УДК 007.52, 519.878, 519.1, 004.942, 006.72 **DOI** 10.15622/sp.18.1.57-84

Р.О. Лавренов, Е.А. Магид, Ф. Мацуно, М.М. Свинин, Дж. Сутакорн

РАЗРАБОТКА И ИМПЛЕМЕНТАЦИЯ СПЛАЙН-АЛГОРИТМА ПЛАНИРОВАНИЯ ПУТИ В СРЕДЕ ROS/GAZEBO

Лавренов Р.О., Магид Е.А., Мацуно Ф., Свинин М.М., Сутакорн Дж. **Разработка и имплементация сплайн-алгоритма планирования пути в среде ROS/Gazebo.**

Аннотация. Планирование пути для автономных мобильных устройств является важной задачей в робототехнике. При планировании пути принято использовать один из двух классических подходов: глобальный, когда карта полностью известна, и локальный, в котором устройство по мере движения обнаруживает препятствия с помощью различных бортовых датчиков. На основе этих двух подходов также создаются алгоритмы, сочетающие в себе сильные стороны глобального и локального планирования.

В ходе предыдущих исследований нами был разработан и реализован в среде Matlab прототип многокритериального сплайн-алгоритма глобального построения маршрута. Алгоритм использует граф Вороного при вычислении первой аппроксимации маршрута для запуска итерационного метода, что позволило находить путь во всех конфигурациях карты при условии существования пути от начальной точки до целевой точки. В ходе итеративного поиска использовалась целевая функция, в которой каждому критерию присваивался его вес в целевой функции. Для реализации критериев в том числе использовался метод потенциальных полей.

В данной статье представлена реализация модифицированного сплайн-алгоритма для применения его на реальных автономных мобильных робототехнических системах. Для этого проводится корректирование уравнений характеристических критериев оптимальности пути. Карта препятствий, представленная в ранней версии алгоритма в виде пересечений кругов, в реальных условиях может быть представлена в виде динамически изменяемой вероятностной карты на основе сетки занятости (OccupancyGrid), а робот уже не представляет из себя геометрическую точку.

Для реализации сплайн-алгоритма и дальнейшего использования его в системах управления реальных мобильных робототехнических устройств исходный код прототипа алгоритма был перенесен из среды Matlab в модуль программного обеспечения, написанный на языке программирования C++. Тестирование быстродействия алгоритма и оптимальность многокритериальной целевой функции проводились в среде ROS/Gazebo, являющимся на сегодняшний день де-факто стандартом программирования и моделирования робототехнических устройств.

Полученный в результате сплайн-алгоритм поиска пути можно интегрировать в системы управления наземных колесных и гусеничных робототехнических устройств, оборудованных лазерным дальномером, а также модифицировать предложенный алгоритм для использования шагающими наземными роботами, беспилотными летающими аппаратами и беспилотными судами. Алгоритм работает в режиме реального времени и параметры влияния критериев на целевую функцию доступны для динамических изменений во время движения мобильного робота.

Ключевые слова: планирование пути, мобильный робот, алгоритм планирования, ROS, Gazebo, сплайн-алгоритм.

1. Введение. Мобильная робототехника постепенно проникает во все сферы нашей жизни. Разнообразные робототехнические системы (РТС) постепенно заменяют людей на тяжелых и опасных

участках работ. Отдельно в этом списке стоят РТС, которых используют при различных техногенных и природных катаклизмах [1]. Благодаря РТС, способным проникать и обследовать опасные для человека зараженные или радиоактивные зоны, могут быть спасены человеческие жизни. В процессе выполнения поставленной задачи автономная или полуавтономная РТС, действующая индивидуально или в составе группы [2], не должна быть уничтожена, повреждена или потеряна вследствие, например, застревания среди препятствий окружающей среды или переворачивания в результате потери равновесия.

Планирование пути автономного РТС, наряду с автономным картографированием и локализацией [3], является важной научной задачей современной робототехники [4]. Основными источниками данных для одновременной локализации и картографирования (англ. simultaneous localization and mapping, SLAM) [5] и дальнейшего планирования пути являются лазерные дальномеры или цифровые камеры [6]. Рассчитываемый маршрут должен быть безопасным для РТС и удовлетворять различным требованиям, включая минимально возможную длину пути, отсутствие резких поворотов и петель, избегание непреодолимых препятствий и столкновений с другими роботами в случае работы в группе [7]. Алгоритм расчета маршрута для РТС должен всегда находить путь, если таковой существует; если путь найти невозможно – алгоритм должен информировать об этом систему управления РТС [8].

Методы планирования пути принято разделять на глобальные и локальные [9]. В первом случае РТС доступна полная карта препятствий окружающей среды, во втором – РТС стоит карту по мере своего продвижения к цели, используя бортовые датчики. К классическим глобальным алгоритмам планирования пути относятся все алгоритмы поиска по графу, построенному на основе карты окружающей среды, включая алгоритмы DFS, BFS, оптимальный алгоритм A* [10] и его частный случай алгоритм Дейкстры, основанный на быстрорастущих случайных деревьях алгоритм RRT [11] и другие.

Для планирования маршрута РТС при выполнении реальных задач чаще всего происходит комбинирование методов: имея первоначальную карту и построив по ней глобальный маршрут, РТС двигается к цели и при этом динамически перестраивает свой путь при обнаружении новых препятствий, используя локальные методы. Такой подход применяется как для наземных РТС, так и для беспилотных летательных аппаратов и беспилотных судов [12]. Примерами классических алгоритмов динамического планирования пути являются модификации алгоритма A* — алгоритм D* [13] и его более

распространенная версия D* Lite [14] — и алгоритмы семейства RRT, включая RRT* [15] и RRT^X [16].

Исследование проблемы оптимальности алгоритма планирования маршрута в рамках нашего исследования основано на ранних работах Магида и др. о сплайн-алгоритме для навигации автономной наземной РТС [17]. Прототип алгоритма был разработан в среде Matlab. Результаты тестирования прототипа, наряду с имевшимися изначально упрощениями реальной окружающей среды и препятствий, неприспособленностью прототипа алгоритма к работе в режиме реального времени, неумением работать с динамическими препятствиями и другими недостатками, показали, что в некоторых случаях алгоритм мог не находить существующий маршрут к целевой точке.

Для усовершенствования оригинального алгоритма было предложено воспользоваться графом Вороного [18]. В новом подходе граф Вороного используется для построения первоначального пути, который служит отправной точкой для итерационного поиска. В отличии от алгоритмов поиска по графу видимости и алгоритмов семейства RRT, которые находят кратчайший путь, граф Вороного позволяет находить наиболее безопасные для РТС маршруты относительно расстояний до препятствий окружающей среды в каждой точке маршрута, что обусловлено принципом построения графа.

Прототип нового алгоритма сначала был разработан и протестирован в среде Matlab [19]. Он гарантировано находил путь во всех случаях за существенно меньшее число итераций, чем оригинальный алгоритм [17]. Кроме того, в ходе исследования были предложены новые дополнительные критерии оценки качества маршрута и изучено их влияние на рассчитываемый путь.

В данной работе описывается реализация модифицированного сплайн-алгоритма для использования на реальных РТС. В ходе реализации был изменен ряд оригинальных уравнений, критериев и оптимизации [17], исходя из реального представления карт и невозможности реалистичного упрощения РТС до геометрической точки. Полученный в результате многокритериальный сплайн-алгоритм можно использовать при автономном планирования пути для систем управления РТС, оборудованных бортовым лазерным дальномером (2D лидаром), а также адаптировать его для использования с бортовыми 2D и 3D камерами.

2. Оригинальный сплайн-алгоритм планирования пути. Оригинальный алгоритм [17] предлагает решение задачи планирования пути для мобильного робота в известной среде, заполненной статическими препятствиями. Препятствия представляются в виде

конечной группы пересекающихся кругов различного диаметра. Каждая группа может содержать один или несколько кругов. Идея такого представления заключается в том, что любой многоугольник можно аппроксимировать конечной группой кругов.

Чтобы гарантировать построение пути, не пересекающего препятствий, отталкивающая потенциальная функция должна обладать высоким значением внутри препятствия и на его границе, и небольшим значением вне препятствия [20]. Таким образом, во время локальной процедуры оптимизации пути, высокое значение потенциальной функции в центре препятствия «выталкивает» все точки пути из препятствия, чтобы минимизировать стоимость пути [21]. Потенциальное поле начинает резко меняться (уменьшаться) на границе препятствия, продолжает уменьшаться с расстоянием, когда точка пути удаляется от границы препятствия, и довольно быстро обращается в ноль еще в непосредственной близости от препятствия. Метод потенциального поля является классическим методом планирования пути РТС, в том числе в динамических условиях [22].

Положение РТС в момент времени t определяется как конфигурация $q(t)=(x(t), y(t))$. Вклад одного круга (где круг является частью препятствия) в общую функцию потенциала отталкивания определяется следующим выражением:

$$U_{rep}(q) = 1 + \tanh(\alpha(\rho - \sqrt{(x(t)-x)^2 + (y(t)-y)^2})), \quad (1)$$

где ρ — радиус круга (препятствия) с центром в точке с координатами (x, y), а α — эмпирически определяемый параметр, который отвечает за выталкивание пути из границ препятствия. Рисунок 1 демонстрирует пример выбора параметра α=0.3 для среды с одним препятствием, которое образовано тремя пересекающимися кругами. Потенциальная функция имеет четко выделяющиеся пики на пересечениях кругов.

Функция топологии $T(q)$, учитывающая все N кругов окружающей среды и их влияние на робота на протяжении всего пути, которая определяется параметрическим выражением на интервале [0,1]:

$$T(q) = \sum_{j=0}^{N-1} \int_{t=0}^{1} U^{j}{}_{rep}(q) \cdot \delta l(t) \cdot dt, \quad (2)$$

где $\delta l(t)$ длина отрезка:

$$\delta l(t) = \sqrt{(x'(t))^2 + (y'(t))^2}. \quad (3)$$

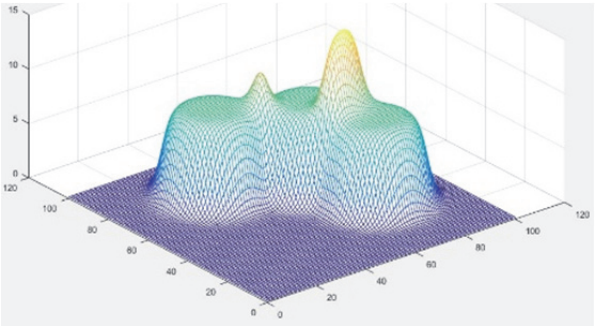

Рис. 1. Пример потенциальной функции для одного препятствия

Функция кривизны пути также определяется параметрическим выражением:

$$R(q) = \sqrt{\int_{t=0}^{1} (x''(t))^2 + (y''(t))^2 \, dt}. \quad (4)$$

Функция, которая учитывает длину пути ($L(q)$), суммирует длины всех сегментов пути:

$$L(q) = \int_{t=0}^{1} \delta l(t) \cdot dt. \quad (5)$$

Таким образом, итоговая оценка каждой траектории рассчитывается как сумма трех компонентов:

$$F(q) = \gamma_1 T(q) + \gamma_2 R(q) + \gamma_3 L(q), \quad (6)$$

где $\gamma_{i=1..3}$ — числовые значения влияния каждого компонента на общую стоимость маршрута. В частности, значение влияния потенциального поля $\gamma_{i=1} = \beta/2$, где параметр β эмпирически коррелирует с параметром α из уравнения (1).

Оригинальный сплайн-алгоритм планирования пути работает итеративно. Путь на первой итерации представляет из себя прямую линию между точками старта *S* и цели *T*. Сплайн определяется тремя точками: *S*, *T* и равноудаленной точкой, лежащей на прямом отрезке, соединяющем точки *S* и *T*. Дальнейшие итерации пути используют уравнение (6), и путь оптимизируется с помощью метода Нелдера-Мида (симплекс-метод, [23]). Лучший результат (маршрут) каждой итерации служит начальным значением маршрута для последующей итерации.

Процедура оптимизации работает только с точками пути, которые определяют его сплайн, в то время как оценка стоимости пути учитывает все точки маршрута. Маршрут перестраивается на каждой итерации, используя информацию с предыдущего этапа, увеличивая количество точек сплайна на единицу и корректируя параметры целевой функции стоимости пути. Алгоритм завершается, если количество итераций превышает заданный пользователем предел, или если новая итерация не улучшает предыдущую, но при этом увеличивается порядковая сложность сплайна.

Исходный сплайн-метод позволяет получить гладкий путь без столкновений с препятствиями, если каждое препятствие аппроксимируется одним кругом или линия начальной итерации не проходит через пересечение кругов. Однако, когда препятствия аппроксимированы группой пересекающихся кругов, пересечения образуют локальные максимумы потенциального поля (рисунок 1). Когда в такой среде начальный сплайн проходит через пересечение нескольких кругов, то целевая функция $F(q)$ «выталкивает» сплайн из области пересечения и далее сплайн «застревает» в локальных минимумах, а следующий итерационный сплайн не может «перепрыгнуть» некоторые компоненты препятствия из-за локального характера процесса оптимизации (рисунок 2).

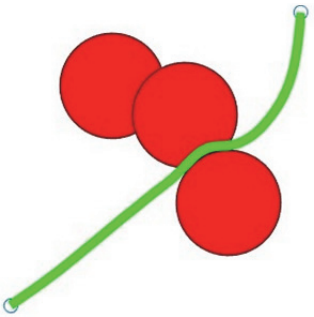

Рис. 2. Пример застревания пути в локальном минимуме потенциальной функции. В результате итеративного процесса путь все еще пересекает препятствия.

Из-за множества пересекающихся кругов, которые генерируют собственные локальные отталкивающие потенциалы, соответствующее глобальное потенциальное поле скрывает узкие проходы между различными препятствиями и позволяет совершить только локальную оптимизацию пути. Таким образом, локальная оптимизация успешно устраняет локальные максимумы, но даже после большого числа

итераций, результирующий путь все еще пересекает препятствия и, следовательно, бесполезен для движения робота.

3. Модифицированный алгоритм поиска пути с использованием графа Вороного. Для создания качественного начального сплайна применяется алгоритм построения графа Вороного [24], что помогает избежать ключевого недостатка оригинального подхода: невозможности выхода из локального минимума функции стоимости пути при неудачном стартовом сплайне независимо от количества последующих итераций алгоритма оптимизации [25]. Для создания графа Вороного требуется предварительно подготовить информацию о препятствиях; данная процедура выполняется в два следующих этапа [24]:

1) Компоновка отдельных препятствий из конечного множества пересекающихся кругов. Данная операция производится итеративно. Осуществляется поиск кругов, связанных пересечением, и такие группы кругов нумеруются как единое препятствие. Когда все круги окружающей среды идентифицированы как составляющие какого-либо препятствия, итеративный поиск завершается. Например, на рисунке 3 показаны четыре препятствия, которые образованы группами кругов, а на рисунке 4 в окружающей среде показаны три препятствия. При построении препятствия каждая пара пересекающихся кругов (i, j) образует одну точку пересечения θ_{ij} в случае граничного касания и пару точек θ_{ij}, θ_{ji} в общем случае пересечения кругов.

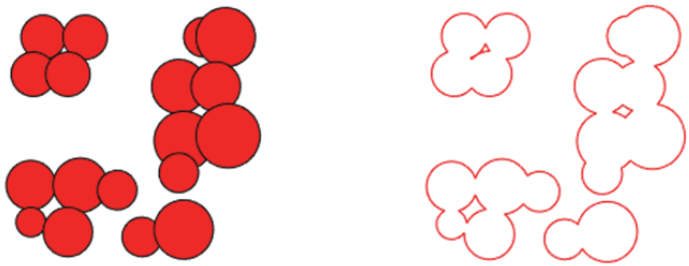

Рис. 3. Среда с препятствиями (слева) и рассчитанные контуры препятствий (справа)

2) Поиск внешних контуров препятствий множества $Obst = \{O_1, O_2, ..., O_k\}$, где k количество сложносоставных препятствий в окружающей среде. Для этого находятся точки пересечения окружностей и из внешних дуг окружностей собираются контуры препятствий. Эта процедура повторяется для каждого препятствия, и по ее завершению

формируется множество контуров. Если размер последнего множества превышает k, это указывает на наличие внутренних контуров. Чтобы избавиться от внутренних контуров, для каждого препятствия O_i вычисляется выпуклая оболочка; если у i-того препятствия имеется несколько контуров, удаляются контуры, которые находятся внутри выпуклой оболочки. К примеру, такая процедура успешно удалила три внутренних контура внутри сложносоставных препятствий среды на рисунке 3. Таким образом, получатся несколько невыпуклых многоугольников, по одному для каждого препятствия из *Obst*.

При построении графа Вороного помимо границ препятствий окружающей среды необходимо добавить внешние границы среды, что позволит находить линии графа, обходящие препятствия с внешней стороны. Для этого рассчитывается квадрат минимального размера со сторонами, параллельными осям глобальной системы координат окружающей среды, который включает в себя все препятствия среды (англ. axis-aligned bounding box — AABB), и каждая его сторона увеличивается на 40%. Пример такой внешней границы можно увидеть на рисунке 4 (центральное и правое изображения).

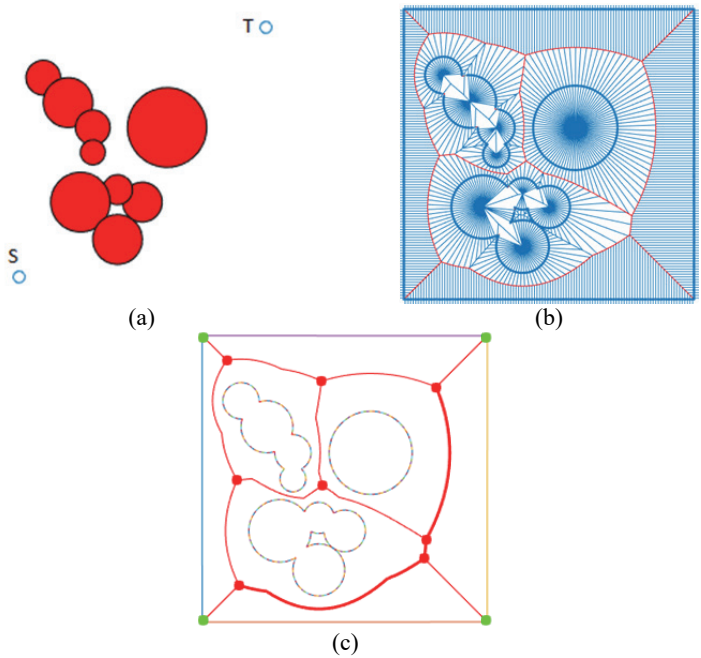

Рис. 4. Среда с препятствиями (a), процедура построения графа Вороного (b), путь на графе Вороного (c)

Далее, имея карту препятствий в виде многоугольников, которые являются приближением контуров препятствий, с использованием классического подхода строится граф Вороного.

1) От границ препятствий и карты с равными промежутками строятся лучи. На рисунке 4b демонстрируется пример построения этих лучей в виде тонких лучей, которые начинаются с границ препятствий и ограничивающей карты и являются перпендикулярами. Толстые линии на рисунке 4b отображают границы препятствий и карты.

2) Вычисляются точки пересечения лучей от разных препятствий. Точки пересечения соединяются отрезками, совокупность которых формируют граф Вороного (тонкие линии на рисунке 4b,c).

После построения графа Вороного, на нем выбираются точки S' и T' ближайшие к начальной позиции S и целевой позиции T, соответственно. Отрезки $[S, S']$ и $[T, T']$ не должны пересекать имеющиеся препятствия окружающей среды. Далее, кратчайший путь между S' и T' находится внутри графа Вороного с использованием алгоритма Дейкстры. На рисунке 4c построенный путь изображен толстыми красными линиями. Любой путь между точками (S, T) на графе Вороного гарантированно не пересекает препятствия и максимально безопасен в отношении расстояния от границ препятствий. Таким образом, он обеспечивает хороший начальный сплайн для исходного сплайн-метода [17].

Далее выбираются начальные точки будущего сплайна, которые надлежащим образом отражают характеристики пути, но при этом позволяют избежать избыточной сложности сплайна. Для этого среди всех точек пути вычисляется минимальный набор точек, последовательное соединение которых образует ломаную линию, не пересекающую препятствия и находящуюся в том же гомотопическом пространстве, что и найденный путь [24]. Найденный набор точек включает точки старта и цели. Данное множество точек используется для построения начального сплайна при запуске сплайн-метода.

Для проверки новый алгоритм был реализован в среде Matlab, и было проведено несколько тысяч симуляций на различных картах препятствий. Особое внимание было уделено случаям, когда оригинальный алгоритм [17] оказывался неудачным и не находил путь, застревая среди препятствий. Так, на рисунке 5 изображена среда, в которой оригинальный сплайн-алгоритм не смог построить маршрут. Граф Вороного предоставляет безопасный путь без столкновений с препятствиями. Следовательно, этот путь гарантирует, что модифицированный сплайн-алгоритм вычислит конечный путь меньшим числом итераций.

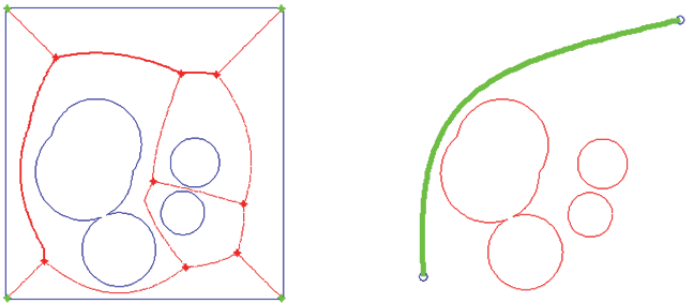

Рис. 5. Путь на графе Вороного (рисунок слева, толстая линия) и результирующий сплайн-путь, найденный итерационным алгоритмом на основе первоначального пути по графу (рисунок справа).

В тестах использовалась целевая функция уравнения (6) со следующим выбором параметров: $\gamma_1 = 1$, $\gamma_2 = 1$, $\gamma_3 = 0.5$. Алгоритм смог успешно рассчитать маршрут во всех случаях, что было естественным следствием применения начального пути, найденного по графу Вороного, в качестве первого сплайна для итерационного алгоритма.

4. Фреймворк ROS и среда симуляции Gazebo. Робототехническая операционная система ROS (англ. «Robot Operating System») — платформа с открытым исходным кодом для моделирования и программирования РТС, которая на сегодня де-факто является стандартом в программировании РТС. Она предоставляет такие услуги операционной системы, как управление устройствами на низком уровне, аппаратную абстракцию, передачу сообщений между процессами и управление пакетами. Несколько взаимосвязанных процессов, выполняющихся в ROS, могут быть представлены в виде графа, основными элементами которого являются узлы, называемые нодами. Ноды могут принимать и передавать данные от датчиков, приводов, энкодеров и обрабатывать различные сообщения. Программное обеспечение на основе ROS можно разделить на три группы:

1. Языки и независимые от платформы инструменты, используемые для создания и распространения программного обеспечения.

2. Клиентские библиотеки для разработки собственных пакетов, такие как roscpp, rospy и roslisp.

3. Готовые клиентских библиотеки для различных датчиков, роботов и отдельных алгоритмов.

Для моделирования окружения робота с ROS могут применяться симуляторы RViz или Gazebo. Gazebo использует Open

Dynamics Engine как физический движок и объектно-ориентированный графический рендеринг для 3D-рендеринга, имеет бесплатную лицензию для исследователей, предоставляет графический интерфейс пользователя и позволяет описывать динамику роботов на языке программирования Python. Пример созданного окружения и запущенный в нем робот Husky [26] можно увидеть на рисунке 6.

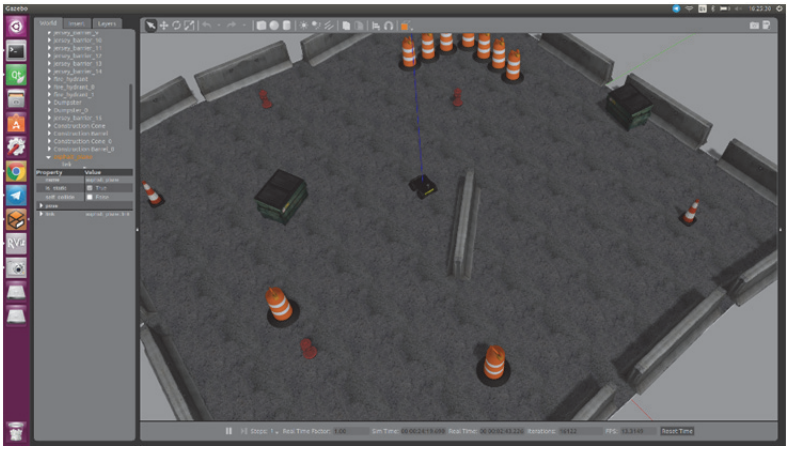

Рис. 6. Робот Husky в симуляторе ROS\Gazebo

Подробные туториалы и документация ROS предоставляют шаблоны для реализации собственных пользовательских алгоритмов планирования маршрута. Однако, в ROS в настоящее время реализовано лишь несколько алгоритмов планирования пути автономных РТС [27]. Наиболее распространены алгоритмы, работающие с данными лазерного дальномера. В этих случаях карта окружающей среды является так называемой OccupancyGrid картой — решеткой занятости (рисунок 7). То есть двумерным массивом значений, каждое из которых обозначает наличие препятствия в указанной области. Значения в ячейках могут быть бинарными (0 для пустых ячеек, 1 для занятых препятствием ячеек) или принимать значения в заданном диапазоне, обозначая неровности поверхности и проходимость указанной области. Значения занятости по умолчанию находятся в диапазоне от 0 до 100. Решетка занятости может быть визуализирована в виде изображения в черно-белом виде с значениями равными 0 для полностью свободных и 100 для абсолютно непроходимых областей. Серые области обозначают неизвестные области карты.

Реализованные в ROS алгоритмы планирования упрощают задачу поиска пути с помощью искусственного программного

расширения найденных на карте препятствий на физический размер робота. Пример таких областей можно увидеть на рисунке 8.

Рис. 7. OccupancyGrid карта, построенная РТС «Сервосила Инженер» при помощи лазерного дальномера Hokuyo UTM-30LX-EW LRF

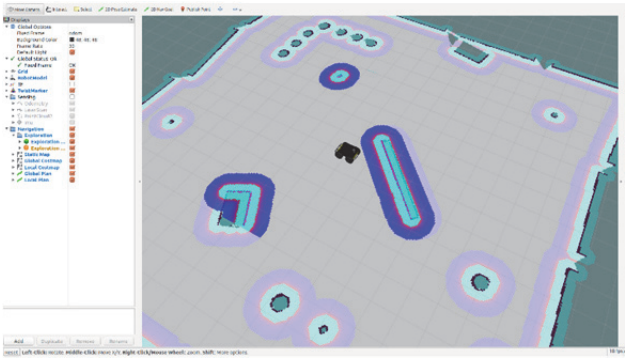

Рис. 8. Робот Husky в симуляторе RViz

Препятствия на рисунке 8 показаны бирюзовым цветом, красным цветом показаны границы препятствий. Синим цветом выделены области вокруг препятствий, которые исключаются из планирования пути, так как считаются опасными ввиду близости к границам препятствий. Окружающая среда, в которой находится РТС, идентична представленной ранее на рисунке 6 среде. При проведении расчетов РТС становится математической точкой. Работа этого процесса в ROS обеспечивается библиотекой Costmap. Данная библиотека является частью стека навигации — набора библиотек, обеспечивающих глобальное и локальное планирование пути,

локализацию, картографирование и вывод робота из состояния застревания [28]. Сплайн-алгоритм будет реализовываться как глобальный алгоритм для стека навигации ROS.

5. Реализация модифицированного сплайн алгоритма в ROS/Gazebo. Основной новизной модифицированного сплайн-алгоритма относительно оригинального алгоритма является использование предварительно рассчитанного графа Вороного по имеющейся карте местности. Проблема реализованного в Matlab первого прототипа нового алгоритма [24, 25] заключается в том, что он не подходит для реального использования системой управления РТС. Если в среде Matlab препятствия представлялись в виде пересечений кругов, реальные карты, получаемые мобильной РТС с помощью лазерного дальномера, представляют из себя ячейки в OccupancyGrid карте с вероятностью выше определенного предела. Двумерные вероятностные сетки занятости являются наиболее распространенным методом представления окружающей среды [29].

Для расчета диаграммы Вороного было решено использовать подход, описанный в работе [30]. Суть его заключается в проходе по всей двумерной карте и нахождение для каждой свободной ячейки расстояния до ближайшей ячейки с препятствием. Для каждой ячейки запоминается индекс ближайшей ячейки с препятствием. Далее, соседние ячейки с препятствиями, имеющие общие точки, объединяются и маркируются как единое препятствие [31]. После этого производится повторный проход по всем индексам карты. Проверяется каждая ячейка — если у нее ближайшее препятствие с номером i, а у ее соседней ячейки ближайшее препятствие с номером, отличным от i, то обе ячейки специальным образом маркируются. После прохода по всей карте совокупность промаркированных ячеек образует граф Вороного; пример карты с выделенными красным цветом маркированными ячейками представлен на рисунке 9. Данный алгоритм так же может быть адаптирован и расширен для применения на 3D карте, если препятствия будут представлены в виде вокселей [32].

Метод расчета графа Вороного на дискретной вероятностной карте для ROS [30] был предварительно реализован в статье [33], где автор применил алгоритм для планирования пути автономного мини-судна. Автор использует специальную структуру для различных маркировок каждой ячейки карты, а также для хранения в ней расстояния до препятствий и индексы этих препятствий. После построения графа Вороного, по нему в три этапа осуществляется поиск пути из точки старта в целевую точку в три этапа:

1. Из точки старта S осуществляется поиск кратчайшего пути до ближайшей точки графа S_1

2. Из целевой точки T осуществляется поиск кратчайшего пути до ближайшей точки графа T_1

3. Используя только помеченные как принадлежащие к графу Вороного точки карты, ищется путь от S_1 до T_1

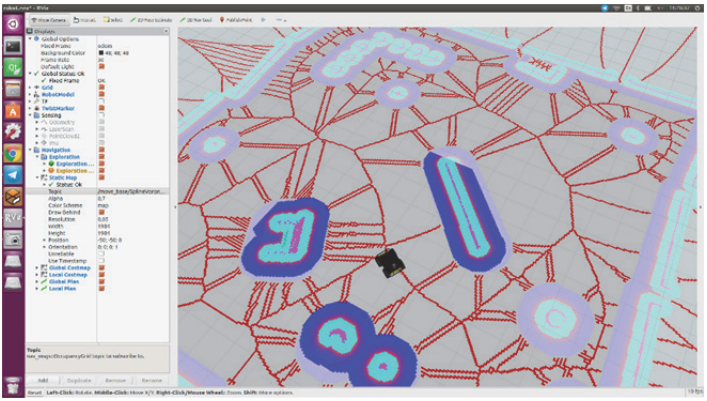

Рис. 9. Пример построенной диаграммы Вороного в симуляторе RViz.

Реализованный и представленный в публикации алгоритм [33] был существенно оптимизирован, что позволило уменьшить среднее время расчета графа Вороного с 8 секунд до 0.5 секунды. Пример результата расчета графа Вороного показан на рисунке 9; расчет и построение диаграммы Вороного в симуляторе RViz заняли менее 1 секунды.

Существенного ускорения расчета графа Вороного удалось добиться благодаря оптимизации используемых алгоритмов и контейнеров. Основным контейнером данных для расчета графа Вороного является динамический массив, каждый элемент которого – очередь индексов ячеек двумерной карты. Так как расстояние до препятствий счетно, в каждой очереди динамического массива хранятся ячейки с одинаковым расстоянием от препятствий. Данный контейнер был оптимизирован таким образом, что он генерируется только в случаях увеличения карты, а непрерывное расположение массива в памяти ускоряет операции с ним. Дополнительно ускорению работы алгоритма способствовало избавление от дублирующих расчетов и избыточных сложных вычислительных операций. Например, в отличие от оригинального подхода, вместо расчета точного расстояния до препятствий хранится квадрат расстояния, что избавляет от многочисленных расчетов корня числа.

После получения с помощью алгоритма Дейкстры пути по графу Вороного, путь преобразовывается в B-сплайн [34] (рисунок 10).

Для хранения сплайна был создан специальный тип данных, хранящий пару переменных типа *boost::math::cubic_b_spline<double>*: параметрические функции y*(t)* и *x(t).* Переменная t изменяется от 0 до 1 включительно и указывает на положение РТС на траектории от точки старта *(t=0)* до целевой точки *(t=1).* Функции *x(t)* и y*(t)* рассчитывают координаты робота (x, y) на карте в зависимости от положения робота на траектории.

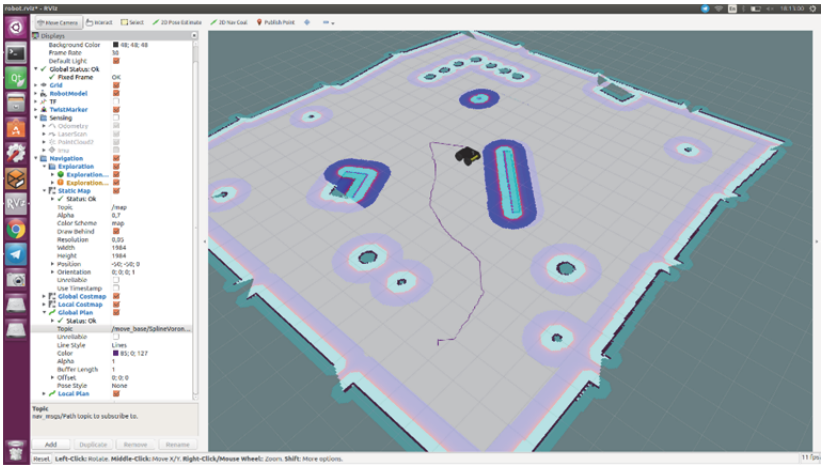

Рис. 10. Путь, рассчитанный по графу Вороного в симуляторе RViz

Чтобы преобразовать найденный путь в B-сплайн, следует задать точность, с которой сплайн будет соответствовать траектории, рассчитываемой по графу Вороного. Для этого путь, представляемый в виде последовательности точек, равномерно делится на N сегментов (первоначальное значение N задано равным 4) и, соответственно, N+1 точек: начальная (старт S), конечная (цель T) и N-1 промежуточных точек. Эти N+1 точек формируют пару сплайнов, задаваемых параметрической функцией с переменной *t*. Полученный сплайн сравнивается с траекторией, состоящей из ребер графа Вороного, и подсчитывается максимальное отклонение между ними. Если отклонение больше или равно половине ширины робота, то сплайн используется для дальнейшего итерационного поиска. В противном случае, переменная N увеличивается на 1, путь делится на N+1 характерных точек, по ним строится сплайн и сравнивается с исходным путем из графа Вороного. Итерационный процесс продолжается до выполнения требования по отклонению.

После расчета пары сплайнов, соответствующих исходному пути в рамках установленной точности, следует итерационный процесс

поиска оптимального сплайн-пути, используя целевую функцию. В начале итерационного процесса подсчитывается значение целевой функции у получившегося сплайна, и он определяется как основной.

N-1 точек промежуточных точек основного сплайна случайным образом перемещаются на небольшое расстояние ɾ→0 и для получаемого нового сплайна подсчитывается значение целевой функции. Если значение целевой функции получаемого нового сплайн-пути меньше, чем у основного сплайна, он сохраняется и определяется как основной. Далее, уже его промежуточные точки случайным образом перемещаются, а новые сплайны сравниваются с ним. Итерационный процесс продолжается до тех пор, пока улучшение результатов расчета стоимости пути (в соответствии с целевой функцией) полученного сплайна не станет меньше выбранного эмпирическим путем порогового значения ε=0.1, а общее количество итераций не превысит 1000 итераций.

Чтобы целевая функция оптимальным образом характеризовала сплайн-путь $q(t)=\{x(t), y(t)\}$, были реализованы три основных критерия оптимальности пути: длина, кривизна и топология. Функция, которая рассчитывает длину пути $L(q)$ аналогично формуле (5), суммирует длины всех сегментов пути. В исходном коде это реализовано с помощью прохода по паре сплайнов с шагом Δ_t, эмпирическим путем установленным $\Delta_t=0.001$. На каждом шаге рассчитывается Δ_x и Δ_y — изменения по осям x и y при изменении параметра t на величину Δ_t. Длина сегмента рассчитывается как корень из суммы их квадратов. Таким образом, длина пути рассчитывается по формуле:

$$L_R(q) = \sum_{t=0,\Delta_t\to 0}^{1-\Delta_t} \sqrt{\left(x(t+\Delta_t)-x(t)\right)^2 + \left(\left(y(t+\Delta_t)-y(t)\right)^2}. \quad (7)$$

Критерий кривизны пути рассчитывается по уравнению (4) с приближением интеграла по аналогии с уравнением (7), только вместо значения функций сплайнов в точках, в формуле используются производные от функций в этих точках:

$$R_R(q) = \sqrt{\sum_{t=0,\Delta_t\to 0}^{1-\Delta_t} \sqrt{\frac{\left(x'(t+\Delta_t)-x'(t)\right)^2 + \left(\left(y'(t+\Delta_t)-y'(t)\right)^2}{\Delta_t^2}}}. \quad (8)$$

Критерий топологии пути, который в том числе определяет безопасность маршрута относительно удаленности РТС от границ препятствий при движении по данному маршруту, изменяется более кардинально по сравнению с оригинальным алгоритмом [17]. Так как при новом подходе препятствия не представляют из себя пересечения

кругов и количество препятствий может быть достаточно велико, формула расчета критерия топологии пути должна быть адаптирована соответствующим образом. Карту и рассчитанный по ней граф Вороного предоставляют большой набор данных в каждой ячейке карты, включая расстояние до ближайшего препятствия, выраженное в размере ячейки OccupancyGrid карты (который по умолчанию составляет 0.05 метра). Расстояние до препятствий используется в функции расчета потенциального поля. В новых условиях описания препятствий окружающей среды функция потенциала отталкивания (1) в каждой точке пути принимает вид:

$$U_R(t) = 1 - \tanh(\alpha * dist(t)), \tag{9}$$

где α — это эмпирически определяемый параметр, dist(t) — минимально расстояние до препятствий в точке t пути. Параметр α отвечает за отталкивание маршрута движения от препятствия. Так как гиперболический тангенс при увеличении аргумента быстро приближается к единице, потенциальное поле быстро ослабевает при отдалении от препятствий. Уменьшение параметра α заставляет робота держаться дальше от препятствий. Эмпирически было получено значение α=0.1.

Так как уравнения сплайнов пути являются непрерывными функциями, а расстояния до препятствий дискретны и берутся из карты занятости, то при прохождении по маршруту следует учитывать длину пути, проходящую через каждую ячейку карты. Таким образом, необходимо просуммировать по всем ячейкам карты, которые пересекает путь, произведение потенциала поля в данной ячейке на длину пути, попадающего внутрь данной ячейки. Если путь пересекает N ячеек, то результирующее значение критерия Топологии этого пути будет равно:

$$U_R(q) = \sum_{k=0}^{N} U_R^k(t) * \delta_q(k), \tag{10}$$

где переменная k обозначает индекс ячейки карты, которую пересекает путь, $\delta_q(k)$ — длина пути, находящегося внутри ячейки k. $U_R^k(t)$ — значение потенциального поля любой точки t пути, которая находится внутри ячейки под номером k.

Итоговая стоимость пути по аналогии с уравнением (6) состоит из суммы трех компонентов:

$$F_R(q) = \gamma_1 T_R(q) + \gamma_2 R_R(q) + \gamma_3 L_R(q). \tag{11}$$

В ходе тестирования алгоритма в среде ROS/Gazebo на роботе Husky [20] эмпирическим образом было установлено, что для достижения паритета критериев, влияние критерия топологии следует существенно увеличить относительно влияния других критериев. После задания веса каждого из критериев в целевой функции: $\gamma_1 = 1$, $\gamma_2 = 1$, $\gamma_3 = 5000$ дальнейшее увеличение веса топологии не изменяет оптимальный путь. Критерий минимального удаления от препятствий регулируется параметром α из функции топологии.

Результаты тестирования алгоритма подтвердили успешность подхода модифицированного сплайн алгоритма. Так, первоначально рассчитанный путь (представлен на рисунке 10) по графу Вороного, который является базой для первого сплайна итеративного алгоритма, является ломаной кривой с множеством резких поворотов. После первых ста итераций (рисунок 11) путь является сплайном, он короче исходного и в нем нет резких изменений пути, для преодоления которых роботу необходимо останавливаться. После 400 итераций (1.3 секунды), итерационный процесс завершился. Рисунок 12 демонстрирует, что в процессе итеративного поиска изгиб траектории в начале пути исчез и длина пути уменьшилась при сохранении безопасного расстояния до препятствий.

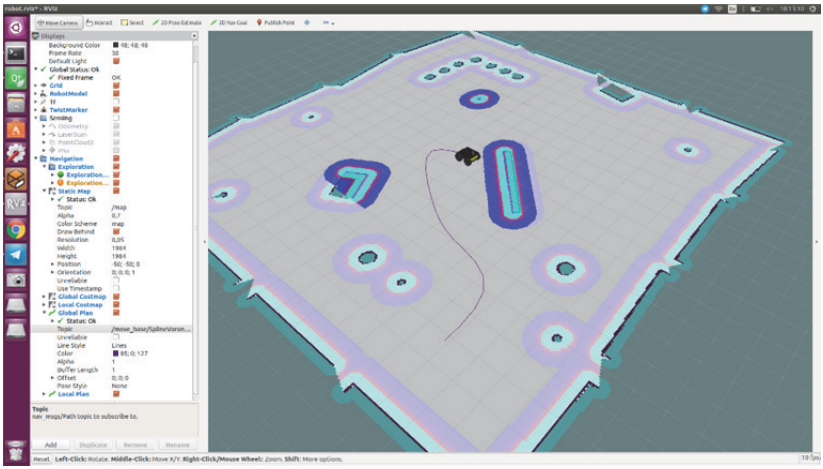

Рис. 11. Маршрут, построенный новым сплайн-алгоритмом за первых 100 итераций. Робот Husky находится на стартовой позиции S

Эксперименты производились на компьютере с процессором Intel Core i7-9700 с 32Гб оперативной памяти. Несколько тысяч симуляционных экспериментов были проведены на двадцати картах

сопоставимых размеров с различными конфигурациями препятствий. Время расчета графа Вороного было стабильным, составляло менее 0.5 секунды и зависело только от сложности карты. Примеры типовой карты показаны на рисунках 8-12, при изменении карты препятствия перемещались и добавлялось от 2 до 10 новых препятствий. Итерационный алгоритм рассчитывал путь в среднем за 2 секунды; время его работы не зависело от конкретной карты, что в первую очередь объясняется сопоставимыми размерами и сложностью использованных в экспериментах карт. Время работы итерационного алгоритма зависело от расстояния между произвольно выбранными стартовой позицией S и целевой позицией T, и сложности участка карты между этими позициями.

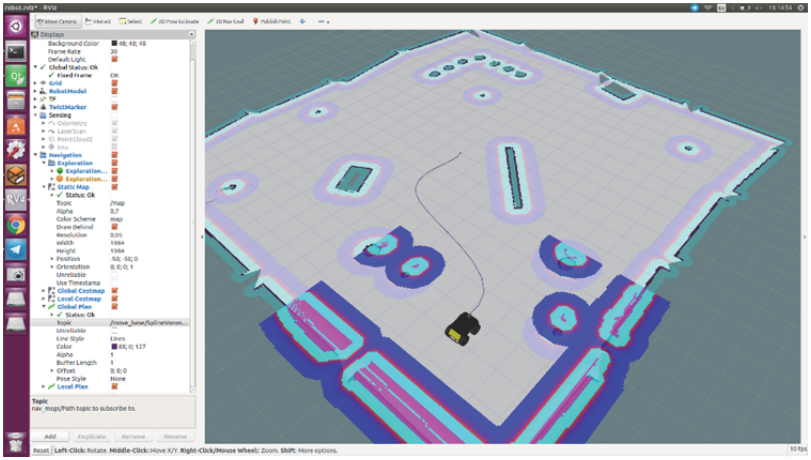

Рис. 12. Маршрут, построенный новым сплайн-алгоритмом за первых 400 итераций. Робот Husky находится на целевой позиции T

Таблица 1 демонстрирует 15 примеров запуска нового алгоритма на одной и той же карте (рисунок 13) с произвольно выбранными позициями S и T. Время первого расчета траектории больше, чем время расчета последующих, из-за необходимости выделения необходимой для расчетов памяти.

В будущем мы планируем оценить увеличение времени работы алгоритма в случае увеличения размеров и сложности экспериментальных карт на несколько порядков. В настоящее время проводится оптимизация структур данных и алгоритмов, и переход к многопоточному подходу для функций расчета критериев и вычисления графа Вороного. Это позволит существенно ускорить работу алгоритма и

исключит остановки робота (2-3 сек.), необходимые для вычисления нового маршрута при задании новой целевой позиции.

Таблица 1. Время расчета в секундах на одной карте: построение графа Вороного, расчет путь по графу Вороного и последующая оптимизация пути

Номер эксперимента	Время расчета (сек.)		
	Граф Вороного	Путь по графу Вороного	Оптимизация пути
1	0.41	0.03	4.63
2	0.35	0.02	1.81
3	0.34	0.03	2.27
4	0.35	0.02	1.62
5	0.34	0.02	2.28
6	0.34	0.02	1.7
7	0.34	0.02	1.22
8	0.34	0.02	0.82
9	0.35	0.02	0.99
10	0.34	0.02	1.02
11	0.35	0.01	1.12
12	0.34	0.03	1.92
13	0.35	0.01	1.49
14	0.34	0.03	1.38
15	0.35	0.02	1.91

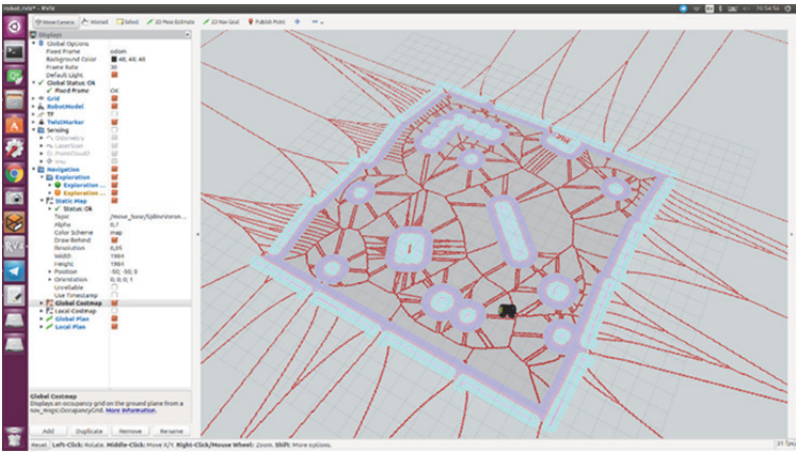

Рис. 13. Карта в симуляторе RViz, использовавшаяся для примеров в таблице 1, и построенная диаграмма Вороного

Также стоит задача интеграции алгоритма в системы управления наземных мобильных РТС лаборатории интеллектуальных

робототехнических систем (ЛИРС) КФУ — гусеничного робота «Сервосила Инженер», колесных «Юниор» и РМВ-2.

Несмотря на то, что алгоритм формально является глобальным и требует наличия статической заранее известной карты, его возможно использовать и в динамически изменяемых условиях, так как он позволяет осуществить перерасчет маршрута движения при локальном изменении карты в режиме реального времени без полного перестроения карты.

8. Заключение. В данной работе описывается модифицированный сплайн-алгоритм планирования пути, а также его реализация для использования в реальных системах управления РТС. В ходе реализации нового алгоритма была проведена адаптация оригинальных формулировок критериев и уравнений, учитывающая реальное представление карт и невозможность аппроксимации РТС геометрической точкой. Полученный в результате многокритериальный сплайн-алгоритм был верифицирован в среде разработки ROS/Gazebo. Данный алгоритм можно использовать для планирования маршрута системой управления любого наземного колесного или гусеничного РТС, оборудованного лазерным дальномером (лидаром). Алгоритм открыт для использования и редактирования, исходный код находится в репозитории Gitlab [35]. При использовании в целевую функцию оптимизации пути могут быть включены новые критерии или изменен вес уже реализованных критериев.

В разработанный алгоритм будут добавлены новые критерии, которые могут быть полезны при проведении поисково-спасательных операций, при работе автономного РТС в опасных и зараженных территориях. Также целевая функция будет модифицирована для учета опасных точек на карте, которые нужно обходить, или, наоборот, ретрансляторов связи, которые нужно держать в поле зрения как можно дольше.

В качестве дальнейшей работы будет проведено сравнение предложенного алгоритма с типовыми алгоритмами поиска пути, включая A* и RRT*, в симуляционной среде ROS\Gazebo по таким критериям как быстродействие и качество полученного пути. Особое внимание будет уделено комбинированным случаям, когда существующие базовые алгоритмы глобального планирования пути будут использованы при создании стартового сплайна для предложенного в данной работе итеративного сплайн-алгоритма.

Литература

1. *Michael N. et al.* Collaborative mapping of an earthquake damaged building via ground and aerial robots // Journal of Field Robotics. 2012. vol. 29. no. 5. pp. 832–841.

2. *Nagatani K. et al.* Multirobot exploration for search and rescue missions: A report on map building in RoboCupRescue 2009 // Journal of Field Robotics. 2011. vol. 28(3). pp. 373–387.
3. *Pshikhopov V., Medvedev M., Krukhmalev V., Shevchenko V.* Base Algorithms of the Direct Adaptive Position-Path Control for Mobile Objects Positioning // Applied Mechanics and Materials. 2015. vol. 763. pp. 110–119.
4. *Loevsky I., Shimshoni I.* Reliable and efficient landmark-based localization for mobile robots // Robotics and Autonomous Systems. 2010. vol. 58(5). pp. 520–528.
5. *Filipenko M., Afanasyev I.* Comparison of Various SLAM Systems for Mobile Robot in an Indoor Environment // International Conference on Intelligent Systems. 2018. 8 p.
6. *Sokolov M., Bulichev O., Afanasyev I.* Analysis of ROS-based Visual and Lidar Odometry for a Teleoperated Crawler-type Robot in indoor environment // International Conference on Informatics in Control, Automation and Robotics (ICINCO). 2017. pp. 316–321.
7. *Wagner G., Choset H.* Subdimensional expansion for multirobot path planning // Artificial Intelligence. 2015. vol. 219. pp. 1–24.
8. *Vokhmintcev A., Yakovlev K.* A Real-Time Algorithm for Mobile Robot Mapping Based on Rotation-Invariant Descriptors and Iterative Close Point Algorithm // International Conference on Analysis of Images, Social Networks and Texts. 2016. pp. 357–369.
9. *Sullivan J.C.W., Pipe A.G.* Path planning for redundant robot manipulators: a global optimization approach using evolutionary search // 1998 IEEE International Conference on Systems, Man, and Cybernetics. 1998. vol. 3. pp. 2396–2400
10. *Hart P.E., Nilsson N.J., Raphael B.*A formal basis for the heuristic determination of minimum cost paths // IEEE transactions on Systems Science and Cybernetics. 1968. vol. 4. no. 2. pp. 100–107.
11. *LaValle S.M.* Rapidly-exploring random trees: A new tool for path planning // Iowa State University Ames. 1998. 4 p.
12. *Яковлев К.С., Макаров Д.А., Баскин Е.С.* Метод автоматического планирования траектории беспилотного летательного аппарата в условиях ограничений на динамику полета // Искусственный интеллект и принятие решений. 2014. № 4. С. 3.
13. *Stentz A.* Optimal and efficient path planning for partially-known environments // International Conference on Robotics and Automation (ICRA). 1994. vol. 94. pp. 3310–3317.
14. *Koenig S., Likhachev M.* Fast replanning for navigation in unknown terrain // IEEE Transactions on Robotics. 2005. vol. 21. no. 3. pp. 354–363.
15. *Karaman S., Frazzoli E.* Sampling-based algorithms for optimal motion planning // The International Journal of Robotics Research. 2011. vol. 30. no. 7. pp. 846–894.
16. *Otte M., Frazzoli E.* RRT X: Real-Time Motion Planning/Replanning for Environments with Unpredictable Obstacles // Algorithmic Foundations of Robotics XI. 2015. pp. 461–478.
17. *Magid E., Keren D., Rivlin E., Yavneh I.* Spline-based robot navigation // 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2006. pp. 2296–2301.
18. *Nagatani K., Iwai Y., Tanaka Y.* Sensor based navigation for carlike mobile robots using Generalized Voronoi Graph // 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2001. vol. 2. pp. 1017–1022.
19. *Lavrenov R., Matsuno F., Magid E.* Modified Spline-Based Navigation: Guaranteed Safety for Obstacle Avoidance // International Conference on Interactive Collaborative Robotics. 2017. pp. 123–133.
20. *Koren Y., Borenstein J.* Potential field methods and their inherent limitations for mobile robot navigation // 1991 IEEE International Conference on Robotics and Automation. 1991. pp. 1398–1404.

21. *Khatib O.* Real-Time Obstacle Avoidance for Manipulators and Mobile Robots // Autonomous robot vehicles. 1986. pp. 396–404.
22. *Ge S.S., Cui Y.J.* Dynamic motion planning for mobile robots using potential field method // Autonomous robots. 2002. vol. 13. no. 3. pp. 207–222.
23. *Lagarias J.C., Reeds J.A., Wright M.H., Wright P.E.* Convergence properties of the Nelder--Mead simplex method in low dimensions // SIAM Journal on optimization. 1998. vol. 9(1). pp. 112–147.
24. *Лавренов Р.О., Афанасьев И.М., Магид Е.А.* Планирование маршрута для беспилотного наземного робота с учетом множества критериев оптимизации // Третий Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта. 2016. С. 10–20.
25. *Lavrenov R., Magid E.* Towards heterogeneous robot team path planning: acquisition of multiple routes with a modified spline-based algorithm // MATEC Web of Conferences. 2017. vol. 113. pp. 02015.
26. ROS программная библиотека робота Husky. URL: http://wiki.ros.org/Robots/Husky (дата обращения: 11.10.2018).
27. *Zheng K*. ROS Navigation Tuning Guide // arXiv preprint arXiv: 1706.09068. 2017.
28. *Akai N. et al.* Autonomous driving based on accurate localization using multilayer LiDAR and dead reckoning // Intelligent Transportation Systems (ITSC). 2017. pp. 1–6.
29. *Thrun S., Bücken A*. Integrating grid-based and topological maps for mobile robot navigation // Proceedings of the National Conference on Artificial Intelligence. 1996. pp. 944–951.
30. *Lau B., Sprunk C., Burgard W.* Improved updating of Euclidean distance maps and Voronoi diagrams // 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2010. pp. 281–286.
31. *Lau B., Sprunk C., Burgard W.* Efficient grid-based spatial representations for robot navigation in dynamic environments // Robotics and Autonomous Systems. 2013. vol. 61. no. 10. pp. 1116–1130.
32. *Lau B., Sprunk C., Burgard W.* Incremental Updates of Configuration Space Representations for Non-Circular Mobile Robots with 2D 2.5 D or 3D Obstacle Models // ECMR. 2011. pp. 49–54.
33. *Федоренко Р.В., Гуренко Б.В.* Планирование траектории автономного мини-корабля // Инженерный вестник Дона. 2015. Т. 38. № 4-1.12 с.
34. *Soldea O., Elber G., Rivlin E.* Global curvature analysis and segmentation of volumetric data sets using trivariate B-spline functions // Geometric Modeling and Processing. 2004. pp. 217–226.
35. Реализация сплайн-алгоритма планирования пути. URL: https://gitlab.com/LIRS_Projects/Simulaion-Spline-Voronoi-Planner (дата обращения: 04.10.2018).

Лавренов Роман Олегович — младший научный сотрудник лаборатории интеллектуальных робототехнических систем, ассистент кафедры интеллектуальной робототехники высшей школы информационных технологий и интеллектуальных систем, Казанский (Приволжский) федеральный университет (КФУ). Область научных интересов: алгоритмы планирования пути; алгоритмы локализации и картографирования для автономных робототехнических систем; промышленные роботы-манипуляторы. Число научных публикаций — 24. lavrenov@it.kfu.ru, https://kpfu.ru/lirs.html; ул. Кремлевская, 35, Казань, 420008; р.т.: +7(843) 221-3433.

Магид Евгений Аркадьевич — Ph.D., профессор, руководитель лаборатории интеллектуальных робототехнических систем, и.о. заведующего кафедры интеллектуальной робототехники высшей школы информационных технологий и

интеллектуальных систем, Казанский (Приволжский) федеральный университет (КФУ). Область научных интересов: поисково-спасательная робототехника; мобильная робототехника; групповое взаимодействие роботов; планирование маршрута; антропоморфная робототехника; взаимодействие человека и робота; обработка изображений и техническое зрение. Число научных публикаций — 110. magid@it.kfu.ru, https://kpfu.ru/lirs.html; ул. Кремлевская, 35, Казань, 420008; р.т.: +7(843) 221-3433.

Мацуно Фумитоши — Ph.D., профессор департамента инженерной механики и науки, Киотский Университет, Япония. Область научных интересов: робототехника; системы управления; поисково-спасательная робототехника. Число научных публикаций – 486. matsuno@me.kyoto-u.ac.jp, ул. Киёдайкацура, 3., Нисикё, Киото, 615-8540, Япония, р.т.: +81-75-383-3593.

Свинин Михаил Михайлович — Ph.D., профессор колледжа информационных наук и инженерии, Университет Рицумейкан. Область научных интересов: компьютерные науки, математическое моделирование, математические методы в исследованиях. Число научных публикаций — 154. svinin@fc.ritsumei.ac.jp, ул. Нойи Хигаши, 1, Кусацу, Киото, 525-8577, Япония, р.т.: +077-561-3099.

Сутакорн Джакрит — Ph.D., профессор департамента биомедицинской инженерии, Университет Махидол. Область научных интересов: медицинская робототехника; поисково-спасательная робототехника; экзоскелеты; мобильная робототехника. Число научных публикаций — 104. jackrit.sut@mahidol.ac.th, ул. 999 Футтамонтон, 4, Салайа, 73170, Тайланд, р.т.: +66 81 913 6220.

Поддержка исследований. Исследование выполнено при финансовой поддержке РФФИ (проект № 19-58-70002).

DOI 10.15622/sp.18.1.57-84

R.O. LAVRENOV, E.A. MAGID, F. MATSUNO, M.M. SVININ, J. SUTHAKORN

DEVELOPMENT AND IMPLEMENTATION OF SPLINE-BASED PATH PLANNING ALGORITHM IN ROS/GAZEBO ENVIRONMENT

Lavrenov R.O., Magid E.A., Matsuno F., Svinin M.M., Suthakorn J. **Development and Implementation of Spline-based Path Planning Algorithm in ROS/Gazebo Environment.**

Abstract. Path planning for autonomous mobile robots is an important task within robotics field. It is common to use one of the two classical approaches in path planning: a global approach when an entire map of a working environment is available for a robot or local methods, which require the robot to detect obstacles with a variety of onboard sensors as the robot traverses the environment.

In our previous work, a multi-criteria spline algorithm prototype for a global path construction was developed and tested in Matlab environment. The algorithm used the Voronoi graph for computing an initial path that serves as a starting point of the iterative method. This approach allowed finding a path in all map configurations whenever the path existed. During the iterative search, a cost function with a number of different criteria and associated weights was guiding further path optimization. A potential field method was used to implement some of the criteria.

This paper describes an implementation of a modified spline-based algorithm that could be used with real autonomous mobile robots. Equations of the characteristic criteria of a path optimality were further modified. The obstacle map was previously presented as intersections of a finite number of circles with various radii. However, in real world environments, obstacles' data is a dynamically changing probability map that could be based on an occupancy grid. Moreover, the robot is no longer a geometric point.

To implement the spline algorithm and further use it with real robots, the source code of the Matlab environment prototype was transferred into C++ programming language. The testing of the method and the multi criteria cost function optimality was carried out in ROS/Gazebo environment, which recently has become a standard for programming and modeling robotic devices and algorithms.

The resulting spline-based path planning algorithm could be used on any real robot, which is equipped with a laser rangefinder. The algorithm operates in real time and the influence of the objective function criteria parameters are available for dynamic tuning during a robot motion.

Keywords: path planning, mobile robot, planning algorithm, ROS, Gazebo, spline-based algorithm.

Lavrenov Roman Olegovich — Junior Researcher of Intelligent Robotic Systems Laboratory, lecturer of Intelligent Robotics Department of High School of Information Technology and Intelligent Systems, Kazan Federal University. Research interests: path planning and localization and mapping algorithms for autonomous robots; industrial robot manipulators. The number of publications — 24. lavrenov@it.kfu.ru, https://kpfu.ru/lirs.html; 35, Kremlevskaya str., Kazan, 420008, Russia; office phone: +7(843) 221-3433.

Magid Evgeni Arkadievich — Ph.D., Professor, Head of Laboratory of Intelligent Robotic Systems, Head of Department of Intelligent Robotics of High School of Information Technology and Intelligent Systems, Kazan Federal University. Research interests: mobile robotics, group interaction of robots, path planning, anthropomorphic robotics, human-robot interaction, image processing and computer vision. The number of publications — 110. magid@it.kfu.ru, https://kpfu.ru/lirs.html; 35, Kremlevskaya st., Kazan, 420008, Russia; office phone: +7(843) 221-3433.

Matsuno Fumitoshi — Ph.D., Full Professor of Mechanical Engineering and Science Department, Kyoto University. Research interests: robotics, control engineering, rescue robotics. The number of publications – 486. matsuno@me.kyoto-u.ac.jp, 3, Kyoudaikatsura, Nishigyo-ku, Kyoto, 615-8540, Japan; office phone: +81-75-383-3593.

Svinin Mikhail Mikhailovich — Ph.D., Professor of College of Information Science and Engineering, Ritsumeikan University. Research interests: computer science, mathematical modeling and mathematical methods in scientific research. The number of publications – 154. svinin@fc.ritsumei.ac.jp, 1, Noji Higashi, Kusatsu, Kyoto, 525-8577, Japan; office phone: +077-561-3099

Suthakorn Jackrit — Ph.D., Professor of Biomedical Engineering Department, Mahidol University. Research interests: medical robotics, rescue robotics, surgical navigation, exoskeleton, mobile robotics. The number of publications — 104. jackrit.sut@mahidol.ac.th; 4, 999 Phuttamonthon, Salaya, 73170, Thailand; office phone: +66 81 913 6220.

Acknowledgements. This research is supported by RFBR (project No. 19-58-70002).

References

1. Michael N. et al. Collaborative mapping of an earthquake damaged building via ground and aerial robots. *Journal of Field Robotics.* 2012. vol. 29. no. 5. pp. 832–841.
2. Nagatani K. et al. Multirobot exploration for search and rescue missions: A report on map building in RoboCupRescue 2009. *Journal of Field Robotics*. 2011. vol. 28(3). pp. 373–387.
3. Pshikhopov V., Medvedev M., Krukhmalev V., Shevchenko V. Base Algorithms of the Direct Adaptive Position-Path Control for Mobile Objects Positioning. *Applied Mechanics and Materials*. 2015. vol. 763. pp. 110–119.
4. Loevsky I., Shimshoni I. Reliable and efficient landmark-based localization for mobile robots. *Robotics and Autonomous Systems.* 2010. vol. 58(5). pp. 520–528.
5. Filipenko M., Afanasyev I. Comparison of Various SLAM Systems for Mobile Robot in an Indoor Environment. International Conference on Intelligent Systems. 2018. 8 p.
6. Sokolov M., Bulichev O., Afanasyev I. Analysis of ROS-based Visual and Lidar Odometry for a Teleoperated Crawler-type Robot in indoor environment. International Conference on Informatics in Control, Automation and Robotics (ICINCO). 2017. pp. 316–321.
7. Wagner G., Choset H. Subdimensional expansion for multirobot path planning. *Artificial Intelligence*. 2015. vol. 219. pp. 1–24.
8. Vokhmintcev A., Yakovlev K. A Real-Time Algorithm for Mobile Robot Mapping Based on Rotation-Invariant Descriptors and Iterative Close Point Algorithm. International Conference on Analysis of Images, Social Networks and Texts. 2016. pp. 357–369.
9. Sullivan J.C.W., Pipe A.G. Path planning for redundant robot manipulators: a global optimization approach using evolutionary search. 1998 IEEE International Conference on Systems, Man, and Cybernetics. 1998. vol. 3. pp. 2396–2400
10. Hart P.E., Nilsson N.J., Raphael B.A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*. 1968. vol. 4. no. 2. pp. 100–107.
11. LaValle S.M. Rapidly-exploring random trees: A new tool for path planning. Iowa State University Ames. 1998. 4 p.
12. Yakovlev K.S., Makarov D.A., Baskin E.S. [The method of automatic planning of the trajectory of an unmanned aerial vehicle under restrictions on flight dynamics].

Iskusstvennyj intellekt i prinyatie reshenij – Artificial intelligence and decision making 2014. vol. 4. pp. 3. (In Russ.).
13. Stentz A. Optimal and efficient path planning for partially-known environments. International Conference on Robotics and Automation (ICRA). 1994. vol. 94. pp. 3310–3317.
14. Koenig S., Likhachev M. Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics*. 2005. vol. 21. no. 3. pp. 354–363.
15. Karaman S., Frazzoli E. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*. 2011. vol. 30. no. 7. pp. 846–894.
16. Otte M., Frazzoli E. RRT X: Real-Time Motion Planning/Replanning for Environments with Unpredictable Obstacles. Algorithmic Foundations of Robotics XI. 2015. pp. 461–478.
17. Magid E., Keren D., Rivlin E., Yavneh I. Spline-based robot navigation. 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2006. pp. 2296–2301.
18. Nagatani K., Iwai Y., Tanaka Y. Sensor based navigation for carlike mobile robots using Generalized Voronoi Graph. 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2001. vol. 2. pp. 1017–1022.
19. Lavrenov R., Matsuno F., Magid E. Modified Spline-Based Navigation: Guaranteed Safety for Obstacle Avoidance. International Conference on Interactive Collaborative Robotics. 2017. pp. 123–133.
20. Koren Y., Borenstein J. Potential field methods and their inherent limitations for mobile robot navigation. 1991 IEEE International Conference on Robotics and Automation. 1991. pp. 1398–1404.
21. Khatib O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. Autonomous robot vehicles. 1986. pp. 396–404.
22. Ge S.S., Cui Y.J. Dynamic motion planning for mobile robots using potential field method. *Autonomous robots*. 2002. vol. 13. no. 3. pp. 207–222.
23. Lagarias J.C., Reeds J.A., Wright M.H., Wright P.E. Convergence properties of the Nelder--Mead simplex method in low dimensions. *SIAM Journal on optimization.* 1998. vol. 9(1). pp. 112–147.
24. Lavrenov R.O., Afanasyev I.M., Magid E.A. [Path planning for an unmanned ground robot based on a variety of optimization criteria]. *Tretij Vserossijskij nauchno-prakticheskij seminar "Bespilotnye transportnye sredstva s ehlementami iskusstvennogo intellekta"* [The third All-Russian Scientific and Practical Seminar "Unpiloted Vehicles with Artificial Intelligence Elements"]. 2016. pp. 10–20. (In Russ.).
25. Lavrenov R., Magid E. Towards heterogeneous robot team path planning: acquisition of multiple routes with a modified spline-based algorithm. *MATEC Web of Conferences*. 2017. vol. 113. pp. 02015.
26. ROS programmnaya biblioteka robota Husky [ROS program library of Husky robot]. Available at: http://wiki.ros.org/Robots/Husky. (accessed: 11.10.2018). (In Russ.).
27. Zheng K. ROS Navigation Tuning Guide. arXiv preprint arXiv: 1706.09068. 2017.
28. Akai N. et al. Autonomous driving based on accurate localization using multilayer LiDAR and dead reckoning. *Intelligent Transportation Systems (ITSC).* 2017. pp. 1–6.
29. Thrun S., Bücken A. Integrating grid-based and topological maps for mobile robot navigation. Proceedings of the National Conference on Artificial Intelligence. 1996. pp. 944–951.
30. Lau B., Sprunk C., Burgard W. Improved updating of Euclidean distance maps and Voronoi diagrams. 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2010. pp. 281–286.
31. Lau B., Sprunk C., Burgard W. Efficient grid-based spatial representations for robot navigation in dynamic environments. *Robotics and Autonomous Systems*. 2013. vol. 61. no. 10. pp. 1116–1130.

32. Lau B., Sprunk C., Burgard W. Incremental Updates of Configuration Space Representations for Non-Circular Mobile Robots with 2D 2.5 D or 3D Obstacle Models. ECMR. 2011. pp. 49–54.
33. Fedorenko R.V., Gurenko B.V. [Planning the trajectory of an autonomous mini-ship]. *Inzhenernyj vestnik Dona – Engineering Journal of Don.* 2015. vol. 38. no. 4-1. 12 p. (In Russ.).
34. Soldea O., Elber G., Rivlin E. Global curvature analysis and segmentation of volumetric data sets using trivariate B-spline functions. Geometric Modeling and Processing. 2004. pp. 217–226.
35. Realizaciya splajn-algoritma planirovaniya puti [Implementation of the spline algorithm for path planning]. Available at: https://gitlab.com/LIRS_Projects/Simulaion-Spline-Voronoi-Planner (accessed: 04.10.2018).