МБОУ Школа №10 «Успех» г.о. Самары

Исследование систем инерциальной навигации на примере робота-разведчика.

Выполнил:

Алексеев Егор

ученик 10 «А» класса

МБОУ СОШ № 10 «Успех»

г.о. Самара

Научный руководитель:

Бражников Артем Максимович

Самара, 2018

Аннотация и ключевые слова

РОБОТ, ИНЕРЦИАЛЬНЫЕ СИСТЕМЫ, НАВИГАЦИЯ, ДАТЧИК, СХЕМОТЕХНИКА, ПРОГРАММНАЯ РЕАЛИЗАЦИЯ.

В проекте рассмотрены принципы управления роботами в инерциальных системах навигации (без сигналов из внешнего мира) с детальным рассмотрением датчиков, выделением элементарной базы. Реализованы принципы инерциальной навигации, основанные на данных о пройденном расстоянии, угле ориентации прототипа в пространстве. На базе микроконтроллера семейства AVR были реализованы алгоритмов локального позиционирования роботов для решения определенных задач в условиях полной автономии. Был описан метод, являющийся довольно перспективным для различных видов транспорта, постоянно находящихся в условии ограниченной видимости или не имеющих доступа к системам навигации, связанными с поступающими извне сигналами.

Оглавление

[Введение 3](#_Toc510328966)

[Отработка методов инерциальных систем навигации 5](#_Toc510328967)

[Описание применяемых датчиков 6](#_Toc510328968)

[Описание особенностей использования применяемых датчиков 8](#_Toc510328969)

[Программное обеспечение, реализующее алгоритм 14](#_Toc510328970)

[Заключение 17](#_Toc510328971)

[Библиографический список 19](#_Toc510328972)

[Приложение 1 20](#_Toc510328973)

[Приложение 2 21](#_Toc510328974)

# Введение

Тема: Исследование систем инерциальной навигации.

Инерциальная навигация — метод навигации (определения координат и параметров движения различных объектов — судов, самолётов, ракет и др.) и управления их движением, основанный на свойствах инерции тел, являющийся автономным, то есть не требующим наличия внешних ориентиров или поступающих извне сигналов.[[1]](#footnote-1)

Сущность инерциальной навигации состоит в определении [ускорения](https://ru.wikipedia.org/wiki/%D0%A3%D1%81%D0%BA%D0%BE%D1%80%D0%B5%D0%BD%D0%B8%D0%B5) объекта и его [угловых скоростей](https://ru.wikipedia.org/wiki/%D0%A3%D0%B3%D0%BB%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%BA%D0%BE%D1%80%D0%BE%D1%81%D1%82%D1%8C) с помощью установленных на движущемся объекте приборов и устройств, а по этим данным — местоположения (координат) этого объекта, его курса, скорости, пройденного пути и др., а также в определении параметров, необходимых для стабилизации объекта и автоматического управления его движением. Это осуществляется с помощью:

1. датчиков линейного ускорения ([акселерометров](https://ru.wikipedia.org/wiki/%D0%90%D0%BA%D1%81%D0%B5%D0%BB%D0%B5%D1%80%D0%BE%D0%BC%D0%B5%D1%82%D1%80));
2. [гироскопических](https://ru.wikipedia.org/wiki/%D0%93%D0%B8%D1%80%D0%BE%D1%81%D0%BA%D0%BE%D0%BF) устройств, позволяющих определять углы поворота и наклона объекта, используемые для его стабилизации и управления движением.
3. вычислительных устройств ([ЭВМ](https://ru.wikipedia.org/wiki/%D0%AD%D0%92%D0%9C)), которые по ускорениям находят скорость объекта, его координаты и др. параметры движения;

Преимущества методов инерциальной навигации состоят в автономности, помехозащищённости и возможности полной автоматизации всех процессов навигации. Благодаря этому методы инерциальной навигации получают всё более широкое применение при решении проблем навигации надводных, подводных и воздушных судов, космических судов и аппаратов и других движущихся объектов.

Может быть использована в контроле местонахождения транспорта, постоянно находящегося в условиях ограниченной видимости или не имеющего доступа к системам навигации, связанными с поступающими извне сигналами. Преимущества методов инерциальной навигации состоят в автономности, помехозащищённости и возможности полной автоматизации всех процессов навигации.

Планируется создать робота, определяющего свое положение в пространстве с помощью комплекса датчиков. Используя только те датчики, что есть на роботе, мы стараемся определить его траекторию движения, путь, скорость, ближайшее окружение и на основании этих данных исследовать маршрут его движения.

Объектом моего исследования является использование алгоритмов локального позиционирования роботов.

Предметом моего исследования является реализация алгоритмов инерциальной навигации роботов на микроконтроллере семейства AVR.

Цели данной работы:

1. Разработать алгоритм инерциальной навигации робота в незнакомом помещении.
2. Создать действующий прототип, демонстрирующий алгоритмы.

Для достижения поставленной цели, необходимо решить следующие задачи:

1. Исследовать применения систем инерциальной навигации роботов для решения определенных задач и возможности автономного управления роботом с получением данных о его местоположении, применяемые на практике.
2. Изучить методы, применяемые в системах инерциальной навигации.
3. Выделить элементную базу, необходимую для реализации проекта; изучить библиотеки С/С++, необходимые для реализации программной составляющей роботов.
4. Создать действующий прототип, на котором будет осуществляться отработка алгоритмов.

# Отработка методов инерциальных систем навигации

Принцип инерциальной системы навигации может быть использован в качестве робота-разведчика с целью решения практической проблемы составления плана или карты местности в условиях полной изоляции от внешних сигналов.

Среди уже существующих роботов-разведчиков есть несколько роботов, такие как Eyedrive, Eyeball или SIM-ROBTEC для обследования помещений с системой видеонаблюдения. Недостатки этих роботов:

1. Возможности оценивать перемещения робота только через видео и аудио, т.е. мы не можем отследить его местонахождение если он уехал достаточно далеко (не всегда может быть понятно где именно находиться робот).
2. Невозможности составить конкретный план здания, в котором робот перемещался.
3. Ограниченный радиус передачи информации (всего 50 метров в здании).
4. Так как эти роботы являются частью вооружения армий разных стран, то они становятся недоступны для простых людей.

Придуманный мной робот из бюджетных компонентов может записывать свои перемещения, координаты встреченных на пути препятствий и после выполнения программы выдавать данные о них (радиус действия робота не ограничен т.к. он полностью автономен и передает данные о своих перемещениях лишь тогда, когда закончиться выполнение программы). Эти данные микроконтроллер будет получать от инкрементных энкодеров и гироскопа. На их основании можно будет говорить о маршрутах движения в данной локации, схемах здания, т.е. одометрии. Для этого был придуман следующий план действий для робота:

1. Робот просто ездит и корректирует свой курс в соответствии с наличием препятствий на пути.
2. После того как робот проедет достаточное расстояние, он должен будет приехать в ту точку, из которой начал движение.
3. Ему необходимо будет выдать данные о собственных перемещениях.

В качестве устройства управления была выбрана платформа Arduino Uno, основанный на микроконтороллере **ATmega328P**.

Другие возможные использования этой идеи при продолжении работы с данным алгоритмом – возвращение транспорта на стоянки, базы или проверка того насколько водитель транспорта придерживался маршрута.

# Описание применяемых датчиков

Необходимо определиться, что мы установим на опытный образец, из чего он будет состоять, и как мы это применим. Обязательным условием являлось то, чтобы компоненты можно было купить в любом магазине электроники для возможности дальнейшего распространения этих роботов как в качестве игрушки, так и в качестве реального устройства, способного отслеживать свои перемещения, координаты встреченных на пути препятствий:

1. Основа:
   1. Акриловая пластина, моторы, колеса, батарейный отсек.
   2. В качестве обработчика данных микроконтроллер семейства AVR, а именно Arduino Uno с расширяющим шилдом.
   3. Для управления моторами драйвер **L298N**. Микросхема **L298N** — это сборка из четырёх H-полумостов. Это может быть использовано для независимого управления двумя двигателями постоянного тока, с возможностью включения реверса. Также чип может быть использован для контроля одного биполярного шагового двигателя.[[2]](#footnote-2)
2. Изменение траектории в связи с препятствиями:
   1. 4 инфракрасных датчика препятствий YL-63 (2 вперед и 2 по бокам). Датчик предназначен для применения, когда не требуется информация о расстоянии до объекта, а только о его наличии или отсутствии. Датчик имеет подстроечный элемент для регулирования предельной дистанции регистрации препятствия

Это оптический датчик регистрирующий интенсивность отраженного инфракрасного излучения в контролируемом пространстве. Изменение отраженного излучения происходит из-за перемещения окружающих предметов или самого датчика.

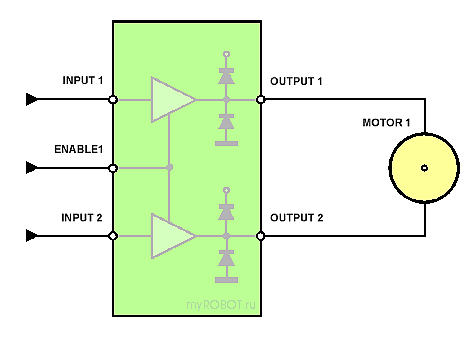
1. Инерциальная навигация:
   1. Герконы KY-025 в роли 2 инкрементных магнитных энкодеров, по 1 на каждое колесо, для измерения пройденного расстояния. Коммутационные устройства или просто контакты очень широко применяются в различной электрической и радиотехнической аппаратуре. С целью улучшения эксплуатационных свойств, прежде всего срока службы и надежности соединения и были разработаны **магнитоуправляемые герметизированные контакты** получившие название **герконы**. Энкодер - это устройство для преобразования угловых или линейных перемещений в аналоговый или цифровой сигнал. Часто энкодер - называют датчиком угловых положений или датчиком обратной связи. Инкрементальные энкодеры - это устройства подсчета количества импульсов от точки отсчета на единицу оборота вращения вала. Они генерируют последовательный импульсный цифровой код, содержащий информацию относительно угла поворота вала. То есть основная задача инкрементального энкодера - считать единичные импульсы за цикл, равный одному обороту диска. Импульсы в инкрементальном энкодере формируются при помощи вращения диска с метками. [[3]](#footnote-3)
   2. Акселерометр + гироскоп (датчик GY-521 на основе MPU-5060). Позволяет определить положение и перемещение прибора в пространстве: углы крена, тангажа и рысканья (см. Рис.8) ориентируясь по вектору силы тяжести и скорости вращения. Так же измеряет температуру. Определяет линейное ускорение и угловую скорость по трем осям. Основа модуля – микросхема [MPU-6050](http://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/). Содержит два устройства акселерометр и гироскоп. Их данные проходят предварительную обработку и передаются по последовательному интерфейсу I2C в микроконтроллер. Модуль 3-х осевого гироскопа и акселерометра GY-521 MPU-6050 является электронным аналогом нашего вестибулярного аппарата.
2. Выдача данных о перемещениях:
   1. Lcd1602 на базе контроллера [HD44780](https://ru.wikipedia.org/wiki/HD44780) с символьным типом отображения с возможностью загрузки символов. Экран имеет 16 контактов для питания логики, взаимодействия с управляющей электроникой и подсветки.

Можно было бы использовать модуль FC-113 на базе микросхемы PCF8574T, которая представляет собой 8-битный сдвиговый регистр – «расширитель» входов-выходов для последовательной шины I2C, но выводы SDA и SCL уже заняты датчиком GY-521.

# Описание особенностей использования применяемых датчиков

1. **L298N** содержит сразу два драйвера (четыре независимых канала, объединенных в две пары) для управления электродвигателями небольшой мощности. Имеет две пары выходов для подключения электромоторов и две пары входов для управляющих сигналов. Кроме того, у **L298N** есть два входа для включения каждого из драйверов, которые используются для управления скоростью вращения электромоторов с помощью широтно-модулированного сигнала (ШИМ).  
   **L298N** обеспечивает разделение электропитания для микросхемы и для управляемых ею двигателей, что необходимо для уменьшения помех, вызванных бросками напряжения, связанными с работой моторов. Разделение электропитания микросхем и электродвигателей может быть также использовано для подключения электродвигателей с большим напряжением питания, чем у микросхемы.  
   Принцип работы каждого из драйверов, входящих в состав микросхемы, идентичен, поэтому рассмотрим принцип работы одного из них.

**Рассмотрим принцип работы лишь одного из них, так как принцип работы каждого из драйверов, входящих в состав микросхемы идентичен.**



*Рис.1.* Устройство драйвера для управления электродвигателей.

К выходам OUTPUT1 и OUTPUT2 подключим электромотор MOTOR1.   
На вход ENABLE1, включающий драйвер, подадим сигнал, т.е. соединим его с положительным полюсом источника питания +5V. Если при этом на входы INPUT1 и INPUT2 не подаются сигналы, то мотор вращаться не будет.  
Если вход INPUT2 соединить с отрицательным полюсом источника питания, а вход INPUT1 - с положительным, то мотор начнет вращаться.  
Теперь попробуем соединить вход INPUT1 с отрицательным полюсом источника питания, а вход INPUT2 - с положительным. Мотор начнет вращаться в другую сторону.  
Попробуем подать сигналы одного уровня (соединить оба входа или с положительным или с отрицательным полюсом источника питания) сразу на оба управляющих входа INPUT1 и INPUT2 - мотор вращаться не будет.  
Если мы уберем сигнал с входа ENABLE1, то даже при наличии сигналов на входах INPUT1 и INPUT2 мотор не будет вращаться.



*Рис.2.* Внешний вид драйвера электродвигателей.

EТA и ENB подключаем в один digital порт, in1, in2, in3, in4 в другие 4 порта.

1. Инфракрасные датчики. Устройство содержит источник ИК излучения и фотоприемник. Излучение отражается от препятствия и регистрируется фотоприемником. Он передает сигнал на компаратор LM393, который настроен на срабатывание при определенном уровне освещенности фотоприемника. Компаратор формирует сигнал на выходе датчика YL-63 низкого или высокого логического уровня.



*Рис.3.* Внешний вид инфракрасного датчика.

*Рис.4.* Принцип работы датчика и отражение названия класса диффузных датчиков.

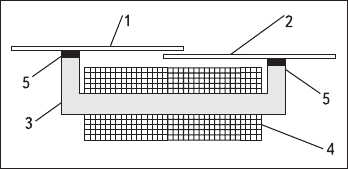
|  |
| --- |
|  |

Оптический датчик YL-63 относится к классу диффузионных. Название группы датчиков возникло из-за лежащего в основе работы датчика отражения излучения по множествам направлений – диффузии излучения отражающей поверхностью.  
Работа устройства заключается в определении освещенности фотоприемника. Поскольку YL-63 фиксирует отраженное излучение, то возникает погрешность измерения расстояния, вызванная различной отражающей способностью поверхностей объектов изготовленных из разнообразных материалов.[[4]](#footnote-4)

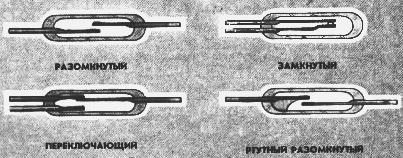
1. Магнитные инкрементные энкодеры. Магнитный энкодер - датчик, фиксирующий прохождение вращающегося магнита, расположенного поблизости от чувствительного элемента. Данные магнитного энкодера на выходе имеют вид цифрового кода. Было решено использовать геркон вместо датчика Холла из-за его надежности и долговечности.

Для того чтобы вызвать срабатывание контактной группы, необходимо создать магнитное поле достаточной напряженности вокруг геркона. При этом не важно, как это поле будет создано. Это может быть или постоянный магнит, или электромагнит. Внешнее магнитное поле намагничивает внутренние контакты – сердечники геркона, в результате чего они притягиваются и замыкают электрическую цепь. Контакты будут находиться в таком состоянии до тех пор, пока вокруг них есть магнитное поле достаточной напряженности. Достаточно выключить или увеличить расстояние между магнитом и герконом, для того чтобы контакты сразу разомкнутся. Следующее срабатывание контактов произойдет, когда магнитное поле появится вновь.

Устроен такой контакт следующим образом. К сердечнику 3 (см. Рис.5) из магнитомягкого материала (обладающего свойствами ферромагнетика или ферримагнетика) через изолирующие прокладки 5 прикреплены контакты 1 и 2, выполненные также из магнитомягкого материала. При пропускании тока через катушку 4 в сердечнике 3 возникает магнитное поле и намагничивает контакты 1 и 2, которые замыкаются. Размыкание контактов происходит при прекращении тока через катушку.[[5]](#footnote-5)

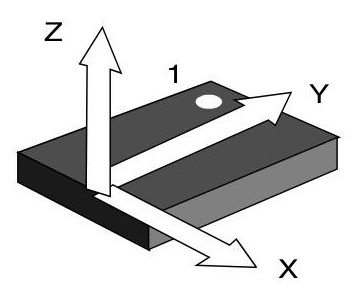
1. [](http://electrik.info/main/school/)

*Рис.5.* Магнитоуправляемый контакт профессора В. Коваленкова

[](http://electrik.info/main/school/)

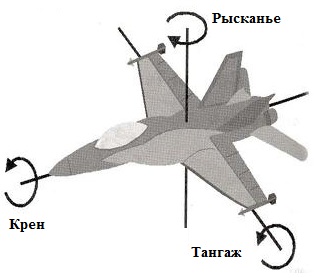
*Рис.6.* Разновидности конструкций герконов

1. Акселерометр. Трехосевой акселерометр измеряет проекции ускорения на оси X, Y и Z. Если прибор размещен строго горизонтально и не движется то проекции ускорения силы тяжести на оси X и Y равны нулю. Сила тяготения воспринимается только чувствительными элементами вертикальной оси Z. Время от времени в состоянии покоя производят проверку и калибровку акселерометра. Во время движения объект постоянно то ускоряется, то замедляется. Идеально равномерного движения не существует. Это и позволяет использовать акселерометр не только для определения положения объекта, но и для определения динамических параметров при движении. Акселерометр регистрирует сумму ускорения при движении и гравитацию. Если от всех элементов акселерометра по осям X, Y, Z поступают показания близкие к нулю находится в свободном падении.



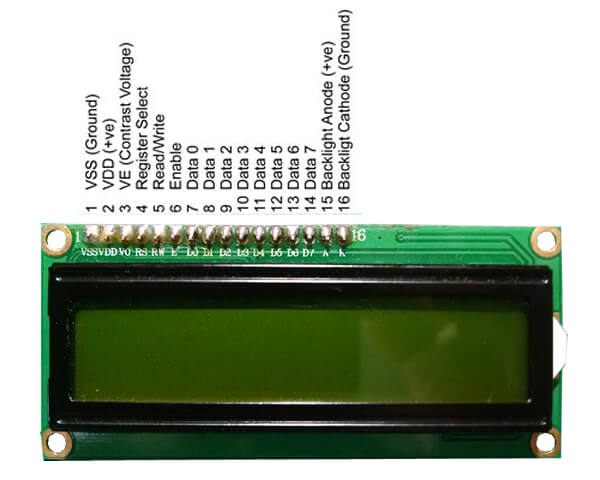
*Рис.7.* Устройство коллекторного двигателя постоянного тока.

1. Гироскоп. Трехосевой гироскоп – датчик поворота объекта, позволяющий вычислить углы поворотов по осям X, Y, Z благодаря определению угловых скоростей. При этом углы отклонения (углы Эйлера) от нормального положения по этим осям носят специальные названия:
2. Крен - поворот аппарата вокруг продольной оси;
3. Тангаж - поворот аппарата относительно его поперечной оси;
4. Рысканье - поворот аппарата относительно его вертикальной оси;



*Рис.8.* Углы отклонения при работе с 3-х осевым гироскопом.

1. Жидкокристаллический дисплей LCD1602.



*Рис.9.* Обозначение выводов на дисплее.

| Вывод | Обозначение | Описание |
| --- | --- | --- |
| 1 | GND | Общий вывод (земля) |
| 2 | Vcc | Напряжение питания (3,3—5 В) |
| 3 | Vo | Управление контрастностью |
| 4 | RS | Выбор регистра |
| 5 | R/W | Выбор режима записи или чтения |
| 6 | E | Разрешение обращений к индикатору (а также строб данных) |
| 7 | DB0 | Шина данных (8-ми битный режим)(младший бит в 8-ми битном режиме) |
| 8 | DB1 | Шина данных (8-ми битный режим) |
| 9 | DB2 | Шина данных (8-ми битный режим) |
| 10 | DB3 | Шина данных (8-ми и 4-х битные режимы)(младший бит в 4-х битном режиме) |
| 11 | DB4 | Шина данных (8-ми и 4-х битные режимы) |
| 12 | DB5 | Шина данных (8-ми и 4-х битные режимы) |
| 13 | DB6 | Шина данных (8-ми и 4-х битные режимы) |
| 14 | DB7 | Шина данных (8-ми и 4-х битные режимы) |
| 15 | +LED | + питания подсветки |
| 16 | –LED | – питания подсветки |

Дисплей может работать в двух режимах:

* 8-битный режим — в нём используются и младшие и старшие биты (DB0-DB7)
* 4-битный режим — в нём используются только младшие биты (DB4-DB7)

Использовать восьмибитный режим не целесообразно. Для его работы требуется на 4 дополнительных вывода больше, а выигрыша по скорости практически нет.

# Программное обеспечение, реализующее алгоритм

1. **L298N**. Для него создаем 5 функций управления моторами (вперед-назад, влево-вправо, стоп). Так же настраиваем ШИМ на моторы.
2. Инфракрасные датчики. Просто считываем сигнал и относительно показаний корректируем путь. Видим препятствие, отъезжаем назад, поворачиваем, едем дальше.
3. Энкодеры. Для них настраиваем внешние прерывания и настраиваем их так, чтобы при движении вперед счетчик увеличивал показания, при движении назад – уменьшал. Для этого используем функции для моторов.
4. Экран lcd1602. Для работы дисплея используется встроенная в среду Arduino IDE библиотека LiquidCrystal.h. Будем использовать 4-битный режим.
5. Акселерометр.  
   Основная проблема, связанная с использование акселерометра состоит в том, что данный прибор имеет довольно высокий уровень шума. Этот шум является следствием не только внешних воздействий (вибрации двигателей, например), но и внутренних причин, связанных с конструкцией датчика. Таким образом, чтобы использовать показания акселерометра их нужно отфильтровать. Шум акселерометра является высокочастотным, а по амплитуде (в некоторых случаях) сопоставим с полезным сигналом. На практике, чтобы избавится от шума, применяют несколько разных подходов, в зависимости от мощности вычислительного устройства.

В простейшем случае можно применить фильтр верхних частот (сглаживающий фильтр). Механизм его действия очень прост: усреднение нескольких значений исходной функции. Значение выходного сигнала определяется формулой (1).

Также может применяться фильтр Гаусса. Это простейший рекурсивный фильтр, т.е. его показания определяются не только текущим состоянием системы, но и предыдущим (предыдущими). Значение выходного сигнала определяется формулой (2).

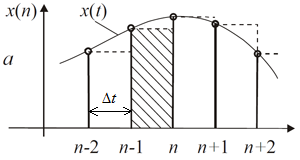
*Рис.10.* Модели различных цифровых фильтров низких частот.

1. Гироскоп.

Гироскоп - электронный прибор, измеряющий угловую скорость по одной из осей (в случае 3-х осевого гироскопа - по всем 3-м осям). Современные гироскопы совершенно нечувствительны к линейным ускорениям, что делает их показания очень точными. Следовательно, информация, поступающая с них, не нуждается в фильтрации. Однако большинство гироскопов имеют один недостаток дрейф нуля в зависимости от температуры датчика. Для коррекции возникающей неточности современные модели гироскопов оснащаются встроенным термометром. Но возникает другая проблема: для расчёта углов крена, тангажа и рысканья требуется интегрировать показания гироскопа, так как он показывает значение угловой скорости, а не угла поворота. Интегрирование производится численными методами, при этом, происходит накопление ошибки. В простейшем случае (метод прямоугольников) формула будет иметь вид (3).

где gyro - показания гироскопа, ∆*t* - интервал между измерениями;

Суть метода прямоугольников наглядно продемонстрирована на рисунке 6. Геометрический смысл определённого интеграла: площадь некоторой криволинейной трапеции. На рисунке кривая *x(t) -* изменение угла поворота с течением времени, *n-*номера отсчётов, показания, получаемые с гироскопа. Можно повысить точность интегрирования, применяя другие способы вычисления. В частности, хорошее увеличение точности даёт использование метода трапеций. Существуют и многие другие методы численного вычисления интегралов, однако все они являются ресурсоёмкими, а при использовании микроконтроллера, который должен выполнять ещё ряд других задач, их реализация становится нецелесообразной.



*Рис.11.* Геометрический смысл определённого интеграла.

# Заключение

Создан робот, полностью отвечающий нашим целям и задачам. Мы можем отследить маршрут его движения, вернуть его в точку начала пути. Поставленные цели достигнуты:

1. Разработан алгоритм инерциальной навигации робота в незнакомом помещении.
2. Создан действующий прототип, демонстрирующий алгоритмы.

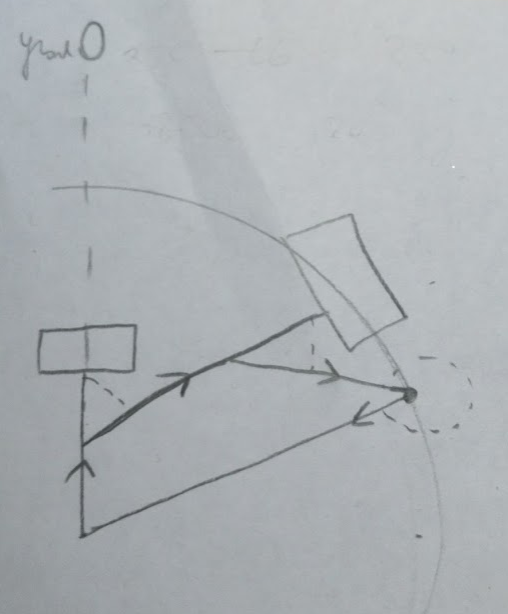
По пути достижения целей были решены следующие задачи:

1. Были исследованы применения систем инерциальной навигации роботов для решения определенных задач и возможности автономного управления роботом с получением данных о его местоположении, применяемые на практике.
2. Изучены методы, применяемые в системах инерциальной навигации.
3. Выделена элементная база, необходимая для реализации проекта; изучены библиотеки С/С++, необходимые для реализации программной составляющей роботов.
4. Создан прототип, на котором будет осуществляться отработка алгоритмов.

Сам метод, представленный мной, является довольно перспективным для различных видов транспорта, постоянно находящихся в условии ограниченной видимости или не имеющих доступа к системам навигации, связанными с поступающими извне сигналами. При более углубленном изучении данной темы мной в будущем будет создан более точный и более надежный образец с защитой от внешних механических повреждений, усовершенствованным программным кодом и алгоритмами отслеживания перемещения робота. В дальнейшем я планирую сделать возможность просмотра карты перемещений робота.

В демонстрации проекта показаны основные принципы придуманного алгоритма робота-разведчика, который способен вывести информацию о препятствиях, вернуться на стартовую точку с последующим выводом на экран данных о пространстве в данном помещении.

Робота-разведчика в работе можно посмотреть по ссылке на YouTube: <https://drive.google.com/open?id=1S-U8CcaTik5JedBxIHO3Iqb09o_x9OmL>

- схема представленная в видео.

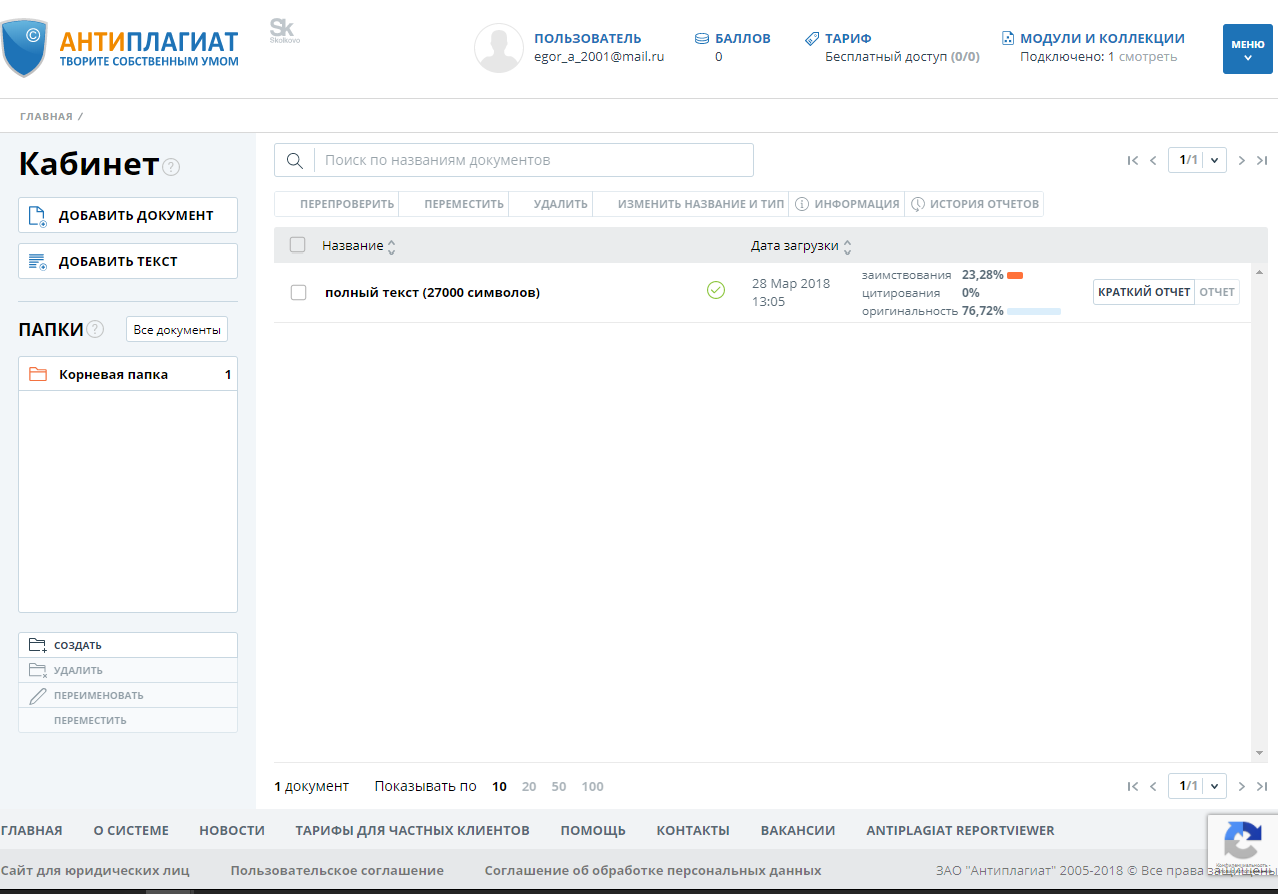
Код можно посмотреть по ссылке: <https://drive.google.com/open?id=1op5HO_Q5sqPnh2trO6btx5ZAGG7vvdXU>

Ссылка на папку с кодом, видео и фотографиями: <https://drive.google.com/open?id=1yhhyLssTwziLgL0apVynYylHOpTbCjZX>

Библиографический список

1. Инерциальная навигация. https://ru.wikipedia.org
2. Техническая документация на L293D, датчики <https://arduino-kit.ru>
3. Техническая документация на датчики <http://www.servotechnica.ru>
4. Техническая документация на Arduino Uno. http://arduino.ru /
5. Устройство датчиков http://electrik.info
6. Техническая документация на микроконтроллер ATmega328. <http://www.atmel.com/>
7. Сайт паяльник. http://cxem.net/

# Приложение 1



# Приложение 2

#include <util/delay.h> // подключаем библиотеку задержек

#include <LiquidCrystal.h> // библиотека для дисплея

LiquidCrystal lcd(13, 12, 9, 8, 3, 2); // (RS, E, DB4, DB5, DB6, DB7)

#define F\_CP 16000000 // тактовая частота процессора

#define F\_PWM\_for\_motors 1000 // частота ШИМ для моторов (1000 Гц)

#include <avr/interrupt.h> // библиотека для работы с прерывниями

#include "timer-api.h"

#define ger 2

#define ger2 3

#include<Wire.h> //гироскоп и аксель

const int MPU\_addr=0x68; // I2C address of the MPU-6050

int16\_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZFIRST,GyZSECOND;

const byte averageFactor=10; // переменная для сглаживания

int GyZC = 0;//показания с сенсора

int OneWhell=2\*3.14; // треть колеса(0,666\*пr)

int DistanceX0=0; int DistanceY0=0; // проекции перемещений

int DistanceX=0; int DistanceY=0; // проекции перемещений

int DistanceXNEC=0; int DistanceYNEC=0; // проекции перемещений (сохращение значений)

int GyzNEC=0;// угол для постановки на необходимый радиус-вектор

//константы для датчика

const float vref = 3.3;

const float vzero = 1.23;

const float sens = 0.0005;

const float adc = 1023;

const int INTEGR\_DELAY = 20;

const int SERIAL\_DELAY = 100;

const int gyrPin = A4;

int time;

int integr\_time, real\_delta;

short gyr\_raw;

float angle, aspeed;

int integr\_timer, real\_delta\_timer;

short gyr\_raw\_timer;

float angle\_timer, aspeed\_timer;

int LCD\_COUNT\_ANDLE [50]; // массивы для записи пройденных расстояний и углов

int LCD\_COUNT\_DIS [50];

int COUNT\_DIS\_ANDLE =0; // переменная для счета препятствий (количества отрезков на каждом из которых измеряется угол и расстояние)

int OUTPUT\_ANDLE\_DIS=0; // переменная для вывода отрезков по одному

volatile int n1=0; // счётчик импульсов для левого геркона

volatile int n2=0; // счётчик импульсов для правого геркона

unsigned char volatile FLAG\_of\_direct\_1 = 0; // переменная-флаг для указания текущего направления вращения мотора (левого), 0-вперёд, 1-назад

unsigned char volatile FLAG\_of\_direct\_2 = 0; // переменная-флаг для указания текущего направления вращения мотора (правого), 0-вперёд, 1-назад

int enA = 11;

int enB = 10;

int LeftForvard = 7; // левый мотор назад

int LeftBack =6; // левый мотор вперёд

int RightBack =5; // правый мотор назад

int RightForvard =4; // правый мотор вперёд

ISR(INT1\_vect) {

if(FLAG\_of\_direct\_1 = 0){

n1=n1+1;

}

else {

n1=n1-1;

}

}

ISR(INT0\_vect) {

if(FLAG\_of\_direct\_2 = 0){

n2=n2+1;

}

else {

n2=n2-1;

}

}

void setup() {

pinMode(enA, OUTPUT);

pinMode(enB, OUTPUT);

pinMode(LeftBack, OUTPUT); // инициализация пинов для упраления драйвером моторов

pinMode(LeftForvard, OUTPUT);

pinMode(RightBack, OUTPUT);

pinMode(RightForvard, OUTPUT);

// инициализация внешних прерываний

EICRA = (1<<ISC11)|(0<<ISC10)|(1<<ISC01)|(0<<ISC00); // прерывания по заднему фронту на обоих входах

EIMSK = (1<<INT1)|(1<<INT0); // разрешаем оба внешних прерывания

timer\_init\_ISR\_1Hz(TIMER\_DEFAULT);// частота=1Гц Внутренние прерывания по таймеру

Wire.begin(); // гироскоп и аксель

Wire.beginTransmission(MPU\_addr);

Wire.write(0x6B); // PWR\_MGMT\_1 register

Wire.write(0); // set to zero (wakes up the MPU-6050)

Wire.endTransmission(true);

Serial.begin (9600); // подключаем монитор порта

pinMode(A0,INPUT); //FULeft //инфракрасные датчики

digitalWrite (A0,HIGH);

pinMode(A2,INPUT);//FULCentre

digitalWrite (A2,HIGH);

pinMode(A1,INPUT);//FURCentre

digitalWrite (A1,HIGH);

pinMode(A3,INPUT);//FURight

digitalWrite (A3,HIGH);

angle = 0;

sei(); // разрешаем внешние прерывния

lcd.begin(16, 2);

}

void timer\_handle\_interrupts(int timer) { //внутренние прерывания по таймеру для счета проекций

static unsigned long prev\_time = 0;

unsigned long \_time = micros();

unsigned long \_period = \_time - prev\_time;

prev\_time = \_time;

Wire.beginTransmission(MPU\_addr); // блок измерения угла и проекций

Wire.write(0x3B); // starting with register 0x3B (ACCEL\_XOUT\_H)

Wire.endTransmission(false);

Wire.requestFrom(MPU\_addr,14,true); // request a total of 14 registers

GyZFIRST=Wire.read()<<8|Wire.read(); // 0x47 (GYRO\_ZOUT\_H) & 0x48 (GYRO\_ZOUT\_L)

time = millis();

real\_delta\_timer = time - integr\_timer;

if( real\_delta > INTEGR\_DELAY ){

integr\_timer = time;

gyr\_raw\_timer = GyZFIRST;

aspeed\_timer = ((gyr\_raw\_timer \* vref)/adc - vzero)/sens;

angle\_timer = angle\_timer + aspeed\_timer \* real\_delta\_timer;

}

DistanceY=DistanceY + OneWhell/(sin (angle\*3.14/180));

DistanceX=DistanceX + OneWhell/(cos (angle\*3.14/180));

Serial.print("timer: ");

Serial.println(\_period, DEC);

}

void motors\_forward() { // процедура запуска мотов вперёд

FLAG\_of\_direct\_1 = 0; // левый мотор вращается вперёд

FLAG\_of\_direct\_2 = 0; // правый мотор вращается вперёд

digitalWrite(LeftForvard, HIGH);

digitalWrite(RightBack, LOW);

digitalWrite(LeftBack, LOW);

digitalWrite(RightForvard, HIGH);

}

void motors\_back() { // процедура запуска моторов назад

FLAG\_of\_direct\_1 = 1; // левый мотор вращается назад

FLAG\_of\_direct\_2 = 1; // правый мотор вращается назад

digitalWrite(RightForvard, LOW);

digitalWrite(RightBack, HIGH);

digitalWrite(LeftBack, HIGH);

digitalWrite(LeftForvard, LOW);

}

void left\_rotate() { // процедура повора налево

FLAG\_of\_direct\_1 = 0; // левый мотор вращается вперёд

FLAG\_of\_direct\_2 = 1; // правый мотор вращается назад

digitalWrite(RightForvard, HIGH);

digitalWrite(LeftForvard, LOW);

digitalWrite(LeftBack, HIGH);

digitalWrite(RightBack, LOW);

}

void right\_rotate() { // процедура поворота направо правый геркон- второй

FLAG\_of\_direct\_1 = 1; // левый мотор вращается назад

FLAG\_of\_direct\_2 = 0; // правый мотор вращается вперёд

digitalWrite(RightBack, HIGH);

digitalWrite(RightForvard, LOW);

digitalWrite(LeftForvard, HIGH);

digitalWrite(LeftBack, LOW);

}

void stop\_mashina() { // процедура поворота направо

digitalWrite(RightBack, LOW);

digitalWrite(RightForvard, LOW);

digitalWrite(LeftForvard, LOW);

digitalWrite(LeftBack, LOW);

}

void prepiatstvia(){ //блок осмотра препятствий

if (digitalRead(A0)==0){ //0- препятствие. 1- пусто

Serial.println("FULeft");

motors\_back();

delay(1000);

right\_rotate();

delay(100);

motors\_forward();

LCD\_COUNT\_DIS[COUNT\_DIS\_ANDLE]=OneWhell\*n1 ;// вводим в массив дистанцию до препятствия и угол под которым теперь движется робот

COUNT\_DIS\_ANDLE=COUNT\_DIS\_ANDLE+1;

giro ();

LCD\_COUNT\_ANDLE [COUNT\_DIS\_ANDLE]=angle;

}

if (digitalRead(A3)==0){ //0- препятствие. 1- пусто

Serial.println("FURight");

motors\_back();

delay(1000);

left\_rotate();

delay(100);

motors\_forward();

LCD\_COUNT\_DIS[COUNT\_DIS\_ANDLE]=OneWhell\*n1 ;

COUNT\_DIS\_ANDLE=COUNT\_DIS\_ANDLE+1;

giro ();

LCD\_COUNT\_ANDLE [COUNT\_DIS\_ANDLE]=angle;

}

if ((digitalRead(A2)==0)||(digitalRead(A1)==0)){ //0- препятствие. 1- пусто

Serial.println("FUCenter");

motors\_back();

delay(1000);

right\_rotate();

delay(100);

motors\_forward();

LCD\_COUNT\_DIS[COUNT\_DIS\_ANDLE]=OneWhell\*n1 ;

COUNT\_DIS\_ANDLE=COUNT\_DIS\_ANDLE+1;

giro ();

LCD\_COUNT\_ANDLE [COUNT\_DIS\_ANDLE]=angle;

}

}

void giro (){ // показания гироскопа (считывание и интегрирование)

Wire.beginTransmission(MPU\_addr);

Wire.write(0x3B); // starting with register 0x3B (ACCEL\_XOUT\_H)

Wire.endTransmission(false);

Wire.requestFrom(MPU\_addr,14,true); // request a total of 14 registers

GyZC=Wire.read()<<8|Wire.read();

time = millis();

real\_delta = time - integr\_time;

if( real\_delta > INTEGR\_DELAY ){ // интегрирование

integr\_time = time;

gyr\_raw = GyZC;

aspeed = ((gyr\_raw \* vref)/adc - vzero)/sens;

angle = angle + aspeed \* real\_delta;

}

}

void loop() {

analogWrite(enA, 160);// левый двигатель (устанавливаем скорость 160 из доступного диапазона 0~255)

analogWrite(enB, 170);//правый (чтобы ехало ровно)

motors\_forward(); // вращаем моторы вперёд

if((DistanceY+DistanceX<300)){ // проехал 3 метра по радиусу от начальной точки

Serial.println(n1);

prepiatstvia();

}

else {

Serial.println(n1);

DistanceYNEC=DistanceY;// переменные для сохренения значений x и y

DistanceXNEC=DistanceX;

GyzNEC= tan(DistanceYNEC/DistanceXNEC)\*180/3.14;//нужный градус в градусах

giro ();

while((180+GyzNEC) != (angle)){ //блок разворота и движения по радиус вектору

right\_rotate();

delay (50);

giro ();

}

DistanceY=0;

DistanceX=0;

motors\_forward();

while ((DistanceY=DistanceYNEC)&&(DistanceX=DistanceXNEC)){ // пока не приедешь в исходную точку (по X и Y)

delay(500);

Serial.println(n1);

prepiatstvia();

giro ();

GyzNEC= tan((DistanceYNEC-DistanceY)/(DistanceXNEC-DistanceX))\*180/3.14;//нужный градус в градусах

while((180+GyzNEC) != (angle)){ //блок разворота и движения по радиус вектору

right\_rotate();

delay (50);

giro ();

}

motors\_forward();

prepiatstvia();

}

}

stop\_mashina(); //финальный блок выдачи значений

delay(3000);

for (OUTPUT\_ANDLE\_DIS=0;OUTPUT\_ANDLE\_DIS<COUNT\_DIS\_ANDLE;OUTPUT\_ANDLE\_DIS++){

lcd.setCursor(0, 0);

lcd.print("distanse:");

lcd.setCursor(9, 0);

lcd.print(LCD\_COUNT\_DIS[OUTPUT\_ANDLE\_DIS]);

lcd.setCursor(0, 1);

lcd.print("andle:");

lcd.setCursor(6, 1);

lcd.print(LCD\_COUNT\_ANDLE[OUTPUT\_ANDLE\_DIS]);

delay (5000);

lcd.clear();

}

delay(999999); // время для того чтобы оператор выключил робота для лишь единичного выполнения программы

} // или для повторного запуска

1. https://ru.wikipedia.org/wiki/Инерциальная\_навигация [↑](#footnote-ref-1)
2. http://arduinoprom.ru/obzory-modulej/271-obzor-chipa-l293d-dvuhtaktnyj-chetyrehkanalnyj-drajver-dvigatelej-postojannogo-toka.html [↑](#footnote-ref-2)
3. http://www.servotechnica.ru/catalog/type/index.pl?id=118 [↑](#footnote-ref-3)
4. https://arduino-kit.ru/catalog/id/ik-datchik-prepyatstviy-dlya-robotov-mashin-yl-63-\_fc-51\_ [↑](#footnote-ref-4)
5. http://electrik.info/main/fakty/417-chto-takoe-gerkony.htm [↑](#footnote-ref-5)