Алгоритм планирования и согласования совокупности траекторий для группы интеллектуальных агентов*

А.А. Андрейчук

Российский университет дружбы народов, г. Москва, Россия

Аннотация. В работе рассматривается задача планирования совокупности неконфликтных траекторий для группы интеллектуальных агентов. Предлагаемый алгоритм использует приоритизированный подход и работает в два этапа: независимое планирование индивидуальных траекторий агентов и согласование траекторий. Предлагается оригинальный метод согласования траекторий, который работает исключительно за счет добавления временных задержек. Благодаря представлению временной оси в виде последовательности интервалов, а не дискретных моментов времени, алгоритм способен согласовывать траектории, построенные с возможностью перемещения в произвольном направлении. Проведенные модельные экспериментальные исследования продемонстрировали применимость предлагаемого алгоритма и его высокую вычислительную эффективность.

Ключевые слова: планирование траектории, многоагентные системы, устранение конфликтов, A*, Theta*, MT-граф.

DOI 10.14357/20718594180407

Введение

Задача планирования совокупности неконфликтных траекторий для группы агентов является актуальной проблемой, решения которой востребованы в различных современных автоматизированных системах управления, которые так или иначе связаны с перемещениями объектов [1]. К ним относятся различные интеллектуальные системы, используемые в логистике [2], транспортных системах [3], сельском хозяйстве [4], при ликвидации ЧП [5] и т.д. Для решения задач, связанных с навигацией множества агентов, применяются различные подходы, которые условно можно разделить на реактивные и делиберативные. В случае применения реактивных подходов [6], планирование как таковое не осуществляется, а решения о направлениях и скоростях движения агентов принимаются непосредственно во время исполнения. Делиберативные подходы характеризуются тем, что они строят план перемещений агентов, моделируя возможные ситуации, до того как агенты начнут движение. В данной работе рассматриваются именно делиберативные подходы. Как правило, для моделирования пространства поиска используют граф специального вида, вершины которого соответствуют некоторым областям пространства, где могут находиться агенты, а в качестве ребер выступают элементарные траектории – отрезки, по которым агенты могут перемещаться. Моделирование пространства поиска подобным образом позволяет использовать графовые методы планирования. При планировании траектории для одного агента наиболее часто применяется алгоритм A* [7] и его различные модификации, такие как Theta* [8], LIAN [9]. Однако его применение для задач многоагентного планирования имеет слишком высокую вычислительную сложность.

* Работа выполнена при поддержке программы «РУДН 5-100», гранта РФФИ № 17-07-00281 и специальной программы президиума РАН № 0063-2018-0002.

✉ *Андрейчук Антон Андреевич. E-mail: andreychuk@mail.ru*

Если рассматривать комбинированное пространство поиска, то из начального положения возможен переход в k^n различных состояний.

Существуют алгоритмы, которые строят комбинированное пространство поиска, но при этом используют дополнительные методы и процедуры, позволяющие сократить перебор. К алгоритмам этой группы относятся OD+ID [10], CBS [11], M* [12] и др. Благодаря построению комбинированного пространства поиска эти алгоритмы сохраняют такие важные теоретические свойства, как полнота и оптимальность, однако скорость их работы довольно низкая. Под полнотой алгоритма подразумевается гарантия нахождения решения при условии его существования, а под оптимальностью – гарантия нахождения лучшего решения из всех существующих по какому-либо рассматриваемому критерию качества.

Существуют также алгоритмы, которые не строят сложное комбинированное пространство поиска. Вместо этого траектория для каждого агента строится по отдельности, за счет чего данный подход работает значительно быстрее. Независимое планирование траекторий приводит к тому, что между траекториями возникают конфликты, которые необходимо устранять. Как правило, для этого используются специальные процедуры локального перепланирования, т.е. модификации уже построенной траектории с целью устранения конфликта [13]. Однако устранение конфликта путем локального перепланирования приводит к тому, что алгоритмы не обладают свойством оптимальности. Более того, устранение конфликта не всегда возможно, в результате чего в общем случае отсутствует и свойство полноты. К алгоритмам, использующим этот подход, относятся WHCA* [14], MAPP [15] и др.

В работе [16] был предложен приоритизированный подход, суть которого заключается в следующем: всем агентам присваивается приоритет от 1 до n, где n – число агентов в задании. Один за другим агенты планируют свои траектории в порядке приоритета. Каждый следующий агент должен спланировать свою траекторию так, чтобы она не конфликтовала с траекториями всех агентов с более высоким приоритетом. То есть, если агент имеет приоритет k, то он не должен конфликтовать с агентами, имеющими приоритет от 1 до k-1. После того как траектория была согласована с траекториями агентов, имеющих более высокий приоритет, она фиксируется и больше не изменяется. При этом устранение возможных конфликтов с агентами, имеющими приоритет от k+1 до n, будет являться уже их задачей, так как они имеют более низкий приоритет.

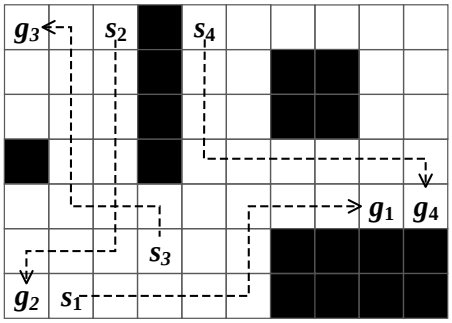


Рис. 1. Пример задания, обладающего свойствами правильно-сформированной инфраструктуры

s_i соответствуют стартовым положениям агентов, g_i – целевым, черные клетки являются непроходимыми (препятствиями)

Алгоритмы, использующие этот подход, также не обладают свойствами оптимальности и полноты, однако в работе [17] были сформулированы условия, при соблюдении которых они могут гарантировать нахождение решения. Условия правильно-сформированной инфраструктуры (от англ. well-formed infrastructures) сводятся к тому, что для каждого агента должен существовать путь, который не проходит через стартовые или целевые положения других агентов. На практике эти условия часто соблюдаются, например, в задачах связанных с логистикой. В этой работе рассматривается именно такой класс задач, где приоритизированные алгоритмы обладают свойством полноты. Пример задания, обладающего свойствами правильно-сформированной инфраструктуры, приведен на Рис. 1.

На Рис. 1 также изображены траектории, построенные алгоритмом A*, рассматривающим возможность перемещения только в четырех направлениях. Подобного ограничения придерживается большинство существующих алгоритмов многоагентного планирования. Такая упрощенная модель движения позволяет разбить время на дискретные шаги (timesteps), за каждый из которых агент может либо перейти в смежную вершину, либо остаться на месте и ждать. Благодаря этому конфликты между агентами возникают только в дискретные мо-

менты времени и их всегда можно однозначно отнести либо к вершине, либо к ребру графа.

При этом существуют алгоритмы планирования траектории для одного агента, которые позволяют планировать траектории с возможностью перемещения в произвольном направлении (в англ. any-angle pathfinding). К ним относятся такие алгоритмы как Theta* [8], Anya [18] и др. Преимуществом этих алгоритмов являются более короткие траектории с меньшим числом смен направления движения. Пример траектории, построенной с возможностью перемещения в произвольном направлении, представлен на Рис. 2.

Однако применение алгоритмов, планирующих путь с возможностью перемещения в произвольном направлении, для многагентного планирования является нетривиальной задачей по нескольким причинам. Во-первых, при движении в произвольном направлении конфликт может возникнуть в любом месте пространства и не всегда можно однозначно отнести это место к конкретной вершине или ребру графа. Во-вторых, если агенты движутся в произвольном направлении, то конфликты между ними могут возникнуть в любой произвольный момент времени, из-за чего возникает необходимость решать задачу в непрерывном времени (Рис. 3).

В работе [19] был предложен алгоритм AA-SIPP (Any-Angle SIPP), который является модификацией алгоритма SIPP (Safe Interval Path Planning)[20], предназначенный для планирования траектории одного агента в среде с динамическими препятствиями. Отличительной особенностью этого алгоритма является использование интервалов, а не дискретных моментов времени, для представления временной оси. Интервалы могут иметь произвольную продолжительность, и привязаны к определенным положениям агента в пространстве. Использование интервалов позволило реализовать алгоритм AA-SIPP, который планирует траекторию одного агента с возможностью перемещения в произвольном направлении в среде с динамическими препятствиями. На основе AA-SIPP был предложен алгоритм многоагентного планирования AA-SIPP(m).

Предлагаемый алгоритм согласования траекторий также использует парадигму безопасных интервалов и поэтому может согласовывать тра-

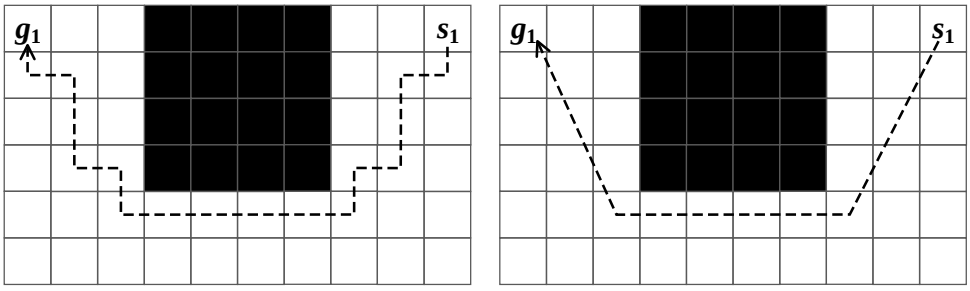


Рис. 2. Слева – путь, построенный алгоритмом A* с возможностью перемещения только в четырех направлениях, справа – путь, построенный алгоритмом Theta*

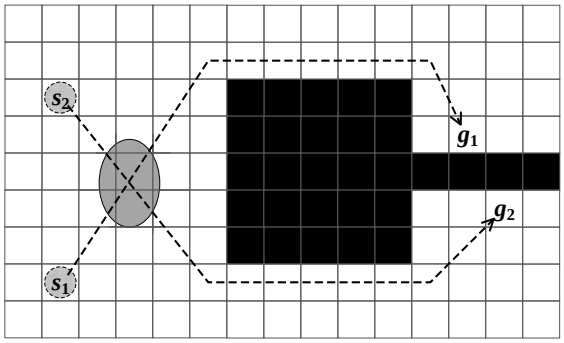


Рис. 3. Пример конфликтной зоны между двумя агентами, траектории которых построены с возможностью перемещения в произвольном направлении

С учетом размеров тел агентов, конфликт между ними возможен в любой точке на участках секций, находящихся внутри конфликтной зоны

ектории, построенные с возможностью перемещения в произвольном направлении. В качестве такого алгоритма предлагается использовать Theta*. В отличие от существующих алгоритмов, использующих приоритизированный подход, таких как, например, AA-SIPP(m), предлагаемый алгоритм работает в два этапа. На первом происходит независимое планирование траекторий, т.е. без учета траекторий других агентов. Вторым этапом является процесс их согласования. Разработанная процедура согласования траекторий устраняет конфликты исключительно за счет добавления временных задержек и не модифицирует сами пути. За счет того, что планирование траекторий происходит независимо, а их согласование осуществляется уже после того как они были построены, достигается более высокая эффективность работы алгоритма. Как показали результаты проведенных экспериментальных исследований, предлагаемый подход обладает высокой вычислительной эффективностью, однако при большом числе агентов качество решений может быть хуже в сравнении с существующими аналогами.

1. Постановка задачи

В задачах планирования траектории как для одного агента, так и для группы агентов, для моделирования пространства поиска применяются различные типы графов, однако наиболее часто используются так называемые графы регулярной декомпозиции, они же метрические-топологические графы (МТ-графы), отличающиеся простотой и при этом достаточной информативностью для решения задач планирования траектории [21]. МТ-граф представляет собой взвешенный неориентированный граф, каждый элемент которого соответствует некоторой области пространства, а в качестве ребер выступают пары смежных проходимых вершин. Вершина МТ-графа считается непроходимой (заблокированной), если в соответствующей ей области пространства находится какое-либо препятствие.

Формально МТ-граф (MT-Gr) можно представить в следующем виде:

$$\text{MT–Gr}= \langle A, ADJ, dist\rangle,$$

где A – множество вершин, составляющих матрицу $A_{m\times n} = \{a_{ij}\}: a_{ij} = 0 \vee a_{ij} = 1, \forall i,j: 0 \le i < m, 0 \le j < n,\ m,n \in N\backslash\{0\}$; Если $a_{ij} = 0$, то вершина является проходимой, в противном случае – непроходимой. В тех случаях, когда координаты не важны, будем обозначать вершины буквами латинского алфавита: *a*, *b*, *c* и т.д.;

$ADJ \subseteq A\times A$ – множество всех смежных пар вершин. Вершины a_{ij} и a_{lk} являются смежными, если $|i-l|\le 1 \vee |j-k|\le 1$. Фактически – смежным вершинам соответствуют смежные элементы матрицы $A_{m\times n}$.;

dist: $A\times A \rightarrow \Re$ – метрика на множестве *A* (функция, задающая способ определения расстояния между вершинами). В качестве такой функции будем использовать Евклидову метрику:

$$dist(a_{ij}, a_{lk}) = \sqrt{(i-l)^2 + (j-k)^2}.$$

Агенты моделируются открытым диском радиуса *r* и могут двигаться с постоянной скоростью *v*. Упорядоченную пару вершин будем называть секцией и обозначать $e=\langle a, b\rangle$. Длиной секции является величина $len(e) = len\langle a, b\rangle = dist(a, b)$. Время, которое требуется агенту для того, чтобы пройти от *a* к *b,* будем обозначать как $dur\langle a, b\rangle = dist(a, b)v$.

Каждую траекторию π можно представить как пару <путь, расписание>: $\pi_i = \langle\{s_i, a_1, a_2, \dots, a_k, g_i\}|\{t_0, t_1, t_2, \dots, t_k\}\rangle$. Путь представляет собой последовательность исполнимых секций, концы которых находятся в центрах вершин МТ-графа, начинается в стартовом положении и заканчивается в целевом. Секция является исполнимой в том случае, когда она имеет, по крайней мере, зазор *r* от непроходимых вершин, т. е. агент, следующий по данной секции, гарантировано не столкнется со статическими препятствиями. Расписание – это последовательность временных меток, указывающих в какой момент времени агент должен начать движение вдоль той или иной секции, составляющей путь. Разница во времени, указанном в расписании и фактическом моменте достижения агентом секции, означает, что агент должен подождать в начале секции, прежде чем начать движение: $wait(a_i) = t_i - (t_{i-1} + dur\langle a_i, a_{i-1}\rangle), i \in [1,k]$; $wait(s) = \mathrm{t}_0$.

Для оценки качества решения введем понятие его стоимости. Без ограничения общности, будем считать, что единицы измерения времени и расстояния эквивалентны, т.е. скорость *v* такова, что за одну единицу времени агенты преодолевают расстояние между двумя ортогонально-смежными вершинами. Стоимостью траектории

является суммарное время, требуемое на преодоление всех секций, составляющих путь, а также продолжительность всех ожиданий, которые необходимо совершить агенту, чтобы достичь цели без столкновений: $cost(\pi) = \sum_{i=0}^{k} dur\langle a_i, a_{i+1}\rangle + \sum_{i=0}^{k} wait(a_i)$. Общая стоимость решения (в англ. flowtime) для n агентов равна сумме стоимостей их индивидуальных решений: $cost_{flowtime}(solution) = \sum_{i=1}^{n} cost(\pi_i)$. Также существует альтернативная оценка стоимости решения (в англ. makespan), которая равна максимальному времени, которое требуется одному из агентов для того, чтобы достичь своего целевого положения: $cost_{makespan}(solution) = max(cost(\pi_i)), i = \overline{1,n}$. Таким образом, целью предлагаемого алгоритма является поиск решения, т.е. совокупности неконфликтных траекторий, стоимость которого нужно минимизировать.

Пусть заданы n дискообразных агентов радиуса r, которые имеют постоянную скорость v, перемещающиеся в двухмерном пространстве, моделируемом с помощью MT-графа. Для каждого агента i задано стартовое s_i и целевое g_i положение, соответствующее центру некоторой проходимой вершины MT-графа. Задачей каждого агента является достижение целевого положения, избегая при этом статических препятствий и других агентов.

Решением рассматриваемой задачи является совокупность траекторий $\{\pi_1, \dots, \pi_n\}$ таких, что любая пара π_i, π_j является неконфликтной. Траектории π_i, π_j являются неконфликтными, если в любой момент времени t расстояние между центрами агентов i и j не меньше чем сумма их радиусов: $\forall t: dist\left(\pi_i(t), \pi_j(t)\right) \geq r_i + r_j$, где $\pi_i(t)$ – положение агента i при движении по траектории π_i в момент времени t.

В данной работе рассматривается класс задач, обладающих свойством правильно-сформированной инфраструктуры. Для выполнения этого свойства стартовые и целевые положения агентов должны располагаться таким образом, что ни одно из них не блокирует пути других агентов. Иными словами, для каждого агента существует путь из его стартового положения в целевое, который не включает в себя стартовые или целевые положения других агентов: $\forall i \in [1;n]: \exists \pi_i: \pi_i \cap (s_j \cup g_j) = \emptyset, \forall j \in [1;n], j \neq i$. Если условие выполнено, то предлагаемый алгоритм гарантирует нахождение решения, при условии, что оно существует.

2. Алгоритм согласования траекторий

Предлагаемый алгоритм работает в два этапа. На первом происходит планирование индивидуальных траекторий агентов. Для этого может быть использован любой известный алгоритм планирования, применимый к поставленной задаче, например, A* [7], Theta* [8], LIAN [9] и др. Вторым этапом является процесс их согласования. Для этого используется принцип проритизированного подхода, т.е. каждому из агентов назначается приоритет, после чего они по очереди согласуют свои траектории с траекториями агентов, имеющих более высокий приоритет. В зависимости от выбранной последовательности приоритетов можно получить решения разной стоимости, однако в данной работе этот аспект не рассматривается и считается, что последовательность приоритетов агентов уже задана. Также предлагаемый метод согласования траекторий использует представление временной оси в виде последовательности безопасных и конфликтных интервалов произвольной величины, за счет чего достигается возможность согласовывать траектории, построенные с возможностью перемещения в произвольном направлении.

2.1. Модификация индивидуальных пространств поиска агентов

Прежде чем приступать к согласованию траекторий, необходимо их спланировать. Несмотря на то, что для планирования индивидуальных траекторий возможно использование любого алгоритма планирования траектории, применимого к этой задаче, для того, чтобы гарантировать нахождение решения, необходимо модифицировать индивидуальные пространства поиска агентов. Модификация заключается в блокировании вершин MT-графа, соответствующих стартовым и целевым положениям других агентов.

Блокирование стартовых положений агентов необходимо для соблюдения условий правильно-сформированной структуры, т.к. для гарантии нахождения решения у них должна быть возможность находиться в стартовом положении неограниченное количество времени [20]. Блокирование целевых положений необязательно для соблюдения условий правильно-сформированной структуры, однако требуется предлагаемому алгоритму согласования. Из-за

того, что устранение конфликтов происходит только за счет изменения расписания, в случае, если возникнет конфликт с агентом, имеющим более высокий приоритет, в его целевом положении, разрешить конфликт добавлением задержки не получится, т.к. агент, достигнув своего целевого положения, будет продолжать стоять на месте и блокировать проход.

2.2. Метод определения и устранения конфликтов

Устранить конфликты между траекториями можно за счет изменения пути агента, а также за счет изменения его расписания. Изменение пути является более затратной процедурой в сравнении с модификацией расписания и не всегда позволяет устранить конфликт. В связи с этим предлагается устранять конфликты исключительно путем изменения расписания траекторий агентов.

Напомним, что траектории являются конфликтными, если существует момент времени *t*, в который расстояние между центрами агентов, следующих вдоль этих траекторий, меньше чем сумма их радиусов. Для проверки наличия конфликта между секциями, составляющими траектории агентов, используется метод, основанный на формуле, описанной в [22]:

$$|(\overrightarrow{x_1} + \overrightarrow{v_1}t) - (\overrightarrow{x_2} + \overrightarrow{v_2}t)| = r_1 + r_2,$$

где x_i – вектор, описывающий начальное положение агента; v_i – вектор скорости движения агента; r_i – радиус агента.

Так как рассматривается задача планирования на плоскости, эти векторы являются двумерными. Формулу можно привести к квадратичному виду:

$$(\vec{v}\cdot\vec{v})t^2 + 2(\vec{w}\cdot\vec{v})t + \vec{w}\cdot\vec{w} - (r_1 + r_2)^2 = 0;$$
$$\vec{w} = \overrightarrow{x_2} - \overrightarrow{x_1};$$
$$\vec{v} = \overrightarrow{v_2} - \overrightarrow{v_1}.$$

Обозначим $(\vec{v}\cdot\vec{v})$ как a, $2(\vec{w}\cdot\vec{v})$ как b, $\vec{w}\cdot\vec{w} - (r_1 + r_2)^2$ как c, тогда

$$t^{\pm} = \frac{(-b \pm \sqrt{b^2 - 4ac})}{2a}.$$

Если уравнение не имеет решения ($b^2 < 4ac$) или оно только одно ($t^+ = t^-$), то конфликта нет. Если же существуют два различных решения, то конфликт между агентами есть в тех случаях, если хотя бы одно из решений имеет положительное значение.

Пусть заданы пара агентов *A*, *B* и их траектории $\pi_A = \{a_1, a_2, \dots, a_n\}$, $\pi_B = \{b_1, b_2, \dots, b_m\}$. Рассмотрим пару секций $\langle a_1, a_2\rangle$ и $\langle b_1, b_2\rangle$. Начальными положениями являются a_1 и b_1 соответственно. Вектор скорости движения при этом определяется через разницу координат вершин, составляющих секцию:

$$\overrightarrow{v_1} = \left\{\frac{k(a_2.i - a_1.i)}{|\overrightarrow{a_1,a_2}|}; \frac{k(a_2.j - a_1.j)}{|\overrightarrow{a_1,a_2}|}\right\},$$
$$\overrightarrow{v_2} = \left\{\frac{k(b_2.i - b_1.i)}{|\overrightarrow{b_1,b_2}|}; \frac{k(b_2.j - b_1.j)}{|\overrightarrow{b_1,b_2}|}\right\}.$$

Нормировка на длину секций осуществляется для сохранения эквивалентности скоростей агентов. Коэффициент k регулирует скорость агентов, при этом в рассматриваемом сценарии $k = 1$. Допустим, что был обнаружен конфликт, который возникает в момент времени τ_1. Однако данная проверка не учитывает, что агенты движутся в текущем направлении пока не достигнут концов секций, поэтому, если $\tau_1 > \min(len\langle a_1, a_2\rangle, len\langle b_1, b_2\rangle)$, то в действительности конфликта между агентами нет, по крайней мере, пока они оба движутся вдоль рассмотренных секций.

Если агенты начинают двигаться вдоль проверяемых секций в разные моменты времени, то их необходимо «синхронизировать». Например, в случае когда $time_B(b_1) > time_A(a_1)$, начальным положением агента *A* будет $a_1' = \overrightarrow{v_1}(time_B(b_1) - time_A(a_1))$. Здесь и далее $time_C(c_k)$ означает момент времени, когда агент *C* начинает движение из вершины c_k и берется из расписания траектории. Очевидно, что если один агент достигает конца секции, прежде чем другой начинает двигаться вдоль своей секции, то конфликт между ними на рассматриваемых секциях невозможен. Конфликт также может возникнуть, когда агент ждет в начале секции. Для проверки конфликта такого рода используется та же процедура, с той лишь разницей, что вектор скорости движения агента, который ожидает, является нулевым.

Рассмотрим пару секций $\langle a_l, a_{l+1}\rangle$ и $\langle b_k, b_{k+1}\rangle$ между которыми есть конфликт. Для его устранения модифицируем расписание агента B, добавив ему задержку величиной δ в начало секции. δ – входной параметр, значение которого устанавливается пользователем. После того, как агенту B была добавлена задержка, он начнет движение вдоль секции $\langle b_k, b_{k+1}\rangle$ позже, а значит, его положение относительно агента A изменится и это может привести к устранению конфликта. После изменения расписания требуется повторить процедуру проверки и в случае, если конфликт остался, снова

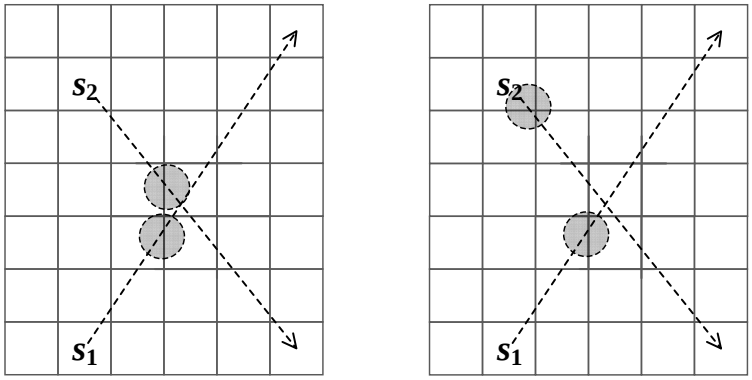


Рис. 4. Слева – между агентами есть конфликт, справа – второму агенту была добавлена задержка, устраняющая конфликт

добавить задержку и делать это до тех пор, пока конфликт не будет устранен. Пример устранения конфликта представлен на Рис. 4.

В процессе модификации расписания траектории агента B может произойти ситуация, что агенты начнут конфликтовать, пока агент B ожидает. Для того чтобы определить как долго он может находиться в той или иной вершине, используются безопасные и конфликтные интервалы [20]. Конфликтный интервал – промежуток времени, в течение которого агент не может находиться в некотором положении в связи с тем, что через него проходит динамическое препятствие (агент с более высоким приоритетом). Соответственно безопасными являются те интервалы, в течение которых агент может свободно стоять в вершине и ждать. При этом не может быть два подряд идущих безопасных интервала, т.к. между ними обязательно должен быть конфликтный интервал, в противном случае – это один безопасный интервал. Таким образом, число безопасных интервалов для некоторого положения (вершины MT-графа) всегда не более чем на единицу превышает число динамических препятствий, проходящих через него. Более того, у всех вершин MT-графа, кроме тех, что соответствуют целевым положениям агентов, гарантирован безопасный интервал, который следует за последним конфликтным интервалом.

Рассчитать безопасные и конфликтные интервалы можно для любой произвольной точки, однако делать это целесообразно лишь для положений, где агенты могут ждать, т.е. для центров вершин MT-графа. Для этого необходимо построить окружность радиусом $2r$ и центром, совпадающим с центром вершины. Если через эту окружность проходят какие-либо секции, вдоль которых движутся динамические препятствия, значит, они образуют конфликтные интервалы. Чтобы определить размер и положение конфликтного интервала на временной оси, необходимо вычислить сектор, который находится внутри окружности, а также момент времени, когда динамическое препятствие начнет двигаться вдоль него. Графический пример определения безопасных и конфликтных интервалов представлен на Рис. 5.

В случае если агент B не может устранить конфликт между секциями $\langle a_l, a_{l+1}\rangle$ и $\langle b_k, b_{k+1}\rangle$ и выходит за пределы текущего безопасного интервала в вершине b_k, необходимо модифицировать траекторию π_B так, чтобы агент приходил в начало секции $\langle b_k, b_{k+1}\rangle$ уже в следующий безопасный интервал. Иными словами, необходимо модифицировать преды-

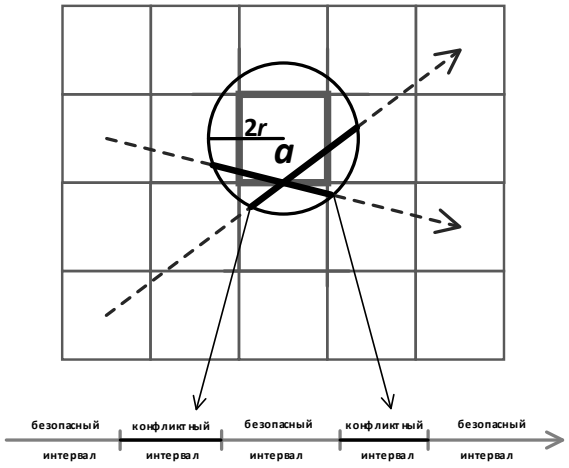


Рис. 5. Определение конфликтных интервалов для вершины a, через которую проходят траектории двух динамических препятствий

Algorithm 1: Multi-agent Path Finding with Post-coordination

```
1  paths := ∅;
2  for each agent in agents do
3      π := plan individual path for agent;
4      add π to paths;
5  trajectories := ∅;
6  for each π = {a_1, a_2, ..., a_n} in paths do
7      time(a_1) := 0;
8      i := 2;
9      while i ≤ n do
10         time(a_i) := time(a_{i-1}) + len⟨a_{i-1}, a_i⟩;
11         if time(a_i) ∉ any safe interval in a_i then
12             time(a_i) := beginning of the closest next safe interval in a_i;
13         safe_interval(a_i) := safe interval that includes time(a_i);
14     i := 1;
15     while i < n do
16         section := ⟨a_i, a_{i+1}⟩;
17         constraints := find all possibly colliding sections from trajectories;
18         collision_free := true;
19         for each constraint in constraints do
20             if section and constraint collides then
21                 collision_free := false;
22                 time(a_i) := time(a_i) + δ;
23                 if time(a_i) ∉ safe_interval(a_i) then
24                     safe_interval(a_i) := next safe interval in a_i;
25                     time(a_i) := beginning of safe_interval(a_i);
26                     i := i − 1  //need to modify previous section
27                 break;
28         if collision_free = true then
29             i := i + 1 //current section is collision free, check the next one;
30     add π to trajectories;
31     modify safe intervals of the vertices, affected by π;
32 return trajectories;
```

Рис. 6. Псевдокод предлагаемого алгоритма планирования неконфликтных траекторий для группы интеллектуальных агентов

дущую секцию $\langle b_{k-1}, b_k \rangle$, таким образом, чтобы устранить конфликт, возникающий в вершине b_k. Если же не удается устранить конфликт и при помощи добавления задержки в вершину b_{k-1}, то необходимо модифицировать секцию $\langle b_{k-2}, b_{k-1} \rangle$ и достигать b_{k-1} в следующий безопасный интервал. Стоит отметить, что если агент вышел за пределы текущего безопасного интервала, то следующий гарантированно существует, т.к. траектории агентов построены таким образом, что они не проходят через целевые положения агентов с более высоким приоритетом. В худшем случае агенту придется ожидать в стартовом положении до тех пор, пока все динамические препятствия не пройдут. Учитывая соблюдение условий правильно-сформированной инфраструктуры, агенты не могут спланировать свои траектории через стартовые положения агентов с более низким приоритетом, поэтому агенты гарантировано могут находиться в своих стартовых положениях сколь угодно долго и в конечном итоге устранить все конфликты. Следуя этой логике, происходит последовательное согласование траекторий агентов, и в итоге получается совокупность неконфликтных траекторий. Псевдокод описанного алгоритма согласования траекторий представлен на Рис. 6.

3. Экспериментальные исследования

Для проведения экспериментальных исследований были программно реализованы алгоритм Theta* для планирования индивидуальных

траекторий агентов и предложенный алгоритм их согласования (далее AA-Repair). Реализация была написана на языке C++11. Тестирование проводилось на персональном компьютере следующей конфигурации: CPU - AMD FX-8350 4.0 ГГц, 16 Гб ОЗУ, ОС – Windows 8.1 x64. Для сравнения был протестирован алгоритм AA-SIPP(m), программная реализация которого также написана на языке C++, использует аналогичные структуры данных и находится в открытом доступе. Также были протестированы: алгоритм SIPP(m), который планирует с возможностью перемещения только в четырех ортогональных направлениях, и предложенный алгоритм согласования, траектории для которого были построены алгоритмом A* с аналогичным ограничением на перемещения только в четыре стороны (далее Repair).

В качестве эвристической функции для алгоритмов Theta* и AA-SIPP(m) была использована Евклидова метрика, а для SIPP(m) и A* – метрика Манхэттена. Для сохранения свойства допустимости эвристической функции, ее вес был равен 1. Важным параметром, влияющим на работу алгоритмов и итоговые решения, является размер агентов. В проведенных экспериментальных исследованиях использовался радиус $\sqrt{2}/4$. Выбор этого значения обусловлен тем, что он позволяет алгоритмам, планирующим с возможностью перемещения лишь в четырех направлениях, совершать агентам переходы под углом 90° через одну вершину в один и тот же момент времени.

3.1. Планирование совокупности неконфликтных траекторий для множества агентов

В первой серии экспериментов рассматривалась задача планирования неконфликтных траекторий для группы агентов. Как отмечалось ранее, в этой работе не рассматривается задача назначения приоритетов, поэтому их выбор был обусловлен лишь последовательностью, в которой агенты перечислены во входном файле-задании.

Для оценки эффективности работы алгоритмов и качества найденных решений отслеживались показатели:

•Time – время, затраченное алгоритмом на поиск решения. В случае с предложенным алгоритмом учитывалось как время на поиск индивидуальных траекторий, так и время, затраченное на их согласование;

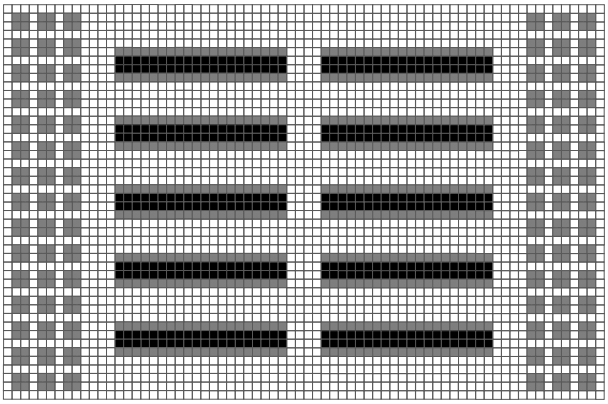


Рис. 7. Графическое представление карты размером 46x70 вершин, моделирующей складские помещения

•Makespan – оценка качества решения, которая равна максимальному времени, которое требуется одному из агентов, чтобы достичь своего целевого состояния;

•FlowTime – оценка качества решения, которая равна суммарному времени, требуемому всем агентам, чтобы достичь своих целевых положений;

•FlowLength – дополнительная оценка качества решения, аналогичная FlowTime, которая не учитывает время, которое агенты затрачивают на ожидания.

Рассматриваемые алгоритмы были протестированы на коллекции заданий для карты размером 46x70, моделирующей складские помещения (Warehouse environment), которые обладают свойством правильно-сформированной инфраструктуры и содержат 50, 100, 150, 200, 250 или 300 агентов. Для каждого числа агентов было сгенерировано по 100 заданий, т.е. общее их число составило 600. На Рис. 7 показано графическое представление карты, на котором черным цветом отмечены непроходимые области (стеллажи), а серым возможные стартовые и целевые положения агентов. Возле препятствий находятся – возможные стартовые положения, а по краям карты – целевые.

На Рис. 8 представлены результаты тестирования всех четырех алгоритмов на карте Warehouse 46x70, демонстрирующие показатели их скорости работы.

Результаты тестирования показывают, что предложенный алгоритм работает более чем на порядок быстрее, чем алгоритмы AA-SIPP(m) и

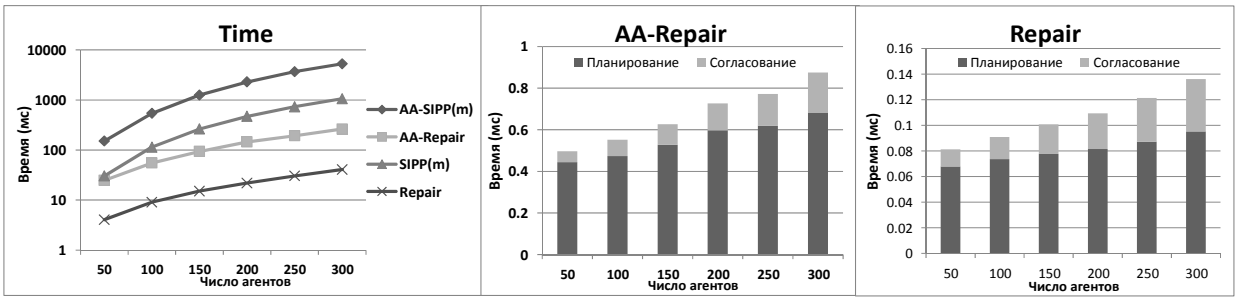


Рис. 8. Показатели скорости работы алгоритмов AA-SIPP(m) и SIPP(m), AA-Repair, Repair

SIPP(m) соответственно. Более того, чем большее число агентов содержит задание, тем больше преимущество предложенного подхода. Объяснить это можно тем, что в алгоритмах AA-SIPP(m) и SIPP(m) каждый последующий агент вынужден учитывать траектории всех агентов с более высоким приоритетом в процессе планирования, в результате чего увеличивается размер его пространства поиска и время, требуемое на отыскание траектории. В предлагаемом подходе траектории агентов планируются независимо, с учетом только стартовых и целевых положений других агентов. В результате чего, время, требуемое на отыскание траектории каждого агента, практически не зависит от их общего числа. На Рис.8 также представлено среднее время, требуемое алгоритму на то, чтобы найти траекторию и согласовать ее в зависимости от того, сколько агентов содержит задание. Для более наглядного отображения время работы было нормировано по числу агентов в задании. Как видно на рисунке, с ростом числа агентов растет доля времени, затрачиваемая на согласование траектории, а время, требуемое на отыскание индивидуальной траектории, увеличивается незначительно. При этом алгоритм AA-Repair затрачивает на планирование и согласование траектории в среднем в 6 раз больше времени, чем алгоритм Repair при любом количестве агентов в заданиях.

На Рис. 9 представлены результаты тестирования алгоритмов с точки зрения показателей качества найденных решений.

Полученные результаты показывают, что с точки зрения критерия Makespan, т.е. максимального времени, требуемого для того, чтобы все агенты достигли своих целевых положений, предложенный метод уступает алгоритмам AA-SIPP(m) и SIPP(m). С ростом числа агентов этот показатель у алгоритмов AA-SIPP(m) и SIPP(m) практически не меняется, в то время как у алгоритмов AA-Repair и Repair он постоянно растет и в итоге увеличивается почти в 2 раза. Объяснить это можно тем, что в предложенном методе все конфликты между агентами разрешаются исключительно с помощью задержек, которые накапливаются и в итоге приводят к тому, что агенты с низким приоритетом вынуждены ждать много времени. С точки зрения суммарного затрачиваемого времени (FlowTime), предложенный метод уступает алгоритмам AA-SIPP(m) и SIPP(m). С увеличением числа агентов разница достигает 20%. Если же оценивать только время, затрачиваемое на перемещения без учета ожиданий (FlowLength), то AA-Repair и Repair имеют незначительно (на 1-3%) лучшие показатели в сравнении с алгоритмами AA-SIPP(m) и SIPP(m) соответственно.

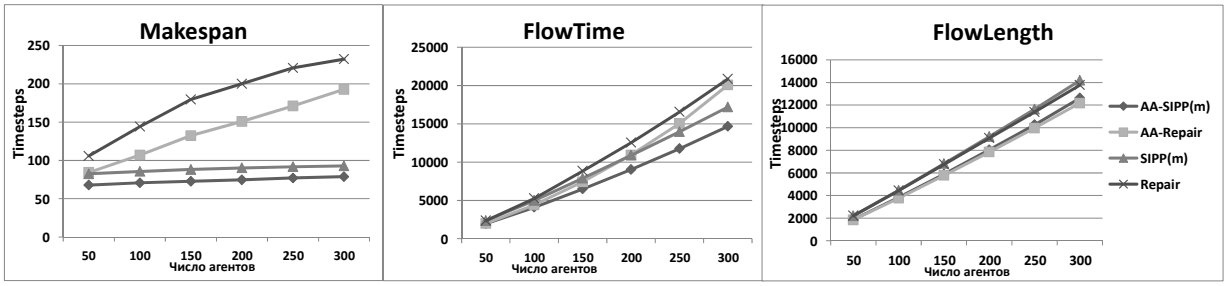
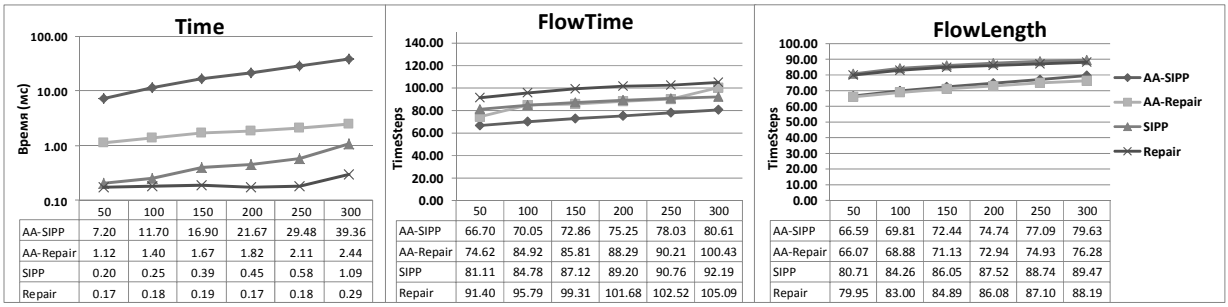


Рис. 9. Показатели качества решений, найденных алгоритмами AA-SIPP(m) и SIPP(m), AA-Repair, Repair



	50	100	150	200	250	300
AA-SIPP	7.20	11.70	16.90	21.67	29.48	39.36
AA-Repair	1.12	1.40	1.67	1.82	2.11	2.44
SIPP	0.20	0.25	0.39	0.45	0.58	1.09
Repair	0.17	0.18	0.19	0.17	0.18	0.29

	50	100	150	200	250	300
AA-SIPP	66.70	70.05	72.86	75.25	78.03	80.61
AA-Repair	74.62	84.92	85.81	88.29	90.21	100.43
SIPP	81.11	84.78	87.12	89.20	90.76	92.19
Repair	91.40	95.79	99.31	101.68	102.52	105.09

	50	100	150	200	250	300
AA-SIPP	66.59	69.81	72.44	74.74	77.09	79.63
AA-Repair	66.07	68.88	71.13	72.94	74.93	76.28
SIPP	80.71	84.26	86.05	87.52	88.74	89.47
Repair	79.95	83.00	84.89	86.08	87.10	88.19

Рис. 10. Результаты тестирования алгоритмов AA-SIPP, AA-Repair, SIPP и Repair

3.2. Планирование траектории в среде с динамическими препятствиями

Во второй серии экспериментов рассматривалась задача планирования траектории для одного агента в среде с динамическими препятствиями. В качестве входных данных использовались те же самые задания, что и в первой серии экспериментов. Траектории первых n-1 агентов были спланированы алгоритмами AA-SIPP(m) или SIPP(m) и выступали в качестве динамических препятствий. Траектория последнего агента строилась либо алгоритмами Theta* и A* и согласовывалась предложенным алгоритмом, либо алгоритмами AA-SIPP и SIPP. В качестве последнего выбирался тот агент, у которого расстояние между стартовым и целевым положением являлось максимальным.

Для оценки эффективности работы алгоритмов, а также качества найденных решений, отслеживались показатели Time, FlowTime и FlowLength. В отличие от первой серии экспериментов, при расчете оценок качества решений учитывалась лишь траектория последнего агента. В связи с этим Makespan в рассматриваемой задаче эквивалентен показателю FlowTime.

На Рис. 10 представлены результаты тестирования алгоритмов AA-SIPP, AA-Repair, SIPP и Repair.

Результаты, представленные на Рис. 10, показывают, что, как и в случае с задачей многоагентного планирования, AA-Repair и Repair работают значительно быстрее алгоритмов AA-SIPP и SIPP соответственно. Наибольшее различие наблюдается между алгоритмами AA-SIPP и AA-Repair на заданиях с большим числом агентов (250-300). Это объясняется тем, что в процессе своей работы, алгоритм AA-SIPP выполняет затратные процедуры, связанные с проверками на наличие конфликтов. В алгоритме AA-Repair подобные процедуры также есть, но они выполняются для уже готовой траектории, а не в процессе ее планирования, в результате чего достигается значительно более высокая эффективность работы. С точки зрения времени, которое требуется агенту для достижения целевого положения (FlowTime), то с увеличением числа агентов алгоритмы AA-Repair и Repair уступают алгоритмам AA-SIPP и SIPP порядка 20-25%. Объясняется это бо́льшим числом задержек, которое требуется совершить агенту при следовании по траектории, которая была согласована с траекториями динамических препятствий уже после того, как была спланирована независимо. Это подтверждается показателем FlowLength, который не учитывает задержки и при этом имеет почти эквивалентные значения для алгоритмов AA-SIPP и AA-Repair, так же как и для алгоритмов SIPP и Repair.

Заключение

В работе предложен оригинальный метод согласования траекторий, основанный на использовании безопасных интервалов и приоритизированного подхода. В отличие от существующих методов, использующих приоритизированный подход, согласование траекторий происходит уже после этапа планирования траекторий, за счет чего достигается более высокая скорость работы. Предложенный метод не обладает свойством полноты в общем случае, однако гарантирует нахождение решения в случае, если задание обладает свойствами правильно-сформированной структуры. Он может быть использован как для планирования совокупности неконфликтных траекторий для множества агентов, так и для планирования траектории индивидуального агента в среде с

динамическими препятствиями. Стоит также отметить, что предложенный алгоритм согласования траекторий не зависит от того каким алгоритмом были изначально спланированы траектории, т.е. может быть использован и с другими алгоритмами, помимо A* и Theta*, такими как, например, LIAN, который отыскивает траектории удовлетворяющие ограничению на угол поворота.

Проведенные экспериментальные исследования показали высокую вычислительную эффективность предложенного алгоритма и его применимость к рассмотренному классу задач. Качество отыскиваемых решений по ряду критериев при этом может быть хуже, чем у существующих аналогов, таких как, например, алгоритм AA-SIPP(m). Если же не учитывать время, которое агенты затрачивают на ожидания, то предложенный алгоритм находит решения, качество которых не хуже чем у алгоритмов AA-SIPP(m) и SIPP(m).

Литература

1. 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.18-32.
2. Бухвалов О.Л., Городецкий В.И., Карасев О.В. и др. Производственная логистика: Стратегическое планирование, прогнозирование и управление конфликтами // Известия ЮФУ – 2012. – №3. – С. 209–218.
3. Иванов А.М. Разработка системы межобъектного взаимодействия интеллектуальных транспортных средств // Известия ВолгГТУ. Серия «Наземные транспортные системы». Вып. 7: межвуз. сб. науч. ст./ВолгГТУ. – Волгоград, 2013б – № 21 (124) – С. 74-77.
4. Ронжин А.Л., Ву Д.К., Нгуен В.В., Соленая О.Я. Концептуальная и алгоритмические модели совместного функционирования роботизированной платформы и набора БЛА при выполнении аграрных операций // IV всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта». Труды семинара. Казань. – 2017. – С. 183 -192.
5. Мотиенко А.И. Планирование тактической траектории движения автоматизированных робототехнических средств при ликвидации последствий чрезвычайных ситуаций// Научные ведомости Белгородского государственного университета. Серия: Экономика. Информатика. – 2016. – Т. 37. – № 2 (223). – С. 139-143.
6. Van Den Berg J., Guy S., Lin M., Manocha D. Reciprocal n-body collision avoidance //Robotics research. – Springer, Berlin, Heidelberg, 2011. – P. 3-19.
7. 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. – Т. 4. – №. 2. – P. 100-107.
8. Daniel K., Nash A., Koenig S., Felner A. Theta*: Any-angle path planning on grids //Journal of Artificial Intelligence Research. – 2010. – Т. 39. – P. 533-579.
9. Андрейчук А.А., Яковлев К.С. Методы планирования траектории на плоскости с учетом геометрических ограничений //Известия РАН. Теория и системы управления. – 2017. – № 6, С. 125-140.
10. Standley T.S. Finding Optimal Solutions to Cooperative Pathfinding Problems //Proceedings of The 24th AAAI Conference on Artificial Intelligence (AAAI2010). – 2010. – Т. 1. – P. 28-29.
11. Sharon G., Stern R., Felner A., Sturtevant N.R. Conflict-based search for optimal multi-agent pathfinding //Artificial Intelligence. – 2015. – Т. 219. – P. 40-66.
12. Wagner G., Choset H. M*: A complete multirobot path planning algorithm with performance bounds //Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on. – IEEE, 2011. – P. 3260-3267.
13. Андрейчук А.А., Яковлев К.С. Метод разрешения конфликтов при планировании пространственных траекторий для группы беспилотных летательных аппаратов //Третий Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2016, 22-23 сентября 2016 г., г. Иннополис, Республика Татарстан, Россия): Труды семинара. – М: Изд-во «Перо», 2016. – 184 с. С. 31-40.
14. Silver D. Cooperative Pathfinding //AIIDE. – 2005. – Т. 1. – P. 117-122.
15. Wang K. H. C., Botea A. MAPP: a scalable multi-agent path planning algorithm with tractability and completeness guarantees //Journal of Artificial Intelligence Research. – 2011. – Т. 42. – P. 55-90.
16. Erdmann M., Lozano-Perez T. On multiple moving objects //Algorithmica, – 1987. – Т. 2, P. 1419–1424
17. C˘ap M., Nov´ak P., Kleiner A., Seleck´y M. Prioritized planning algorithms for trajectory coordination of multiple mobile robots. //IEEE Transactions on Automation Science and Engineering, – 2015. – Т. 12, №3. P. 835–849
18. Harabor D., Grastien A., Oz D., Aksakalli V. Optimal any-angle pathfinding in practice. //Journal of Artificial Intelligence Research, – 2016. – Т. 56. С. 89–118.
19. Yakovlev K., Andreychuk A. Any-angle pathfinding for multiple agents based on sipp algorithm. //Proceedings of The 27th International Conference on Automated Planning and Scheduling (ICAPS2017) – 2017. – P. 586–593.
20. Phillips M., Likhachev M. SIPP: Safe interval path planning for dynamic environments. //Proceedings of The 2011 IEEE International Conference on Robotics and Automation (ICRA-2011) – 2011. – С. 5628–5635.
21. Yap P. Grid-based path-finding //In Proceedings of 15th Conference of the Canadian Society for Computational Studies of Intelligence, Springer: Berlin, Heidelberg – 2002. – P.44-55.
22. Guy S.J., Karamouzas I. Guide to anticipatory collision avoidance. //Game AI Pro 2: Collected Wisdom of Game AI Professionals, S. Rabin, Ed., – 2015. – ch. 19. P. 195–208.

Algorithm for planning and coordination of a set of trajectories for a group of intelligent agents

A.A. Andreychuk

Peoples' Friendship University of Russia (RUDN University), Moscow, Russia

The paper considers the problem of planning a set of non-conflicting trajectories for a group of intelligent agents. The proposed algorithm uses a prioritized approach and works in two stages: independent planning of individual trajectories of agents; trajectories' coordination. A new method for trajectories' coordination is proposed that works by adding time delays only. Due to the representation of the time axis in the form of a sequence of intervals, rather than discrete moments of time, the algorithm is able to coordinate trajectories constructed any-angle path-planning algorithms. The carried out model experimental studies have confirmed the applicability of the proposed algorithm, its high computational efficiency and the comparable quality of the solutions sought.

Keywords: path-planning, multi-agent systems, conflict resolving, A*, Theta*, grid.

DOI 10.14357/20718594180407

References

1. Makarov, D.A., Panov, A.I., Yakovlev, K.S. 2015. Arkhitektura mnogourovnevoj intellektual'noj sistemy upravleniya bespilotnymi letatel'nymi apparatami [The architecture of a multi-level intelligent control system for unmanned aerial vehicles] // Iskusstvennyj intellekt i prinyatie reshenij [Artificial Intelligence and Decision Taking]. 3:18–32.
2. Bukhvalov, O.L., Gorodetskij, V.I., Karasev, O.V. et al. 2012. Proizvodstvennaya logistika: Strategicheskoe planirovanie, prognozirovanie i upravlenie konfliktami [Industrial Logistics: Strategic Planning, Forecasting and Conflict Management] // Izvestiya YUFU [SFU Tidings]. 3:209–218.
3. Ivanov, A.M. 2013. Razrabotka sistemy mezhob"ektnogo vzaimodejstviya intellektual'nykh transportnykh sredstv [Development of a system of interobject interaction of intelligent vehicles] // Izvestiya VolgGTU. Seriya «Nazemnye transportnye sistemy» [VolgSTU Tidings. Series "Land transport systems."] Vol. 7, 21(124):74–77.
4. Ronzhin, A.L., Vu, D.K., Nguen, V.V., Solenaya, O.Ya. 2017. Kontseptual'naya i algoritmicheskie modeli sovmestnogo funktsionirovaniya robotizirovannoj platformy i nabora BLA pri vypolnenii agrarnykh operatsij [Conceptual and algorithmic models of the joint operation of a robotic platform and a set of UAVs in the performance of agrarian operations] // IV vserossijskij nauchno-prakticheskij seminar "Bespilotnye transportnye sredstva s ehlementami iskusstvennogo intellekta". Trudy seminara. Kazan'. [IV All-Russian Scientific and Practical Seminar "Unmanned Vehicles with Artificial Intelligence Elements". Proceedings of the seminar. Kazan]. 183 -192.
5. Motienko, A.I. 2016. Planirovanie takticheskoj traektorii dvizheniya avtomatizirovannykh robototekhnicheskikh sredstv pri likvidatsii posledstvij chrezvychajnykh situatsij [Planning of the tactical trajectory of the movement of automated robotic means during the liquidation of the consequences of emergency situations] // Nauchnye vedomosti Belgorodskogo gosudarstvennogo universiteta. Seriya: EHkonomika. Informatika. [Scientific bulletins of the Belgorod State University. Series: The Economy. Computer science.]. Vol. 37, 2(223):139–143.
6. Van Den Berg, J., Guy, S., Lin, M., Manocha, D. 2011. Reciprocal n-body collision avoidance //Robotics research. – Springer, Berlin, Heidelberg. 3–19.
7. Hart, P. E., Nilsson, N. J., Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths //IEEE transactions on Systems Science and Cybernetics. Vol. 4, 2:100–107.
8. Daniel, K., Nash, A., Koenig, S., Felner, A. 2010. Theta*: Any-angle path planning on grids //Journal of Artificial Intelligence Research. Vol. 39:533–579.
9. Andreychuk, A.A., Yakovlev, K.S. 2017. Metody planirovaniya traektorii na ploskosti s uchetom geometricheskikh ogranichenij [Methods for planning a trajectory on a plane with geometric constraints] // Izvestiya RAN. Teoriya i sistemy upravleniya. [Tidings of RAS. Theory and control systems]. 6:125–140.
10. Standley, T.S. 2010. Finding Optimal Solutions to Cooperative Pathfinding Problems //Proceedings of The 24th AAAI Conference on Artificial Intelligence (AAAI2010). Vol. 1:28–29.
11. Sharon, G., Stern, R., Felner, A., Sturtevant, N.R. 2015. Conflict-based search for optimal multi-agent pathfinding //Artificial Intelligence. Vol. 219:40–66.
12. Wagner, G., Choset, H. 2011. M*: A complete multirobot path planning algorithm with performance bounds //Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on. – IEEE. 3260–3267.
13. Andreychuk, A.A., Yakovlev, K.S. 2016. Metod razresheniya konfliktov pri planirovanii prostranstvennykh traektorij dlya gruppy bespilotnykh letatel'nykh apparatov [Method of conflict resolution in the planning of spatial trajectories for a group of unmanned aerial vehicles] //Tretij Vserossijskij nauchno-prakticheskij seminar «Bespilotnye transportnye sredstva s ehlementami iskusstvennogo intellekta» (BTS-II-2016, 22-23 sentyabrya 2016 g., g. Innopolis, Respublika Tatarstan, Rossiya): Trudy seminara. – M: Izd-vo «Pero», [The Third All-Russian Scientific and Practical Seminar "Unmanned Vehicles with Artificial Intelligence Elements" (BPS-AI-2016, September 22-23, 2016, Innopolis, Republic of Tatarstan , Russia): Proceedings of the Seminar. - M: Publishing house "Pero"]. 31–40.

14. Silver, D. 2005. Cooperative Pathfinding //AIIDE. Vol. 1:117–122.
15. Wang, K. H. C., Botea, A. 2011. MAPP: a scalable multi-agent path planning algorithm with tractability and completeness guarantees //Journal of Artificial Intelligence Research. Vol. 42:55-90.
16. Erdmann, M., Lozano-Perez, T. 1987. On multiple moving objects //Algorithmica. Vol. 2:1419–1424
17. C˘ap, M., Nov´ak, P., Kleiner, A., Seleck´y, M. 2015. Prioritized planning algorithms for trajectory coordination of multiple mobile robots. //IEEE Transactions on Automation Science and Engineering, Vol. 12(3):835–849
18. Harabor, D., Grastien A., Oz D., Aksakalli V. 2016. Optimal any-angle pathfinding in practice. //Journal of Artificial Intelligence Research. 56:89–118
19. Yakovlev, K., Andreychuk, A. 2017. Any-angle pathfinding for multiple agents based on sipp algorithm. //Proceedings of The 27th International Conference on Automated Planning and Scheduling (ICAPS2017). 586–593
20. Phillips, M., Likhachev, M. 2011. SIPP: Safe interval path planning for dynamic environments. //Proceedings of The 2011 IEEE International Conference on Robotics and Automation (ICRA-2011). 5628–5635.
21. Yap, P. 2002. Grid-based path-finding //In Proceedings of 15th Conference of the Canadian Society for Computational Studies of Intelligence, Springer: Berlin, Heidelberg. 44-55.
22. Guy, S.J., Karamouzas, I. 2015. Guide to anticipatory collision avoidance. //Game AI Pro 2: Collected Wisdom of Game AI Professionals, S. Rabin, Ed., ch. 19:195–208.