Обзор современных методов планирования движения[1]

[1]К.А. Казаков <kazakov@ispras.ru>
[1, 2]В.А. Семенов <sem@ispras.ru>
[1]Институт системного программирования РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
[2]Московский физико-технический институт,
141700, Московская область, г. Долгопрудный, Институтский пер., 9

Аннотация. Автоматизация технологически сложных процессов в машиностроении, энергетике, транспорте, медицине, строительстве, а также создание новых продуктов и сервисов невозможны без решения задач планирования движения. В последнее время интерес к ним заметно возрос в связи с развитием средств компьютерного моделирования и становлением таких дисциплин как комплексное планирование индустриальных программ, реалистичная анимация трехмерных сцен, роботизированная хирургия, навигация в динамическом окружении, автоматическая сборка продуктов, организация транспортных потоков в мегаполисах. Данная работа посвящена обзору и сравнительному анализу современных математических методов планирования движения.

Ключевые слова: планирование движения, поиск пути, маршрутные сети, определение столкновений.

DOI: 10.15514/ISPRAS-2016-28(4)-14

Для цитирования: Казаков К.А., Семенов В.А. Обзор современных методов планирования движения. Труды ИСП РАН, 2016 г., стр. 241-294. DOI: 10.15514/ISPRAS-2016-28(4)-14

1. Введение

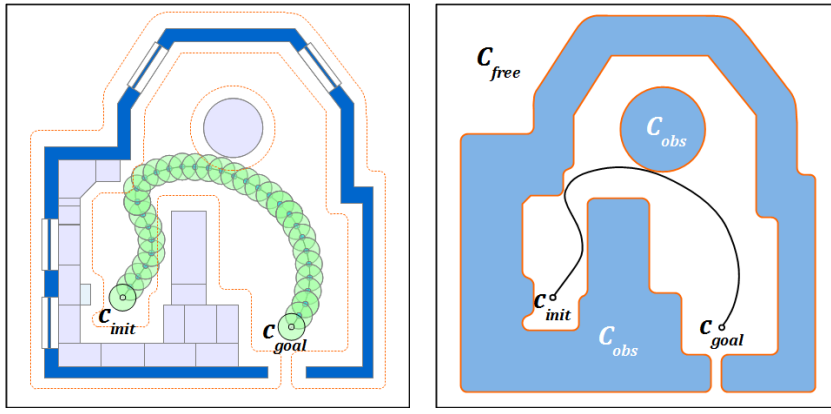
Автоматизация технологически сложных процессов в машиностроении, энергетике, транспорте, медицине, строительстве, а также создание новых продуктов и сервисов невозможны без решения задач планирования движения. В последнее время интерес к ним заметно возрос в связи с развитием средств

[1] Работа поддержана РФФИ (грант 16-07-00606)

компьютерного моделирования и становлением таких дисциплин как комплексное планирование индустриальных программ, реалистичная анимация трехмерных сцен, роботизированная хирургия, навигация в динамическом окружении, автоматическая сборка продуктов, организация транспортных потоков в мегаполисах. Данная работа посвящена обзору и сравнительному анализу современных математических моделей, методов и программных средств планирования движения [1].

Обычно под планированием движения понимается поиск бесконфликтного пути для перемещения твердого тела или кинематической конструкции в пространственно-трехмерной сцене. Искомый путь представляет собой непрерывную кривую в конфигурационном пространстве объекта, которая соединяет его начальное и конечное положения, исключает столкновения с препятствиями сцены и удовлетворяет всем установленным кинематическим и динамическим ограничениям.

Пусть задана сцена как непустое множество препятствий $O \subset W$ в области эвклидова пространства $W \subset E^N$, $N \in \{2,3\}$. Пусть также задано твердое тело $A \subset W$ либо кинематическая цепь $A\langle B,J\rangle$, где $B = \{B_1, B_2, \dots, B_n\} \subset W$ – множество твердотельных звеньев, а $J = \{J_1, J_2, \dots, J_k\}$ – множество кинематических ограничений таких, что при корректной конфигурации цепи предикаты ограничений $J_1(c), J_2(c), \dots, J_k(c)$ принимают истинное значение. Под конфигурацией $c \in C_A$ здесь понимается набор значений параметров, однозначно определяющий положение точек объекта A в пространстве сцены. Обычно используется минимальный набор параметров, соответствующий количеству степеней свободы объекта и определяющий пространство состояний или конфигурационное пространство объекта C_A.



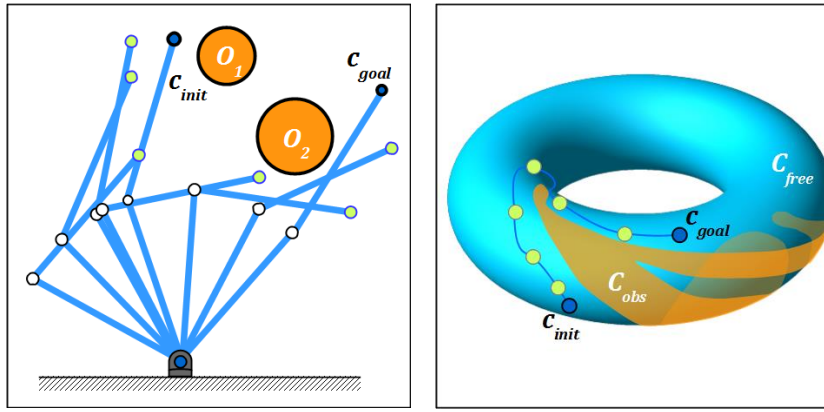
(a) (b)

Рис. 1. Конфигурационное пространства двумерного твердого тела

Fig. 1. Configuration space of two dimensional solid body

Определение. Пространством допустимых состояний назовем множество всех конфигураций объекта $c \in C_A$, удовлетворяющих кинематическим ограничениям и исключающих столкновения с препятствиями сцены $C_{free} = \{c \in C_A \mid J_1(c) \wedge J_2(c), \ldots, \wedge J_k(c) \wedge B_1(c) \cap O = \emptyset, \ldots, \wedge B_n(c) \cap O = \emptyset\}$. Для простого твердого тела свободное множество определяется как $C_{free} = \{c \in C_A \mid A(c) \cap O = \emptyset\}$.

Тогда постановка задачи поиска пути может быть сформулирована следующим образом. Для пары заданных бесконфликтных конфигураций $c_{init}, c_{goal} \in C_{free}$ требуется найти непрерывный путь $p(\tau)\colon [0,1] \to C_{free}$ такой, что $p(0) = c_{init}$ и $p(1) = c_{goal}$.



(a) *(b)*

Рис. 2. Конфигурационное пространства двухзвенного манипуляционного робота

Fig. 2. Configuration space of 2-DOF manipulation robot

Поскольку планирование маршрута, как правило, допускает бесконечное множество решений (хотя может не существовать ни одного решения), иногда данную задачу формулируют в постановке оптимизационной задачи с целевой функцией, соответствующей минимальной длине маршрута или максимальной удаленности перемещаемого объекта от препятствий [2,3].

На практике поиск пути даже в простых сценах с относительно небольшим количеством препятствий становится трудноразрешимой задачей, если перемещаемый объект имеет сложную геометрию или высокое число степеней свободы. В современных индустриальных приложениях часто требуется моделировать поведение сложных кинематических систем с шестью и более степенями свободы в статическом или динамическом окружении, насчитывающим тысячи препятствий.

Функционал существующих прикладных программных пакетов, таких как Kineo CAM (Kineo Computer Aided Motion) [4], ROS (Robot Operating System)

[5], OMPL (Open Motion Planning Library) [6], OpenRAVE (The Open Robotics Automation Virtual Environment) [7] и CuikSuite [8] главным образом обеспечивает решение задач моделирования неголономных механических систем в режиме реального времени и локального планирования движения. Математический арсенал упомянутых пакетов в основном ограничен сэмплинг-методами, которые эффективны в приложениях, связанных, в частности, с управлением движением промышленных роботов при сборке компонентов, программированием координатно-измерительных машин при контроле точности производимых изделий. Однако данные методы демонстрируют несостоятельность в тех случаях, когда требуется определение протяженных бесконфликтных траекторий в трехмерном окружении со сложной топологией. Подобные задачи возникают, например, при пространственно-временной верификации календарно-сетевых графиков архитектурно-строительных проектов и нуждаются в более развитом математическом аппарате [9–11]. При этом архитектура программных пакетов и особенности организации интерфейсов препятствуют реализации в их составе новых глобальных методов планирования движения и интеграции в индустриальные системы, ориентированные на работу с масштабными сценами, состоящими из сотен тысяч и миллионов объектов с индивидуальными геометрическими и динамическими характеристиками.

Таким образом, разработка эффективных математических методов и программных средств планирования движения представляет собой актуальную научную проблему. Настоящая статья посвящена систематическому обзору и сравнительному анализу современных математических методов планирования движения. Важное внимание уделяется ключевым подходам, основанным на пространственной декомпозиции, маршрутных сетях и физических аналогиях с потенциальными полями. Предполагается, что обзор поможет в выработке общих рекомендаций по использованию методов, а также в их выборе при решении практических классов задач. Ожидается также, что обзор послужит конструктивной основой для концептуализации теории планирования движения и создания единой программно-инструментальной среды разработки приложений.

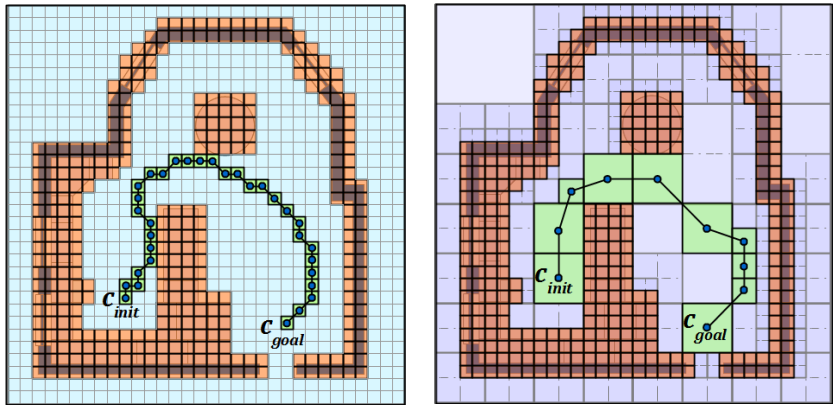
2. Методы на основе пространственной декомпозиции

Наиболее простой и в то же время распространенный подход к планированию движения состоит в применении методов пространственной декомпозиции. Он предполагает разбиение свободных областей сцены на множество простых регионов с использованием тех или иных методов декомпозиции, определение смежности регионов и формирование графа связности, который в дальнейшем может использоваться для навигации в сцене. Фактически, подход реализует схему редукции вычислительно сложной задачи планирования движения в эвклидовом пространстве к типовой задаче поиска пути в графе.

2.1 Регулярная декомпозиция

Одним из способов реализации данного подхода является применение методов регулярной пространственной декомпозиции. Данные методы предполагают разбиение всего объема сцены сеткой с фиксированным размером ячеек, ориентированных по осям координат (рис. 3, a). Для всех ячеек определяется статус занятости. Ячейки могут быть целиком заполненными объектами сцены, частично заполненными или свободными. Маршрут строится путем анализа смежных свободных ячеек, выбора направлений для распространения и проверки принадлежности перемещаемого объекта свободным ячейкам сетки. Таким образом, вместо исходной точной геометрической модели сцены используется ее упрощенное дискретное представление, что существенно упрощает процесс маршрутизации. Вопросы использования методов регулярной декомпозиции применительно к планированию движения широко освещены в литературе [12,13].

Метод относительно прост в реализации, однако требует значительных вычислительных ресурсов из-за необходимости детальной дискретизации всего пространства сцены и обеспечения приемлемой точности маршрута.



(a) (b)

Рис. 3. Пример построения декомпозиции двумерной сцены с помощью (a) равномерной сетки и (b) дерева квадрантов

Fig. 3. An example of decomposition of 2D scene using (a) regular grid and (b) quad tree

Указанного недостатка лишены методы регулярной адаптивной декомпозиции пространства, одним из известных представителей которых является метод на основе октодеревьев занятости [10,14,15]. Благодаря рекурсивной процедуре декомпозиции такие структуры оказываются более экономичными (рис. 3, b). Если принимать во внимание возможную неравномерность распределения объектов по объему сцены и существенную вариацию их размеров, октарные структуры оказываются более рациональными при дискретизации сцены и

исполнении типовых запросов, связанных с пространственной локализацией объектов, поиском соседей, определением столкновений. Однако процедура маршрутизации становится более сложной, а с учетом несбалансированности октодеревьев может приводить и к неестественным траекториям, содержащим большое число изломов. Для преодоления этой проблемы был предложен гибридный метод, использующий окаймление на границах ячеек [16]. Однако он вряд ли может рассматриваться в качестве универсального, поскольку неизбежно порождает избыточное представление сцены и требует существенных дополнительных расходов.

2.2 Объектно-зависимая декомпозиция

Альтернативу методам пространственной декомпозиции составляют методы объектно-зависимой декомпозиции пространства. За счет выделения свободных областей непосредственно по границам объектов сцены устраняются проблемы, обусловленные погрешностью дискретизации сцены. К сожалению, методы этого семейства обладают высокой вычислительной сложностью и в большинстве случаев не могут конкурировать с методами пространственной декомпозиции. Наиболее известными методами объектно-зависимой декомпозиции являются метод вертикального разбиения или трапецеидальной декомпозиции (Trapezoidal Decomposition) [17], триангулированное разбиение [18], цилиндрическая декомпозиция (Cylindrical decomposition) [19], а также декомпозиция Морса [1,20].

3. Методы на основе маршрутных сетей

Важное семейство методов глобального планирования движения составляют методы на основе маршрутных сетей, которые обычно строятся в конфигурационном пространстве перемещаемого объекта или в пространстве сцены. Подобные сети объединяют в себе бесконфликтные переходы или участки путей и расширяются по мере совершения новых успешных попыток перемещения. Иногда сети накапливают также информацию о предпринятых неудачных попытках и связанных с ними вычислительных затратах, что позволяет в дальнейшем выбирать более перспективные направления и ускорить процесс маршрутизации при решении как одиночных, так и множественных задач планирования движения.

Формально маршрутная сеть представляет собой топологический граф $G(V, E)$, вершины V которого соответствуют бесконфликтным конфигурациям перемещаемого объекта $S \subset C_{free}$, а ребра E – бесконфликтным переходам между ними $P(\tau)$: $[0,1] \to C_{free}$, $P(0) \subset S, P(1) \subset S$. Обычно маршрутная сеть строится в предположении выполнения условий достижимости и связности.

Определение. Маршрутная сеть достижима для конфигурационного пространства объекта, если для любой его точки $c \in \mathrm{C}_{free}$ существует вершина

маршрутной сети $s \in S$ и бесконфликтный путь к ней $p(\tau)\colon [0,1] \to C_{free}$, $p(0) = \mathrm{c}$ и $p(1) = s$.

Определение. Маршрутная сеть объекта связна для конфигурационного пространства объекта, если для любой его пары точек $c_{init}, c_{goal} \in \mathrm{C}_{free}$, между которыми существует бесконфликтный путь и найдены сопряженные вершины маршрутной сети $s_{init}, s_{goal} \in S$, также существует маршрут $p'(\tau)\colon [0,1] \to S$, где $p'(0) = c_{init}$ и $p'(1) = c_{goal}$.

Данные условия обеспечивают гарантированное нахождение бесконфликтного маршрута между любыми двумя положениями объекта при условии, что он существует. Первое условие позволяет соединить любую пару заданных точек свободного пространства с маршрутной сетью. Второе условие обеспечивает навигацию по маршрутной сети. К сожалению, данные условия носят декларативный характер и не могут быть конструктивно применены при разработке и реализации методов планирования движения.

Однако сама идея маршрутной сети достаточно плодотворна и нашла воплощение в ряде методов, прежде всего в методах PRM и RRT, подробно обсуждаемых в следующих разделах. Маршрутные сети, развернутые непосредственно в пространстве сцены, предоставляют дополнительные возможности для решения задач планирования движения с разными объектами. Однако в этом случае переходы, бесконфликтные для одного перемещаемого объекта, могут оказаться конфликтными для других объектов. Поэтому построенные маршруты нуждаются в дополнительной верификации, а сама сеть – в достоверных методах анализа переходов для маршрутизации новых объектов.

3.1 Графы видимости

Один из известных методов организации маршрутных сетей основан на применении графов видимости (Visibility Graphs) [18]. В простейших случаях граф видимости строится на множестве вершин полигонов или полиэдров, являющихся геометрическими моделями препятствий сцены, и дополняется начальными и конечными точками маршрутов. Все вершины попарно соединяются линейными отрезками, которые принимаются в качестве ребер графа при условии попарной “видимости” инцидентных им вершин и отсутствия пересечения с препятствиями сцены (рис. 4, а). Сложность построения графа видимости для сцены, представленной полигонами с общим числом вершин n, составляет $O(n^2 log\ n)$ [1], что является довольно затратным для применения в индустриальных приложениях.

Кроме того, преобразование топологического графа в маршрутную сеть также представляет собой сложную задачу, поскольку ребра графа видимости проходят точно через вершины препятствий, и для успешной навигации требуется коррекция путей или предварительное расширение границ препятствий. Избыточное количество порождаемых маршрутов, крайне

нежелательное при моделировании сложных сцен, также является недостатком данного метода. Тем не менее, в двумерном окружении с относительно небольшим числом препятствий данный метод может успешно применяться, например, для решения задач поиска кратчайшего пути в сцене. Более того, существуют попытки использования графов видимости для планирования движения в динамическом окружении [21,22].

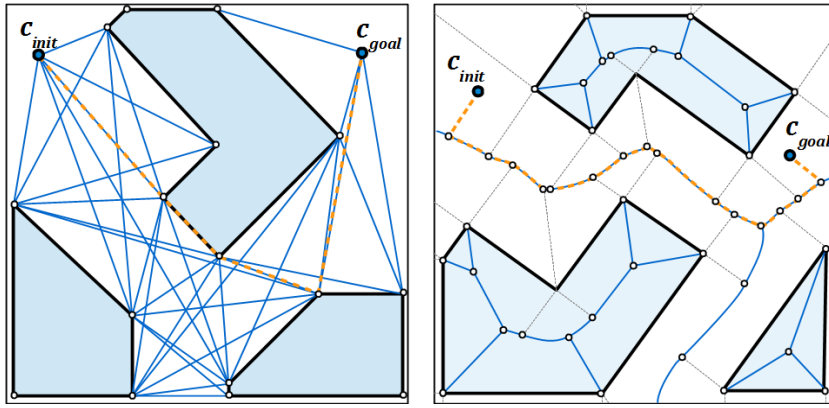


Рис. 4. Пример построения (a) графа видимости и (b) диаграммы Вороного в двумерной полигональной сцене

Fig. 4. An example of (a) visibility graph and (b) Voronoi diagram in two dimensional polygon scene

3.2 Диаграммы Вороного

Альтернативный метод определения маршрутов основан на использовании обобщенных диаграмм Вороного (Generalized Voronoi Diagrams) [1]. Наиболее привлекательное свойство диаграммы Вороного заключается в том, что она является деформационным ретрактом свободного пространства. Тем самым, любой бесконфликтный путь может быть непрерывно отображен в диаграмму Вороного. Более того, метод обеспечивает наибольшее удаление от границ препятствий при условии, что движение осуществляется вдоль ребер диаграммы. Данное свойство позволяет определять наиболее “безопасные” маршруты перемещения в сцене (рис. 4, b).

Наивная реализация данного метода обладает высокой вычислительной сложностью $O(n)^4$, где n – общее количество вершин и граней препятствий [23]. Известны эффективные алгоритмы с лучшими асимптотическими оценками сложности, однако они неустойчивы к вырожденным случаям и трудны для надежной реализации.

Диаграмма Вороного определяется для произвольного метрического пространства. Несмотря на это, ее применение для навигации в эвклидовых пространствах размерности выше двух является затруднительным ввиду того, что маршрутная сеть уже не представляет топологически одномерную структуру. В этих случаях обычно применяются методы на основе обобщенных диаграмм Вороного [24].

4. Дискретный поиск

Методы на основе пространственной декомпозиции и маршрутных сетей позволяют редуцировать вычислительно трудные задачи навигации в сложном многомерном окружении к более простым задачам теории графов, для решения которых имеется развитый математический аппарат. Тем не менее, задачи планирования движения привносят свои особенности в постановку и решение подобных задач [25–27].

Например, маршрутные сети не только представляют топологию свободных областей сцены, но и часто накапливают дополнительную информацию для выбора более достоверных и оптимальных маршрутов навигации. Такими данными могут быть, например, протяженность переходов в маршрутной сети, расстояние до ближайших препятствий сцены, информация о предпринятых удачных и неудачных попытках расширения сети и затраченных для этого вычислительных ресурсах. Подобные данные позволяют использовать эвристические оценки рисков и перспективности маршрутов и, тем самым, ускорить решение одиночных и множественных задач маршрутизации.

4.1 Неинформированный поиск

Классические алгоритмы теории графов реализуют поиск в ширину, поиск в глубину, поиск по критерию стоимости [25]. Данные алгоритмы предполагают полный обход вершин графа без какой-либо информации о том, насколько каждый шаг приближает процесс поиска к решению. Применение данных алгоритмов сопряжено со значительными вычислительными затратами, и поэтому они редко применяются при решении задач планирования движения.

4.2 Эвристический поиск

Эвристические алгоритмы поиска пути предназначены для быстрого отыскания маршрута в графе путем распространения в направлении более перспективных вершин. Для этого обычно организуется очередь из непосещенных вершин, порядок в которой определяется на основе эвристических правил приоритизации.

Одним из наиболее популярных алгоритмов поиска пути данного семейства является A* [28]. Приоритет вершины в данном алгоритме определяется суммой: $f(v) = g(v) + h(v)$, где $g(v)$ – стоимость пути из начальной вершины в данную, а $h(v)$ – эвристическая оценка стоимости пути до целевой вершины.

При этом значение $h(v)$ должно быть неотрицательным и удовлетворять неравенству треугольника. Таким образом, A* является обобщением алгоритма Дейкстры при $f(v) = g(v)$ и алгоритма жадного поиска при $f(v) = h(v)$.

Использование эвристики, как правило, позволяет значительно уменьшить количество посещаемых вершин и, как следствие, повысить производительность поиска. Кроме того, алгоритм находит оптимальные решения при условии, что эвристическая оценка является оптимистической, т.е. значение функции $h(v)$ всегда меньше либо равно точной стоимости пути до целевой вершины.

В наихудшем случае выполнение алгоритма A* может быть сопряжено с экспоненциальным ростом числа непосещенных вершин в очереди, что при большом коэффициенте ветвления дерева поиска порождает проблему чрезмерного потребления памяти. В таких случаях более предпочтительным является использование алгоритма A* с итеративным углублением (Iterative Deeping A*) [29] или алгоритма A* с ограничением памяти (Memory Bounded A*) [30].

4.3 Поиск в регулярной сетке

Методы пространственной декомпозиции на основе регулярных равномерных сеток приводят к дискретному представлению сцены в виде графов с большим количеством вершин и неопределенностью в их приоритизации из-за равной стоимости переходов. Одним из способов повысить эффективность алгоритмов в таких случаях является привлечение идей иерархического поиска. Алгоритмы Hierarchical A* [31] и HPA* [32,33] реализуют данные идеи в результате построения многоуровневого иерархического представления сцены и группировки смежных ячеек сетки. Данное представление затем применяется для построения путей с необходимой степенью детализации на каждом уровне иерархии.

Альтернативный алгоритм Jump Point Search, не требующий дополнительных расходов памяти, предложен в работе [34]. Он использует набор простых эвристических правил для выбора перспективных направлений, в которых могут быть предприняты длинные прямолинейные и диагональные переходы (“прыжки”). В результате из анализа исключается значительное число вершин, не влияющих на стоимость переходов и качество конечного решения, а эффективность поиска пути существенно повышается.

4.4 Алгоритм Lifelong Planning A*

Маршрутные сети, построенные в пространстве сцены для некоторых типовых объектов, могут содержать переходы, непреодолимые для объектов нового типа. В подобных случаях следует попытаться проложить новый маршрут, исключающий конфликтные переходы. Наивный алгоритм мог бы состоять в повторении процедуры поиска в измененном графе с самого начала. Однако с

вычислительной точки зрения такой способ не является рациональным, поскольку не учитывает предшествующие результаты.

Алгоритм LPA* [35] является развитием A* для инкрементального поиска кратчайшего пути в условиях динамического изменения весов ребер. Для каждой вершины алгоритм предусматривает хранение ее фактической стоимости $g(v)$ и предварительной оценки стоимости в соответствии со следующей формулой:

$$rhs(v) = \begin{cases} 0 & , \ v = v_{initial} \\ min_{v' \in pred(v)}\big(g(v') + c(v', v)\big), & v \neq v_{initial} \end{cases},$$

где $pred(v)$ – множество ранее посещенных смежных вершин, а $c(v', v)$ – стоимость перехода из вершины v' в вершину v. Для непосещенных вершин $rhs(v) = g(v) = \infty$.

Приоритетная очередь LPA* содержит множество локально неустойчивых вершин, в которых оценка расходится с фактической стоимостью $g(v) \neq rhs(v)$. Приоритет определяется составным ключом $k(v) = [k_1(v), k_2(v)]$, где $k_1(v) = min\big(g(v), rhs(v)\big) + h(v)$, $k_2(v) = min\big(g(v), rhs(v)\big)$, а $h(v)$ – эвристическая оценка стоимости пути до целевой вершины.

На каждом шаге построения пути из очереди выбирается вершина с минимальным ключом. Если вершина является недооцененной $g(v) < rhs(v)$ и значит изменения могли повлиять на ее стоимость, то она помечается как непосещенная $g(v) = \infty$. В противном случае, если вершина является переоцененной $g(v) > rhs(v)$, то она переводится в локально устойчивое состояние $g(v) = rhs(v)$. Далее производится переоценка смежных вершин. При любом изменении веса одного из ребер графа переоцениваются инцидентные ему вершины, а локально неустойчивые вершины добавляются к приоритетной очереди. Поиск останавливается, если выполняется условие локальной устойчивости целевой вершины или условие, при котором значение ключа целевой вершины оказалось меньше минимального значения в приоритетной очереди.

4.5 Алгоритм D*

Часто в приложениях планирования движения возникает необходимость поиска пути в сценах с неполной, перманентно обновляемой информацией. Примером может служить задача навигации мобильного робота, получающего информацию об окружении с датчиков в режиме реального времени. При установленных изменениях в окружении робота следует перепланировать его маршрут. Для этих целей более подходят такие алгоритмы, как Focussed D* [36] и D*-light [37]. Последний использует принципы, лежащие в основе LPA* и, как правило, демонстрирует лучшую производительность. В работе [38] отмечается простота его реализации.

5. Метод потенциальных полей

Важное семейство методов планирования движения составляют методы потенциальных полей, первоначально разработанные для навигации мобильных роботов и обхода локальных препятствий в режиме реального времени [39]. Методы основаны на физической аналогии с движением заряженной частицы в электростатическом поле. Препятствия сцены генерируют отталкивающие силы, а целевая точка маршрута – значительную притягивающую силу. Направление и скорость движения тела определяются градиентом потенциального поля (рис. 5).

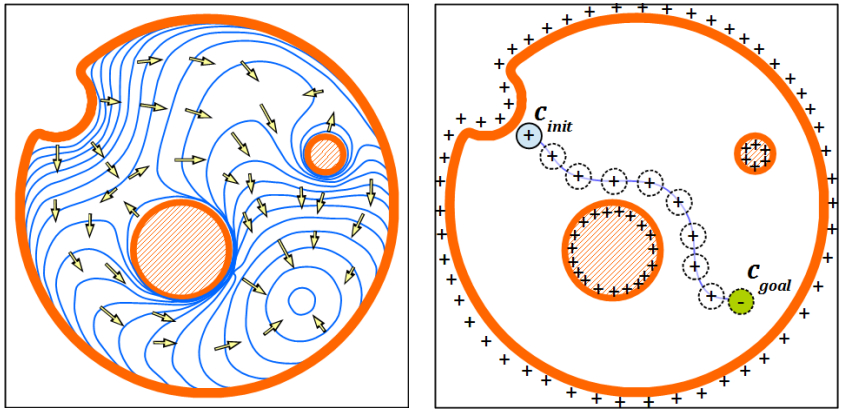


Рис. 5. Движение заряженной частицы в электростатическом поле

Fig. 5. The movement of charged particle in electrostatic field

Навигационная функция, как правило, задается суммой притягивающих и отталкивающих потенциалов: $\mathrm{U(c)} = \mathrm{U_{att}(c)} + \mathrm{U_{rep}(c)}$. При продвижении робота к целевой точке c_{goal} функция притягивающего потенциала $\mathrm{U_{att}}$ должна монотонно возрастать, однако ее конкретный вид может существенно варьироваться. Например, в работе [1] предлагается следующая формула:

$$U_{att}(\mathrm{c}) = \begin{cases} \frac{1}{2}\zeta d^2(\mathrm{c}, \mathrm{c}_{goal}), & d(\mathrm{c}, \mathrm{c}_{goal}) \le D^* \\ D^*\zeta d(\mathrm{c}, \mathrm{c}_{goal}) - \frac{1}{2}\zeta (D^*)^2, & d(\mathrm{c}, \mathrm{c}_{goal}) > D^* \end{cases},$$

где d – функция, определяющая расстояние между точками, ζ – масштабирующий коэффициент. D^* является константным параметром, который определяет пороговое значение расстояния до цели, при достижении которого скорость движения объекта начинает снижаться. Таким способом удается избежать проблемы, связанной с “дрожанием” объекта вблизи целевой точки пути при недостаточно малом временном шаге.

Функция отталкивающего потенциала, формируемого множеством препятствий $O = \{o_1, o_2, \dots o_N\}$, задается следующим образом:

$$U_{rep}(c) = \sum_{i=1}^{N} U_{rep_i}(c, o_i),$$

$$U_{rep_i}(c, o_i) = \begin{cases} \frac{1}{2} n(\frac{1}{d_{min}(c,o_i)} - \frac{1}{D'})^2, & d_{min}(c, o_i) \le D' \\ 0, & d_{min}(c, o_i) > D' \end{cases},$$

где минимальное расстояние от заданной точки до препятствия задается функцией $d_{min}(\mathrm{c}, o) = min_{p \in o} \|c - p\|$, а константный параметр D' определяет предельное расстояние от заданной точки до удаленных препятствий сцены, вклад от которых учитывается при подсчете потенциала.

5.1 Метод рандомизированных потенциальных полей

Основным недостатком методов потенциальных полей, использующих градиентный спуск, является проблема локальных минимумов. Один из способов ее преодоления заключается в попытке “вытолкнуть” перемещаемый объект в случайном направлении при попадании в потенциальную яму. Данную идею реализует известный метод рандомизированных потенциальных полей (Randomized Potential Fields) [40,41]. Она же получила развитие в семействе сэмплинг-методов, подробно обсуждаемых в следующих разделах.

6. Сэмплинг методы

Применение методов планирования движения, основанных на точной или приближенной реконструкции пространства допустимых конфигураций, зачастую невозможно из-за высокой вычислительной сложности, которая становится критичной для сцен с большим количеством препятствий и перемещаемых объектов со значительным количеством степеней свободы. Избежать данной проблемы позволяют популярные на сегодняшний день сэмплинг-методы планирования движения, часто ассоциируемые с методами квази-Монте Карло.

Они предусматривают исследование областей конфигурационного пространства путем многократных испытаний, результатом которых является вердикт о допустимости отдельно выбранных конфигураций. Это обычно осуществляется путем генерации случайных конфигураций и проверки для них кинематических ограничений и столкновений с препятствиями сцены. На множестве полученных в результате сэмплирования бесконфликтных конфигураций строится маршрутная сеть. Важным достоинством методов данного семейства является независимость от геометрического представления и размерности моделируемого окружения.

Недетерминированный характер построения маршрута привносит ряд особенностей, связанных с невозможностью обеспечения полноты алгоритмов и оптимальности найденных решений. Вместе с тем, эвристические стратегии сэмплирования обеспечивают перманентный рост вероятности нахождения бесконфликтного пути с увеличением числа итераций, естественно, если

решение существует (в противном случае алгоритм может выполняться бесконечно долго). В связи с этим вводится понятие вероятностной успешности алгоритма.

Определение. Алгоритм называется вероятностно успешным, если для любой задачи поиска пути $\langle C_{free}, c_{init,} c_{goal} \rangle$, имеющей решение, имеет место $lim_{N\to\infty} P_{success} = 1$, где $P_{success}$ – вероятность найти бесконфликтный путь $p(\tau)$: $[0,1] \to C_{free}$, $p(0) = c_{init}$, $p(1) = c_{goal}$ за N итераций.

Современные алгоритмы, основанные на сэмплировании пространства, как правило, справляются с решением типовых задач поиска пути за приемлемое время. Кроме того, их применение позволяет во многих случаях компенсировать нежелательные эффекты, связанные с неизбежными погрешностями представления геометрических моделей.

Методы сэмплирования представлены двумя основными группами, а именно: методами вероятностных маршрутных сетей [42] и методами на основе деревьев поиска [43–45]. Методы успешно применяются на практике для планирования движения в пространствах высокой размерности. В частности, они широко используются для моделирования движения твердых и деформируемых тел [46,47], а также для анализа неголономных систем со сложными кинематическими связями [48,49] и дифференциальными ограничениями [50].

6.1 Выборка конфигураций

Вообще говоря, множество допустимых состояний объекта в непрерывном конфигурационном пространстве, как правило, является бесконечным. Однако маршрутная сеть должна строится за разумное число испытаний, ограниченное как размером сети, так и успешностью выбора точек при ее построении. В связи с этим, стратегия сэмплирования может являться ключевым фактором эффективности обсуждаемых методов.

В идеальном случае необходимо, чтобы множество выбранных конфигураций было плотным, чтобы отражать все особенности допустимых и недопустимых областей в конфигурационном пространстве. На интуитивном уровне это означает, что наибольшая несэмплированная область пространства должна быть как можно меньше. Следующие определения параметров дисперсии и расхождения обычно используются для оценки качества выборки конфигураций [19].

Определение. Дисперсией выборки точек P в метрическом пространстве $\langle X, dist \rangle$ называется величина, определяемая формулой $\delta(P) = sup_{x\in X} \left\{ min_{p\in P} \{dist(x,p)\} \right\}$.

Рис. 6 дает ее геометрическую интерпретацию в двумерном пространстве с метриками L_2 и L_∞.

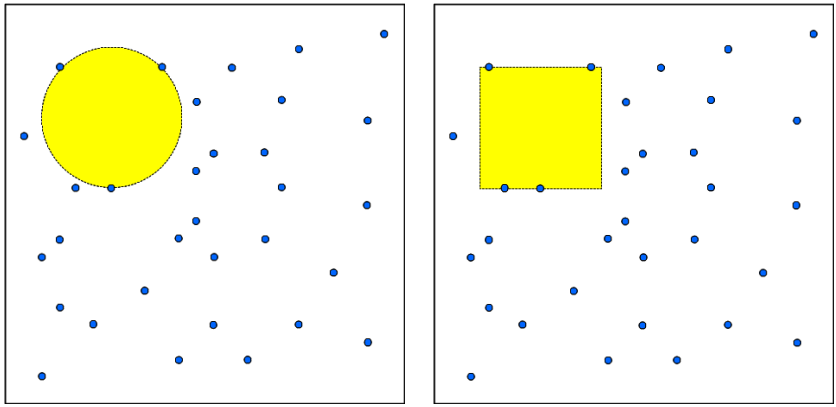


Рис. 6. Геометрическая интерпретация дисперсии выборки в метрическом пространстве L_2(a) и L_∞ (b)

Fig. 6. Geometric interpretation of the dispersion of the set of samples in metric space L_2(a) and L_∞ (b)

Параметр расхождения определяет меру того, насколько выбранные точки равномерно распределены в заданном объеме.

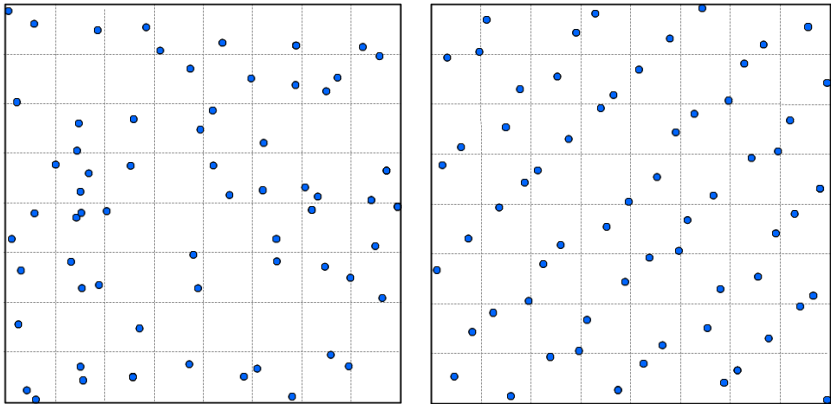
Определение. Пусть заданы компактная область $D \subset X$ в n-мерном метрическом пространстве $\langle X, dist \rangle$ и набор прямоугольников в ней $R = \{r_1, r_2, \dots, r_M\} \subset D$. Расхождение для выборки точек $P = \{p_1, p_2, \dots, p_N\}$ определяется как $\theta(P,R) = sup_{r_i \in R} \left| \frac{|P \cap r_i|}{N} - \frac{\mu(r_i)}{\mu(D)} \right|$, где $\mu(r)$ – функция объема области r.

Точки могут выбираться случайно с равномерным распределением. Однако ввиду недостатков генераторов псевдослучайных чисел данный подход, как правило, не позволяет достичь достаточно равномерного и плотного покрытия за небольшое число итераций. Улучшить качество выборки, выражаемое введенными параметрами, можно за счет использования квазислучайных последовательностей [51].

Известно, что алгоритмы квази-Монте Карло, использующие квазислучайные последовательности точек вместо псевдослучайных чисел, сходятся к искомому результату по крайней мере не медленнее, а обычно быстрее, чем соответствующие алгоритмы Монте Карло [52]. Примером таких последовательностей точек могут служить последовательности Холтона и ЛП$_\tau$-последовательности [53,54].

На рис. 7 представлен единичный квадрат, который разбит на 64 подквадрата и на него нанесены 64 псевдослучайных точки (a) и точки ЛП$_\tau$-последовательности (b). Видно, что в каждый квадрат на правом рисунке

попало ровно по одной точке, в то время, как для левого рисунка это далеко не так.



(a) *(b)*

Рис. 7. Примеры выборок с использованием псевдослучайных (a) и квазислучайных точек (b)

Fig. 7. An example of pseudorandom (a) and quasirandom sampling (b)

Вопросы применения квазислучайных последовательностей в алгоритмах планирования движения подробно рассматриваются в работах [55,56].

6.2 Метрика конфигурационного пространства

Во всех методах маршрутных сетей и сэмплирования конфигурационного пространства требуется определять расстояние между точками выборки. В частности, это требуется при оценке дисперсии выборки в локальной области конфигурационного пространства. Вопрос о выборе функции метрики оказывается нетривиальным, поскольку влияет не только на частные характеристики выборок, но и на качество результирующей маршрутной сети в целом.

Функция метрики может быть реализована путем вычисления евклидова расстояния между заданными конфигурациям по обобщенным координатам. Однако, использование такой метрики оправдано лишь в простых случаях, когда перемещение объекта ограничено поступательным движением. В более общем случае, допускающем группы вращений, например, SE(3), правильно оценить перемещение объекта без учета его геометрической модели уже невозможно.

Наиболее удачный способ оценки расстояния между точками в конфигурационном пространстве заключается в определении наибольшего пути, проходимого точками тела в пространстве сцены (Robot Displacement) [57]. Метрика задается следующим образом: $\rho(c_1, c_2) = \max_{p \in A}\{\|p(c_2) -$

$p(c_1)\|\}$, где $p(c_i)$ – положение точки тела A в конфигурации c_i. Данная метрика позволяет оценить наибольшее расстояние, проходимое каждой точкой твердого тела или любого компонента кинематической системы в пространстве сцены при переходе из положения q_1 в положение q_2 (рис. 8).

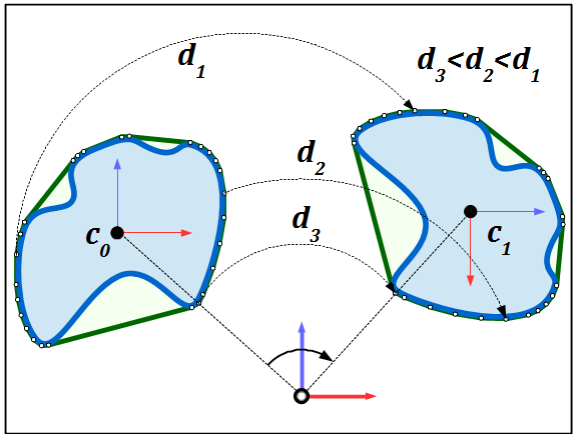


Рис. 8. Определение расстояния между точками в конфигурационном пространстве SO(2)

Fig. 8. Distance calculation between points in SO(2) configuration space

Очевидно, что вычислительные затраты при использовании такой метрики существенно зависят от сложности геометрического представления объекта и его кинематических связей. Для выпуклого твердого тела, заданного полигональной поверхностью, метрика ρ определяется расстоянием, которое проходят только его вершины. Поэтому один из способов ее вычисления заключается в использовании предварительно построенных выпуклых оболочек или ограничивающих параллелепипедов.

6.3 Определение столкновений

В ходе сэмплирования, предусматриваемого методами маршрутных сетей, необходимо определить статус допустимости или бесконфликтности каждой выбранной конфигурации объекта, в том числе, с учетом препятствий сцены. Данная задача относится к известной проблеме определения столкновений в сцене [58,59].

Как правило, определение столкновений представляет собой процесс, условно состоящий из двух основных фаз: широкой и узкой. Задачей первой фазы является быстрая локализация потенциальных столкновений с использованием относительно дешевых негативных тестов. Она эффективно решается с помощью методов ограничивающих объемов [60] и методов пространственной

декомпозиции [61–63]. В узкой фазе выполняется точная проверка выбранных пар объектов на пересечение с учетом их детального геометрического представления. Существует довольно развитый математический аппарат, позволяющий эффективно выполнять данную операцию для различных видов геометрических моделей, в том числе для полиэдральных моделей [59], сплайн-поверхностей [64], твердых тел в граничном представлении [65], CSG-моделей [66] и облаков точек [67].

Формально процедуру определения столкновений можно рассматривать как исчисление предиката $\varphi(\mathrm{c}) = \begin{cases} false, \mathrm{c} \in C_{free}, \\ true, otherwise \end{cases}$ для заданной конфигурации объекта $\mathrm{c} \in C_{free}$. Однако для объединения найденных допустимых конфигураций в маршрутную сеть требуется разрешение более сложных запросов, связанных с существованием бесконфликтных переходов между ними $p(\tau)$: $[0,1] \rightarrow C_{free}$.

На практике данная проблема может быть решена путем соединения пар точек конфигураций непрерывными переходами и анализа их бесконфликтности. Для этого переходы могут быть разбиты на множество отрезков с некоторым фиксированным шагом, а каждый из концов $p(\tau_i)$, $1 \leq i \leq N$ проверен на принадлежность пространству допустимых конфигураций. Выбор шага разбиения представляет отдельную проблему. Недостаточное количество выбранных точек приведет к тому, что перемещаемый объект будет “пролетать” сквозь препятствия. Чтобы избежать данной проблемы, обычно накладывается условие $\|p(\tau_i), p(\tau_{i+1}))\| \leq \varepsilon$, для всех $0 \leq i < N$, которое определяет точность дискретного представления перехода.

Повысить эффективность анализа переходов удается с помощью построения протяжек ограничивающих объемов, а также при использовании результатов проверок предыдущих точек переходов [68].

6.4 Поиск ближайших соседей

Шансы на то, что с помощью простого алгоритма поиска удастся проложить бесконфликтный путь между удаленными друг от друга конфигурациями, как правило, крайне малы. Поэтому построение маршрутной сети на множестве допустимых конфигураций производится путем соединения преимущественно близлежащих точек. Существенно повысить производительность поиска ближайших конфигураций возможно с помощью их предварительного индексирования. В частности, удается эффективно разрешать типовые запросы, связанные с поиском k ближайших точек и поиск ближайших точек в заданном радиусе, за сублинейное время.

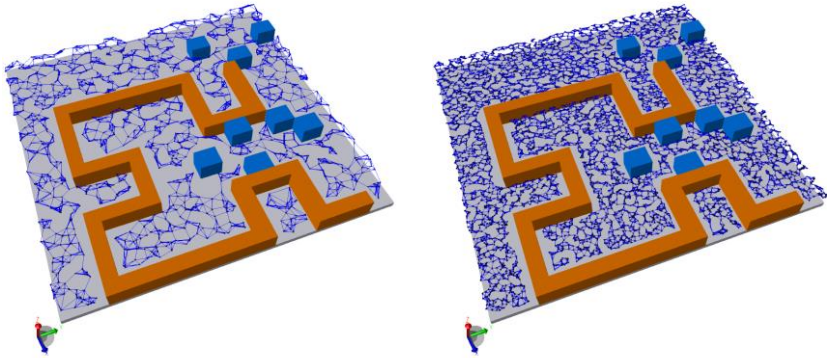
Наиболее широко используемыми для этих целей индексными структурами являются бинарные kd-деревья [69–71], которые строятся путем рекурсивного разбиения пространства гиперплоскостями, ориентированными по осям координат.

Альтернативу им составляют метрические деревья, такие как, M-деревья [72], GNAT (Geometric Near-neighbor Access Tree) [73,74], iDistance-структуры [75] и деревья покрытий (Cover Tree) [76], в основе построения которых лежат алгоритмы иерархической кластеризации. Например, в ходе построения структуры GNAT формирование узлов на каждом уровне дерева выполняется путем выбора фиксированного числа опорных точек с помощью жадного рандомизированного алгоритма *k*-средних. При этом принадлежность точек ветвям дерева определяется на основе матрицы расстояний до центров кластеров.

При решении задач планирования движения *kd*-деревья демонстрируют лучшую производительность по сравнению с другими упомянутыми выше структурами [69]. Однако их практическая реализация сталкивается с проблемой адаптации процедуры пространственной декомпозиции к конкретному способу представления трансформаций объектов. Сопутствующая задача построения ограничивающих параллелепипедов становится нетривиальной в сложных конфигурационных пространствах, включающих подпространства вращения [77]. Наиболее простыми с реализационной точки зрения являются метрические деревья, для реализации которых требуется определить функцию расстояния между конфигурациями. При этом обеспечивается хорошая сбалансированность результирующей структуры. К недостаткам метрических деревьев следует отнести необходимость выбора опорных точек, от расположения и количества которых может существенно зависеть эффективность поиска ближайших соседей.

7 Оффлайн планирование. Методы вероятностных маршрутных сетей

Метод вероятностных маршрутных сетей (Probabilistic Roadmap Method) [42] широко применяется для решения задач поиска путей как в локальной, так и глобальной постановке. Как и в случае ранее рассмотренных детерминированных методов маршрутных сетей, процесс планирования движения включает фазу анализа сцены и фазу построения пути. Отличительной особенностью в данном случае является способ формирования сети из локальных маршрутов, полученных в результате сэмплирования конфигурационного пространства.



(a) *(b)*

Рис. 9. Пример построения вероятностной маршрутной сети (k=5)

((a) 2000 итераций, (b)10000 итераций)

Fig. 9. An example of probabilistic roadmap (k=5)

((a) 2000 iterations, (b)10000 iterations)

В листинге 1 приводится обобщенный алгоритм построения маршрутной сети сэмплинг методом. На каждом из N_{steps} шаге алгоритма генерируется точка c_{rand} в конфигурационном пространстве объекта планирования. Способ генерации новой точки определяется предопределенной стратегией сэмплирования, например, с помощью выбора точки случайным образом с равномерным распределением координат в границах конфигурационного пространства $D.bounds$. Затем выполняется проверка найденной конфигурации на столкновения с препятствиями сцены. Если точка оказывается конфликтной, то она отбрасывается. В успешном случае она включается в сформированное на текущий момент представление вершин маршрутной сети $G.V$. На следующей фазе проводится попытка расширить множество ребер маршрутной сети $G.E$ путем отыскаяни бесконфликтных переходов между ее вершинами. В простейшем случае определяется k ближайших вершин относительно точки c_{rand}. В качестве альтернативного способа могут быть определены все вершины, лежащие внутри n-мерного шара радиуса r с центром в c_{rand}. Значения данных параметров могут фиксироваться или адаптивно меняться в ходе детализации маршрутной сети. Возможность создания ребра с одной из ближайших вершин определяется локальным планировщиком, в качестве которого обычно используют относительно простой и быстрый эвристический алгоритм поиска прямолинейного бесконфликтного пути. Алгоритмические особенности его реализации обсуждались в предыдущем разделе. Однако в ряде случаев возможно применение и более сложных методов. Анализ эффективности методов

локального планирования при построении вероятностных маршрутных сетей приводится в работе [78].

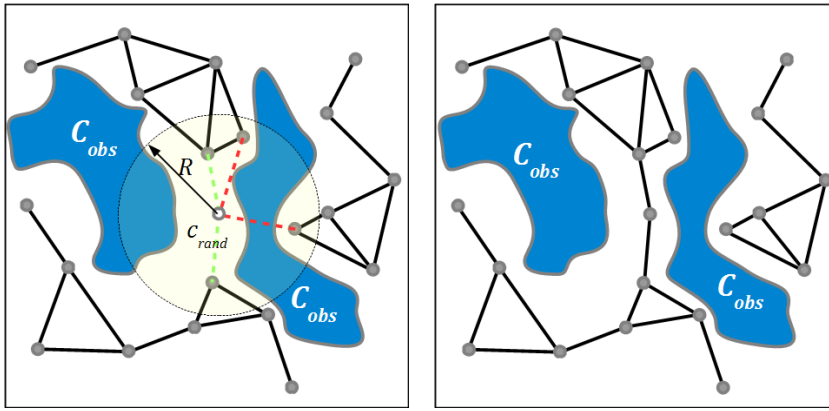


Рис. 10. Включение новой конфигурации маршрутную сеть алгоритмом PRM

Fig. 10. Construction of new roadmap vertex by PRM algorithm

BuildPRM *(N_{steps})*
1. $G(V,E) \leftarrow \emptyset$
2. $step \leftarrow 0$
3. ***while*** *(step <* N_{steps}*)*
4. $c_{rand} \leftarrow SamplingStrategy.GenerateState(D.bounds)$
5. ***if*** *(*$c_{rand} \in C_{free}$*)*
6. $G.V \leftarrow G.V \cup c_{rand}$
7. $U \leftarrow NearestNeighbours(c_{rand}, G)$
8. ***for each*** $c_{neighbour} \in U$
9. ***if*** *(LocalPlanner.VerifyPath(*c_{rand}*,*$c_{neighbour}$*))*
10. $G.E \leftarrow G.E \cup (c_{rand}, c_{neighbour})$
11. $step = step \leftarrow 1$

Листинг 1. Алгоритм построения вероятностной маршрутной сети

Algorithm. 1. An algorithm of probabilistic roadmap construction

Известный недостаток рассмотренного метода связан с недостаточно плотным покрытием, что может приводить к нарушению связности маршрутной сети в узких областях допустимых конфигураций. Решение проблемы путем простого увеличения количества испытаний крайне нежелательно, поскольку определение столкновений является вычислительно затратной операцией. Кроме того, успешные испытания приводят к разрастанию маршрутной сети и увеличению размерности решаемых задач поиска соседних вершин и

верификации переходов. Использование индексных структур, в частности *kd*-деревьев, отчасти решает проблему деградации производительности.

Другим недостатком метода является сложность выбора параметров, таких как радиус поиска соседних вершин и предельное количество инцидентных ребер. Данные параметры существенно влияют на качество маршрутной сети и часто требуют тонкой настройки с учетом специфики прикладной задачи.

7.1 Visibility PRM

Одним из способов уменьшения числа вершин в графе является метод построения рандомизированных маршрутных сетей на основе областей видимости [79]. Данный метод позволяет исключить из анализа избыточные точки, принадлежащие обширным областям допустимых конфигураций, при этом выделив ключевые точки, существенные для навигации в труднопреодолимых зонах.

Для этого в ходе формирования маршрутной сети вершины сети классифицируются как guard (страж) и connector (звено). Случайно выбранная точка относится к первому классу, если не существует вершины, помеченной как guard и лежащей в ее области видимости. Точка классифицируется как connector, если она попадает в область видимости, по меньшей мере, двух вершин guard. И наконец, если точку не удалось отнести ни к одному из двух классов, она вовсе не включается в маршрутную сеть. Тем самым достигается существенное сокращение размера маршрутной сети.

К недостаткам метода следует отнести тот факт, что положение опорных точек класса guard формируется на основе случайного выбора. Поэтому области допустимых конфигураций могут быть построены и распределены неоптимальным образом, что может негативно сказаться на эффективности поиска маршрутов в определенных зонах.

7.2 Vertex Enhancement

Для идентификации слабо связных зон пространства допустимых конфигураций и детализации маршрутной сети в них в основную схему метода часто добавляют процедуру пост-обработки присоединяемых к сети вершин и ребер. В работе [42] для этих целей предлагается использовать информацию о неудачных попытках присоединения вершин, накапливаемую непосредственно в процессе построения сети.

Вес вершины v маршрутной сети рассчитывается на основе статистики испытаний: $s(v) = \frac{n_f}{n_t+1}$, где n_t – общее число попыток присоединения точек к v, а n_f – количество неудавшихся попыток. Таким образом, значение $s(v)$ отражает условную сложность сцены в окрестности v. Распространяя данную оценку на множество всех вершин сети V, можно определить функцию

распределения вероятностей неуспешных попыток как $P(v) = \frac{s(v)}{\sum_{v \in V} s(v)}$ и ее использовать для генерации дополнительных точек в проблемных зонах.

7.3 Obstacle-based PRM

В ряде случаев, например, при наличии узких переходов между обширными областями допустимых конфигураций, представляется разумным выбирать точки преимущественно вдоль границ препятствий и, тем самым, снижать общее количество испытаний и размер сети. Существует ряд модификаций метода построения вероятностных маршрутных сетей, реализующих данный принцип.

В методе Obstacle-based PRM, описанном в работах [80,81], для этих целей предлагается модифицировать базовый алгоритм сэмплирования следующим образом. На каждом шаге выбирается точка c_{rand} с равномерным распределением в границах конфигурационного пространства. Если найдена недопустимая конфигурация $c_{rand} \notin C_{free}$, то проводится некоторое количество попыток обнаружить в ее окрестности допустимую конфигурацию $c' \in C_{free}$. На интервале прямой (c_{rand}, c') ищется новая точка $c_{new} \in C_{free}$, которая лежит как можно ближе к границе препятствия. Чтобы найти такую точку, можно использовать алгоритмы разбиения интервала пополам, золотого сечения или квадратичной интерполяции.

Позже была предложена модификация метода UOBPRM, основанная на использовании ограничивающих параллелепипедов объектов сцены и позволяющая получать бесконфликтные точки с равномерным распределением вдоль границ препятствий [82].

7.4 Bridge-test sampling

Аналогичный принцип применяется в эвристической стратегии сэмплирования узких проходов (Bridge-test Sampling) [83]. Данная стратегия основана на предположении о том, что если для случайно выбранной точки $c \in C_{obs}$ существует отрезок проходящей через нее прямой $s(t) \subset C$ такой, что $c = s(t_0)$, $c' = s(t_1) \in C_{free}$ и $c'' = s(t_2) \notin C_{free}$ при $t_0 < t_1 < t_2$, то точка c' предположительно находится внутри узкого прохода и должна быть включена в маршрутную сеть.

Стоит отметить, что данная стратегия, как правило, отсекает большую часть случайно сгенерированных конфигураций. Однако вычислительные затраты на определение допустимости отдельных избыточных конфигураций существенно ниже, чем на построение и анализ бесконфликтных переходов между ними.

7.5 Gaussian PRM

Другие модификации базового алгоритма используют Гауссово распределение для формирования выборок с большей плотностью вблизи границ препятствий

[84]. По аналогии с формулой размытия, применяемой в алгоритмах обработки изображений, нормальное распределение Гаусса в конфигурационном пространстве C определяется как $\varphi(\mathrm{c},\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}^n} e^{-\frac{\mathrm{c}^2}{2\sigma^2}}$, где $\mathrm{c} \in \mathrm{C}$, n – размерность пространства, а σ – среднеквадратическое отклонение.

Функция размытия определяется следующим образом:

$f(\mathrm{c},\sigma) = \int Obs(y)\,\varphi(\mathrm{c}-y,\sigma)dy$, где $Obs(q) = \begin{cases}1, \mathrm{c} \notin \mathrm{C}_{free}\\ 0, \mathrm{c} \in \mathrm{C}_{free}\end{cases}$.

Для того, чтобы исключить недопустимые конфигурации, распределение задается как $g(\mathrm{c},\sigma) = \max(0, f(\mathrm{c},\sigma) - Obs(\mathrm{c}))$. Тем самым, функция $g(\mathrm{c},\sigma)$ принимает значение 0 для всех недопустимых конфигураций и тождественна функции $f(\mathrm{c})$ для допустимых. Параметр σ определяет близость генерируемых конфигураций к границам препятствий.

Алгоритм построен таким образом, что первая точка выбирается случайным образом с равномерным распределением, а каждая следующая – с распределением, соответствующим приведенной выше формуле.

7.6 Medial axis PRMs

Альтернативный принцип формирования выборок заключается в том, что предпочтение отдается конфигурациям равноудаленным от препятствий сцены [85–87]. Один из возможных способов его реализации заключается в предварительном построении срединной оси в пространстве сцены, которая используется для генерации новых точек.

7.7 Lazy-PRM

Характерным недостатком вероятностных маршрутных сетей является избыточность их представления, поскольку на практике для построения путей обычно требуется гораздо меньшее количество вершин. Это приводит к тому, что значительная часть времени выполнения алгоритма расходуется на вычислительно затратную операцию определения столкновений для конфигураций, несущественных для построения маршрута. Для устранения данного недостатка были предложены методы Fuzzy-PRM [88] и Lazy-PRM [89,90] с отложенной проверкой на бескофликтность переходов между конфигурациями.

Алгоритм поиска пути может быть представлен следующим образом (листинг 2). Вначале строится маршрутная сеть алгоритмом, аналогичным PRM, с тем исключением, что ребра не верифицируются локальным планировщиком (строка 1). Далее предпринимается попытка построить путь в сети (строка 5). В процессе построения ребра проверяются на бесконфликтность (строки 7-11), а конфликтные ребра удаляются из представления маршрутной сети (строка 10). В случае, если не удалось построить достоверный маршрут, соединяющий начальную и конечную конфигурации, разрешение маршрутной сети

увеличивается путем дополнительной генерации точек в слабо связных областях допустимых конфигураций (строка 13).

LazyPRM $(c_{init}, c_{goal}, K_{steps}, N_{samples})$
1. $G(V,E) \leftarrow PRM(N_{samples})$
2. $success \leftarrow false$
3. $step \leftarrow 0$
4. ***while*** $((step < K_{steps})$ & $(success = false))$
5. $p(V',E') \leftarrow FindShortestPath(G, c_{init}, c_{goal})$
6. ***if*** $(p \neq \emptyset)$
7. $success \leftarrow true$
8. ***for each*** $e \in p.E'$
9. ***if*** $(e \notin C_{free})$
10. $G.E = G.E \backslash e$
11. $success \leftarrow false$
12. ***else***
13. *VertexEnhancement(G,* $N_{samples}$ *)*
14. $step = step \leftarrow 1$

Листинг 2. Алгоритм поиска пути на основе вероятностной маршрутной сети с отложенной проверкой на бесконфликтность

Algorithm 2. Probabilistic roadmap algorithm with lazy collision checking

7.8 Dynamic PRM

Вероятностные маршрутные сети позволяют значительно повысить эффективность поиска пути в случае повторных запросов, поскольку вычислительные затраты на предварительный анализ сцены и формирование исходной маршрутной сети уже осуществлены. Однако данное утверждение справедливо исключительно для статических сцен. В случае динамики использование метода затруднительно, поскольку разворачивание новой маршрутной сети при каждом изменении в сцене представляется крайне неэффективным. Существует ряд модификаций метода, позволяющих перестраивать маршрутную сеть более рациональным образом [91–93].

8. Онлайн планирование. Деревья поиска

8.1 Нить Ариадны

Одним из известных сэмплинг-методов планирования движения на основе деревьев поиска является алгоритм “Нить Ариадны” [94,95]. Алгоритм строит маршрутную сеть таким образом, чтобы на каждом шаге покрыть как можно большую новую область конфигурационного пространства, не подлежащую анализу на предыдущих шагах. В зависимости от успешности распространения

алгоритм переключается в один из двух возможных режимов: режим поиска или режим анализа.

В режиме анализа осуществляется рандомизированный выбор опорных точек, или “ориентиров”, максимально удаленных от вершин разворачиваемой сети. В режиме поиска предпринимается попытка присоединить целевую конфигурацию к одному из ориентиров, найденных в режиме анализа.

Ключевым принципом алгоритма является перераспределение вычислительных ресурсов на анализ пространства сцены вместо исключительного продвижения к целевой конфигурации. Это позволяет повысить эффективность маршрутизации в сложных сценах за счет снижения роли “жадной” эвристики в поиске пути и его глобализации. Примечательно, что данный принцип получил свое развитие и в других алгоритмах данного семейства.

В качестве недостатка алгоритма следует указать высокую вычислительную сложность сопутствующей оптимизационной задачи выбора новой вершины в режиме анализа. Для преодоления этой проблемы предлагалось использовать генетические алгоритмы [95], но целесообразность их применения в данном случае вряд ли можно считать оправданной ввиду необходимости настройки множества параметров, специфичных для каждой прикладной задачи планирования движения [19].

8.2 Random-Walk Planner

Простой в реализации и в то же время достаточно эффективный алгоритм поиска пути Random-Walk Planner описан в работе [96]. Представленный алгоритм построен исключительно на рандомизированных перемещениях в конфигурационном пространстве, не используя никакой маршрутной сети.

Направление движения и длина шага определяются на каждой итерации случайным образом на основе нормального гауссова распределения. При этом матрица ковариации формируется на основе фиксированного числа последних испытаний. Таким образом, параметры, определяющие способ распространения, адаптивно настраиваются в ходе работы алгоритма. При этом добавление в путь каждой новой вершины выполняется за константное время в отличие от методов на основе маршрутных сетей.

Основную проблему для описанного алгоритма составляют сцены с узкими проходами и длинными коридорами. Для ее преодоления предложен гибридный вариант, объединяющий базовый алгоритм Random-walk Planner и рандомизированный метод потенциальных полей [97].

8.3 Expansive-Spaces Tree Planner

Планировщик на основе EST-деревьев (Expansive-Spaces Tree planner), описанный в работах [43,98,99], реализует некоторые принципы методов

вероятностных маршрутных сетей применительно к разрешению одиночных запросов поиска пути.

Стратегия сэмплирования в данном методе построена таким образом, чтобы предпочтение отдавалось областям конфигурационного пространства с наименьшей плотностью покрытия маршрутной сетью. Для этого вводится понятие веса конфигурации, который определяется количеством вершин маршрутной сети, лежащих в некоторой окрестности радиуса R от данной точки: $w(c,R) = |\{v \in V | Dist(c,v) < R\}|$. На каждой итерации алгоритма выбирается вершина дерева с вероятностью обратно пропорциональной ее весу. Далее в заданном радиусе генерируется K случайных бесконфликтных конфигураций и предпринимается попытка их включения в маршрутную сеть (листинг 3).

В работе [98] изложен способ повышения производительности алгоритма за счет двунаправленного поиска на основе EST деревьев с отложенной проверкой на бесконфликтность конфигурации (Single Query, Bidirectional Lazy Collision Checking).

К недостаткам рассмотренного алгоритма можно отнести то, что успех поиска сильно зависит от значений константных параметров R и K, которые следует выбирать с учетом особенностей решаемой прикладной задачи.

EST *(c_{init}, N_{steps},K, R)*
1. $T(V,E) \leftarrow (\{c_{init}\}, \emptyset)$
2. $step \leftarrow 0$
3. ***while*** *(step <* N_{steps}*)*
4. $v \leftarrow PickNodeWithProbability\left(T, \frac{1}{w(v,R)}\right)$
5. $U \leftarrow GenerateStates(K, \{c \in C_{free} | Dist(c,v) < R\})$
6. ***for each*** $u \in U$
7. ***if*** *(* $RetainWithProbability\left(\frac{1}{w(u,R)}\right)$*)*
8. ***if*** *(* $e(u,v) \in C_{free}$ *)*
9. $T.V \leftarrow T.V \cup u$
10. $T.E \leftarrow T.E \cup e(v,u)$
11. $step \leftarrow step + 1$

Листинг 3. Алгоритм построения EST дерева

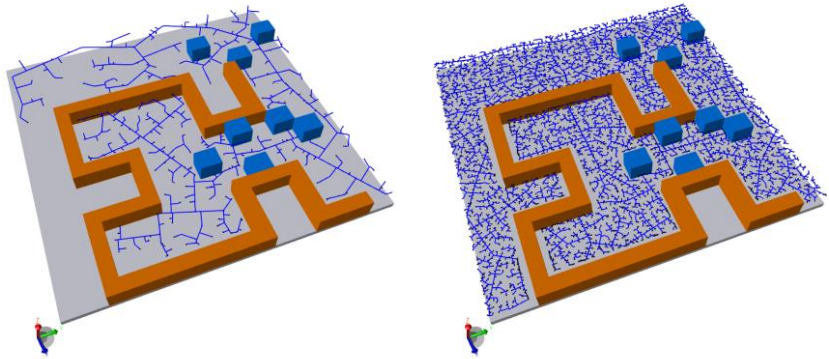
Algorithm 3. EST tree construction algorithm

8.4 Rapidly Exploring Random Trees

Алгоритм на основе быстро растущих случайных деревьев (Rapidly Exploring Random Tree) был изначально разработан для планирования движения неголономных механических систем в режиме реального времени [44,50]. По сравнению с другими известными сэмплинг-методами, эффективность которых

зависит от большого количества настраиваемых входных параметров, алгоритм RRT является наиболее универсальным средством решения широкого круга задач планирования движения.

Данный алгоритм использует в качестве дискретного представления конфигурационного пространства дерево, корень которого соответствует исходному положению объекта. Дерево достраивается таким образом, чтобы разрешение получаемой маршрутной сети увеличивалось на всем пространстве допустимых конфигураций (рис. 11). Таким образом, RRT деревья обладают свойствами, во многом схожими с кривыми, заполняющими пространство (Space-Filling Curves) [100].



(a) *(b)*

Рис. 11. Пример построения дерева поиска алгоритмом RRT

((a) 1000 итераций, (b)20000 итераций)

Fig. 11. An example of search tree construction by RRT algorithm

((a) 1000 iterations, (b)20000 iterations)

Рассмотрим алгоритм распространения RRT дерева (*листинг 4*). На каждой итерации выбирается случайная точка $\mathrm{c}_{\mathrm{rand}} \in \mathrm{C}$. Далее ищется ближайшая к ней вершина дерева $c_{near} \in T$. Отрезок прямой $p' = [\mathrm{c}_{near}, c_{rand}]$ проверяется на конфликтность. В процессе проверки определяется точка $\mathrm{c}_{new} = \begin{cases} c_{rand}, & p' \in C_{free} \\ c_{stop}, & p' \notin \mathrm{C}_{free} \end{cases}$, где c_{stop} – бесконфликтная конфигурация на данном отрезке такая, что $[\mathrm{c}_{near}, c_{stop}] \in p' \cap C_{free}$ и $\left\|\mathrm{c}_{stop}, C_{obs} \cap p'\right\| \leq \varepsilon$, а ε – погрешность определения конфликтов (рис. 12). Точка c_{new} и ребро $[\mathrm{c}_{near}, c_{new}]$ включаются в маршрутную сеть при условии $\mathrm{c}_{new} \neq \mathrm{c}_{rand}$.

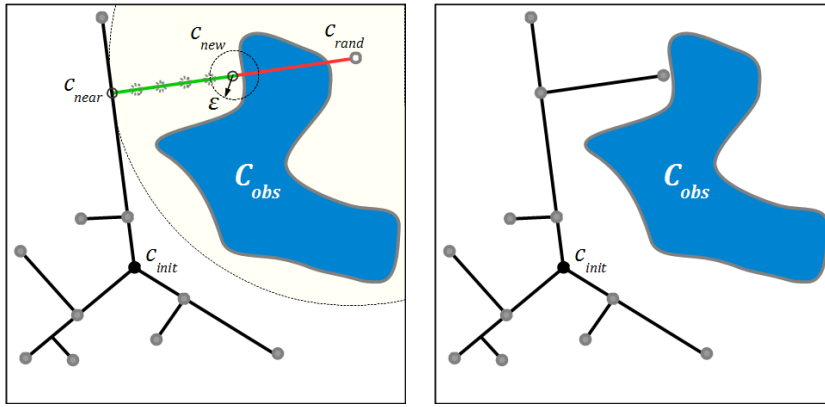


Рисунок 12. Включение новой конфигурации в дерево поиска алгоритмом RRT

Fig. 12. Construction of new tree vertex by RRT algorithm

Важно отметить, что алгоритм распространения построен таким образом, что вероятность выбора каждой вершины дерева пропорциональна объему региона вороного, которому она принадлежит. Данная особенность алгоритма обуславливает высокую скорость “разрастания” дерева в пространстве.

RRT $(c_{init}, c_{goal}, N_{steps}, N_{extend}, \rho)$
$T(V,E) \leftarrow (\{c_{init}\}, \emptyset)$
$step \leftarrow 0$
$success \leftarrow false$
while *(step <* N_{steps}*) & (success=false))*
  if $(step \bmod N_{extend} \neq 0)$
    $c_{rand} \leftarrow GenerateState()$
  else
    $c_{rand} \leftarrow c_{goal}$
  $c_{near} \leftarrow NearestNeighbour(c_{rand}, T)$
  $c_{new} \leftarrow FindStoppingState(c_{near}, c_{rand}, \rho)$
  if *(* $c_{new} \neq c_{near}$ *)*
    $T.V = T.V \cup c_{new}$
    $T.E = T.E \cup (c_{near}, c_{new})$
    $success \leftarrow \left(Distance(c_{new}, c_{goal}) \leq \rho\right)$
$step \leftarrow step + 1$
return *success*

Листинг 4. Алгоритм поиска пути с использованием RRT дерева

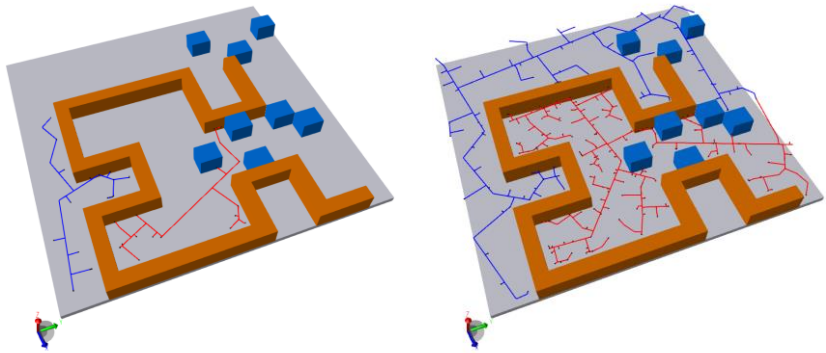
Algorithm 4. Path planning using RRT algorithm

Поиск пути осуществляется в результате периодически предпринимаемых попыток включить целевую конфигурацию в маршрутную сеть. Следует

отметить, что частое выполнение данной операции повлечет за собой неоправданное завышение вычислительных затрат, свойственное методам локального планирования и, в частности, методам потенциальных полей.

8.5 RRT-Connect

В качестве одного из способов повышения эффективности классического метода RRT-деревьев было предложено использовать двунаправленный поиск [50,101].



(a) *(b)*

Рис. 13. Пример построения деревьев поиска алгоритмом RRT-connect

((a) 250 итераций, (b)964 итерации)

Fig. 13. An example of search trees construction by RRT-connect algorithm

((a) 250 iterations, (b)964 iterations)

Для формирования маршрутной сети в алгоритме RRT-connect (листинг 5.) используется пара деревьев T_1 и T_2, корням которых соответствуют начальная конфигурация c_{init} и конечная конфигурация c_{goal} искомого пути. На каждом шаге одно из деревьев дополняется новыми бесконфликтными конфигурациями в соответствии с алгоритмом RRT. Однако вместо периодических попыток включения c_{goal} выполняется операция сращивания деревьев. При каждом включении новой вершины в T_1 предпринимается аналогичная попытка включить ее и в T_2.

При выполнении некоторого количества шагов деревья меняются ролями, что обеспечивает их более сбалансированный рост. Алгоритм построен таким образом, что предпочтение отдается тому дереву, распространение которого затруднено (например, в связи с наличием большого количества препятствий в его области). Количественным критерием в данном случае может служить общее число вершин дерева или суммарная длина ребер.

RRTConnect *(c_{init}, c_{goal}, N_{steps},ρ)*
1.$T_1(V,E) \leftarrow (\{c_{init}\},\emptyset)$
2.$T_2(V,E) \leftarrow (\{c_{goal}\},\emptyset)$
3.$step \leftarrow 0$
4.$success \leftarrow false$
*5.****while*** *((step <* N_{steps}*) & (success=false))*
6. $c_{rand} \leftarrow GenerateState()$
7. $c_{near} \leftarrow NearestNeighbour(c_{rand}, T_1)$
8. $c_{new} \leftarrow FindStoppingState(c_{near}, c_{rand}, \rho)$
9. ***if*** *(* $c_{new} \neq c_{near}$ *)*
10. $T_1.V = T_1.V \cup c_{new}$
11. $T_1.E = T_1.E \cup (c_{near}, c_{new})$
12. $c^*_{near} \leftarrow NearestNeighbour(c_{new}, T_2)$
13. $c^*_{new} \leftarrow FindStoppingState(c^*_{near}, c_{new}, \rho)$
14. ***if*** *(* $c^*_{new} \neq c^*_{near}$ *)*
15. $T_2.V = T_2.V \cup c^*_{new}$
16. $T_1.E = T_1.E \cup (c^*_{near}, c^*_{new})$
17. $success \leftarrow (Distance(c_{new}, c^*_{new}) \leq \rho)$
18. ***if*** *(* $|T_2| > |T_1|$ *)*
19. *Swap(* T_1, T_2 *)*
20. $step \leftarrow step + 1$
*21.****return*** *success*

Листинг 5. Алгоритм двунаправленного поиска пути с использованием RRT деревьев (RRT-connect)

Algorithm 5. Bidirectional version of RRT algorithm (RRT-connect)

Существует ряд работ, посвященных применению многих быстро растущих деревьев для локального планирования [102–104], однако вопрос о разумном компромиссе между количеством разворачиваемых деревьев и затратами на их распространение остается открытым [19].

8.6 Execution Extended RRT

В работе [105] предлагается способ повысить эффективность исполнения множественных запросов планирования движения за счет кэширования бесконфликных конфигураций. Таким образом, каждый последующий запрос планирования использует бесконфликтные переходы, обнаруженные в результате работы RRT алгоритма ранее.

Данная идея хорошо сочетается с двунаправленным поиском. Предложенный алгоритм Multi-Bridge ERRT [106] заимствует основной принцип RRT-connect, однако процедура распространения деревьев продолжается и после сращивания. Ввиду того, что структура, полученная в результате работы

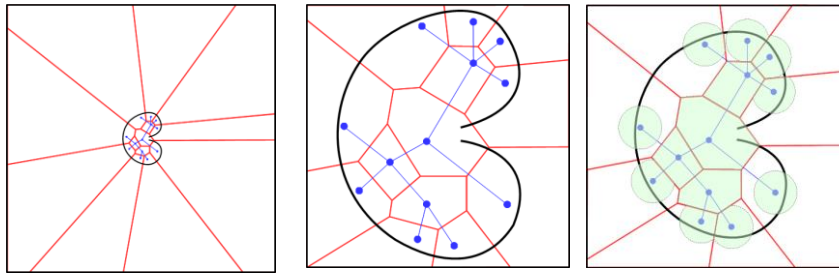
алгоритма уже не является деревом, результирующий путь ищется в графе с помощью алгоритма A^*.

8.7 Dynamic Domain RRT

Как было отмечено, на каждой итерации RRT алгоритма вероятность выбора вершины дерева прямо пропорциональна объему региона Вороного, которому она принадлежит. Данный факт обуславливает высокую скорость разрастания дерева в конфигурационном пространстве. Однако распространение может быть затруднено в тех случаях, когда внешние вершины дерева распределены преимущественно вдоль границ препятствий, а покрываемая деревом область конфигурационного пространства относительно невелика.

В качестве примера можно привести ситуацию, когда начальная точка пути находится внутри небольшой комнаты с единственным узким проходом (рис. 14 a, b). Если новые конфигурации генерируются с равномерным распределением во всем доступном объеме, то вероятность попадания случайно выбранной точки в область прохода может быть ничтожно мала.

Для решения описанной проблемы в работе [107] был предложен алгоритм Dynamic Domain RRT, динамически ограничивающий область сэмплирования в ходе формирования дерева. Для RRT дерева, построенного на множестве вершин V, вводится понятие динамической области $O = \bigcup_{v \in V} Domain(v, R)$. Каждый фрагмент области определяется как $Domain(v, R) = \begin{cases} D(v) \cap B_R(v), & min(\|v, C_{obs}\|) \leq \varepsilon \\ D(v), & min(\|v, C_{obs}\|) > \varepsilon \end{cases}$, где $D(v)$ – регион Вороного, которому принадлежит вершина v, а $B_R(v)$ – шар радиуса R с центром в v (рис. 14, c).



(a) *(b)* *(c)*

Рис. 14. Регионы Вороного, построенные на множестве вершин RRT дерева (a, b) и динамическая область сэмплирования (c).

Fig. 14. Voronoi regions associated with the nodes of tree constructed by RRT planner (a, b) and dynamic sampling domain (c).

Алгоритм построен таким образом, что распространение маршрутной сети осуществляется за счет конфигураций, принадлежащих динамической области O (листинг 6). В ходе построения дерева вершины помечаются как внешние или

внутренние. Включение случайно выбранной точки $c_{rand} \in C$ в дерево выполняется лишь при условии, что ближайшая к ней вершина является внутренней или лежит от нее на расстоянии, меньшем R. Значение радиуса R рекомендуется выбирать, исходя из максимально допустимой погрешности определения конфликтов, определяемой, например, линейной функцией $R = \lambda\varepsilon$, где λ – положительный целочисленный коэффициент.

DynamicDomainRRT *(c_{init},N_{steps},R)*
1. $T(V,E) \leftarrow (\{c_{init}\},\emptyset)$
2. $step \leftarrow 0$
3. ***while*** *(step <* N_{steps}*)*
4. repeat
5. $c_{rand} \leftarrow GenerateState(T.bounds)$
6. $c_{near} \leftarrow NearestNeighbour(c_{rand},T)$
7. ***until*** *(* $Distance(c_{near},c_{rand}) < c_{near}.radius$ *)*
8. $c_{new} \leftarrow FindStoppingState(c_{near},c_{rand})$
9. ***if*** *(* $c_{new} \neq c_{near}$ *)*
10. $c_{new}.radius \leftarrow \infty$
11. $T.V = T.V \cup c_{new}$
12. $T.E = T.E \cup (c_{near},c_{new})$
13. else
14. $c_{new}.radius \leftarrow R$
15. UpdateBounds(T.bounds, c_{new} *)*
16. $step = step \leftarrow 1$

Листинг 6. Построение дерева в алгоритме Dynamic-Domain-RRT

Algorithm 6. Construction of tree using Dynamic-Domain-RRT algorithm

8.8 Iterative Diffuse Path Planner

Применение метода RRT деревьев вместе с лежащей в его основе “жадной” эвристикой может быть крайне нежелательным для ряда задач планирования движения. Примером может служить задача автоматической разборки изделий, состоящих из множества деталей. Движение детали вблизи исходной конфигурации, как правило, сильно ограничено, и возможно лишь за счет малых поворотов. И наоборот, в области целевой конфигурации, находящейся на значительном удалении, препятствия вовсе отсутствуют.

Принцип отложенной проверки на бесконфликтность конфигураций в сочетании с динамически изменяемой погрешностью лежит в основе диффузионного алгоритма планирования движения (Iterative Diffuse Path Planner) (листинг 7). Алгоритм позволяет значительно повысить эффективность поиска в постановках, подобных приведенной [45].

В данном алгоритме используется лес растущих деревьев. Распространение осуществляется из произвольной вершины в направлении, определяемом случайным вектором, с шагом, равному текущему значению погрешности

определения конфликтов σ. После включения каждой новой вершины в дерево предпринимается попытка его слияния. Верификация ребер на конфликты в процессе распространения не выполняется.

Значение σ инициализируется достаточно большим значением, например, равным расстоянию между c_{init} и c_{goal}, и итерационно уменьшается вплоть до заданного минимально допустимого значения. На каждой итерации выполняется поиск пути описанным выше способом с последующей его верификацией. В случае, если путь оказался конфликтным, из представления деревьев исключаются все ребра, не прошедшие проверку, а процедуры распространения и поиска пути повторяются.

DiffuseRRT *(T(V,E), σ)*
1.$v_{rand} \leftarrow ChooseRandomNode(T.V)$
2.$c_{rand} \leftarrow ShootNearNode(v_{rand})$
3.$v_{near} \leftarrow NearestNeighbour(c_{rand}, T)$
4.$c_{new} \leftarrow FindStoppingState(v_{near}, c_{rand})$
5.***if*** $(c_{new} \neq v_{near})$
6. $T.V \leftarrow T.V \cup c_{new}$
7. $T.E \leftarrow T.E \cup (v_{near}, c_{new})$
8. ***for each*** $T^* \in G \backslash T$
9. *LinkTrees*(c_{new}, T^*)

IterativeDiffusionPathPlanner $(c_{init}, c_{goal}, \varepsilon)$
1.$G(V,E) \leftarrow \emptyset$
2.$\sigma \leftarrow Distance(c_{init}, c_{goal})$
3.***while*** $(\sigma > \varepsilon)$
4. $p(V', E') \leftarrow DiffuseRRT(c_{init}, c_{goal}, G, \sigma)$
5. $\sigma \leftarrow {}^{\sigma}/_{\alpha}$
6. ***for each*** $e \in p.E'$
7. ***if*** $(e \notin C_{free})$
8. $G.E \leftarrow G.E \backslash e$
9.$p \leftarrow DiffuseRRT(c_{init}, c_{goal}, G)$
10.***return*** *p*

Листинг 7. Итеративный дифузионный алгоритм поиска пути

Algorithm 7. Iterative diffuse path planning algorithm

9. Поиск оптимальных путей сэмплинг-методами

Удовлетворяя требованиям вероятностной успешности, рассмотренные выше сэмплинг-методы с успехом применяются к практическим задачам планирования движения, прежде неразрешимым за приемлемое время. Однако рандомизированный характер поиска, присущий PRM и RRT методам, крайне негативно влияет на качество получаемых решений. Подразумевается, что в

качестве количественных критериев качества решений могут выступать длина и гладкость траектории, расстояние между движущимся объектом и препятствиями сцены, стоимость преодоления препятствий, затраты энергии и т.п.

9.1 Офлайн оптимизация

Один из подходов к решению данной проблемы состоит в пост-обработке найденных бесконфликтных путей, основанной на их итерационной локальной оптимизации.

Довольно простым и популярным является итерационный алгоритм укорачивания пути [2]. На каждом шаге алгоритма выбирается пара точек, лежащих на пути. Первая точка выбирается случайно, а вторая – в некотором радиусе от нее. Причем точки не обязаны совпадать с вершинами пути. Далее точки соединяются отрезком прямой в конфигурационном пространстве, а отрезок проверяется на бесконфликтность или, другими словами, на принадлежность пространству допустимых конфигураций. В случае успеха все точки, лежащие между выбранными на исходном пути, выбрасываются, а отрезок включается в представление пути.

В результате пост-обработки, как правило, получается семейство гомотопных путей и, следовательно, оптимальное решение может быть недостижимо. Альтернативный способ оптимизации пути состоит в гибридизации или объединении участков из нескольких ранее найденных бесконфликтных путей [108].

9.2 Онлайн оптимизация

Несмотря на то, что RRT и PRM методы не учитывают характер переходов между конфигурациями, улучшить получаемые решения возможно путем модификации базовых алгоритмов. Оптимизация может осуществляться как за счет изменения способов распространения в пространстве [109,110], так и за счет изменения правил включения бесконфликтных конфигураций в маршрутную сеть [3].

9.3 Heuristic RRT

Несколько иной способ повысить качество путей предложен в работе [109]. В алгоритме hRRT (Heuristic RRT) вводится дополнительная эвристика, корректирующая функцию распределения вероятностей при выборе новых конфигураций таким образом, чтобы отдавать предпочтение путям наименьшей стоимости. Вероятность выбора конфигурации определяется двумя факторами. Во-первых, как и в базовом алгоритме RRT, она зависит от объема региона Вороного, которому она принадлежит. А во-вторых, она определяется предполагаемой стоимостью пути, проходящего через данную

конфигурацию. Для этого вводится следующая оценка: $m_{quality} = 1 - \frac{c_{vertex} - c_{opt}}{c_{max} - c_{opt}}$, где c_{vertex} – суммарная стоимость пути из начальной конфигурации в данную вершину и из данной вершины до целевой конфигурации, c_{opt} – предполагаемая стоимость оптимального пути из начальной конфигурации в целевую, а c_{max} – максимальная стоимость пути до любой вершины дерева на текущем шаге. Таким образом, числитель представляет собой оценку отклонения от предполагаемого оптимального пути, а знаменатель служит масштабирующим коэффициентом для нормирования результата. Данная оценка используется при сэмплировании конфигурационного пространства. На каждом шаге RRT алгоритма случайным образом выбирается несколько конфигураций, однако в дерево включается конфигурация с наименьшей стоимостью пути.

9.4 Transition-based RRT

Эвристика, применяемая в алгоритме hRRT, заставляет прорастать дерево к целевой конфигурации, часто игнорируя при этом более оптимальные пути. Метод T-RRT (Transition-base RRT) больше подходит для планирования движения в сложных стоимостных пространствах [110]. Для выбора конфигураций наименьшей стоимости в данном методе используется алгоритм имитации отжига (Simulated Annealing). Решение о включении новой конфигурации в дерево принимается с учетом ее стоимости относительно стоимости ближайшей вершины. Вероятность включения конфигурации c при заданной функции стоимости $f(c)$ определяется следующим образом: $P(\mathrm{c}) = \begin{cases} \exp\left(-\frac{\Delta f}{KT}\right), \Delta f > 0 \\ 1 \quad\quad , \Delta f \leq 0 \end{cases}$, где Δf – отношение приращения стоимости между конфигурациями к расстоянию между ними $\Delta f = \frac{f(c_{new}) - f(c_{near})}{dist(c_{near}, c_{new})}$, $K = \frac{f(c_{init}) + f(c_{goal})}{2}$ – нормирующий коэффициент, рассчитываемый как среднее значение стоимости начальной и целевой конфигураций, а T – числовой параметр, называемый температурой.

Таким образом, чем больше приращение стоимости, тем меньше вероятность включения конфигурации в дерево. При этом допускаются все переходы в конфигурации с меньшей стоимостью. Параметр T позволяет контролировать характер поиска. При очень высоких значениях температуры принимаются почти все новые точки. При низких значениях – распространение происходит преимущественно за счет исключительных точек, увеличивающих стоимость пути незначительно. Процесс поиска начинается при низкой температуре, а затем продолжается при возрастающих значениях температуры, которая повышается всякий раз когда количество точек, не удовлетворяющих критерию включения в дерево, достигает заданного предельного значения.

9.5 PRM*, RRT*

В работе [3] предложены модификации алгоритмов PRM и RRT, предназначенные для получения асимптотически оптимальных решений.

Определение. Алгоритм ALG асимптотически оптимален, если он вероятностно успешен, а также для любой задачи поиска пути $\langle C_{free}, c_{init,} c_{goal}\rangle$ со стоимостной функцией $f_c(p)$ и оптимальным решением p^* имеет место $P(\{lim_{n\to\infty} \sup Y_n^{ALG} = f_c(p^*)\}) = 1$, где Y_n^{ALG} – минимальное значение стоимости для всех решений, полученных за n шагов алгоритма.

RRT* повторяет базовый алгоритм RRT за исключением процедуры включения новых конфигураций в дерево поиска (листинг 8). Данная процедура построена таким образом, чтобы стоимость путей, входящих в дерево, уменьшалась с каждым шагом алгоритма.

В ходе построения дерево поиска дополняется значениями стоимости пути, ведущего в каждую из его вершин. Как и в базовом алгоритме, на каждом шаге для случайным образом выбранной точки c_{rand} ищется ближайшая вершина графа c_{near}, а также точка $c_{new} \in C_{free}$, полученная в результате распространения из c_{near} в c_{rand} (строки 4-7). Далее определяется подмножество вершин дерева C^*, лежащих внутри шара радиуса r с центром в c_{new}. Среди них находят вершину c_{min} такую, что стоимость пути из начальной конфигурации c_{init} в c_{new} была минимальной (строки 8-10).

После того как c_{new} включается в дерево в качестве дочернего узла c_{min} (строки 11-12), проводится локальная оптимизация путей внутри шара. Для каждой конфигурации в нем $c' \in C^*$ предпринимается попытка построения пути, ведущего в c_{new}, и в случае, если он имеет меньшую стоимость, то конфигурация c' становится дочерней вершиной c_{new} (строка 13).

Радиус поиска в данном алгоритме представляет собой функцию от количества вершин дерева и адаптивно уменьшается с его детализацией. В работе [3] доказывается, что алгоритм RRT* асимптотически оптимален при $r(|V|) = \gamma_{RRG}\left(\frac{\log(|V|)}{|V|}\right)^{\frac{1}{d}}$, где $\gamma_{RRG} > 2\left(1+\frac{1}{d}\right)^{\frac{1}{d}}\left(\frac{\mu(C_{free})}{\mu(B_1(\cdot))}\right)$, $|V|$ – количество вершин дерева на данном шаге алгоритма, d – размерность конфигурационного пространства, $\mu(C_{free})$ – объем пространства допустимых конфигураций, $\mu(B_1(\cdot))$ – объем d-мерного единичного шара.

RRTStar *(c_{init},N_{steps})*
1. $T(V,E) \leftarrow (\{c_{init}\}, \emptyset)$
2. $step \leftarrow 0$
3. ***while*** *(step <* N_{steps}*)*
4. $c_{rand} \leftarrow GenerateState()$
5. $c_{near} \leftarrow NearestNeighbour(c_{rand}, T)$
6. $c_{new} \leftarrow FindStoppingState(c_{near}, c_{rand})$
7. ***if*** *(* $c_{new} \neq c_{near}$ *)*
8. $r \leftarrow SearchRadius(step)$

9. $C^* \leftarrow NearestNeighbours(c_{new}, T, r)$
10. $c_{min} \leftarrow MinCostParent(C^*)$
11. $T.V = T.V \cup c_{new}$
12. $T.E = T.E \cup (c_{min}, c_{new})$
13. *Rewire(T, C*, c_{min}, c_{new})*
14. $step = step \leftarrow 1$

*Листинг 8. Построения дерева поиска алгоритмом RRT**

Algorithm 8. Construction of tree using RRT algorithm*

В последние годы методы PRM* и RRT* получили дальнейшее развитие. Разработано большое количество алгоритмов, демонстрирующих более высокие показатели эффективности поиска и обладающих лучшей сходимостью к оптимальным решениям.

В работах [111,112] предложены алгоритмы Lazy-PRM* и Lazy-RRG*, реализующие принцип отложенной проверки на конфликты. Оба алгоритма асимптотически оптимальны и в ряде случаев позволяют ускорить поиск пути.

Для устранения проблемы избыточного количества ребер в маршрутных сетях, полученных в результате работы алгоритма PRM*, были предложены модификации для поиска путей, близких к оптимальным. В основе алгоритма IRS [113–115] лежит инкрементальный способ построении α-спаннера графа. При количестве итераций, стремящемся к бесконечности, и заданном коэффициенте α алгоритм сходится к оптимальному решению с точностью $1 + \alpha$ и вероятностью, равной единице. Схожий принцип используется в алгоритме SPARS, который заимствует идеи инкрементального построения спаннера и отсечения точек на основе областей видимости [116,117].

Для повышения скорости сходимости к оптимальному решению было предложено дополнить алгоритм RRT* процедурой перепланирования, не ограничивающейся локальной оптимизацией ребер. В алгоритмах RRT*-smart [118] и RRT# [119] участки, потенциально являющиеся частью оптимального пути наименьшей стоимости, оптимизируются по всей длине, начиная от корня дерева.

В работе [120] описывается асимптотически оптимальный алгоритм FMT* (Fast Marching Tree), который строит маршрутную сеть на множестве случайных бесконфликтных конфигураций и одновременно поддерживает ее остовное дерево с корнем в начальной точке пути и минимальными значениями стоимости путей до его вершин.

Алгоритм LBT-RRT (Lower Bound Tree RRT) [121], также являющийся развитием RRT*, позволяет повысить эффективность поиска за счет смягчения требования асимптотической оптимальности. Для планирования движения в сложных стоимостных пространствах был разработан гибридный алгоритм TRRT*, использующий алгоритм имитации отжига [122]. В работах [123,124]

были предложены модификации RRT* и FMT*, использующие двунаправленный поиск.

10 Заключение

Таким образом, в работе представлен обзор задач и методов современной теории планирования движения. Рассмотрены основные факторы, определяющие особенности прикладных задач и влияющие на выбор применяемых математических методов. К ним отнесены характер постановки планирования движения (локальный или глобальный), геометрическое представление перемещаемого объекта (твердое тело или кинематическая конструкция), свойства окружения (статическое или динамическое), свойства конфигурационного пространства (равномерное распределение допустимых состояний или наличие узких областей), характер запросов планирования (однократный или многократный). Также представлены основные подходы к планированию движения, связанные с пространственной декомпозицией сцены, физическими аналогиями с потенциальными полями, маршрутными сетями и быстро растущими деревьями. Детально рассмотрены ключевые методы и алгоритмы, реализующие данные подходы, и аспекты их практического применения.

Предполагается, что обзор поможет в выборе методов с учетом особенностей решаемых прикладных задач и в их эффективной программной реализации и настройке. Ожидается также, что обзор послужит основой для систематизации и объектной концептуализации теории планирования движения, необходимой для создания единой программно-инструментальной среды разработки приложений.

Список литературы

[1]. Choset H., Lynch K., Seth H., Kantor G., Burgard W., Kavraki L.E., Thrun S. Principles of Robot Motion-Theory, Algorithms , and Implementation. MIT Press, 2005.
[2]. Geraerts R., Overmars M.H. Creating High-quality Paths for Motion Planning. Int. J. Rob. Res., vol. 26, № 8, 2007, pp. 845–863.
[3]. Karaman S., Frazzoli E. Sampling-based algorithms for optimal motion planning. Int. J. Robot., vol. 30, № 7, 2011, pp. 846–894
[4]. Laumond J.-P. Kineo CAM: A success story of motion planning algorithms. IEEE Robot. Autom. Mag., vol. 13, № 2, 2006, pp. 90–93.
[5]. Rockel S., Klimentjew D., Zhang L., Zhang J. An hyperreality imagination based reasoning and evaluation system (HIRES). Proc. IEEE Int. Conf. on Robotics and Automation, 2014. pp. 5705–5711.
[6]. Sucan I., Moll M., Kavraki L.E. The Open Motion Planning Library. IEEE Robot. Autom. Mag. vol. 19, № 4, 2012, pp. 72–82.
[7]. Diankov R. Automated Construction of Robotic Manipulation Programs. PhD thesis, Carnegie Mellon University, Robotics Institute, August 2010, 263 p.

[8]. Porta J.M., Ros L., Bohigas O., Manubens M., Rosales C., Jaillet L. The CUIK Suite: Motion Analysis of Closed-chin Multibody Systems. IEEE Robot. Autom. Mag., vol. 21, № 3, 2014, pp. 105–114.
[9]. Semenov V.A., Kazakov K.A., Zolotov V.A. Effective spatial reasoning in complex 4D modeling environments. eWork Ebus. Archit. Eng. Constr. eds. A.Mahdavi, B. Martens, R. Scherer, CRC Press. Taylor Fr. Group, London, UK. 2015, pp. 181–186.
[10]. Kazakov K.A., Semenov V.A., Zolotov V.A. Topological Mapping Complex 3D Environments Using Occupancy Octrees. 21st Int. Conf. Comput. Graph. Vision, Sept. 26-30, 2011, Moscow, Russ. 2011, pp. 111–114.
[11]. Semenov V.A., Kazakov K.A., Morozov S. V., Tarlapan O.A., Zolotov V.A., Dengenis T. 4D modeling of large industrial projects using spatio-temporal decomposition. eWork Ebus. Archit. Eng. Constr. eds. K. Menzel R. Scherer, CRC Press. Taylor Fr. Group,London, UK. 2010, pp. 89–95.
[12]. Brooks R.A., Lozano-Peres T. A Subdivision Algorithm in Configuration Space For Find Path with Rotation. IEEE Trans. Syst. Man. Cybern., vol. SMC-15, № 2, 1985, pp. 224–233.
[13]. D. Zhu and J.-C. Latombe. Constraint reformulation in a hierarchical path planner. In Proc. IEEE Int. Conf. on Robotics and Automation, 1990, pp. 1918-1923.
[14]. Hornung A., Wurm K.M., Bennewitz M., Stachniss C., Burgard W. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. Auton. Robots., vol. 34, № 3, 2013, pp. 189–206.
[15]. Semenov V.A., Kazakov K.A., Zolotov V.A. Global path planning in 4D environments using topological mapping. eWork Ebus. Archit. Eng. Constr. 2012, pp. 263–269.
[16]. Chen D.Z., Szczerba R.J., Uhran Jr J.J. Using Framed-Quadtrees to Find Conditional Shortest Paths in an Unknown 2-D Environment. Technical Report #95-2, Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, Indiana, Jan. 1995, 33 p.
[17]. Chazelle B. Approximation and Decomposition of Shapes. Advances in Robotics 1: Algorithmic and Geometric Aspects of Robotics, 1987, pp. 145–185.
[18]. De Berg M., Cheong O., Van Kreveld M., Overmars M. Computational Geometry: Algorithms and Applications. 3rd Edition. Springer-Verlag, 2008, 386 p.
[19]. LaValle S.M. Planning Algorithms. Cambridge University Press, 2006, 844 p.
[20]. Canny J.F., Lin M.C. An opportunistic global path planner. Algorithmica, vol. 10, № 2-4, 1993, pp. 102–120.
[21]. Huang H.-P.H.H.-P., Chung S.-Y.C.S.-Y. Dynamic visibility graph for path planning. Proc. Int. Conf. Intell. Robot. Syst., vol. 3, 2004, pp. 2813–2818.
[22]. Kaluder H., Brezak M., Petrovic I. A visibility graph based method for path planning in dynamic environments. MIPRO, 2011, Proceedings of the 34th International Convention, Opatija, Croatia, 2011, pp. 717–721.
[23]. Aurenhammer F. Voronoi Diagrams – A Survey of a Fundamental Data Structure. ACM Comput. Surv., vol. 23, № 3, 1991, pp. 345–405.
[24]. Choset H. Sensor based motion planning: The hierarchical generalized voronoi graph. Ph.D. Thesis, California Institute of Technology, 1996, 201 p.
[25]. Russell S.J., Norvig P. Artificial Intelligence: A Modern Approach. Neurocomputing, vol. 9, № 2, 1995, pp. 215-218.
[26]. Bell S., An Overview of Optimal Graph Search Algorithms for Robot Path Planning in Dynamic or Uncertain Environments. Oklahoma Christian University, 2010, 9 p.

[27]. De Filippis L., Guglieri G. Advanced graph search algorithms for path planning of flight vehicles. Recent Adv. Aircr. Technol., 2012, pp. 159–192.
[28]. Zeng W., Church R.L. Finding shortest paths on real road networks: the case for A*. Int. J. Geogr. Inf. Sci., vol. 23, № 4, 2009, pp. 531–543.
[29]. Korf R.E. Depth-first iterative-deepening. An optimal admissible tree search. Artif. Intell., vol. 27, № 1, 1985, pp. 97–109.
[30]. Zhou R., Hansen E.A. Memory-Bounded {A*} Graph Search. The Florida AI Research Society Conference - FLAIRS, 2002, pp. 203–209.
[31]. Holte R., Perez M., Zimmer R., MacDonald A. Hierarchical A*: Searching abstraction hierarchies efficiently. Proceeding AAAI'96 Proceedings of the thirteenth national conference on Artificial intelligence – Volume 1, 1996, pp. 530–535.
[32]. Botea A., Muller M., Schaeffer J. Near optimal hierarchical path-finding. Journal of Game Development, vol. 1, issue 1, 2004, pp. 7-28.
[33]. Kring A., Champandard A., Samarin N. DHPA* and SHPA*: Efficient Hierarchical Pathfinding in Dynamic and Static Game Worlds. Proc. Sixth Artificial Intelligence and Interactive Digital Entertainment Conference, 2010, pp. 39–44.
[34]. Harabor D., Grastien A. Online Graph Pruning for Pathfinding On Grid Maps. AAAI Conf. Artif. Intell., 2011, pp. 1114–1119.
[35]. Koenig S., Likhachev M., Furcy D. Lifelong Planning A*. Artif. Intell., vol. 155, № 1-2, 2004, pp. 93–146.
[36]. Stentz A. The Focussed D* Algorithm for Real-Time Replanning. Proceeding IJCAI'95 Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2, 1995, pp. 1652-1659.
[37]. Koenig S., Likhachev M. D* Lite. Proceedings of the AAAI Conference of Artificial Intelligence (AAAI), 2002, pp. 476–483.
[38]. Koenig S., Likhachev M. Fast Replanning for Navigation in Unknown Terrain. IEEE Trans. Robot., vol. 21, issue 3, 2005, pp. 354-363..
[39]. Khatib O. Real time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics and Research, vol. 5, № 1, 1986, pp. 90–98.
[40]. Barraquand J., Latombe J.C. A Monte-Carlo algorithm for path planning with many degrees of freedom. Proceedings 1990 IEEE International Conference on Robotics and Automation, vol.3, 1990, pp. 1712–1717.
[41]. Barraquand J., Latombe J.C. Robot motion planning: A distributed representation approach. International Journal of Robotics Research, vol. 10, issue 6, 1991. pp. 628 - 649
[42]. Kavraki L.E., Svestka P., Latombe J.C., Overmars M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom., vol. 12, № 4, 1996, pp. 566–580.
[43]. Hsu D., Latombe J.C., Motwani R. Path Planning in Expansive Configuration Spaces. Proceedings 1997 IEEE International Conference on Robotics and Automation, vol. 3, 1997, pp. 2719–2726.
[44]. LaValle S.M., Kuffner J.J. Rapidly-exploring random trees: Progress and prospects. 2000 Workshop on the Algorithmic Foundations of Robotics, 2000, pp. 293–308.
[45]. Ferre E., Laumond J.-P. An iterative diffusion algorithm for part disassembly. Proceedings 2004 IEEE International Conference on Robotics and Automation, vol. 3, 2004, pp. 3149–3154.
[46]. Bayazit O.B., Lien J.-M., Amato N.M. Probabilistic Roadmap Motion Planning for Deformable Objects. Proceedings 2002 IEEE International Conference on Robotics and Automation, vol. 2, 2002, pp. 2126–2133.

[47]. Guibas L.J., Holleman C., Kavraki L.E. A Probabilistic Roadmap Planner for Flexible Objects with aWorkspace Medial-Axis-Based Sampling Approach. IEEE/RSJ Int. Conf. Intelligent Robot. Syst., 1999, pp. 254–260.
[48]. Berenson D., Srinivasa S., Kuffner J.J. Task Space Regions: A framework for pose-constrained manipulation planning. Int. J. Rob. Res., vol. 30, № 12, 2011, pp. 1435–1460.
[49]. Yakey J.H., LaValle S.M., Kavraki L.E. Randomized path planning for linkages with closed kinematic chains. IEEE Trans. Robot. Autom., vol. 17, № 6, 2001, pp. 951–958.
[50]. LaValle S.M., Kuffner J.J. Randomized Kinodynamic Planning. Int. J. Rob. Res., vol. 20, № 5, 2001, pp. 378–400.
[51]. Niederreiter H. Random number generation and Quasi-Monte Carlo methods. Society for Industrial and Applied Mathematics, 1992, 241 p.
[52]. Дмитриев К.А. От Монте-Карло к Квази Монте-Карло. Труды 12-ой международной конференции по компьютерной графике и машинному зрению ГрафиКон'2002, 2002, стр. 53-58
[53]. Соболь И.М. Многомерные квадратурные формулы и функции Хаара. М.: Главная редакция физико-математической литературы изд-ва «Наука», 1969, 288 стр.
[54]. Соболь И. М. Численные методы Монте-Карло. М.: Главная редакция физико-математической литературы изд-ва «Наука», 1973, 312 стр.
[55]. Khaksar W., Hong T.S., Khaksar M., Motlagh O. A low dispersion probabilistic roadmaps (LD-PRM) algorithm for fast and efficient sampling-based motion planning. Int. J. Adv. Robot. Syst.vol. 10, № 397, 2013, pp. 1-10.
[56]. LaValle S.M., Branicky M.S. On the Relationship Between Classical Grid Search and Probabilistic Roadmaps. In Algorithmic Foundations of Robotics V. Springer Tracts in Advanced Robotics, vol. 7, 2004, pp. 59-75
[57]. Zhang L., Kim Y.J., Manocha D. C-DIST: efficient distance computation for rigid and articulated models in configuration space. Proc. 2007 ACM Symp. Solid Phys. Model. - SPM ’07, 2007, pp. 159-169.
[58]. Lin M.C. Efficient Collision Detection for Animation and Robotics. Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, USA, 1993, 159 p.
[59]. Christer Ericson. Real-time Collision Detection. The Morgan Kaufmann Series in Interactive 3-D Technology. CRC Press, 2004, 632 p.
[60]. Andersen K. A., Bay C. A survey of algorithms for construction of optimal Heterogeneous Bounding Volume Hierarchies. Technical Report. Copenhague, Denmark: Department of Computer Science, University of Copenhagen, 2006, 32 p.
[61]. Золотов В.А., Семенов В.А. Современные методы поиска и индексации многомерных данных в приложениях моделирования больших динамических сцен. Труды ИСП РАН, том 24, 2013, стр. 381-416. DOI: 10.15514/ISPRAS-2013-24-17.
[62]. Золотов В.А., Семенов В.А. Перспективные схемы пространственно-временной индексации для визуального моделирования масштабных индустриальных проектов. Труды ИСП РАН, том 26, вып. 2, 2014, стр. 197-230. DOI: 10.15514/ISPRAS-2014-26(2)-9.
[63]. Золотов В.А., Петрищев К.С., Семенов В.А. Исследование методов пространственного индексирования динамических сцен на основе регулярных октодеревьев. Труды 25-й международной конференции GraphiCon2015, 2015, pp. 115-122.

[64]. Pungotra H. Collision Detection and Merging of Deformable B-spline Surfaces in Virtual Reality Environment. Ph. D. thesis, The Univeristy of Western Ontario, Canada, 2010, 180 p.
[65]. Sandqvist J., Collision detection using boundary representation, BREP. Master thesis, Umeå University, Sweden, 2015, 54 p.
[66]. Su C.J., Lin F., Ye L. New collision detection method for CSG-represented objects in virtual manufacturing. Computers in Industry, vol. 40, № 1, 1999, pp. 1–13.
[67]. Klein J., Zachmann G. Point cloud collision detection. Proc. Eurographics 2004, 2004, pp. 567–576.
[68]. Schwarzer F., Saha M., Latombe J.C. Exact Collision Checking of Robot Paths. In Algorithmic Foundations of Robotics V. Springer Tracts in Advanced Robotics, vol. 7, 2004, pp. 25–41.
[69]. Yershova A., LaValle S.M. Improving Motion Planning Alorithms by Efficient Nearest-Neighbor Searching. IEEE Transactions on Robotics, vol. 23, issue 1, 2007, pp. 151–157.
[70]. Yershova A., LaValle S.M. Planning for closed chains without inverse kinematics. Department of Computer Science, University of Illinois, Urbana, USA, 2007, 7 p. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.115.7831&rep=rep1&type=pdf.
[71]. Brown R.A. Building a Balanced k-d Tree in O (kn log n) Time. J. Comput. Graph. Tech, vol. 4, № 1, 2015, pp. 50–68.
[72]. Ciaccia P., Patella M., Rabitti F., Zezula P. Indexing Metric Spaces with M-Tree. Proc. Atti del Quinto Convegno Nazionale SEBD, 1997, pp. 67–86.
[73]. Gipson B., Moll M., Kavraki L.E. Resolution Independent Density Estimation for motion planning in high-dimensional spaces. Proc. 2003 IEEE International Conference on Robotics and Automation, 2013, pp. 2437–2443.
[74]. Fredriksson K. Geometric Near-neighbor Access Tree (GNAT) revisited. arXiv:1605.05944v2, 2016, 7 p.
[75]. Yu C., Ooi B.C., Tan K.-L. Indexing the Distance: An Efficient Method to KNN Processing. Proceedings of the 27th International Conference on Very Large Data Bases, 2001, pp. 421-430.
[76]. Beygelzimer A., Kakade S., Langford J. Cover trees for nearest neighbor. ICML '06, Proc. 23rd Int. Conf. Mach. Learn., 2006, pp. 97–104.
[77]. Ichnowski J., Alterovitz R. Fast Nearest Neighbor Search in SE (3) for Sampling-Based Motion Planning. Algorithmic Foundations of Robotics XI, Volume 107 of the series Springer Tracts in Advanced Robotics, 2015, pp 197-214.
[78]. Amato N.M., Bayazit O.B., Dale L.K., Jones C., Vallejo D. Choosing good distance metrics and local planners for probabilistic roadmap methods. IEEE Trans. Robot. Autom., vol. 16, № 4, 2000, pp. 442–447.
[79]. Nissoux C., Simeon T., Laumond J.-P. Visibility based probabilistic roadmaps. 1999 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol. 3, 1999, pp. 1316–1321.
[80]. Amato N.M., Wu Y. A randomized roadmap method for path and manipulation planning. Proc. 1996 IEEE International Conference on Robotics and Automation, vol. 1, 1996, pp. 113–120.
[81]. Amato N.M., Bayazit O.B., Dale L.K., Jones C., Vallejo D. OBPRM: An Obstacle-Based PRM for 3D Workspaces. Proceeding WAFR '98 Proceedings of the third workshop on the algorithmic foundations of robotics on Robotics : the algorithmic perspective, 1998, pp. 155–168.

[82]. Yeh H.Y., Thomas S., Eppstein D., Amato N.M. UOBPRM: A uniformly distributed obstacle-based PRM. IEEE Int. Conf. Intell. Robot. Syst., 2012, pp. 2655–2662.
[83]. Hsu D., Jiang T., Reif J., Sun Z. The bridge test for sampling narrow passages with probabilistic roadmap planners. Proc. 2003 IEEE International Conference on Robotics and Automation, vol. 3, 2003, pp. 4420–4426.
[84]. Boor V., Overmars M.H., Stappen a. F. Van Der. The Gaussian sampling strategy for probabilistic roadmap planners. Proc. 1999 IEEE International Conference on Robotics and Automation, vol 2, 1999, pp. 1018–1023.
[85]. Lien J.-M., Thomas S., Amato N.M. A general framework for sampling on the medial axis of the free space. Proc. 2003 IEEE International Conference on Robotics and Automation, vol. 3, 2003, pp. 4439–4444.
[86]. Wilmarth S. a., Amato N.M., Stiller P.F. MAPRM: a probabilistic roadmap planner with sampling on the medial\naxis of the free space. Proc. 1999 IEEE International Conference on Robotics and Automation, vol 2, 1999, pp. 1024–1031.
[87]. Holleman C., Kavraki L.E. A framework for using the workspace medial axis in PRM planners. Proc. 2000 IEEE International Conference on Robotics and Automation, vol 2, 2000, pp. 1408–1413.
[88]. Nielsen C.L.L., Kavraki L.E. A two level fuzzy PRM for manipulation planning. 2000 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol. 3, 2000, pp. 1716–1721.
[89]. Bohlin R., Kavraki L.E. Path planning using lazy PRM. Proc. 2000 ICRA Millenn. Proc. 2000 IEEE International Conference on Robotics and Automation, vol 1, 2000, pp. 521–528.
[90]. Bohlin R., Kavraki L.E. A Randomized Approach to Robot Path Planning Based on Lazy Evaluation. In Handbook of Randomized Computing, volume I/II, Springer, 2001, pp. 221–253.
[91]. Leven P., Seth H. Toward Real-Time Path Planning in Changing Environments. Proceedings of the Fourth International Workshop on the Algorithmic Foundations of Robotics, 2000, pp. 363-376.
[92]. Nieuwenhuisen D., Van Den Berg J., Overmars M.H. Efficient path planning in changing environments. 2007 IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2007, pp. 3295–3301.
[93]. Jaillet L., Simeon T. A PRM-based motion planner for dynamically changing environments. 2004 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol. 2, 2004, pp. 1606–1611.
[94]. Bessiere P., Ahuactzin J.M., Talbi E.-G., Mazer E. The Ariadne's Clew Algorithm: Global Planning With Local Methods. Proceeding of the Workshop on Algorithmic Foundations of Robotics, 1995, pp. 39-49.
[95]. Mazer E., Ahuactzin J.M., Bessiere P. The Ariadne ' s Clew Algorithm. Journal of Artifical Intelligence Research, vol. 9, 1998, pp. 295–316.
[96]. Carpin S., Pillonetto G. Motion planning using adaptive random walks. IEEE Trans. Robot., vol. 21, № 1, 2005, pp. 543–548.
[97]. Carpin S., Pillonetto G. Merging the adaptive random walks planner with the randomized potential field planner. Proc. Fifth Int. Work. Robot Motion Control, 2005, pp. 151–156.
[98]. Sanchez G., Latombe J.C. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In Robotics Research, Volume 6 of the series Springer Tracts in Advanced Robotics, Springer, 2003, pp 403-417.
[99]. Hsu D. Randomized Single-query Motion Planning in Expansive Spaces. Ph. D. thesis, Stanford University, USA, 2000, 118 p.
[100]. Sagan H. Space-Filling Curves. Springer-Verlag, New York, 1994, 193 p.

[101]. Kuffner J.J., LaValle S.M. RRT-connect: An efficient approach to single-query path planning. Proc. 2000 IEEE International Conference on Robotics and Automation, vol 2, 2000, pp. 995–1001.
[102]. Bekris K.E., Chen B.Y., Ladd A.M., Plaku E., Kavraki L.E. Multiple Query Motion Planning using Single Query Primitives. IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2003, pp. 656–661.
[103]. Plaku E., Kavraki L.E. Distributed Sampling-Based Roadmap of Trees for Large-Scale Motion Planning. Proc. 2005 IEEE International Conference on Robotics and Automation, 2005, pp. 3879–3884.
[104]. Strandberg M. Robot Path Planning : An Object-Oriented Approach. Ph.D.thesis, Automatic and Control Departmentof, Signals, Sensors and Systems Royal Institute of Technology (KTH), Stockholm, Sweden, 2004, 244 p.
[105]. Bruce J.R., Veloso M. Real-time multi-robot motion planning with safe dynamics. Multi-Robot Systems. From Swarms to Intelligent Automata, Volume III. Proceedings from the 2005 International Workshop on Multi-Robot Systems, Springer, 2005, pp. 1–12.
[106]. Bruce J.R. Real-time motion planning and safe navigation in dynamic multi-robot environments. PhD. Thesis, Carnegie Mellon University, Pittsburgh, USA, 2006, 204 p.
[107]. Yershova A., Jaillet L., Siméon T., LaValle S.M. Dynamic-Domain RRTs: Efficient Exploration by Controlling the Sampling Domain. Proc. 2005 IEEE International Conference on Robotics and Automation, 2005, pp. 3856–3861.
[108]. Raveh B., Enosh A., Halperin D. A little more, a lot better: Improving path quality by a path-merging algorithm. IEEE Trans. Robot., vol. 27, № 2, 2011, pp. 365–371.
[109]. Urmson C., Simmons R. Approaches for heuristically biasing RRT growth. Proc. 2003 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol.2, 2003, pp. 1178–1183.
[110]. Jaillet L., Cortes J., Simeon T. Sampling-Based Path Planning on Costmaps Configuration-space. Ieee Trans. Robot., vol. 26, № 4, 2010, pp. 635–646.
[111]. Hauser K. Lazy collision checking in asymptotically-optimal motion planning. Proc. 2015 IEEE International Conference on Robotics and Automation, 2015, pp. 2951–2957.
[112]. Luo J., Hauser K. An Empirical Study of Optimal Motion Planning. IEEE/RSJ Conf. Intell. Robot. Syst., 2014, pp. 1761–1768.
[113]. Marble J.D., Bekris K.E. Asymptotically Near-Optimal is Good Enough for Motion Planning. 15th Int. Symp. Robot. Res., 2011, pp. 419-436.
[114]. Marble J.D., Bekris K.E. Computing spanners of asymptotically optimal probabilistic roadmaps. IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2011, pp. 4292–4298.
[115]. Marble J.D., Bekris K.E. Towards small asymptotically near-optimal roadmaps. Proc. 2012 IEEE International Conference on Robotics and Automation, 2012, pp. 2557–2562.
[116]. Dobson A., Bekris K.E. Improving Sparse Roadmap Spanners. Proc. 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 4106–4111.
[117]. Dobson A., Bekris K.E. Sparse roadmap spanners for asymptotically near-optimal motion planning. Int. J. Rob. Res., vol. 33, № 1, 2014, pp. 18–47.
[118]. Nasir J., Islam F., Malik U., Ayaz Y., Hasan O., Khan M., Muhammad M.S. RRT*-SMART: A rapid convergence implementation of RRT*. Int. J. Adv. Robot. Syst., vol. 10, 2013, pp. 1-12.
[119]. Arslan O., Tsiotras P. Use of relaxation methods in sampling-based algorithms for optimal motion planning. Proc. 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 2421–2428.

[120]. Janson L., Pavone M. Fast Marching Trees : a Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions – Extended Version, arXiv:1306.3532v4, 2013, pp. 1–60.
[121]. Salzman O., Halperin D. Asymptotically near-optimal RRT for fast, high-quality, motion planning. Proc. 2014 IEEE International Conference on Robotics and Automation, 2014, pp. 4680–4685.
[122]. Devaurs D., Simeon T., Cortes J. Optimal Path Planning in Complex Cost Spaces with Sampling-Based Algorithms. IEEE Trans. Autom. Sci. Eng., vol. 13, № 2, 2016, pp. 415–424.
[123]. Klemm S., Oberlander J., Hermann A., Roennau A., Schamm T., Zollner J.M., Dillmann R. RRT∗-Connect: Faster, asymptotically optimal motion planning. 2015 IEEE Int. Conf. Robot. Biomimetics, 2015, pp. 1670–1677.
[124]. Starek J.A., Gomez J. V, Schmerling E., Janson L., Moreno Luis, Pavone M. An Asymptotically-Optimal Sampling-Based Algorithm for Bi-directional Motion Planning. IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2015, pp. 2072 – 2078.

An overview of modern methods for motion planning

[1]K.A. Kazakov <kazakov@ispras.ru>
[1,2]V.A. Semenov <sem@ispras.ru>
[1]Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia
[2]Moscow Institute of Physics and Technology (State University),
9 Institutskiy per., Dolgoprudny, Moscow Region, 141700, Russia

Abstract. Currently there is growing interest in motion planning problems that play a key role in automation of technologically difficult processes in mechanical, power and transport engineering, medicine, construction and creation of new products and services. This interest particularly relates to the increasing role of computer simulation and such disciplines as complex planning of industrial projects, realistic 3D animation, robotic surgery, automotive products assembly and organization of movement of transport streams. This paper is devoted to the overview and analysis of modern motion planning methods.

Keywords: motion planning, path planning, roadmaps, collision detection.

DOI: 10.15514/ISPRAS-2016-28(4)-14

For citation: Kazakov K.A., Semenov V.A. An overview of modern methods for motion planning. *Trudy ISP RAN/Proc. ISP RAS*, vol. 28, issue 4, 2016, pp. 241-294 (in Russian). DOI: 10.15514/ISPRAS-2016-28(4)-14

References

[1]. Choset H., Lynch K., Seth H., Kantor G., Burgard W., Kavraki L.E., Thrun S. Principles of Robot Motion-Theory, Algorithms , and Implementation. MIT Press, 2005.

[2]. Geraerts R., Overmars M.H. Creating High-quality Paths for Motion Planning. Int. J. Rob. Res., vol. 26, № 8, 2007, pp. 845–863.
[3]. Karaman S., Frazzoli E. Sampling-based algorithms for optimal motion planning. Int. J. Robot., vol. 30, № 7, 2011, pp. 846–894
[4]. Laumond J.-P. Kineo CAM: A success story of motion planning algorithms. IEEE Robot. Autom. Mag., vol. 13, № 2, 2006, pp. 90–93.
[5]. Rockel S., Klimentjew D., Zhang L., Zhang J. An hyperreality imagination based reasoning and evaluation system (HIRES). Proc. IEEE Int. Conf. on Robotics and Automation, 2014. pp. 5705–5711.
[6]. Sucan I., Moll M., Kavraki L.E. The Open Motion Planning Library. IEEE Robot. Autom. Mag. vol. 19, № 4, 2012, pp. 72–82.
[7]. Diankov R. Automated Construction of Robotic Manipulation Programs. PhD thesis, Carnegie Mellon University, Robotics Institute, August 2010, 263 p.
[8]. Porta J.M., Ros L., Bohigas O., Manubens M., Rosales C., Jaillet L. The CUIK Suite: Motion Analysis of Closed-chin Multibody Systems. IEEE Robot. Autom. Mag., vol. 21, № 3, 2014, pp. 105–114.
[9]. Semenov V.A., Kazakov K.A., Zolotov V.A. Effective spatial reasoning in complex 4D modeling environments. eWork Ebus. Archit. Eng. Constr. eds. A.Mahdavi, B. Martens, R. Scherer, CRC Press. Taylor Fr. Group, London, UK. 2015, pp. 181–186.
[10]. Kazakov K.A., Semenov V.A., Zolotov V.A. Topological Mapping Complex 3D Environments Using Occupancy Octrees. 21st Int. Conf. Comput. Graph. Vision, Sept. 26-30, 2011, Moscow, Russ. 2011, pp. 111–114.
[11]. Semenov V.A., Kazakov K.A., Morozov S. V., Tarlapan O.A., Zolotov V.A., Dengenis T. 4D modeling of large industrial projects using spatio-temporal decomposition. eWork Ebus. Archit. Eng. Constr. eds. K. Menzel R. Scherer, CRC Press. Taylor Fr. Group,London, UK. 2010, pp. 89–95.
[12]. Brooks R.A., Lozano-Peres T. A Subdivision Algorithm in Configuration Space For Find Path with Rotation. IEEE Trans. Syst. Man. Cybern., vol. SMC-15, № 2, 1985, pp. 224–233.
[13]. D. Zhu and J.-C. Latombe. Constraint reformulation in a hierarchical path planner. In Proc. IEEE Int. Conf. on Robotics and Automation, 1990, pp. 1918-1923.
[14]. Hornung A., Wurm K.M., Bennewitz M., Stachniss C., Burgard W. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. Auton. Robots., vol. 34, № 3, 2013, pp. 189–206.
[15]. Semenov V.A., Kazakov K.A., Zolotov V.A. Global path planning in 4D environments using topological mapping. eWork Ebus. Archit. Eng. Constr. 2012, pp. 263–269.
[16]. Chen D.Z., Szczerba R.J., Uhran Jr J.J. Using Framed-Quadtrees to Find Conditional Shortest Paths in an Unknown 2-D Environment. Technical Report #95-2, Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, Indiana, Jan. 1995, 33 p.
[17]. Chazelle B. Approximation and Decomposition of Shapes. Advances in Robotics 1: Algorithmic and Geometric Aspects of Robotics, 1987, pp. 145–185.
[18]. De Berg M., Cheong O., Van Kreveld M., Overmars M. Computational Geometry: Algorithms and Applications. 3rd Edition. Springer-Verlag, 2008, 386 p.
[19]. LaValle S.M. Planning Algorithms. Cambridge University Press, 2006, 844 p.
[20]. Canny J.F., Lin M.C. An opportunistic global path planner. Algorithmica, vol. 10, № 2-4, 1993, pp. 102–120.

[21]. Huang H.-P.H.H.-P., Chung S.-Y.C.S.-Y. Dynamic visibility graph for path planning. Proc. Int. Conf. Intell. Robot. Syst., vol. 3, 2004, pp. 2813–2818.
[22]. Kaluder H., Brezak M., Petrovic I. A visibility graph based method for path planning in dynamic environments. MIPRO, 2011, Proceedings of the 34th International Convention, Opatija, Croatia, 2011, pp. 717–721.
[23]. Aurenhammer F. Voronoi Diagrams – A Survey of a Fundamental Data Structure. ACM Comput. Surv., vol. 23, № 3, 1991, pp. 345–405.
[24]. Choset H. Sensor based motion planning: The hierarchical generalized voronoi graph. Ph.D. Thesis, California Institute of Technology, 1996, 201 p.
[25]. Russell S.J., Norvig P. Artificial Intelligence: A Modern Approach. Neurocomputing, vol. 9, № 2, 1995, pp. 215-218.
[26]. Bell S., An Overview of Optimal Graph Search Algorithms for Robot Path Planning in Dynamic or Uncertain Environments. Oklahoma Christian University, 2010, 9 p.
[27]. De Filippis L., Guglieri G. Advanced graph search algorithms for path planning of flight vehicles. Recent Adv. Aircr. Technol., 2012, pp. 159–192.
[28]. Zeng W., Church R.L. Finding shortest paths on real road networks: the case for A*. Int. J. Geogr. Inf. Sci., vol. 23, № 4, 2009, pp. 531–543.
[29]. Korf R.E. Depth-first iterative-deepening. An optimal admissible tree search. Artif. Intell., vol. 27, № 1, 1985, pp. 97–109.
[30]. Zhou R., Hansen E.A. Memory-Bounded {A*} Graph Search. The Florida AI Research Society Conference - FLAIRS, 2002, pp. 203–209.
[31]. Holte R., Perez M., Zimmer R., MacDonald A. Hierarchical A*: Searching abstraction hierarchies efficiently. Proceeding AAAI'96 Proceedings of the thirteenth national conference on Artificial intelligence – Volume 1, 1996, pp. 530–535.
[32]. Botea A., Muller M., Schaeffer J. Near optimal hierarchical path-finding. Journal of Game Development, vol. 1, issue 1, 2004, pp. 7-28.
[33]. Kring A., Champandard A., Samarin N. DHPA* and SHPA*: Efficient Hierarchical Pathfinding in Dynamic and Static Game Worlds. Proc. Sixth Artificial Intelligence and Interactive Digital Entertainment Conference, 2010, pp. 39–44.
[34]. Harabor D., Grastien A. Online Graph Pruning for Pathfinding On Grid Maps. AAAI Conf. Artif. Intell., 2011, pp. 1114–1119.
[35]. Koenig S., Likhachev M., Furcy D. Lifelong Planning A*. Artif. Intell., vol. 155, № 1-2, 2004, pp. 93–146.
[36]. Stentz A. The Focussed D* Algorithm for Real-Time Replanning. Proceeding IJCAI'95 Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2, 1995, pp. 1652-1659.
[37]. Koenig S., Likhachev M. D* Lite. Proceedings of the AAAI Conference of Artificial Intelligence (AAAI), 2002, pp. 476–483.
[38]. Koenig S., Likhachev M. Fast Replanning for Navigation in Unknown Terrain. IEEE Trans. Robot., vol. 21, issue 3, 2005, pp. 354-363..
[39]. Khatib O. Real time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics and Research, vol. 5, № 1, 1986, pp. 90–98.
[40]. Barraquand J., Latombe J.C. A Monte-Carlo algorithm for path planning with many degrees of freedom. Proceedings 1990 IEEE International Conference on Robotics and Automation, vol.3, 1990, pp. 1712–1717.
[41]. Barraquand J., Latombe J.C. Robot motion planning: A distributed representation approach. International Journal of Robotics Research, vol. 10, issue 6, 1991. pp. 628 - 649

[42]. Kavraki L.E., Svestka P., Latombe J.C., Overmars M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom., vol. 12, № 4, 1996, pp. 566–580.
[43]. Hsu D., Latombe J.C., Motwani R. Path Planning in Expansive Configuration Spaces. Proceedings 1997 IEEE International Conference on Robotics and Automation, vol. 3, 1997, pp. 2719–2726.
[44]. LaValle S.M., Kuffner J.J. Rapidly-exploring random trees: Progress and prospects. 2000 Workshop on the Algorithmic Foundations of Robotics, 2000, pp. 293–308.
[45]. Ferre E., Laumond J.-P. An iterative diffusion algorithm for part disassembly. Proceedings 2004 IEEE International Conference on Robotics and Automation, vol. 3, 2004, pp. 3149–3154.
[46]. Bayazit O.B., Lien J.-M., Amato N.M. Probabilistic Roadmap Motion Planning for Deformable Objects. Proceedings 2002 IEEE International Conference on Robotics and Automation, vol. 2, 2002, pp. 2126–2133.
[47]. Guibas L.J., Holleman C., Kavraki L.E. A Probabilistic Roadmap Planner for Flexible Objects with aWorkspace Medial-Axis-Based Sampling Approach. IEEE/RSJ Int. Conf. Intelligent Robot. Syst., 1999, pp. 254–260.
[48]. Berenson D., Srinivasa S., Kuffner J.J. Task Space Regions: A framework for pose-constrained manipulation planning. Int. J. Rob. Res., vol. 30, № 12, 2011, pp. 1435–1460.
[49]. Yakey J.H., LaValle S.M., Kavraki L.E. Randomized path planning for linkages with closed kinematic chains. IEEE Trans. Robot. Autom., vol. 17, № 6, 2001, pp. 951–958.
[50]. LaValle S.M., Kuffner J.J. Randomized Kinodynamic Planning. Int. J. Rob. Res., vol. 20, № 5, 2001, pp. 378–400.
[51]. Niederreiter H. Random number generation and Quasi-Monte Carlo methods. Society for Industrial and Applied Mathematics, 1992, 241 p.
[52]. Dmitriev K.A. From Monte Carlo to Monte Carlo Quasi. International Conference GraphiCon2002, 2002, pp. 53-59 (in Russian).
[53]. Sobol' I.M. Multidimensional quadrature formulas and Haar functions. M.: Home edition of Physical and Mathematical literature, Publishing house "Science", 1969, 288 p. (in Russian).
[54]. Sobol' I.M. Numerical Monte Carlo methods. M.: Home edition of Physical and Mathematical literature, Publishing house "Science", 1973, 312 p. (in Russian).
[55]. Khaksar W., Hong T.S., Khaksar M., Motlagh O. A low dispersion probabilistic roadmaps (LD-PRM) algorithm for fast and efficient sampling-based motion planning. Int. J. Adv. Robot. Syst.vol. 10, № 397, 2013, pp. 1-10.
[56]. LaValle S.M., Branicky M.S. On the Relationship Between Classical Grid Search and Probabilistic Roadmaps. In Algorithmic Foundations of Robotics V. Springer Tracts in Advanced Robotics, vol. 7, 2004, pp. 59-75
[57]. Zhang L., Kim Y.J., Manocha D. C-DIST: efficient distance computation for rigid and articulated models in configuration space. Proc. 2007 ACM Symp. Solid Phys. Model. - SPM ’07, 2007, pp. 159-169.
[58]. Lin M.C. Efficient Collision Detection for Animation and Robotics. Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, USA, 1993, 159 p.
[59]. Christer Ericson. Real-time Collision Detection. The Morgan Kaufmann Series in Interactive 3-D Technology. CRC Press, 2004, 632 p.

[60]. Andersen K. A., Bay C. A survey of algorithms for construction of optimal Heterogeneous Bounding Volume Hierarchies. Technical Report. Copenhague, Denmark: Department of Computer Science, University of Copenhagen, 2006, 32 p.
[61]. Zolotov V.A., Semenov V.A. Advanced indexing methods for large multidimensional data in complex dynamic scenes. Trudy ISP RAN/Proc. ISP RAS, vol. 24, 2013, pp. 381-416 (in Russian). DOI: 10.15514/ISPRAS-2013-24-17.
[62]. Zolotov V.A., Semenov V.A. [Effectie spatio-temporal indexing methods for visual modeling of large industrial projects]. Trudy ISP RAN/Proc. ISP RAS, vol. 26, issue 2, 2014, pp. 175-196 (in Russian). DOI: 10.15514/ISPRAS-2014-26(2)-9.
[63]. Zolotov V.A., S. P.K., Semenov V.A. Octree-based approach to spatial indexing of complex dynamic scenes. International Conference GraphiCon2015, Russia, 2015, pp. 115-122 (in Russian).
[64]. Pungotra H. Collision Detection and Merging of Deformable B-spline Surfaces in Virtual Reality Environment. Ph. D. thesis, The Univeristy of Western Ontario, Canada, 2010, 180 p.
[65]. Sandqvist J., Collision detection using boundary representation, BREP. Master thesis, Umeå University, Sweden, 2015, 54 p.
[66]. Su C.J., Lin F., Ye L. New collision detection method for CSG-represented objects in virtual manufacturing. Computers in Industry, vol. 40, № 1, 1999, pp. 1–13.
[67]. Klein J., Zachmann G. Point cloud collision detection. Proc. Eurographics 2004, 2004, pp. 567–576.
[68]. Schwarzer F., Saha M., Latombe J.C. Exact Collision Checking of Robot Paths. In Algorithmic Foundations of Robotics V. Springer Tracts in Advanced Robotics, vol. 7, 2004, pp. 25–41.
[69]. Yershova A., LaValle S.M. Improving Motion Planning Alorithms by Efficient Nearest-Neighbor Searching. IEEE Transactions on Robotics, vol. 23, issue 1, 2007, pp. 151–157.
[70]. Yershova A., LaValle S.M. Planning for closed chains without inverse kinematics. Department of Computer Science, University of Illinois, Urbana, USA, 2007, 7 p. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.115.7831&rep=rep1&type=pdf.
[71]. Brown R.A. Building a Balanced k-d Tree in O (kn log n) Time. J. Comput. Graph. Tech, vol. 4, № 1, 2015, pp. 50–68.
[72]. Ciaccia P., Patella M., Rabitti F., Zezula P. Indexing Metric Spaces with M-Tree. Proc. Atti del Quinto Convegno Nazionale SEBD, 1997, pp. 67–86.
[73]. Gipson B., Moll M., Kavraki L.E. Resolution Independent Density Estimation for motion planning in high-dimensional spaces. Proc. 2003 IEEE International Conference on Robotics and Automation, 2013, pp. 2437–2443.
[74]. Fredriksson K. Geometric Near-neighbor Access Tree (GNAT) revisited. arXiv:1605.05944v2, 2016, 7 p.
[75]. Yu C., Ooi B.C., Tan K.-L. Indexing the Distance: An Efficient Method to KNN Processing. Proceedings of the 27th International Conference on Very Large Data Bases, 2001, pp. 421-430.
[76]. Beygelzimer A., Kakade S., Langford J. Cover trees for nearest neighbor. ICML '06, Proc. 23rd Int. Conf. Mach. Learn., 2006, pp. 97–104.
[77]. Ichnowski J., Alterovitz R. Fast Nearest Neighbor Search in SE (3) for Sampling-Based Motion Planning. Algorithmic Foundations of Robotics XI, Volume 107 of the series Springer Tracts in Advanced Robotics, 2015, pp 197-214.

[78]. Amato N.M., Bayazit O.B., Dale L.K., Jones C., Vallejo D. Choosing good distance metrics and local planners for probabilistic roadmap methods. IEEE Trans. Robot. Autom., vol. 16, № 4, 2000, pp. 442–447.
[79]. Nissoux C., Simeon T., Laumond J.-P. Visibility based probabilistic roadmaps. 1999 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol. 3, 1999, pp. 1316–1321.
[80]. Amato N.M., Wu Y. A randomized roadmap method for path and manipulation planning. Proc. 1996 IEEE International Conference on Robotics and Automation, vol. 1, 1996, pp. 113–120.
[81]. Amato N.M., Bayazit O.B., Dale L.K., Jones C., Vallejo D. OBPRM: An Obstacle-Based PRM for 3D Workspaces. Proceeding WAFR '98 Proceedings of the third workshop on the algorithmic foundations of robotics on Robotics : the algorithmic perspective, 1998, pp. 155–168.
[82]. Yeh H.Y., Thomas S., Eppstein D., Amato N.M. UOBPRM: A uniformly distributed obstacle-based PRM. IEEE Int. Conf. Intell. Robot. Syst., 2012, pp. 2655–2662.
[83]. Hsu D., Jiang T., Reif J., Sun Z. The bridge test for sampling narrow passages with probabilistic roadmap planners. Proc. 2003 IEEE International Conference on Robotics and Automation, vol. 3, 2003, pp. 4420–4426.
[84]. Boor V., Overmars M.H., Stappen a. F. Van Der. The Gaussian sampling strategy for probabilistic roadmap planners. Proc. 1999 IEEE International Conference on Robotics and Automation, vol 2, 1999, pp. 1018–1023.
[85]. Lien J.-M., Thomas S., Amato N.M. A general framework for sampling on the medial axis of the free space. Proc. 2003 IEEE International Conference on Robotics and Automation, vol. 3, 2003, pp. 4439–4444.
[86]. Wilmarth S. a., Amato N.M., Stiller P.F. MAPRM: a probabilistic roadmap planner with sampling on the medial\naxis of the free space. Proc. 1999 IEEE International Conference on Robotics and Automation, vol 2, 1999, pp. 1024–1031.
[87]. Holleman C., Kavraki L.E. A framework for using the workspace medial axis in PRM planners. Proc. 2000 IEEE International Conference on Robotics and Automation, vol 2, 2000, pp. 1408–1413.
[88]. Nielsen C.L.L., Kavraki L.E. A two level fuzzy PRM for manipulation planning. 2000 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol. 3, 2000, pp. 1716–1721.
[89]. Bohlin R., Kavraki L.E. Path planning using lazy PRM. Proc. 2000 ICRA Millenn. Proc. 2000 IEEE International Conference on Robotics and Automation, vol 1, 2000, pp. 521–528.
[90]. Bohlin R., Kavraki L.E. A Randomized Approach to Robot Path Planning Based on Lazy Evaluation. In Handbook of Randomized Computing, volume I/II, Springer, 2001, pp. 221–253.
[91]. Leven P., Seth H. Toward Real-Time Path Planning in Changing Environments. Proceedings of the Fourth International Workshop on the Algorithmic Foundations of Robotics, 2000, pp. 363-376.
[92]. Nieuwenhuisen D., Van Den Berg J., Overmars M.H. Efficient path planning in changing environments. 2007 IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2007, pp. 3295–3301.
[93]. Jaillet L., Simeon T. A PRM-based motion planner for dynamically changing environments. 2004 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol. 2, 2004, pp. 1606–1611.
[94]. Bessiere P., Ahuactzin J.M., Talbi E.-G., Mazer E. The Ariadne's Clew Algorithm: Global Planning With Local Methods. Proceeding of the Workshop on Algorithmic Foundations of Robotics, 1995, pp. 39-49.

[95]. Mazer E., Ahuactzin J.M., Bessiere P. The Ariadne ' s Clew Algorithm. Journal of Artifical Intelligence Research, vol. 9, 1998, pp. 295–316.
[96]. Carpin S., Pillonetto G. Motion planning using adaptive random walks. IEEE Trans. Robot., vol. 21, № 1, 2005, pp. 543–548.
[97]. Carpin S., Pillonetto G. Merging the adaptive random walks planner with the randomized potential field planner. Proc. Fifth Int. Work. Robot Motion Control, 2005, pp. 151–156.
[98]. Sanchez G., Latombe J.C. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In Robotics Research, Volume 6 of the series Springer Tracts in Advanced Robotics, Springer, 2003, pp 403-417.
[99]. Hsu D. Randomized Single-query Motion Planning in Expansive Spaces. Ph. D. thesis, Stanford University, USA, 2000, 118 p.
[100]. Sagan H. Space-Filling Curves. Springer-Verlag, New York, 1994, 193 p.
[101]. Kuffner J.J., LaValle S.M. RRT-connect: An efficient approach to single-query path planning. Proc. 2000 IEEE International Conference on Robotics and Automation, vol 2, 2000, pp. 995–1001.
[102]. Bekris K.E., Chen B.Y., Ladd A.M., Plaku E., Kavraki L.E. Multiple Query Motion Planning using Single Query Primitives. IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2003, pp. 656–661.
[103]. Plaku E., Kavraki L.E. Distributed Sampling-Based Roadmap of Trees for Large-Scale Motion Planning. Proc. 2005 IEEE International Conference on Robotics and Automation, 2005, pp. 3879–3884.
[104]. Strandberg M. Robot Path Planning : An Object-Oriented Approach. Ph.D.thesis, Automatic and Control Departmentof, Signals, Sensors and Systems Royal Institute of Technology (KTH), Stockholm, Sweden, 2004, 244 p.
[105]. Bruce J.R., Veloso M. Real-time multi-robot motion planning with safe dynamics. Multi-Robot Systems. From Swarms to Intelligent Automata, Volume III. Proceedings from the 2005 International Workshop on Multi-Robot Systems, Springer, 2005, pp. 1–12.
[106]. Bruce J.R. Real-time motion planning and safe navigation in dynamic multi-robot environments. PhD. Thesis, Carnegie Mellon University, Pittsburgh, USA, 2006, 204 p.
[107]. Yershova A., Jaillet L., Siméon T., LaValle S.M. Dynamic-Domain RRTs: Efficient Exploration by Controlling the Sampling Domain. Proc. 2005 IEEE International Conference on Robotics and Automation, 2005, pp. 3856–3861.
[108]. Raveh B., Enosh A., Halperin D. A little more, a lot better: Improving path quality by a path-merging algorithm. IEEE Trans. Robot., vol. 27, № 2, 2011, pp. 365–371.
[109]. Urmson C., Simmons R. Approaches for heuristically biasing RRT growth. Proc. 2003 IEEE/RSJ Int. Conf. Intell. Robot. Syst., vol.2, 2003, pp. 1178–1183.
[110]. Jaillet L., Cortes J., Simeon T. Sampling-Based Path Planning on Costmaps Configuration-space. Ieee Trans. Robot., vol. 26, № 4, 2010, pp. 635–646.
[111]. Hauser K. Lazy collision checking in asymptotically-optimal motion planning. Proc. 2015 IEEE International Conference on Robotics and Automation, 2015, pp. 2951–2957.
[112]. Luo J., Hauser K. An Empirical Study of Optimal Motion Planning. IEEE/RSJ Conf. Intell. Robot. Syst., 2014, pp. 1761–1768.
[113]. Marble J.D., Bekris K.E. Asymptotically Near-Optimal is Good Enough for Motion Planning. 15th Int. Symp. Robot. Res., 2011, pp. 419-436.
[114]. Marble J.D., Bekris K.E. Computing spanners of asymptotically optimal probabilistic roadmaps. IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2011, pp. 4292–4298.
[115]. Marble J.D., Bekris K.E. Towards small asymptotically near-optimal roadmaps. Proc. 2012 IEEE International Conference on Robotics and Automation, 2012, pp. 2557–2562.

[116]. Dobson A., Bekris K.E. Improving Sparse Roadmap Spanners. Proc. 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 4106–4111.
[117]. Dobson A., Bekris K.E. Sparse roadmap spanners for asymptotically near-optimal motion planning. Int. J. Rob. Res., vol. 33, № 1, 2014, pp. 18–47.
[118]. Nasir J., Islam F., Malik U., Ayaz Y., Hasan O., Khan M., Muhammad M.S. RRT*-SMART: A rapid convergence implementation of RRT*. Int. J. Adv. Robot. Syst., vol. 10, 2013, pp. 1-12.
[119]. Arslan O., Tsiotras P. Use of relaxation methods in sampling-based algorithms for optimal motion planning. Proc. 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 2421–2428.
[120]. Janson L., Pavone M. Fast Marching Trees : a Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions – Extended Version, arXiv:1306.3532v4, 2013, pp. 1–60.
[121]. Salzman O., Halperin D. Asymptotically near-optimal RRT for fast, high-quality, motion planning. Proc. 2014 IEEE International Conference on Robotics and Automation, 2014, pp. 4680–4685.
[122]. Devaurs D., Simeon T., Cortes J. Optimal Path Planning in Complex Cost Spaces with Sampling-Based Algorithms. IEEE Trans. Autom. Sci. Eng., vol. 13, № 2, 2016, pp. 415–424.
[123]. Klemm S., Oberlander J., Hermann A., Roennau A., Schamm T., Zollner J.M., Dillmann R. RRT∗-Connect: Faster, asymptotically optimal motion planning. 2015 IEEE Int. Conf. Robot. Biomimetics, 2015, pp. 1670–1677.
[124]. Starek J.A., Gomez J. V, Schmerling E., Janson L., Moreno Luis, Pavone M. An Asymptotically-Optimal Sampling-Based Algorithm for Bi-directional Motion Planning. IEEE/RSJ Int. Conf. Intell. Robot. Syst., 2015, pp. 2072 - 2078.