


МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по научно-исследовательской деятельности *2*
аспиранта

Аспирант

направление 09.06.01

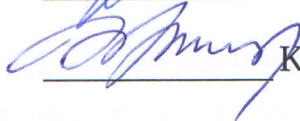
специальность 05.13.11



Филатов Антон Юрьевич

Руководитель

к.т.н., доцент



Кринкин Кирилл Владимирович

Санкт-Петербург

2019

Цель работы

Изучить различия в существующих алгоритмах SLAM, а также сравнить некоторые популярные решения с целью выяснить, какие из них наиболее подходят для мобильных роботов, работающих в помещениях.

1.1 Постановка и формальное описание задачи SLAM

Прежде, чем говорить о многоагентных системах необходимо проклассифицировать существующие одноагентные подходы и определить, возможно ли масштабирование этих подходов и сколько это будет стоить.

Задача SLAM — это задача, которая ставится перед движущимися платформами, которым необходимо исследовать неизвестное окружение и построить карту местности. Необходимо отметить, что цели построить маршрут, позволяющий исследовать окружение, не ставится. Для решения задачи достаточно, чтобы платформа двигалась по какому-либо маршруту и имела возможность снимать измерения окружения.

Очевидно, что задача SLAM состоит из двух задач: построение карты, если в каждый момент времени известно положение агента, а также состояние окружающей среды; и локализация, то есть определение местоположения, если известна карта. Из формулировки этих подзадач понятно, что для успешного решения одной задачи необходимо, чтобы сначала была решена другая задача. Именно поэтому краеугольным камнем становится их одновременное решение, и поиск робастного алгоритма ведётся уже больше двадцати лет.

Данная задача актуальна для мобильных агентов в абсолютно разных сферах жизни: начиная от роботов-пылесосов, и заканчивая автопилотируемыми автомобилями, роботами-спасателями или даже марсоходами

Кроме того необходимо учитывать, что автономные мобильные агенты не обладают пока выдающимися вычислительными мощностями. Это накладывает ограничение на сложность алгоритмов решения поставленной задачи. Также существует класс сложных алгоритмов, которые должны выполняться на серверах, в то время как мобильные агенты выполняют лишь роль наблюдателей за окружением. В последнем случае необходимо следить за качеством связи между всеми узлами системы, что приводит к снижению устойчивости и автономности системы.

Дополнительную сложность в рассматриваемую задачу вносит тот факт, что часто нет возможности воспользоваться такими сильными инструментами, как GPS. Например, это невозможно в помещении, где погрешность GPS сопоставима с шириной коридора. Таким образом мобильный агент может полагаться только на датчики, вроде камеры, лазерного дальномера, гироскопа или сонара.

1.2 Классификация алгоритмов решения задачи SLAM

Объём 2 страницы

Исходя из различных начальных условий, появлялись разные подходы к решению задачи SLAM. Можно провести такую классификацию решений.

По размерности наблюдений

- Трёхмерные. Если оснастить агента видеокамерой, то он полно наблюдает окружающее трёхмерное пространство, и построенная им карта также будет иметь размерность равную трём.

- Двумерные. Часто для упрощения обработки входных данных или по иным причинам используются лазерные дальномеры или сонары, которые проводят измерения в плоскости. Построенная при использовании таких датчиков карта представляет собой план окружения.

По типу используемых сенсоров

Обычно этот способ классификации напрямую связан с предыдущим, однако, есть и некоторые исключения

- визуальный

В качестве сенсоров используется видеокамера. Причём это может быть как одна камера, так и набор из нескольких, а также так называемая «всенаправленная» камера, которая снимает за один раз всё вокруг на 360°.

- лазерный

В качестве сенсоров используется лазерный дальномер — устройство, которое измеряет расстояние во всех направлениях с шагом дискретизации 0.5-1°.

Обычно визуальный алгоритм строит трёхмерную карту, а лазерный — двумерную, однако, эта зависимость не является обязательной. Так, например, возможно строить двумерную карту, используя видео камеру, более того, классические алгоритмы, стоящие у истоков развития задачи SLAM работали именно так: по набору видео изображений строилась двумерная карта препятствий.

По способу обработки входных измерений

- основанный на выделении особых точек

Часто нет необходимости строить полную карту местности, а достаточно ограничиться некоторым набором ориентиров. Такие ориентиры должны быть выделены из данных сенсоров и использованы для построения карты. Исторически именно такой тип алгоритмов был разработан первым.

- использующий «сырые» измерения

Гораздо более полную карту можно получить, если использовать все данные, полученные от сенсоров. Будь то видеокادر или лазерный скан, если ис-

пользовать его полностью, то информации для построения карты становится больше, и, следовательно, карта становится намного точнее.

По способу представления карты

- использующий сетку занятости

Карта в алгоритме SLAM — это один из результатов работы алгоритма, поэтому вопрос о её структуре стоит наиболее остро. Очевидным выглядит решение представить карту в виде массива ячеек (двумерного или трёхмерного), где каждая ячейка содержит вероятность быть занятой. Так получается, что всё пространство разделяется на небольшие области, каждая из которых либо занята каким-то препятствием, либо свободна.

- графовый

Современные подходы всё чаще используют другой способ представления карты. Графовый подход предполагает, что карта распределена по узлам графа. В каждом узле находится одно или несколько измерений, а рёбра графа содержат в себе информацию о перемещении агента, которое необходимо совершить, чтобы начать наблюдать измерения из соответствующего узла.

По характеру изменения окружающей среды

- статический

Такой тип алгоритмов предполагает, что окружение не изменяется со временем, и если агент полностью изучил какую-то комнату и покинул её, а через некоторое время вернулся обратно, то он обнаружит, что ничего в этой комнате не изменилось. Такой тип алгоритмов может быть применён в офисных помещениях или складах.

- динамический

В случае, если SLAM алгоритм применяется за пределами зданий, например, в городе, где присутствует интенсивное движение, появляется необходимость уметь корректировать карту с учётом перемещающихся препятствий.

1.3 SLAM, основанные на выделении особых точек.

1.3.1 Общий подход

Объём: 2 страницы

Исторически первыми появились алгоритмы, которые выделяли особые точки на наблюдаемом окружении и строили карту, состоящую из таких ориентиров. Идея очень близка к человеческому восприятию мира. Ориентируясь в незнакомом городе, человек отмечает для себя, например, высокие башни и определяет приблизительное расстояние между ними, а дальше использует их, как ориентир.

Для автономных вычислительных агентов можно реализовать подобный алгоритм: необходимо выделять на снятых измерениях «особые» точки, из которых и будет в будущем состоять карта. Чем больше таких точек будет определено в процессе работы, тем точнее будет результирующая карта и позиция агента. Также на точность влияет количество позиций, с которых удалось различить «особую» точку. Поэтому для таких алгоритмов вопрос о робастном выделении особых точек стоит очень остро.

Работа алгоритма разделена на несколько этапов, в числе которых можно выделить снятие измерений (скан, кадр и пр.), скан матчинг (сопоставление измерения и карты) и обновление карты. Схема этапов представлена на рисунке 1.

<рисунок1>

На очередном шаге алгоритма входными данными являются построенная к текущему моменту карта, гипотеза о положении агента, а также очередное наблюдение. Первым делом необходимо выделить особые точки или ориентиры. Затем после такой предобработки запускается компонент, который носит название *scan matcher*, задача которого сопоставить текущее наблюдение с уже построенной картой, чтобы определить текущее положение агента. После того, как позиция оценена появляется возможность обновить карту, если это необходимо.

В рассматриваемых алгоритмах карта состоит из набора ориентиров (их координат относительно точки, где алгоритм начал свою работу). Построенные координаты ориентиров зависят от точки начала, однако расстояние и взаимное расположение между ними остаётся неизменным. Именно этот принцип лежит в основе такого рода алгоритмов. Если составить матрицу ковариаций ориентиров, то появится возможность применить алгоритм фильтрации Калмана для обновления карты.

1.3.2 EKF SLAM

Первая идея решения задачи SLAM, предложенная [R.Smith,M.Self,andP. Cheeseman] в 1990 году, опиралась на расширенный фильтр калмана. Идея в том, что все препятствия должны коррелировать между собой, то есть (если агент работает в неподвижном окружении) взаимное расположение препятствий не должно меняться. А это значит, что, пронаблюдав пару ориентиров под разными углами, можно повысить точность оценки их расположения.

Более подробно алгоритм выглядит следующим образом. Алгоритм состоит из множества итераций, каждая из которых включает шаги, описанные ниже.

Шаг 1. Агент производит очередное наблюдение, выделяет особые точки. В соответствии с текущей априорной оценкой позиции агента исходя из расположения ориентиров относительно агента вычисляется расположение ориентиров в глобальных координатах.

Шаг 2. Координаты найденных ориентиров сопоставляются с координатами ориентиров, которые были включены в карту на предыдущих шагах. Этот шаг позволит уточнить позицию ориентиров, которые уже наблюдались ранее, либо изменить дисперсию погрешности расположения ориентиров. Если найденный ориентир отсутствует в карте, то он добавляется в карту.

Шаг 3. Исходя из уточнённой карты (в том числе уточнённых позиций наблюдаемых ориентиров) строится апостериорная оценка позиции агента.

Необходимо заметить, что априорная оценка на шаге 1 строится исходя из апостериорной оценки, построенной на шаге 3 на предыдущей итерации, а также исходя из данных одометрии — сведений о перемещении агента, измеренных датчиками перемещения (например, гироскопом).

Однако следует заметить, что этот алгоритм непригоден для масштабирования. Каждый новый ориентир, будучи включённым в карту, увеличивает размерность матрицы ковариаций ориентиров. Следует напомнить, что карта с точки зрения рассматриваемого алгоритма — это вектор ориентиров, содержащий их координаты. Матрица ковариаций ориентиров «связывает» ориентиры между собой. Это нужно для того, чтобы в случае уточнения координат одного из препятствий можно было также уточнить координаты остальных ориентиров. Эта ситуация имеет очевидный аналог с человеком, ориентирующимся в незнакомом городе. Допустим, этот человек выяснил, что два высоких здания располагаются друг относительно друга через дорогу. Затем, спустя какое-то время он обнаружил третье здание неподалёку от одного из высоких, однако второе высокое здание он в данный момент не видит. Тем не менее это не мешает ему представить, как располагаются друг относительно друга все три здания.

Такой подход, однако, начинает требовать больших вычислительных затрат, как только ориентиров набирается значительное количество. Помимо того, что растёт вектор ориентиров, так к тому же при добавлении нового элемента в вектор, необходимо вычислять ковариацию нового препятствия со всеми предыдущими. Кроме того специфика алгоритма, основанного на фильтре Калмана требует вычислять матрицу, обратную к матрице ковариаций, что влечёт за собой увеличение времени работы алгоритма с квадратичной скоростью.

Современные сенсоры позволяют снимать наблюдения окружающей среды с частотой не менее 30 Гц, а, следовательно, алгоритм должен успевать обрабатывать не менее 30 итераций в секунду. Именно поэтому применение EKF SLAM в современных системах практически исчезло, а ему на замену пришли алгоритмы, основанные на более сложных идеях или допущениях, но менее требовательные к вычислительным ресурсам.

1.3.3 FastSLAM

Объём: 1 страница

Предложенная [Thrun и компания] идея продолжает концепцию EKF SLAM, но увеличивает производительность за счёт отсутствия корреляции между препятствиями.

Другими словами общий подход к алгоритму остаётся практически таким же, каким он был для алгоритма EKF SLAM, однако считается, что каждый ориентир не зависит от всех остальных.

Возвращаясь к примеру с человеком, ориентирующимся в городе, такой алгоритм позволит ему построить на своей карте первые два здания, а третье здание будет стоять возле первого с некоторой погрешностью. То есть взаимное расположение первого и третьего зданий будет «верным» - между ними будет правильное расстояние, однако о взаимном расположении второго и третьего здания алгоритм FastSLAM не заботится.

Конечно, такая постановка вопроса порождает неоднозначность карты. Разрешить её можно единственным образом — найти точку, с которой можно будет пронаблюдать здания два и три одновременно. Если же такая точка не найдена, остаётся лишь надеяться, что человек не ошибся, когда строил траекторию между точками, в которых он производил наблюдения. Графически последовательность наблюдений и возможная результирующая карта представлены на рисунке 2.

Чтобы разрешить неопределённость часто прибегают к аппарату, называемого фильтром частиц. Подробно он будет рассмотрен в следующей главе, здесь же он будет описан кратко. На этапе определения положения выбирается не одна наиболее вероятная гипотеза, а несколько, например, три. В этом случае будет существовать три возможные карты и каждое новое наблюдение необходимо будет сопоставлять с тремя картами до тех пор, пока какая-то из гипотез не будет отброшена, поскольку новые наблюдения перестанут каким бы то ни было образом соотноситься с картой.

Тесты, проведенные авторами в исходной статье, показывают, что точность подхода FastSALM, несмотря на существенное пространство для неопределённости остаётся высокой. И действительно, алгоритм FastSLAM — это один из базовых на сегодняшний день алгоритмов.

1.4.1 Байесовская теория

Объём 3 страницы

Выше было показано, что идея выделения особых точек на кадре накладывает значительные ограничения на структуру карты и на алгоритм в целом. Существует альтернативный метод представления карты, который позволяет хранить модель карты в динамически изменяющемся массиве. В простом случае речь идёт о двумерном массиве, каждый элемент которого - это “клетка”, соответствующая области пространства. Другими словами карта представляет собой двумерный план окружающей среды, разделённый на ячейки. Каждая ячейка такой карты содержит число - вероятность быть занятой. Пример карты с ячейкой размером 10 x 10 см показан на рисунке 3.

По рисунку понятно, что при достаточно маленьком размере ячеек, такая карта неотличима от обыденного плана окружения, а также содержит больше информации, чем набор ориентиров.

Лучше всего такой формат карты подходит для лазерных SLAM алгоритмов, когда в качестве источника наблюдений выступает лазерный дальномер. Если визуализировать лазерный скан имеет, то окажется, что он имеет похожую на карту структуру, и, “накладывая” скан на карту, как показано на рисунке 4, можно определить текущую позицию агента.

Для SLAM алгоритмов, которые основаны на карте, состоящей из ячеек, наибольшую популярность имеет подход к вычислению позиции агента, основанной на Байесовской теории. Другими словами, основываясь на построенной карте m , последнем известном перемещении агента u , текущем наблюдении z , а также последней известной позиции агента x можно вычислить вероятность текущего положение агента X , при котором возможно произвести именно такое наблюдение $p(X | m, u, z, x)$. Задача алгоритма в том, чтобы вычислить X , вероятность которой максимальна

Байесовская теория гласит, что поиск такой вероятности пропорционален произведению следующих вероятностей:

$$p(X|m,u,z,x) \propto p(z|m,X) * p(X|x,u) \quad (1)$$

Первый множитель - это вероятность наблюдать данное измерение, при условии нахождения в позиции X в итеративно построенной карте m . Второй

множитель - это вероятность оказаться в позиции X , если подействовать управляющим воздействием u на предыдущую известную позицию x .

Остановимся подробнее на втором множителе. Входное воздействие - обычно это управляющий сигнал или сведения, полученные от некоторого датчика, о том, на сколько агент передвинулся. Например, команда двигаться с максимальной скоростью вперёд в течение одной секунды. Если посмотреть на такую команду обывательским взглядом, то кажется, что, зная предыдущую позицию x , найти теперь новую позицию агента X не составляет труда, однако здесь также есть условности. Агент, представляет собой тележку на колёсах. Находясь в идеальных условиях с максимальных коэффициентом трения, а также возможностью мгновенно набирать постоянную скорость, действительно достаточно провести элементарные математические вычисления, чтобы найти $X = x + u$. Однако на практике агенту необходимо время, чтобы разогнаться, затормозить, колёса могут прокручиваться из-за плохого сцепления с дорогой. Обычно позиции $X = x + u$ выбирают, как начальную оценку позиции агента и называют одометрией. Понятно, что настоящая позиция агента находится в некоторой окрестности одометрии.

Подробно опишем первый множитель. Он показывает вероятность того, что наблюдение z доступно с позиции X на карте m . Понятно, что есть позиция X является “истинной”, то такая вероятность будет равна 1. И должна быть близкой к нулю, если позиция неверна. Именно для этих целей выбирается карта в виде ячеек занятости. Накладывая скан (который является наблюдением Z) на ячейки карты m , можно оценить, как много точек лазерного скана попало в “занятые” ячейки карты.

Поиском позиции X , для которой вероятность $p(z|m,X)$ максимальна, занимается компонент алгоритма SLAM, называемый скан матчер.

Если карта является фрактальной, то может существовать несколько позиций, для которых вероятность $p(z|m,x)$ одинаково высока. В этом случае важную роль играет второй множитель из уравнения (1), который ограничивает область поиска позиции X .

После того, как наиболее вероятная позиция агента найдена, необходимо обновить карту в случае, если скан z конфликтует с картой или может её дополнить. Случай конфликта - это наиболее неприятная ситуация, поскольку в простом случае нет инструмента, чтобы проверить, произошла ли ошибка в определении позиции X на текущем шаге или на одном из предыдущих (случай динамического окружения не рассматривается). Наиболее популярным решением является полагаться на более новые данные и отбрасывать “старые” вероятности в конфликтующих ячейках карты.

Заключение

В ходе научно-исследовательской работы была изучена задача SLAM, а также было проведено сравнение нескольких популярных подходов к реализации процесса скан матчинга — основного компонента различия SLAM-ов. В результате было получено, что наиболее подходящим для мобильных роботов алгоритмом скан матчера является алгоритм Монте-Карло. Алгоритм, реализующий такой скан матчер называется `tinySLAM`, его работа была протестирована на записанных последовательностях входных данных, на которых было подтверждено, что этот алгоритм выполняется точно.

Список литературы

1. Past, present, and future of simultaneous localization and mapping: Toward the robust perception age / C. Cadena, L. Carlone, H. Carrillo, et al. // IEEE Transactions on Robotics. - 2016. - Vol. 32. - P. 1309–1332.
2. A comparison of slam algorithms based on a graph of relations / W. Burgard, C. Stachniss, G. Grisetti, et al. // 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. - 2009. - Vol 41. - P. 2089–2095.
3. B. Gerkey, Ros slam gmapping [Electronic resource] / B. Gerkey // <http://wiki.ros.org/slam-gmapping>. [Accessed 15-Jan-2018].
4. Google, 2d cartographer backpack deutsches museum / Google // <https://github.com/googlecartographer>
5. The mit stata center dataset. / M. Fallon, H. Johannsson, M. Kaess, J. J. Leonard. // The International Journal of Robotics Research. - 2013. - Vol 32. - P. 1695–1699.