Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Петрозаводский государственный университет»
Институт математики и информационных технологий

# Отчет о прохождении производственной практике

Выполнил студент группы 22307:
Устинов Данил Андреевич

Направление подготовки:
Системное и прикладное программное обеспечение

Место прохождения практики:
Кафедра ИМО, ЦИИ

Сроки прохождения практики:
30.05.2023-09.06.2023

Руководитель:
к.т.н., доцент
Богоявленская Ольга Юрьевна

Оценка _____

Дата _____

Петрозаводск — 2023

# Содержание

# Введение

Робототехника — это быстро развивающаяся прикладная наука, занимающаяся разработкой автоматизированных технических систем и являющаяся важнейшей технической основой развития производства. Основной целью создания роботов является замена человека машиной в опасных для его жизни профессиях. Помимо этого робот может быть использован в бытовых, производственных целях и не только. Например, роботы-пылесосы, роботы-сиделки и другие.

Наш проект по ТППО[1] как раз непосредственно связан с роботами, а именно колесными роботами-тележками. Целью нашего проекта - разработка библиотеки для шестиколесного робота, который с помощью датчиков будет получать информацию об окружающих объектах и классифицировать их, а также получать информацию о положении робота в пространстве.

Таким образом, задачами проекта были:

1. Разработка библиотеки для взаимодействия с датчиками;
2. Обработка полученных данных с датчиком;
3. Представление обработанных данных в виде графиков и карты местности.

При сборке робота были задействованы следующие компоненты:

1. Набор инерциальных датчиков - IMU 10dof;
2. Лидар (Light Detection and Ranging) - Hokuyo URG-04-LX-UG01;
3. Видеокамера - RDI Camera Module3;
4. Аккумулятор;
5. Манипулятор;
6. Raspberry PI 3 B;
7. Шестиколесная платформа;
8. Драйвер моторов.

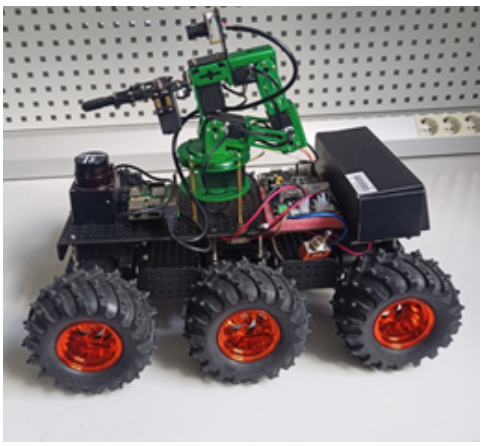


Рис. 1: Конечное состояние нашего робота

Библиотека была написана на Python.

Положение робота может определяться следующими характеристиками: креном (вращение вокруг оси X), тангажом (вращение вокруг оси Y) и рысканьем (вращение вокруг оси Z).

В нашем проекте, я как раз работал в том числе и с магнитометром (датчиком IMU).

**Цель:**

Описать решение проекта ТППО по работе с датчиком IMU и провести сравнение с существующими аналогами.

**Задачи:**

1) Описать полученное нами решение по работе с датчиком;

2) Сделать обзор аналогичных решений, как робота в целом, так и по работе с IMU;

3) Провести сравнение полученного нами решения и найденных аналогов.

# 1 Наше решение

## 1.1 Датчик

В нашем проекте мы использовали следующую плату – imu 10dof, которая включает в себя сразу несколько датчиков, а именно: акселерометр (adxl345), гироскоп (itg3205) и магнитометр (HMC5883L). Это те инерциальные датчики, с которых мы получаем и обрабатываем данные.

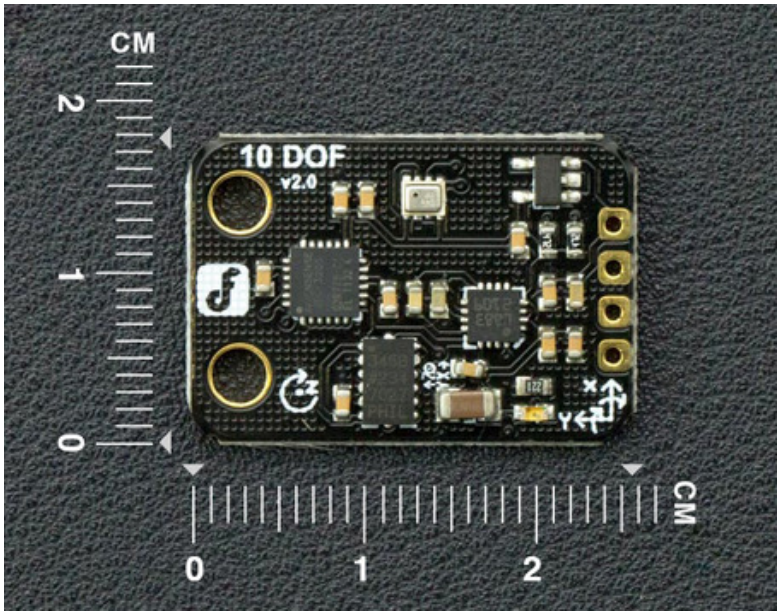



Рис. 2: Датчик imu

## 1.2 Получение данных

Данные с датчика HMC5883L получаем с помощью шины i2c.[2] Шина i2c – это последовательная ассиметричная шина для связи между интегральными схемами. Шина спользует две двунаправленные линии связи (SDA и SCL).

Для получения данных используем библиотеку smbus[3] на Python.

Настривать датчик для получения данных необходимо с помощью регистров, которые указаны в спецификациях (datasheet[4]), в так называемой карте регистров.

Для начала мы настраиваем конфигурационный регистр CRA: в этот регистр 0x01 мы записываем значение нормального режима измерения – 0x00.

Для непрерывного получения данных в регистр 0x02, отвечающий за режим измерения, записываем 0x00 – регистр непрерывного измерения.

Таким образом, мы настроили наш датчик на постоянное получение данных.

| Address Location | Name | Access |
|---|---|---|
| 00 | Configuration Register A | Read/Write |
| 01 | Configuration Register B | Read/Write |
| 02 | Mode Register | Read/Write |
| 03 | Data Output X MSB Register | Read |
| 04 | Data Output X LSB Register | Read |
| 05 | Data Output Z MSB Register | Read |
| 06 | Data Output Z LSB Register | Read |
| 07 | Data Output Y MSB Register | Read |
| 08 | Data Output Y LSB Register | Read |

Рис. 3: Карта регистров

Для того, чтобы получить значения необходимо считать данные начиная с регистра 0x03, заканчивая 0x08.

Данные мы получаем по трем осям: X, Y и Z. Одной оси соответствует два значения (то есть два регистра), которые складываются, как младший байт + старший байт, путем сдвига первого считанного значения на 8 бит.

Таким образом, регистры 0x03 и 0x04 соответствуют оси X, регистры 0x05 и 0x06 – оси Z и регистры 0x07 и 0x08 – оси Y.

## 1.3 Обработка данных

Полученные данные используются для нахождения рысканья, аналогично крену и тангажу (по сути ориентация робота в пространстве).[5] Ниже представлена схема, визуально иллюстрирующая, приведенные выше термины:

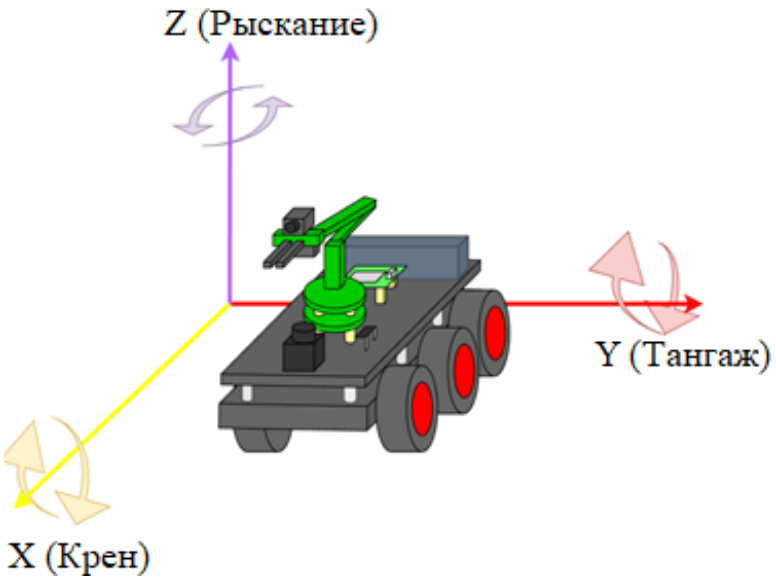



Рис. 4: Расположение крена, тангажа и рысканья

Рысканье (yaw) находится по следующей формуле[6]:

$$yaw = atan2(\frac{m_X}{m_Y}) \tag{1}$$

Функция atan2 берет значения из диапазона от -180 до 180 градусов.

Таким образом, берем «сырые» данные магнитометра по оси X и оси Y и рассчитываем рысканье. Ответ получаем в радианах, для того чтобы преобразовать значение в градусы, необходимо домножить его на 180 и поделить на $\pi$.

Причем, значение, равное нулю градусов, соответствует Северу.

Также, можно выводить значение от 0 до 360 градусов, для этого достаточно прибавлять к найденному значению 360, если оно отрицательно.

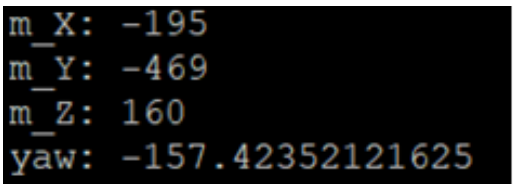



Рис. 5: Пример вывода «сырых» данных магнитометра и рысканья

## 1.4 Анализ полученного решения

К сожалению, ввиду того, что наш датчик установлен на металлическом корпусе робота и вблизи других работающих электронных компонентов, значения с магнитометра приходят с огромной погрешностью, по сути обработанные довольно сложно интерпретировать.

Тогда как в «руках» (отдельно от робота) датчик корректно меняет свои значения при совершении вращательных движений по оси Z из стороны в сторону.

# 2 Аналогичные решения

## 2.1 Сравнение с нашим решением по определению ориентации робота

Данная подглава будет включать в себя обзор и сравнение проекта «СИСТЕМА ЛОКАЛЬНОЙ НАВИГАЦИИ ДЛЯ НАЗЕМНЫХ МОБИЛЬНЫХ РОБОТОВ» от Черножкина В.А. и Половко С.А. (Центральный научно-исследовательский и опытно-конструкторский институт робототехники и технической кибернетики) с нашим решением.[7]

Здесь, как и в нашем проекте используется IMU датчики, только у них набор состоит из трех акселерометров и трех гироскопов, чтобы получать информацию обо всех шести степенях свободы мобильного робота.

С помощью них они определяют такие параметры, как курсовой угол, тангаж, крен, пройденный путь, скорость и ускорение ($\alpha$, $\beta$, $\gamma$, $s$, $v$, $w$). Мы же определяем только первые три параметра.

Также у них в работе учитываются такие особенности при движении колесных роботов, как:

1. Отсутствие бокового сноса;
2. Малые диапазоны изменения углов крена и тангажа робота за короткие промежутки времени;
3. Независимость угловой скорости вокруг вертикальной оси робота от угловых скоростей вокруг других осей;
4. Малая скорость движения робота.

В свою очередь, мы никак не учитываем эти особенности.

Таким образом, для определения угловой ориентации используются следующие формулы:

$$\alpha = \omega_z \tag{2}$$

$$\gamma = \omega_y \tag{3}$$

$$\beta = \omega_x \cdot Cos\gamma + \omega_z \cdot Sin\gamma \tag{4}$$

где $\alpha$, $\beta$, $\gamma$ – курсовой угол, тангаж и угол крена, $\omega_x$, $\omega_y$, $\omega_z$ – измеряемые в связанной системе координат (ССК)[8] угловые скорости мобильного робота;

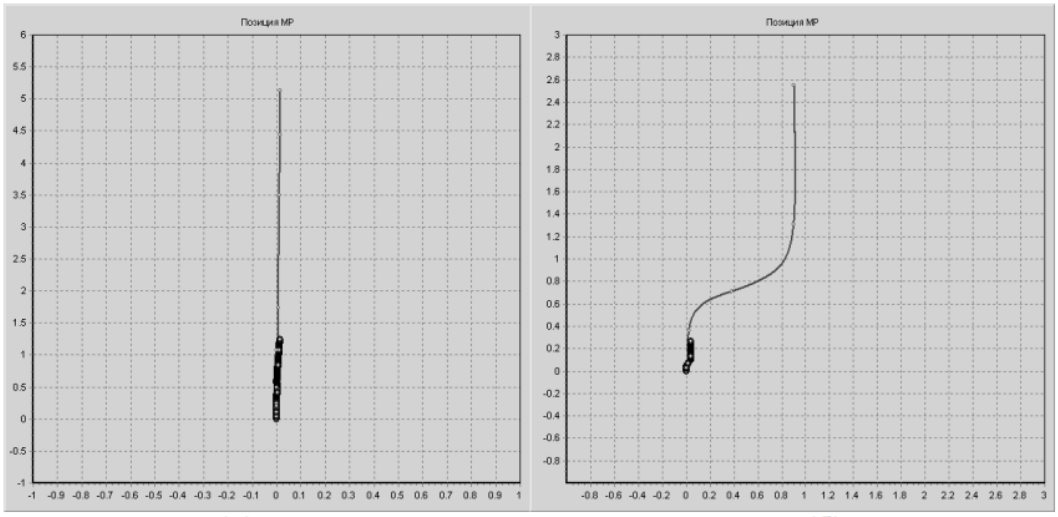
Здесь формулы также отличаются от наших, которые мы используем для определения крена и тангажа. Так для их расчета мы используем данные акселерометра.

Помимо всего прочего для уменьшения накопленной ошибки и других случайных погрешностей в данной работе используется обобщенный фильтр Калмана (ОФК)[9].

При тестировании модели они получили следующие результаты:

1. «Отработка углов ориентации объекта производится с точностью 0,1–0,3° (для крена и тангажа), 2–3° (для угла курса).»

2. «Погрешность определения местоположения объекта системой ЛН в автономном режиме составляет при прямолинейном движении $0,012t^2$ (1,2 м за 10 с), при маневрировании $0,026t^2$ (2,6 м за 10 с).»




(а) (б)

Рис. 3. Траектории объекта, полученные с помощью физической модели (темные кривые), в сравнении с реальными (светлые кривые) при: (а) – прямолинейном движении, (б) – маневрировании

Таким образом, на основании сравнения, можно сделать вывод, что мы погрузились в тему навигации (определение ориентациии и местоположения) не сильно глубоко, в т.ч. не использовали дополнительные фильтры для устранения погрешности, не вычисляли дополнительные параметры (путь, скорость и т.п.).

Также на примере этого проекта, можно попробовать вычислять курсовой угол (рысканье) не используя магнитометр, а используя набор из гироскопов и акселерометров.

## 2.2 Сравнение найденных решений с ТППО проектом

Также можно провести сравнение всего проекта ТППО с аналогичным проектом под названием «РАЗРАБОТКА РОБОТИЗИРОВАННОЙ ПЛАТФОРМЫ ДЛЯ ПОДЗЕМНОГО ГЕОМОНИТОРИНГА». Авторы: А.С. Лутонин, К.А. Богданова.[10]

Главная задача данного проекта - построение модели замкнутых пространств, где отсутствует возможность применения GNSS измерений, посредством комплексирования данных инерциального измерительного блока, датчиков угла поворота колес и лидара.

Они использовали следующие детали для построения своего робота:

1. 4-х колесная платформа на базе открытого проекта Smartcar Shield под управлением 32-х битного микроконтроллера ESP32;
2. Одноплатный компьютер Raspberry Pi 4;
3. Активный дальномер оптического диапазона (лидар);
4. Инерциальный измерительный блок (IMU).

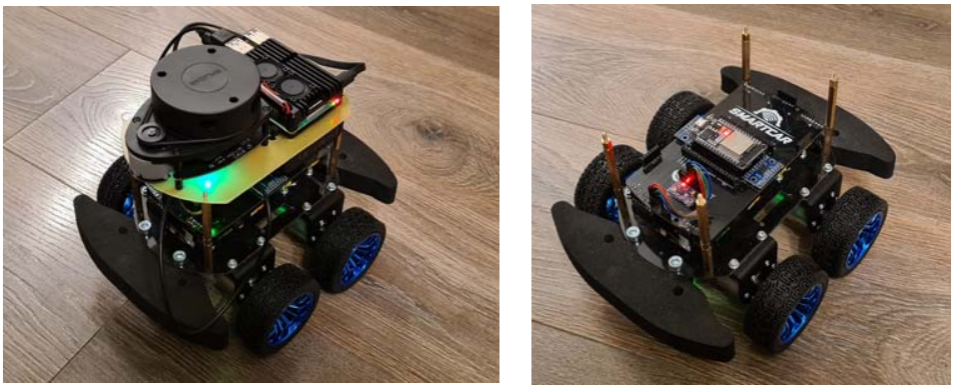


***Рис. 1. Общий вид мобильной передвижной платформы***

Краткое описание обработки данных: «Полученные с датчиков данные поступают на одноплатный компьютер по последовательному соединению, где, на основании их величин, а также информации с лидара, строится карта окружающей местности, а также формируется задание для каждого колеса мобильной платформы. Управление двигателями осуществляется через подключенный к ESP32 двухканальный мостовой (H-bridge) драйвер TB6612FNG. Параллельно, все данные с одноплатного компьютера поступают на ноутбук, который подключен к Raspberry PI4 по протоколу SSH через Wi-Fi соединение для их последующей обработки.»

Программный код был написан на C++, сбор данных осуществлялся с помощью Arduino IDE и ROS для передачи информации внутри проекта. Движение осуществлялось с помощью четырех электродвигателей.

IMU датчики использовались для позиционирования мобильного робота, в этом проекте также использовался фильтр Калмана для фильтрациии погрешностей.

В результате были проведены тесты построения траектории на основании разных датчиков.

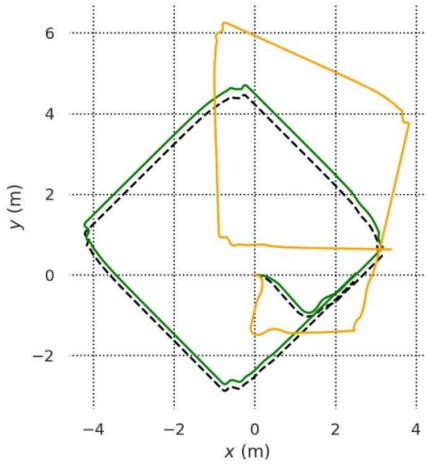



***Рис. 4. Траектории мобильной платформы: пунктирная линия – эталонная траектория, зеленая линия – траектория, построенная с помощью EKF, оранжевая линия – траектория, построенная на основании одометрических данных***

Наши проекты похожи прежде всего набором датчиков, за исключением камеры, так как прототип разрабатывался для подземного использования.

Также есть отличие в платформе, у нас она шестиколесная, в рассматриваемом аналоге - четырехколесная. Шестиколесная тележка более устойчивая, нежели четырехколесная. Так как при резком старте, робот с меньшим набором колес, может, к примеру, вставать на «дыбы».

Также мы не строим траекторию движения робота, а исключительно визуализируем данные (строим графики) с акселерометров (крен, тангаж, рысканье) и строим карту местности, которая каждый раз обновляет текущее положение предметов перед роботом.

Само движение в целом похоже и осуществляется с помощью нажатий клавиш на клавиатуре, правда в данном проекте использовался пакет teleop twist keyboard, входящего в состав ROS, у нас же использовался протокол I2C для связи между одноплатным контроллером и драйвером моторов.

Еще одна схожесть – это подключение к роботу посредством Wi-Fi по протоколу SSH, у нас это реализовано аналогичным образом.

В итоге, было проведено сравнение между нашим проектом по ТППО и проектом «РАЗРАБОТКА РОБОТИЗИРОВАННОЙ ПЛАТФОРМЫ ДЛЯ ПОДЗЕМНОГО ГЕОМОНИТОРИНГА».

# 3 Заключение

В ходе работы были выполнены следующие задачи:

1. Было представлено решение нахождения рысканья с помощью магнитометра;
2. Был проведен обзор и сравнение похожего проекта с нашим решением определения ориентации робота;
3. Был проведен обзор и сравнение похожего проекта с работой по ТППО.

# Список литературы


1. [Электронный ресурс] https://se.cs.petrsu.ru/wiki/Robot (Дата обращения:03.06.2023).

2. [Электронный ресурс] https://ru.wikipedia.org/wiki/I%C2%B2C (Дата обращения:04.06.2023).

3. [Электронный ресурс] https://pypi.org/project/smbus2/ (Дата обращения:04.06.2023).

4. [Электронный ресурс] https://cdn-shop.adafruit.com/datasheets/HMC5883L_3-Axis_Digital_Compass_IC.pdf (Дата обращения:04.06.2023).

5. [Электронный ресурс] https://rcsearch.ru/wiki/%D0%9A%D1%80%D0%B5%D0%BD,_%D0%A2%D0%B0%D0%BD%D0%B3%D0%B0%D0%B6,_%D0%A0%D1%8B%D1%81%D0%BA%D0%B0%D0%BD%D0%B8%D0%B5 (Дата обращения:04.06.2023).

6. [Электронный ресурс] https://habr.com/ru/articles/491476/ (Дата обращения:04.06.2023).

7. [Электронный ресурс] https://cyberleninka.ru/article/n/sistema-lokalnoy-navigatsii-dlya-nazemnyh-mobilnyh-robotov/viewer (Дата обращения:05.06.2023).

8. [Электронный ресурс] https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%8F%D0%B7%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%BE%D0%BE%D1%80%D0%B4%D0%B8%D0%BD%D0%B0%D1%82 (Дата обращения:05.06.2023).

9. [Электронный ресурс] https://en.wikipedia.org/wiki/Extended_Kalman_filter (Дата обращения:05.06.2023).

10. [Электронный ресурс] https://cyberleninka.ru/article/n/razrabotka-robotizirovannoy-platformy-dlya-podzemnogo-geomonitoringa (Дата обращения:05.06.2023).