**УДК 004.8; 519.688; 681.3**

# Анализ и модификация алгоритма Lian - планирования траектории на основе сетки с ограничением по углу поворота.

**Сизых В.С.** (ИГУ), **Халматова А.С.** (ИГУ)
**Научный руководитель – к.ф.-м.н., доцент Лебедев В.П. (ИГУ)**

**Введение.** В настоящее время эвристический алгоритм поиска кратчайшего пути на квадратной сетке с учетом ограничения на крутизну поворота - Lian [1] является одним из самых перспективных с практической точки зрения. Lian может широко использоваться для расчета траектории движения программируемых беспилотных летательных аппаратов (БПЛА). Искомая траектория имеет ограничения, она должна быть С-кривой, т.к. БПЛА не может двигаться скачками. Кривизна траектории не может превосходить заданного значения из-за ограниченного радиуса виража. Не учет этих ограничений приводит к износу аппаратных элементов. Поэтому сглаженность пути имеет большое значение в программировании БПЛА [2].

В работе представлен доработанный алгоритм Lian*, позволяющий минимизировать с учетом заданных весовых функций, как общую длину пути, так и общее количество поворотов. При разработке алгоритма и реализации программы особое внимание уделено оптимальности кода и скорости работы программы.

Целью работы - разработка идеи модификации Lian (Lian*) и разработка программы с бóльшим акцентом на сглаженность пути.

**Основная часть.** Базовый Lian работает на двух сетах (очередях с приоритетом): в одной (открытой) лежат рассмотренные точки, а в другой (закрытой) – потенциальные для рассмотрения. Каждая точка представляется в виде [*(x, y), d, (p.x, p.y)*], где *(x, y)* – это координаты точки, *d* – это длина пути от начальной точки, *(p.x, p.y)* – это координаты предыдущей точки (родителя). На каждой итерации происходит следующий набор действий:

1. Из открытой очереди выбирается точка с наименьшим *d*.
2. Вокруг нее строится окружность из точек с заданным в начале фиксированным радиусом. Все они добавляются в открытую очередь.
3. Выбранная точка уходит в закрытую очередь.

Эти действия повторяются, пока не будет найдена целевая точка, или не закончится открытая очередь.

В новую версию Lian* были введены следующие изменения:

- Точка теперь хранится в виде [*(x, y), d, a, (p.x, p.y)*], где *a* – это сумма углов, которая является суммой *p.a* и угла поворота.
- Добавлено новое условие на угол: если точка с совпадающими *(x, y)* была рассмотрена ранее, то в открытой очереди остается та, у которой *a* меньше. В старой версии лучшая точка выбиралась по значению *d*.
- Изменен приоритет открытой очереди. Если в старой версии точка с наименьшим *d* считалась лучшей, то теперь в приоритет добавлен параметр *a*. Для обоих значений введены коэффициенты значимости.

Основными критериями оценки данных изменений были параметры: время работы, итоговая сумма углов, длина пути. Приоритет значимости таков:

1. итоговая сумма углов;
2. длина пути;
3. время работы алгоритма.

Ускорить работу алгоритма можно распараллеливанием процесса. При столкновении пути с препятствием алгоритм разветвляет путь для обхода. Поочередно выбирается точка с каждой ветки, т.о. ветки лучше рассматривать параллельно.

**Выводы.** Разработанная версия Lian* обладает всеми достоинствами, что и Lian, т.е. алгоритм всегда находит гладкий и короткий путь между точками с учетом сложного рельефа и препятствий. Однако, среди множества близких траекторий (близких по длине пройденного пути) позволяет выбрать ту, которая имеет меньшее суммарное количество поворотов.

Проведено серьезное тестирование алгоритма Lian*, в результате которого выяснено, что достаточно часто встречаются случаи, когда отличие в общей длине пути в множестве найденных траекторий составляет менее 0.01%, однако общее количество поворотов может варьироваться до 10-20%, что дает возможность выбрать более гладкую траекторию без существенного удлинения пути. Проведена оптимизация Lian*, в результате которой время работы программы по сравнению с базовой версией Lian увеличивается не значительно.

Ссылка на репозито́рий: https://github.com/CodeR-na-r1/LianStar

**Список использованных источников**:

1. Yakovlev K., Baskin E., Hramoin I. Grid-based angle-constrained path planning // Proceedings of the 38th German Conference on Artificial Intelligence (KI-2015). 2015. P.208-221.

2. А.А. Ардентов, И. Ю. Бесчастный, А.П. Маштаков, А.Ю. Попов, Ю.Л. Сачков, Е.Ф. Сачкова. *Алгоритмы вычисления положения и ориентации БПЛА* // Программные системы: теория и приложения: электрон. научн. Журн. 2012. Т. 3, № 3(12). с. 24.