Разработка бортовой системы навигации и ориентации на основе SLAM модели

КФ МГТУ им. Н.Э. Баумана
Р.И. Гришин
e-mail: mr.grischinroman@yandex.ru

Калуга, 2020

Постановка задачи

Цель работы: разработать бортовую систему навигации и ориентации на основе SLAM модели.

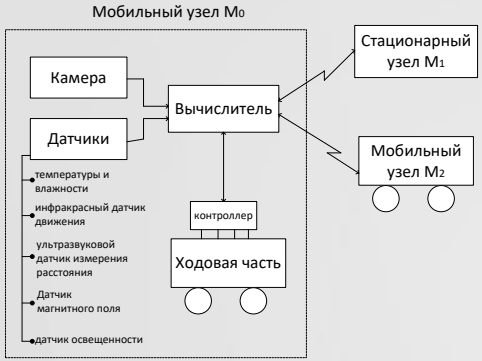


Рисунок 1 – Структурная схема мобильного узла

Ограничения:

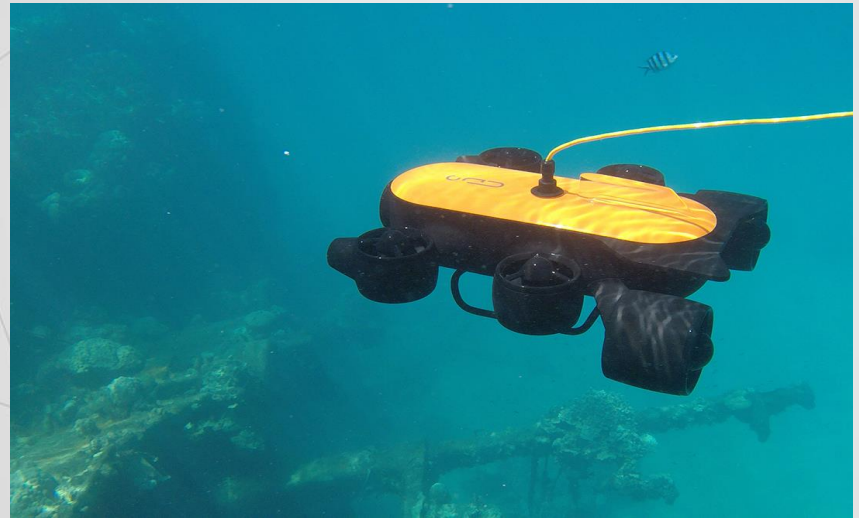
-работа в замкнутом помещении
-невысокая производительность
-низкая скорость движения

Задачи:

- Выполнить анализ алгоритмов SLAM;
- Разработать стенд для экспериментальной проверки решений;
- Разработать схему эксперимента;
- Провести эксперимент и выполнить анализ полученных результатов.

Актуальность (область применения):

- обнаружение задымления в помещении и оповещение о ЧП;
- функционирование систем ПВО на базе сочлененных гусеничных машин;
- проведение работ в среде, непригодной для нахождения в ней человека;
- обслуживание социальной и бытовой сферы;
- транспортировка грузов в складских помещениях;
- выполнение подзадач системы «умного дома»;
- автоматизации видеонаблюдения;



Анализ методов позиционирования и картирования

- **Метод инвариантных моментов**

Основывается на теории алгебраических инвариантов Ху [19]

Алгоритм содержит следующие этапы:

$$M_1' = r \cdot h, \;\; M_2' = \frac{M_2}{r^4}, \;\; M_3' = \frac{M_3}{r^6}, \ldots \quad (1)$$

1) определение центральных моментов порядка не выше третьего;
2) получение моментов, инвариантных к операциям поворота, переноса и зеркального отображения;
3) получение моментов, инвариантных к полной группе аффинных преобразований:

- **Бесплатформенная инерциальная система (БИС)**

Инерциальная навигация - метод навигации, основанный на свойствах инерции тел, являющийся автономным, не требующим наличия внешних ориентиров или поступающих извне сигналов.

Сущность инерциальной навигации состоит в определении ускорения объекта и его угловых скоростей с помощью установленных на движущемся объекте приборов и устройств.

---

[19]Hu M. K. Visual pattern recognition by moment invariants. – IRE Transactions on Information Theory 8, 1962, pp. 179-187

Анализ методов позиционирования и картирования

Reinforcement Learning (обучение с подкреплением)
Агент обучается взаимодействуя со средой.
Откликом среды на принятые решения являются сигналы подкрепления, поэтому такое обучение является частным случаем обучения с учителем, но учителем является среда или её модель.

- SLAM (Simultaneous Localization and Mapping) – алгоритм одновременного картирования и локализации

--Шаг 1: Предсказательный
Используем данные, полученные с датчиков, чтобы вычислить оценку её нового положения.
--Шаг 2: Обновление состояния из повторно наблюдаемых ключевых точек
--Шаг 3: Добавление новых ориентиров в текущее состояние
На данном шаге обновим вектор состояния X и ковариационную матрицу P с новыми ориентирами.
Цель – увеличить количество ключевых точек.

Термины и определения

- **SLAM** (*Simultaneous Localization and Mapping*) – одновременное вычисление позиции робота и построение карты окружения
- **Localization** – обнаружение позиции робота
- **Mapping** – построение карты окружения
- **VSLAM** (*Visual Simultaneous Localization and Mapping*) – метод SLAM, использующий в качестве датчика оптическую систему
- **Особая точка** — точка на изображении, окрестность которой выделяется среди остального изображения
- **Ориентир** — точка в пространстве, изображение окрестности которой было классифицировано как особая точка на нескольких кадрах

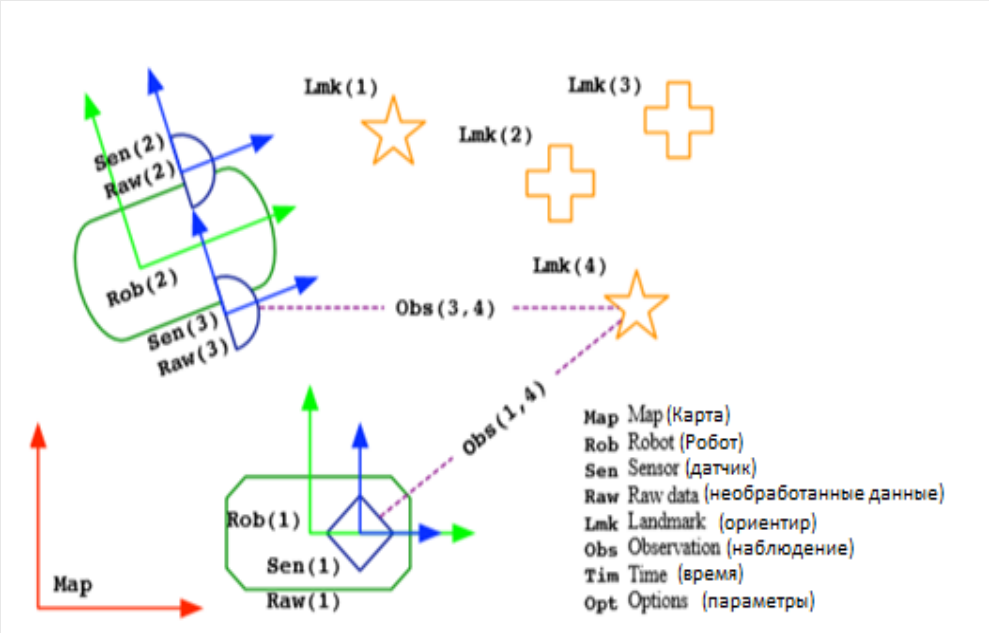


Рисунок 4 – Схема работы алгоритма

Математический аппарат

Шаг 1: Предсказательный - обновление данных, с использованием одометрии

Вектор оценки текущего местоположения X_p и ковариационная матрица P_p

$$X_p = [x_p \quad y_p \quad \varphi_p]^T \qquad (2)$$

x_p – оценка координаты робота по оси абсцисс,
y_p – оценка координаты робота
по оси ординат,
φ_p – оценка ориентации робота

$$P_p = \begin{pmatrix} \sigma^2_{x_p x'_p} & \sigma^2_{x_p y_p} & \sigma^2_{x_p \varphi_p} \\ \sigma^2_{x_p y_p} & \sigma^2_{y_p y_p} & \sigma^2_{y_p \varphi_p} \\ \sigma^2_{x_p \varphi_p} & \sigma^2_{y_p \varphi_p} & \sigma^2_{\varphi_p \varphi_p} \end{pmatrix} \qquad (3)$$

Математический аппарат

Шаг 2: Обновление состояния из повторно наблюдаемых ключевых точек

Коэффициент Калмана (4) содержит информацию о количестве ориентиров и позиций робота которые должны быть обновлены в соответствии с повторно наблюдаемыми ориентирами.

X - новый вектор состояния (5)

$$K = P \cdot H^T \cdot (H \cdot P \cdot H^T + V \cdot R \cdot V^T)^{-1} \quad (4)$$

$$X = X + K \cdot (z - h) \quad (5)$$

Математический аппарат

Шаг 3: Добавление новых ориентиров в текущее состояние

добавляем в вектор оценки состояния системы два новых элемента – оценку x и y координат обнаруженного ориентира (5)

$$X_k = \begin{bmatrix} X_k \\ f_i(X_{pk}, z_i) \end{bmatrix} \quad (5)$$

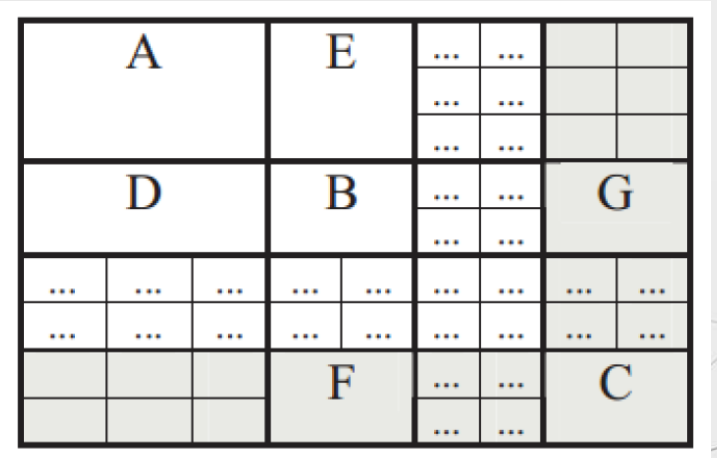


Рисунок 5 - Общий вид матрицы P

Метод VSLAM: ORB-SLAM

- Tracking — отслеживание кадров.
- Local Mapping — выполняет построение карты вблизи текущего положения камеры и оптимизирует карту.
- Loop Closing — алгоритм замыкания циклов, который ищет и объединяет похожие кадры.

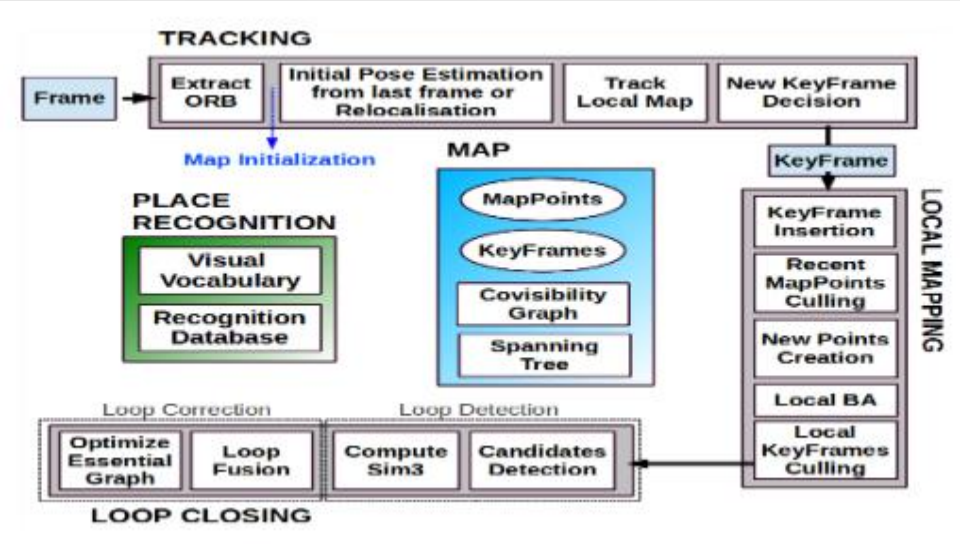


Рисунок 6 – Основные компоненты ORB-SLAM [23]

---

[23] Rublee, Ethan; Rabaud, Vincent; Konolige, Kurt; Bradski, Gary (2011). "ORB: an efficient alternative to SIFT or SURF" (PDF). IEEE International Conference on Computer Vision (ICCV).

Метод VSLAM: LSD-SLAM

- tracking непрерывно отслеживает новые изображения с камеры и определяет перемещение камеры.
- depthmapestimation сравнивает новый кадр с текущим, а затем уточняет или полностью заменяет текущий кадр.
- mapoptimization выполняет оптимизацию карты.

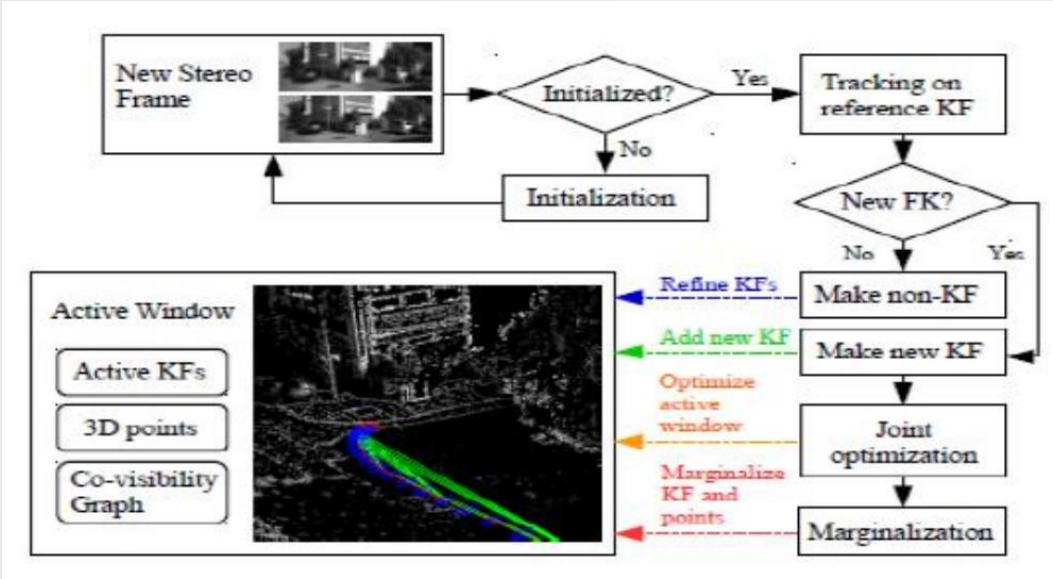


Рисунок 7 – Основные компоненты LSD-SLAM [28]

---

[28]. LSD-SLAM: Large-Scale Direct Monocular SLAM Contact: Jakob Engel, Prof. Dr. Daniel Cremers https://vision.in.tum.de/research/vslam/lsdslam

Анализ методов VSLAM

Таблица 1 – Сравнительная характеристика алгоритмов ORB и LSD-SLAM

Параметр	ORB-SLAM	LSD-SLAM
Открытый исходный код	+	+
Интеграция с ROS*	+	-
Поддерживаемые типы видеосенсоров	Монокамерные, стереокамерные, RGB-D оптические системы	Монокамерные оптические системы
Ресурсоемкость	низкая	высокая
Карта глубины	разреженная	плотная

*ROS - *(Robot Operating System)* – фреймворк для программирования роботов

Оценка алгоритмов SLAM

Как оценить работу SLAM:

-- Использовать истинные данные;
-- Использовать «псевдоистинные данные» - сравнение с другим источником (например лидар)
-- Сравнение результатов по коэффициенту NEES

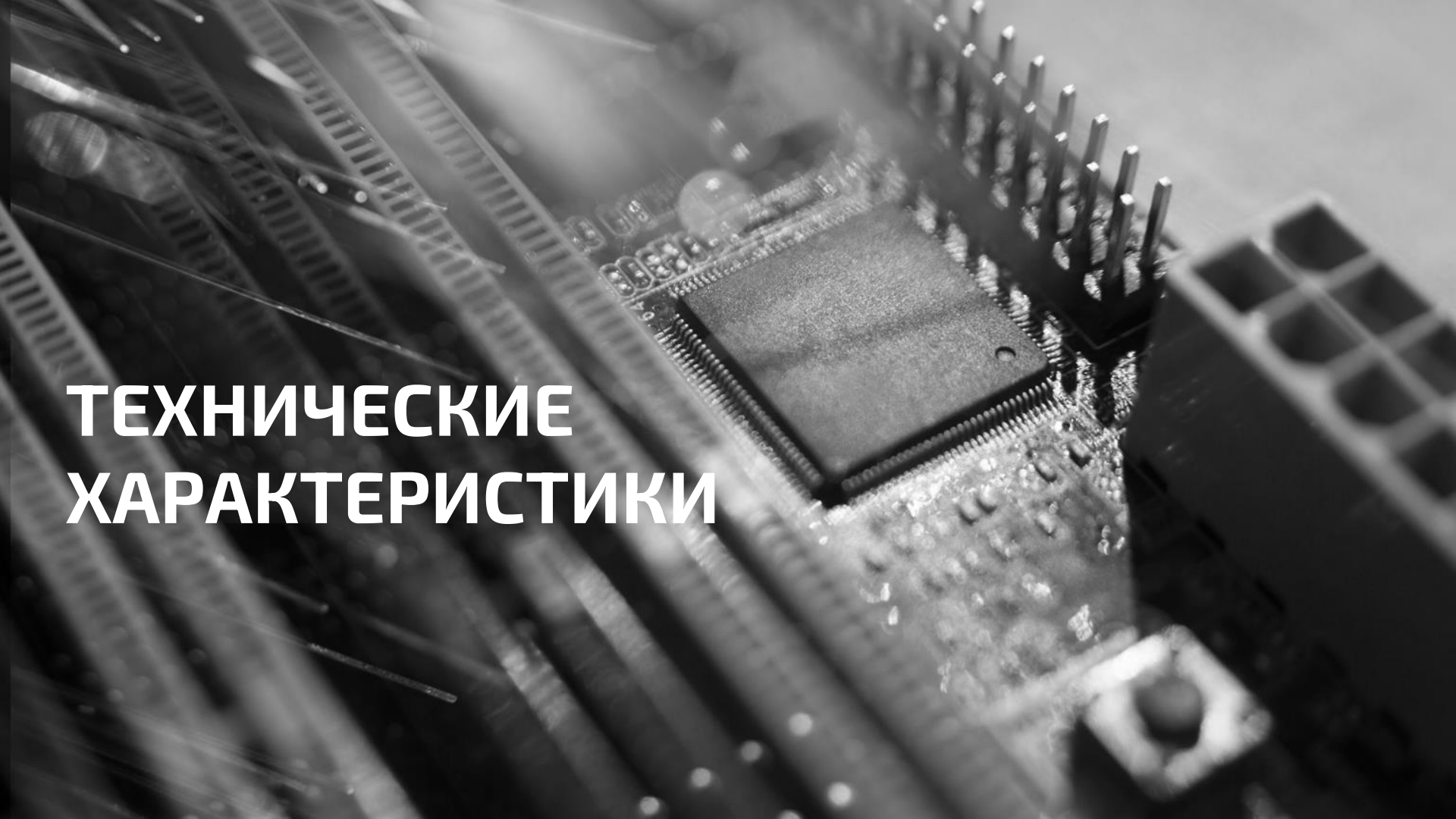
NEES (normalized estimation error squared)
нормализованная ошибка оценки в квадрате

$$\eta_k = \frac{1}{N}\sum_{j=1}^{N}(C_k - \hat{C}_k^j)^T P_k^{j-1}(C_k - \hat{C}_k^j) \quad (9)$$

Гомогенная точка: $L_{HP} = \underline{p} = \begin{bmatrix} m \\ \rho \end{bmatrix} = [m_x \;\; m_y \;\; m_z \;\; \rho]^T \in P^3$ (6)

Закрепленная однородная точка: $L_{AHP} = \begin{bmatrix} P_0 \\ m \\ \rho \end{bmatrix} = [x_0 y_0 z_0 m_x m_y m_z \rho]^T$ (7)

Привязанные модифицированные полярные точки: $L_{AMPP} = \begin{bmatrix} P_0 \\ (\varepsilon, \alpha) \\ \rho \end{bmatrix} = [x_0 y_0 z_0 \alpha \rho]^T$ (8)



ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Оценка доступности ресурсов

Таблица 2 – Характеристики платы

Процессор	**Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz**
Память	1GB LPDDR2 SDRAM
Интерфейсы	2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE Gigabit Ethernet over USB 2.0 4 × USB 2.0
Порты	Extended 40-pin GPIO header
Видео/звук	1 × full size HDMI MIPI DSI дисплей порт MIPI CSI порт для подключения камеры
Мультимедиа	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
Поддержка SD карты	Формат Micro SD для загрузки операционной системы и хранилище данных (не рекомендуется больше 16 Гб)
Питание	5V/2.5A DC via micro USB connector 5V DC via GPIO header Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Связь	Wi-Fi, Ethernet (RJ-45);
Размеры	Длина 85 мм Ширина 56 мм.

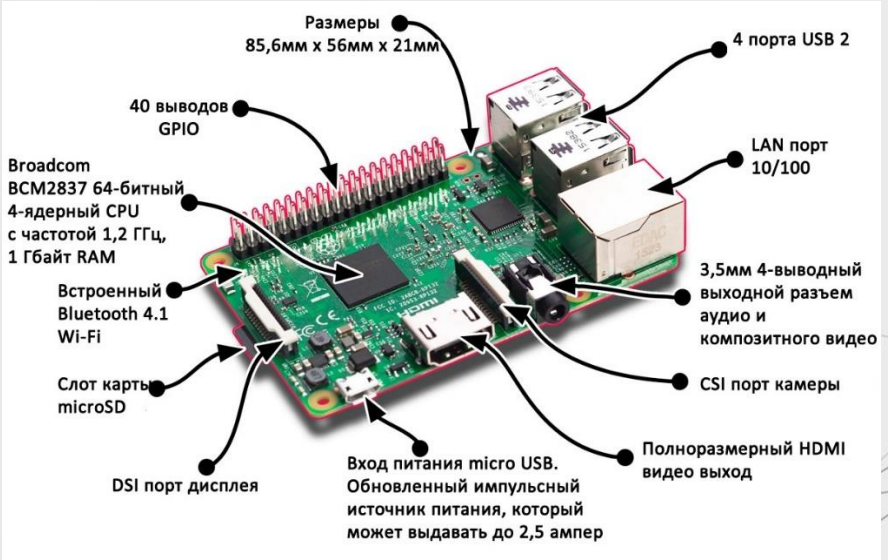


Рисунок 8 – Состав Raspberry Pi 3B+

Структурные схемы алгоритмов

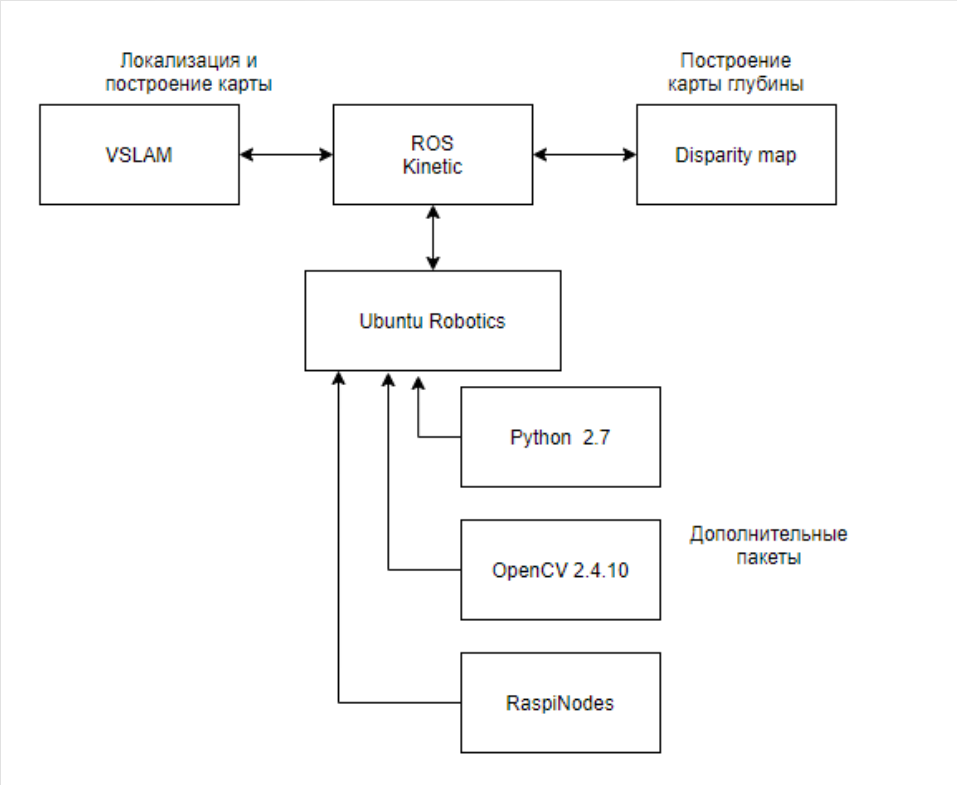


Рисунок 9 – Схема взаимодействия компонентов ПО

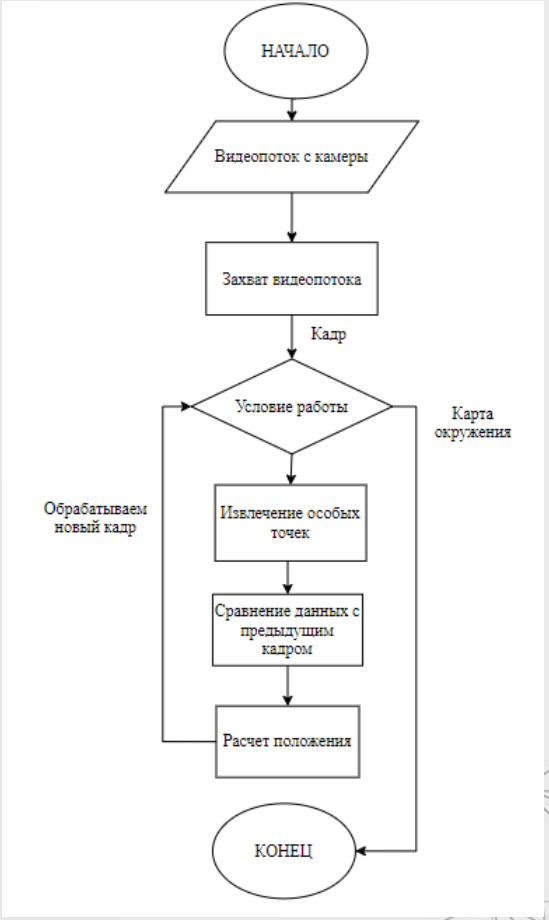


Рисунок 10 – Схема решения задачи

Моделирование

- Создали шаблонное окружения для тестирования навигации
- Создали шаблонного робота
- Подключили алгоритм SLAM
- Настроили конфигурационные файлы
- Запустили в ручном и автоматическом режиме

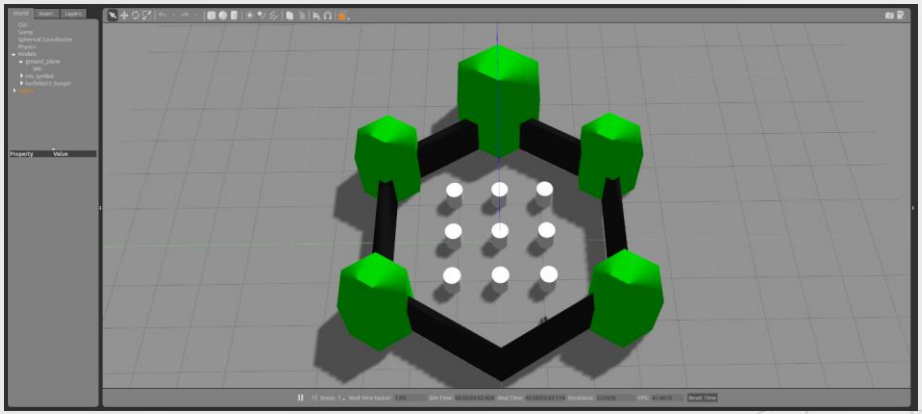
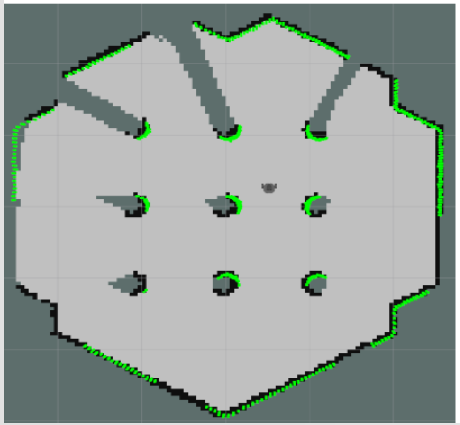
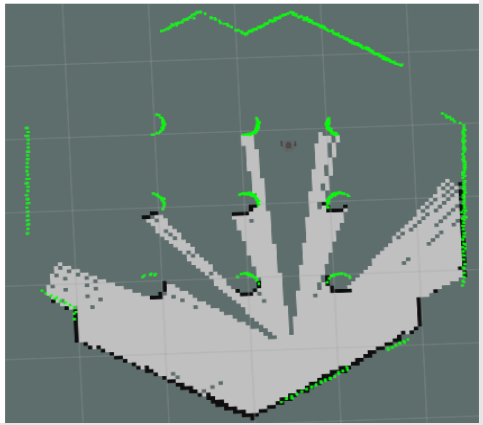


Рисунок 9 – Карта окружения





← Рисунок 10 – Исследование карты объектом

Результаты моделирования

TurtleBot3 полностью исследовал сцену за 68 минут и проехал в общей сложности 358 м.

Ошибка вычисления собственного положения

Конечные координаты, см	Расстояние между начальными и конечными координатами, см	Пройденный путь, см	Погрешность, %
(3,-1)	3,16	100	3,16
(5,2)	5,38	200	2,69
(-4,6)	7,2	300	2,4
(5,7)	8,6	400	2,15
(-3,-4)	5	500	1
(-8,7)	10,63	600	1,77
(8,12)	14,4	700	2,05
(-9,9)	12,72	800	1,59
(10,12)	15,62	900	1,73
(14,9)	16,64	1000	1,66

Ошибка вычисления расстояния

Реальное расстояние, см	Вычисленное расстояние, см	Ошибка, %
50	49,1	1,8
80	79,3	0,875
110	110,3	0,27
140	141	0,7
170	172,2	1,29
200	203,8	1,9
230	235,4	2,3

--Для идеальных данных: Средняя относительная погрешность вычисления собственного положения составляет 2,02%, наибольшая относительная погрешность составляет 3,16%.

Разработка и настройка стенда

Классическая калибровка методом Р. Тсая

$$param = \begin{pmatrix} f_x & a_{12} & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \quad (10)$$

$$distorsion = \left[k_1, k_2, p_1, p_2, k_3\right] \quad (11)$$

Метод состоит из 2 – х этапов:

1. Определение параметров внешней калибровки
2. Определение параметров внутренней калибровки и дисторсии.

Рисунок 11 – Калибровочный шаблон

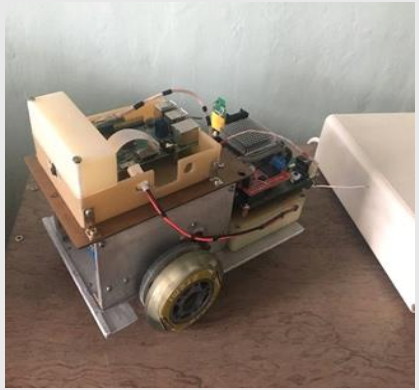


Рисунок 12 – Мобильная платформа

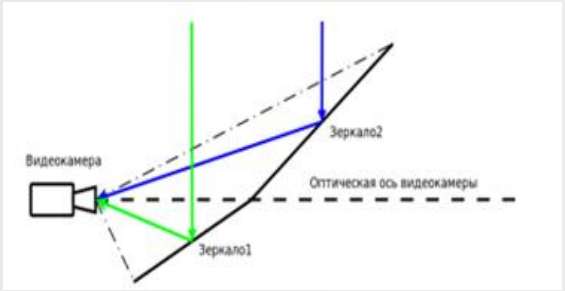


Рисунок 13 – Система зеркал

Реализация SLAM на Raspberry – Stereo Mode

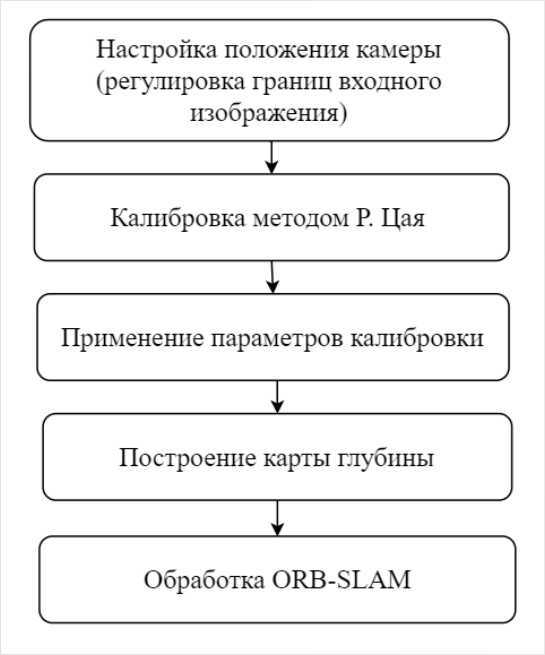
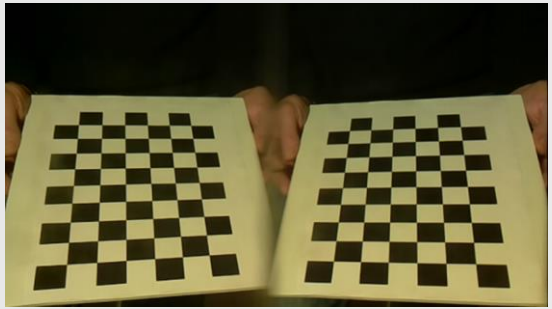


Рисунок 14 – Схема эксперимента

Рисунок 15 – Изображение с пары камер ->



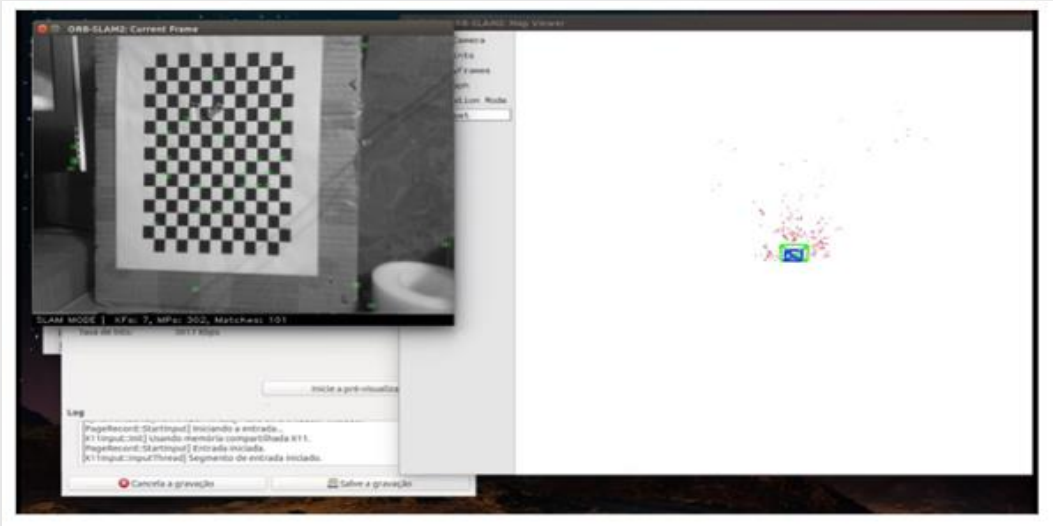


Рисунок 16 – Карта окружения

Реализация SLAM на Raspberry – Monocam

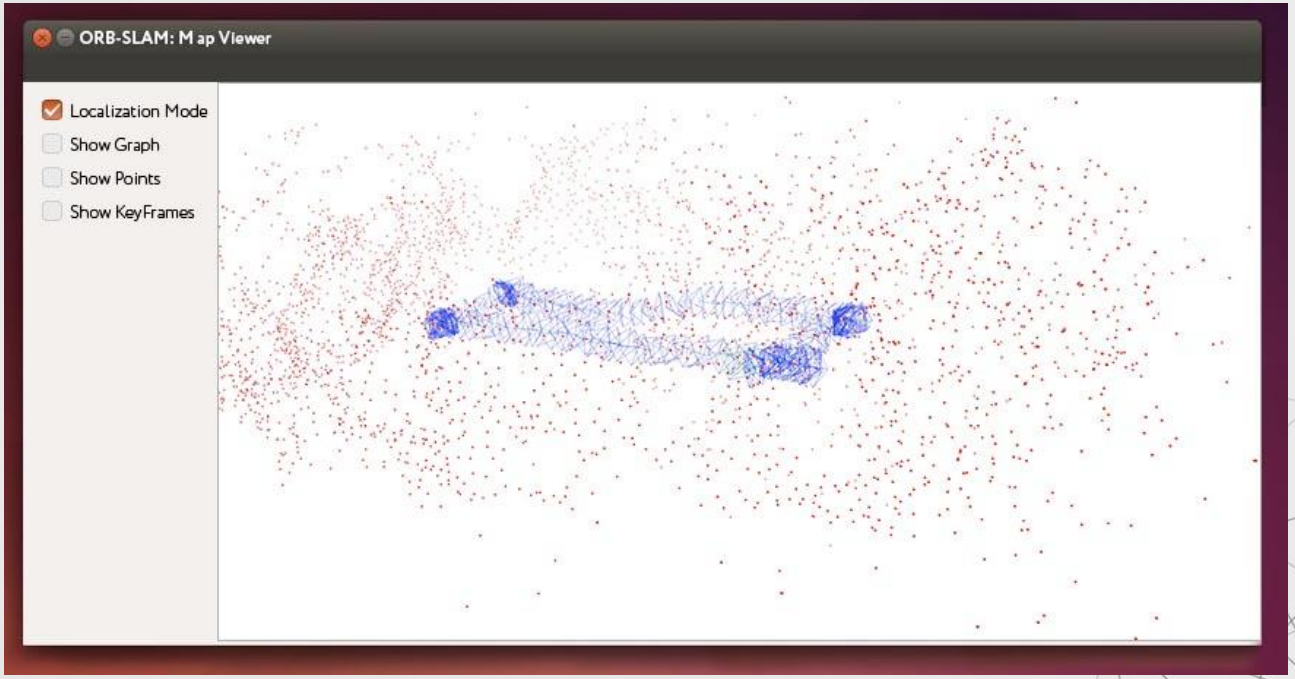


Рисунок 17 – Карта окружения и траектория

Анализ результатов

Получить 3D точку в системе координат кадра:

$$x = (u * f_{xi} + c_{xi}) * depth$$
$$y = (\mathrm{v} * f_{yi} + c_{yi}) * depth \quad (10)$$
$$z = dept * (1 + 2 * sqrt(f_{xi} + f_{yi} * f_{yi}))$$

где u, v – координаты точки на кадре, idepth – обратная глубина. fxi, fyi - инверсные фокусные расстояния камеры, cxi, cyi- инверсные положения центра камеры

Получить точку в глобальной системе координат:

Для этого необходимо умножить координаты точки в системе координат кадры, на матрицу преобразования – [R|t], где R – матрица поворота кадра, t – вектор смещения кадра.

Анализ результатов

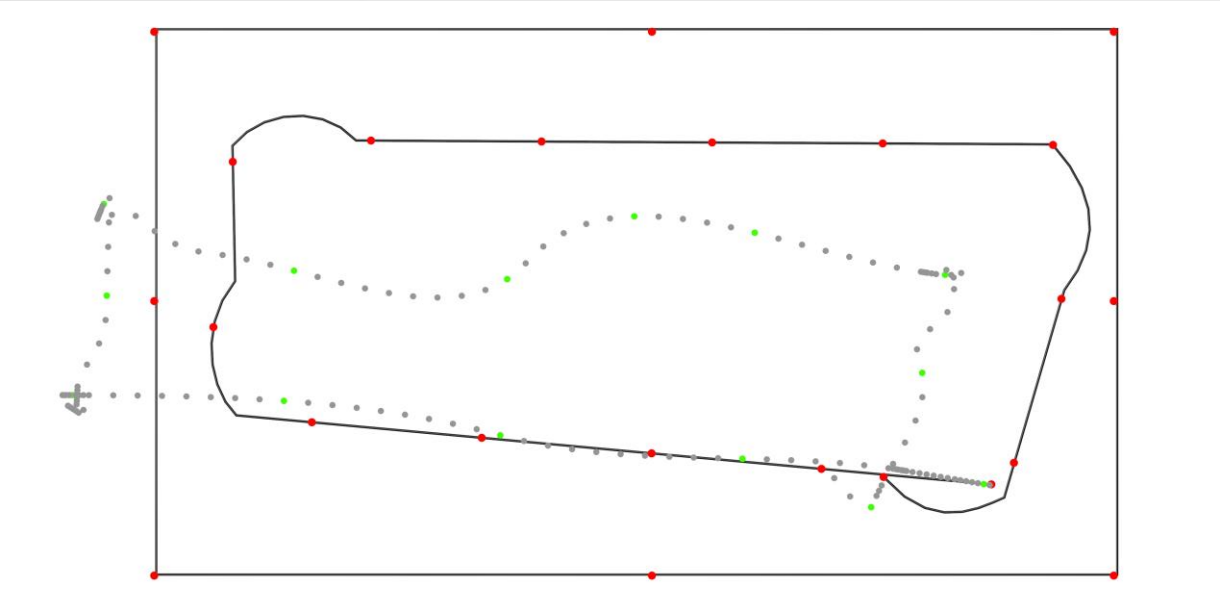


Рисунок 18 – Сравнение траекторий

--Время обработки ~1500 кадров составило 12 минут
--Максимальная ошибка составила: 35,168 см
--Минимальная ошибка составила: 2,149 см

СПАСИБО ЗА ВНИМАНИЕ!