

Департамент образования и науки города Москвы
Государственное бюджетное общеобразовательное учреждение
«Школа №1034 имени Героя Советского Союза имени В.В. Маркина

ДОКУМЕНТАЦИЯ ПО КОМАНДНОМУ КЕЙСУ №2
МОСКОВСКОЙ ПРЕДПРОФЕССИОНАЛЬНОЙ ОЛИМПИАДЫ
ШКОЛЬНИКОВ
«Безопасный маршрут»

Работу выполнила
Команда «Буравчики»
Ученики 11 «И» класса ГБОУ Школы
№1034

Симбирских Иван
Ткаченко Александр
Забаров Дмитрий
Горюнов Тихон
Вовк Тимофей

Научный руководитель
Ольховская Ирина Григорьевна
учитель физики.

Оглавление

| | |
|--|----|
| Оглавление | 2 |
| 1.Цель и задачи | 3 |
| 2.Команда | 4 |
| 3. Функции | 5 |
| 4. Описание используемых аппаратных и программных узлов, модулей, фреймворков и других инструментов | 5 |
| 5. Этапы работы..... | 6 |
| 5.1. Создание поля и цилиндров | 6 |
| 5.2. Программирование и создание схемы | 8 |
| 5.3. Сборка всех компонентов..... | 13 |
| 6. UML-диаграммы..... | 14 |
| 6.1. Диаграмма вариантов пользовательского взаимодействия с системой (use case diagram)..... | 14 |
| 6.2. Диаграмма автомата (state machine diagram)..... | 14 |
| 6.3 Диаграмма последовательности (sequence diagram)..... | 15 |
| 6.4 Диаграмма компонентов (component diagram)..... | 15 |
| 7. Итог работы | 16 |
| 8. Список литературных источников | 16 |

1.Цель и задачи

Цель: спроектировать модель роботизированного мобильного устройства и разработать алгоритм для этого устройства, которое способно перемещаться по полигону, распознавать, захватывать и перемещать объекты.

Задачи:

1. Изучить принцип работы устройства
2. Изучить основы 3D моделирования и печати с использованием учебных программ (Компас-3D)
3. Освоить основы программирования языков: C++, Python
4. Разработать код для плат: Raspberry PI, Arduino Uno
5. Разработать полигон для устройства по заданным размерам
6. Напечатать объекты заданных размеров разных цветов на 3D принтере
7. Отладить совместную работу плат
8. Протестировать устройство и отладить работу

2. Команда

Наша команда «Буравчики», состоит из 5 человек (Таблица 1). Каждый участник умеет работать с платами и имеет широкие знания в области программирования.

Таблица 1

| Ф.И. Участника | Роль | Функции | Обязанности |
|-----------------------|---------------------|--------------------------------------|---|
| Ткаченко Александр | Инженер-конструктор | Разработка и создание конструкций | Собрать все подсистемы и выполнить финальную сборку |
| Симбирских Иван | Схемотехник | Разработка структур электронных схем | Разработать электронную схему для модели |

| | | | |
|--------------------|-------------------------|--|---|
| Забаров Дмитрий | Программист | Программирование роботизированного устройства | Написать код для робота на Arduino и Raspberry PI |
| Горюнов Тихон | Инженер- конструктор | Разработка и создание конструкций | Собрать все подсистемы и выполнить финальную сборку |
| Вовк Тимофей | 3D-моделист | Создание 3D моделей с помощью учебных программ | Создать и напечатать объекты разных цветов |

3. Функции

Функции разработанного решения:

- 1) Робот способен автономно перемещаться по трассе, избегая падения в люки
- 2) Робот способен распознавать препятствия
- 3) Используя клешню, робот может взаимодействовать с грузами
- 4) При потере необходимого люка робот способен найти его
- 5) Робот способен ориентироваться по полю при помощи метки в центре поля

4. Описание используемых аппаратных и программных узлов, модулей, фреймворков и других инструментов

1) Программное обеспечение:

- a. КОМПАС-3D
- b. Arduino IDE
- c. Python IDLE

2) Электронные компоненты:

- a. Плата Raspberry Pi 3

- b. Плата Technolab
- c. Моторы 5шт.
- d. Камера Raspberry Pi
- e. Драйвер L298N
- f. Сервопривод SG90
- g. Батарейный отсек с батарейками

3) Материалы и оборудование

- a. Фанера
- b. Пластик PLA
- c. Мультиметр
- d. Паяльник
- e. Клей-пистолет
- f. 3D-принтер
- g. ПК
- h. Набор инструментов
- i. Лобзик

5. Этапы работы

5.1. Создание поля и цилиндров

Первым этапом нашей работы стало создание цилиндрических грузов и поля. Грузы были напечатаны на 3D-принтере. При их разработке использовалась программа КОМПАС-3D (Рисунок 1). После цилиндров нами было изготовлено поле, состоящее из фанеры. Границы поля и люков были сделаны при помощи изоленды и краски (Рисунок 2).

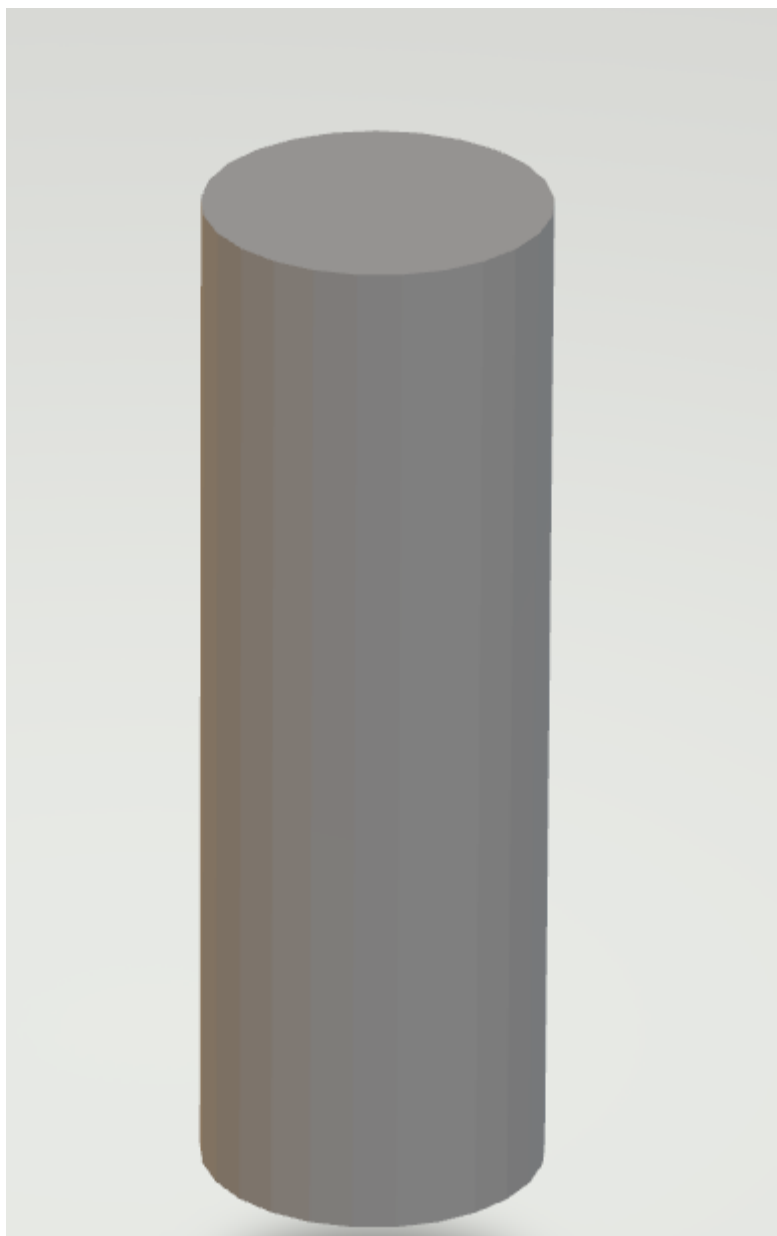


Рисунок 1. Цилиндр

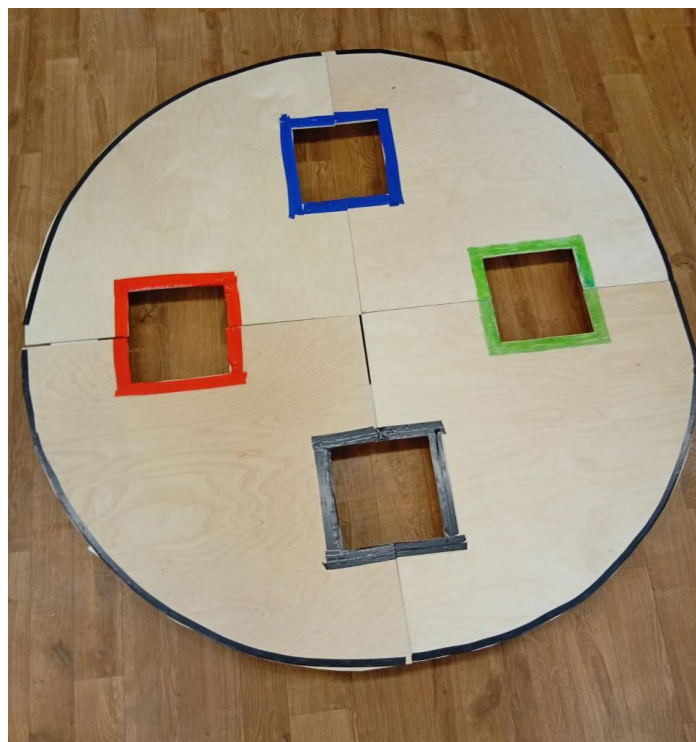


Рисунок 2. Поле

5.2. Программирование и создание схемы

Сперва была разработана блок-схема алгоритма действий робота (Рисунок 3). Затем нужно было запрограммировать плату Technolab (Рисунок 4), чтобы робот мог взаимодействовать со всеми компонентами электрической схемы, и смоделировать принципиальную схему её работы (Рисунок 5). Также было необходимо разработать программу на Python для Raspberry Pi, чтобы робот мог использовать камеру Open-CV (Рисунок 6, Рисунок 7). Ссылка на программный код: <https://github.com/Zloybig/PredProf>

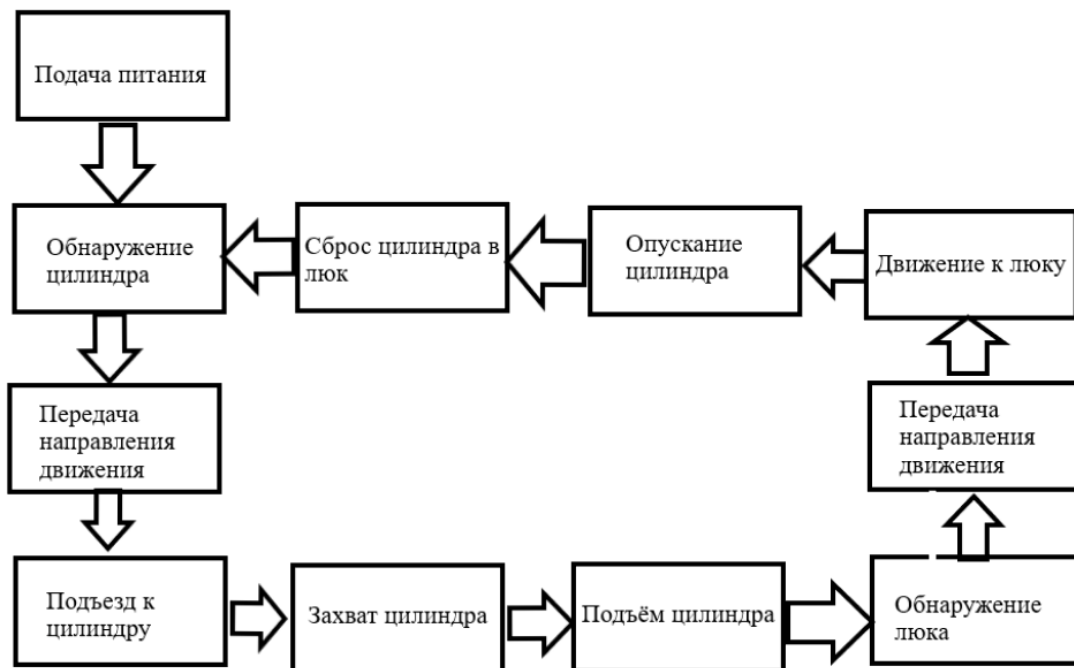


Рисунок 3. Блок-схема

```

1  #include <Servo.h>
2
3  // Моторы (Танковая схема: 4 мотора через 2 драйвера или параллельно)
4  const int L_PWM = 5; const int L_DIR = 6;
5  const int R_PWM = 10; const int R_DIR = 9;
6  const int SERVO_PIN = 11;
7
8  Servo grabber;
9
10 void setup() {
11   Serial.begin(9600);
12   pinMode(L_PWM, OUTPUT); pinMode(L_DIR, OUTPUT);
13   pinMode(R_PWM, OUTPUT); pinMode(R_DIR, OUTPUT);
14   grabber.attach(SERVO_PIN);
15   grabber.write(90); // Открыто
16 }
17
18 void drive(int l, int r) {
19   digitalWrite(L_DIR, l >= 0 ? HIGH : LOW);
20   analogWrite(L_PWM, abs(l));
21   digitalWrite(R_DIR, r >= 0 ? HIGH : LOW);
22   analogWrite(R_PWM, abs(r));
23 }
24
25 void loop() {
26   if (Serial.available() > 0) {
27     String cmd = Serial.readStringUntil('\n');
28     cmd.trim();
29
30     if (cmd == "FORWARD") drive(150, 150);
31     else if (cmd == "BACK") drive(-150, -150);
32     else if (cmd == "LEFT") drive(-130, 130);
33     else if (cmd == "RIGHT") drive(130, -130);
34     else if (cmd == "SCAN") drive(100, -100);
35     else if (cmd == "STOP") drive(0, 0);
36     else if (cmd == "GRAB") grabber.write(170); // Закрыть
37     else if (cmd == "DROP") grabber.write(90); // Открыть
38   }
39 }
  
```

Рисунок 4. Программа для Technolab

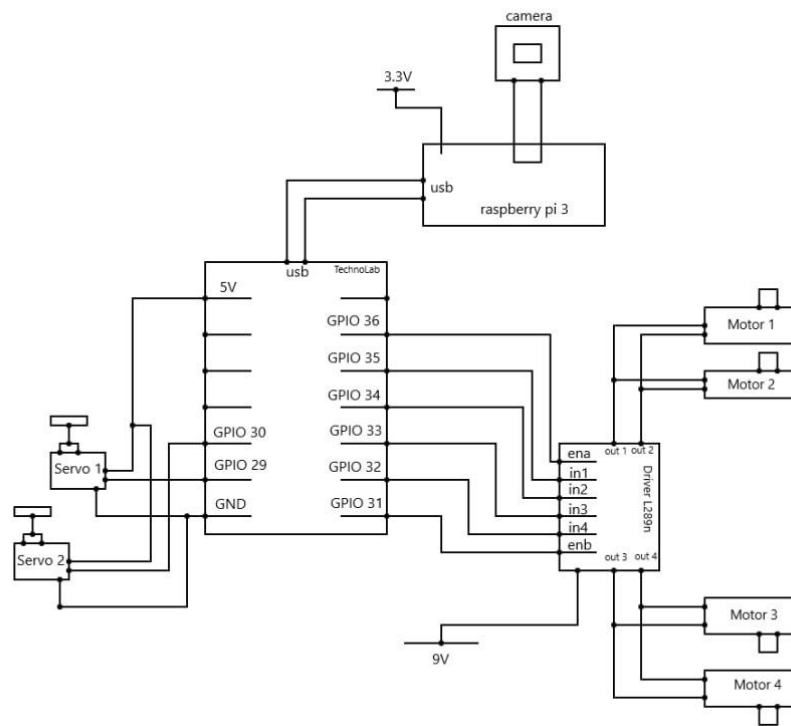


Рисунок 5. Принципиальная схема

```

picam2 = Picamera2()
config = picam2.create_preview_configuration(main={"format": "RGB888", "size": (640, 480)})
picam2.configure(config)
picam2.start()

picam2.set_controls({"AfMode": 2})

print("Робот запущен. Ошибка 'Screen' устранена.")

try:
    while True:
        frame = picam2.capture_array()

        frame = cv2.cvtColor(frame, cv2.COLOR_RGB2BGR)
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
        mask = cv2.inRange(hsv, np.array([0, 150, 70]), np.array([10, 255, 255]))

        contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

        found = False
        for cnt in contours:
            if cv2.contourArea(cnt) > 500:
                x, y, w, h = cv2.boundingRect(cnt)

                ratio = h / float(w)
                if 2.5 < ratio < 4.0:
                    cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
                    cx = x + w // 2

                    error = cx - 320
                    if abs(error) < 40:
                        if ser: ser.write(b"FORWARD\n")
                    elif error > 0:
                        if ser: ser.write(b"RIGHT\n")
                    else:
                        if ser: ser.write(b"LEFT\n")
                    found = True
                    break

            if not found:
                if ser: ser.write(b"SCAN\n")

        cv2.imshow("Vision Debug", frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

finally:
    if ser: ser.write(b"STOP\n")
    picam2.stop()

```

Рисунок 6. Фрагмент программы на Python

```

if STATE == "SEARCH_OBJ":
    found, cx, w, _ = get_target_center(frame, target_color)
    if found:
        STATE = "APPROACH_OBJ"
    else:
        send_to_arduino("MOVE", 0, 30)

elif STATE == "APPROACH_OBJ":
    found, cx, w, _ = get_target_center(frame, target_color)
    if found:
        error = cx - center_scr
        turn = int(error * 0.4)

        if w > 120:
            send_to_arduino("STOP")
            send_to_arduino("GRAB")
            time.sleep(2)
            STATE = "SEARCH_HOLE"
        else:
            send_to_arduino("MOVE", 40, turn)
    else:
        STATE = "SEARCH_OBJ"

elif STATE == "SEARCH_HOLE":
    search_c = target_color if target_color != "white" else "black"
    found, cx, w, _ = get_target_center(frame, search_c)

    if found:
        error = cx - center_scr
        turn = int(error * 0.4)
        if w > 180:
            send_to_arduino("STOP")
            send_to_arduino("DROP")
            time.sleep(2)
            STATE = "SEARCH_OBJ"
        else:
            send_to_arduino("MOVE", 35, turn)
    else:
        send_to_arduino("MOVE", 0, -30)

cv2.imshow("CSI Camera Robot", frame)
if cv2.waitKey(1) & 0xFF == ord('q'): break

finally:
    send_to_arduino("STOP")
    cap.release()
    cv2.destroyAllWindows()

```

Рисунок 7. Фрагмент программы на Python

5.3 Сборка всех компонентов

Последним этапом решения кейсового задания является финальная сборка. (Рисунок 8). Движущиеся элементы робота представлены на кинематической схеме (Рисунок 9).

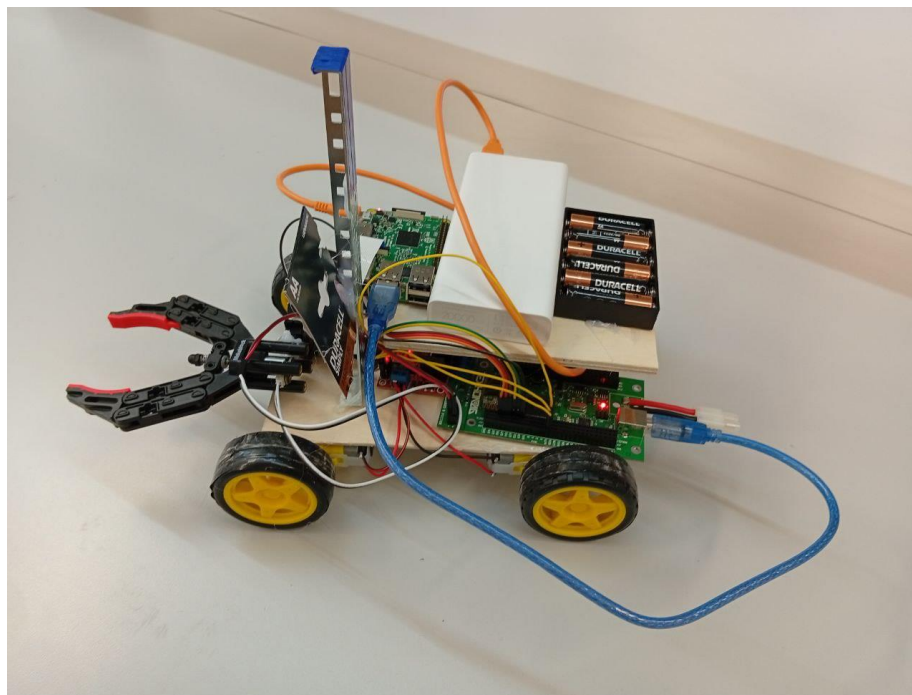


Рисунок 8. Робот

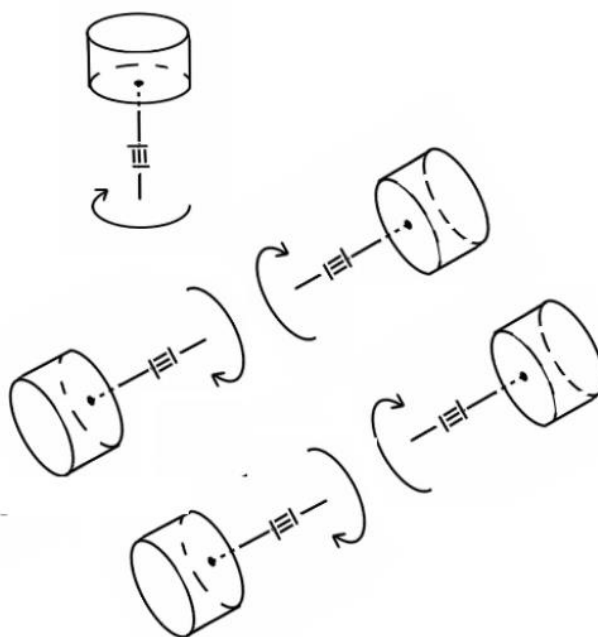
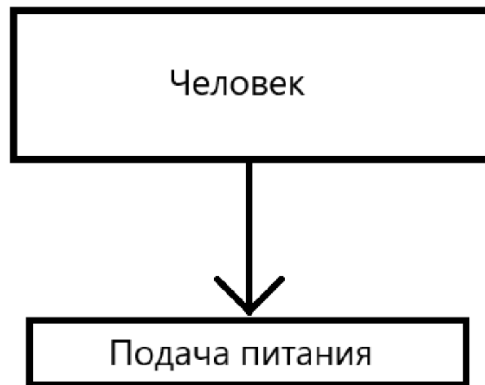


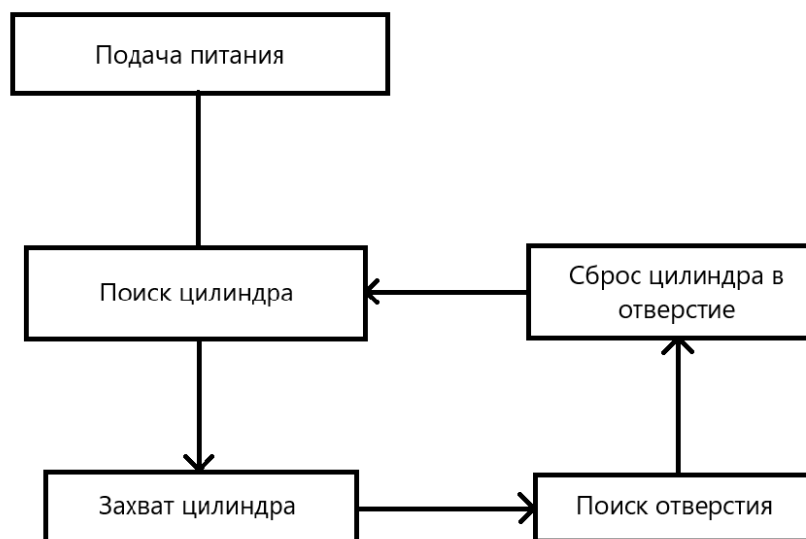
Рисунок 9. Кинематическая схема

6. UML-диаграммы

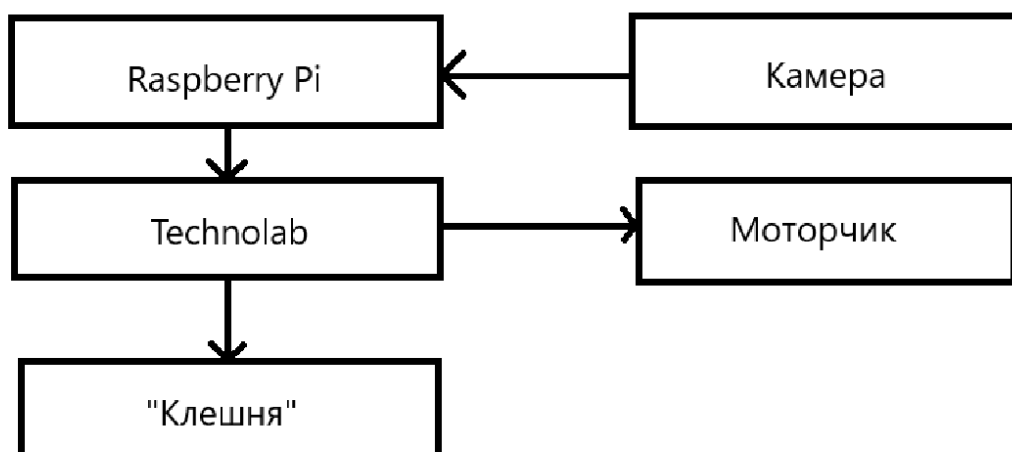
6.1 Диаграмма вариантов пользовательского взаимодействия с системой (use case diagram)



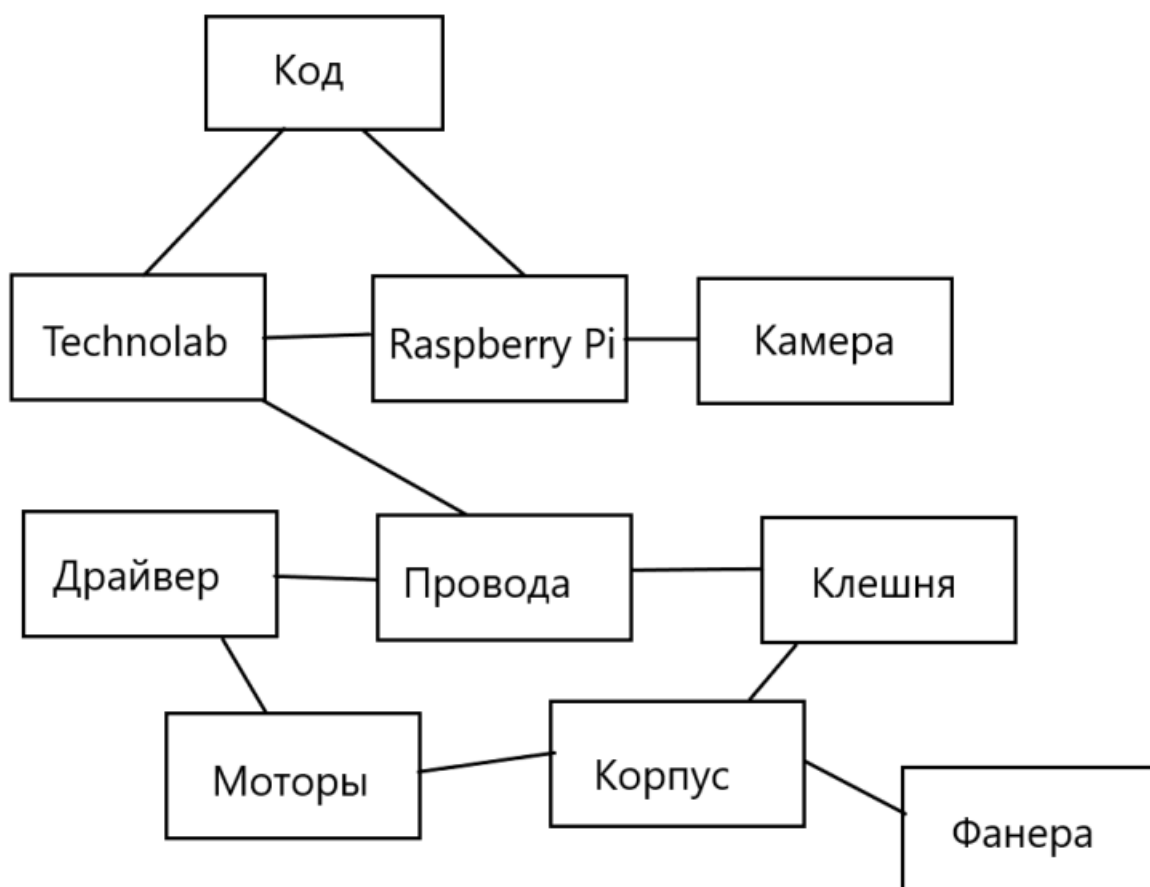
6.2 Диаграмма автомата (state machine diagram)



6.3 Диаграмма последовательности (sequence diagram)



6.4 Диаграмма компонентов (component diagram)



7. Итог работы

В ходе работы нашей командой было создано поле для проведения испытаний и цилиндры, выступающие в качестве груза; разработан робот, способный автономно перемещаться по полю и доставлять грузы определённого цвета в необходимые люки; проведены испытания робота и записано финальное видео.

8. Список литературных источников

1. TinkerCad создание схем и Arduino проектов. URL: <https://arduino-tex.ru/news/1/izuchaem-arduino-bez-arduino-c-pomoshchyu-tinkercad-i-ego-servisov.html>
2. Raspberry gPIo. URL: <https://learn.sparkfun.com/tutorials/raspberry-gpio/all>
3. База знаний Амперки: инструкции и подсказки по Arduino и Raspberry Pi, оригинальные проекты, схемы распиновки модулей и datasheet'ы, теория электричества для начинающих и другая полезная информация. URL: <http://wiki.amperka.ru/>
4. Документация по GitHub. URL: <https://docs.github.com/ru/get-started/quickstart/hello-world>