Использование RGBD-камер (моно, стерео) в задачах навигации внутри и вне помещений.

Существуют разные реализации методов построения карты и навигации. Самый универсальный подход к получению карты и навигации по данной карте включает в себя метод SLAM, процедуру сбора данных, построенную карту и метод локализации по данной карте. Поскольку навигация может осуществляться с разных устройств, возможно наличие разных конфигураций оборудования на устройстве.

В большинстве современных устройств установлена камера, на автомобили с автопилотами роботы устанавливают большое количество внешних сенсоров - камеры, лидары.

Существуют три метода и типа сенсоров соответственно для определения глубины в кадре: датчик глубины, который сразу дает нужное представление, метод структурного света, который требует излучатель и приемник, стереокамера и соответствующие методы реконструкции трехмерной сцены.

Как правило методы SLAM умеют напрямую работать датчиком глубины и стереокамерой. Если метод умеет напрямую работать с измерениями глубины, то правильнее передать необработанные данные в алгоритм и выполнить обработку данных в самом алгоритме.

Существующие решения показывают, что можно успешно решать задачу навигации по данным с датчика глубины. Именно для задачи локализации и навигации есть точные методы одометрии. Одометрия не предполагает наличия карты глубины, но осуществляет более точное отслеживание динамики движения. Например, работа DVIO: Depth-Aided Visual Inertial Odometry for RGBD Sensors[1] представляет визуально-инерциальную одометрию с использованием датчика глубины. Безотносительно наличия карты, этот метод сравним или превосходит свои аналоги такие как DUI-VIO, VINS-Mono[2], VINS-RGBD[3].

Определение глубины с помощью камеры смартфона

Последние модели телефонов сейчас оснащаются датчиками для определения глубины. Наличие доступного модуля глубины в смартфоне позволяет получать больше информации о сцене и делать более качественные фотографии, что является одним из основных направлений развития камер смартфонов сейчас.

Информация о поддержке определенным смартфоном функции определения глубины в изображении доступна в спецификации устройства. Для работы с камерой телефона существует API интерфейс от производителя устройства.

Система ARCore[4] работает с камерой на большинстве устройств на базе операционной системы Android, она представляет данные о поддержке устройствами стандарта depth-API в следующем виде (google.com, 2021):

Manufacturer	**Device model**	**Comments**
Google	Pixel 2	Supports 60 fps camera capture frame rate on the rear-facing camera Supports multiple GPU texture resolutions - 1080p, 720p, 480p Supports Depth API

Для определенного устройства Google (производитель) Pixel 2 (модель) сообщается о поддержке работы с датчиком глубины (Depth API). При этом физическая реализация определения глубины неизвестна и не принципиальна, возможно в устройстве установлен специальный сенсор или несколько сенсоров или ведется реконструкция карты глубины по паре камер. Если необходимо обеспечить поддержку работы для большинства устройств (в данном контексте описывается реализация для операционной системы Android), то можно воспользоваться существующим интерфейсом Depth API.

Использование Android Depth API для получения данных глубины сцены

Depth API предоставляет на выбор два метода выдачи данных - исходные или обработанные данные.

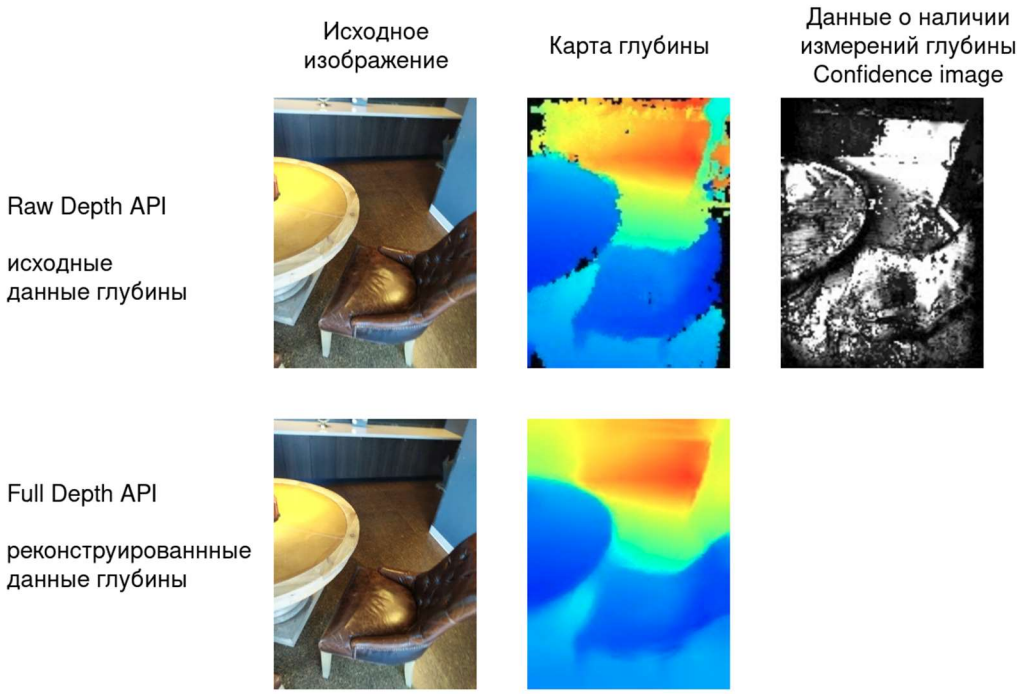

Рисунок 1 Сравнение Raw Depth API и сглаженного изображения Full Depth API

Raw Depth API предоставляет данные глубины для изображения камеры, которые имеют высокую точность, но не всегда покрывают каждый пиксель. *Raw Depth API* возвращает необработанное изображение глубины и маску достоверности для этого изображения. Необработанное изображение глубины, содержит очень точную оценку глубины для некоторых, но не всех пикселей изображения камеры. Маска достоверности - битовая маска изображения камеры, показывает пиксели, для которых нет измерений глубины, т.е. имеющих нулевую достоверность.

Full Depth API сглаживает данные и предоставляет аппроксимацию исходного изображения, но данные глубины для каждого пикселя могут быть менее точными из-за сглаживания и интерполяции оценок глубины.

Необработанные изображения глубины вместе с соответствующими им достоверными изображениями также могут подвергаться дальнейшей обработке, что позволяет приложениям использовать только те данные глубины, которые имеют достаточную точность для их индивидуального

варианта использования. Формат и размер изображений глубины одинаковы для обоих API.

Модели телефонов, их камеры и соответственно характеристики изображений будут значительно различаться. Независимо от качества и характеристик камеры и устройства вообще, возможно получить необходимые данные для запуска алгоритма SLAM. С использованием depth api или напрямую от устройства можно получить данные глубины, передать их в алгоритм локализации и навигации.

Работа с данными глубины в алгоритмах SLAM

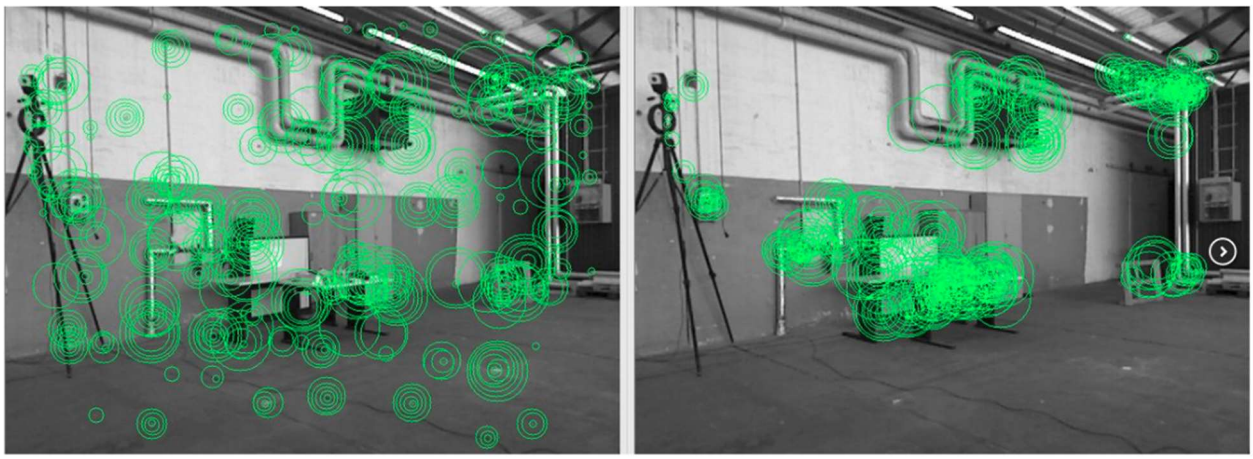

Рисунок 2 Определение характерных точек ORB[5]

Для локализации и навигации необходимо первоначально получить карту. Методы SLAM, основанные на характерных точках (*Feature based* [6]–[9]), должны определить координаты для соответствующих характерных точек чтобы на их основе потом построить карту. Соответственно, чтобы получить измерение глубины для любой определенной координаты (характерной точки), нужно использовать аппроксимацию карты глубины. А чтобы полученные данные были максимально точными и не зависели от точности аппроксимации, наоборот нужно использовать исходные данные без всякой аппроксимации.

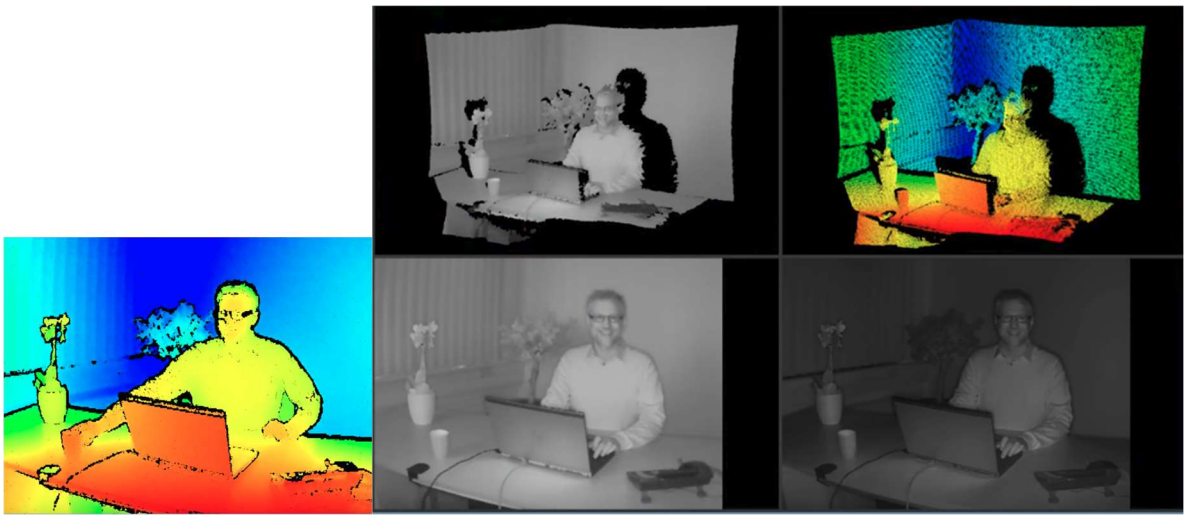

Рисунок 3 Карта глубины с TOF сенсора, реконструкция 3d изображения по карте глубины

Если рассмотреть изображение с сенсора глубины, можно заметить что в точках контура объектов нет линейной аппроксимации между глубиной объекта и фона. Если в изображении есть близкий объект и далекий фон, то чем точнее будет определен контур, тем меньше будет неправильных и пропущенных данных с датчика глубины.

Другое наблюдение, алгоритмы FAST и BRIEF фиксируют именно положение углов/контуров объектов на изображении. Для таких характерных точек данные о глубине фактически отсутствуют, а значит метод не будет работать правильно.

Представление трехмерной карты в задачах SLAM

Основные типы представлений карты в задачах SLAM:

1. Карта препятствий (Occupancy Map, Grid)
2. Облако точек (Point Cloud)
3. Плотное облако точек (Octomap[10], dense point cloud)

Карты делятся на плотные и неплотные. Плотные хранят полную информацию о пространстве. Неплотные регистрируют координату для каждой записи о карте. Подходы к организации абсолютно противоположные.

Плотные карты позволяют осуществлять быстрый поиск, в том числе по графу (Octomap - трехмерное дерево). В неплотных картах информация о пространстве не хранится, вместо этого используются функции кодирования которые по изображению определяют ключевые точки (дескрипторы) и ищут

соответствующие дескрипторы в словаре карты. Сама карта представлена функцией отображения и графом состояний, где каждая вершина — это дескриптор в пространстве. Карта занимает меньше места, возможен быстрый поиск, но при этом теряется часть информации и невозможно напрямую сопоставить изображения.

Method	Sensors	Back-end	Geometry	Sema-ntics
ORB-SLAM [22]	mono	g2o	points	✗
DSO [23]	mono	g2o	points	✗
VINS-mono [24]	mono/IMU	Ceres	points	✗
VINS-Fusion [25]	mono/Stereo/IMU	Ceres	points	✗
ROVIOLI [26]	stereo/IMU	EKF	points	✗
ElasticFusion [18]	RGB-D	alternation	surfels	✗
Voxblox [27]	RGB-D	[26]	TSDF	✗
SLAM++ [16]	RGB-D	alternation	objects	✓
SemanticFusion [17]	RGB-D	[18]	surfels	✓
Mask-fusion [28]	RGB-D	[29]	surfels	✓
SegMap [30]	lidar	GTSAM	points/segments	✓
XIVO [31]	mono/IMU	EKF	objects	✓
Voxblox++ [14]	RGB-D	[26]	TSDF	✓
Kimera	**mono/stereo/IMU**	**GTSAM**	**mesh/TSDF**	✓

Рисунок 4 Типы представления данных в методах SLAM[6]

Существуют также промежуточные реализации, которые помимо облака точек и графа карты хранят дополнительные данные о локации.

Использование семантической сегментации в методе SLAM

Улучшить результаты работы с картой глубины можно с помощью методов сегментации изображения (semantic segmentation), с помощью нейросетевых методов или аналогичными алгоритмами определяется контур объекта[6]. Семантическая разметка изображения позволяет правильно аппроксимировать карту глубины и получать более точную трехмерную реконструкцию и карту.

Задача семантической разметки требует дополнительных вычислительных операций, но в целом требуется только для построения карты.

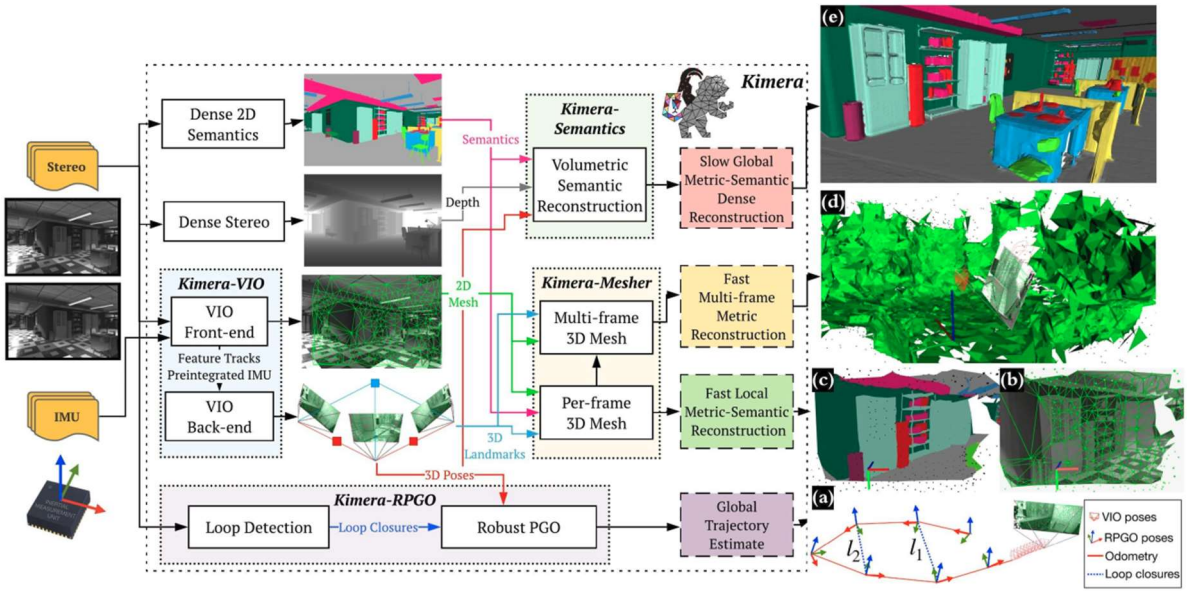

Рисунок 5 Архитектура Kimera. Kimera использует изображения и данные IMU в качестве входных (показано слева) и выходных данных (a) оценки позы и (b-e) множественные метрико-семантические реконструкции[11].

Примером полноценной семантической системы SLAM является Kimera[11]. Она содержит 4 ключевых модуля:

- визуально инерциальная одометрия
- оптимизация графа
- построение полигональной сетки карты
- семантическая сегментация карты.

В данной реализации, Kimera использует реконструкцию глубины из стерео. При хорошей производительности отдельных элементов, система позиционируется как решение для одометрии. Точность построения триангуляции геометрии порядка 0.4 м в среднем, время обновления цикла навигации 45 мс, средняя ошибка позиционирования 10-20 см.

Семантическая карта может быть получена и без данных глубины, для mono-SLAM методов[2], [8], [12]–[15] тоже возможно применять такой подход. В Dyna-SLAM[16] показывается работа метода для обычной камеры без поддержки обработки глубины в сравнении с обычным mono-SLAM.

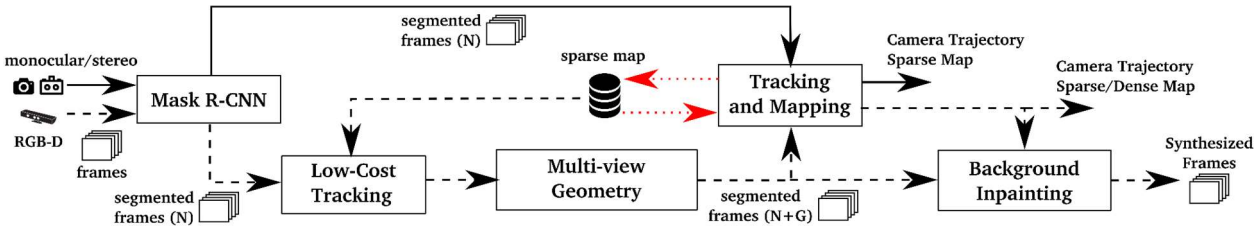

Рисунок 6 Dyna-SLAM , структурная схема [16]

Dyna-SLAM рассматривают случай построения карты в динамическом окружении. Если по видео сгенерировать карту и попробовать осуществлять навигацию, то ничего не заработает.

Оказывается, что люди, автомобили и другие подвижные объекты все еще являются частью карты, но не сохраняют свои местоположения на карте. В итоге можно сопоставить изображение человека и получить координату его прошлого местоположения. В случае машин получается, что существует множество одинаковых машин, и привязываться к машине как к уникальному идентификатору вообще некорректно. Соответственно надо научиться определять такие классы объектов в карте как люди и автомобили.

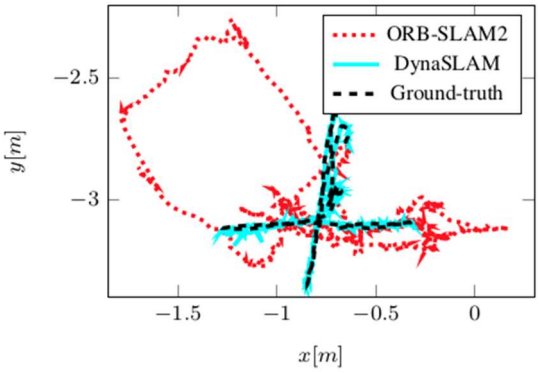

Рисунок 7 Сравнение навигации методом ORB-SLAM2 (красный) и DynaSLAM (голубой) с реальной траекторией (черный) [16]

Для примера динамического окружения, навигация методом ORB-SLAM2 дает большую погрешность, хотя метод в целом показывает хорошую точность в нормальных условиях. Проблема возникает, потому что метод построен на предположении о статичном окружении при построении карты.

Автор DynaSLAM рассматривает следующие конфигурации системы:

1. DynaSLAM (N) — система, в которой производится только сегментация кадров с помощью Mask R-CNN[17], что и является априорным выходным объектом системы.
2. DynaSLAM (G) представляет собой набор геометрических методов определения трехмерной геометрии на основе глубины изменения.
3. DynaSLAM (N + G) это система, в которой выделены динамические объекты, и объединяет одновременно геометрические и нейросетевые методы.
4. Дополнительно можно рассматривать конфигурацию (N + G + BI), где выполняется закрашивание/дорисовывание фона изображения (BI background inpainting) перед циклом алгоритма SLAM. Мотивация для такого эксперимента заключается в том, что, если динамические области заранее закрасить известным статическим содержимым, система может работать в качестве стандартной SLAM-системы в предположении статичности внешней сцены. В этом случае ORB характерные точки ищутся как в реальной, так и в реконструированной области кадров, в поиске совпадений с ключевыми точками в ранее обработанных ключевых кадрах.

Рисунок 8 Работа алгоритма Mask R-CNN [17]

Авторы DynaSLAM приводят статистику производительности метода в сравнении с ORB-SLAM.

Монокулярная версия ORB-SLAM позволяет получить в среднем более точные результаты чем ORB-SLAM с камерой глубины, но при этом требует времени на инициализацию: на наборе данных TUM, время установления навигации для mono ORB-SLAM составляет порядка 13-20% от общего времени навигации. При этом DynaSLAM в монокулярной версии выдает

результаты навигации порядка 97% от общего времени навигации, то есть с незначительной задержкой.

При этом оба метода показывают примерно равнозначную точность на общем наборе данных датасета KITTY[18].

В основе алгоритма лежит простая идея игнорировать все ключевые точки, которые принадлежат динамическим объектам. При этом за динамические объекты априорно принимаются определенные классы объектов, то есть автомобили, велосипеды и т. д. Для ситуации, где все объекты на виде движутся, игнорирование динамических объектов улучшает точность и алгоритм работает лучше, чем ORB-SLAM[7]. Наоборот, на последовательностях, в которых большинство зарегистрированных автомобилей и транспортных средств припаркованы (следовательно, статичны), абсолютная среднеквадратичная ошибка траектории обычно больше, так как большинство ключевых точек, используемые для отслеживания просто игнорируются, а остальные более удалены и обычно относятся к мало текстурным областям.

DynaSLAM не оптимизирован не предназначен для работы в реальном времени. Однако возможность создавать карты извлекая только статическое содержимое сцены актуальна для работы в автономном режиме. Возможность классифицировать характерные точки на статичные и динамические должна позволить работать только со статической картой в реальном времени не выполняя задачу сегментации. При этом более новая версия алгоритма ORB-SLAM 3 может работать с несколькими участками карты и возможно получится интегрировать динамическую карту с помощью метода ORB-SLAM 3[6, p. 3].

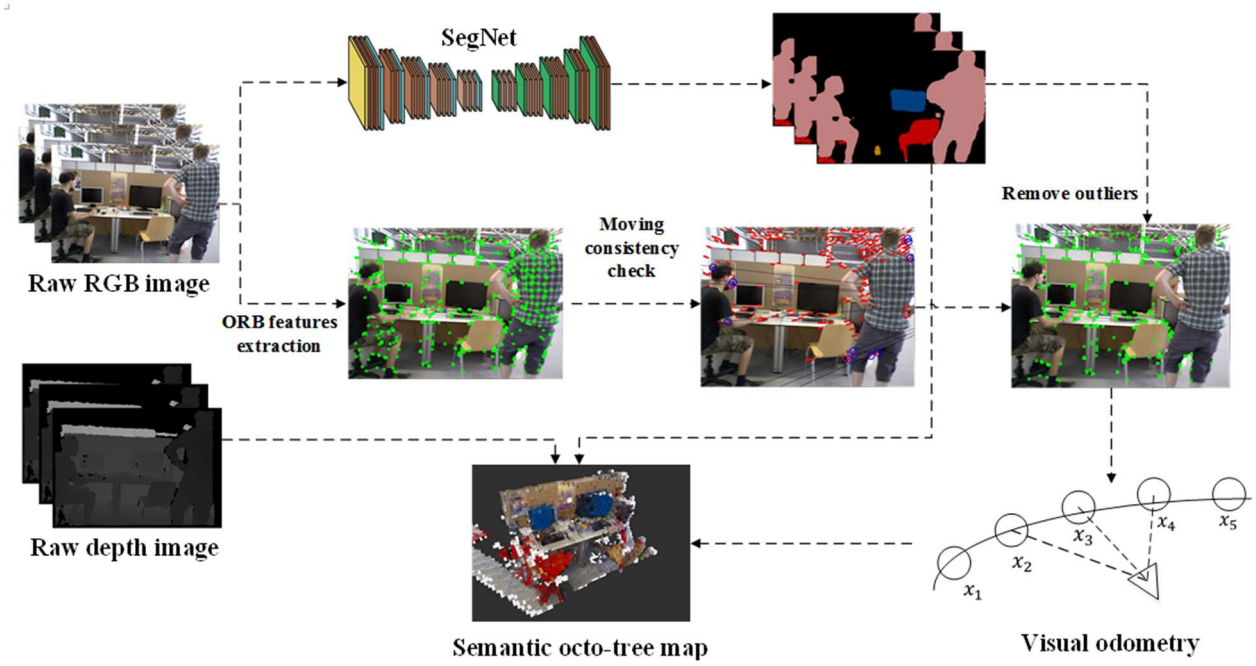

Рисунок 9 Структурная схема DS-SLAM[19].

DS-SLAM: Необработанное изображение RGB используется одновременно для семантической сегментации и проверки согласованности движений. Затем удаляется шум и определяется координата объекта. Семантическая карта представлена в виде октодерева и строится в независимом потоке на основе расположения объекта, карты глубины и результата семантической сегментации.

Аналогичное решение реализовано в работе DS-SLAM, при этом авторы заявляют о значительном улучшении точности и стабильности метода в их эксперименте по сравнению с ORB-SLAM 2[7]. Система производит семантическую сегментацию в реальном времени (59 мс на обработку одного кадра) и может определять и фильтровать движущиеся объекты. Характерные точки с движущихся объектов не добавляются в карту и не используются при локализации. На динамических последовательностях средняя точность у DS-SLAM[19] примерно в 2-10 раз лучше точности ORB-SLAM 2 соответственно.

При этом можно в любой момент времени перестать выполнять семантическую сегментацию и получить исходный алгоритм ORB-SLAM[7] с его производительностью.

Технологии определения глубины сцены

Основные технологии определения глубины классифицируются следующим образом:

- Structured Light камеры (Autodesk, 2019), или камеры структурного света, когда есть проектор (часто инфракрасный) и камера, снимающая структурный свет проектора;
- Time of Flight камеры, или камеры, основанные на измерении задержки отраженного света;
- Depth from Stereo камеры — классическое и, пожалуй, наиболее известное направление построения глубины из стерео;
- Light Field Camera — они же камеры светового поля или пленоптические камеры
- И, наконец, камеры, основанные на Lidar-технологиях, особенно свежие Solid State Lidars, которые работают без отказа примерно в 100 раз дольше обычных лидаров и выдают привычную прямоугольную картинку.

Для всех типов устройств есть свои области применения и ограничения.

По статистике, в телефонах сейчас применяются Time of Flight камеры (Quero, 2021), Depth from Stereo и Solid State лидары.

Для автомобилей решение компании Tesla основано на технологии Depth from Stereo, все еще является самым перспективным и дает хорошие результаты.

Применение камеры светового поля ограничено наличием соответствующих моделей на рынке и в разработке. Гораздо больше готовых решений сейчас есть в области TOF камер, которые устанавливаются в смартфоны (ближнего действия), в магазинах (2-5 м радиус обзора).

Для большего радиуса действия подходят лидары и камеры, при этом у первых значительно больше сложностей с расположением, меньше надежность, меньше частота получения данных.

Список литературы

Autodesk. (2019). *Structured Light 3D Scanning* . Получено из Autodesk: Structured Light 3D Scanning : https://www.instructables.com/id/Structured-Light-3D-Scanning/

google.com. (2021). *developers.google.com*. Получено из ARCore: https://developers.google.com/ar/develop/c/depth/developer-guide?hl=ur

Quero, A. (12 12 2021 г.). *ToF puede llegar a detectar cámaras ocultas*. Получено из tododisca.com: https://www.tododisca.com/telefono-detectar-camaras-ocultas/

[1] A. Tyagi, Y. Liang, S. Wang, and D. Bai, "DVIO: Depth aided visual inertial odometry for RGBD sensors," *ArXiv211010805 Cs*, Oct. 2021, Accessed: Dec. 27, 2021. [Online]. Available: http://arxiv.org/abs/2110.10805

[2] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018, doi: 10.1109/TRO.2018.2853729.

[3] Y. Zhu, R. Jin, T. Lou, and L. Zhao, "PLD-VINS: RGBD visual-inertial SLAM with point and line features," *Aerosp. Sci. Technol.*, vol. 119, p. 107185, Dec. 2021, doi: 10.1016/j.ast.2021.107185.

[4] A. Morar, M. A. Balutoiu, A. Moldoveanu, F. Moldoveanu, A. Butean, and V. Asavei, "Evaluation of the ARCore Indoor Localization Technology," in *2020 19th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, Bucharest, Romania, Dec. 2020, pp. 1–5. doi: 10.1109/RoEduNet51892.2020.9324849.

[5] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, Barcelona, Spain, Nov. 2011, pp. 2564–2571. doi: 10.1109/ICCV.2011.6126544.

[6] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," 2021.

[7] R.-D. Cameras, R. ur-Artal, Juan, and D. Tardos, "ORB SLAM 2 : an Open-Source SLAM System for Monocular, Stereo and."

[8] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Trans. Robot.*, vol. 31, no. 5, Art. no. 5, 2015, doi: 10.1109/tro.2015.2463671.

[9] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real Time Localization and 3D Reconstruction," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, New York, NY, USA, 2006, pp. 363–370. doi: 10.1109/CVPR.2006.236.

[10] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: an efficient probabilistic 3D mapping framework based on octrees," *Auton. Robots*, vol. 34, no. 3, pp. 189–206, Apr. 2013, doi: 10.1007/s10514-012-9321-0.

[11] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping," in *2020*

IEEE International Conference on Robotics and Automation (ICRA), Paris, France, May 2020, pp. 1689–1696. doi: 10.1109/ICRA40945.2020.9196885.

[12] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," in *Computer Vision – ECCV 2014*, vol. 8690, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 834–849. doi: 10.1007/978-3-319-10605-2_54.

[13] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 15–22. doi: 10.1109/ICRA.2014.6906584.

[14] M. De Croce, T. Pire, and F. Bergero, "DS-PTAM: Distributed Stereo Parallel Tracking and Mapping SLAM System," *J. Intell. Robot. Syst.*, vol. 95, no. 2, Art. no. 2, Aug. 2019, doi: 10.1007/s10846-018-0913-6.

[15] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, Art. no. 6, Jun. 2007, doi: 10.1109/TPAMI.2007.1049.

[16] B. Bescos, J. M. Facil, J. Civera, and J. Neira, "DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 4076–4083, Oct. 2018, doi: 10.1109/LRA.2018.2860039.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *ArXiv170306870 Cs*, Jan. 2018, Accessed: Dec. 27, 2021. [Online]. Available: http://arxiv.org/abs/1703.06870

[18] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A Review on Deep Learning Techniques Applied to Semantic Segmentation," *ArXiv170406857 Cs*, Apr. 2017, Accessed: Dec. 27, 2021. [Online]. Available: http://arxiv.org/abs/1704.06857

[19] C. Yu *et al.*, "DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Oct. 2018, pp. 1168–1174. doi: 10.1109/IROS.2018.8593691.