT // Proceedings of the 2011 IEEE International Conference on Robotics and Automation. Shanghai: IEEE, 2011. P. 1478–1483. DOI: 10.1109/ICRA.2011.5980270.

References

1. LaValle S.M. Rapidly-exploring random trees: A new tool for path planning // Computer Science Dept, Iowa State University, Tech. Rep. TR: 98–11, 1998.

2. Karaman S., Frazzoli E. Sampling-based algorithms for optimal motion planning // The International Journal of Robotics Research. 2011. Vol. 30. No. 7. P. 846–894. DOI: 10.1177/0278364911400001.

3. Garcia I., How J.P. Improving the efficiency of rapidly-exploring random trees using a potential function planner // Proceedings of the 44th IEEE Conference on Decision and Control and the European Control Conference. Seville: IEEE, 2005. P. 7965–7970. DOI: 10.1109/CDC.2005.1582308.

4. Kuffner J., LaValle S.M. RRT-connect: An efficient approach to single-query path planning // Proceedings of the 2000 IEEE International Conference on Robotics and Automation. San Francisco: IEEE, 2000. Vol. 2. P. 995–1001. DOI: 10.1109/ROBOT.2000.844730.

5. Kavraki L.E., Svestka P., Latombe J.C., Overmars M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces // IEEE Transactions on Robotics and Automation. 1996. Vol. 12. No. 4. P. 566–580. DOI: 10.1109/70.508439.

6. Hsu D., Latombe J.C., Motwani R. Path planning in expansive configuration spaces // International Journal of Computational Geometry and

Applications. 1999. Vol. 9. No. 4/5. P. 495–512. DOI: 10.1142/S0218195999000289.

7. LaValle S.M., Kuffner J.J. Randomized kinodynamic planning // The International Journal of Robotics Research. 2001. Vol. 20. No. 5. P. 378–400. DOI: 10.1177/02783640122067453.

8. Sucan I., Kavraki L.E. A sampling-based tree planner for systems with complex dynamics // IEEE Transactions on Robotics. 2012. Vol. 28. No. 1. P. 116–131. DOI: 10.1109/TRO.2011.2160466.

9. LaValle S.M., Kuffner J.J. Rapidly exploring random trees: Progress and prospects // Algorithmic and Computational Robotics: New Directions. Natick: A K Peters, 2000. P. 293–308.

10. Karaman S., Walter M.R., Perez A., Frazzoli E., Teller S. Anytime motion planning using the RRT // Proceedings of the 2011 IEEE International Conference on Robotics and Automation. Shanghai: IEEE, 2011. P. 1478–1483. DOI: 10.1109/ICRA.2011.5980270.