ORES
Научно-практический электронный журнал
Оригинальные исследования (ОРИС), *ISSN 2222-0402*
www.ores.su
original-research.ru

УДК 62-5

МОДЕЛИРОВАНИЕ, ОДОМЕТРИЯ КОЛЕС, ROBOT LOCALIZATION, ТЕСТИРОВАНИЕ НАВИГАЦИИ, РЕКОНСТРУКЦИЯ КАРТЫ ПОМЕЩЕНИЯ

Положай Владислав Григорьевич
Магистр
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»
e-mail: vlados988111@gmail.com

Аннотация

Объектом исследования являются навигация автономного устройства в пространстве и определение границ помещения. Цель работы – решение проблем, связанных с проектированием, тестированием на ранних стадиях разработки и улучшением навигационных алгоритмов в условиях замкнутого пространства. В рамках работы были промоделированы два алгоритма: алгоритм реконструкции контура помещения на основе данных одометрии; алгоритм сбора данных одометрии колес робота пылесоса. Данные, полученные вследствие моделирования алгоритмов, могут быть использованы для определения недостатков алгоритмов навигации и последующего их улучшения и оптимизации. Были использованы технологии: Python3, Webots, NymPy, Matplotlib, NetworkX, SciPy. Результаты и разработки из работы могут быть использованы в дальнейшем на ранних этапах тестирования алгоритмов навигации автономных устройств.

Ключевые слова: моделирование, навигационные алгоритмы, одометрия, колесная одометрия.

MODELING, ROBOT LOCALIZATION, ALGORITHM FOR RECONSTRUCTING THE CONTOUR OF A ROOM BASED ON WHEEL ODOMETRY DATA

Vladislav G. Polozhay
Master,
Saint Petersburg Electrotechnical University "LETI"
e-mail: vlados988111@gmail.com

ABSTRACT

This research focuses on the navigation of autonomous devices in space and the determination of room boundaries. The aim of the study is to resolve issues associated with designing, testing at early stages of development, and improving navigation algorithms in closed spaces. Two algorithms were modelled within the scope of this study: algorithm for reconstructing

the contour of a room based on odometry data and; algorithm for collecting wheel odometry data of a robot vacuum cleaner.

The data obtained because of algorithm modeling can be used to identify the shortcomings of navigation algorithms and to subsequently improve and optimize them. Technologies such as Python3, Webots, NymPy, Matplotlib, NetworkX, and SciPy were employed during the research. The findings and developments from this study can be applied in future early-stage testing of navigation algorithms for autonomous devices.

Keywords: modeling, navigation algorithms, odometry, wheel odometry.

ВВЕДЕНИЕ

Робототехника и искусственный интеллект стремительно развиваются, тем самым прокладывая путь к разработке различных автономных систем, которые могут ориентироваться и работать в различных средах и пространствах. Одна из фундаментальных проблем, с которой сталкиваются эти система – это способность воспринимать и понимать свое окружение. Поэтому необходимо разрабатывать и совершенствовать алгоритмы локализации и оценки карты, которые могут эффективно и надежно использовать данные одометрии для построения карт помещения, снижая при этом влияние накопленных ошибок, в то же время существенно снижая стоимость производства, и обслуживания автономных устройств. Вслед за этим возрастает потребность в тестировании на ранней стадии разработки алгоритмов навигации робота в пространстве и определения геометрии помещения, в котором оперирует робот [1 – 3].

Тестирование алгоритмов навигации в замкнутом пространстве требует проверки значительного числа сценариев, которые практически невозможно регулярно проводить в виде натурных экспериментов. Для решения проблемы проведения множественных экспериментов с алгоритмами предлагается разработать виртуальный стенд, генерирующий данные с датчиков одометрии по заданному сценарию движения робота, а также реализовать прототип алгоритма построения карты замкнутого окружения на основе сгенерированных данных.

Моделирование алгоритма реконструкции комнаты

Описание работы алгоритма

Поскольку для определения границ помещения доступны только данные одометрии с колес робота, обрабатывать следует данные, полученные при движении робота по стенам помещения. Таким образом, путем построения позиционного графа на основе траектории движения робота, его оптимизации и получения замкнутой траектории робота, мы и определим искомые границы комнаты.

Сегментация пути робота

При движении робот рассчитывает свои координаты каждые 64 миллисекунды. Эти данные в дальнейшем сохраняются в виде массива точек в формате - (x; y). На данном этапе уже можно построить позиционный граф, но поскольку вычислительная мощностью робота ограничена с целью увеличение энергоэффективности и уменьшения стоимости, необходимо произвести сегментацию пути робота. Для этого будет использован Алгоритм Дугласа-Пекера. Этот алгоритм эффективен для уменьшения количества точек на кривой без значительной потери точности [4].

Алгоритм выполняется следующим образом:

1. Вход: множество точек, представляющих кривую.

2. Выбирается начальная и конечная точки кривой. Эти две точки формируют линию.
3. Для каждой точки на кривой находится перпендикулярное расстояние от точки до линии, образованной начальной и конечной точками.
4. Находится точка, расстояние которой до линии максимально. Если это расстояние больше заданного порога (например, значения ошибки, которую мы готовы допустить), то линия разбивается в этой точке на две.
5. Предыдущий шаг повторяется рекурсивно для каждого из новых отрезков (между начальной точкой и точкой максимального отклонения, а затем между точкой максимального отклонения и конечной точкой).
6. Процесс продолжается до тех пор, пока расстояние от каждой точки до линии не станет меньше заданного порога.
7. Выход: сокращенное множество точек, представляющих кривую.

Сегментация пути представляет из себя преобразование индивидуальных точек пути робота в прямолинейные сегменты. Это позволяет уменьшить общее количество координат в пути робота, путем создания доминантных точек. Пример сегментации показан на рисунке 1.

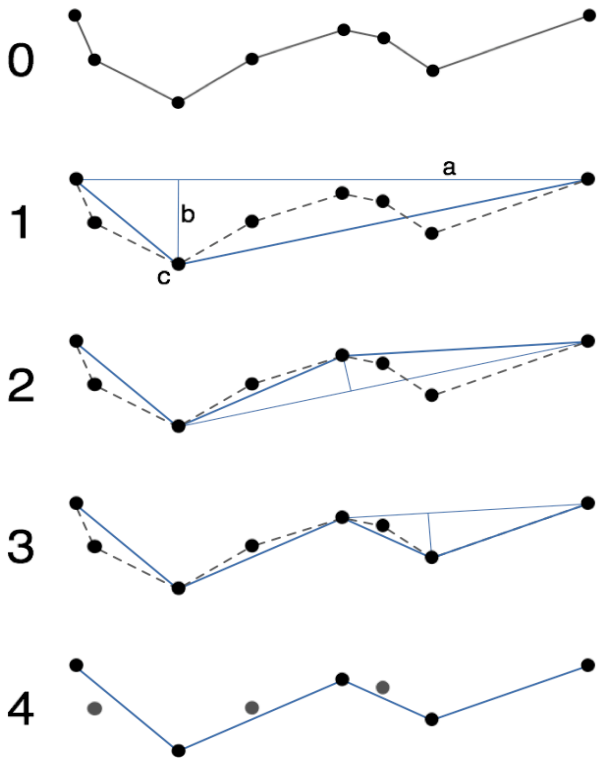

Рисунок 1 - Создание доминантных точек при сегментации пути робота

Реализация сегментации с помощью Алгоритма Дугласа-Пекера (Douglas-Peucker) [7] показана на рисунке 2.

Algorithm 1 DP Generation

- **Parameters:** L_{min}, e_{max}
- **Inputs:** $\boldsymbol{x}$
- **Outputs:** DP

$DP = [\boldsymbol{x}_0]$
$S = [\boldsymbol{x}_0]$
if new $\boldsymbol{x}$ available **then**
    $d = ||DP_{\text{end}} - \boldsymbol{x}||$
    if $d < L_{\text{min}}$ **then**
        $S \leftarrow [S, \boldsymbol{x}]$
    else
        $S_{\text{tmp}} = [S, \boldsymbol{x}]$
        $e = \text{errorLineFit}(S_{\text{tmp}})$
        if $e < e_{\text{max}}$ **then**
            $S \leftarrow S_{\text{tmp}}$
        else
            $DP \leftarrow [DP, S_{\text{end}}]$
            $S \leftarrow [S_{\text{end}}, \boldsymbol{x}]$
        end if
    end if
end if

Рисунок 2 - Алгоритм Дугласа-Пекера (Douglas-Peucker) для упрощения линий

Функция принимает список точек, минимальный порог расстояния (Lmin) и максимальный порог ошибки (emax) в качестве входных данных и возвращает упрощенную полилинию списком точек, выбранных из входного списка точек.

Алгоритм работает следующим образом:

1. Инициализируется упрощенная полилиния (DP) с первой точкой входного списка (x).
2. Инициализируется временный список точек (S) с первой точкой входного списка.
3. Итерация по оставшимся точкам во входном списке (x [1:]) и выполнение следующих шагов для каждой точки (xi):
   3.1. Расчёт расстояния (d) между последней точкой в DP и текущей точкой (xi).
   3.2. Если расстояние меньше минимального порога расстояния (Lmin), добавляется текущая точка (xi) во временный список (S).
   3.3. Если расстояние больше или равно Lmin, создается новый временный список (Stmp), добавляя текущую точку (xi) в существующий список (S). Рассчитывается ошибка (e) между фактическими точками в Stmp и линией наилучшего приближения с использованием функции error_line_fit.
      3.3.1. Если ошибка меньше максимального порога ошибки (emax), устанавливается S на новый временный список (Stmp).
      3.3.2. Если ошибка больше или равна emax, добавляется последняя точка в S к DP и сбрасывается S, чтобы содержать последнюю точку в S и текущую точку (xi).
4. Добавляется последняя точка в S к DP.

5. Возвращается упрощенная полилиния (DP).

Функция error_line_fit вычисляет квадратичную ошибку между точками и линией наилучшего приближения. Она принимает список точек в качестве входных данных и возвращает значение ошибки. Функция работает следующим образом:

1. Рассчитывается линия наилучшего приближения с использованием функции, которая возвращает коэффициенты уравнения линии (y = ax + b) в виде [a, b].
2. Рассчитывается ошибка, суммируя квадратичные разности между фактическими значениями y точек и значениями y, предсказанными линией наилучшего приближения.
3. Возвращается значение ошибки.

В итоге: алгоритм упрощает полилинию, выбирая точки, которые поддерживают минимальное расстояние (Lmin) между собой и дают ошибку (e) ниже максимального порога ошибки (emax) при аппроксимации линии к точкам.

Обнаружение замыкания траектории или Loop Closure Detection

Обнаружение замыкания траектории является важным этапом в оптимизации графа поз для оценки траектории робота. Этот процесс позволяет определить, когда робот возвращается в ранее посещенное место, что может значительно улучшить точность оценки траектории [5].

Обнаружение замыканий работает следующим образом:

1. Рассматриваются пары поз робота, которые близки по евклидовому расстоянию. Предполагается, что если две позы находятся достаточно близко друг к другу, то существует вероятность, что они представляют одно и то же место на траектории робота.
2. Для каждой пары близких поз анализируются их локальные окрестности. Локальная окрестность каждой позы включает соседние позы на траектории.
3. Рассчитывается сходство локальных окрестностей для каждой пары поз (1). Сходство определяется путем сравнения евклидовых расстояний между соседними позами в каждой окрестности. Сходство рассчитывается по следующей формуле:

$$S = \Sigma \, | d(A_i, B_j) - d(A_i, A_k) |, \qquad (1)$$

где S – сходство, d(A_i, B_j) – евклидово расстояние между позой A_i и позой B_j, d(A_i, A_k) – евклидово расстояние между позой A_i и позой A_k, а сумма берется по всем соседним позам в локальных окрестностях.

4. Если сходство между локальными окрестностями пары поз меньше заданного порога сходства, то считается, что обнаружено замыкание траектории между этими позами.
5. Обнаруженные замыкания траектории затем используются для построения графа поз и оптимизации траектории робота (рис. 3).

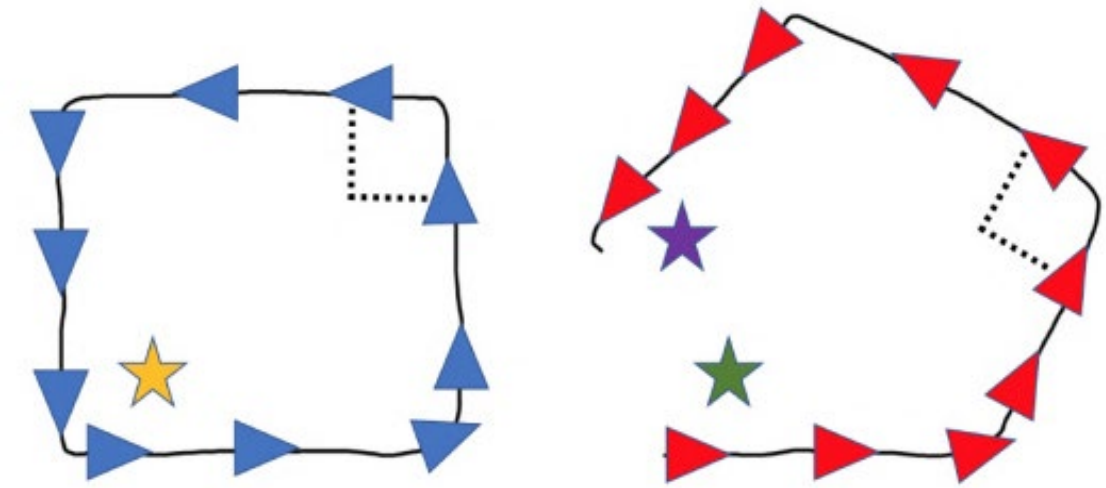

Рисунок 3 - Пример обнаружения замыкания траектории

Таким образом, обнаружение замыкания траектории играет важную роль в оценке траектории робота, позволяя определить моменты, когда робот возвращается в ранее посещенные места. Также, это помогает минимизировать влияние дрейфа в первоначальных координатах движения робота, которые были взяты с датчиков позиции на колесах.

Построение графа поз и его оптимизация

Построение графа поз и оптимизация графа поз являются следующими этапами в методе оценки траектории мобильного робота. Эти процессы позволяют роботу использовать информацию о его движении и обнаруженные замыкания траектории для улучшения точности оценки траектории [6].

Построение графа поз:

1. Граф поз представляет собой набор вершин и ребер. Вершины графа соответствуют позам робота в пространстве, а ребра - связям между соседними позами и обнаруженным замыканиям траектории.
2. Ребра графа могут быть двух типов: ребра одометрии и ребра замыкания траектории. Ребра одометрии соединяют последовательные позы робота, отражая информацию о его движении. Ребра замыкания траектории соединяют позы робота, которые были обнаружены как замыкания траектории.
3. Веса ребер графа представляют собой расстояния между соответствующими позами робота.

Оптимизация графа поз:

1. Оптимизация графа поз заключается в корректировке поз робота в графе с целью минимизации ошибок, связанных с накоплением ошибок одометрии и неправильным обнаружением замыкания траектории.
2. Оптимизация может быть выполнена с использованием различных алгоритмов, таких как алгоритм Левенберга-Марквардта, алгоритм Гаусса-Ньютона и алгоритм Gradient Descent.
3. В процессе оптимизации алгоритмы минимизируют функцию ошибки (2), выраженную в виде суммы квадратов разностей между измеренными и ожидаемыми значениями поз робота.

Минимизация функции ошибки позволяет устранить или снизить влияние ошибок одометрии и обнаружения замыкания траектории на оценку траектории робота.

Обозначим позы робота как $X = \{x_1, x_2, ..., x_n\}$, где n - количество поз в графе. Функция ошибки E определяется следующим образом:

$$E(X) = \sum_k (||\Delta x_{k+1} - x_k||^2 + ||\Delta x\_lc - x\_lc||^2), \quad (2)$$

где:

$\Delta x_{k+1} - x_k$ – представляет разность между позами x_{k+1} и x_k, соединенными одометрическим ребром;

$\Delta x\_lc - x\_lc$ – представляет разность между позами, соединенными ребром замыкания траектории (loop closure);

$||\cdot||$ – обозначает Евклидову норму;

$\sum_k$ – означает суммирование по всем ребрам графа.

Цель оптимизации графа поз состоит в том, чтобы найти набор поз X* (3), который минимизирует функцию ошибки E(X):

$$X^* = \text{argmin}(E(X)). \quad (3)$$

4. В нашем случае используется алгоритм Левенберга-Марквардта (LM). В алгоритме LM используется линейная аппроксимация функции ошибки и итеративный подход для поиска оптимума (4):

$$\Delta X = -(J^T J + \lambda I)^{-1} J^T \Delta F, \quad (4)$$

где:

ΔX – вектор изменений поз;

J – матрица Якоби, которая представляет частные производные функции ошибки по позам;

ΔF – вектор разностей между измеренными и ожидаемыми значения поз;

λ – параметр доверия, который регулирует величину шага оптимизации;

I – единичная матрица.

Алгоритм LM повторяет итерации, пока не будет достигнута сходимость или максимальное число итераций. После завершения оптимизации получается набор оптимизированных поз X*, который представляет улучшенную оценку траектории робота (рис. 4)

5. После оптимизации графа поз получается улучшенная оценка траектории робота, которая более точно отражает его движение в пространстве.

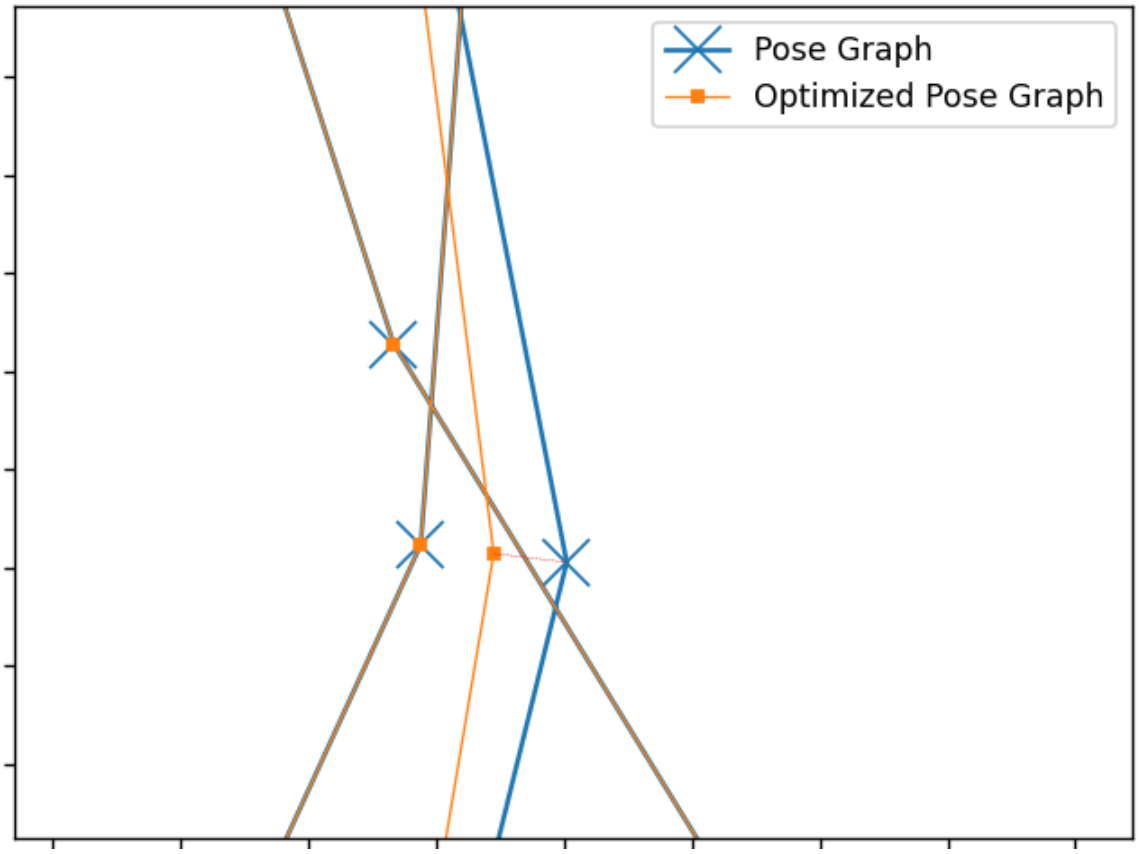

Рисунок 4 - Результат оптимизации графа поз

Вывод. В этом разделе представлен алгоритм, который упрощает линию траектории движения робота, обнаруживает замыкания траектории, которые необходимо выделить, поскольку робот сделал по помещению несколько кругов, строит оптимизированный граф на основе прошлых преобразований. Этот граф и будет являться конечной траекторией робота. В дальнейшем будут произведены эксперименты на данных, приближенных к реальным условиям эксплуатации.

Моделирование работы алгоритма реконструкции комнаты

Тестирование производилось на данных, полученных при моделировании движения робота-пылесоса по стенам сложного помещения (рис. 5 – 11)

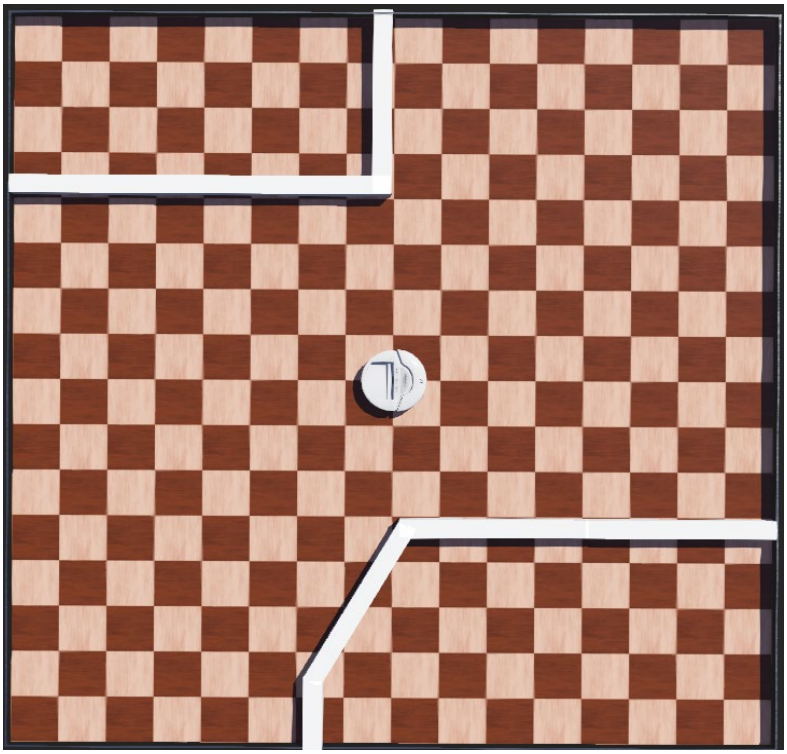

Рисунок 5 - Усложненное помещение 2

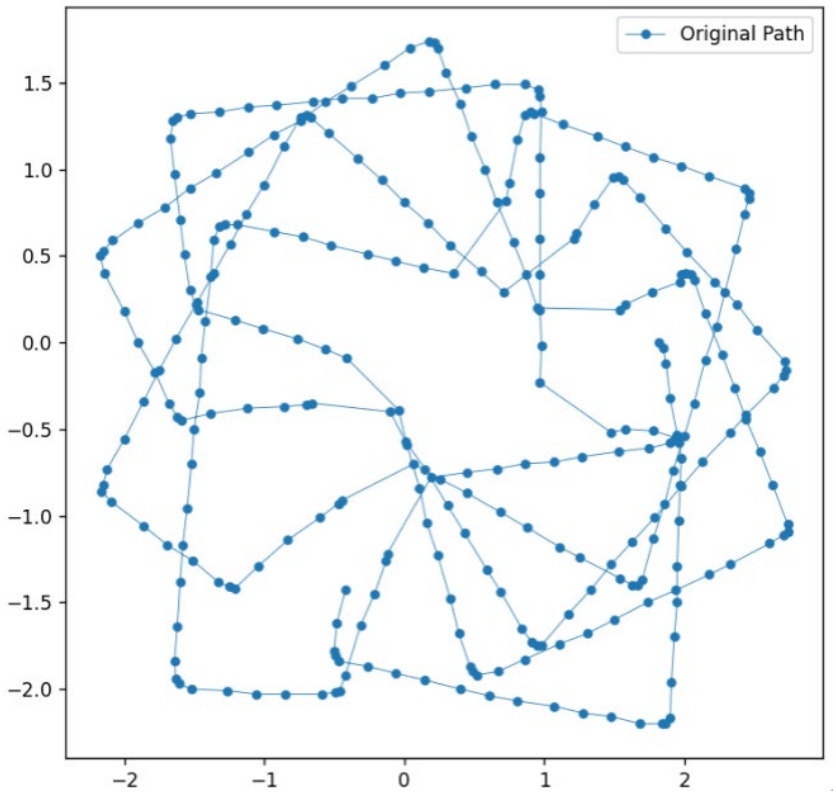

Рисунок 6 - Данные одометрии колес робота, усложненное помещение 2

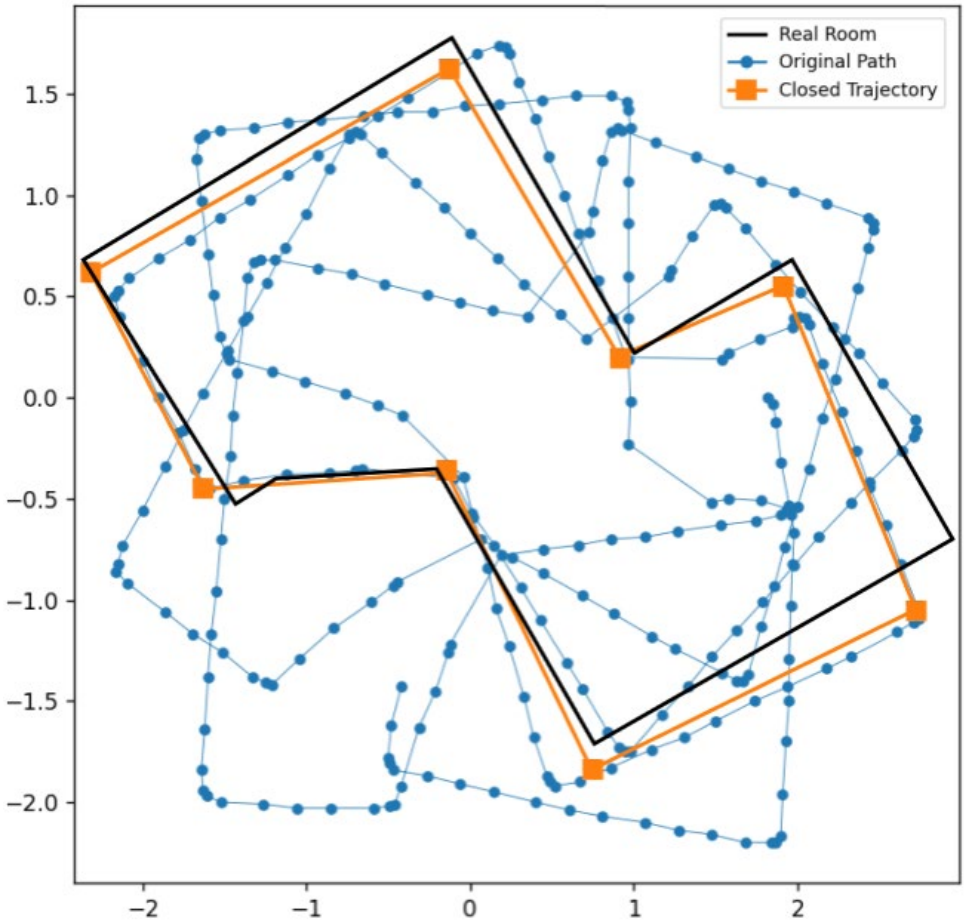

Рисунок 7 - Сравнение результатов одометрии, построения карты и реального усложненного помещения 2

На рисунке 8 видно, что данный алгоритма сбора данных колес робота не может обработать мелкие детали в геометрии комнаты

Рисунок 8 - Недостатки работы алгоритма сбора данных одометрии колес

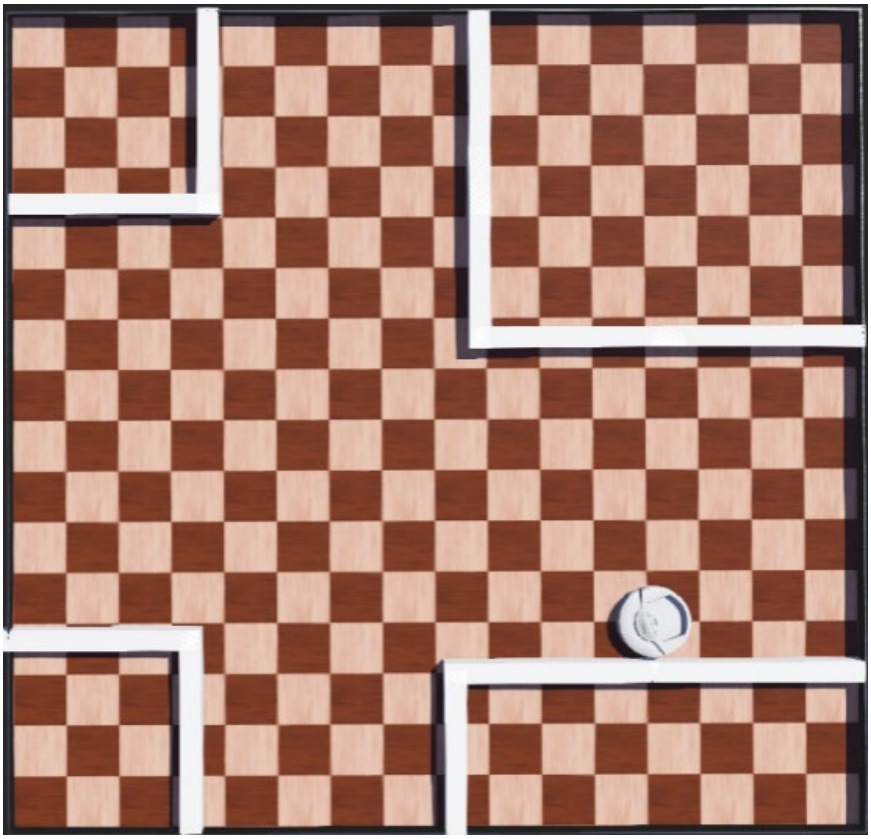

Рисунок 9 - Усложненное помещение 3

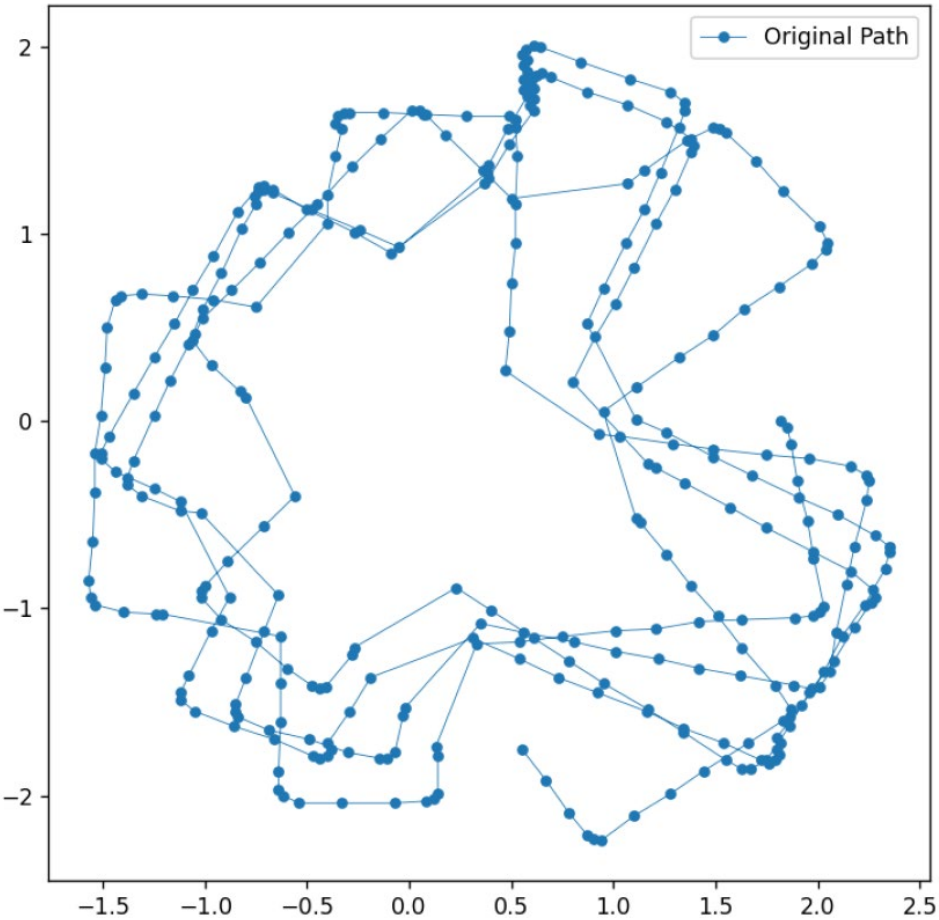

Рисунок 10 - Данные одометрии колес робота, усложненное помещение 3

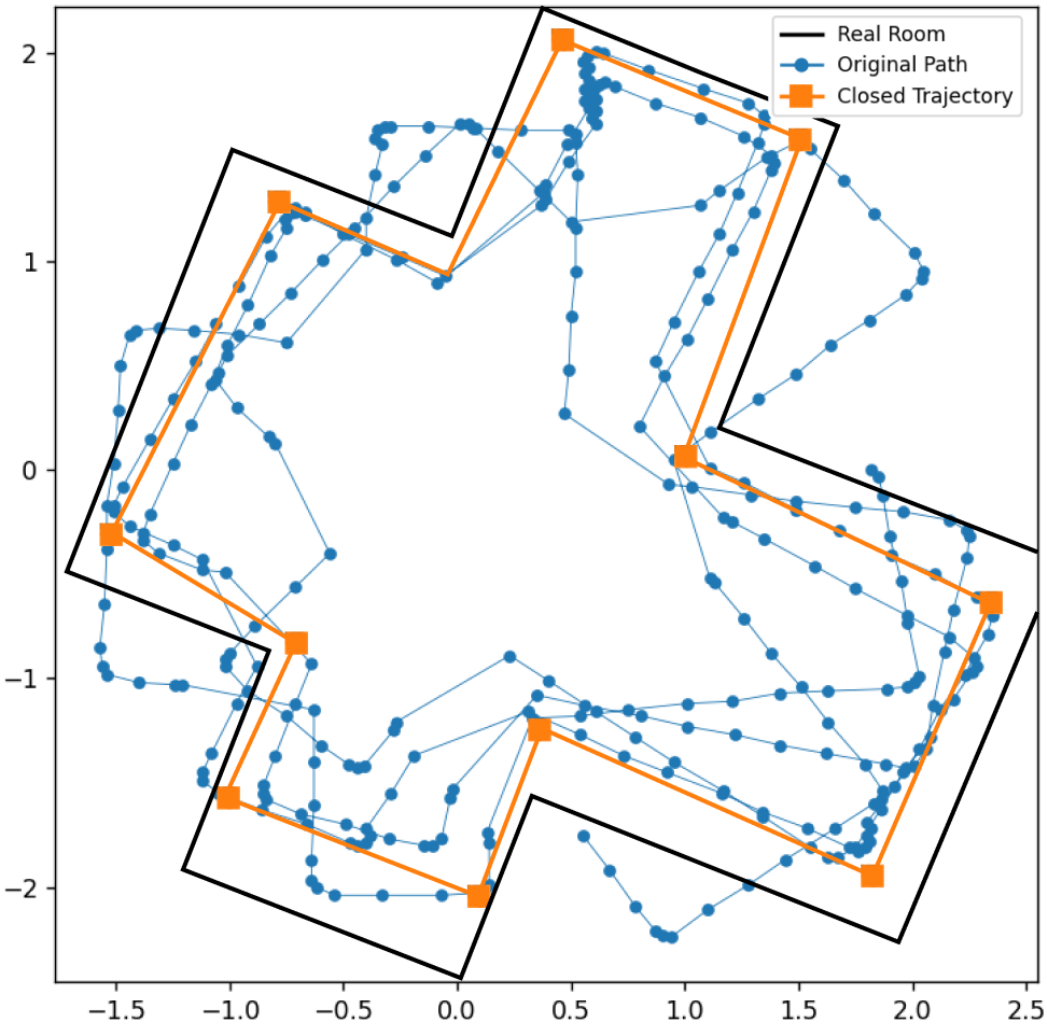

Рисунок 11 - Сравнение результатов одометрии, построения карты и реального усложненного помещения 3

Вывод. При тестировании алгоритмов локализации робота на усложненных пространствах выявляется ряд недочетов. Некоторые из них – это отсутствие обработки проскальзывания колес при движении по стенам и столкновениями с ними, неточности в масштабе реконструированных помещений, которые появляются при усложнении тестируемого помещения. Моделирование алгоритмов реконструкции контура помещения, и алгоритма сбора данных одометрии на ранних стадиях разработки и их тестирование позволило выявить соответствующие недочеты.

Заключение

Разработанная система моделирования алгоритмов позволяет выявить недочеты разрабатываемых алгоритмов локализации робота и картографии на ранних стадиях разработки. После успешного тестирования и доработки алгоритмов, разработчики смогут продолжить работу над решением проблемы ориентации автономного устройства в пространстве.

После определения контура помещения можно перейти к следующему этапу - построению полигона помещения. Этот полигон представляет собой детализированную карту, которую робот может использовать для навигации внутри помещения. Карта может включать в себя информацию о различных препятствиях, таких как стены, мебель или другие объекты, которые робот должен избегать. Она также может включать информацию о "зонах интереса", таких как точки зарядки для робота или места, где он должен выполнять определенные задачи.

Результаты работы разработанных алгоритмов сбора данных одометрии колес робота и реконструкции контура помещения могут быть использованы в дальнейшем моделировании и тестировании различных автономных устройств:

Роботы-пылесосы: Эти алгоритмы могут быть использованы в роботах-пылесосах для более эффективного покрытия всей доступной площади. После того как контур помещения будет определен, робот сможет эффективно распределять свои ресурсы и время для уборки всего пространства.

Роботы для инвентаризации складов: на больших складах роботы могут использовать эти алгоритмы для картографирования пространства и оптимизации маршрутов.

Роботы-курьеры: для доставки товаров или писем внутри больших зданий, роботы могут использовать эти алгоритмы для определения наиболее эффективных путей.

С использованием этой карты робот может определить наиболее эффективный путь до любой точки в помещении, избегая препятствий и минимизируя время, необходимое для выполнения его задач. Это делает использование робота более эффективным и позволяет сэкономить время и ресурсы.

Список литературы:

1. Pedrosa, E., L. Reis, C. M. D. Silva and H. S. Ferreira. Autonomous Navigation with Simultaneous Localization and Mapping in/outdoor. 2020.
2. RTAB-Map, Real-Time Appearance-Based Mapping [Электронный ресурс] URL: http://introlab.github.io/rtabmap/, свободный ресурс.
3. Labbé, M, Michaud, F. RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. J Field Robotics. 2019.
4. Silva, B.M.F.D.; Xavier, R.S.; Gonçalves, L.M.G. Mapping and Navigation for Indoor Robots under ROS: An Experimental Analysis. Preprints 2019.
5. Mathieu Labbé and François Michaud. Online Global Loop Closure Detection for LargeScale Multi-Session Graph-Based SLAM. 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems.
6. Ying Zhang, Juan Liu, Gabriel Hoffmann, Mark Quilling, Kenneth Payne, Prasanta Bose, and Andrew Zimdars. Real-time indoor mapping for mobile robots with limited sensing. In Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on, pages 636–641. IEEE, 2010.
7. Giorgio Grisetti, Rainer Kummerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based slam. IEEE Intelligent Transportation Systems Magazine, 2(4):31–43, 2010.