|  |  |
| --- | --- |
|  |  |
|  |  |

*СОДЕРЖАНИЕ*

[***АННОТАЦИЯ*** *4*](#_Toc134649479)

[***ВВЕДЕНИЕ*** *5*](#_Toc134649480)

[***1.1.******Наименование программы*** *5*](#_Toc134649481)

[***1.2.******Документ, на основании которого ведется разработка*** *5*](#_Toc134649482)

[***2.******НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ*** *6*](#_Toc134649483)

[***2.1.******Назначение программы*** *6*](#_Toc134649484)

[**2.1.1.** **Функциональное назначение** 6](#_Toc134649485)

[**2.1.2.** **Эксплуатационное назначение** 6](#_Toc134649486)

[***2.2.******Краткая характеристика области применения*** *6*](#_Toc134649487)

[***3.******ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ*** *7*](#_Toc134649488)

[***3.1.******Постановка задачи на разработку программы*** *7*](#_Toc134649489)

[***3.2.******Описание и обоснование архитектуры программы*** *9*](#_Toc134649490)

[**3.2.1.** **Описание архитектуры программы** 9](#_Toc134649491)

[**3.2.2.** **Обоснование архитектуры программы** 10](#_Toc134649492)

[***3.3.******Описание и обоснование алгоритма работы программы*** *10*](#_Toc134649493)

[**3.3.1.** **Описание алгоритма работы программы** 10](#_Toc134649494)

[**3.3.2.** **Обоснование алгоритма работы программы** 11](#_Toc134649495)

[***3.4.******Описание и обоснование выбора метода организации входных и выходных данных*** *11*](#_Toc134649496)

[***3.5.******Описание и обоснование выбора состава технических и программных средств*** *12*](#_Toc134649497)

[**3.5.1.** **Состав технических и программных средств** 12](#_Toc134649498)

[**3.5.2.** **Обоснование выбора технических и программных средств** 13](#_Toc134649499)

[***4.******ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ*** *14*](#_Toc134649500)

[***4.1.******Ориентировочная экономическая эффективность*** *14*](#_Toc134649501)

[***4.2.******Предполагаемая потребность*** *14*](#_Toc134649502)

[***4.3.******Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами*** *14*](#_Toc134649503)

[***5.******ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ*** *16*](#_Toc134649504)

[***ПРИЛОЖЕНИЕ 1*** *17*](#_Toc134649505)

[***ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ*** *18*](#_Toc134649506)

# **АННОТАЦИЯ**

Данная программная документация представляет собой пояснительную записку к программному проекту «Приложение для контроля за оборотом инструментов “КОИ”», предназначенного для упрощения процесса отслеживания и контроля инструментов на стенде, отправления на почту актуальной информации, просмотра истории использования того или иного инструмента.

Раздел «Введение» включает в себя наименование программы и документ, на основании которого ведётся разработка, с указанием утвердившей данный документ организации.

В разделе «Назначение и область применения» содержатся функциональное и эксплуатационное назначения программы и ее краткая характеристика области применения.

В разделе «Технические характеристики» содержатся следующие подразделы: постановка задачи на разработку программы, описание функционирования программы, описание и обоснование алгоритма работы программы, описание и обоснование выбора метода организации входных и выходных данных, описание работы с базой данных, описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указываются предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными приложениями с такой же функциональностью.

Программный документ разработан в соответствии с требованиями:

1. ГОСТ 19.101-77 Виды программ и программных документов[1];
2. ГОСТ 19.102-77 Стадии разработки[2];
3. ГОСТ 19.103-77 Обозначения программ и программных документов[3];
4. ГОСТ 19.104-78 Основные надписи[4];
5. ГОСТ 19.105-78 Общие требования к программным документам[5];
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом[6];
7. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению[7].

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603-78[8], ГОСТ 19.604-78[9].

**ВВЕДЕНИЕ**

* 1. **Наименование программы**

Наименование программы – «Приложение для контроля за оборотом инструментов “КОИ”».

Наименование программы на английском языке – «Application for controlling the turnover of instruments “CTI”».

* 1. **Документ, на основании которого ведется разработка**

Основанием для разработки стал учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденная академическим руководителем тема курсового проекта.

Документ был утвержден организацией: Национальный исследовательский университет «Высшая школа экономики», Факультет компьютерных наук, образовательная программа «Программная инженерия».

1. **НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ**
   1. **Назначение программы**
      1. **Функциональное назначение**

Под функциональным назначением программы понимаются отслеживание и контроль инструментов на стенде, отправление на почту актуальной информации, просмотр истории использования того или иного инструмента.

* + 1. **Эксплуатационное назначение**

“КОИ” будет применяться в сфере промышленности. Она предоставит возможность хранить информацию о наличии или отсутствии инструментов с помощью камеры, отслеживать сведения о текущем пользователе инструмента и формировать отчеты о времени взятия и имени текущего работника с этим инструментом, а также смогут получать напоминание о возврате инструментов на место.

Это приложение облегчает дальнейшее взаимодействие между работниками, поскольку им необходимо знать о наличии или отсутствии определенных инструментов на предприятиях.

* 1. **Краткая характеристика области применения**

Целевой аудиторией проекта являются различные предприятия, пользующиеся инструментами в своей работе и нуждающиеся в отслеживании их оборота. Основная цель разрабатываемой программы – автоматизация процессов, связанных с контролем оборота инструментов на стенде, и упрощение процесса отслеживания инструментов на различных предприятиях России, которые пользуются строительными инструментами ежедневно.

1. **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ**
   1. **Постановка задачи на разработку программы**

Разрабатываемая программа должна выполнять следующие поставленные задачи:

1. Распознавать инструменты (ножницы, отвертка) и выделять их на видео.

Выделение см. на скриншотах ниже:

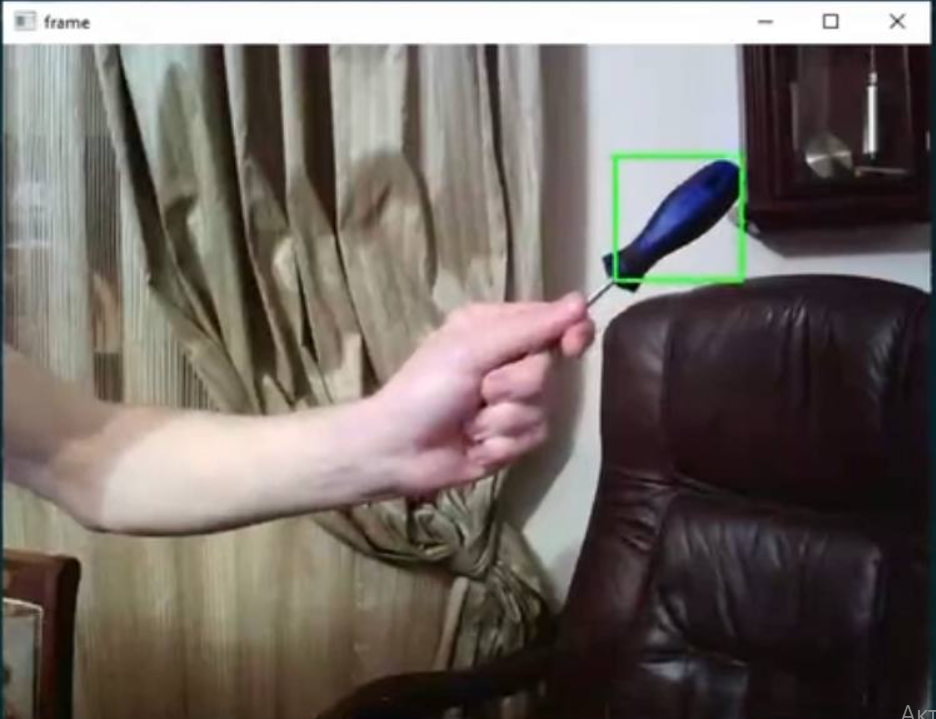


Рисунок 1. Выделение отвертки

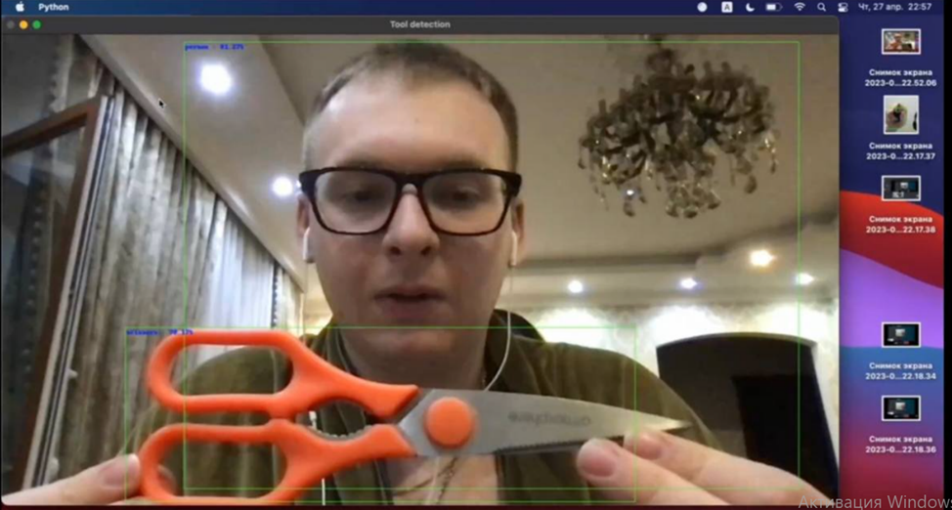


Рисунок 2. Выделение ножниц

1. Распознавать человека на видео и выделяет.

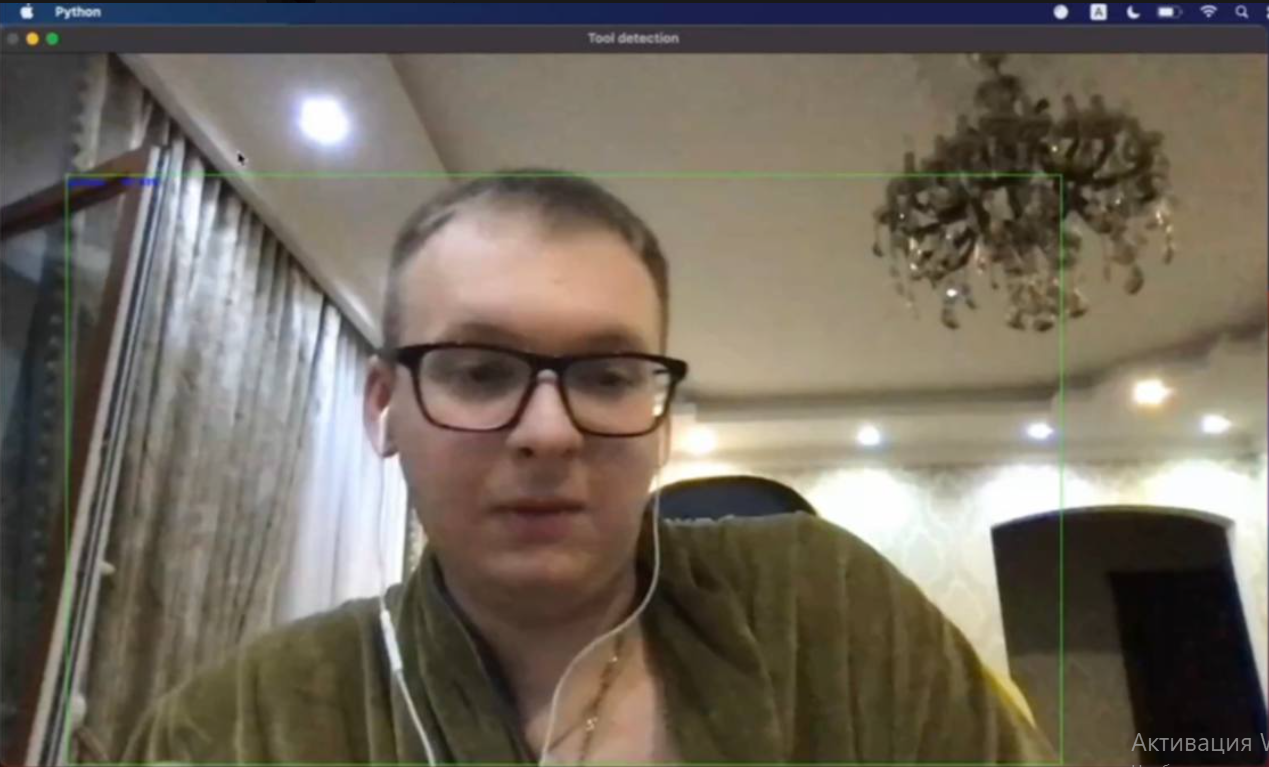


Рисунок 3. Персона распознана.

1. В качестве выполнения доп. функционала предоставлялась отправка на почту отчета о взятии инструмента.
2. Авторизация пользователя по логину и паролю.

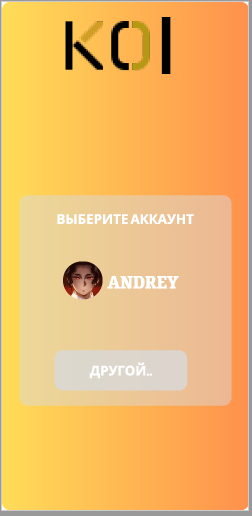
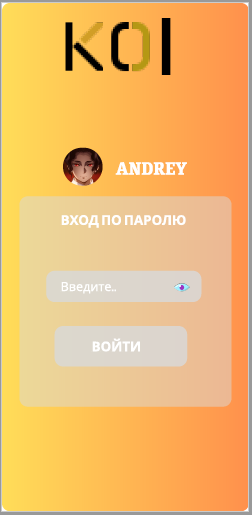
 

Рисунок 4. Авторизация пользователя по логину и паролю.

1. Просмотр наличия инструмента

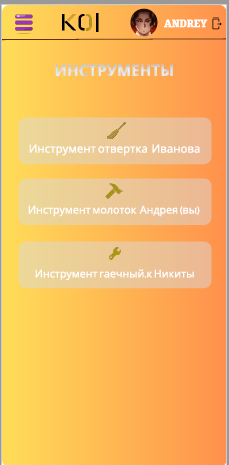


Рисунок 5. Интерфейс «Инструментов в наличии»

1. Просмотр отчета об инструментах

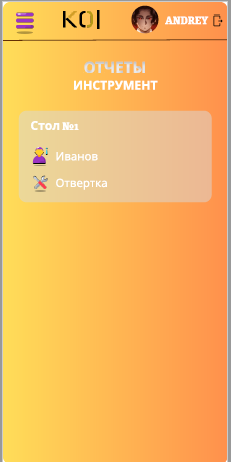


Рисунок 6. Интерфейс «Отчета об инструментах»

* 1. **Описание и обоснование архитектуры программы**
     1. **Описание архитектуры программы**

Программа использует принцип Наблюдателя, который распознает инструменты и человека перед камерой и выделяет его (реагирует на изменение).

Программа состоит из двух модулей: “KOI V2” и “KOI ActiveBoys”.

Каждый из модулей отслеживает свою область ответственности.

“KOI V2” – отслеживает и распознает отвертку на видео, отправляет уведомление о возврате инструмента на стенд.

“KOI ActiveBoys” – отслеживает и распознает ножницы и человека на видео.

Данные о распознаваемых объектов хранятся в XML файлах и подтягиваются в сами программы, во время работы приложения.

Архитектура приложения представлена на рисунке ниже.

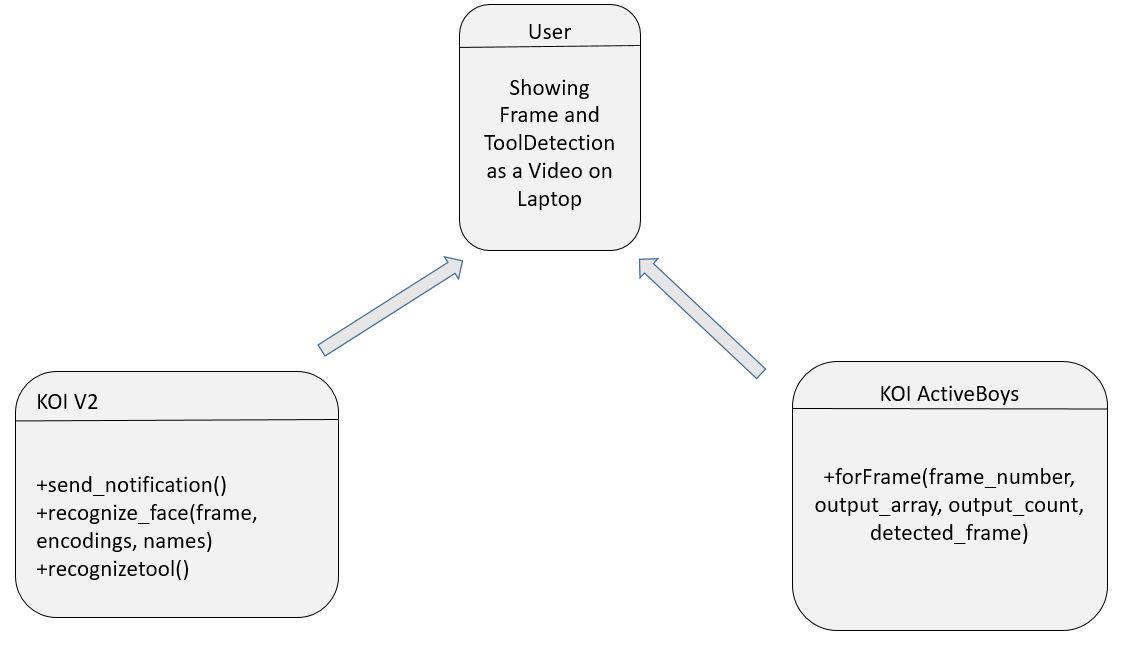


Рисунок 7. Архитектура приложения «Наблюдатель»

* + 1. **Обоснование архитектуры программы**

Принцип Наблюдателя был выбран в связи с его удобством и практичностью для работы данного приложения. Данные распознаваемых объектов подтягиваются из XML файлов и задействуются в программном коде. Благодаря чему весь программный код работает и выполняет поставленные задачи.

Так же данная архитектура позволяет удобно отображать результаты работы программы в отдельном окне, что помогает в распознании объектов (Ножницы, Отвертка, Человек)

* 1. **Описание и обоснование алгоритма работы программы**
     1. **Описание алгоритма работы программы**

После сборки проекта и начала работы программного кода появляется отдельное окно, которое отображает видео с камеры в режиме реального времени. Все данные о нужных распознаваемых объектах подтягиваются из ранее обученных моделей, содержащихся в XML файлах. С помощью функции cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY мы преобразуем изображении с камеры в чёрно-белый формат и выделяем контур объекта на изображении и таким образом определяем наличие объекта. При наличии инструмента или человека в кадре, он выделяется определенным зеленым прямоугольником и подписывается. При отсутствии объекта на видео программа транслирует изображение с камеры без выделения иных объектов.

В качестве показа алгоритма работы будет приведен код ниже из двух файлов программного кода на языке Python:

koi\_v2

import cv2

import numpy as np

import datetime

import smtplib

import imghdr

from email.message import EmailMessage

import face\_recognition

import time

# Загрузка классификатора Haar Cascade для распознавания инструмента

tool\_cascade = cv2.CascadeClassifier('cascade.xml')

# Функция для отправки отчета на почту

def send\_email(image\_path):

sender\_email = "sender@example.com"

receiver\_email = "receiver@example.com"

password = "password"

message = EmailMessage()

message['From'] = sender\_email

message['To'] = receiver\_email

message['Subject'] = 'Отчет о возврате инструмента'

message.set\_content('Отчет о возврате инструмента прилагается')

with open(image\_path, 'rb') as f:

file\_data = f.read()

file\_type = imghdr.what(f.name)

file\_name = f.name

message.add\_attachment(file\_data, maintype='image', subtype=file\_type, filename=file\_name)

with smtplib.SMTP\_SSL('smtp.gmail.com', 465) as smtp:

smtp.login(sender\_email, password)

smtp.send\_message(message)

# Функция для отправки уведомления о возврате инструмента на стенд

def send\_notification():

def send\_email(to, subject, body):

gmail\_user = "your\_email@gmail.com" # замени на свой email

gmail\_password = "your\_password" # замени на свой пароль

try:

server = smtplib.SMTP('smtp.gmail.com', 587)

server.starttls()

server.login(gmail\_user, gmail\_password)

message = 'Subject: {}\n\n{}'.format(subject, body)

server.sendmail(gmail\_user, to, message)

server.quit()

print("Отчет успешно отправлен!")

except:

print("Не удалось отправить отчет по почте.")

# Функция для распознавания лиц

def recognize\_face(frame, encodings, names):

rgb\_frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

face\_locations = face\_recognition.face\_locations(rgb\_frame)

face\_encodings = face\_recognition.face\_encodings(rgb\_frame, face\_locations)

for face\_encoding, face\_location in zip(face\_encodings, face\_locations):

matches = face\_recognition.compare\_faces(encodings, face\_encoding, tolerance=0.5)

name = "Unknown"

face\_distances = face\_recognition.face\_distance(encodings, face\_encoding)

best\_match\_index = np.argmin(face\_distances)

if matches[best\_match\_index]:

name = names[best\_match\_index]

top, right, bottom, left = face\_location

cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)

cv2.putText(frame, name, (left, top - 6), cv2.FONT\_HERSHEY\_SIMPLEX, 0.75, (0, 0, 255), 2)

# Основной код

cap = cv2.VideoCapture(0)

cv2.startWindowThread()

# Инициализация лицевых кодировок и имен для распознавания лиц

encodings = []

names = []

# Загрузка фотографий и кодировок известных лиц

known\_face\_names = ['nbekit1']

for name in known\_face\_names:

img = cv2.imread(f'{name}.jpg')

rgb = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

encoding = face\_recognition.face\_encodings(rgb)[0]

encodings.append(encoding)

names.append(name)

# Загрузка изображения с инструментом

frame = cv2.imread('tool\_image.jpg')

while True:

ret, frame = cap.read()

if not ret:

break

# Обработка изображения для определения наличия инструмента на стенде

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

blurred = cv2.GaussianBlur(gray, (11, 11), 0)

edged = cv2.Canny(blurred, 30, 150)

contours, \_ = cv2.findContours(edged.copy(), cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

# Обнаружение инструмента на изображении

tools = tool\_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5)

# Отображение области с инструментом на изображении

for (x, y, w, h) in tools:

cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

cv2.imshow('frame', frame)

if cv2.waitKey(25) & 0xFF == ord('q'):

break

# Отображение результата

cv2.imshow('Tool detection', frame)

cv2.waitKey(0)

cv2.destroyAllWindows()

#получение текущей даты и времени для использования в именах файлов

now = datetime.time()

date\_time = now.strftime("%Y-%m-%d\_%H-%M-%S")

#цикл по всем контурам

for contour in contours:

# вычисление площади контура

area = cv2.contourArea(contour)

MIN\_AREA = 100 # присваиваем значение 100 переменной MIN\_AREA

# игнорирование слишком маленьких контуров

if area < MIN\_AREA: (x, y, w, h) = cv2.boundingRect(contour)

# выделение области с инструментом на изображении

instrument\_roi = gray[y:y+h, x:x+w]

# распознавание лица в области с инструментом

face = recognize\_face(instrument\_roi)

# сохранение изображения с выделенной областью и, если было распознано лицо, с его обведенным прямоугольником

cv2.imwrite(f'ROI\_{date\_time}\_{x}\_{y}.jpg', instrument\_roi)

if face is not None:

cv2.rectangle(gray, (x, y), (x + w, y + h), (0, 255, 0), 2)

cv2.imwrite(f'ROI\_with\_face\_{date\_time}\_{x}\_{y}.jpg', gray)

# отправка уведомления о возврате инструмента на стенд на указанный email

send\_email('notification@example.com', 'Return of tool', f'Tool with ID {x}\_{y} has been returned to the stand by {face}.')

else:

# отправка уведомления о возврате инструмента на стенд без указания имени пользователя, если лицо не распознано

send\_email('notification@example.com', 'Return of tool', f'Tool with ID {x}\_{y} has been returned to the stand.')

#отображение и сохранение обработанного изображения

cv2.imshow("Processed Image", gray)

cv2.imwrite(f'Processed\_Image\_{date\_time}.jpg', gray)

#ожидание нажатия клавиши для завершения работы

cv2.waitKey(0)

# освобождаем ресурсы

cv2.destroyAllWindows()

KOI ActiveBoys:

from imageai.Detection import VideoObjectDetection

import os

import cv2

execution\_path = os.getcwd()

# делаем захват видео с камеры и показываем в отдельном окне

camera = cv2.VideoCapture(0)

cv2.startWindowThread()

# определяем детектор и показываем с каким каскадом будем производить работу

# загружаем также обучченную модкль, которая и будет помогать в распознавании предметов на изображении

detector = VideoObjectDetection()

detector.setModelTypeAsRetinaNet()

detector.setModelPath(os.path.join(execution\_path , "retinanet\_resnet50\_fpn\_coco-eeacb38b.pth"))

detector.loadModel()

# описание и вывод в консоль результатов

# будем использовать в ф-ции опредления объектов на видеопотоке

def forFrame(frame\_number, output\_array, output\_count, detected\_frame):

print("FOR FRAME " , frame\_number)

print("Output for each object : ", output\_array)

print("Output count for unique objects : ", output\_count)

print("------------END OF A FRAME --------------")

cv2.imshow('Tool detection', detected\_frame)

if cv2.waitKey(25) & 0xFF == ord('q'):

return

# главный код распознавания объектов работы программы нашей

video\_path = detector.detectObjectsFromVideo(camera\_input=camera,

output\_file\_path=os.path.join(execution\_path, "camera\_detected\_video")

, frames\_per\_second=20, log\_progress=True, minimum\_percentage\_probability=40, detection\_timeout=120,

save\_detected\_video=True, return\_detected\_frame=True, per\_frame\_function=forFrame)

Для показа работы программного кода, его алгоритм были так же записаны заранее наглядные видео через средства записи экрана.

Это было сделано на случай непредвиденных обстоятельств на защите у комиссии.

* + 1. **Обоснование алгоритма работы программы**

Данный алгоритм работы был выбран в связи с его целесообразностью и простотой использования и написания кода через библиотеку компьютерного зрения OpenCV, которая и помогла в написании данного программного кода и распознавании объектов с помощью камеры. Выделение контура объекта на видеопотоке помогает ускорить работу программы и распознавание предмета на видеопотоке. Данные, содержащиеся в XML файлах, были получены в результате обучения и тренировки моделей на различных изображениях инструментов и человеческих лиц. В дальнейшем, обученные модели использовались в определении объектов с видеопотока камеры.

Так же с помощью написанной программы можно будет сделать распознавание иных объектов, при тренировке каскадов на иных изображениях предметов (Например, как груша или яблоко).

* 1. **Описание и обоснование выбора метода организации входных и выходных данных**

Данными, получаемыми на вход, у данного программного кода являются:

Трансляция видеопотока с камеры, получаемая при запуске и исполнении (выполнении) программы.

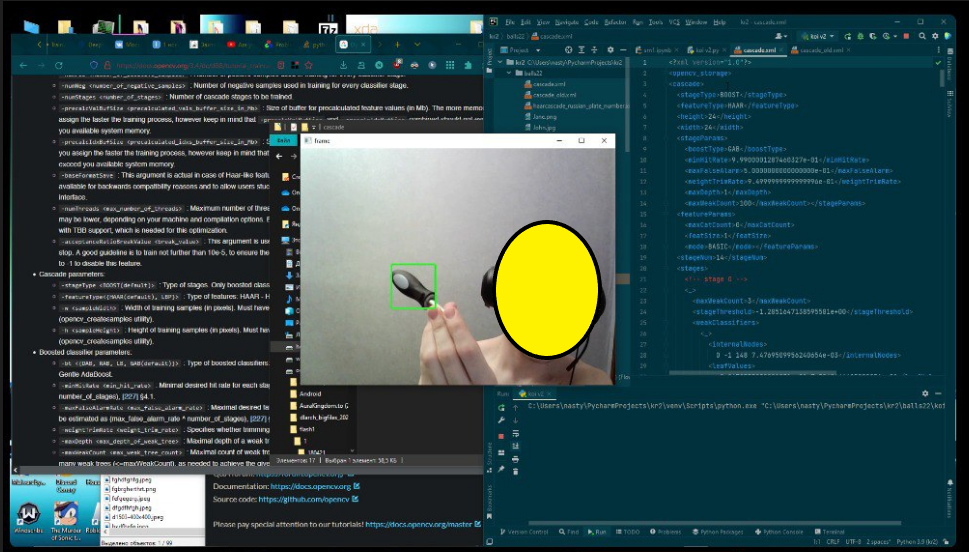


Рисунок 8. Трансляция видеопотока с камеры

А выходными данными нашего программного кода являются:

Выделяемые объекты (отвертка, ножницы, человек) на видео и формальное подписание объектов на видео.

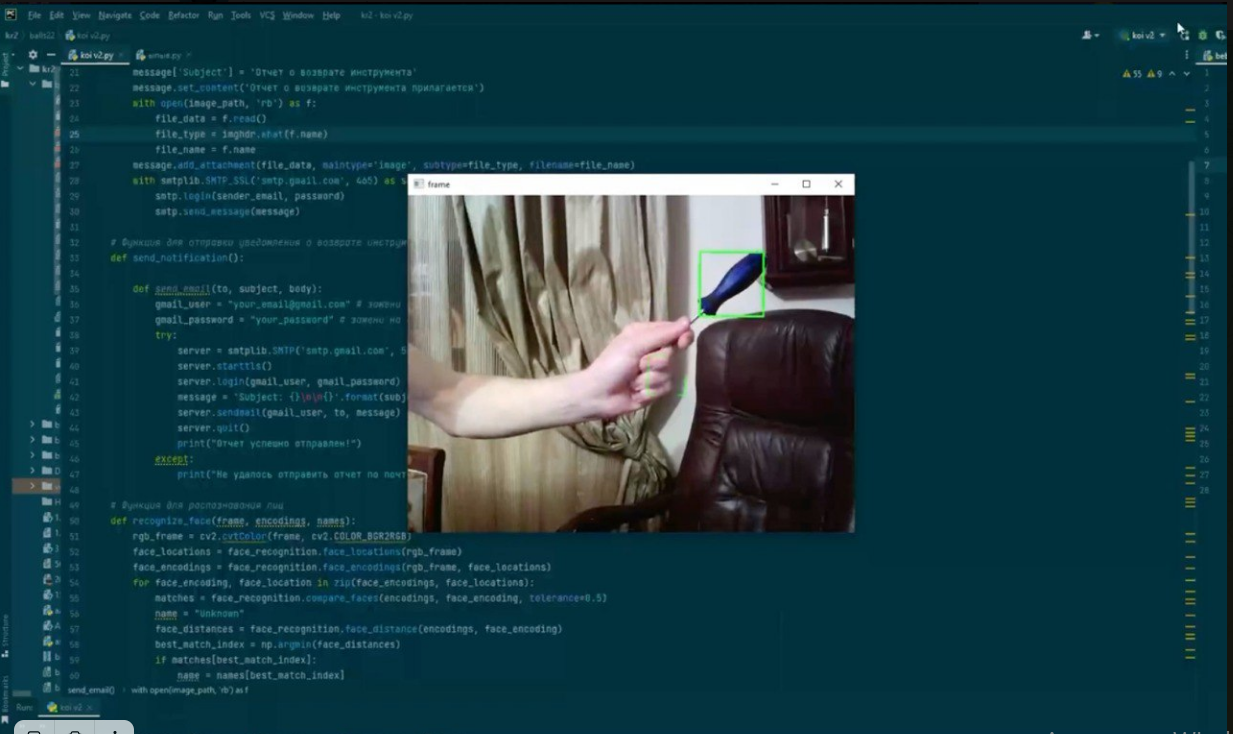


Рисунок 9. Выделение объекта на камере

* 1. **Описание и обоснование выбора состава технических и программных средств**
     1. **Состав технических и программных средств**

Для нормальной работы кода нашей программы предъявляются данные требования:

1. 100 мб свободного дискового пространства;
2. 256 мб ОЗУ или более;
3. Видеоадаптер и монитор, подходящие по совместимости VGA;
4. Разрешение на доступ к камере;
5. Разрешение на установление новых обученных моделей;
6. Разрешение на отправку уведомлений и почтовых сообщений на эл. ящик.

Для работы программы необходим следующий состав программных средств:

1. macOS Ventura 13.3.1, OC Windows 10;
2. Raspberry Pi 4 камера или камера ноутбука;
3. Совместимость с библиотекой OpenCV;
4. Поддержка XML файлов.
   * 1. **Обоснование выбора технических и программных средств**

При несоответствии требований программных средств компьютера или смартфона код может некорректно работать, в результате чего программа завершится с ошибкой аварийно. При невозможности доступа к камере, приложение не будет работать.

Для разработки программного продукта был сделан выбор в пользу языка программирования Python с использованием библиотеки OpenCV, так как это является удобным решением с точки зрения распознавания и обработки объектов с помощью камеры.

1. **ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**
   1. **Ориентировочная экономическая эффективность**

Данный программный код разрабатывался в рамках курсовой работы для ОП "Программная инженерия" на 2 курсе, не ориентирован на получение прибыли и не имеет цели обеспечения экономической эффективности, поскольку он абсолютно бесплатно предоставляется пользователям для демонстрации работы алгоритма распознавания объектов с помощью видеопотока, полученного с камеры, с использованием средств библиотеки OpenCV и языка Python.

* 1. **Предполагаемая потребность**

«Приложение для контроля за оборотом инструментов “КОИ”» может быть использован для отслеживания и контроля оборота инструментов на стенде. Таким образом, оно поможет отслеживать оборот инструментов и при утере одного с помощью отчетов найти последнего пользователя данного инструмента.

* 1. **Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами**

Пользуясь поиском по Интернет просторам, были найдены такие альтернативы приложению «КОИ»:

1. FaceKit:

FaceKit – это приложение для смартфона, которое полностью обеспечивает наведение порядка в учете и выдаче инструментов. В момент выдачи инструмента работникам, приложение фиксирует, кому, когда и какой инструмент был выдан, на какой объект перемещен, создает отчет, актуальный в каждый момент времени.

Данное решение позволяет пользователям не тратить время на самостоятельное занесение отчетов о пользовании инструментов и материалов. Однако, в таком случае занесение информации происходит через чат путем самостоятельного фотографирования объектов, в то время как КОИ реализует все самостоятельно с помощью камеры. Еще одним минусом использования этого приложения является доступность только для смартфонов, в то время как КОИ может быть открыт еще с компьютера.

1. Union EAM:

Union EAM – программа учета рабочего инструмента на предприятии. Union всегда отображает актуальные пользовательские характеристики для каждого инструмента и отчетность о пользовании того или иного объекта.

Данное решение позволяет ускорить работу предприятия и отслеживание доступных объектов на нем, упростить регистрацию выдачи и возврата инструмента. Однако, приложение отслеживает объекты с помощью специальных физических серийных номеров, которые со временем стираются, в результате чего контроль за оборотом объектов утрачивается. Также несомненными минусами являются неудобный интерфейс, заоблачная стоимость использования приложения и личное занесение информации об объекте в приложение работником предприятия.

1. EqMan:

EqMan – это приложение для учета и выдачи инструментов. В момент выдачи инструмента работникам, приложение фиксирует, кому, когда и какой инструмент был выдан, показывает историю перемещений, используя двойную верификацию. Однако, данное приложение несовместимо с некоторыми версиями Операционных Систем мобильных устройств, многое зависит еще от камеры и распознавания QR-кодов.

1. **ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ**
2. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
12. Eric Matthes Python Crash Course: A hand-on, Project-based Introduction to Programming. – San Francisco: no starch press/Перевод Питер, 2017, 492 с.
13. Mark Lutz Python Pocket Reference Fifth Edition – O’Reilly/Издательский дом “Вильямс”.
14. Learning OpenCV 5 Computer Vision with Python by Joseph Howse and Joe Minichino

# 

**ПРИЛОЖЕНИЕ 1**

**ПОКАЗ И ПОЯСНЕНИЕ НАЗНАЧЕНИЯ ФУНКЦИЙ В ПРОГРАММНОМ КОДЕ**

|  |  |
| --- | --- |
| **Название** | **Описание** |
| def send\_email(image\_path) | функция отправки отчета на почту, которая возможна с помощью метода EmailMessage из модуля email.message, по итогу отправляется изображение объекта |
| def forFrame(frame\_number, output\_array, output\_count, detected\_frame) | функция отображения в консоле распознанного предмета на видеопотоке камеры и выделение его в окне видеопотока зеленым прямоугольником с подписью названия |
| def send\_notification() | функция для отправки уведомления о возврате инструмента. |
| Основное тело программы | распознавание отвертки и объектов с выделением происходит в основном теле программы. |

# **ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированх |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |