УДК 004.8 **DOI** 10.15622/ia.22.6.4

В.А. Головин, К.С. Яковлев

ПРИМИТИВЫ ДВИЖЕНИЯ РОБОТА В ЗАДАЧЕ ПЛАНИРОВАНИЯ ТРАЕКТОРИИ С КИНЕМАТИЧЕСКИМИ ОГРАНИЧЕНИЯМИ

Головин В.А., Яковлев К.С. **Примитивы движения робота в задаче планирования траектории с кинематическими ограничениями.**

Аннотация. Автоматическое планирование траектории – актуальная научно-техническая задача, решения которой востребованы во многих областях: беспилотный транспорт, роботизированная логистика, социальная робототехника и т.д. Зачастую при планировании траектории необходимо учитывать тот факт, что агент (робот, беспилотный автомобиль и др.) не может произвольно менять ориентацию при движении, другими словами – необходимо учитывать кинематические ограничения при планировании. Одним из широко-распространенных подходов к решению этой задачи является подход, опирающийся на конструирование траектории из заранее подготовленных фрагментов, примитивов движения, каждый из которых в свою очередь удовлетворяет кинематическим ограничениям. Зачастую, акцент при разработке методов, реализующих этот подход, делается на сокращении перебора вариантов при планировании (эвристический поиск), при этом сам набор доступных примитивов считается заданным извне. В этой же работе, мы наоборот ставим своей целью провести исследование и анализ влияния различных доступных примитивов движения на качество решения задачи планирования при фиксированном алгоритме поиска. В частности, рассматриваются 3 различных набора примитивов движения для колесного робота с дифференциальным приводом. В качестве алгоритма поиска используется известный в искусственном интеллекте и робототехнике алгоритм A*. Качество решения оценивается по 6 метрикам, включая время планирования, длину и кривизну результирующей траектории. На основании проведенного исследования делаются выводы о факторах, оказывающих наибольшее влияние на результат планирования, и даются рекомендации по построению примитивов движения, использование которых позволяет достичь баланса между скоростью работы алгоритма планирования и качеством отыскиваемых траекторий.

Ключевые слова: планирование траектории, кинематическое планирование, примитивы движения, эвристический поиск.

1. Введение. Одной из наиболее активно-развивающихся областей современной науки и техники является разработка мобильных роботов, способных к функционированию в сложно-структурированных средах без активного участия оператора. Отдельный интерес представляют полностью автономные мобильные роботы, которые могут функционировать и решать поставленные перед ними задачи самостоятельно. Использование подобных систем в таких приложениях как беспилотный транспорт, доставка, логистика, мониторинг позволяет повысить эффективность решаемых задач (например, сократить время обработки и маршрутизации посылки на складе) [1, 2]. Более того, в ряде случаев использование роботов – это единственный способ

решения задачи в принципе. Например, при устранении последствий чрезвычайных ситуаций, когда нахождение человека в определенных областях невозможно из-за высокого риска потери здоровья (например, при проведении спасательных работ в непосредственной близости от поврежденных объектов ядерной энергетики).

Системы управления мобильными роботами с высокой степенью автономности обычно построены по модульному принципу [3]. Так, зачастую, выделяют следующие модули высокого уровня, каждый из которых, обычно, разбивается на множество более специализированных модулей:

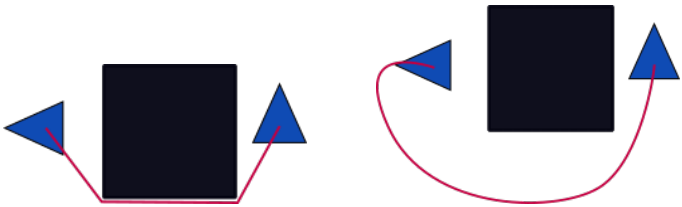
– модуль восприятия (в англоязычной терминологии – perception) – этот модуль ответственен за обработку и анализ данных, поступающих от датчиков, построение карты, локализацию робота в ней, выделение объектов интереса и т.д.;

– модуль планирования (в англоязычной терминологии – planning) – этот модуль ответственен за планировавание траектории с учетом информации, предоставленной модулем восприятия (карта, текущее положение статических препятствий, прогнозные траектории движения динамических препятствий и пр.);

– модуль управления (в англоязычной терминологии – control) – этот модуль ответственен за следование по построенной траектории.

Все указанные модули, безусловно, важны для эффективного функционирования системы управления мобильным роботом, однако в этой работе мы концентрируем свое внимание на модуле планирования траектории. В простейшем виде этот модуль по известной карте строит геометрический путь – ломанную линию, соединяющую две точки пространства, не пересекающую ни одно из нанесенных на карту препятствий. Очевидно, что такой подход целесообразен лишь, когда речь идет о построении траектории для робототехнических систем, обладающих возможностью движения в произвольном направлении без смены ориентации. Примерами таких систем являются квадрокоптеры или мобильные роботы с омни-колесами. В более общей постановке при планировании необходимо учитывать дополнительные ограничения робототехнической системы, которые могут быть весьма специфичны в различных случаях [4 – 6]. Если же концентрироваться на планировании траектории с точки зрения последующей ее выполнимости (с помощью модуля управления), то целесообразно учитывать в первую очередь ограничения на смену направления движения. Например, беспилотный автомобиль не может повернуться на 90 градусов мгновенно, ему требуется совершить поворот для этого, причем радиус этого поворота

– ограниченная снизу величина. Такие ограничения носят названия кинематических и именно проблеме автоматического планирования траектории с учетом кинематических ограничений и посвящена эта работа. Примеры траекторий, учитывающей (и не учитывающей) кинематические ограничения изображены на рисунке 1.



а) б)

Рис. 1. Пример: а) траектории не учитывающей; б) траектории учитывающей кинематические ограничения

Одним из широко распространенных подходов к планированию траектории с учетом кинематических ограничений является декомпозиционный подход, основанный на построении траектории из небольших фрагментов, т.н. примитивов движения. Каждый примитив – это рассчитанный для заданной робототехнической системы короткий участок траектории, удовлетворяющий кинематическим ограничениям движения. Набор всех возможных примитивов называется сеткой или решеткой примитивов (в англо-язычной литературе – lattice). На основе этой решетки в процессе планирования происходит эвристический перебор вариантов последовательного комбинирования примитивов до построения полной траектории, соединяющий стартовое и целевое положении. Обычно в литературе, посвященной такому подходу к планированию, основное внимание уделяется вопросу сокращения перебора (разработке различных эвристик, способов комбинации множества эвристических функций и пр.), при этом сама решетка примитивов считается заданной. В данной работе, мы рассматриваем обратную задачу – как повысить эффективность планирования траектории (при фиксированном алгоритме поиска) за счет вариации решетки примитивов. Для этого в работе разрабатываются различные наборы примитивов, проводится обширное экспериментальное исследование наборов, результаты которого используются для определения свойств примитивов положительно (и отрицательно) влияющих на качество планирования по шести количественным метрикам (включающим,

скорость работы алгоритма поиска, длину и кривизну отыскиваемых траекторий и др.).

2. Обзор литературы. Задаче планирования с учетом кинематических ограничений посвящено достаточно большое количество работ. В целом, методы решения этой задачи можно разделить на следующие классы: методы на основе сэмплирования, методы на основе оптимизации, методы на основе машинного обучения, методы на основе примитивов движения. Рассмотрим указанные классы методов более подробно.

2.1. Методы на основе сэмплирования. Широко используемым классом подходов является планирование на основе сэмплирования (sampling-based). При таком подходе планирование представляет собой стохастический процесс. Новые состояния на пути к целевому генерируются случайным образом, постепенно заполняя доступное пространство, в надежде однажды достигнуть цели. Несмотря на кажущуюся наивность такого подхода, подобные алгоритмы успешно применяются в различных автономных системах и прочно зарекомендовали себя в области планирования траектории. Популярным примером подобного алгоритма является алгоритм RRT [7] – Rapidly-exploring Random Trees. Его идея заключается в следующем: из стартового положения в случайном направлении ставится точка. Далее производится попытка соединить новое и старое положение с помощью метода, обеспечивающего соблюдение кинематических ограничений (т.н. локальное планирование). Если соединение удается, то новое состояние добавляется в дерево состояний, если – нет, то состояние отбрасывается. Далее процесс повторяется, и таким образом строится дерево из состояний, которое постепенно покрывает доступное пространство и в достигает желаемой цели (с вероятностью 1 при числе сэмплов, стремящемся к бесконечности). Соблюдение кинематических ограничений достигается за счет использования особых методов соединения состояний. В таких методах могут применяться, например, кривые Ридса-Шеппа или кривые Дюбинса, которые за счет параметризации позволяют учесть ограничения на максимальный радиус поворота. Также могут использоваться и более продвинутые методы, т.н. функции руления (англ. steeting function). В таких методах заложена кинематическая модель агента, которая используется для генерации траектории, соединяющий состояния в дереве. Примером подобного подхода служит метод POSQ [8].

Популярны различные модификации RRT. К примеру, RRT-connect [9] позволяет растить дерево одновременно из стартовой и

конечной точки, и прекращать поиск тогда, когда будет раскрыто состояние, принадлежащее обоим деревьям. Существуют и другие модификации [10 – 12], различным образом улучшающие базовый алгоритм.

Преимуществом сэмплирующих подходов является их простота, а также теоретическая возможность работы в непрерывном пространстве без необходимости дискретизации. С другой стороны, в практических задачах дискретизация зачастую возникает достаточно естественно, т.к. обычно робот строит именно дискретную модель представления окружающего пространства, а не непрерывную. Поэтому указанное теоретическое преимущество теряется. Более того из-за опоры на случайный выбор, алгоритм не гарантирует повторяемости результата, т.е. выдает разные решения одной и той же задачи. Также на практике существуют проблемы с подбором основного параметра алгоритма – числа сэмплов. При низком значении траектории отыскиваются быстро, но имеют низкое качество (большую длину, наличие большого числа поворотов и др.). При высоком значении параметра на отыскание траектории тратится слишком много времени. В целом задача подбора подходящего значения этого параметра не имеет общего решения.

2.2. Методы на основе оптимизации. Задача планирования траектории может быть математически сформулирована как задача оптимизации некоторой (непрерывной) функции – стоимости пути. Таким образом становится возможным использование оптимизационных подходов для планирования траектории. Подобные подходы в большинстве своем устроены следующим образом: на вход оптимизационному алгоритму дается начальная догадка – некоторая кривая, соединяющая начальную и конечную точку, далее производится попытка деформировать данную кривую так, чтобы она была оптимальной по заданным критериям. Для соблюдения кинематических ограничений, последние выражаются в аналитическом виде, например, в виде уравнений, которые определяют дополнительные ограничения оптимизационного процесса. Примеры таких алгоритмов представлены в работах [13 – 15]. Подобные подходы часто применяются при движении в хорошо структурированных средах, например на автодорогах общего пользования, когда в качестве начальной догадки берется середина дорожной полосы. В неструктурированной же среде вопрос формирования начальной догадки стоит острее. Зачастую в качестве такой догадки берется отрезок прямой между стартом и финишем, однако при такой начальной догадке методы оптимизации способны качественно строить траектории лишь при наличии препятствий простых форм (выпуклые многоугольники), которые на практике

встречаются довольно редко в неструктурированных средах. При сложной конфигурации препятствий процесс оптимизации зачастую сходится к локальному минимуму и не способен сгенерировать траектории, обходящие сложные препятствия.

Одним из существующих способов частичного решения подобной проблемы являются оптимизационные подходы, которые в качестве начальной догадки используют геометрическую траекторию, соединяющую стартовую и финишную точку в обход статических препятствий. Подобные методы также носят название пост-оптимизационных. Так, популярен метод пост-оптимизации GRIPS [16], или гибридный подход, который для второго этапа использует алгоритм Timed-Elastic Band (TEB) [17]. Методы пост-оптимизации так же, как и методы оптимизации сильно зависят от начальной догадки и при неудачном ее выборе не в состоянии найти какое-либо решение, даже если оно существует. Также большинство оптимизационных подходов требуют аналитического описания препятствий, что значительно затрудняет их применение в робототехнических системах, которые в основном используют клеточное представление об окружающем пространстве.

2.3. Методы на основе машинного обучения. В последнее время набирают популярность также методы основанные на машинном обучении. Данные подходы весьма разнообразны по архитектуре нейросетей, входным данным и параметрам. Часто используются алгоритмы [18 – 20] обучения с подкреплением (RL – Reinforcement learning), суть которых заключается во взаимодействии агента со средой и зависимости награды от действий агента. Отдельным классом являются так называемые end-to-end методы [21, 22], которые в качестве входных данных принимают наблюдения среды, а в качестве выходных данных выдают траекторию, или сразу управляющие воздействия. Учет кинематических ограничений осуществляется на этапе обучения – действия, которые ведут к их нарушениям штрафуются функцией потерь. Методы машинного обучения многообразны, и зачастую адаптированы под некоторую конкретную задачу, поэтому, к примеру, алгоритм движения по шоссе неприменим к алгоритму движения по парковке, что приводит к недостатку универсальности и сильной зависимости от входных данных у этих подходов.

2.4. Методы на основе примитивов движения. Но широко распространен и другой подход, при котором мы не адаптируем новые состояния агента под кинематические ограничения, а сразу генерируем дерево с учетом всех ограничений. Таким является подход с использованием примитивов движения. Его суть заключается в

построении траектории из отдельных фрагментов, – примитивов движения. Каждый такой примитив представляет собой дугу или прямолинейный участок, разработанный индивидуально для каждого набора кинематических ограничений. Проследовав любому из примитивов движения агент гарантированно удовлетворит всем заданным кинематическим ограничениям. Этот подход интересен тем, что мы заранее определяем наши возможные шаги в пространстве поиска, в противовес другим, например оптимизационным, алгоритмам, которые в свою очередь действуют в непрерывном пространстве, где поиск вариантов выполнять значительно сложнее. Плюс, мы не ограничиваем себя необходимостью проверять сконструированную траекторию на выполнимость – она при любом составе из примитивов будет удовлетворять всем кинематическим ограничениям, поскольку каждый элементарный примитив им удовлетворяет, а на стыке между двух примитивов гарантируется соблюдение граничных условий на гладкость необходимой степени.

Главным преимуществом является то, что с использованием примитивов движения мы можем выбрать любой эвристический алгоритм поиска – A*, Theta*, JPS (с некоторыми дополнительными ограничениями), TBA* и многие другие. Это возможно, поскольку мы не меняем принцип алгоритма, а меняем только способ генерации новых состояний. При этом даже с использованием таких простых алгоритмов, как A*, мы можем применить наш подход к решению современных и достаточно сложных задач [1]. Описанным ранее подходам присущи различные недостатки: случайность в сэмплирующих подходах, отсутствие гарантированного решения (при его существовании) в оптимизационных подходах, зависимость от входных данных в подходах на основе машинного обучения. Эвристические подходы лишены всех этих минусов, и напротив обладают собственными преимуществами – хорошо исследованное математическое обоснование, широкий выбор модификаций, возможность использования множества различных эвристических функций. Поэтому использование эвристических подходов открывает широкие возможности для решения задачи кинематического планирования.

2.4.1. Методы генерации примитивов. Ключевым вопросом является выбор подходящих примитивов движения. Именно от качества и приемлемости заданных примитивов движения и будет зависеть результирующая траектория. Так, например, довольно популярными в последнее время стали подходы, основанные на имитации траекторий человека-водителя [24, 25]. В таких работах обычно анализируются реальные данные, на основе которых происходит выделение и отбор

основных примитивов движения. Многие работы посвящены генерации примитивов на основе оптимизационных подходов [26, 27], что открывает новые возможности, ограниченные только выбором целевой функции. Также очень распространенными являются подходы основанные на динамических [28, 29] и адаптивных [30, 31] примитивах движения. Обе группы подходов отличает то, что примитивы в них могут изменяться от итерации к итерации, подстраиваясь под состояние агента и текущую задачу. Отличие между динамическими и адаптивными подходами заключается в том, что первые меняют примитивы пропорционально, т.е. один и тот же примитив может быть в два-три раза длиннее или короче на каждой итерации, а вторые же могут полностью менять форму примитива, а не только его размер.

Перечисленные методы могут достаточно подробно описывать способ генерации примитивов. Однако наблюдается явный недостаток исследований, направленных на сравнение различных примитивов между собой. Также неясным остается вопрос о качествах генерируемых примитивов – какими они должны быть? Длинными или короткими? Их должно быть много в наборе или мало? Данное исследование направлено на устранение этого пробела.

2.5. Результаты анализа литературы. В настоящее время существуют различные способы решения задачи планирования с кинематическими ограничениями, каждый из которых обладает определенными недостатками и преимуществами. Одним из перспективных является декомпозиционный подход на основе примитивов движения. При этом в литературе наблюдается недостаток исследований, направленных на анализ применимости различных примитивов движения к конкретной задаче и методики сравнения их между собой. Именно решению этой актуальной задачи и посвящена данная работа.

3. Планирование траектории с кинематическими ограничениями. Рассмотрим (точечного) мобильного робота, чья модель движения в общем случае имеет вид $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$, где $\mathbf{x}$ – состояние, $\mathbf{u}$ – управление, f – вектор-функция. Здесь и далее будем акцентировать свое внимание на колесном роботе с дифференциальным приводом. В этом случае имеем: $\mathbf{x}(t) = (x, y, \theta)$, $\mathbf{u}(t) = (v, \omega)$, а уравнения кинематики материальной точки представляются как:

$$)\dot{x} = v \cdot \cos(\theta) \qquad \dot{y} = v \cdot \sin(\theta) \qquad \dot{\theta} = \omega. \qquad (1)$$

Примитивом движения $\mathbf{p}$ назовем кортеж функций $(x(t), y(t), \theta(t)), t \in [0, t_f]$, где $t_f > 0$ – продолжительность примитива, которые удовлетворяют модели движения (1). Пусть $\hat{t} \in [0, t_f]$ – некоторый момент времени, тогда состояние робота в этот момент времени при совершении примитива $\mathbf{p}$ обозначим как $\mathbf{p}(\hat{t})$: $\mathbf{p}(\hat{t}) = (\hat{x}, \hat{y}, \hat{\theta})$, где $\hat{x} = x(\hat{t})$, $\hat{y} = y(\hat{t})$, $\hat{\theta} = \theta(\hat{t})$. Геометрически каждый примитив представляется как ограниченный (по длине) сегмент некоторой кривой. Пример изображен на рисунке 2.

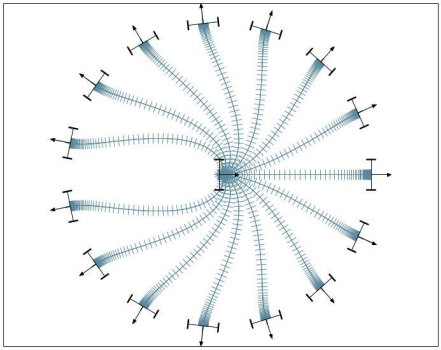


Рис. 2. Пример примитивов из работы [8]

С каждым примитивом движения будем ассоциировать действительное положительное число – стоимость (выполнения) примитива: $Cost(p) \in \mathbb{R}$. Чем меньше стоимость, тем предпочтительней примитив движения. В общем случае функция стоимости может быть произвольной и отражать любые пользовательские требования к задаче планирования. Самой распространенной функцией стоимостью является длина примитива. Также функция стоимости может учитывать кривизну, энергозатраты на выполнение примитива и пр.

Траекторией π назовем такую конечную последовательность примитивов, $\pi = \{p^i | \, i \in (0, N)\}$, $\pi[i] = \mathbf{p}^i$, для которой выполняются условия:

$$\mathbf{p^i}(t_f^i) = \mathbf{p^{i+1}}(0), \qquad \dot{\mathbf{p}}^{\mathbf{i}}(t_f^i) = \dot{\mathbf{p}}^{\mathbf{i+1}}(0), \quad \forall i \in (0, N-1), \qquad (2)$$

подразумевающие совпадение состояний и скоростей на стыке каждых двух последовательно идущих примитивов (другими словами, каждый

последующий примитив начинается из того же состояния, в котором заканчивается предыдущий).

Стоимостью траектории назовем сумму стоимостей входящих в нее примитивов $\text{Cost}(\pi) = \sum_i \text{Cost}(\mathbf{p}^i)$.

Определим далее рабочее пространство, в котором перемещается робот, как замкнутое подмножество на плоскости, $W \in \mathbb{R}^2$, состоящие из проходимых и непроходимых областей (препятствий): $W = W_{free} \cup W_{obst}$.

Задачей планирования траектории назовем кортеж $S = \langle W, \mathbf{x_0}, \mathbf{x_f} \rangle$, где $\mathbf{x_0}$, $\mathbf{x_f}$ – это начальное и целевое состояние робота соответственно. Без ограничения общности будем считать, что координаты начального и целевого положения принадлежат свободному пространству (в противном случае задача планирования считается некорректно поставленной).

Решением задачи планирования является траектория π, обладающая следующими свойствами:

$$\pi[0](0) = \mathbf{p^0}(0) = \mathbf{x}_0, \tag{3}$$

$$\pi[N](t_f^N) = \mathbf{p^N}(t_f^N) = \mathbf{x}_f, \tag{4}$$

$$\forall i \in (0, N),\ \forall t \in (0, t_f^i) : \mathbf{p^i}(t) \in W_{free}, \tag{5}$$

то есть начало траектории совпадает с начальным состоянием, конец траектории с конечным состоянием, а вся траектория содержится полностью в области свободного пространства (не пересекает препятствия).

Заметим, что в общем случае решение задачи планирования может быть не единственным. Обозначим множество всех решений задачи планирования как Π. Оптимальным решением задачи планирования называется траектория π^*, для которой выполняется условие:

$$\forall \pi \in \Pi : \text{Cost}(\pi^*) \leqslant \text{Cost}(\pi).$$

Другими словами, ни одна другая траектория из заданного начальное состояния в заданное конечное не имеет меньшей стоимости.

4. Исследуемая задача. Будем считать, что в нашем распоряжении имеется алгоритм, Alg, получения оптимального решения произвольной (корректно определенной) задачи планирования для заданной робототехнической системы. Формально алгоритм принимает на вход задачу планирования (описание рабочей области робота, начального

и целевого состояния), а также некоторый набор примитивов, из которых допускается конструирование траектории: $Alg(S, \mathbf{pr})$. Здесь $\mathbf{pr} = \{\mathbf{p}\}$ – это набор примитивов движения, удовлетворяющих кинематическим ограничениям (1). На выходе алгоритм возвращается оптимальную траекторию π^*.

Обычно работы по планированию траектории посвящены вопросам конструирования алгоритма Alg, исследования его теоретических свойств (доказательство сходимости, оптимальности), повышения вычислительной эффективности и так далее. С практической же точки зрения интерес представляет также изучение влияния второго аргумента, $\mathbf{pr}$) на эффективность работы Alg.

Для формализации этой задачи, которой и посвящена статья, рассмотрим конечную совокупность наборов примитивов: $\mathbf{pr_1}, \mathbf{pr_2}, ..., \mathbf{pr_n}$, каждый из которых может быть использован для получения оптимального решения задачи планирования (с помощью алгоритма Alg, который считается фиксированным). Введем также в рассмотрение набор числовых функций (метрик), ставящих в соответствие траектории некоторое число: $M = \{m_i\}$, $m_i(\pi) \in \mathbb{R}$. Каждая такая функция количественно отражает определенное качество траектории – степень кривизны, среднее расстояние до препятствий и т.д. Без ограничения общности будем считать, что время получения траектории, т.е. время работы алгоритма Alg, также относится к метрикам. Теперь для произвольной задачи планирования S и фиксированного набора примитивов из рассматриваемой совокупности, $\mathbf{pr_i}$, мы можем получить количественную оценку качества решения задачи планирования в виде набора числовых значений (метрик). Проводя анализ этих значений для разных наборов примитивов и на разных задачах планирования (при фиксированном Alg), мы стремимся выявить зависимость между видом используемых примитивов (количество примитивов в наборе, их длина, форма и пр.) и качеством решения задачи планирования. Нашей конечной целью является выработка общих рекомендаций (по внешнему виду примитивов, их количеству, длине, кривизне и пр.), которые необходимо учитывать исследователям и разработчикам, использующим методы планирования на основе примитивов движения для решения задач планирования траектории движения колесного мобильного робота с дифференциальным приводом (т.е. робота, движение которого описывается (1)).

Таким образом, рассматриваемая в данной работе задача заключается в выявлении влияния характеристик примитивов движения на эффективность работы алгоритма планирования траектории

с кинематическими ограничениями, а также в выработке общих рекомендаций по конструированию примитивов движения.

5. Методы. В данном разделе рассмотрены методы и подходы, необходимые для решения поставленной выше задачи, а именно: алгоритм генерации примитивов, метод планирования траектории, а также методика проведения анализа зависимости эффективности работы алгоритма планирования траектории от характеристик примитивов движения.

5.1. Метод генерации примитивов. Для генерации примитивов, учитывающих кинематические ограничения (1), использовался метод из [32]. Будем на него ссылаться как на $GenPrim$. Этот метод основан на принципе накрытия для решения краевых задач для плоских систем управления (подробнее – в оригинальной публикации). Он подразумевает получение примитивов движения в виде кривых третьего порядка:

$$x(t) = \alpha_0 \cdot t^3 + \alpha_1 \cdot t^2 + \alpha_2 \cdot t + \alpha_3, \tag{6}$$

$$y(t) = \beta_0 \cdot t^3 + \beta_1 \cdot t^2 + \beta_2 \cdot t + \beta_3, \tag{7}$$

$$\theta(t) = \arctan(\frac{\dot{y}}{\dot{x}}), \tag{8}$$

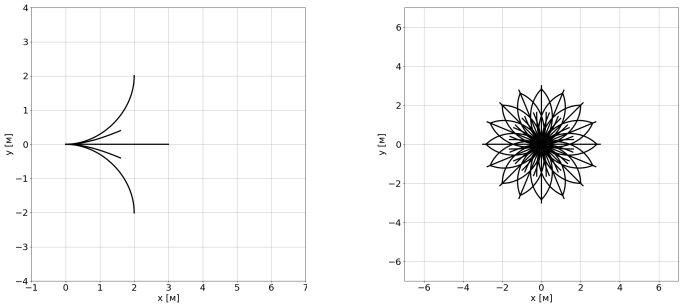
$$t \in [0,\ t_f]. \tag{9}$$

Здесь коэффициенты α_i, β_i определяются $GenPrim$, а продолжительность примитива t_f задается пользователем.

Для создания примитивов использовалась программная реализация $GenPrim$ на языке Matlab. В качестве входных данных передавалась желаемая финальная точка кривой (начальная точка всегда находится в начале координат), ориентация робота в начале и в конце движения, а так же предельное значение радиуса кривизны. Все эти параметры позволяли учитывать модель робота при разработке примитивов.

Изначально при генерации примитивов стартовая ориентация робота считалась равной нулю градусов. После генерации примитивов из этой ориентации полученный набор поворачивался на плоскости с некоторым, заранее определенным, шагом. В итоге получалась роза примитивов, покрывающих определенные стартовые ориентации агента. Финальные ориентации примитивов подбираются так, чтобы в итоге выйти в одну из допустимых стартовых ориентаций, т.е. дискретизация шага финальных ориентаций совпадала с дискретизацией шага стартовых ориентаций. Поскольку в робототехнике наиболее часто используются графы-сетки (occupancy grid) для моделирования окружающего робота пространства, то мы в своей работе также опирались на это представление.

Таким образом при генерации примитивов их длины подбирались так, чтобы соединять центры клеток в графе-сетке. Пример набора примитивов изображен на рисунке 3.



а) б)

Рис. 3. Пример примитивов движения: а) примитивы для одной ориентации; б) роза примитивов

Для каждого построенного примитива сохранялась информация о клетках графа-сетки, которые накрывает робот при движении по данному примитиву. Это необходимо для дальнейшего определения столкновений с препятствиями при планировании всей траектории. Визуально пример такого набора клеток можно увидеть на рисунке 4.

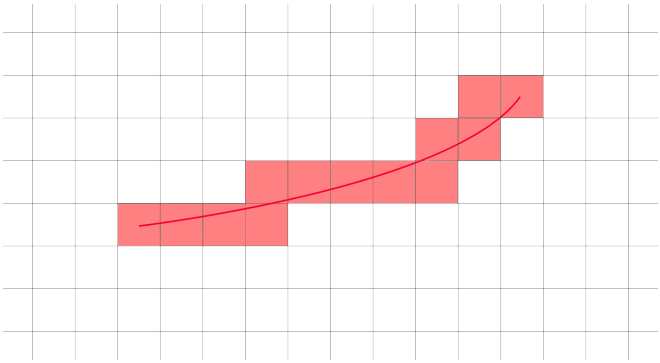


Рис. 4. Демонстрация множества клеток, накрываемого одним примитивом

Для расчета множества клеток использовался python-скрипт, код которого представлен в листинге 1. Здесь функция *calc_sweep_cells* принимает на вход отдельный примитив, далее увеличивая время t от 0 до t_f регистрируется положение агента вдоль кривой и вычисляется

текущая клетка пространства. Каждая клетка несет в себе целочисленные координаты, поэтому при разных значениях t агент может находится в пределах одной и той же клетки. Далее обновляется список перекрываемых клеток, содержащихся в классе примитива.

```
def calcSweepCells(prim)
    t_init = t = 0
    t_final = primitive.getFinalTime()
    cells = []
    while t != t_final:
        cell = prim.getOccupiedCell(t)
        if cell not in cells:
            cells.append(cell)
        t = t + dt
    return cells
```

Листинг 1. Алгоритм расчета перекрывыаемых клеток

Возникает вопрос о расчете стоимости отдельного примитива. В качестве базового решения может использоваться только общая длина. Однако эксперименты показали, что в таком случае траектории могут быть «виляющими» – это объясняется тем, что планирование ведется исключительно из соображения приблизиться к цели за как можно меньшее число итераций. Поэтому было принято решение об использовании модифицированной функции стоимости. В нашем случае функция стоимости представляет собой линейную комбинацию из длины примитива и его средней кривизны. Длина примитива находится из параметрических уравнений на x и y с помощью численного интегрирования:

$$L(p) = \int_0^{t_f} \sqrt{\dot{x}^2(t) + \dot{y}^2(t)} dt.$$

Для нахождения средней кривизны примитив разбивается на множество точек с некоторым шагом по времени Δt. В нашем случае этот шаг составлял 0.05 с. Далее считается кривизна в каждой полученной точке, не считая крайних. Напомним, что математически кривизна считается как обратный радиус кривизны траектории. После нахождения кривизны в каждой точке, находится средняя кривизна как среднее арифметическое. Таким образом:

$$NC(p) = \frac{1}{I-2} \cdot \sum_1^{I-1} \kappa_i = \frac{1}{I-2} \cdot \sum_1^{I-1} \frac{1}{R_i},$$

где I – количество промежуточных точек примитива, κ_i – кривизна i-й точки, R_i – радиус кривизны траектории в i-й точке.

Стоимость примитива тогда находится как:

$$Cost(p) = \alpha_1 \cdot L(p) + \alpha_2 \cdot NC(p),$$

где α_1, α_2 – линейные коэффициенты. Изменяя величину коэффициентов можно менять вклад той или иной составляющей в общую стоимость примитива. В рамках данной работы предлагается больший вклад уделить длине и меньший кривизне траектории. Как будет показано далее, характерное значения длины примитива $2m$, а кривизны $0.5\,m^{-1}$. Таким образом достаточно взять единичные коэффициенты, чтобы увеличить вклад линейной составляющей. Тогда стоимость примитива:

$$Cost(p) = L(p) + NC(p).$$

5.2. Метод планирования траектории. В качестве эвристического алгоритма планирования был выбран классический алгоритм A*. Он прост в реализации, удобен в использовании и позволяет добавить различные модификации, об одной из которых пойдет речь в разделе . В случае использования примитивов движения требуется изменить алгоритм генерации новых состояний, в частности с использованием предрассчитанных перекрываемых клеток указанных выше. В листинге 2 приведен псевдокод алгоритма генерации новых состояний. Функция *get_successors* принимает на вход состояние, из которого нужно сгенерировать потомков, а на выходе выдает список допустимых потомков. Для этого состояние передается в функцию *get_prims*, которая возвращает список допустимых примитивов, которые могут следовать из этого состояния. Примитивы отсеиваются по стартовому углу (он должен совпадать с ориентации в состоянии) и по коллизиям с препятствиями, которые находятся на карте около этого состояния. После выполнения этой функции из допустимых примитивов генерируются новые допустимые состояния (потомки) и, если ранее они не были исследованы, добавляются в итоговый список.

```
def getSuccessors(node):
    successors = []
    prims = getPrims(node)
    for prim in prims:
        new_node = genNode(node, prim)
        if new_node not in Closed:
            successors.append(new_node)
    return successors
```

```
. . .

def getPrims(node):
    prims = getPrimsByAngle(node.orientation)
    for prim in prims:
        if any prim.cell is in Map.obstacles:
            delete prim from prims
    return prims
```

Листинг 2. Алгоритм генерации новых состояний

Во всех остальных аспектах алгоритм полностью повторяет классический алгоритм A*. Результатом работы алгоритма является траектория π, состоящая из множества примитивов.

5.3. Методика проведения анализа. Задача выявления влияния характеристик примитивов движения на эффективность работы алгоритма планирования траектории может быть решена путем эмпирического исследования различных наборов примитивов движения и сравнения результатов работы алгоритма по совокупности количественных показателей. Такими показателями являются метрики качества – численные функции, отображающие траекторию на множество действительных чисел. Опишем их более подробно.

Метрики. Для оценки качества работы алгоритмов требуется рассчитать различные количественные метрики. Существуют разнообразные наборы данных и методик сравнения алгоритмов планирования [33 – 35], из которых можно выделить и использовать метрики качества. Нами был выбран широко известный и общепризнанный набор Bench-MR [36], которые направлен на тестирование алгоритмов в неструктурированной среде, содержащей только препятствия и испытуемого агента. Как будет показано в следующем разделе, данные условия полностью соответствуют нашей постановке, что и обосновывает сделанные выбор.

Для расчета каждой метрики требуется представить траекторию в виде последовательности точек (состояний). Поскольку в нашей постановке траектория представляет собой последовательность примитивов, требуется разбить каждый примитив на последовательность точек. Так как примитивы представляют собой параметрические уравнения, сделать это можно элементарно с помощью подстановки различных значений времени в выражения соответствующего примитива. Тогда для всей траектории последовательность точек представляет собой объединение последовательностей точек каждого из включенных примитивов движения.

Для дальнейшего пояснения введем обозначения: s_i – i-й элемент траектории, содержащий состояние (координаты и ориентация) агента, $d(s_i, s_j)$ – евклидово расстояние между состояниями, $K(s_{i-1}, s_i, s_{i+1})$ – кривизна участка траектории между тремя точками, считается с помощью подгонки окружности по трем точкам и последующим делением единицы на ее радиус. Также I – это количество промежуточных точек во всей траектории. Тогда перечень метрик:

– Длина траектории (Path length, LEN) – считается как:

$$LEN = \sum_{0}^{I-1} d(s_i, s_{i+1}).$$

– Расстояние до препятствий (Clearing) – показывает среднее расстояние траектории до препятствий Для расчетов используется предрасчитанная матрица расстояний, которая по размерам совпадает с тестовой картой, а в каждой клетке которой записано кратчайшее расстояние до препятствия из этой клетки. Тогда метрика считается как:

$$Clearing = \frac{\sum_{i=1}^{I} DistanceMatrix(s_i)}{I}.$$

– Угол-на-траекторию (Angle-over-length, AOL) – показывает в среднем насколько сильно меняется ориентация вдоль траектории, считается как:

$$AOL = \frac{\sum_{i=0}^{I-1} abs(s_i(\theta) - -s_{i+1}(\theta))}{LEN}.$$

– Средняя кривизна (Normalized curvature, NormC) – показывает среднюю кривизну, т.е. средний радиус поворота вдоль траектории, считается как:

$$NormC = \frac{\sum_{i=1}^{I-1} K(s_{i-1}, s_i, s_{i+1})}{I}.$$

Помимо этих метрик вводилась также дополнительная метрика отвечающая времени работы алгоритма – $ComputationTime$.

Методика сравнения. Сами по себе метрики качества не позволяют выполнить сравнение между различными наборами примитивов движения. Требуется определить правила упорядочивания результатов сравнения в зависимости от значения показателя. В соответствии с использованными метриками существует единственное правило, по которому меньшее

значение метрики соответствует лучшему результату. Действительно, длина таректории, время работы алгоритма, средняя кривизна и т.д. – показатели, значение которых ожидается на минимальном уровне.

Сравнение различных прмитивов движения выполнялось следующим образом:

1. На нескольких картах генерировалось множество различных заданий;

2. Для каждого сравниваемого набора примитивов проводился экспериментальный запуск алгоритма планирования для каждого из заданий;

3. По полученным траекториям выполняется подсчет метрик качества;

4. Для каждого набора примитивов на каждой карте подсчитывалось среднее из 100 значений метрик;

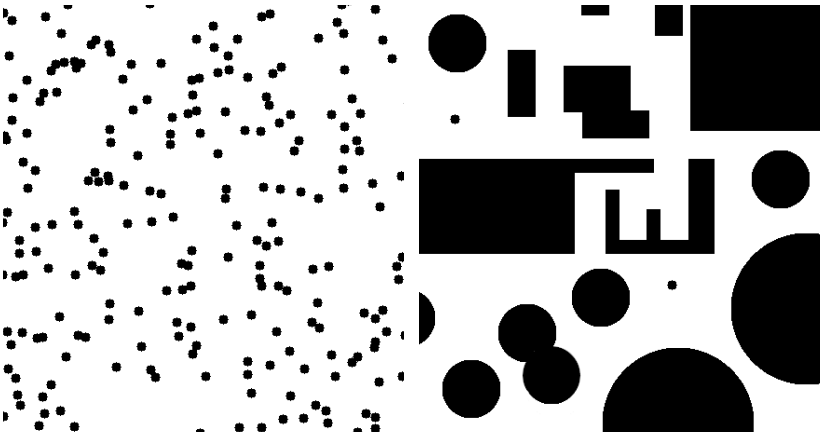
5. Полученные результаты заносились в таблицу и упорядочивались в зависимости от оцениваемого показателя.

Далее будет показано, что различные примитивы имеют преимущества по различным метрикам качества. Поэтому сравнивать наборы примитивов движения можно только по совокупности полученных результатов.

На основе выполненного сравнения строятся общие рекомендации ко свойствам генерируемых примитивов движения, следование которым должно привести исследователей к получению лучших результатов при решения задачи планирования траектории с кинематическими ограничениями.

6. Входные данные экспериментов. Для тестирования использовались две клеточные карты: первая (рисунок 5(а)) представляла собой свободное пространство, заполненное различными небольшими статичными препятствиями, вторая (рисунок 5(б)) моделировала реальную среду с препятствиями различного размера, в т.ч. зданиями. Каждая карта имела размер 600×600 клеток (пикселей), разрешение составляло 0.2 метра на клетку.

Было сгенерировано 100 различных заданий для каждой карты. Каждое задание представляло собой состояние (x, y, θ). Гарантируется выполнение каждого из заданий по отдельности для точечного дифференциального робота. Также были проведены дополнительные тесты на пустой карте, где 45 заданий были равномерно распределены по окружности вокруг агента, с ориентациями направленными от центра наружу.



а) б)

Рис. 5. Карты, используемые для тестирования: а) карта 1; б) карта 2

Для генерации примитивов был выбран шаг по ориентации в 22.5 градуса ($\frac{\pi}{8}$ радиан). Также для упрощения анализа скорость агента считается постоянной и равной 1 м/с. Для ускорения работы алгоритма была добавлена искусственная погрешность в 1 клетку для нахождения решения. Таким образом алгоритм может найти состояние не в точности совпадающее с заданным финальным, а отличающимся от него на эту погрешность. Также была добавлена погрешность в 11.25 градусов по ориентации, но это обусловлено только случайной генерацией ориентации финальных точек, а поскольку финальные ориентации примитивов дискретизованны, необходимо добавить погрешность размером 1/2 от шага в 22.5 градуса, чтобы гарантировать решение задачи.

Было сгенерировано 3 версии примитивов, но основе которых можно было делать выводы о связи параметров этих примитивов с качеством построенных траекторий. Все эти версии изображены на рисунке 6. Слева изображены примитивы для одной ориентации, справа – роза для всех ориентаций. Масштаб выбран единый для всех изображений, чтобы примитивы можно было сравнить визуально. Здесь и далее, если указывается угол у примитива, имеется ввиду его изменение по модулю от первоначального угла, т.е. если примитив стартует из ориентации $\frac{\pi}{8}$ и заканчивается с ориентацией $-\frac{\pi}{4}$, то соответствующее ему обозначение будет равно $abs(-\frac{\pi}{4} - (-\frac{\pi}{8})) = \frac{3\pi}{8}$.

В качестве первоначальной догадки были выбраны примитивы, изображенные на рисунке 6(а). На картинке слева видны примитивы

для одной ориентации агента. Для каждой ориентации насчитывается 6 примитивов. Классификация примитивов ведется по их финальной ориентации, таким образом в этом наборе присутствуют:

– 2 прямолинейных примитива длинами 0.4 и 3 метра,
– 2 примитива с финальными ориентациями $\frac{\pi}{8}$,
– 2 примитива с финальными ориентациями $\frac{\pi}{2}$.

Роза примитивов изображена на рисунке 6(б). Всего в наборе $6 \times 16 = 96$ примитивов.

Дальнейшие версии примитивов получались путем модификации первой версии – увеличении длины примитивов, количества примитивов, формы и т.д.

На рисунке 6(в) изображены примитивы второй версии. Их основное отличие в дополнительных кривых, цель добавления которых заключается в проверке гипотезы о том, как вариативность влияет на качество траекторий. Таким образом в данном наборе добавлены примитивы:

– Для движения по прямой со смещением;
– С другими выходными ориентациями $\frac{\pi}{4}$ и $\frac{3\pi}{8}$;
– Выходящие в ту же точку, что и примитив с ориентацией $\frac{\pi}{2}$, но с другими ориентациями – $\frac{3\pi}{8}$ и $\frac{\pi}{4}$.

Роза примитивов изображена на рисунке 6(г). Всего в наборе $17 \times 16 = 272$ примитива.

Далее на рисунке 6(д) изображены примитивы третей версии. Они отличаются от примитивов первой версии как увеличенными размерами, так и количеством примитивов и выходными ориентациями. Это позволит проверить, в первую очередь, как длина примитивов влияет на качество траекторий. Таким образом в наборе:

– Примитивы $\frac{\pi}{8}$ обладают большей длиной;
– Дополнительные прямолинейные примитивы длинами 1 и 6 м;
– Дополнительные примитивы с выходными ориентациями $\frac{\pi}{4}$ и $\frac{3\pi}{8}$.

Роза примитивов изображена на рисунке 6(е). Всего в наборе $14 \times 16 = 224$ примитива.

Также для наглядности была составлена таблица 1, содержащая информацию о примитивах в каждом наборе. В каждой строке таблицы указывается длина отдельного примитива, а в каждом столбце его финальная ориентация. На пересечении строки и столбца при наличии в наборе примитива с заданными параметрами ставится значок, соответствующий одному из трех наборов. Соответствие значков указано в описании к таблице.

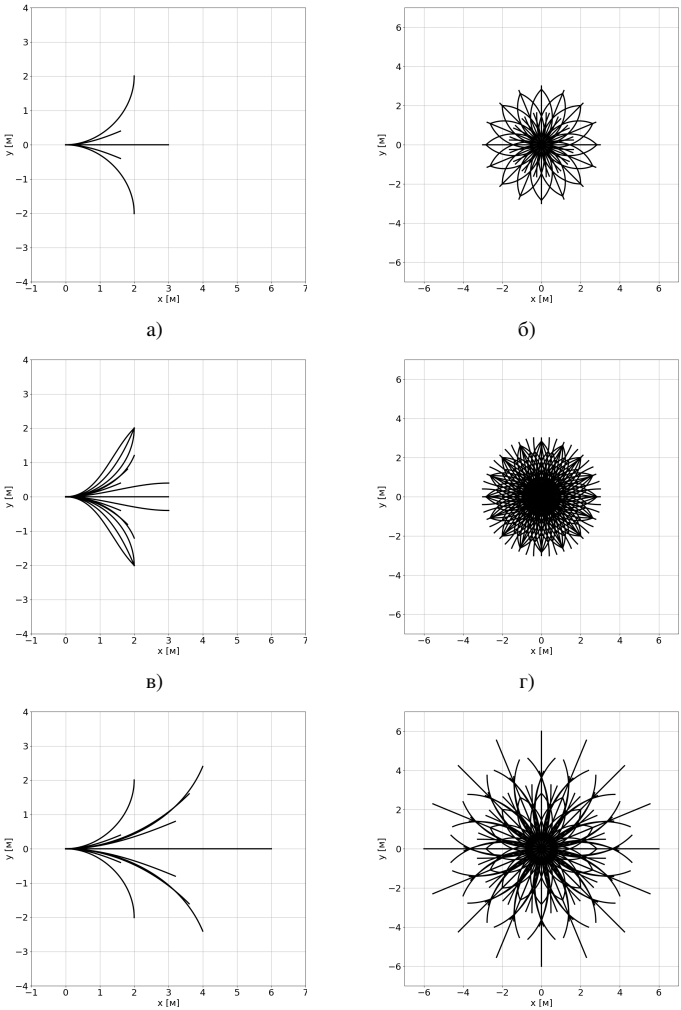


Рис. 6. Примитивы, используемые в экспериментах: а) примитивы 1; б) роза примитивов 1; в) примитивы 2; г) роза примитивов 2; д) примитивы 3; е) роза примитивов 3

Были проведены все тесты на всех картах и примитивах, и по итогу были получены величины метрик, которые были сведены в таблицы и графики.

Таблица 1. Сводная таблица с характеристиками примитивов в используемых наборах. Здесь символ в каждой ячейке обозначает наличие примитива с заданной длиной и результирующей сменой угла в соответствующем наборе. Введены обозначения: ▲ – 1-й набор, ■ – 2-й набор, ★ – 3-й набор

Длина (м) \ Угол	0	$\pi/8$	$\pi/4$	$3\pi/8$	$\pi/2$
0.400	▲■★				
1.000	■★				
1.660		▲■★			
2.020			■		
2.467				■	
2.935			■		
2.998				■	
3.000	▲■★				
3.032	■				
3.117					▲■★
3.320		★			
4.041			★		
4.934				★	
6.000	★				

7. Результаты экспериментов

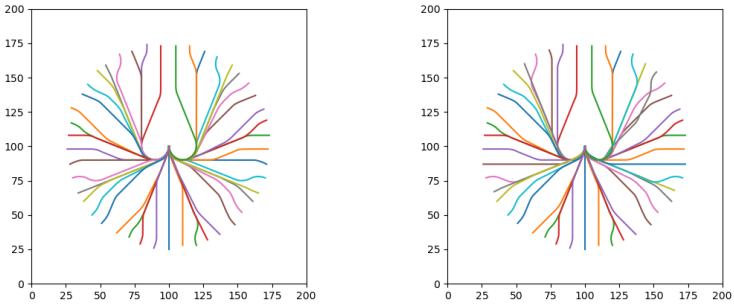
7.1. Анализ результатов. В ходе экспериментов были получены численные значения метрик из раздела 5.3, но для наглядности предлагается рассмотреть значения относительно результатов первых примитивов.

Согласно полученным результатам (таблица 2) видно, что вторая и третья версия примитивов находят решение примерно в 3 раза дольше, чем первый примитивы.

Таблица 2. Результаты проведенных экспериментов на обеих картах с использованием трех видов примитивов. В каждой ячейке указано итоговое значение метрики относительно первой версии примитивов. Жирным шрифтом выделены лучшие результаты в каждой метрике на каждой карте

	Карта 1			Карта 2		
Номер примитивов	I	II	III	I	II	III
Длина пути	1.0	**0.993**	0.994	1.0	0.986	**0.985**
Время работы	**1.0**	3.157	2.829	**1.0**	3.149	2.531
Угол-на-длину	1.0	**0.983**	1.007	1.0	**0.959**	1.001
Расстояние до препятствий	**1.0**	1.015	1.011	**1.0**	1.165	1.196
Средняя кривизна	1.0	**0.983**	1.006	1.0	**0.961**	1.001

При этом отклонение по остальным метрикам на первой карте составило порядка 1%, и порядка 2-4% на второй карте. Такая большая разница во времени работы свидетельствует о том, что любое увеличение количества примитивов приводит к существенному повышению времени работы алгоритма. Это лишний раз подтверждается тем, что вторые примитивы оказались самыми ресурсозатратными, поскольку имели в своем наборе наибольшее число примитивов. Для проверки связи между временем работы и количеством примитивов были проведены дополнительные эксперименты на пустой карте без препятствий. Стартовое состояние находилось в центре карты с ориентацией вниз, а задания в количестве 45 штук были равномерно распределены по окружности радиусом 75 м с ориентациями направленными радиально наружу. Если посмотреть на траектории, построенные на пустой карте примитивами 1 и 2 (рисунок 7), то можно заметить, что итоговые траектории очень похожи, что свидетельствует о применении одинаковых примитивов. Таким образом примитивы, добавленные во второй версии лишь увеличивают фактор ветвления поиска, но по сути не изменяют траектории.



а) б)

Рис. 7. Траектории, построенные с помощью первых и вторых примитивов на пустой карте: а) примитивы № 1; б) примитивы № 2

Рассмотрим значения остальных метрик. Результаты на карте 1 и карте 2 демонстрируют схожие тенденции, поэтому далее приводятся графики только для значений метрик на первой карте.

На графике распределения длин траекторий (рисунок 8(а)) можно заметить, что длины траекторий для всех трех версий практически совпадают, и лишь незначительно уменьшаются с увеличением средней длины примитивов.

Достаточно неожиданными оказались результаты по метрике Clearing. Если взглянуть на графики (рисунок 8(б)), то можно сделать вывод, что бóльшее разнообразие примитивов приводит к менее аккуратным траекториям.

Метрики средней кривизны (рисунок 8(в)) и угла-на-длину (рисунок 8(г)) наоборот демонстрируют большую похожесть между примитивами. Для того, чтобы это объяснить, обратимся к статистике по примитивам.

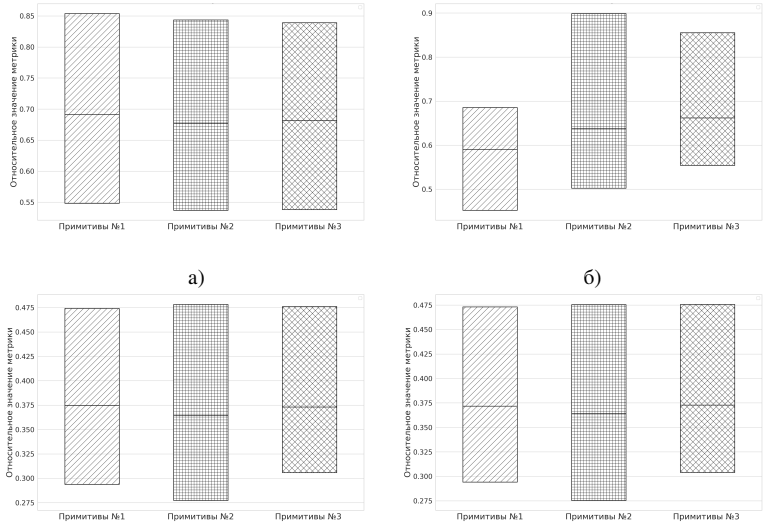


Рис. 8. Сводные графики метрик по результатам экспериментов на первой карте: а) длина пути; б) среднее расстояние до препятствий; в) средняя кривизна; г) угол-на-длину

Статистика использования примитивов состояла в трех сравнениях: соотношение между прямолинейными и криволинейными примитивами, соотношение между типами прямолинейных примитивов (по длине), и между типами криволинейных примитивов (по финальному углу). Полученные графики для обеих карт демонстрируют одинаковые тенденции, поэтому продемонстрируем рисунки только для первой карты (рисунок 9). Из графиков видно, что среди криволинейных примитивов всегда преобладает самый малый примитив с поворотом на $\frac{\pi}{8}$. Поэтому общая кривизна траектории равно как и изменения угла на единицу

длины обусловлены именно им. Так что нет ничего удивительного, что графики кривизны практически совпадают между первыми и вторыми примитивами, и лишь немного отличаются для третьих примитивов.

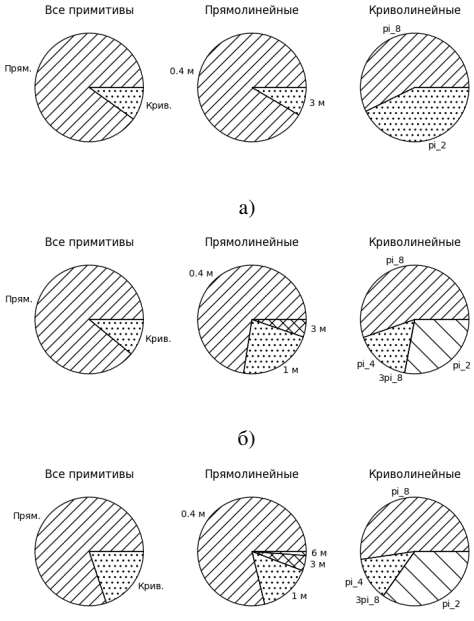


Рис. 9. Статистика использования примитивов: а) примитивы № 1; б) примитивы № 2; в) примитивы № 3

Также видно, что прямолинейные примитивы используются среди чаще криволинейных, причем предпочтение отдается опять же более коротким примитивам.

8. Заключение. В работе рассматривалась задача планирования траектории для мобильного робота с учетом кинематическим ограничений, в частности рассматривался случай колесного мобильного робота с дифференциальным приводом. Для решения этой задачи использовался подход планирования на основе примитивов движения. Основное внимание уделялось вопросу выявления взаимосвязей между характеристиками используемых примитивов и качеством решения задачи планирования траектории в неструктурированной среде. Качество решения оценивалось по шести различным метрикам. На основе более

двухсот заданий было произведено сравнение трех различных версий примитивов движения.

Анализ результатов проведенных экспериментальных исследований позволяет сделать следующие выводы:

1. Увеличение числа примитивов ведет к существенному (кратному) увеличению времени работы алгоритма, при этом улучшение качественных метрик (длина траектории, кривизна и пр.) – незначительно.

2. Увеличенная длина примитивов в наборе приводит к уменьшению общей длины траектории на 5–15%.

3. Прямолинейные примитивы используются гораздо чаще криволинейных, что объяснимо, поскольку поворотов в среднестатистической траектории гораздо меньше прямых участков.

4. Среди криволинейных примитивов самыми часто используемыми оказались кривые с малым углом поворота, при примерно равномерном распределении остальных типов.

Эти выводы позволяют сформировать следующий набор рекомендаций к набору примитивов для задач планирования с кинематическими ограничениями для колесного робота с дифференциальным приводом:

1. Количество примитивов на шаг угла составляет порядка 6–10 штук.

2. Преимущество при генерации отводится более коротким, маневренным примитивам.

3. Возможно добавление малого (1–2) числа удлиненных примитивов для получения более плавных траекторий на открытом пространстве.

Учет рекомендаций необходим для обеспечения наиболее качественного результата при планировании траектории с учетом кинематических ограничений.

Литература

1. Дудакова Д.С., Анохин В.М., Дудаков М.О., Ронжин А.Л. О теоретических основах аэролимнологии: изучение пресных водоемов и прибрежных территорий с применением воздушных робототехнических средств // Информатика и автоматизация. 2022. Т. 21. № 6. С. 1359–1393.
2. Балабанов А.Н., Безуглая А.Е., Шушляпин Е.А. Управление манипулятором подводного робота // Информатика и автоматизация. 2021. Т. 20. № 6. С. 1307–1332. DOI: 10.15622/ia.20.6.5.
3. Макаров Д.А., Панов А.И., Яковлев К.С. Архитектура многоуровневой интеллектуальной системы управления беспилотными летательными аппаратами // Искусственный интеллект и принятие решений. 2015. № 3. С. 18–33.
4. Otsu K., Matheron G., Ghosh S., Toupet O., Ono M. Fast approximate clearance evaluation for rovers with articulated suspension systems // Journal of Field Robotics.

2020. vol. 37. no. 5. pp. 768–785.

5. Al Mashhadany Y.I. Design and analysis of 7-DOF human-link manipulator based on hybrid intelligent controller // Informatics and Automation. 2020. vol. 19. no. 4. pp. 774–802.
6. Пшихопов В.Х., Медведев М.Ю. Планирование движения группы подвижных объектов в двумерной среде с препятствиями // Известия Южного федерального университета. Технические науки. 2016. Т. 2(175). С. 6–22.
7. LaValle S.M. Rapidly-exploring random trees: A new tool for path planning. Research Report 9811. 1998.
8. Kuffner J.J., LaValle S.M. RRT-connect: An efficient approach to single-query path planning // Proceedings ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings. 2000. vol. 2. pp. 995–1001.
9. Sharma P., Gupta A., Ghosh D., Honkote V., Nandakumar G., Ghose, D. PG-RRT: A Gaussian Mixture Model Driven, Kinematically Constrained Bi-directional RRT for Robot Path Planning // Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. 2021. pp. 3666–3673.
10. Webb D.J., Van Den Berg J. Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics // IEEE international conference on robotics and automation. IEEE. 2013. pp. 5054–5061.
11. Gammell J.D., Srinivasa S.S., Barfoot T.D. Informed RRT: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic // IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE. 2014. pp. 2997–3004.
12. Zhang Y., Sun H., Zhou J., Pan J., Hu J., Miao J. Optimal vehicle path planning using quadratic optimization for baidu apollo open platform // IEEE Intelligent Vehicles Symposium (IV). IEEE. 2020. pp. 978–984.
13. Li B., Wang K., Shao Z. Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach // IEEE Transactions on Intelligent Transportation Systems. 2016. vol. 17. no. 11. pp. 3263–3274.
14. Karlsson J., Murgovski N., Sjoberg J. Computationally efficient autonomous overtaking on highways // IEEE Transactions on Intelligent Transportation Systems. 2019. vol. 21. no. 8. pp. 3169–3183.
15. Heiden E., Palmieri L., Koenig S., Arras K.O., Sukhatme G.S. Gradient-informed path smoothing for wheeled mobile robots // Proceedings of the IEEE International Conference on Robotics and Automation. 2018. pp. 1710–1717.
16. Yongzhe Z., Ma B., Wai C.K. A practical study of time-elastic-band planning method for driverless vehicle for auto-parking // International Conference on Intelligent Autonomous Systems (ICoIAS). IEEE. 2018. pp. 196–200.
17. Kicki P., Gawron T., Cwian K., Ozay M., Skrzypczynski P. Learning from experience for rapid generation of local car maneuvers // Engineering Applications of Artificial Intelligence. 2021. vol. 105. pp. 104399. DOI: 10.1016/j.engappai.2021.104399.
18. Vitelli M., Chang Y., Ye Y., Ferreira A., Wolczyk M., Osinski B., Niendorf M., Grimmett H., Huang Q., Jain A., Ondruska P. Safetynet: Safe planning for real-world self-driving vehicles using machine-learned policies // International Conference on Robotics and Automation (ICRA). IEEE. 2022. pp. 897–904.
19. Nasiriany S., Pong V., Lin S., Levine S. Planning with goal-conditioned policies // Advances in Neural Information Processing Systems. 2019. vol. 32.
20. Chen L., Hu X., Tang B., Cheng Y. Conditional DQN-Based Motion Planning With Fuzzy Logic for Autonomous Driving // IEEE Transactions on Intelligent Transportation Systems. 2020. vol. 23. no. 4. pp. 2966–2977.

21. Wu K., Wang H., Esfahani M.A., Yuan S. Achieving Real-Time Path Planning in Unknown Environments Through Deep Neural Networks. IEEE Transactions on Intelligent Transportation Systems. 2022. vol. 23. no. 3. pp. 2093–2102.
22. Cohen B.J., Chitta S., Likhachev M. Search-based planning for manipulation with motion primitives // IEEE International Conference on Robotics and Automation. 2010. pp. 2902–2908. DOI: 10.1109/ROBOT.2010.5509685.
23. Low T., Bandyopadhyay T., Borges P.V. Identification of effective motion primitives for ground vehicles // IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE. 2020. pp. 2027–2034.
24. Wang B., Gong J., Chen H. Motion primitives representation, extraction and connection for automated vehicle motion planning applications // IEEE Transactions on Intelligent Transportation Systems. 2019. vol. 21. no. 9. pp. 3931–3945.
25. Jarin-Lipschitz L., Paulos J., Bjorkman R., Kumar V. Dispersion-minimizing motion primitives for search-based motion planning // IEEE International Conference on Robotics and Automation (ICRA). IEEE. 2021. pp. 12625–12631.
26. Bergman K., Ljungqvist O., Axehill D. Improved optimization of motion primitives for motion planning in state lattices // IEEE Intelligent Vehicles Symposium (IV). IEEE. 2019. pp. 2307–2314.
27. Koutras L., Doulgeri Z. Dynamic movement primitives for moving goals with temporal scaling adaptation // IEEE International Conference on Robotics and Automation (ICRA). IEEE. 2020. pp. 144–150.
28. Abu-Dakka F.J., Kyrki V. Geometry-aware dynamic movement primitives // IEEE International Conference on Robotics and Automation (ICRA). IEEE. 2020. pp. 4421–4426.
29. Sood R., Vats S., Likhachev M. Learning to use adaptive motion primitives in search-based planning for navigation // IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE. 2020. pp. 6923–6929.
30. Chernik C., Tajvar P., Tumova J. Robust Feedback Motion Primitives for Exploration of Unknown Terrains // IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE. 2021. pp. 8173–8179.
31. Palmieri L., Arras K.O. A novel RRT extend function for efficient and smooth mobile robot motion planning // Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2014. pp. 205–211.
32. Яковлев К.С., Белинская Ю.С., Макаров Д.А., Андрейчук А.А. Безопасно-интервальное планирование и метод накрытий для управления движением мобильного робота в среде со статическими и динамическими препятствиями // Автоматика и телемеханика. 2022. № 6. С. 96–117.
33. Wang X., Krasowski H., Althoff M. CommonRoad-RL: A configurable reinforcement learning environment for motion planning of autonomous vehicles // IEEE International Intelligent Transportation Systems Conference (ITSC). IEEE. 2021. pp. 466–472.
34. Ilievski M. Wisebench: A motion planning benchmarking framework for autonomous vehicles: MS thesis. Ontario: University of Waterloo, 2020. 129 p.
35. Xu C., Ding W., Lyu W., Liu Z., Wang S., He Y., Hu H., Zhao D., Li B. SafeBench: A Benchmarking Platform for Safety Evaluation of Autonomous Vehicles // Advances in Neural Information Processing Systems. 2022. vol. 35. pp. 25667–25682.
36. Heiden E., Palmieri L., Bruns L., Arras K.O., Sukhatme G.S., Koenig S. Bench-MR: A motion planning benchmark for wheeled mobile robots // IEEE Robotics and Automation Letters. 2021. vol. 6. no. 3. pp. 4536–4543.

Головин Владислав Андреевич — инженер, центр когнитивного моделирования, Московский физико-технический институт (МФТИ); студент, кафедра "научно-образовательный центр когнитивного моделирования", Московский физико-технический институт (МФТИ). Область научных интересов: искусственный интеллект, планирование траектории, эвристические алгоритмы. Число научных публикаций — 1. golovin.va@phystech.edu; Институтский переулок, 9, 141701, Долгопрудный, Россия; р.т.: +7(495)408-4554.

Яковлев Константин Сергеевич — канд. физ.-мат. наук, ведущий научный сотрудник, Федеральное государственное учреждение "Федеральный исследовательский центр "Информатика и управление" Российской академии наук" (ФИЦ ИУ РАН); ведущий научный сотрудник, Институт искусственного интеллекта AIRI. Область научных интересов: интеллектуальная робототехника, планирование траектории, эвристический поиск, много-агентные системы. Число научных публикаций — 134. yakovlev@isa.ru; проспект 60-летия Октября, 9, 141701, Москва, Россия; р.т.: +7(495)135-5457.

Поддержка исследований. Работа выполнена при поддержке Аналитического центра при Правительстве Российской Федерации в соответствии с договором о субсидии (идентификатор договора 000000D730321P5Q0002; грант № 70-2021-00138).

DOI 10.15622/ia.22.6.4

V. GOLOVIN, K. YAKOVLEV

MOTION PRIMITIVES IN THE TRAJECTORY PLANNING PROBLEM WITH KINEMATIC CONSTRAINTS

Golovin V., Yakovlev K. **Motion Primitives in the Trajectory Planning Problem with Kinematic Constraints.**

Abstract. Automatic trajectory planning is an urgent scientific and technical problem, whose solutions are in demand in many fields: unmanned transportation, robotic logistics, social robotics, etc. Often, when planning a trajectory, it is necessary to consider the fact that the agent (robot, unmanned car, etc.) cannot arbitrarily change its orientation while moving, in other words, it is necessary to consider kinematic constraints when planning. One widespread approach to solving this problem is the approach that relies on the construction of a trajectory from prepared parts, motion primitives, each of which satisfies kinematic constraints. Often, the emphasis in the development of methods implementing this approach is on reducing the combinations of choices in planning (heuristic search), with the set of available primitives itself being regarded as externally defined. In this paper, on the contrary, we aim to investigate and analyze the effect of different available motion primitives on the quality of solving the planning problem with a fixed search algorithm. Specifically, we consider 3 different sets of motion primitives for a wheeled robot with differential drive. As a search algorithm, the A* algorithm well known in artificial intelligence and robotics is used. The solution quality is evaluated by 6 metrics, including planning time, length and curvature of the resulting trajectory. Based on the study, conclusions are made about the factors that have the strongest influence on the planning result, and recommendations are given on the construction of motion primitives, the use of which allows to achieve a balance between the speed of the planning algorithm and the quality of the trajectories found.

Keywords: path planning, kinematic constraints, motion primitives, heuristic search.

References

1. Dudakova D.S., Anohin V.M., Dudakov M.O., Ronzhin A.L. [On Theoretical Foundations of Aerolimnology: Study of Fresh Water Bodies and Coastal Territories Using Air Robot Equipment]. Informatika i avtomatizaciya – Informatics and Automation. 2022. vol. 21. no. 6. pp. 1359–1393.
2. Balabanov A.N., Bezuglaya A.E., Shushlyapin E.A. [Underwater Robot Manipulator Control]. Informatika i avtomatizaciya – Informatics and Automation. 2021. vol. 20. no. 6. pp. 1307–1332. DOI: 10.15622/ia.20.6.5.
3. Makarov D.A., Panov A.I., Jakovlev K.S. Architecture of a multi-level intelligent control system for unmanned aerial vehicles. Iskusstvennyj intellekt i prinjatie reshenij – Artificial Intelligence and Decision Making. 2015. no 3. pp. 18–33.
4. Otsu K., Matheron G., Ghosh S., Toupet O., Ono M. Fast approximate clearance evaluation for rovers with articulated suspension systems. Journal of Field Robotics. 2020. vol. 37. no. 5. pp. 768–785.
5. Al Mashhadany Y.I. Design and analysis of 7-DOF human-link manipulator based on hybrid intelligent controller. Informatics and Automation. 2020. vol. 19. no. 4. pp. 774–802.
6. Pshihopov V.H., Medvedev M.J. Planning the movement of a group of moving objects in a two-dimensional environment with obstacles. Izvestija Juzhnogo federal'nogo universiteta.

Tehnicheskie nauki – News of the Southern Federal University. Technical science. 2016. vol. 2(175). pp. 6–22.
7. LaValle S.M. Rapidly-exploring random trees: A new tool for path planning. Research Report 9811. 1998.
8. Kuffner J.J., LaValle S.M. RRT-connect: An efficient approach to single-query path planning. Proceedings ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings. 2000. vol. 2. pp. 995–1001.
9. Sharma P., Gupta A., Ghosh D., Honkote V., Nandakumar G., Ghose, D. PG-RRT: A Gaussian Mixture Model Driven, Kinematically Constrained Bi-directional RRT for Robot Path Planning. Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. 2021. pp. 3666–3673.
10. Webb D.J., Van Den Berg J. Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. IEEE international conference on robotics and automation. IEEE. 2013. pp. 5054–5061.
11. Gammell J.D., Srinivasa S.S., Barfoot T.D. Informed RRT: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE. 2014. pp. 2997–3004.
12. Zhang Y., Sun H., Zhou J., Pan J., Hu J., Miao J. Optimal vehicle path planning using quadratic optimization for baidu apollo open platform. IEEE Intelligent Vehicles Symposium (IV). IEEE. 2020. pp. 978–984.
13. Li B., Wang K., Shao Z. Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach. IEEE Transactions on Intelligent Transportation Systems. 2016. vol. 17. no. 11. pp. 3263–3274.
14. Karlsson J., Murgovski N., Sjoberg J. Computationally efficient autonomous overtaking on highways. IEEE Transactions on Intelligent Transportation Systems. 2019. vol. 21. no. 8. pp. 3169–3183.
15. Heiden E., Palmieri L., Koenig S., Arras K.O., Sukhatme G.S. Gradient-informed path smoothing for wheeled mobile robots. Proceedings of the IEEE International Conference on Robotics and Automation. 2018. pp. 1710–1717.
16. Yongzhe Z., Ma B., Wai C.K. A practical study of time-elastic-band planning method for driverless vehicle for auto-parking. International Conference on Intelligent Autonomous Systems (ICoIAS). IEEE. 2018. pp. 196–200.
17. Kicki P., Gawron T., Cwian K., Ozay M., Skrzypczynski P. Learning from experience for rapid generation of local car maneuvers. Engineering Applications of Artificial Intelligence. 2021. vol. 105. pp. 104399. DOI: 10.1016/j.engappai.2021.104399.
18. Vitelli M., Chang Y., Ye Y., Ferreira A., Wolczyk M., Osinski B., Niendorf M., Grimmett H., Huang Q., Jain A., Ondruska P. Safetynet: Safe planning for real-world self-driving vehicles using machine-learned policies. International Conference on Robotics and Automation (ICRA). IEEE. 2022. pp. 897–904.
19. Nasiriany S., Pong V., Lin S., Levine S. Planning with goal-conditioned policies. Advances in Neural Information Processing Systems. 2019. vol. 32.
20. Chen L., Hu X., Tang B., Cheng Y. Conditional DQN-Based Motion Planning With Fuzzy Logic for Autonomous Driving. IEEE Transactions on Intelligent Transportation Systems. 2020. vol. 23. no. 4. pp. 2966–2977.
21. Wu K., Wang H., Esfahani M.A., Yuan S. Achieving Real-Time Path Planning in Unknown Environments through Deep Neural Networks. IEEE Transactions on Intelligent Transportation Systems. 2022. vol. 23. no. 3. pp. 2093–2102.
22. Cohen B.J., Chitta S., Likhachev M. Search-based planning for manipulation with motion primitives. IEEE International Conference on Robotics and Automation. 2010. pp. 2902–2908. DOI: 10.1109/ROBOT.2010.5509685.

23. Low T., Bandyopadhyay T., Borges P.V. Identification of effective motion primitives for ground vehicles. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE. 2020. pp. 2027–2034.
24. Wang B., Gong J., Chen H. Motion primitives representation, extraction and connection for automated vehicle motion planning applications. IEEE Transactions on Intelligent Transportation Systems. 2019. vol. 21. no. 9. pp. 3931–3945.
25. Jarin-Lipschitz L., Paulos J., Bjorkman R., Kumar V. Dispersion-minimizing motion primitives for search-based motion planning. IEEE International Conference on Robotics and Automation (ICRA). IEEE. 2021. pp. 12625–12631.
26. Bergman K., Ljungqvist O., Axehill D. Improved optimization of motion primitives for motion planning in state lattices. IEEE Intelligent Vehicles Symposium (IV). IEEE. 2019. pp. 2307–2314.
27. Koutras L., Doulgeri Z. Dynamic movement primitives for moving goals with temporal scaling adaptation. IEEE International Conference on Robotics and Automation (ICRA). IEEE. 2020. pp. 144–150.
28. Abu-Dakka F.J., Kyrki V. Geometry-aware dynamic movement primitives. IEEE International Conference on Robotics and Automation (ICRA). IEEE. 2020. pp. 4421–4426.
29. Sood R., Vats S., Likhachev M. Learning to use adaptive motion primitives in search-based planning for navigation. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE. 2020. pp. 6923–6929.
30. Chernik C., Tajvar P., Tumova J. Robust Feedback Motion Primitives for Exploration of Unknown Terrains. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE. 2021. pp. 8173–8179.
31. Palmieri L., Arras K.O. A novel RRT extend function for efficient and smooth mobile robot motion planning. Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2014. pp. 205–211.
32. Yakovlev K.S., Belinskaya Yu.S., Makarov D.A., Andrejchuk A.A. Safe interval planning and covering method for controlling the movement of a mobile robot in an environment with static and dynamic obstacles. Avtomatika i telemehanika – Automation and telemechanics. 2022. no. 6. pp. 96–117.
33. Wang X., Krasowski H., Althoff M. CommonRoad-RL: A configurable reinforcement learning environment for motion planning of autonomous vehicles. IEEE International Intelligent Transportation Systems Conference (ITSC). IEEE. 2021. pp. 466–472.
34. Ilievski M. Wisebench: A motion planning benchmarking framework for autonomous vehicles: MS thesis. Ontario: University of Waterloo, 2020. 129 p.
35. Xu C., Ding W., Lyu W., Liu Z., Wang S., He Y., Hu H., Zhao D., Li B. SafeBench: A Benchmarking Platform for Safety Evaluation of Autonomous Vehicles // Advances in Neural Information Processing Systems. 2022. vol. 35. pp. 25667–25682.
36. Heiden E., Palmieri L., Bruns L., Arras K.O., Sukhatme G.S., Koenig S. Bench-MR: A motion planning benchmark for wheeled mobile robots. IEEE Robotics and Automation Letters. 2021. vol. 6. no. 3. pp. 4536–4543.

Golovin Vladislav — Software engineer, Cognitive modeling center, Moscow Institute of Physics and Technology (MIPT); Student, Department "research and educational center for cognitive modeling", Moscow Institute of Physics and Technology (MIPT). Research interests: artificial intelligence, trajectory planning, heuristic algorithms. The number of publications — 1. golovin.va@phystech.edu; 9, Institutskiy Lane, 141701, Dolgoprudniy, Russia; office phone: +7(495)408-4554.

Yakovlev Konstantin — Ph.D., Leading researcher, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences (FRC CSC RAS); Leading researcher, Artificial Intelligence Research Institute (AIRI). Research interests: intelligent robotics, path planning, heuristic search, multi-agent systems. The number of publications — 134. yakovlev@isa.ru; 9, 60-letiya Oktyabrya Ave., 141701, Moscow, Russia; office phone: +7(495)135-5457.

Acknowledgements. This work was supported by the Analytical Center for the Government of the Russian Federation in accordance with a subsidy agreement (agreement identifier 000000D730321P5Q0002; grant no. 70-2021-00138).