

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский
университет имени академика С.П. Королева»
(Самарский университет)

Институт _____ информатики и кибернетики
Кафедра _____ программных систем

ОТЧЕТ ПО ПРАКТИКЕ

Вид практики _____ производственная
(учебная, производственная)

Тип практики _____ технологическая (проектно-технологическая) практика
(в соответствии с ОПОП ВО)

Сроки прохождения практики: с 01.07.2024 г. по 13.07.2024 г.
(в соответствии с календарным учебным графиком)

по направлению подготовки 02.03.02

«Фундаментальная информатика и информационные технологии»
(уровень бакалавриата)

направленность (профиль) «Информационные технологии»

Обучающийся группы № 6301-020302D _____ Д.О. Колбанов

Руководитель практики
от университета, доцент кафедры
программных систем, к.т.н., доцент _____ Л.С. Зеленко

Руководитель практики
от ООО «Открытый код»,
руководитель Департамента
инжиниринга электронных компонентов
искусственного интеллекта _____ Ю.Е. Резников

Дата сдачи 13.07.2024 г.

Дата защиты 13.07.2024 г.

Оценка _____ *отлично*

Самара 2024

СОДЕРЖАНИЕ

Задания по практике для выполнения определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований)	3
Введение	7
1.1 Процесс разработки моделей компьютерного зрения	10
1.2 Выбор готовой нейросети	11
2 Разработка программы распознавания для цеха	13
2.1 Задачи программы	13
2.2 Реализация программы	13
3 Выводы	17
Список использованных источников	18
Приложение А Листинг приложения	19

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский
университет имени академика С.П. Королева»
(Самарский университет)

Институт _____ информатики и кибернетики
Кафедра _____ программных систем

Задания по практике для выполнения определенных видов работ, связанных с
будущей профессиональной деятельностью
(сбор и анализ данных и материалов, проведение исследований)

Обучающемуся Колбанову Дмитрию Олеговичу группы 6301-020302D

Направлен на практику приказом по университету от 28.06.2024 г. № 347
в _____ ООО «Открытый код» _____,
(наименование профильной организации или структурного подразделения университета)

в соответствии с договором о направлении на практику от 21.06.2021 г.
№ 50-д.

Планируемые результаты освоения образовательной программы (компетенции/индикаторы)	Выполнение определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований)	Результаты практики
ПК-4 Способен решать задачи профессиональной деятельности в составе научно-исследовательского и производственного коллектива. ПК-4.1. Знает принципы построения научной работы, методы сбора и анализа полученного материала. ПК-4.2. Умеет решать научные задачи в связи с поставленной целью и в соответствии с выбранной	Изучить применяемые в организации подходы к разработке программных проектов. Изучить виды нейронных сетей, их классификацию, проведение сравнительного анализа и выбор наиболее подходящей сети для решения поставленной	Обучающийся изучил применяемые в организации подходы к разработке программных проектов. Обучающийся изучил виды нейронных сетей, их классификации, провел


методикой. ПК-4.3. Имеет практический опыт выступлений и научной аргументации при анализе объекта научной и профессиональной деятельности.	задачи.	сравнительный анализ и выбрал наиболее подходящую сеть для решения поставленной
<p>ПК-5 Способен критически переосмысливать накопленный опыт, изменять при необходимости вид и характер своей профессиональной деятельности.</p> <p>ПК-5.1. Знает подходы к критическому переосмыслению накопленного опыта, способен изменять при необходимости вид и характер своей профессиональной деятельности.</p> <p>ПК-5.2. Умеет критически переосмысливать накопленный опыт, изменять при необходимости вид и характер своей профессиональной деятельности.</p> <p>ПК-5.3. Склонен критически анализировать накопленный опыт использования современных инструментальных и вычислительных средств и при необходимости изменять парадигму достижения поставленной цели.</p>	<p>Изучить основы языка программирования Python, работы со списками и массивам.</p> <p>Изучить работы библиотеки OpenCV.</p> <p>Изучить основные алгоритмы компьютерного зрения.</p>	<p>Обучающийся изучил основы языка программирования Python, работы со списками и массивам.</p> <p>Обучающийся изучил работу библиотеки OpenCV.</p> <p>Обучающийся изучил основные алгоритмы компьютерного зрения.</p>

<p>ПК-6 Способен эффективно применять базовые математические знания и информационные технологии при решении проектно-технических и прикладных задач, связанных с развитием и использованием информационных технологий.</p> <p>ПК-6.1. Знает современные языки программирования и методы параллельной обработки данных. Знаком с содержанием Единого Реестра Российских программ для электронных вычислительных машин и баз данных.</p> <p>ПК-6.2. Умеет реализовывать численные методы решения прикладных задач в профессиональной сфере деятельности, пакеты программного обеспечения, операционные системы, электронные библиотеки, сетевые технологии.</p> <p>ПК-6.3. Имеет практический опыт разработки и интеграции информационных систем.</p>	<p>Реализовать алгоритм распознавание сотрудников на изображении.</p> <p>Реализовать поиск групп людей на изображении.</p> <p>Реализовать определение сотрудников в каске и без каски, с указанием их количества.</p>	<p>Обучающийся реализовал алгоритм распознавания сотрудников на изображении.</p> <p>Обучающийся реализовал поиск групп людей на изображение.</p> <p>Обучающийся реализовал определение сотрудников в каске и без каски, с указанием их количества.</p>
<p>ПК-7 Способен разрабатывать и реализовывать процессы жизненного цикла информационных систем, программного обеспечения, сервисов систем информационных технологий, а также методы и механизмы</p>	<p>Разработать программу, выполняющую задачи компьютерного зрения.</p> <p>Оценить функциональные возможности и работу</p>	<p>Обучающийся разработал программу, выполняющую задачи компьютерного зрения.</p> <p>Обучающийся оценил</p>


<p>оценки и анализа функционирования средств и систем информационных технологий.</p> <p>ПК-7.1 Знает основы разработки и реализации процессов жизненного цикла программного обеспечения.</p> <p>ПК-7.2 Знает основы, а также методы и механизмы оценки и анализа функционирования средств и систем информационных технологий.</p> <p>ПК-7.3. Умеет оценивать качество, надежность и эффективность информационной системы.</p>	<p>программы на различных тестовых примерах.</p> <p>Устранить выявленные недочеты программы.</p>	<p>функциональные возможности и работу программы на различных тестовых примерах.</p> <p>Обучающийся устранил выявленные недочеты программы.</p>
---	--	---

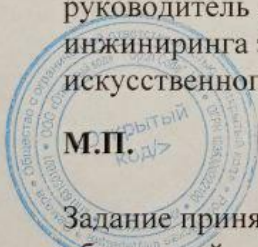
Срок представления на кафедру отчета о практике 13.07.2024 г.

Руководитель практики от университета,
доцент кафедры программных
систем, к.т.н., доцент



(подпись) Л.С. Зеленко

Руководитель практики
от ООО «Открытый код»,
руководитель Департамента
инжиниринга электронных компонентов
искусственного интеллекта


(подпись) Ю.Е. Резников



Задание принял к исполнению
обучающийся группы № 6301-020302D


(подпись) Д.О. Колбанов

ВВЕДЕНИЕ

«Открытый код» входит в консорциум Национального центра когнитивных разработок – отечественной экосистемы развития технологий искусственного интеллекта в рамках парадигмы цифровой экономики. Консорциум определяет стратегические направления развития интеллектуальных когнитивных технологий и технологий машинного обучения в Российской Федерации. Компания также является членом технологического комитета "Искусственный интеллект" [1].

В основе всех разработок компании лежит собственная программная платформа, в состав которой входят:

- базы знаний;
- сервисы текстового распознавания и семантического поиска;
- инструменты построения маршрутов и решения логистических задач;
- системы принятия управленческих решений и мониторинга;
- инструменты обработки Big Data;
- компьютерное зрение и AR/VR-технологии;
- нейросетевые технологии, глубокое машинное обучение;
- инструменты BI с акцентной визуализацией;

Во время прохождения производственной практики необходимо решить следующие задачи:

- изучить применяемые в организации подходы к разработке программных проектов;
- изучить виды нейронных сетей, их классификацию, проведение сравнительного анализа и выбор наиболее подходящей сети для решения поставленной задачи;
- изучить основы языка программирования Python, работы со списками и массивам;
- изучить работы библиотеки OpenCV;
- изучить основные алгоритмы компьютерного зрения;

- реализовать алгоритм распознавание сотрудников на изображении;
- реализовать поиск групп людей на изображении;
- реализовать определение сотрудников в каске и без каски, с указанием их количества;
- разработать программу, выполняющую задачи компьютерного зрения;
- оценить функциональные возможности и работу программы на различных тестовых примерах;
- устранить выявленные недочеты программы.

1 Программы компьютерного зрения

Компьютерное зрение – это область компьютерных наук, которая стремится расширить возможности компьютеров по идентификации и определению объектов и людей на изображениях и в видео. Как и другие типы искусственного интеллекта (ИИ), компьютерное зрение ориентируется на выполнение и автоматизацию задач, имитирующих человеческие возможности. В этом случае компьютерное зрение старается имитировать зрение и восприятие человека [2].

Спектр практического применения технологий компьютерного зрения обуславливает его развитие как центрального компонента множества современных инноваций и решений. Рабочие нагрузки компьютерного зрения можно выполнять в облаке или локально.

Приложения компьютерного зрения используют входные данные с сенсорных устройств, возможности ИИ, машинного обучения и глубокого обучения для имитации того, как работает человеческое зрение. Такие приложения работают на основе алгоритмов, обученных на огромных объемах визуальных данных или изображений в облаке. Они распознают образы в визуальных данных и с их помощью определяют содержимое, присутствующее на изображениях.

Анализ изображения с помощью технологий компьютерного зрения:

- сенсорное устройство создает изображение. Сенсорное устройство часто является обычной фотокамерой, но может быть видеокамерой, устройством медицинской визуализации или любым другим устройством, создающим изображение для анализа;

- изображение затем отправляется на устройство для анализа. Устройство для анализа использует возможности распознавания изображений для разбиения изображения на отдельные части, сравнения найденных образов с библиотекой известных образов и их сопоставления. Образом может быть что-то общее, например внешний вид объекта

определенного типа, или же образ может быть основан на уникальных признаках, таких как черты лица;

– пользователь запрашивает определенную информацию об изображении, а анализирующее устройство предоставляет такие сведения по результатам анализа изображения.

Современные приложения компьютерного зрения все чаще используют для анализа изображений возможности глубокого обучения, а не статистические методы, как раньше. Глубокое обучение позволяет такому приложению выполнять определенный алгоритм, который называется нейронной сетью, позволяющий более точно выполнять анализ. Кроме того, при использовании глубокого обучения программа компьютерного зрения может сохранять информацию о каждом проанализированном изображении и с течением времени еще больше повышать точность.

1.1 Процесс разработки моделей компьютерного зрения

Процесс разработки моделей компьютерного зрения включает в себя несколько этапов [3]:

1) постановка задачи. На этом этапе необходимо определить, какую проблему должна решить модель компьютерного зрения. Это может быть задача классификации изображений, обнаружения объектов или сегментации изображений;

2) сбор данных. Для обучения модели необходимо собрать большой массив данных, на которых модель будет тренироваться и тестироваться. Данные могут быть получены из различных источников, таких как открытые базы данных или собственные данные компании;

3) предварительная обработка данных. Собранные данные необходимо предварительно обработать, чтобы они были пригодны для использования моделью. Предварительная обработка включает в себя изменение размера изображений, нормализация яркости и контрастности, удаление шума;

4) выбор архитектуры модели. На основе поставленной задачи и доступных данных необходимо выбрать архитектуру модели, которая будет наиболее эффективной. Для задач компьютерного зрения часто используют сверточные и рекуррентные нейронные сети;

5) обучение модели. После выбора архитектуры необходимо обучить модель на собранных данных. Обучение включает в себя настройку параметров модели таким образом, чтобы она могла выполнять поставленную задачу, допуская наименьшее количество ошибок;

6) оценка модели. После обучения необходимо провалидировать эффективность модели на тестовом наборе данных, который не использовался при обучении;

7) оптимизация модели. Если результаты оценки показывают, что модель работает недостаточно хорошо, необходимо оптимизировать её, изменив архитектуру, параметры или методы предварительной обработки данных;

8) развертывание модели. Когда модель достигает требуемой точности, её можно развернуть в производственной среде для реального использования. Развёртывание может включать в себя интеграцию модели с другими системами, такими как системы видеонаблюдения или системы распознавания лиц;

Этот процесс является итеративным, и разработчики могут возвращаться к предыдущим этапам для улучшения модели.

1.2 Выбор готовой нейросети

Выбор готовой нейросети для компьютерного зрения – это важный шаг, который может значительно повлиять на успех проекта. Они уже обучены на огромных объемах данных, что позволяет получить точные результаты, не тратя время на собственное обучение с нуля.

Рассмотрим наиболее популярные решения:

– YOLO (You Only Look Once) – это одна из наиболее известных моделей для обнаружения объектов на изображениях в реальном времени. Она основана на сверточных нейронных сетях и позволяет достичь высокой скорости обработки без ущерба точности. YOLO разделяет изображение на сетку ячеек, и каждая ячейка предсказывает границы и классы объектов, содержащихся внутри нее [4];

– ResNet (Residual Neural Network) – это глубокая нейронная сеть, разработанная для решения проблемы затухания градиента. Она использует концепцию «skip connections» или «residual connections», позволяющих передавать информацию непосредственно от одного слоя к другому, минуя промежуточные слои. Это позволяет обучать более глубокие сети с лучшей производительностью [4];

– Mask R-CNN – это модель, которая сочетает в себе возможности обнаружения объектов и сегментации изображений. Она была представлена в статье «Mask R-CNN» в 2017 году и быстро завоевала популярность благодаря своей точности и универсальности. Mask R-CNN может использоваться для различных задач, таких как распознавание лиц или анализ медицинских изображений [5].

2 Разработка программы распознавания для цеха

На рисунке 1 приведено изображение с камеры видеонаблюдения, установленной в цеху. Необходимо реализовать программу поиска человека на изображении, используя библиотеку OpenCV [6] и готовую нейросеть YOLO [7].



Рисунок 1 – Изображение с камеры в цеху

2.1 Задачи программы

В ходе работы программа должна выполнять следующие задачи:

- распознать всех людей на изображении, отобразить их описывающим прямоугольником. Посчитать их количество, отобразить на изображении полученное значение;
- определить наличие группы людей и отобразить её описывающим прямоугольником со значением количества людей в группе;
- определить сотрудников в каске и без каски, отобразить их на изображении с указанием их количества.

2.2 Реализация программы

Чтобы найти людей на изображении, используется модель машинного обучения YOLOv8x. После обработки детекции, используя библиотеку

Чтобы найти группы людей на изображении, используется информация о найденных людях, полученная на предыдущем шаге. Это позволяет избежать повторной обработки изображения.

Для каждого человека определяется его центр и проверяется, находится ли он рядом с другими центрами (ближе, чем 100 пикселей). Если да, то эти люди считаются частью одной группы. Этот процесс повторяется для всех людей в группе, пока не будут обработаны все люди. В итоге получается список групп людей, а также координаты их границ, что позволяет нарисовать прямоугольники, охватывающие все людей в каждой группе. Результат обнаружения групп сотрудников приведен на рисунке 3.

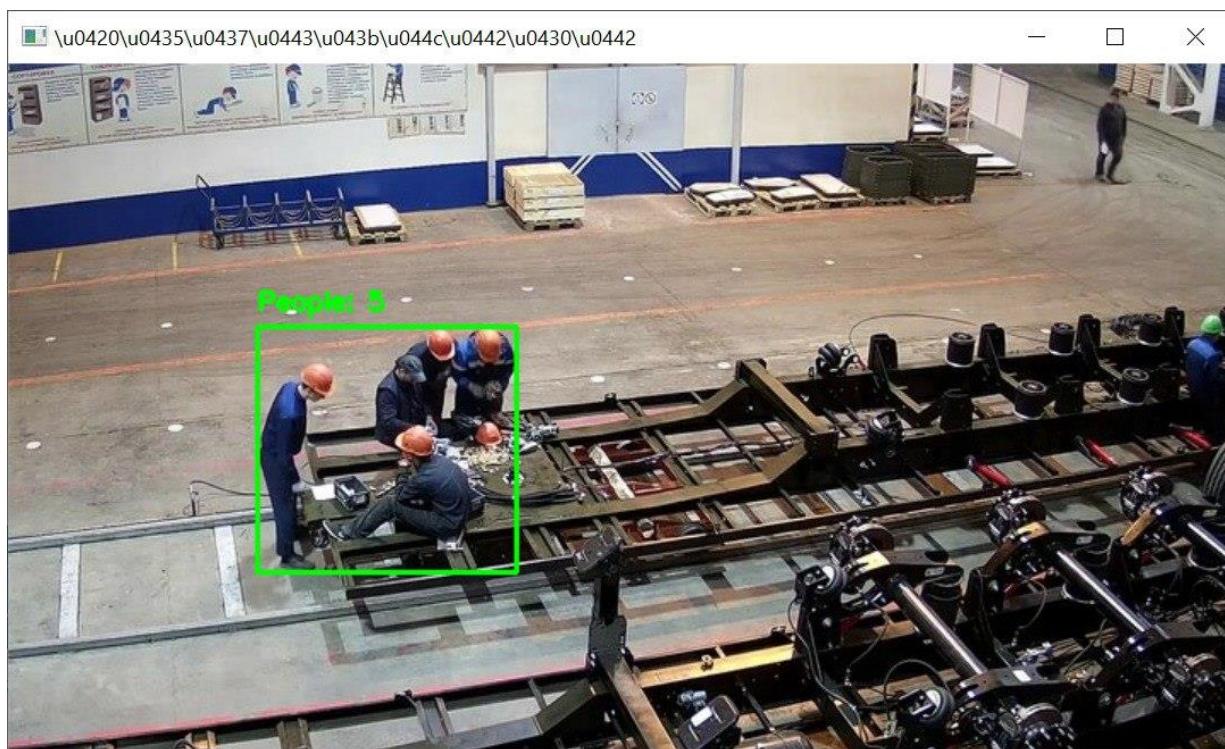


Рисунок 3 – Результат обнаружения групп сотрудников

Для нахождения сотрудников без касок анализируется изображение, на котором уже были обнаружены люди. Для каждого человека, обнаруженного на предыдущем этапе, алгоритм выделяет область головы и анализирует её цвет.

Используются два диапазона цветов, соответствующих оранжевым и зеленым каскам. Если в области головы обнаружено достаточно большое пятно оранжевого или зеленого цвета, то алгоритм считает, что человек носит каску. В противном случае, алгоритм считает, что человек не носит каску.

Затем алгоритм подсчитывает количество сотрудников с касками и без касок, а также визуально выделяет их на изображении: людей с касками – зеленым прямоугольником, а людей без касок – красным прямоугольником.

Результат обнаружения сотрудников без касок приведен на рисунке 4.

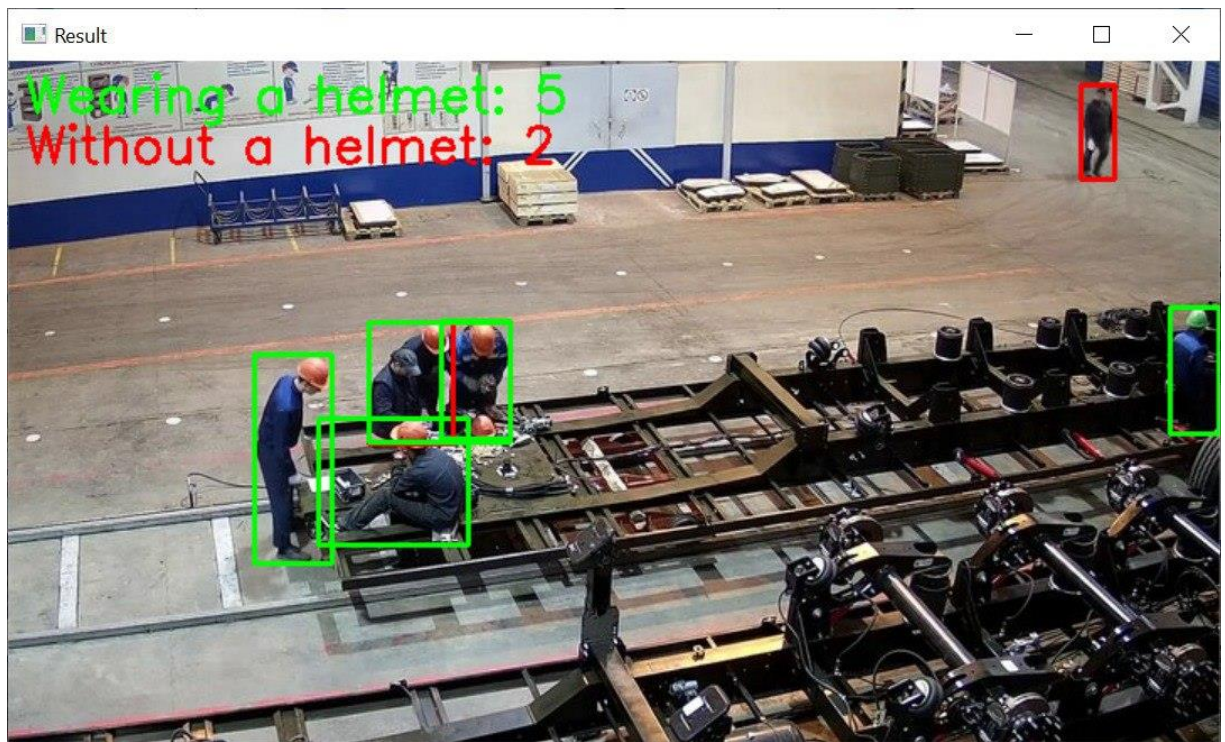


Рисунок 4 – Результат обнаружения сотрудников без касок

3 Выводы

В результате выполнения производственной практики:

- изучены применяемые в организации подходы к разработке программных проектов;
- изучены виды нейронных сетей, их классификация, проведен сравнительный анализ и выбрана наиболее подходящая сеть для решения поставленной задачи;
- изучены основы языка программирования Python, работы со списками и массивам;
- изучена работа библиотеки OpenCV;
- изучены основные алгоритмы компьютерного зрения;
- реализован алгоритм распознавания сотрудников на изображении;
- реализован поиск групп людей на изображении;
- реализовано определение сотрудников в каске и без каски, с указанием их количества;
- разработана программа, выполняющая задачи компьютерного зрения;
- оценены функциональные возможности и работа программы на различных тестовых примерах;
- устранены выявленные недочеты программы.

Таким образом, все поставленные задачи выполнены в полном объеме и освоены все необходимые все необходимые индикаторы (ПК-4.1, ПК-4.2, ПК-4.3, ПК-5.1, ПК-5.2, ПК-5.3, ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1, ПК-7.2, ПК-7.3) компетенций (ПК-4, ПК-5, ПК-6, ПК-7).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 Открытый код [Электронный ресурс]. URL: <https://www.o-code.ru/about> (дата обращения: 06.07.2024).
- 2 Что собой представляет компьютерное зрение [Электронный ресурс]. URL: <https://azure.microsoft.com/ru-ru/resources/cloud-computing-dictionary/what-is-computer-vision> (дата обращения: 11.07.2024).
- 3 Компьютерное зрение [Электронный ресурс]. URL: <https://exponenta.ru/comp-vision> (дата обращения: 11.07.2024).
- 4 Практическое применение моделей YOLO и ResNet для обнаружения нежелательных предметов на фотографиях [Электронный ресурс]. URL: <https://habr.com/ru/articles/761200/> (дата обращения: 11.07.2024).
- 5 Mask R-CNN: архитектура современной нейронной сети для сегментации объектов на изображениях [Электронный ресурс]. URL: <https://habr.com/ru/articles/421299/> (дата обращения: 11.07.2024).
- 6 Документация YOLO [Электронный ресурс]. URL: <https://docs.ultralytics.com/ru> (дата обращения: 11.07.2024).
- 7 Документация OpenCV [Электронный ресурс]. URL: <https://docs.opencv.org/4.x/> (дата обращения: 11.07.2024).

ПРИЛОЖЕНИЕ А

Листинг приложения

```
from ultralytics import YOLO
import numpy as np
import cv2
# Загружаем модель YOLOv8x
model = YOLO("yolov8x.pt")

# Загружаем изображение
original_image = cv2.imread("test_img.jpg")

# Выполняем детекцию
results = model.predict(original_image, classes=[0], conf=0.2, iou=0.7)
# classes=[0] - распознавание объектов с классом person
image = original_image.copy()
people = []

for detection in results[0].boxes.xyxy:
    x1, y1, x2, y2 = detection.cpu().numpy().astype(int)
    center_x = (x1 + x2) // 2
    center_y = (y1 + y2) // 2
    people.append([x1, y1, x2, y2, center_x, center_y])
    cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)

2) cv2.putText(image, f'People: {len(results[0])}', (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0),

cv2.imshow("Результат", image)
cv2.waitKey(0)

groups = []

for person in people:

    # Проверяем, есть ли группа, в которую можно добавить человека
    found_group = False
    for group in groups:
        for person_in_group in group:
            # Расстояние между центрами
            distance = np.sqrt((person[4] - person_in_group[4])**2 + (person[5] - person_in_group[5])**2)
            # Если расстояние меньше 100 пикселей, добавляем человека к группе
            if distance < 100:
                group.append(person)
                found_group = True
                break
        if found_group:
            break

    # Если группа не найдена, создаем новую группу
    if not found_group:
        groups.append([person])
image = original_image.copy()

for group in groups:
    if len(group) > 1:
        x1, y1 = 10**10, 10**10
        x2, y2 = 0, 0
        for person in group:
            cur_x1, cur_y1, cur_x2, cur_y2 = person[:4]
            if cur_x1 < x1:
                x1 = cur_x1
```

```

        if cur_y1 < y1:
            y1 = cur_y1
        if cur_x2 > x2:
            x2 = cur_x2
        if cur_y2 > y2:
            y2 = cur_y2
    cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)
    cv2.putText(image, f'People: {len(group)}', (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
(0,255, 0), 2)
    cv2.imshow("Результат", image)
    cv2.waitKey(0)

image = original_image.copy()

helmet_count = 0
no_helmet_count = 0
# Цветовые модели HSV
hsv_orange_lower = np.array([10, 100, 100], dtype="uint8")
hsv_orange_upper = np.array([25, 255, 255], dtype="uint8")

hsv_green_lower = np.array([40, 50, 50], dtype="uint8")
hsv_green_upper = np.array([80, 255, 255], dtype="uint8")

# Проходим по каждому человеку
for person in people:
    x1, y1, x2, y2 = person[:4]
    # Вырезаем область вокруг человека
    person_roi = image[y1:int(y1 + (y2 - y1) / 3), x1:x2]

    # Преобразуем изображение в HSV
    hsv = cv2.cvtColor(person_roi, cv2.COLOR_BGR2HSV)

    # Создаем маски для оранжевого и зеленого цвета
    mask_orange = cv2.inRange(hsv, hsv_orange_lower, hsv_orange_upper)
    mask_green = cv2.inRange(hsv, hsv_green_lower, hsv_green_upper)

    # Проверяем наличие оранжевого или зеленого пятна
    has_helmet = np.sum(mask_orange) > 1000 or np.sum(mask_green) > 1000

    if has_helmet:
        helmet_count += 1
        cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)
    else:
        no_helmet_count += 1
        cv2.rectangle(image, (x1, y1), (x2, y2), (0, 0, 255), 2)

    cv2.putText(image, f'Wearing a helmet: {helmet_count}', (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1,
(0,255, 0), 2)
    cv2.putText(image, f'Without a helmet: {no_helmet_count}', (10, 60), cv2.FONT_HERSHEY_SIMPLEX,
1, (0, 0, 255), 2)
    cv2.imshow('Result', image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

```