Муниципальное бюджетное образовательное учреждение “Гимназия №11 г. Ельца”

Проектная работа

“Исследование профиля давлений, действующего на крыло при маневрировании”

Ученик: Шерстяных Илья Сергеевич

Руководитель: Злобин Максим Сергеевич

Елец 2020

**Аннотация**

С. 11. Ил. 2. Литература 3 назв.

Данная работа является актуальной так как создание БЛА сейчас является перспективной областью, также существует проблема расчета критических отклонений плоскостей управления.

Целью проектной работы является создание комплекса для определения критических отклонений плоскостей управления у летательных аппаратов на земле

Тестирование комплекса показало возможность применения его на практике для расчета критических отклонений плоскостей управления БЛА и корректировки отклонений в полете.

Графическая часть

Слайд 2. – Проблема цель и задачи 1

Слайд 3. – Принцип управления БЛА 1

Слайд 4. – Комплектующие устройства 1

Слайд 5. – Принцип работы устройства 1

Слайд 6. – Заключение 1

Оглавление

[Введение 4](#_Toc52114845)

[1. Теоретическая часть 5](#_Toc52114846)

[2. Практическая часть 6](#_Toc52114847)

[2.1 Разработка модели 6](#_Toc52114848)

[2.2 Описание комплектующих 6](#_Toc52114849)

[2.2 Принцип работы устройства 6](#_Toc52114850)

[Заключение 11](#_Toc52114851)

[Список использованных источников 11](#_Toc52114852)

# **Введение**

БЛА являются перспективной областью на сегодняшний день. Они применяются во многих областях жизни человека, поэтому для них важно обеспечить стабильное управление.

Проблема, рассматриваемая в этом проекте это срыв воздушного потока с плоскости управления. Он может привести к полной потери управления беспилотным летательным аппаратом, и в следствии привести к тяжелым последствиям, а также потребовать вкладывания значительных средств на восстановление и ремонт. Во избежание таких случаев необходима настройка и корректировка углов отклонения плоскостей управления, особенно при больших углах атаки.

Актуальность выбранной темы обусловлена тем, что на сегодняшний день в полетных контроллерах отсутствует контроль потока, и расчет критических углов.

Целью проектной работы является создание комплекса для определения критических отклонений плоскостей управления у летательных аппаратов на земле

Для достижения поставленной цели были сформулированы следующие задачи:

* разработать программное обеспечение для сбора информации о профиле давлений, действующих на крыло при маневрировании;
* оценить возможность применения на практике.
* по профилям давлений создать алгоритм вычислений
* создать законченный комплекс применимый для расчетов

# **Теоретическая часть**

Управляющий момент на летательном аппарате создаётся путём создания аэродинамической силы на соответствующей поверхности. Величина этой силы определяется скоростным напором (~Vпр2) и углом отклонения поверхности. Управляющую аэродинамическую силу можно создать:

* отклоняя заднюю кромку, что приведёт к изменению кривизны профиля;
* поворачивая всю поверхность целиком;
* уменьшить подъёмную силу и увеличить сопротивление, сорвав поток интерцептором.

При изменении кривизны профиля (крыла, стабилизатора или киля), на нём меняется аэродинамическая сила. На рисунке показано влияние отклонения элерона на коэффициент подъёмной силы секции крыла. Критический угол отклонения плоскости – это угол при котором происходит срыв воздушного потока с этой плоскости. Он не постоянен, например: при выполнении манёвра с увеличением угла тангажа угол атаки стабилизатора увеличивается из-за угловой скорости вращения самолёта (аэродинамическое демпфирование). Это означает, что требуемый угол отклонения руля высоты будет больше, чем при тех же условиях в горизонтальном полёте. Величина дополнительного расхода руля пропорциональна созданной перегрузке. Располагаемый угол отклонения руля высоты должен обеспечивать достижение предельно-допустимой перегрузки. Наибольшее отклонение руля высоты требуется во время выравнивания самолёта на посадке в зоне экрана земной поверхности при предельно передней центровке.

# **Практическая часть**

# Разработка

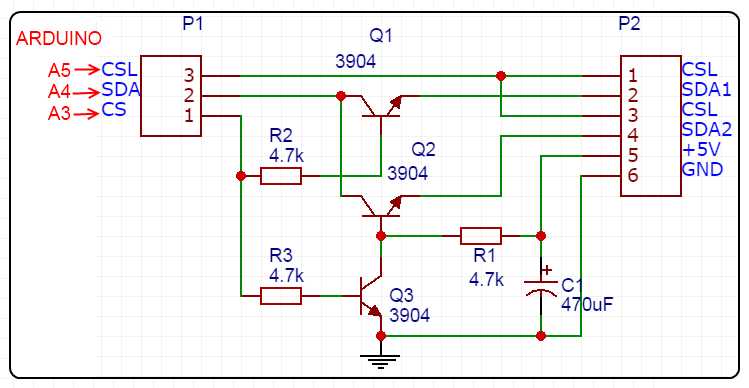


Рисунок 1 – схема I2C хаба

Q1, Q2 – ключи переключающие sda шины датчиков давления. Q3 – фазоинвертор для второго ключа. Тактирование ключей осуществляется с цифрового пина 4 Arduino. При переключении датчиков происходит их инициализация и считывание показаний.

void changeBMP(){

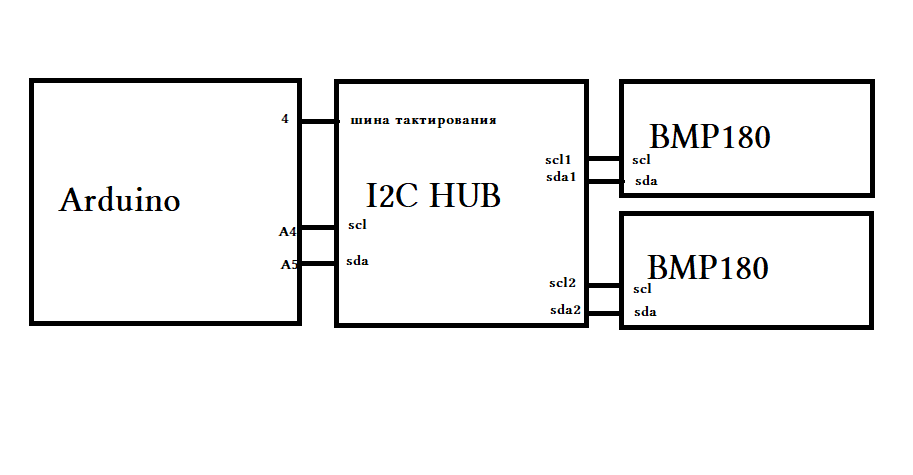
state = !state;

digitalWrite(4, state);

pressure.begin();

delay(10);

}



# 2.2 Описание комплектующих

К комплектующим устройства относятся:

* Arduino Uno аппаратная платформа с микроконтроллером Atmega328;
* Датчик общего давления BMP180;
* Транзистор Bc337 – 3шт;
* Резистор 10кoм – 3шт;

# Принцип работы устройства

changeBMP – функция управления хабом для смены датчиков. getPressure – сбор данных о текущем давлении. Все данные отсылаются в параллельный порт для сбора и последующей обработки на компьютере. Первый датчик отвечает за мониторинг активной части плоскости управления, второй давление непосредственно перед плоскостью.

#include <Adafruit\_BMP085.h>

#include <Servo.h>

#include <Wire.h>

double P01, P02, P1, P2;

bool state = HIGH;

int dead = 500;

//char wer;

Adafruit\_BMP085 pressure;

Servo servo;

Servo servo1;

double getPressure() {

double a = 0;

for (byte i = 0; i < 10; i++) {

a += pressure.readPressure();

}

double b = a / 10;

return b;

}

double getDefaultPress(){

double P, p;

for(int i = 0; i < 11; i++){

p = getPressure();

P += p;

delay(50);

}

return P / 10;

}

void changeBMP(){

state = !state;

digitalWrite(4, state);

pressure.begin();

delay(10);

}

void setup(){

Serial.begin(9600);

Serial.println("RTR");

pinMode(4, OUTPUT);

changeBMP();

P01 = getPressure();

changeBMP();

P02 = getPressure();

servo.attach(9);

servo1.attach(10);

}

void test(){

int alpha = 0;

servo.write(30);

servo1.write(150);

delay(1000);

servo.write(150);

servo1.write(30);

delay(1000);

changeBMP();

P1 = getPressure();

changeBMP();

P2 = getPressure();

servo1.write(90 + alpha);

delay(dead);

while((-(P1 - P01) < P2 - P02) && alpha <= 90){

changeBMP();

P1 = getPressure();

changeBMP();

P2 = getPressure();

servo.write(90 + alpha);

servo1.write(90 + alpha);

delay(dead);

alpha++;

}

Serial.println(alpha);

alpha = 0;

servo1.write(90 + alpha);

delay(dead);

while(-(P1 - P01) < P2 - P02 && alpha <= 90){

changeBMP();

P1 = getPressure();

changeBMP();

P2 = getPressure();

servo.write(90 + alpha);

servo1.write(90 - alpha);

delay(dead);

alpha++;

}

Serial.println(alpha);

}

void loop(){

if (Serial.available()){

Serial.read();

delay(10);

Serial.println("Started");

test();

}

}

Код модуля обработки

void test()- основная функция программы, в ней происходит сбор данных о критических углах. Раз в секунду сервопривод при условии что не произошел срыв потока(определяется через разницы давлений и кавитацию на датчиках (-(P1 - P01) < P2 - P02)) Конечный угол отправляется в паралленьный порт у обрабатывается на компьютере.

from top import Ui\_MainWindowfrom PyQt5 import QtCore, QtGui, QtWidgets

import sys

from time import sleep

import serial

import glob

class main(QtWidgets.QMainWindow, Ui\_MainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setupUi(self)

def connect():

a = serial\_ports()

try:

realport = serial.Serial(a[window.spinBox.value()], 9600)

awer = realport.readline()

print(awer)

except Exception as e:

print(e)

return

window.lineEdit\_2.setText('started')

realport.write(b'RTR\r\n')

print('send')

window.lineEdit.setText(realport.readline().decode())

window.progressBar.setValue(1)

a = realport.readline()

window.progressBar.setValue(50)

b = realport.readline()

window.progressBar.setValue(100)

window.lineEdit.setText(a.decode())

window.lineEdit\_2.setText(b.decode())

def serial\_ports():

if sys.platform.startswith('win'):

ports = ['COM%s' % (i + 1) for i in range(256)]

elif sys.platform.startswith('linux') or sys.platform.startswith('cygwin'):

ports = glob.glob('/dev/tty[A-Za-z]\*')

elif sys.platform.startswith('darwin'):

ports = glob.glob('/dev/tty.\*')

else:

raise EnvironmentError('Unsupported platform')

result = []

for port in ports:

try:

s = serial.Serial(port)

s.close()

result.append(port)

except (OSError, serial.SerialException):

pass

return result

app = QtWidgets.QApplication(sys.argv)

window = main()

window.show()

window.progressBar.setValue(0)

window.pushButton.clicked.connect(connect)

app.exec()

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(441, 194)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.pushButton = QtWidgets.QPushButton(self.centralwidget)

self.pushButton.setGeometry(QtCore.QRect(313, 110, 111, 27))

self.pushButton.setObjectName("pushButton")

self.lineEdit = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit.setGeometry(QtCore.QRect(310, 10, 113, 25))

self.lineEdit.setObjectName("lineEdit")

self.label = QtWidgets.QLabel(self.centralwidget)

self.label.setGeometry(QtCore.QRect(10, 10, 291, 19))

self.label.setObjectName("label")

self.label\_2 = QtWidgets.QLabel(self.centralwidget)

self.label\_2.setGeometry(QtCore.QRect(10, 60, 291, 19))

self.label\_2.setObjectName("label\_2")

self.lineEdit\_2 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_2.setGeometry(QtCore.QRect(310, 60, 113, 25))

self.lineEdit\_2.setObjectName("lineEdit\_2")

self.progressBar = QtWidgets.QProgressBar(self.centralwidget)

self.progressBar.setGeometry(QtCore.QRect(10, 110, 291, 21))

self.progressBar.setProperty("value", 24)

self.progressBar.setObjectName("progressBar")

MainWindow.setCentralWidget(self.centralwidget)

self.menubar = QtWidgets.QMenuBar(MainWindow)

self.menubar.setGeometry(QtCore.QRect(0, 0, 441, 27))

self.menubar.setObjectName("menubar")

MainWindow.setMenuBar(self.menubar)

self.statusbar = QtWidgets.QStatusBar(MainWindow)

self.statusbar.setObjectName("statusbar")

MainWindow.setStatusBar(self.statusbar)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))

self.pushButton.setText(\_translate("MainWindow", "Старт"))

self.label.setText(\_translate("MainWindow", "Критический угол отклонения тангажа")) self.label\_2.setText(\_translate("MainWindow", "Критический угол отклонения элеронов"))

Код Desktop приложения

Компьютер отсылает сигнал старта теста, и проверяет данные параллельного порта, по окончанию он отображает данные на экран, GUI интерфейс построен на базе библиотеки PyQt. Общение через COM-порт на pySerial.

# **Заключение**

Проектная работа посвящена созданию комплекса для определения критических отклонений плоскостей управления у летательных аппаратов на земле.

В ходе выполнения проектной работы было создан комплекс и разработаны алгоритмы применимые для расчетов в полетных контроллерах.

В дальнейшем в плане развития проекта стоит создание алгоритма для работы с дифференциальными элеронами и элеронами Фрайза.

# **Список использованных источников**

1. В. Л. Балакин, Ю. Н. Лазарев динамика полета самолета. устойчивость и управляемость продольного движения
2. М. В. Чудаков Построение поляр и расчет динамики полета дозвуковых транспортных и пассажирских самолетов
3. Определение предельных динамических характеристик рулевого привода на основе предельных режимов полета самолета Т.Б. Близнова, Ю.Г. Оболенский, В.А. Полковников