Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«ПЕТРОЗАВОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных технологий

Отчет о прохождении производственной практики

Выполнил студент группы 22307:
Яскеляйнен Семён Дмитриевич

Направление подготовки:
Программная инженерия

Место прохождения практики:
Кафедра ИМО, ЦИИ

Сроки прохождения практики:
27.05.23 - 08.06.23

Руководитель практики:
к.т.н., доцент
Богоявленская Ольга Юрьевна

Оценка _____

Дата _____

Петрозаводск 2023 г.

Содержание

Введение

Робототехника — прикладная наука, занимающаяся разработкой автоматизированных технических систем и являющаяся важнейшей технической основой развития производства[1]. Робототехника опирается на такие дисциплины, как электроника, механика, кибернетика, телемеханика, мехатроника, информатика, а также радиотехника и электротехника. Выделяют строительную, промышленную, бытовую, медицинскую, авиационную и экстремальную (военную, космическую, подводную) робототехнику.

В рамках весеннего семестра по ТППО нам необходимо было реализовать программную библиотеку, которая получает данные с датчиков. Одним из датчиков, установленных на роботе, являлась видеокамера, для которой надо было реализовать следующее:

1. Получение данных с камеры;
2. Разработка способа представления этих данных оператору.

Таким образом, целью практики будет являться описание актуальных результатов для решения задачи получения видеоизображения с камеры и последующего его отображения.

Выделим задачи практики:

1. Представление решения по получению данных с камеры;
2. Представление решения по отображению данных с камеры;
3. Представление решения по формированию dataset;
4. Поиск и анализ аналогов полученного решения.

1 Полученное решение

1.1 Получение данных с камеры

В нашем проекте использовалась стандартная RPI Camera Module 3, которая была перепаяна, чтобы была возможность подключения по USB проводу. Камера позволяет записывать видео с разрешением 1080p 50 кадров в секунду.[2] На рисунке 1 можно увидеть камеру, присоединённую к зелёному манипулятору.

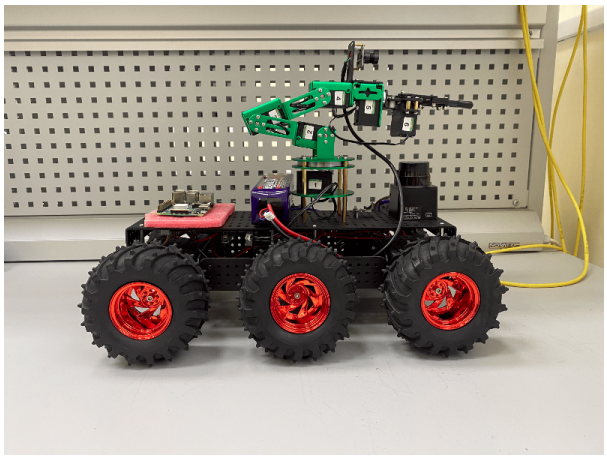

Рис. 1: Прототип робота

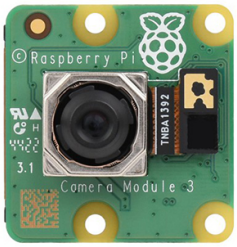

Рис. 2: RPI Camera

Для захвата изображения с камеры использовалась программная библиотека OpenCV, предназначенная для анализа, классификации и обработки изображения. Весь код проекта выполнялся на языке Python, поэтому и установка библиотеки была осуществлена в виде установки модуля данного языка. Для простого получения видео с камеры в интернете очень много фрагментов кода, приведём пример одного из них:

```
1  import numpy as np
2  import cv2 as cv
3  cap = cv.VideoCapture(0)
4  if not cap.isOpened():
5      print("Cannot open camera")
6      exit()
7  while True:
8      # Capture frame-by-frame
9      ret, frame = cap.read()
10     # if frame is read correctly ret is True
11     if not ret:
```

```
12          print("Can't receive frame (stream end?). Exiting ...")
13          break
14      # Our operations on the frame come here
15      gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
16      # Display the resulting frame
17      cv.imshow('frame', gray)
18      if cv.waitKey(1) == ord('q'):
19          break
20  # When everything done, release the capture
21  cap.release()
22  cv.destroyAllWindows()
```

1.2 Отображение данных оператору

В рамках проекта необходимо было не только обеспечить получение данных с камеры, но и реализовать демонстрацию этих данных оператору. Проблема в том, что на самом роботе не находится какой-нибудь экран, который оператор мог бы просматривать, а также сам оператор может находиться удалённо от робота и просто подключаться к роботу по SSH соединению. Отсюда возникает потребность в разработке способа удалённого просмотра данных с камеры.

Для решения данной проблемы было решено использовать робота в качестве веб-сервера, который публикует в локальную сеть данные с камеры. При запуске скрипта на роботе запускался сервер, который ожидал подключений на веб-страницу в локальной сети. При подключении оператора на данную веб-страницу начиналась отправка видеопотока с камеры. На рисунке 3 представлена схема передачи изображения оператору.

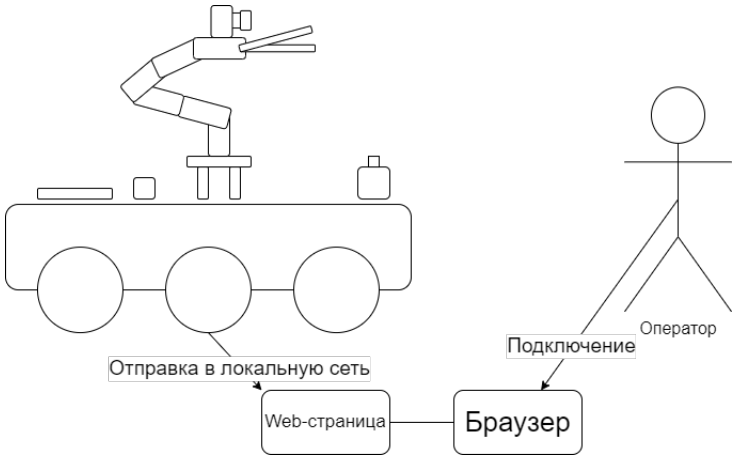

Рис. 3: Схема передачи изображения оператору

Была реализована возможность подключения сразу нескольких пользователь на страницу, у обоих будут корректно отображаться данные с камеры. На рисунке 4 приведён пример просмотра данных с камеры.

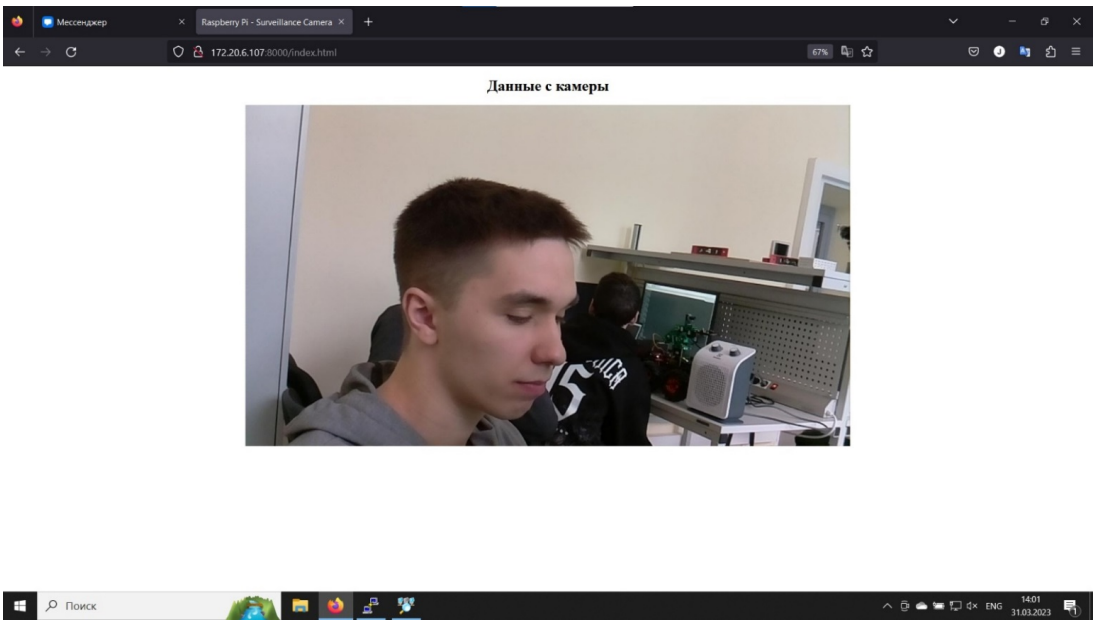

Рис. 4: Отображение данных с камеры

В нашем проекте предусматривалось расширение числа датчиков, поэтому также была предусмотрена возможность использование сразу нескольких камер. В таком случае для каждой из камер будет создана своя веб-страница, на которую также сможет зайти несколько пользователей для просмотра данных с камеры.

Минусом полученного решения является то, что в качестве веб-сервера выступал наш одноплатный компьютер, у которого очень ограниченная вычислительная мощность. Из-за этого все вычисления могут проходить очень долго и робот может медленно работать. Поэтому в дальнейшем планируется, что в качестве сервера будет выступать дополнительная машина, можно будет передавать изображение с raspberry на компьютер, а он уже будет публиковать изображение в локальную сеть.

1.3 Формирование dataset

Также в рамках развития проекта планируется использование нейронной сети, которая будет анализировать данные с камеры и классифицировать препятствия. Для тренировки нейронной сети необходимо формировать dataset, в котором будет очень большое число фоток препятствий с различных ракурсов. Для облегчения работы разработчиков и более точного формирования dataset был разработан скрипт,

который автоматизировал создание снимков и запись их для дальнейшего использования. Ниже приведён листинг данного скрипта:

```
#!/usr/bin/python3
import cv2

cam_port = 0
cam = cv2.VideoCapture(cam_port)

count = 5

for i in range(count):
    result, image = cam.read()

    if result:
        cv2.imwrite("./images/" + str(i) + ".png", image)

    else:
        print("No image detected. Please! try again")
```

2 Аналоги полученного решения

2.1 RTSP-протокол

2Одним из вариантов решения проблемы получения данных с видеокамеры была реализация RTSP-протокола. RTSP - прикладной протокол, предназначенный для использования в системах, работающих с мультимедийными данными (мультимедийным содержимым, медиасодержимым), и позволяющий удалённо управлять потоком данных с сервера, предоставляя возможность выполнения команд, таких как запуск (старт), приостановку (пауза) и остановку (стоп) вещания (проигрывания) мультимедийного содержимого, а также доступа по времени к файлам, расположенным на сервере.[3] Минусом данного решения стало то, что не на всех устройствах есть приложение для просмотра этого самого видео, полученного через RTSP-протокол, эту проблему можно было бы решить, отправляю полученное видео в браузер. Пример, трансляции видео, полученного через RTSP-протокол в браузер можно увидеть на рисунке 5.

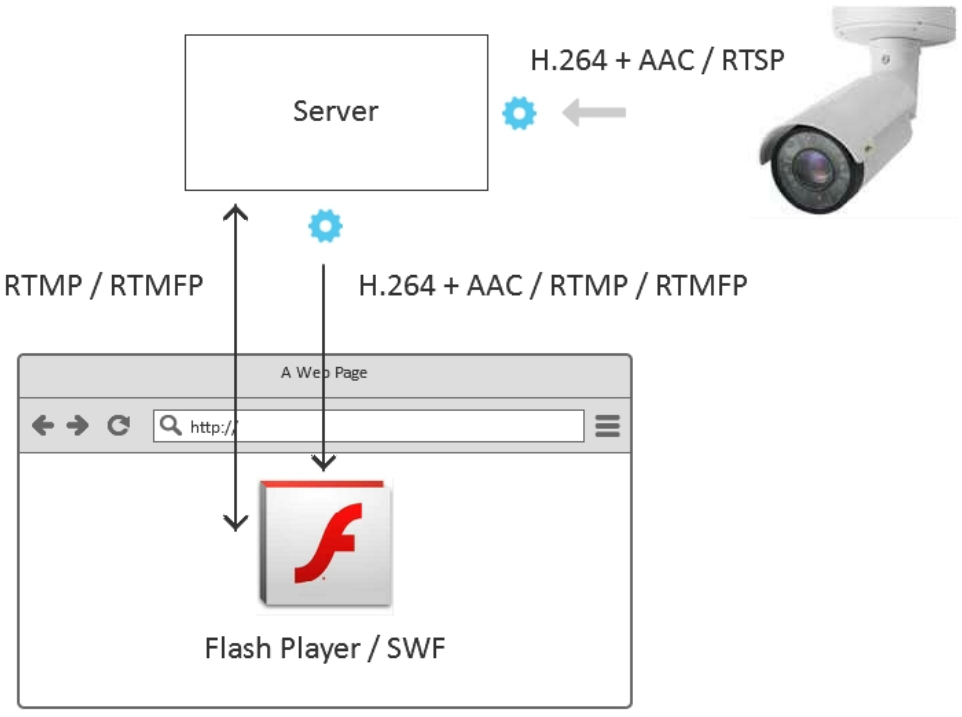

Рис. 5: Трансляция RTSP видео в браузер

Данное решение повлекло бы написание дополнительного кода, поэтому было решено изначально публиковать видео с камеры на страницу в локальной сети, что в итоге и было реализовано, ведь сейчас даже на всех мобильных устройствах есть

какой-либо браузер, который и позволит оператору получать данные с камеры.

2.2 PiCamera со шлейфом

При выборе оборудования для проекта выбор стоял перед использованием стандартной PiCamera, которая обычно используется с Raspberry Pi, и USB-камерой. Главным минусом PiCamera является то, что она подключается по шлейфу, что видно на рисунке 6, а на каждой плате есть лишь один разъём для шлейфа, поэтому может быть использована лишь одна камера.

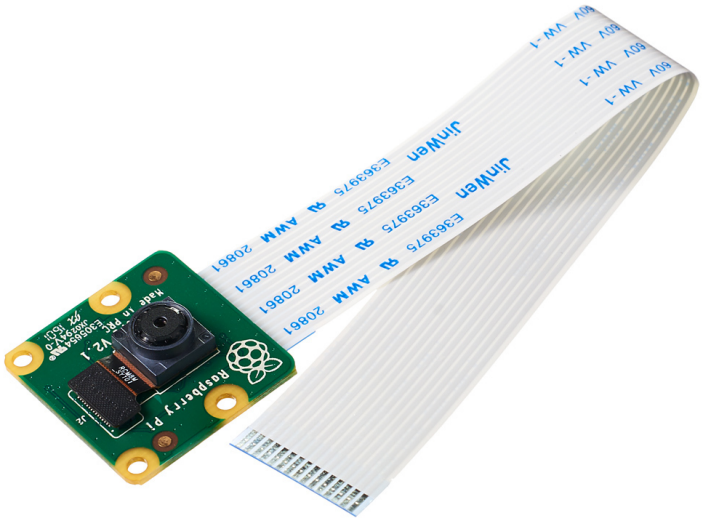

Рис. 6: PiCamera

Как уже упоминалось ранее, в нашем проекте должно предусматриваться использование сразу нескольких датчиков, поэтому выбор был сделан в пользу USB-камеры. Главной сложностью в использовании USB-камеры было то, что большинство фрагментов кода, которые можно найти в интернете написаны для PiCamera, поэтому приходилось большую часть кода переписывать под свою камеру, либо вовсе писать код с нуля.

Так простой захват изображения с камеры для формирования dataset'а был просто скопирован с интернета, а написание своего сервера заняло большую часть времени, так как необходимо было реализовать синхронную отправку изображений клиентам и прекращение отправки уже закрытым соединениям.

Заключение

В ходе работы над проектом были достигнуты следующие задачи:

1. Получено решение для получения данных с камеры;
2. Разработан способ представления оператору данных с камеры.

В рамках практики были решены следующие задачи:

1. Были представлены актуальные результаты по работе с камерой;
2. Было проведено сравнение с актуальными решениями, применяемыми при работе с камерой, а также было объяснено, почему в рамках нашего проекта эти решения не использовались.

Список литературы

1. Робототехника, Wiki.

2. *Raspberry Pi Camera Module 3 Standard NoIR Wide NoIR Wide*. 2023.

3. RTSP, Wiki.