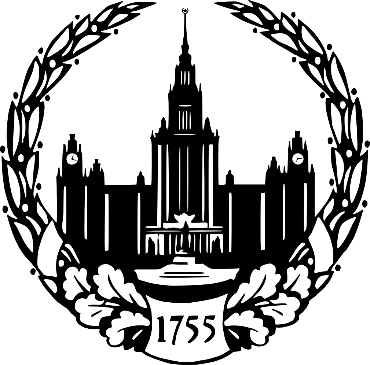
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

имени М.В.ЛОМОНОСОВА

Механико-математический факультет

Кафедра общих проблем управления



ДИПЛОМНАЯ РАБОТА

на тему:

**«Построение оптимального синтеза в одной линейно-квадратичной задаче оптимального управления и его реализация в виде мобильного робота на платформе Arduino»**

**(«Structure of optimal synthesis in the one linear-quadratic optimal control problem and its implementation in the form of a mobile robot on the Arduino platform»)**

Студента 632 группы

Пивачева Александра Павловича

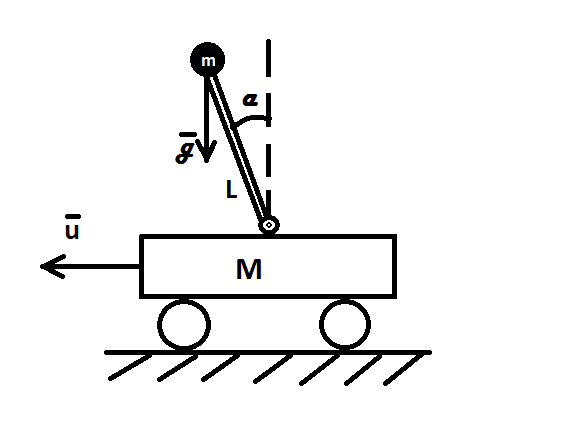
Научный руководитель:

Локуциевский Лев Вячеславович

2019 год

**СОДЕРЖАНИЕ**

1. Введение.
2. Постановка задачи.
3. Решение
4. Измерение параметров.
5. Реализация программного обеспечения.
6. Заключение.
7. Используемая литература.

****

**Рис. 1**

**1 Введение:**

В настоящее время алгоритмы стабилизации различных механических систем находят всё более широкое применение. Стабилизация может применяться для облегчения и повышения качества работы с механическими устройствами. Создание управляемого робота на основе мобильной платформы dfrobot Pirat даёт возможность реализовать на практике оптимальные управления, полученные в ходе решения различных задач стабилизации. Поиск управлений, их последующая реализация и тестирование позволяет совершенствовать алгоритмы стабилизации, повышать их эффективность, оценивать применимость управлений в практических целях. Данный метод также позволяет накапливать опыт в технических вопросах, возникающих в процессе реализации. В нашем случае рассматривается одна из задач стабилизации перевёрнутого маятника. Данные задачи могут быть связаны со стабилизацией ракет, поскольку их двигатели находятся ниже центра тяжести, а также со стабилизацией наземного транспорта, имеющих одну ось (моноколесо, сегвей и т.д.).

**2 Постановка задачи:**

Необходимо найти управление, позволяющее стабилизировать перевёрнутый маятник, установленный на тележке. Введём следующие обозначения:

M – масса тележки

m – масса маятника

L – расстояние от точки крепления до центра масс маятника

u – управление тележки (ускорение)

– угол отклонения маятника от вертикали

Положим за – скорость центра тяжести маятника, – угловое ускорение маятника, тогда:

- задача стабилизации (2.1)

**3 Решение:**

Запишем Лагранжиан:

– кинетическая энергия

– скорость тележки

– скорость центра масс маятника относительно тележки

– потенциальная энергия

Запишем уравнения Эйлера-Лагранжа для и

Линеаризуем в точке окрестности получим

Сделаем замену и получим следующую задачу оптимизации:

Воспользуемся Принципом Максимума Понтрягина:

, что функционал где , достигает своего максимума на оптимальной траектории управления из множества всех управлений

Распишем скалярное произведение:

Запишем Гамильтонову систему:

\*

Поиск максимума гамильтониана по эквивалентен поиску максимума той части, которая зависит от управления :

Обозначим её за

Рассмотрим случаи, когда

:

2. – любое

Случаи 1. и 2. Нас не интересуют, а случай 3. Достигается при

из последней тройки уравнений системы \* находим: , ,

последнее противоречит условию принципа максимума

Найдём :

Возьмём за и получим

(3.1)

Обозначим за и , подставим в управление (3.1) и получим систему дифференциальных уравнений:

решение которой выглядит так:

После решения искомое управление выглядит так:

(3.2)

Чтобы найти константы подставим v,, в условия стабилизации (1) и получим:

Подставим константы в (3.2):

\*

**4 Измерение параметров:**

Для реализации полученного управления на мобильной платформе необходимо получить численные значения параметров, использующихся в нашей задаче, путём измерений. Это параметры двух типов:

Мгновенные (изменяются с течением времени):

скорость тележки

ускорение тележки

– угол отклонения маятника от вертикали

угловая скорость движения маятника

Постоянные:

M – масса тележки

m – масса маятника

L – приведённая длина маятника

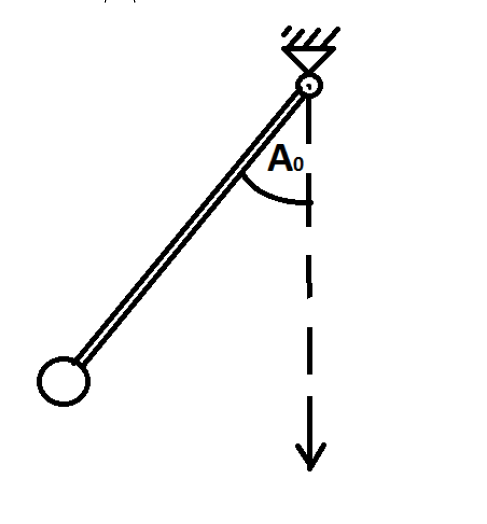
В постоянные параметры так же следует добавить некоторые характеристики двигателя тележки, позволяющие связать управление этим двигателем и изменение ускорения тележки.

Мгновенные параметры определяются по показаниям датчиков. Далее речь пойдёт об определении постоянных параметров.

Масса тележки и масса маятника:

Для данной задачи применялись электронные весы с точностью 0.5 грамм. Результаты взвешивания:

Приведённая длина маятника:



**Рис. 2**

В нашей задаче ранее использовалась модель с математическим маятником. Но в природе существует только физический маятник. Следовательно, нам необходима приведённая длина физического маятника, это длина такого математического маятника, период которого совпадает с данным физическим. Будем искать эту величину через период затухающих колебаний физического маятника, который вычисляется по формуле:

(4.1)

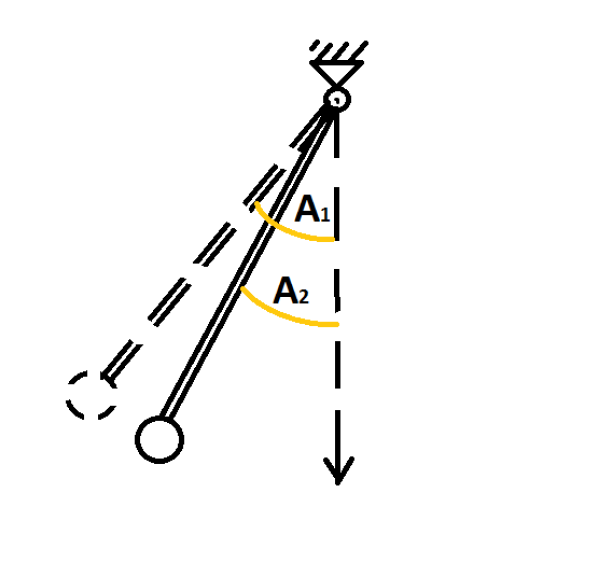
Где – декремент затухания, – циклическая частота собственных незатухающих колебаний маятника. Через приведённую длину L она вычисляется аналогично частоте математического маятника:

Подставим в формулу (4.1) и выразим :

(4.2)

Чтобы определить период, необходимо установить маятник так, чтобы он вращался в строго вертикальной плоскости. Пусть – начальная амплитуда колебаний (**Рис. 2**), тогда с течением времени она будет изменяться по закону:

Под периодом будем условно понимать промежуток времени между двумя последующими локальными максимумами отклонений маятника от положения равновесия: и (**Рис. 3**)



**Рис. 3**

Определив по показаниям датчика моменты и найдём :

А декремент затухания найдём из соотношения :

Подставим в (4.2) и получим

Для большей точности сделаем несколько замеров и возьмём средние значения:

Результат измерения: L

Параметры двигателя тележки:

Для нашей задачи достаточно определить среднее максимальное ускорение тележки, которое способен обеспечить двигатель.

В целях безопасности и сохранности мобильной платформы, мощность двигателя используется в пределах 15% от максимальной. В течении одной секунды мы считываем показания ускорений бортового акселерометра с задержкой в миллисекунду. То есть производим 1000 измерений ускорения:

Затем берём среднее арифметическое:

Результат измерения

**6 Реализация программного обеспечения:**

Bluetooth Модуль

Регулятор хода

Плата 2

Плата 1

Двигатель

Датчики: **…**

**Рис. 4. (Схема мобильного робота)**

Общение с роботом:

Общение с роботом происходит при помощи стационарного компьютера. Команды составляются на компьютере в виде строк в формате:

«Команда» или «Команда;Значение\_1;Значение\_2»

Компьютер посылает строки в виде сигналов на Bluetooth модуль мобильной платформы. В случае успешного получения сигнал подаётся на основную плату Arduino Mega 2560 (Плата 1 **Рис.4.**). В качестве основной платы ранее использовалась DFrobots Rameo - аналог Arduino Leonardoвсе ниже перечисленные программы и их выполнение были реализованы мной на этой плате.

Для обработки платой принятых сигналов и отправки ответов на компьютер используется стандартная библиотека Serial.h. После обработки получаем соответственно строки:

«Команда\r\n» или «Команда;Значение\_1;Значение\_2\r\n»

«\r\n» означает возврат каретки и начало новой строки. Для дальнейшей обработки была реализована функция

ReadMessege(), которая преобразует данные строчки в следующие:

«Команда» или «Команда;Значение\_1;Значение\_2»

А также SplitMessege(), которая разбивает данную строку, разделённую на поля знаком «;» и выдаёт соответствующий массив строк:

[«Команда»] или [«Команда», «Значение\_1», «Значение\_2»]

В таком виде удобно распознавать тип команды, и пользоваться передаваемыми параметрами.

Вспомогательные функции:

Перед запуском управления роботу должна быть доступна информациях о значениях своих параметров, датчики должны быть откалиброваны. Определять постоянные параметры после каждого включения робота не удобно, для этого необходимо записывать их в энергонезависимую память. Для всего перечисленного в программном обеспечении должно быть реализовано множество вспомогательных функций и процедур. Для данной платы мной были реализованы выполнение команд

CALA – калибровка акселерометра и CALE – калибровка мотора, и работа с энергонезависимой памятью.

**CALA()**

CALA() – эта процедура устроена следующим образом: Она производит 1000 измерений компонент вектора ускорения в состоянии покоя робота с задержкой в одну миллисекунду. Затем находит среднее арифметическое по каждой из координат. Результатом работы процедуры является набор усреднённых координат, который мы полагаем за вектор силы тяжести и длина этого вектора. Длина вектора может потребоваться в дальнейшем для перевода в стандартные единицы измерения:

**CALE()**

Первые моторы платформы взаимодействовали с основной платой через библиотеку Motor.h. С помощью данной библиотеки мощность, подаваемая на моторы, задавалась числом от -255 до 255 следующим образом:

0 – отсутствие движения

255 – максимальное ускорение в положительном направлении

-255 – максимальное ускорение в отрицательном направлении

Функция CALE() вызывалась по команде CALE;a[0];a[1];a[2]

Последние три поля команды содержат значения параметров калибровки: время между переключениями a[0], шаг переключения a[1] и количество измерений a[2]. По этим данным создаётся двумерный массив с количеством строк равным количеству шагов и количеством столбцов, равным количеству измерений на каждом шаге: Acc[int(256 / a[1]) + 1][a[2]].

Далее на k – ом шаге на моторы подаётся мощность k \* a[1].

За время шага производятся измерения с задержкой .

Измерения акселерометра представляют собой набор из трёх компонент вектора ускорения. По этим измерениям находим длину вектора и записываем в массив.

После замены платы и моторов на бесколлекторный двигатель повышенной мощности общение основной платы с двигателем и метод калибровки был изменён **Рис. 4.** Регулятор хода посылает на двигатель импульсы с некоторой частотой. Этой частотой контролируется мощность, выдаваемая двигателем. Для общения платы с регулятором хода используется специальная библиотека, которая позволяет задавать сигналы, подаваемые платой на регулятор хода, следующим образом:

1550 – отсутствие движения

3100 – максимальное ускорение в положительном направлении

0 – максимальное ускорение в отрицательном направлении

Эта библиотека реализована на вспомогательной плате Arduino Nano на которую основная плата подаёт числа в диапазоне (0; 3100)

Для обеспечения безопасности и сохранности робота двигатель работает в диапазоне (740; 2360). Актуальная на данный момент процедура CALE() реализует метод, описанный в определении параметров двигателей тележки (см. стр. 11). И вызывается командой: CALE;WorkTime;Direction. Последние два поля команды содержат значения параметров калибровки: продолжительность ускорения WorkTime и направление Direction.

**Работа с энергонезависимой памятью**

На плате Arduino Leonardo существует память объёмом 512 байт с возможностью сохранения информации даже после выключения робота. Для работы с этой памятью в Arduino есть специальная библиотека EEPROM.h. В ней реализованы команды EEPROM.write и EEPROM.read для записи и чтения информации размером в один байт. То есть в качестве аргументов этой команды можно положить информацию, занимающую не больше восьми бит памяти, а также номер байта (будем называть его адресом) в памяти, отводимого под запись. То есть в одну ячейку памяти мы можем записать число от 0 до 256.

Для записи больших значений были реализованы функции eeprom\_write(value, count) и eeprom\_read(addr, count). Первая принимает на вход информацию и её вес в количестве необходимых ячеек, вторая - адрес первой ячейки и количество всех считываемых ячеек. Эти функции работают по принципу перевода в 256-ичную систему исчисления, где ячейка соответствует определённому разряду 256-ичной записи числа.

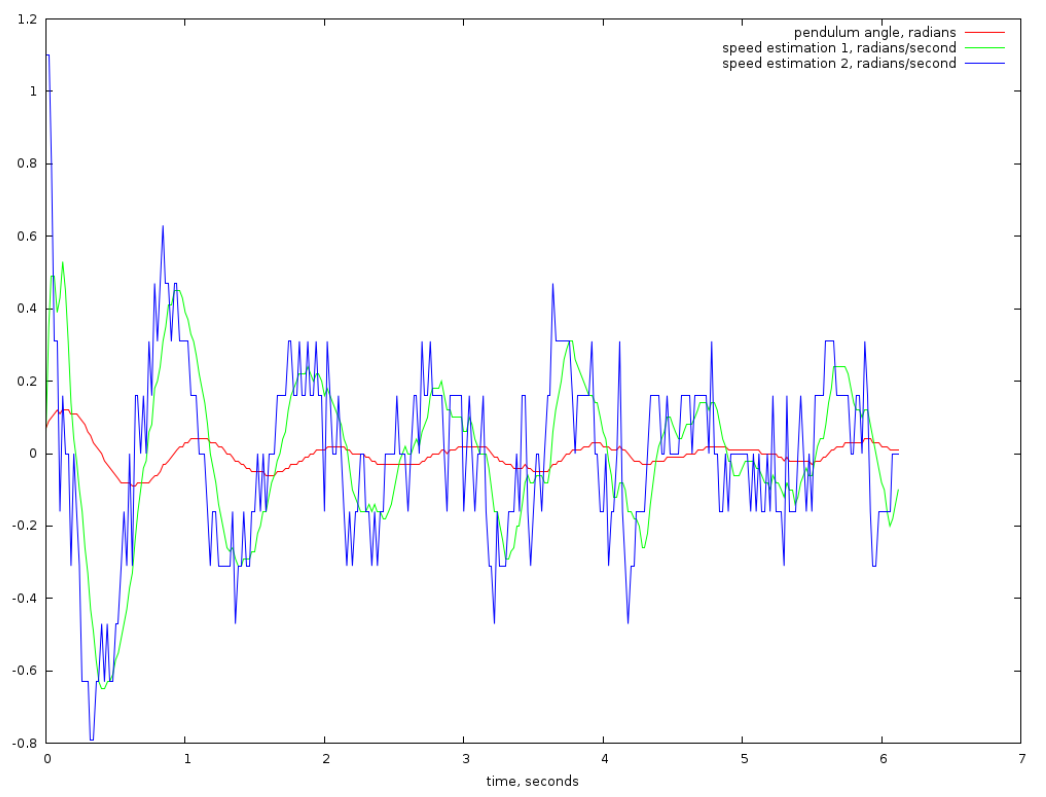
Реализация управления:

В начальный момент времени управления основная плата запускает таймер и фиксирует время t в миллисекундах.

Для определения угла отклонения маятника и положения тележки на прямой на платформе используются два инкрементных энкодера. Каждые 20 миллисекунд эти датчики передают на плату значение угла поворота колеса тележки и угла наклона маятника.

Для определения скорости движения тележки и получения угловой скорости маятника вычисляем оценку следующим образом:

Пусть – текущее показание энкодера, тогда текущая угловая скорость . Однако такая оценка вызывает сильное зашумление (синий график **Рис. 5.**). Чтобы улучшить картину мы будем использовать следующую оценку: , означающую, что мы берём среднюю скорость за 8 предыдущих шагов (зелёный график **Рис. 5.**).



**Рис.5.**

Такая оценка даёт более гладкий график, однако он начинает немного отставать от значений угла. Это видно на **Рис. 5.**: красный график соответствует значениям угла, точки нулевых значений зелёного графика отстают от точек локального максимума красного графика.

В начальный момент управления полагаем , , значение считывается платой с энкодера.

Одна из проблем, связанных с реализацией управления \*, это отсутствие ограничений на управляющую силу .

Управление \* использует заданное ограничение по времени T. Пусть T = 1. Построим график при следующих начальных условиях:

v0 = 0

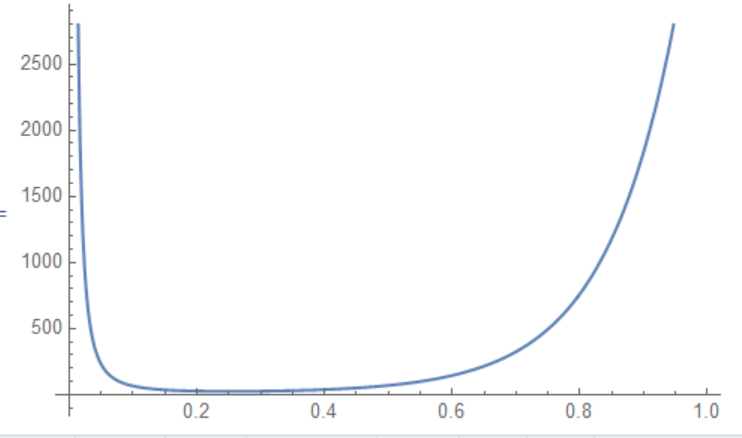
α0 = -10

ω0 = 0



**Рис.6.**

На **Рис.6.** видно, что график резко возрастает в окрестности t = T. Построим график зависимости максимальных значений от конечного времени T (**Рис.7.**).



**Рис.7.**

Наименьшее значение достигается при T сек.

что соответствует

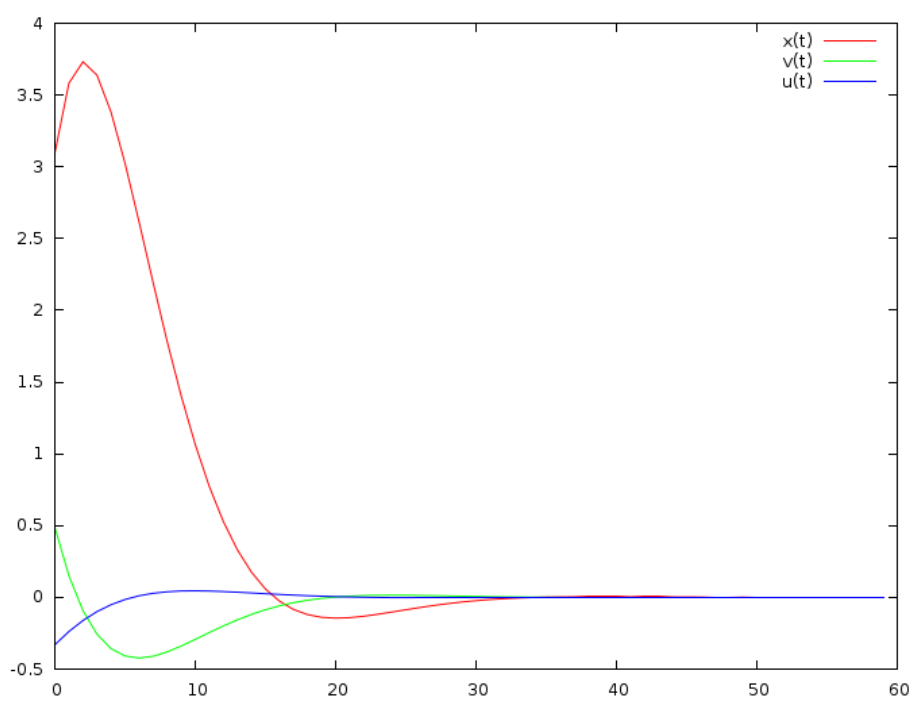
Видно, что наше управление превышает возможности мотора при любом заданном времени T.

Реализация альтернативного управления:

Чтобы протестировать платформу на возможность стабилизации маятника на данном роботе было реализовано управление в виде линейно-квадратичного регулятора. Данный метод широко используется в подобных задачах и прост в реализации. Основной принцип этого метода выглядит следующим образом:

Каждые 20 миллисекунд происходит обновление вектора состояний основная задача найти такие коэффициенты (матрицы) A и B, чтобы , таким образом мы определяем управление на k-м шаге как взвешенную сумму координат и их линейных приращений.

Рассмотрим пример: пусть на k-м шаге система переходит в систему построим графики x, v и



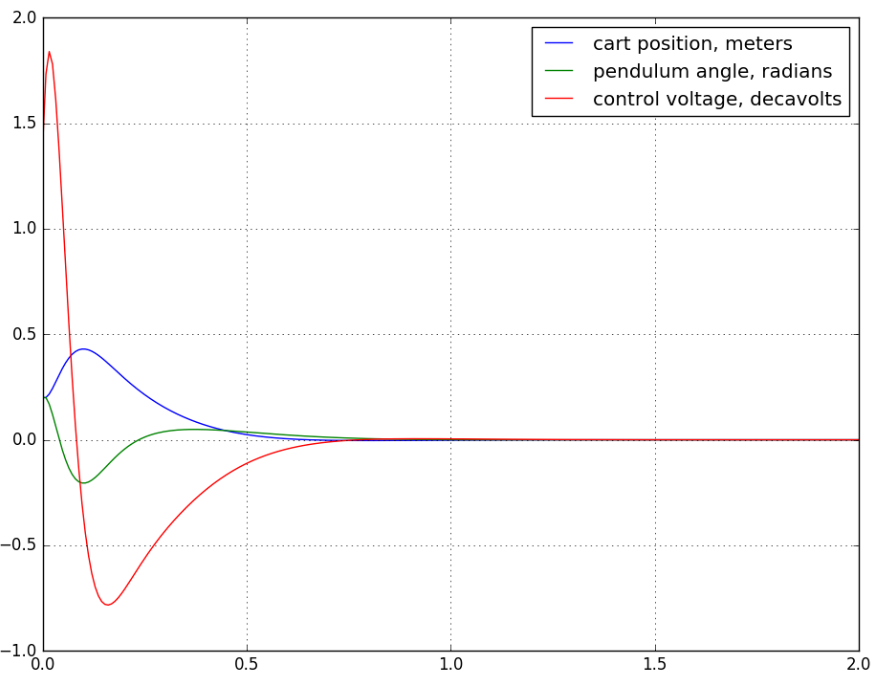
**Рис.8.**

Данный график показывает, что мы можем подобрать коэффициенты a и b такие, что для

Вернёмся к нашему управлению. Из системы дифференциальных уравнений следует:

Дискретизируем нашу систему. Для постоянного шага получаем:

График получившегося управления изображён красным цветом (**Рис.9.**).



**Рис.9.**

Реализация данного управления также не удалась из-за следующих причин:

1) Особенности управления мотором (смена направления движения занимает )

2) Погрешности показаний датчиков (энкодеров) а также оценки скорости изменений параметров приводят к росту погрешности в вычислении управления.

**7 Заключение:**

На данный момент оптимальные управления в чистом виде являются плохо приспособленными для реализации на мобильной платформе. В ходе реализации было выявлено много особенностей использования компонентов платформы. Погрешности датчиков, задержки в управлении двигателем, а также ограниченные возможности двигателя. Для возможной реализации управлений требуется решать задачи с ограничением на u и учитывать в условиях задач задержку по времени, то есть решать задачи поиска ограниченного управления с упреждением, а также по возможности упрощать аналитические решения до линейных итерационных вычислений на примере LQR регулятора.

Что касается реализации самой платформы, следует уделять особое внимание в выборе типов моторов на их совместимость с управляющей платой и задачей, которая перед ними ставится.

**8 Используемая литература**

Формальский, Александр Моисеевич.

Управление движением неустойчивых объектов [Текст] / А. М. Формальский. - Москва : Физматлит, 2012. - 229 с. : ил.; 21 см.; ISBN 978-5-9221-1460-8

М.В. Хлебников, П.С. Щербаков, В.Н. Честнов, (Институт проблем управления им. В.А. Трапезникова РАН, Москва) ЗАДАЧА ЛИНЕЙНО-КВАДРАТИЧНОГО УПРАВЛЕНИЯ

Баландин Д.В., Городецкий С.Ю. КЛАССИЧЕСКИЕ И СОВРЕМЕННЫЕ МЕТОДЫ ПОСТРОЕНИЯ РЕГУЛЯТОРОВ В ПРИМЕРАХ