

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

ОТЧЕТ  
по научно-квалификационной работе 2  
аспиранта

Аспирант

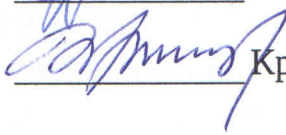
направление 09.06.01

специальность 05.13.11



Филатов Антон Юрьевич

Руководитель к.т.н., доцент



Кринкин Кирилл Владимирович

Санкт-Петербург

2019

## **Цель работы**

Описать существующие многоагентные подходы к решению задачи SLAM, выделить высокоуровневую схему алгоритмов, а также предложить алгоритм, подходящий для использования на низкопроизводительных платформах.

## **Высокоуровневая схема многоагентного SLAM**

### **Предположения и допущения**

Высокоуровневая схема многоагентного SLAM обязательно включает в себя одноагентный SLAM, однако не обязательно, чтобы каждый агент одинаково выполнял одноагентный алгоритм. Можно выделить различные архитектурные подходы, описывающие иерархию агентов в многоагентном SLAM.

- звезда с сервером
- звезда с маршрутизатором
- распределённая сеть

Прежде, чем подробно описывать различные архитектуры, следует остановиться на вопросе об определении взаимного расположения агентов. В простом случае агенты начинают работу, находясь в одной точке или, хотя бы, на известном расстоянии друг от друга (и с известным изначальным направлением). В этом случае неважно, когда и как данные от нескольких агентов будут объединены, поскольку в любой момент известна траектория движения агента и начальная точка. Следовательно для любого агента всегда известно его положение в глобальной системе координат.

Когда известно положение каждого агента, пусть даже с некоторой погрешностью, задача объединения наблюдений каждого агента становится тривиальной, с ней может справиться скан матчер, используемый для решения задачи одноагентного SLAM.

Следует заметить, однако, что невозможно расположить агентов абсолютно в конкретной точке или на абсолютно точно измеренном расстоянии между ними, если речь не идёт о симуляции. Поэтому взаимное расположение агентов обязательно содержит погрешность. Величина этой погрешности зависит от множества условий, но само наличие такой погрешности наводит на мысль, что алгоритм, который может обработать входные данные, содержащие случайную погрешность, может также справиться с задачей, где эта погрешность введена намеренно.

Другими словами, если известно только приблизительное расположение агентов, например, известно, что они находятся в радиусе 10 сантиметров друг от-

носителем друга, возможно совместить данные наблюдений, которые они предоставляют.

Таким образом в задаче многоагентного SLAM появляется дополнительная задача о определении взаимного расположения между агентами. И для решения этой задачи различные авторы используют решение, основанные на разных допущениях.

Так, например, одно из решений - это оснастить мобильных агентов камерой с углом обзора в 360°, чтобы они не имели слепых зон, а также яркими хорошо различимыми метками, например, светодиодами. Тогда при определении на камере таких светодиодов будет возможно определить направление, в котором агентам необходимо двигаться для того, чтобы встретиться в одной точке. Кроме того при помощи дальномера или любого другого инструмента можно определить расстояние между агентами, и этих двух параметров (взаимный угол и расстояние) достаточно, чтобы определить координаты одного агента в координатах другого и, следовательно, приступить к задаче объединения данных.

К недостаткам такого подхода можно отнести как раз сложность установки камеры и ярких меток на агента. В лаборатории возможно соблюсти все условия, чтобы метки всегда были различимы и камера могла их распознавать, однако на практике такой метод обнаружения зависит от освещения, заряда питающей батареи и прочих других факторов.

Другим способом определения взаимного расположения является разметка окружения. Она может быть выполнена различными способами: специальные “зоны коммуникации”, QR-коды, нанесённые на пол и определяющие координаты, разметка стен или потолка - все способы разметки окружения позволяют агентам определить свои координаты в глобальной системе. Как только несколько агентов определяют собственные координаты, появляется возможность объединять их данные.

Описанные выше способы требуют некоторой подготовки агентов или окружения, однако существуют подходы, которые не требуют дополнительных данных, а основываются на данных сенсоров, используемых для решения задачи SLAM. Ниже описана структура такого подхода. Для начала требуется определить, что один из агентов посетил область, в которой побывал другой агент. Затем текущее наблюдение первого сопоставляется с картой, построенной вторым агентом. Этот этап можно назвать задачей глобальной локализации. Другими словами на полной карте необходимо без каких бы то ни было дополнительных сведений определить местонахождение, соответствующее предоставленному наблюдению. Если окружение не является фрактальным, то такое местоположение можно обнаружить существующими методами. Найденное местоположение

и будет являться позицией первого агента в координатах второго, а следовательно, агенты могут приступать к этапу объединения накопленных данных.

## **Архитектура многоагентного SLAM**

### **Звезда с сервером.**

Исторически первая архитектура многоагентного алгоритма состояла из сервера, который собирал данные от агентов и самостоятельно обрабатывал их. Другими словами единственную роль, которую играли агенты - это роль сенсоров. Кроме того, сервер мог вовсе не предоставлять обратной связи агентам. Достоинства такой структуры очевидны: даже если начальное взаимное расположение агентов неизвестно, сервер обладает знаниями о том, что “видит” каждый из наблюдателей в каждый момент времени. То есть сервер решает задачу одиночного SLAM, имея гораздо больше сенсоров. Недостаток такой структуры невозможно игнорировать: система очень зависима от качества связи с сервером. Кроме того, в случае, если сервер по каким-то причинам выйдет из строя, мобильные агенты будут не в состоянии выполнять хоть какие-то шаги алгоритма SLAM, помимо наблюдения за окружением.

В предыдущем разделе поднимался вопрос о способе вычисления взаимного расположения агентов. В том числе рассматривался подход, когда агенты снабжаются дополнительной камерой и яркими метками. В рамках рассматриваемой архитектуры необходимо, чтобы агенты передавали данные с этой камеры на сервер для выполнения вычисления взаимного расположения.

Другими словами сервер выполняет роль единственного вычислительного узла. Такая архитектура значительно удешевляет агентов - они играют роль сенсоров и не должны выполнять никаких вычислительных операций. Таким образом система свободно масштабируется: для увеличения количества агентов в сети достаточно приобрести дешёвую платформу, оснастить датчиками и средством передачи информации.

### **Звезда с маршрутизатором.**

Звезда с маршрутизатором является модификацией архитектуры, описанной выше. Отличительной особенностью является распределение вычислений между агентами. Другими словами каждый агент решает собственную задачу SLAM, но связь с другими агентами осуществляется через центральный узел.

При таком подходе недостаток звезды с сервером немного видоизменяется: с одной стороны “узкое горлышко” остаётся в виде маршрутизатора, однако застраховаться от выхода его из строя можно, установив резервный маршрути-

затор. В системе с сервером в центре звезды ставить дублирующий центральный узел часто не рентабельно, а в данной архитектуре установка резервного оборудования целесообразна.

Необходимо отметить, что в этом случае каждый агент должен обладать достаточными вычислительными мощностями, чтобы снимать наблюдения, решать задачу SLAM и обмениваться данными с маршрутизатором. Плюсом такой системы является децентрализация вычислений, и, следовательно, упрощение масштабирования.

#### Распределённая сеть.

Следующим этапом децентрализации вычислений является отказ от маршрутизатора. Система тогда выглядит следующим образом: агенты выполняют связь друг с другом только тогда, когда они находятся в непосредственной близости друг от друга. Это может достигаться, например, разбиением пространства на участки, где каждый агент становится ответственным за определённый участок пространства, и когда другой агент заходит на “чужую” территорию, он знает, с каким агентом ему следует связаться.

Другим способом организовать взаимодействие между агентами является общение при столкновении агентов. Как только один агент замечает другого посредством собственных датчиков, агенты могут сблизиться, если необходимо, и начать процесс идентификации друг друга и обмена информацией.

В любом случае система становится абсолютно стабильной к потере любого узла, поскольку связь между агентами лежит в области задач самих агентов. Если, например, один из них физически застрял, или повредил датчик, или израсходовал заряд батареи, остальные агенты смогут продолжать работу, даже без вышедшего из строя. Безусловно, выход из строя одного из узлов может повлиять на качество построенного результата, однако он не повлечёт остановку работы всей системы.

Также необходимо упомянуть задачи, которые встают дополнительно, помимо задачи одноагентного SLAM.

Самым важным является вопрос протокола общения между агентами. Если система содержит сервер, то ответ на этот вопрос тривиальный. Однако при использовании других архитектур к протоколу общения следует относиться внимательно. Самые популярные решения - это перед началом взаимодействия агентов либо переместиться в одну точку, либо оценить взаимное расположение друг друга. Затем каждый агент передаёт собственные накопленные данные другому агенту, и связь на этом заканчивается. Хотя возможна ситуация, когда агенты верифицируют объединённые карты.

Другим важным вопросом является способ объединения карт. Как только решён вопрос вычисления взаимного расположения агентов, необходимо соединить карты, построенные агентами. Сложность в этом вопросе состоит в том, что даже если карты построены без ошибок, они построены при помощи вероятностных подходов, а следовательно любая погрешность в измерении может привести к неполному соответствию построенных карт. Например, если рассмотреть многоагентный EKF SLAM, то можно представить ситуацию, когда оба агента пронаблюдали одно и то же препятствие, но один из них допустил погрешность в вычислении координат этого препятствия. Агенты не могут оценить, какая из карт является более точной и вынуждены либо выбрать одну из гипотез, либо строить среднее.

Из описанной выше проблемы возникает вопрос консенсуса карт. Если одна из карт содержит ошибку, или если каждый из агентов построил кардинально отличающиеся друг от друга карты, необходимо принимать решение, как будет выглядеть итоговая карта. Важно ответить на вопрос, будет ли она одна или каждый агент будет предоставлять его собственный вариант карты.

Более подробно эти задачи рассмотрены в разделе 2.4

## **Графовый многоагентный SLAM, `cg_mrslam`**

*Объём 2-3 страницы*

Среди одноагентных алгоритмов, решающих задачу SLAM, наиболее популярными на данный момент являются графовые. Подробно структура графового алгоритма рассматривалась в главе 1. Популярность этих алгоритмов обусловлена масштабируемостью и возможностью исправлять ошибки в уже построенной карте. И последняя особенность таких алгоритмов является очень полезной для многоагентных подходов.

В рамках графового подхода для каждого нового измерения подыскивается узел в графе. Причём это не обязательно делать “на лету” - для выполнения этого действия существует алгоритм замыкания циклов, который может быть запущен в разы реже, чем снятие измерений. Такая идея может быть эффективно использована в графовом алгоритме, чтобы вычислить взаимное расположение агентов. Структуру графового алгоритма можно без изменений применить для этой задачи, поскольку если оба агента посещали одно и то же место, то, получив измерения с этого места от другого агента, будет возможно выполнить замыкание циклов и определить местоположение, с которого были сняты эти измерения.

Наглядная демонстрация схожести одноагентного и многоагентного графовых SLAM алгоритмов представлена на рисунке.

<рисунок>

Таким образом для графового подхода можно выделить два основополагающих достоинства: простота в масштабировании и возможность глобального изменения карты в процессе работы алгоритма. К недостаткам графового подхода относятся те же недостатки, которыми обладали одноагентные графовые SLAM алгоритмы: понижение скорости работы алгоритма со временем, когда граф накапливает большое количество вершин.

Одним из примеров реализации графового алгоритма является [cg\_mrslam]. В качестве датчика измерений здесь выступает лазерный дальномер. Архитектура этого решения - распределённая сеть, агенты могут связаться друг с другом только находясь на некотором небольшом расстоянии друг от друга. Во время обмена информацией каждый агент передаёт другому свои текущие измерения, затем происходит обновление карт всех агентов, а затем происходит обмен обновлёнными графами для выполнения слияния карт.

На этапе обработки текущего наблюдения другого агента происходит выбор узла собственного графа, с которым можно сопоставить текущее наблюдение, и, следовательно, вычисляется взаимное расположение агентов, даже если они находятся не в одной точке пространства, а на некотором расстоянии друг от друга.

Для замыкания циклов данный алгоритм использует популярную библиотеку g2o, использование которой является де-факто стандартом среди разработчиков графовых алгоритмов SLAM.

Другим примером реализации графового алгоритма является [icra18]. Отличительной особенностью этого алгоритма является разбиение всего окружения на участки, за каждый из которых отвечает определённый агент. Таким образом, как только один из агентов оказывается в области другого - они могут установить связь и начать обмен данными. Такой подход позволяет гарантировать, что первый агент в текущий момент снимает наблюдение, которое уже обрабатывал второй агент. А это означает, что возможно вычислить взаимное расположение агентов и произвести объединение карт.

Отличительным достоинством рассматриваемого авторами алгоритма является низкое количество передаваемого по сети трафика, при этом авторы заявляют, что качество построенного решения не уступает аналогичным алгоритмам.

Недостатком подхода является разделение пространства на районы - такой подход возможен только если окружение известно заранее. Другими словами такой алгоритм подходит для офисов, складов, небольших районов города.

## Неграфовый многоагентный SLAM

Главным недостатком графовых алгоритмов является потребность в больших вычислительных ресурсах. В одной из ключевых статей о глобальной локализации [global monte-carlo scan matcher] Трун ссылается на парадигму “точность вычислений должна зависеть от времени, которое дано на обработку”. Таким образом если алгоритм разработан согласно этой парадигме, то его точность напрямую зависит от вычислительных ресурсов мобильного агента: чем производительнее агент, тем больше операций за одно и то же время он сможет выполнить и тем точнее будет ответ. Графовый подход не укладывается в данную парадигму, ему требуется всего фиксированное количество времени для получения точного ответа. Поэтому в данном разделе показаны альтернативные подходы, их преимущества и недостатки.

В Главе 1 говорилось, что одной из первых модификаций базового EKF SLAM алгоритма являлся FastSLAM. Он работал быстрее, погрешность его работы была мала по сравнению с выигранными ресурсами. Таким образом среди многоагентных SLAM алгоритмов закономерно появился многоагентный FastSLAM алгоритм [fastSLAM].

Авторы этого подхода предлагают запускать одноагентный FastSLAM на каждом отдельном агенте. Конкретно в предложенной работе архитектура представляет собой сеть с маршрутизатором, однако алгоритм сможет также хорошо работать и в архитектуре с распределённой сетью. Аналогично одноагентному подходу здесь используется фильтр частиц, то есть одновременно учитываются и вычисляются вероятности сразу нескольких возможных состояний карты и позиции роботов.

Как и в любом алгоритме с фильтром частиц возникает вопрос о том, как сопоставлять два набора частиц, построенных разными агентами. Прямолинейный вариант - сравнивать каждую частицу с каждой - занимает слишком много вычислительного времени, поэтому авторы в рассматриваемой статье идут по пути известного ухудшения точности в обмен на скорость работы алгоритма. В статье предлагается объединить все частицы с учётом их веса и построить “среднюю” карту на каждом агенте, а затем объединить средние карты. Безусловно, такой подход вносит дополнительный шум в построенную карту, негативно влияет на все построенные во время работы частицы, однако он позволяет выполнить слияние карт максимально быстро.

Другим примером неграфового подхода является [vision-based Centralized SLAM]. Его отличительная особенность также состоит в том, что он представ-



ляет собой успешную реализацию архитектуры с сервером. Несколько дронов с помощью видеокамер исследуют окружение и передают данные серверу, где и происходит слияние измерений и построение карты.

Общей чертой рассмотренных выше алгоритмов является сложность обновления уже построенных участков карты. Графовый подход позволял значительно обновлять карту, если новые измерения противоречат старым, не изменяя консистентность карты. В неграфовых подходах нет инструмента изменить часть карты так, чтобы на ней не осталось артефактов. Частично это возможно при использовании фильтра частиц, однако его использование порождает неразрешимые вопросы для многоагентной архитектуры.

Алгоритм [vision-based Centralized SLAM] реализует фильтр частиц, и сервер в зависимости от данных, полученных от дронов, может вычислить, какая из гипотез о положении дронов имеет наибольшую вероятность.

## **Вопросы многоагентного SLAM**

При построении алгоритма многоагентного SLAM возникают новые задачи по сравнению с одноагентным алгоритмом.

Принципиальным является вопрос об отказоустойчивости системы, поскольку в ней участвует несколько агентов. Наиболее безопасной и, одновременно, легко масштабируемой является система, где все агенты выполняют равные функции, и в случае выхода из строя одного, оставшиеся в системе агенты могли продолжить работу. В настоящее время такие подходы являются наиболее популярными, хоть и для равномерного распределения функций приходится снабжать агентов относительно мощными вычислительными единицами. Подходы, где задачи между агентами распределены неравномерно существуют, однако при проектировании систем согласно таким подходам требуется уделить внимание физической отказоустойчивости агентов.

## **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.