УДК 004.942

РАЗРАБОТКА АГЕНТНОЙ МОДЕЛИ ДЛЯ ПЛАНИРОВАНИЯ ТРАЕКТОРИИ ПОЛЕТА БЕСПИЛОТНОГО ЛЕТАТЕЛЬНОГО АППАРАТА

Стороженко А.М. (Санкт-Петербург), Фомин И.Н., Костерев А.А., Петров Д.Ю.(Саратов)

Применяемые методы и инструменты расчета траектории полета БПЛА

На современном этапе развития науки и техники особенную популярность приобретают исследования, которые связаны с проектированием беспилотных летательных аппаратов (БПЛА). Это одна из приоритетных областей разработок в области искусственного интеллекта и робототехники. Приоритет в этих исследованиях имеют БПЛА вертикального взлета и посадки мультироторного типа, так как комплектующие подобных систем достаточно дешевые и распространенные, к тому же, такие БПЛА уже снабжены готовыми платформами. Главной задачей разработчиков любых таких аппаратов является повышение уровня автономности объекта управления [1].

Набор технических характеристик беспилотных транспортных средств, с различными алгоритмами управления представлен в [2], алгоритмы управления для нахождения допустимого и безопасного пути, который не только предупреждает столкновения, но и точно измеряет расстояние до потенциально опасных объектов, хорошо изучены в [3]. В этой публикации были качественно проанализированы процедуры формирования траектории и выявления относительного направления движения беспилотного объекта в условиях мультиагентной среды с помощью имитационных агентных моделей и было найдено применение алгоритму PFRAOR для решения такого рода задач. В [4] предлагается метод, позволяющий на основе модели динамики полета и допустимого управления, в том числе при определенных разумных предположениях, описать ограничения на геометрию траектории.

С целью избежания описания всех менее эффективных алгоритмов, в настоящей работе описана задача расчета траектории полета по обобщенному алгоритму геометрических ограничений. В этом алгоритме траектория представляется виде набора последовательных отрезков, в которых угол отклонения между всякими смежными отрезками последовательности по модулю не более фиксированного значения α. Если предположить, что следование такому условию гарантирует реализуемость полученной траектории, посредством выработки допустимых управляющих сигналов на исполнительные элементы БПЛА, то с учетом вышеобозначенных ограничений, можно применять метод, базирующийся на анализе области достижимости динамических систем.

При моделировании таких процессов главное предположение состоит в том, что условия полета и точность управления, учитывая ограничения, должны быть таковы, что возникает возможность обеспечения допустимого управления [4]. Наличие такой возможности гарантирует удержание мультикоптера в некоторой допустимой области требуемой прямолинейной траектории полета. Эту окрестность определяют «трубкой» с радиусом r_d (рис. 1). Конкретное значение r_d зависит от типа БПЛА, режима и условий полета и множества других факторов.

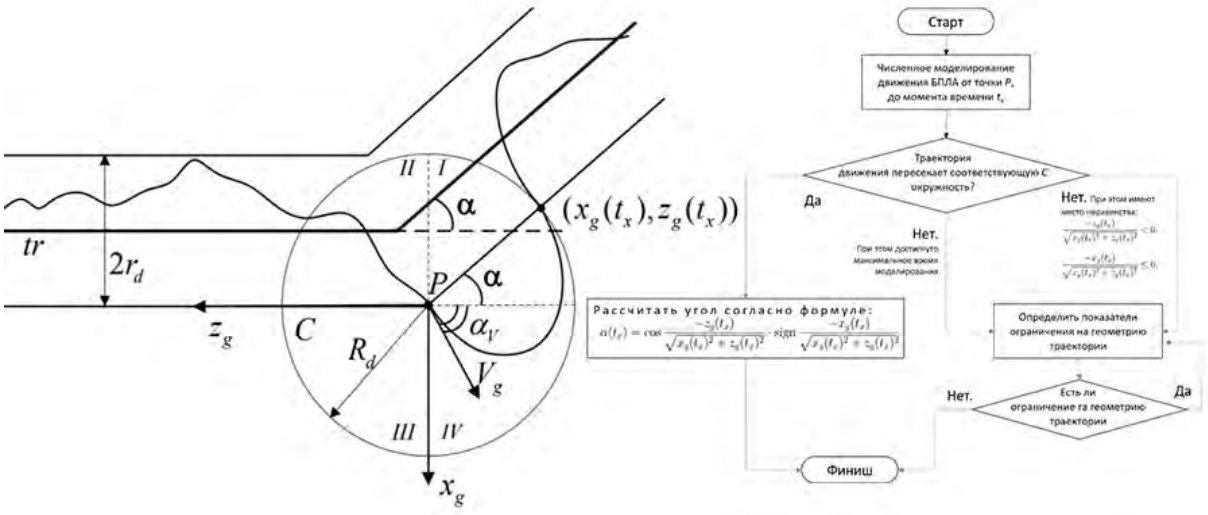

Рис. 1. Иллюстрация метода и алгоритма определения геометрических ограничений

Иллюстрация метода геометрических ограничений показывает, что его алгоритм может быть легко построен с применением основных законов тригонометрии. Для траектории, показанной на рис. 1, самой сложной для управления является положение БПЛА в точке *P*. Если ни одна точка в не содержит препятствий, то задача определения геометрических ограничений является задачей поиска такого максимального угол α, при котором траектория полета мультикоптера, не покидая круга, вернется в допустимую «трубку», которую уже не покинет, так как при допущении, обозначенном выше, всегда существует допустимое управление, гарантирующее это [5]. Радиус R_d, при этом, определяет некоторую область маневра поворота, в которой не должны находиться препятствия. В этом случае, алгоритм по методу определения геометрических ограничений визуализируется в виде, показанном на рис. 2, что дает представления о принципах построения алгоритмов PFRAOR, LIAN, MultiLIAN, MultiLIAN-RU, MultiLIAN-CC для их анализа.

Визуализация алгоритма геометрических ограничений наглядно показывает, что координаты БПЛА $x_g(t_x)$, $z_g(t_x)$ – в точке $Ox_gy_gz_g$ в момент времени t_x, соответствующие системе геометрических неравенств, фактически снимают ограничения на геометрию траектории (рис. 1). Случаи несоответствия этой системе неравенств приводят к сложному поведению системы, при котором траектория не выходит за секторы IV и I. При достаточно длительном времени моделирования можно сделать вывод, что ограничений на геометрию траектории тоже нет. В случае если траектория движения пересекает соответствующую вектору *C* окружность означает, что задано чрезмерно большое значение R_d для выбранных БПЛА и условий полета. Отрицательное значение угла α значит выбор чрезмерно малого значения R_d.

Алгоритмы MultiLIAN

При практических конструированиях БПЛА часто применяется алгоритм MultiLIAN, который представляет собой алгоритм нахождения множества путей, которые удовлетворяют ограничениям на максимальный угол отклонения и влияют на динамику передвижения БПЛА. Этот алгоритм является модификацией популярного алгоритма LIAN.

Алгоритм MultiLIAN формирует альтернативные траектории при помощи алгоритма LIAN, выявляет множество промежуточных вершин построенного таким образом графа, пересчитывает альтернативный путь с учетом найденных

промежуточных вершин и, если составлено требуемое количество путей, дает команду на исполнение оптимального, в противном случае вновь происходит поиск множества промежуточных вершин сформированного графа. Вид составленных алгоритмом альтернативных путей полностью зависит от того, как отбираются промежуточные вершины.

Иногда MultiLIAN формирует альтернативные пути, равноценные основному. Чтобы решить такую проблему, необходимо пробовать разнообразные методы выбора промежуточных вершин. Метод формирования совокупности путей алгоритма MultiLIAN называют «методом грубой силы», поскольку на каждом шаге этого алгоритма альтернативный путь ищется без учета результатов прошедших шагов. На основе этого были разработаны две модификации данного алгоритма, которые учитывают данный недостаток и тем самым сокращают пространство поиска.

Одна из модификаций этого алгоритма MultiLIAN под названием MultiLIAN-RU рассчитывает альтернативные пути, и формирует перечни параметров, которые фактически характеризуют частичный путь до соответствующей вершины графа из начальной. Таким образом строится граф, отбор вершин которого производится при помощи расчета расстояний до промежуточных вершин, что дает возможность, при поиске альтернативного пути к промежуточным вершинам больше не обращаться.

Другая вариация алгоритма MultiLIAN под названием MultiLIAN-CC, использует данные прошлых шагов и после первого шага вместе с вершинами графа, формирующими искомый путь, определяет вершины, которые не были использованы. Предполагается, что подобные вершины «лишние» и при очередных шагах поиска они исключаются из рассмотрения. Основные шаги алгоритма MultiLIAN-CC в виде блок-схемы и параметры его элементов для имитационной модели показаны на рис. 2.

Элемент	Параметры	Применение
g(a)	s — стартовая точка, a — промежуточная точка	Описывает протяжённость кратчайшего пути из s в a
bp(s)	a — промежуточная точка	Определяет родительский указатель точки a
open{b1, b2, …}	b1, b2, … — нерассмотренные вершины	Содержит все нерассмотренные
close{a1, a2, …}	a1, a2, … — рассмотренные вершины	Содержит все рассмотренные вершины
succ{c1, c2, …}	c1, c2, … — потенциальные последователи	Содержит всех потенциальных последователей для точки a
exclosed{e1, e2, …}	e1, e2, … — исключённые вершины	Содержит все тупиковые вершины, исключённые из рассмотрения
paths{p1, p2, …}	p1, p2, … — найденные маршруты	Содержит все найденные маршруты

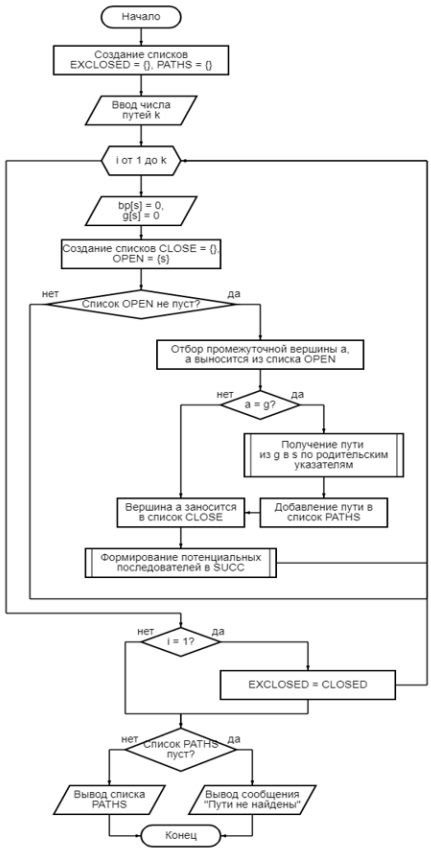

Рис. 2. Основные шаги алгоритма MultiLIAN – CC и параметры его элементов для имитационной модели

Имитационное моделирование функций алгоритма MultiLIAN

Для создания диаграммы состояний и имитационной модели движения БПЛА в среде AnyLogic были определены элементы модели, собранные в таб. 1, и основные сущности алгоритмов расчета траектории полета, отраженные на рис. 2.

Таблица 1

Элементы агентной имитационной модели алгоритма MultiLIAN-CC

Параметр или функция	Комментарий
Copter copter [..]	Тип агента Copter — популяция агентов БПЛА
Obst obst	Тип агента Obst — препятствие
End end	Тип агента End — целевая точка, к которой движется БПЛА
Void setCell (int row, int column)	Функция, помещающая агента в установленную ячейку
Void add_Copter()	Функция, добавляющая агента в популяцию Copter
Int size (Population population)	Функция, возвращающая размер популяции
Agent get (int i)	Функция, возвращающая агента популяции под номером i
Void moveToNextCell (CellDirection direction)	Функция, помещающая агента в соседнюю ячейку, которая располагается в указанном направлении от его текущей ячейки с параметрами направления direction
Int getR()	Функция, возвращающая строку ячейки текущего месторасположения агента
Int getC()	Функция, возвращающая столбец ячейки текущего месторасположения агента
Agent getAgentAtCell (int row, int column)	Функция, возвращающая агента, который занимает указанную ячейку, или null, если ячейка пуста
Void remove_copter (Copter copter)	Функция, удаляющая указанного агента из популяции
Public class Nod	Java класс — класс, элементы которого хранят в себе текущее местоположение: строку x и столбец y
Void getx()	Возвращает строку x элемента класса Nod
Void gety()	Возвращает столбец y элемента класса Nod
ArrayList < Nod > open	Переменная — массив, в котором находятся нерассмотренные вершины
ArrayList < Nod > close	Переменная — массив, в котором находятся рассмотренные вершины
ArrayList < Nod > succ	Переменная — массив, в котором находятся потенциальные последователи вершины
ArrayList paths	Переменная — массив, в котором хранятся построенные маршруты
Void add (Nod nod)	Функция, добавляющая элемент в массив Nod
Double alpha = 24.4	Константа, которая хранит в себе угол отклонения для выбранного БПЛА
ArrayListGFG (double xcenter, double ycenter, double r)	Функция, возвращающая массив точек окружности по алгоритму Midpoint circle
Double NormalizeAngle (double angle)	Вспомогательная функция, нормализующая угол для выбранного диапазона [0, 360]
Double AngleOfReference (Vector v)	Вспомогательная функция, возвращающая значение угла между выбранным и статичным вектором (упрощает вычисления)
Double AngleOfVectors (Vector first, Vector second)	Основная функция, вычисляющая угол между двумя векторами

Такой обширный набор введенных параметров для объектов позволил создавать и учитывать массивы с построенными путями с использованием процедуры выбора промежуточных вершин для альтернативных путей, а также задать оригинальные параметры задействованных элементов.

Имитация движения происходила согласно реализованной диаграммы состояний, показанной на рис. 3. В начальном состоянии задается местоположение целевой точки и препятствия при помощи функции «setCell». В состоянии 1 создается агент популяции и определяется число дополнительных путей, которые необходимо отыскать. Алгоритм по построенной диаграмме в этом состоянии работает, пока не будут найдены все эти альтернативные пути.

В состоянии 2 происходит отбор промежуточной вершины Q (для альтернативных путей). Если массив «open» непустой, то в состоянии 3 происходит отбор промежуточной вершины a. Если агент БПЛА достиг целевой точки, тогда кратчайший путь восстанавливается по родительским указателям в состоянии с параметром «Объект достиг целевой точки». При этом, сформировавшийся путь добавляется в соответствующий массив данных, а вершина заносится в список «close» в состоянии с параметром «Определение последователей точки». Там же формируются потенциальные последователи вершины, пересчитываются g-значения, родительский указатель, а последователи заносятся в список «open». Когда найдено требуемое число маршрутов, соответствующий массив «paths» проверяется на наполненность: если он пустой, выводится соответствующее сообщение, в противном случае маршруты отображаются на экране.

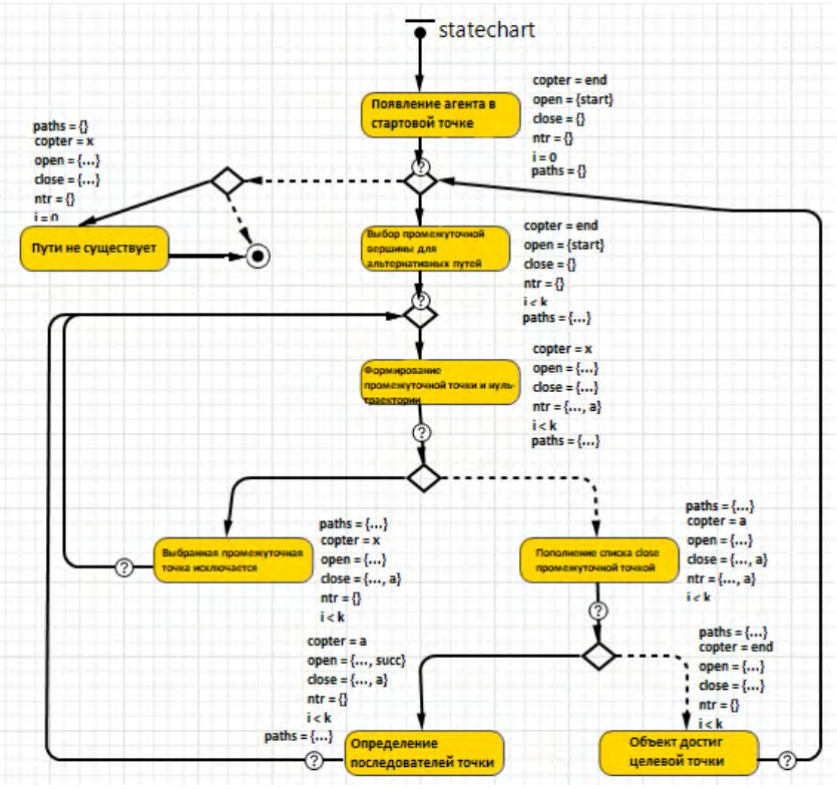

Рис. 3. Диаграмма состояний алгоритма MultiLIAN-CC

Заключение

Формализация алгоритмов, определение набора параметров на диаграмме состояний и имитационное моделирование в среде AnyLogic движения БПЛА под управлением алгоритмов расчета траектории PFRAOR, LIAN, MultiLIAN, MultiLIAN-

RU, MultiLIAN-CC дали возможность выработать набор формальных и неформальных рекомендаций по применению различных алгоритмов управления БПЛА.

Алгоритм PFRAOR является одним из самых быстрых вариантов, однако он слишком сырой, чтобы применять его на реальных объектах. Имитационное моделирование показало, что БПЛА под управлением LIAN работает медленнее в сравнении с другими алгоритмами, но его эффективность существенно выше. Поиск дополнительных маршрутов наделяет алгоритм MultiLIAN множеством преимуществ при управлении БПЛА, кроме того, он достаточно точный и эффективный. Его модификация RU может применяться в системах с ограниченными вычислительными мощностями. При этом, надо иметь в виду, что его применение связано с расходом ресурсов на поиск в том числе не оптимальных вариантов. Модификацию CC можно рекомендовать к применению, если стоит задача построения маршрута БПЛА и альтернативных путей за минимальный временной период. Результаты теоретико-информационного анализа алгоритмов расчета траектории полета БПЛА на основе имитационных моделей представлены в табл. 2.

Таблица 2

Результаты сравнительного анализа алгоритмов расчета траектории полета БПЛА на основе имитационных моделей

Алгоритм	PFRAOR	LIAN	MultiLIAN	MultiLIAN-RU	MultiLIAN-CC
Основной метод оптимизации	Метод обхода препятствий.	h(a) — эвристическая оценка протяжённости пути; Δ — радиус дискретной окружности.	Метод выбора промежуточных вершин.	Метод формирования альтернативных стартовых вершин.	Метод отбора лишних вершин.

Основные методы имитационной модели	void setCell(int row, int column);
	
	Agent get(int i);
	
	void moveToNextCell(CellDirection direction);
	
	Agent getAgentAtCell(int row, int column).
		ArrayList bresenham (double xstart, double ystart, double xend, double yend);
		
		ArrayList GFG(double xcenter, double ycenter, double r);
		
		double AngleOfReference(Vector v);
		
		void add(Nod nod).
			ArrayList GFG(double xcenter, double ycenter, double r);
			
			double AngleOfReference(Vector v);
			
			void add(Nod nod).
				ArrayList GFG(double xcenter, double ycenter, double r);
				
				double AngleOfReference(Vector v);
				
				void add(Nod nod).
					ArrayList GFG(double xcenter, double ycenter, double r);
					
					double AngleOfReference(Vector v);
					
					void add(Nod nod).

Главная особенность	Высокая скорость работы, простота реализации.	Эффективно справляется со своей задачей	Достаточно успешно находит основной и альтернативные пути.	Существенно экономит ресурсы и успешно находит основной маршрут.	Высокий уровень результативности и эффективности.
Главный недостаток	Отсутствует программное описание обхода препятствий.	Находит только один маршрут.	Ресурсозатратен, иногда альтернативные пути совпадают.	В большинстве случаев ошибается при нахождении альтернативных путей.	Возможно сложная реализация.
Условия и особенности использования	Использование не рекомендуется.	Рекомендуется применять, если стоит задача поиска единственного маршрута.	Стоит применять, если на решение задачи поиска альтернативных путей выделены дополнительные ресурсы.	Можно применять в малоресурсных системах. Однако, лучше обратить внимание на другие алгоритмы.	Рекомендуется применять в любых задачах поиска основного и альтернативных путей.

Достигнутые результаты подтверждают теоретические выкладки об особенностях метода определения геометрических ограничений, позволяют далее усложнять модели, проводить эксперименты с количеством просчитываемых альтернативных путей, строить модели с реальными топографическими данными. Ожидаемым результатом экспериментов могут стать новые модификации алгоритмов расчета траектории полета БПЛА или создание новых алгоритмов.

Литература

1. Лопаткин Р.Ю., Петров С.А., Игнатенко С.Н., Иващенко В.А. Перспективы применения имитационного моделирования в задачах автоматизации и управлении технологическими системами // Вісник НТУ «ХПІ». Серія: Механіко-технологічні системи та комплекси. Харків: НТУ «ХПІ», 2015. No 17(1189). С.61–71.

2. Лю Ш. Разработка беспилотных транспортных средств / Ш. Лю, Л. Ли, Ц. Тан, Ш. Ву, Ж-Л. Годье. М: ДМК Пресс, 2022. 246 с.

3. Скакун А.Д. Алгоритм планирования траектории мобильного робота / А.Д. Скакун, Н.Ч. Дас, Л.В. Коломиец, З.Х. Зим, Р. Уддин // Международная конференция по мягким вычислениям и измерениям: материалы. 2021. Т. 1. С. 228–231.

4. Сенчук Д.В. К вопросу о построении системы управления беспилотной авиационной системой // Имитационное моделирование. Теория и практика: девятая всероссийская научно-практическая конференция по имитационному моделированию и его применению в науке и промышленности (ИММОД-2019): труды конференции, 16–18 октября 2019 г. Екатеринбург: Урал. гос. пед. ун-т., 2019. С. 525–529.

5. Яковлев К.С. Метод автоматического планирования траектории беспилотного летательного аппарата в условиях ограничений на динамику полета / К.С. Яковлев, Е.С. Баскин, Д.А. Макаров // Искусственный интеллект и принятие решений. 2014. №4. С. 3–17.