Петрозаводский государственный университет
Институт математики и информационных технологий
Кафедра информатики и математического обеспечения

Направление подготовки бакалавриата

09.03.04 Программная инженерия

Отчет по научно-исследовательской практике

Применение датчика IMU в проекте «Robot»

Выполнил:
студент группы 22307

Д.А.Костин _____
подпись

Место прохождения практики:
Кафедра ИМО

Период прохождения практики:
29.05.23-11.06.23

Руководитель:
Богоявленская О.Ю.

_____
подпись

Итоговая оценка

_____
оценка

Петрозаводск — 2023

Содержание

Введение

«Robot» – название учебного проекта[1], разрабатываемого в рамках предмета ТППО (технология производства программного обеспечения).

Цель проекта «Robot» – разработать прототип малого колесного робота с интеллектуальной сенсорикой для труднодоступной и труднопроходимой местности.

Разработка проекта велась в интересах ЦИИ ПетрГУ, для которого одной из задач является разработка автономного лесопосадочного робота, способного передвигаться в труднопроходимой местности и анализировать своё состояние и состояние окружающей среды с помощью набора датчиков: датчик IMU[1], включающий акселерометр, гироскоп, магнитометр, датчик лидар и камера.

Задачами проекта «Robot» были:

- Разработка API на языке программирования Python для взаимодействия с датчиками.
- На основе данных датчика IMU вычисление крена, тангажа, рыскания.
- На основе данных лидара построение карты местности.
- Разработка способов управления роботом.
- Получение видео с камеры на роботе по сети удалённо.

На рисунке 1 представлена модель разрабатываемого прототипа робота.

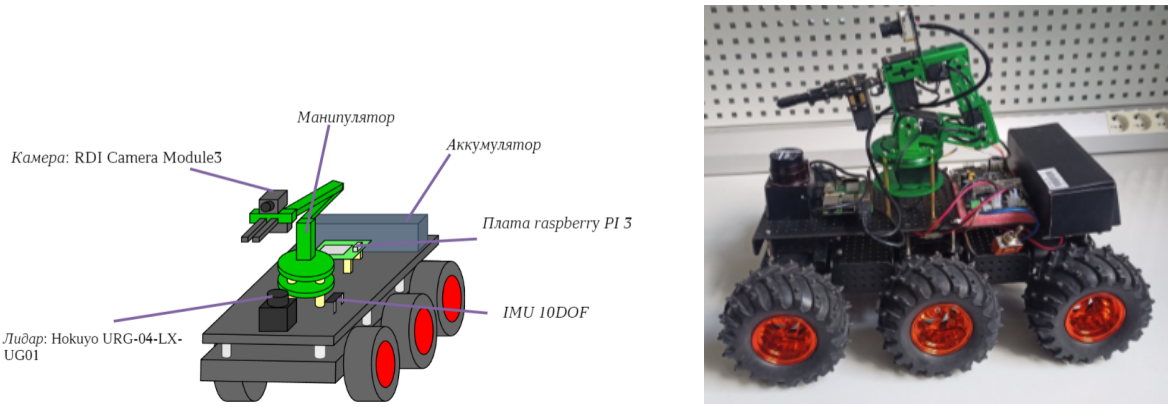

Рис. 1 – Слева – схематическая модель робота, справа – конечный результат.

[1]Inertial measurement unit – инерциальный измерительный прибор

В ходе проекта пришлось столкнулась с отсутствием опыта работы с одноплатным компьютером – *Raspberry Pi Model 3*, и датчиком *IMU 10dof*, а именно:

- Отсутствовало понимание, как подключить датчик IMU к Raspberry.
- Не было ясно, какие программные модули и методы использовать для считывания данных с датчиков.

Таким образом, **целью практики** является описание актуальных результатов решения задачи по разработке прототипа малого колёсного робота, но не в рамках всего проекта по ТППО, а ограниченных рамками работы с датчиком IMU.

Задачами практики являются:

- Представление решения по подключению датчика IMU.
- Представление решения проблемы получения данных с датчика.
- Изучение и сравнение аналогов проекта, связанных с разработкой автономных роботов и получения данных с датчиков.

1 Описание проектных решений

1.1 Подключение датчика IMU

Перед тем, как начать описание метода подключения датчика IMU 10dof к Raspberry, необходимо рассказать о интерфейсе ввода/вывода (general-purpose input/output, **GPIO**) платы.

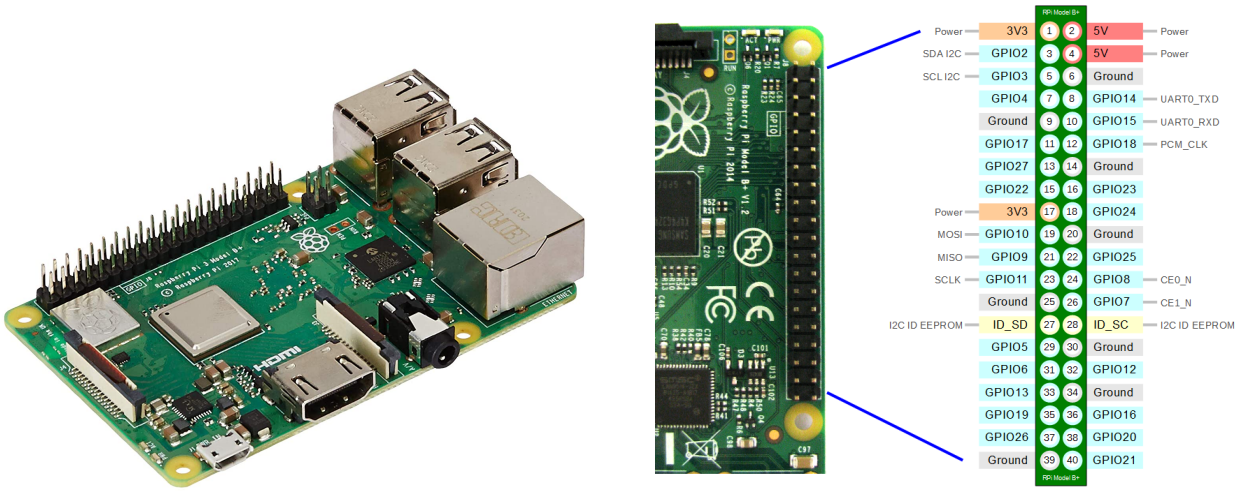

Рис. 2 – Интерфейс ввода/вывода платы Raspberry Pi Model 3

На рисунке 2 представлена так называемая «распиновка» платы – набор из 40 пинов, предназначенных для подключения различных компонент к плате, например, датчики, контроллеры, дисплеи и т.д.

Пины в GPIO могут выполнять 3 функции:

- Подача электричества напряжения в 3.3 или 5 Вольт (маркируются как Power).
- Заземление (могут именоваться RND или Ground). Они нужны, чтобы отводить электричество.
- Прием/отправка сигналов (все оставшиеся пины).

Конечно же для подключения датчиков и получения данных от них наибольший интерес представляют пины типа приём и отправка сигналов. К таким пинам относятся пины номер 3 и 5 на рисунке 2 – соответственно, их названия SDA и SDL, вместе образующие интерфейс *I2C*[2], который и применялся для подключения датчика к плате в проекте.

I2C – последовательная асимметричная шина для связи между интегральными схемами внутри электронных приборов, разработана фирмой Philips Semiconductors в начале 1980-х.

В ней используются две двунаправленные линии с открытым сливом, последовательная линия передачи данных (SDA) и последовательная линия синхронизации (SCL). В обмене сообщениями участвует так называемый *master* и от одного и более *slave*.

Каждый сеанс обмена начинается с подачи master стартового сигнала S (называемого стартовым битом). Стартовый бит — это изменение уровня на линии SDA с высокого на низкий, но только при наличии высокого уровня на линии SCL. И наоборот – изменение уровня на линии SDA с низкого на высокий при наличии высокого уровня на линии SCL является стоп-сигналом P, означающим конец сеанса связи. Все, что происходит между этими событиями, называется сообщением, что и есть передача данных.

На рисунке 3 представлена схема передачи данных по шине I2C.

Рис. 3 – Схема передачи данных по шине I2C.

Таким образом, для подключения датчика IMU к Raspberry, была применена шина I2C. На рисунке 4 представлена используемая в проекте схема подключения, красным цветом представлена линия подачи питания 3.3 вольта, чёрным – заземление, фиолетовым и оранжевым – соответственно, линия SCL и SDA.

Причины, из-за которых было решено применение шины I2C для получения данных с датчика IMU, стали:

1. I2C – естественный способ подключения внешних устройств к Raspberry.
2. Подключение осуществляется напрямую, без дополнительных цепей, что облегчает модификацию и замену устройств на шине.
3. Устройства на шине имеют свой адрес, для обмена сообщениям установлен протокол, в связи с чем разрабатываемая система может быть программно определяемой.

4. Разработка программного обеспечения упрощается наличием программных библиотек.

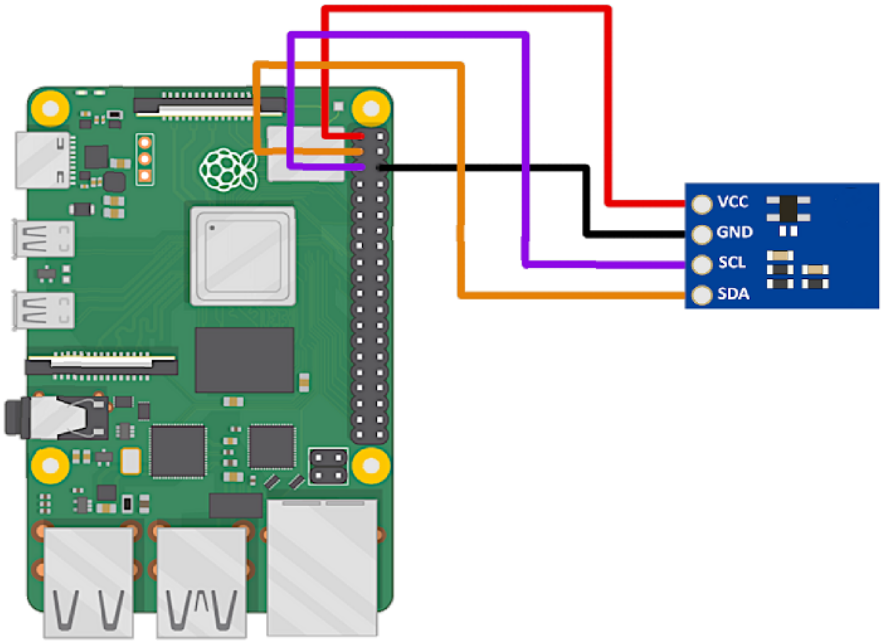

Рис. 4 – Схема подключения датчика к плате. Синим цветом выделен IMU 10dof

1.2 Устройство акселерометра

На рисунке 5 представлен датчик IMU 10dof, который содержит в себе трехмерную координатную ось XYZ (на рисунке справа снизу), которая используется каждым встроенным в IMU датчиком (акселерометром, гироскопом, магнитометром) для замера показаний для каждой из осей.

Особое внимание в данном разделе будет уделено акселерометру, так как данный датчик успешно применялся в проекте для вычисления крена и тангажа робота-тележки.

Перед тем, как приступить к описанию результатов работы с акселерометром, нужно рассказать о его программном устройстве. Название акселерометра – «ADXL345»[3], для получения данных с датчика по шине I2C, необходимо обращаться к определённому *регистру* акселерометра.

Акселерометр «ADXL345» на шине I2C имеет адрес 0x53 в шестнадцатеричной системе счисления, который разбит на 57 регистров от 0x00 до 0x39 (обычно все адреса и регистры на шине I2C записываются в шестнадцатеричной системе счисления), каждый

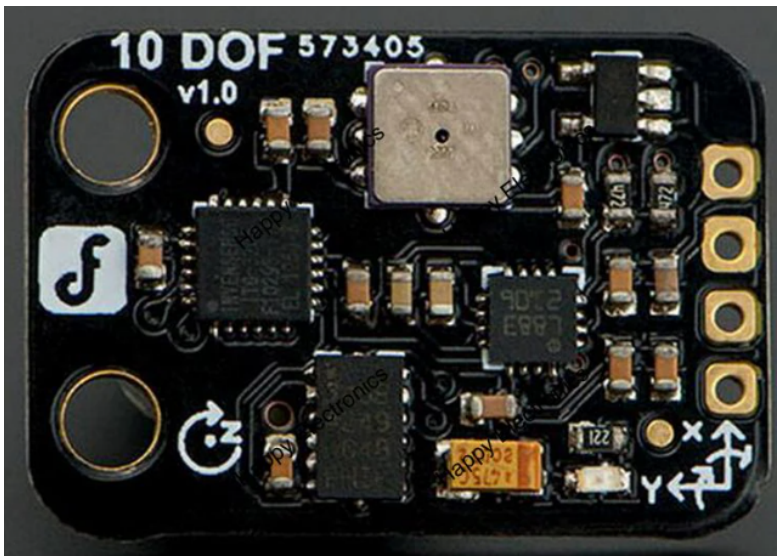

Рис. 5 – Датчик IMU 10dof

из которых хранит в себе 1 байт информации, который соответствует карте регистров акселерометра[3].

Для получения информации по каждой из осей X,Y,Z необходимо обратиться к двум регистрам, хранящим соответствующие значения «верхнего» и «нижнего» байта. На рисунке 6 представлена вырезка из документации, описывающая, какие регистры хранят информацию о ускорении на какой из осей.

0x32	50	DATAX0	R	00000000	X-Axis Data 0.
0x33	51	DATAX1	R	00000000	X-Axis Data 1.
0x34	52	DATAY0	R	00000000	Y-Axis Data 0.
0x35	53	DATAY1	R	00000000	Y-Axis Data 1.
0x36	54	DATAZ0	R	00000000	Z-Axis Data 0.
0x37	55	DATAZ1	R	00000000	Z-Axis Data 1.

Рис. 6 – Датчик IMU 10dof

Для получения ускорения по конкретной оси, например, оси X, необходимо объединить значения регистра DATAX1 и DATAX0 в одно двухбайтовое число. Аналогично получаются значения по осям Y и Z.

1.3 Программные решения

Разобравшись с тем, как устроен датчик и куда в нём обращаться для получения измеренных значений, необходимо разобраться, как обращаться к его регистрам.

Так как разработка программной системы велась на языке Python, то стояла задача

найти модули для взаимодействия с датчиком по шине I2C. Таким модулем стал *smbus2*[4], который предоставляет команды для записи байт в регистры шины и чтение регистров.

В листинге 1 представлен пример считывания данных с датчика по оси X и преобразование в виде ускорения, измеряемого в g (ускорение свободного падения).

Листинг 1 – Пример получения ускорения по оси X

```
from smbus2 import SMBus

#IMU is in first bus
bus = SMBus(1)

#Read 2 bytes starting from register 0x32 into array
data = bus.read_i2c_block_data(self.register, 0x32, 2)

#Build 2-byte number
xAccl = (data[1] & 0x03) * 256 + data[0]
    if xAccl > 511 :
        xAccl -= 1024
        #Get acceleration in g
        xAccl /= 256
```

Так как в задачу проекта входило создание API для удобного оперирования показаниями датчика IMU, то в его основу лёг модуль smbus2, поверх которого с использованием объектно-ориентированного программирования были написаны классы *Acclerometer, Gyroscope, Magnitometer*[5].

На основе полученных от акселерометра данных производился расчёт угла крена (roll) и тангажа (pitch)[6] по следующим формулам:

$$\text{roll} = \text{atan}(\frac{yAccl}{zAccl}) * 180/\Pi$$

$$\text{pitch} = \text{atan}(\frac{-xAccl}{\sqrt{yAccl^2 + zAccl^2}}) * 180/\Pi$$

Для передачи оператору робота информации о текущей ориентации робота – текущие

углы крена и тангажа, было разработано графическое приложение построения графиков этих двух величин в моменты времени с заданной частотой на основе библиотеки *Matplotlib*. На рисунке 7 представлен интерфейс приложения.

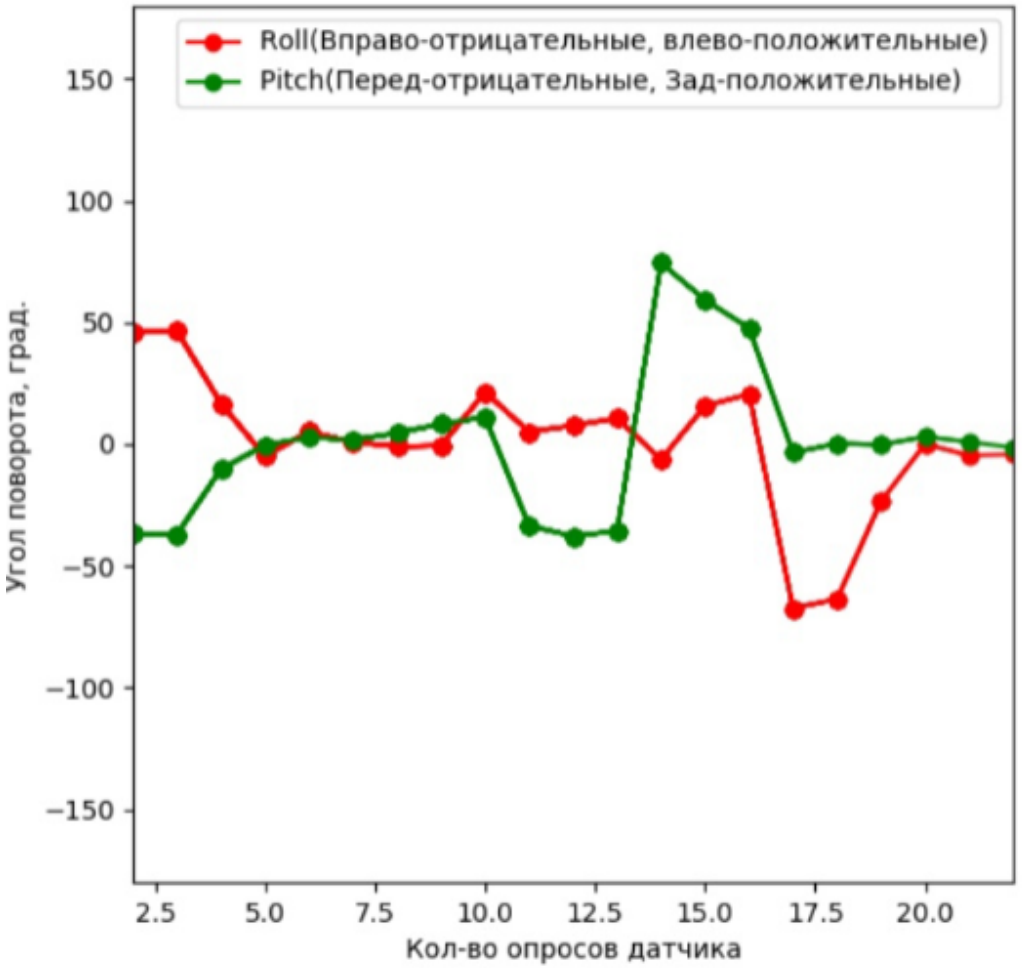

Рис. 7 – Приложение для визуализации углов крена и тангажа.

Одной из функциональностей, которую не внедрили в проект, является фильтрация сырых данных с акселерометра для сглаживания получаемых значений, которые могут искажаться, например, при тряске. Причинами этого можно выделить:

- Визуализация значений углов не может происходить достаточно быстро (например, 10 раз в секунду) в силу малых вычислительных ресурсов Raspberry.
- Из-за малой скорости визуализации сглаживание путём фильтрации могло бы не достигнуть пикового значения, поэтому терялась информация о реальном угле ориентации тележки.

- На графике визуализации было принято выводить значение крена и тангажа полностью основываясь на текущих данных.
- Во всех остальных случаях значения углов крена и тангажа выводятся достаточно точно, с максимальной погрешностью в 3 градуса.

1.4 Калибровка датчика

Ещё одним немаловажным решением стала простая калибровка датчика акселерометра. Известно[3], что значения акселерометра, лежащего горизонтально в состоянии покоя, по осям X, Y, Z должны равняться 0g, 0g, 1g, то есть единственная действующая сила на датчик – это сила тяготения, как раз равная g и направленная по оси Z.

Для калибровки применялась запись в регистры отступов (offset) постоянные значения, на которые должны измениться считываемые акселерометром данные, чтобы в состоянии покоя значения равнялись эталонным.

Для нахождения значений для регистров отступов производился подсчёт среднего арифметического 10 значений показаний датчика акселерометра по каждой из осей, обозначим их $\theta_x,\ \theta_y,\ \theta_z$.

Затем, в соответствующий регистр отступа записывалась разница между средним значением и эталонным, деленным на 4, то есть:

$$\text{off}_x = \frac{\theta_x - 256}{4}$$
$$\text{off}_y = \frac{\theta_y - 256}{4}$$
$$\text{off}_z = \frac{\theta_z - 256}{4}$$

2 Сравнение с аналогами

2.1 Прототип автоматизированной системы для перевозки грузов

Данная глава будет включать в себя обзор проекта аспирантов Южно Уральского государственного университета по созданию прототипа автоматизированной системы для перевозки грузов[8]. Цель данного обзора – понять и сравнить принципы автономной работы обозреваемого и разработанного проекта.

Проект прототипа автоматизированной перевозки грузов призван упростить рабочий процесс и сэкономить деньги на водителе транспортного средства. Для этого будет достаточно одного диспетчера, который будет контролировать сразу несколько транспортных тележек. На рисунке 8 представлена описанная схема работы такой системы.

Рис. 8 – Принцип работы автоматизированной транспортной системы.

Использованное в проекте прототипа (рисунок 9) оборудование:

- Четырёхколёсная несущая платформа с тяговыми электроприводами.
- Модуль питания.
- Микропроцессорный модуль управления и навигации.
- Электромеханическая система наведения, предназначенная для поворота камеры.

- Система сбора данных с датчиков.
- Бортовой компьютер в виде основного вычислительного устройства.
- Устройство для зарядки батарей от электрической сети.

Компонентно прототип транспортной тележки похож на прототип проекта Robot. Отдельно отметим, что прототип тележки для вращения камеры использует специальную электромеханическую систему наведения для вращения камеры, что достаточно для задачи сбора видеоинформации об окружающей среде. В проекте Robot тележка реализует обзор камеры на 360 градусов посредством прикрепления её к манипулятору, который дополнительно может применяться в задаче посадки саженцев.

Ещё одним отличием двух прототипов является намеренное использование шести колёсной тележки в проекте лесного робота, что обеспечивает стабильность, отсутствие переворотов назад при резком старте из-за наличия тяжёлой батареи в задней части.

Рис. 9 – Прототип транспортной робот-тележки.

Основным вычислительным устройством прототипа является микроконтроллер *Arduino Mega2560*, который является сравнительно более слабым, чем используемый в проекте Robot микрокомпьютер Raspberry.

Для сбора информации из окружающего мира, прототип использует датчики:

- Ультразвуковых дальномеров.

- Энкодеров.
- Инфракрасных датчиков цвета.

Передача данных от прототипа к оператору осуществляется без проводов с использованием *Bluetooth модуль HC-05*. На рисунке 10 представлена схема подключения модуля Bluetooth к Arduino посредством интерфейса *UART*.

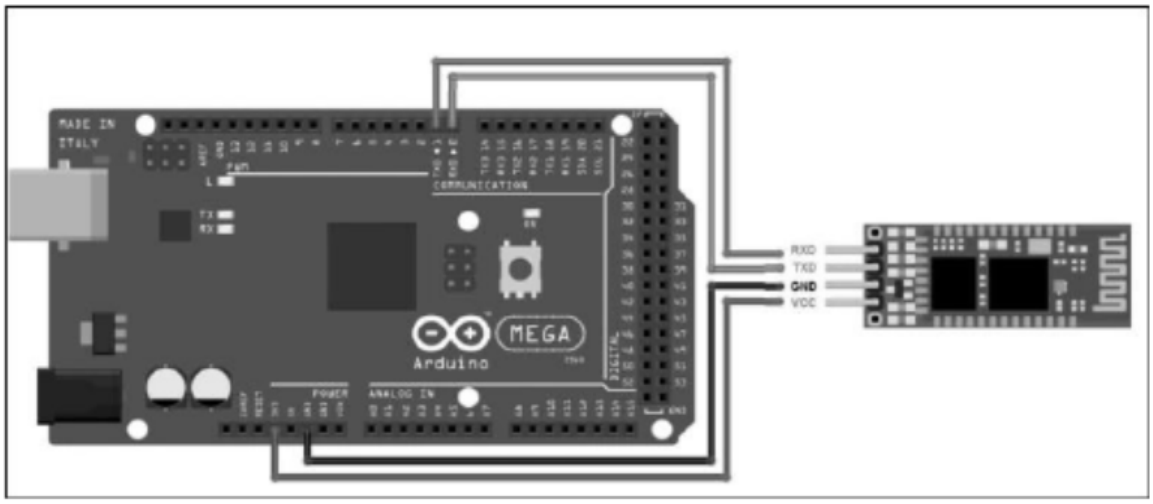

Рис. 10 – Схема подключения модуля Bluetooth к Arduino.

Протокол UART[9] (Universal asynchronous receiver/transmitter) использует два пина контроллера - RX и TX, где первые буквы обозначают, соответственно, Receiver и Transmitter. Для подключение двух устройств по данному протоколу необходимо соединить RX пин первого устройства с TX пином второго. Отличием протокола UART от I2C является то, что UART обеспечивает параллельную передачу данных между двумя устройствами, инициатором передачи может являться любое из соединённых устройств, тогда как в I2C передача последовательная, в одно время от одного устройства к другому, инициатором передачи является master.

В проекте Robot для передачи информации от робота к оператору используется сеть Wifi с использованием протокола SSH. В связи с этим управление роботом и получение данных зависит от наличия обоих – и оператора, и робота в одной сети, поэтому передвижение робота ограничивается зонами работы Wifi.

Разработчики проекта транспортной тележки планируют установку более мощных вычислительных мощностей, а именно микрокомпьютер Raspberry, а также внедрение лидара и алгоритмов компьютерного зрения. В проекте Robot уже были попытки внедрения ал-

горитмов компьютерного зрения, результатом которых стал вывод, что мощностей платы Raspberry для этой задачи не хватает.

В результате было проведено сравнение двух проектов – лесного и транспортного роботов-тележек, в ходе которого было подчёркнута технология Bluetooth, которую можно применить в проекте Robot для создания пульта управления, не зависящего от сети Wifi.

2.2 Прототип робота для автономного движения по данным инерциального датчика

Данная глава будет включать в себя обзор проекта студента Ижевского государственного технического университета имени М. Т. Калашникова по созданию автономного робота, движение которого осуществлялось бы по заданной траектории с использованием инерциальной навигации[7]. Особый интерес представляет математическое описание модели для вычисления скорости и пройденного расстояния. Цель обзора – сравнить принципы использования датчика IMU в двух проектах.

Навигация с использованием инерциальных датчиков имеет преимущества над системами глобального позиционирования, например, GPS, или системами оптического позиционирования в неблагоприятных погодных условиях – дождь, туман, ветер. Проект робота для автономного движения использует для этого показания датчиков акселерометра и гироскопа.

На рисунке 11 представлена схема робота, который представляет собой тележку, состоящую корпуса и двух колёс,

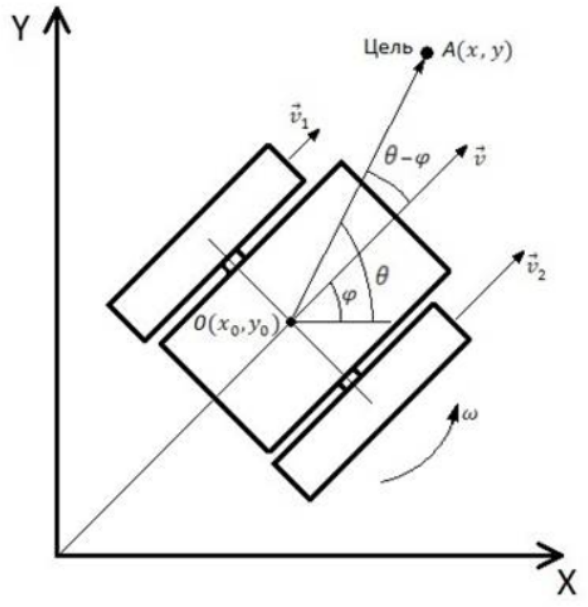

Рис. 11 – Модель автономного робота.

где v_1, v_2 – линейные ускорения колёс, v – линейная курсовая скорость, θ – угол между целью и осью X, ϕ – угол между курсом и целью, ω – угловая скорость.

Основными используемыми компонентами в проекте были:

- Датчик ускорения – акселерометр, и датчик угловой скорости – гироскоп, объединённые в *IMU BNO055*.
- Вычислительное устройство – *Arduino MKR1000 Wi-Fi*.

Проект столкнулся с проблемой, что при изменении углового положения акселерометра ускорение свободного падения проецируется на оси OX и OY, что негативно сказывается при измерениях линейного перемещения. Возникает необходимость компенсировать гравитационную составляющую (ускорение свободного падения) в данных датчика. Проект автономного робота предлагает решить это использованием фильтра нижних частот:

$$g(t) = \beta \cdot g(t-1) + (1-\beta) \cdot a(t)$$
$$a(t) = a(t-1) - g(t),$$

где $a(t)$ – ускорение (показания акселерометра) по любой из осей в момент времени t, $g(t)$ – переменная для компенсации гравитационной составляющей в момент времени t, β — коэффициент скорости компенсации,

Для получения ускорения в плоскости XY складываются ускорения $a(t)_x$ и $a(t)_y$:

$$a(t) = \sqrt{a(t)_x^2 + a(t)_y^2}$$

Для вычисления скорости V и пройденного расстояния за всё время S применяются формулы:

$$V(t) = V(t-1) + a(t) \cdot dt$$
$$S(t) = S(t-1) + V(t) \cdot dt$$

Для вычисления текущего угла поворота ϕ используется гироскоп, для его вычисления применяется следующая формула:

$$\phi(t) = \phi(t-1) + \omega(t) \cdot dt,$$

где $\omega(t)$ – угловая скорость вокруг оси Z (то есть поворот в плоскости XY), dt – временной промежуток между считываниями, то есть между моментами времени t и t-1.

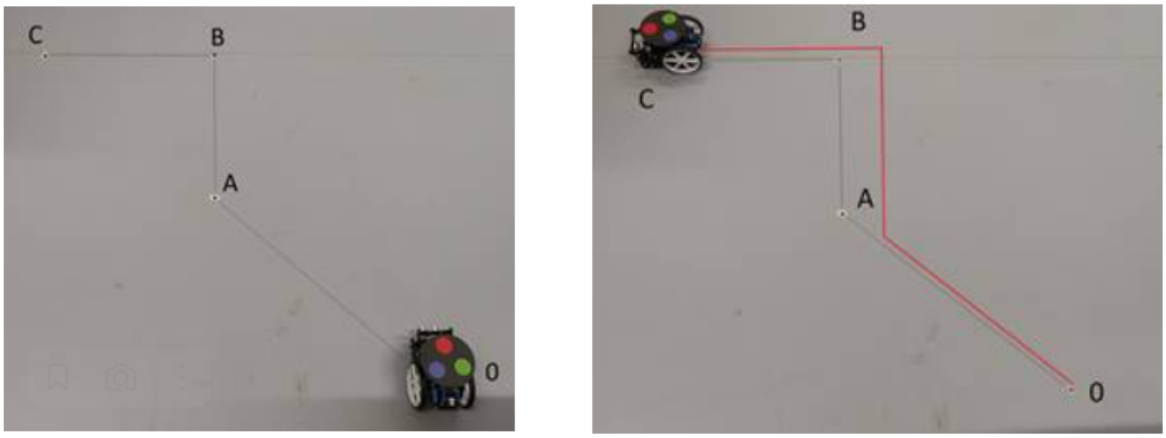

Рис. 12 – Эксперимент по автономному движению.

На рисунке 12 представлен эксперимент, проведённый разработчиками автономного робота с использованием математических формул.

Таким образом, применение датчика IMU в проекте автономного робота пошло дальше, чем в проекте лесного робота: на основе данных датчиков производилась оценка местоположения тележки, посредством отложения угла поворота и пройденного расстояния от начальной точки, в то время как в проекте Robot основной упор был сделан на понимание ориентации тележки в пространстве и использование, помимо акселерометра, камеры и лидара. При этом данные гироскопа не использовались.

Как было сказано в главе описания решений проекта Robot (Глава 1.3), в проекте отсутствовала система фильтрации сырых данных, полученных от датчиков. В проекте автономного робота фильтрация необходима для компенсации гравитационной составляющей в данных датчика.

Автономный робот для задачи передвижения полностью полагается на датчик IMU, благодаря чему от способен самостоятельно лишь на основании его данных произвести поворот и движение в направлении, тем самым отпадает необходимость в управлении оператором. Проект лесного робота находится на стадии удалённого управления – по нажатию клавиш робот будет ехать с определённой скоростью или останавливаться, но разработка в конечном счёте будет сводится к автономному передвижению.

В результате был проведён обзор и сравнение двух проектов – автономного и лесного роботов. В ходе сравнения было отмечено, что использование датчика IMU в проекте Robot может пойти дальше: совмещение данных датчиков акселерометра и гироскопа – одно из возможных решений для реализации автономного движения в проекте Robot.

Заключение

В ходе *работы по проекту Robot*, удалось решить ряд задач:

- Была собрана робот-тележка с прикреплёнными и подключёнными датчиками.
- Был реализован API на языке программирование Python.
- На основе данных датчиков, была реализована визуализация углов ориентации.

В рамках отчёта:

- Были представлены актуальные результаты решения задачи работы с датчиком IMU.
- Было проведено сравнение с аналогами проекта малого колёсного робота для труднопроходимой местности, в результате которого были выявлены направления, по которым можно произвести доработку и улучшение результатов.

Список литературы

1. Страница проекта Robot // Wiki ресурс проектов ТППО : [сайт]. URL: https://se.cs.petrsu.ru/wiki/Robot (дата обращения: 03.06.2023)

2. Михайлов С.А., Лохов А.К. ОПИСАНИЕ ПРОТОКОЛА УПРАВЛЕНИЯ ШИНОЙ I2C // Научное сообщество студентов: МЕЖДИСЦИПЛИНАРНЫЕ ИССЛЕДОВАНИЯ: сб. ст. по мат. XXXV междунар. студ. науч.-практ. конф. № 24(35). URL: https://sibac.info/studconf/science/xxxv/92371 (дата обращения: 03.06.2023)

3. Документация акселерометра ADXL345 [Электронный ресурс] // Compel : [сайт]. URL: https://www.compel.ru/pdf-items/dfrobot/pn/10-dof-mems-imu-sensor/a5b48881a962565250dd590c9f2046c0 (дата обращения: 03.06.2023)

4. Документация по библиотеке smbus2 [Электронный ресурс] : Индекс пакета Python : [сайт]. URL: https://pypi.org/project/smbus2/ (дата обращения: 03.06.2023)

5. Код проекта на GitLab : [сайт]. URL: https://dev.cs.petrsu.ru/ustinov/robot (дата обращения: 03.06.2023)

6. МЭМС акселерометры, магнитометры и углы ориентации // Сообщество IT специалистов : [сайт]. URL: https://habr.com/ru/articles/491476/ (дата обращения: 03.06.2023)

7. Файздрахманов Д.Г. ИСПОЛЬЗОВАНИЕ АКСЕЛЕРОМЕТРА И ГИРОСКОПА ДЛЯ АВТОНОМНОГО ДВИЖЕНИЯ МОБИЛЬНОГО РОБОТА // Научное сообщество студентов XXI столетия. ТЕХНИЧЕСКИЕ НАУКИ: сб. ст. по мат. LXXIX междунар. студ. науч.-практ. конф. № 7(78). URL: https://sibac.info/studconf/tech/lxxix/148775 (дата обращения: 01.06.2023)

8. Черепанов Павел Юрьевич, Романов Пётр Алексеевич Разработка автоматизированной системы для перевозки легких грузов // Наука, техника и образование. 2017. №1 (31). URL: https://cyberleninka.ru/article/n/razrabotka-avtomatizirovannoy-sistemy-dlya-perevozki-legkih-gruzov (дата обращения: 03.06.2023).

9. Geoffrey Hunter, UART Communication Protocol // The embedded engineering website : [сайт]. URL: https://blog.mbedded.ninja/electronics/communication-protocols/uart-communication-protocol/ (дата обращения: 05.06.2023)