МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «ВГУ»)**

Факультет компьютерных наук

Кафедра информационных систем

Отчет по учебной практике (ознакомительной)

<Компьютерное зрение. Распознавание штриховых кодов в реальном времени.>

Направление 09.03.02 «Информационные системы и технологии»

Зав. кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_Д. Н. Борисов, к.т.н., доцент\_\_\_\_\_\_\_\_\_ \_\_. \_\_.2022

Подпись, расшифровка, ученая степень, звание

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_М. Д. Шеменев\_\_\_\_\_\_\_\_\_\_\_\_ \_\_. \_\_.2022

Подпись, расшифровка подписи

Руководитель практики от ВГУ\_\_\_\_\_\_\_\_\_\_\_\_\_Е. В. Попова \_\_. \_\_.2022

Подпись, расшифровка, ученая степень, звание

Воронеж 2022

Содержание

[СОДЕРЖАНИЕ 2](#_Toc105793358)

[ВВЕДЕНИЕ 3](#_Toc105793359)

[1 ПОСТАНОВКА ЗАДАЧИ 4](#_Toc105793360)

[1.1 Средства реализации 4](#_Toc105793361)

[2 Основные понятия штрихового кодирования 7](#_Toc105793362)

[3 РЕАЛИЗАЦИЯ 10](#_Toc105793363)

[3.1 Структура приложения 10](#_Toc105793364)

[3.2 Реализация интерфейса 10](#_Toc105793365)

[ЗАКЛЮЧЕНИЕ 16](#_Toc105793366)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 17](#_Toc105793367)

[ПРИЛОЖЕНИЕ 18](#_Toc105793368)

Введение

Компьютерное зрение - это область искусственного интеллекта, которая обучает компьютеры интерпретировать и понимать визуальный мир. Используя цифровые изображения с камер и видео и модели глубокого обучения, машины могут точно идентифицировать и классифицировать объекты - а затем реагировать на то, что они "видят".

Компьютерное зрение распространилось в самых различных сферах: считывание штриховых кодов, системы видеонаблюдения, системы управления автомобилями, медицинские системы анализа изображений, технологии дополненной и виртуальной реальности и во многих других областях. Машинное зрение — это попытка смоделировать механизм получения и обработки визуальной информации в человеческом мозге.

Компьютерное зрение сосредотачивается на применении, в основном промышленном, например, автономные роботы и системы визуальной проверки и измерений. Это значит, что технологии датчиков изображения и теории управления связаны с обработкой видеоданных для управления роботом и обработка полученных данных в реальном времени осуществляется программно или аппаратно. Обработка изображений и анализ изображений в основном сосредоточены на работе с 2D изображениями.

В современном мире металлургические компании нуждаются в автоматической проверке и контроле качества своих товаров. Внедрение методов компьютерного зрения поможет повысить эффективность бизнеса, увеличит рост отгрузок и выручки, понизит человеческие трудозатраты и процент ошибок, допускаемый сотрудниками.

1. Постановка задачи

Задачей данной учебной практики является разработка программы, реализующей распознавание штриховых кодов в реальном времени.

Наиболее важные алгоритмы этой программы:

* алгоритм обработки изображений с камеры,
* алгоритм декодирования штрихкода,
* алгоритм выделения найденных штриховых кодов на изображении;
  1. Средства реализации

Разработка ведётся с использованием языка программирования Python, среды разработки PyCharm (v.2021.3.2)**,** OpenCV-Python и других библиотек:

Future, Numpy, Pillow, Pyzbar, Python-Barcode.

*PyCharm* — одна из самых мощных и популярных интегрированных сред разработки (IDE) для Python. Она разработана и поддерживается JetBrains и доступна как окончательная версия для сообщества. Эта многофункциональная IDE обеспечивает быструю разработку и помогает улучшить качество кода.

Основные особенности:

* Глубокое понимание. PyCharm действительно понимает и глубоко понимает код, а также контекст, что делает её уникальной среди других IDE.
* Эргономика разработчика. PyCharm разработана на основе принципа кодирования, согласно которому разработчикам должно быть позволено писать код с как можно меньшим количеством отвлекающих факторов. Вот почему в этом случае редактор — единственное, что видно на экране, с выделенными ярлыками для всех других функций, не связанных с кодированием.
* Встроенные инструменты разработчика. Чтобы помочь разработчикам организовать рабочий процесс, PyCharm предлагает им широкий набор инструментов.
* PyCharm поддерживает Pandas, Numpy, Matplotlib и другие библиотеки для научных вычислений. IDE обеспечивает умное редактирование, позволяет просматривать наборы данных в виде графиков и в табличной форме.
* PyCharm можно установить на нескольких компьютерах в операционных системах Windows, Mac OS или Linux — все возможности IDE будут доступны на любом из них [4].

*OpenCV-Python* – это библиотека расширений Python, предназначенная для решения задач компьютерного зрения [5].

*Future* – это отсутствующий уровень совместимости между Python 2 и Python 3. Он позволяет вам использовать единую чистую кодовую базу, совместимую с Python 3.x, для поддержки как Python 2, так и Python 3 с минимальными накладными расходами [6].

*Numpy* – это основной пакет для научных вычислений в Python. Это библиотека Python, которая предоставляет объект многомерного массива, различные производные объекты (такие как маскированные массивы и матрицы) и набор подпрограмм для быстрых операций с массивами, включая математические, логические, манипуляции с фигурами, сортировку, выбор, ввод-вывод, дискретные преобразования Фурье, основы линейной алгебры, основные статистические операции, случайное моделирование и многое другое [7].

*Pillow* – это удобный форк PIL. Эта библиотека обеспечивает обширную поддержку форматов файлов, эффективное внутреннее представление и довольно мощные возможности обработки изображений [8].

*Pyzbar* – это библиотека, которая считывает одномерные штрихкоды и QR-коды из Python 2 и 3 с помощью библиотеки zbar [9].

*Python-Barcode* – библиотека, которая предоставляет легко и просто создать штрихкод в Python [10].

1. Основные понятия штрихового кодирования

Штриховой код представляет собой последовательность расположенных по правилам определенной символики темных (штрихов) и светлых (пробелов) прямоугольных элементов различной ширины, которая обеспечивает представление символов данных в машиночитаемом виде (Рисунок 1).



1. Примеры штрихкодов

Данными могут быть как буквы и цифры, так и специальные графические и управляющие символы, используемые в программных и технических средствах обработки и передачи информации.

Штриховой код является одним из средств систем автоматической идентификации товара, к которой также относятся средства цифровой, магнитной, радиочастотной, звуковой и визуальной идентификации (магнитная карточка, радиочастотная бирка и т. д.). Его главное преимущество перед другими средствами автоматической идентификации заключается в возможности оперативно передавать информацию о товаре по системе электронной связи, т. е. штриховой код является эффективным средством телекоммуникации.

Назначение штрихового кода:

- оперативная идентификация товара и производителя;

- проведение торговых сделок «без бумаг»: штриховой код сокращает издержки на делопроизводство с 15% до 0,5-0,3% от стоимости товара;

- автоматизированный учет и контроль товарных запасов;

- оперативное управление процессом товародвижения: отгрузкой, транспортировкой и складированием товаров (производительность труда по обеспечению товародвижения повышается на 30%, в некоторых случаях – на 80%);

- информационное обеспечение маркетинговых исследований [1, с. 146].

Штрих (полоса) - темная зона изображения на однотонном светлом фоне, ограниченная прямыми параллельными линиями или концентрическими окружностями. Элементы штрихового кодирования наносятся на поверхность носителя, имеющего определенные светотехнические характеристики. При этом штрихи, наносимые с помощью красителей или каких-либо других средств, хорошо поглощают свет на определенных длинах волн, а фоновая поверхность хорошо его отражает, что используется при оптическом считывании.

Пробел – пространство между штрихами. В большинстве кодов в ширине пробела заключена определенная информация, лишь в некоторых кодах пробел – вспомогательная часть изображения и выполняет функцию элемента-разделителя. Высота и ширина штриха (пробела) – размеры изображения, выраженные в единицах измерения (миллиметрах, долях дюйма) или в безразмерных единицах (модулях) [2, c. 3].

Ширина самого узкого элемента (штриха или пробела) принимается в качестве основного размера - модуля. Ширина любого элемента должна быть либо кратна модулю (например, в символике «Код 128» допустимы элементы шириной 1, 2, 3 или 4 модуля), либо должно выдерживаться постоянное отношение между широкими и узкими элементами (например, в символике «Код 39» элементы двух размеров - с заданным отношением ширины широких элементов к узким).

Определенные комбинации штрихов и пробелов образуют набор знаков штрихового кода. Например, в символике «Код 39» каждый знак штрихового кода состоит из девяти элементов (из которых три широких и шесть узких) и должен быть представлен в пяти и четырех пробелах. Каждой комбинацией штрихов и пробелов – знаку штрихового кода соответствует, как правило, знак данных или специальный знак.

Последовательность расположенных слева направо знаков штрихового кода, кодирующих данные, начинаются знаком «Старт» и заканчивающаяся знаком «Стоп» с примыкающими к этим знакам свободными полями, называется символом штрихового кода. Символ штрихового кода и есть тот законченный графический объект, который подлежит машинному считыванию [3, с. 147].

Код двуцветный – код, изображение которого содержит информацию на определенных длинах волн в виде темных и светлых штрихов. Код контролируемый – код, в изображении знаков и кодовых слов которого заложена избыточная информация, обеспечивающая обнаружение ошибки считывания.

Другими словами, штриховой код - символьный ключ к информации в базах данных. Единственная информация, которую он несет, просто ряд чисел и/или символов. Его назначение – уникальная связь с информацией, сохраненной внутри компьютерной системы, которая может быть автоматически быстро, легко и точно извлечена из базы данных.

1. Реализация
   1. Структура приложения

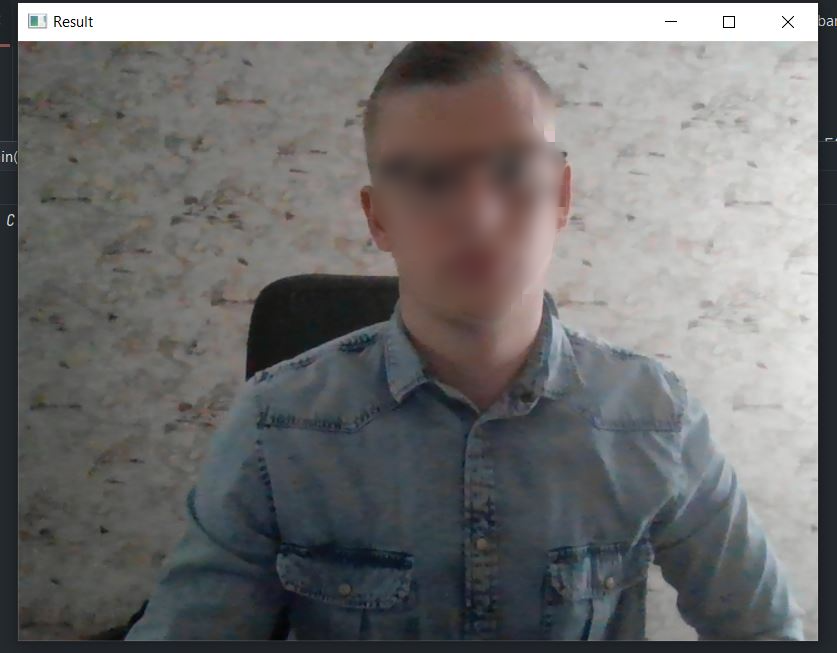
В данном проекте для обработки штрихкодов в реальном времени используется веб-камера ноутбука (по умолчанию). Именно она делает захват изображения, с которым мы дальше работаем.

Далее, с помощью метода *decode* из библиотеки *pyzbar* мы получаем штрихкод, как объект. Найденные штрихкоды выводим в консоль, выделяем на изображении и получаем их данные.

Более подробно код программы представлен в [приложении](#link).

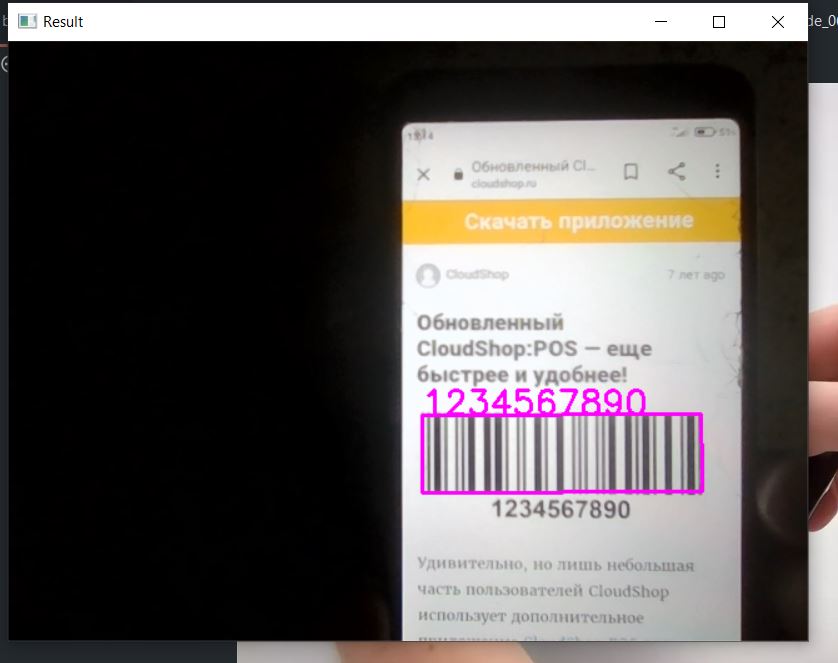
* 1. Реализация интерфейса

Перед запуском программы появляется окно, где отображается результат видеосъёмки (Рисунок 2).



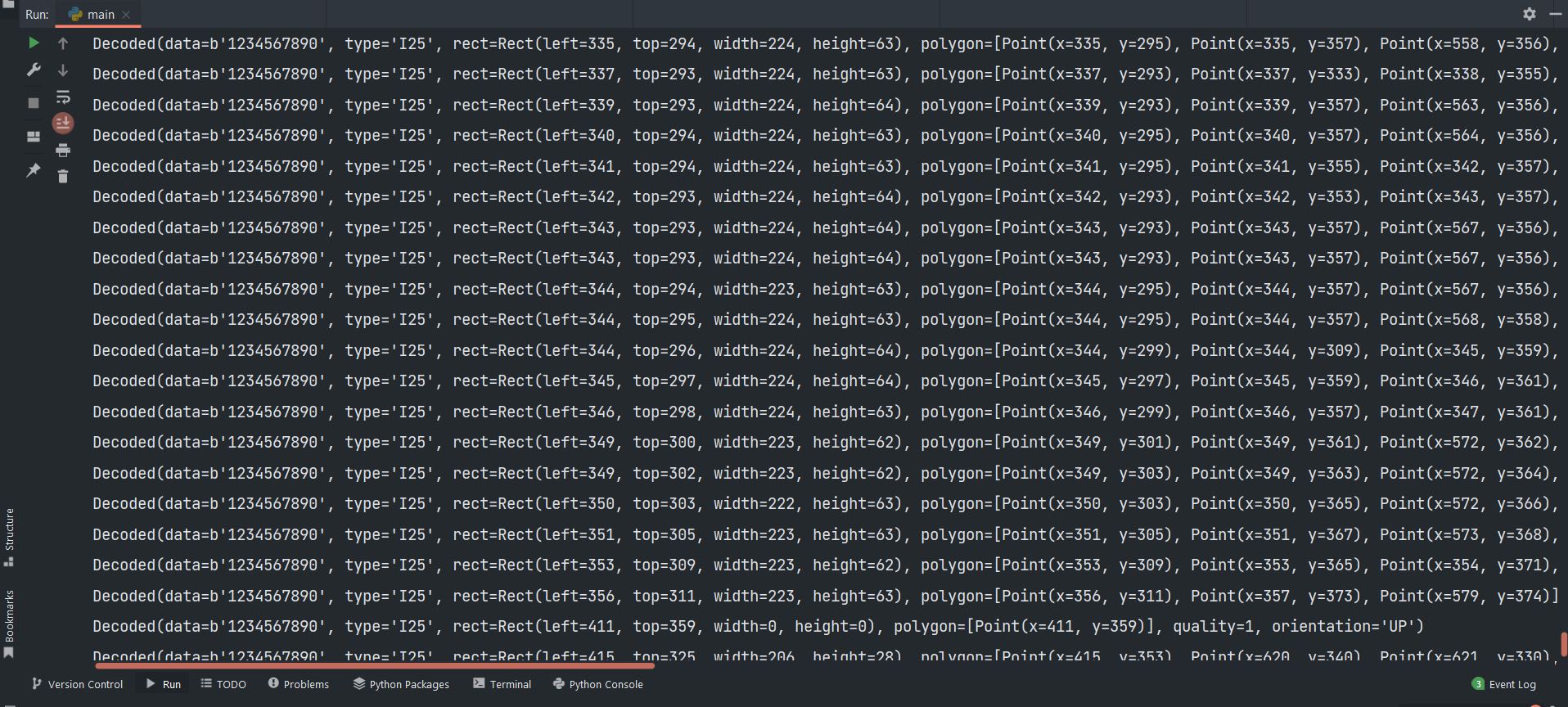
1. Запуск приложения

После показа штрихкода веб-камера захватит изображение с ним. После чего программа выделит данный штрихкод и его данные (Рисунок 3).



1. Обнаружение штрихкода

В консоли также отображаются данные штрихового кода (Рисунок 4).



1. Вывод данных штрихкода в консоль

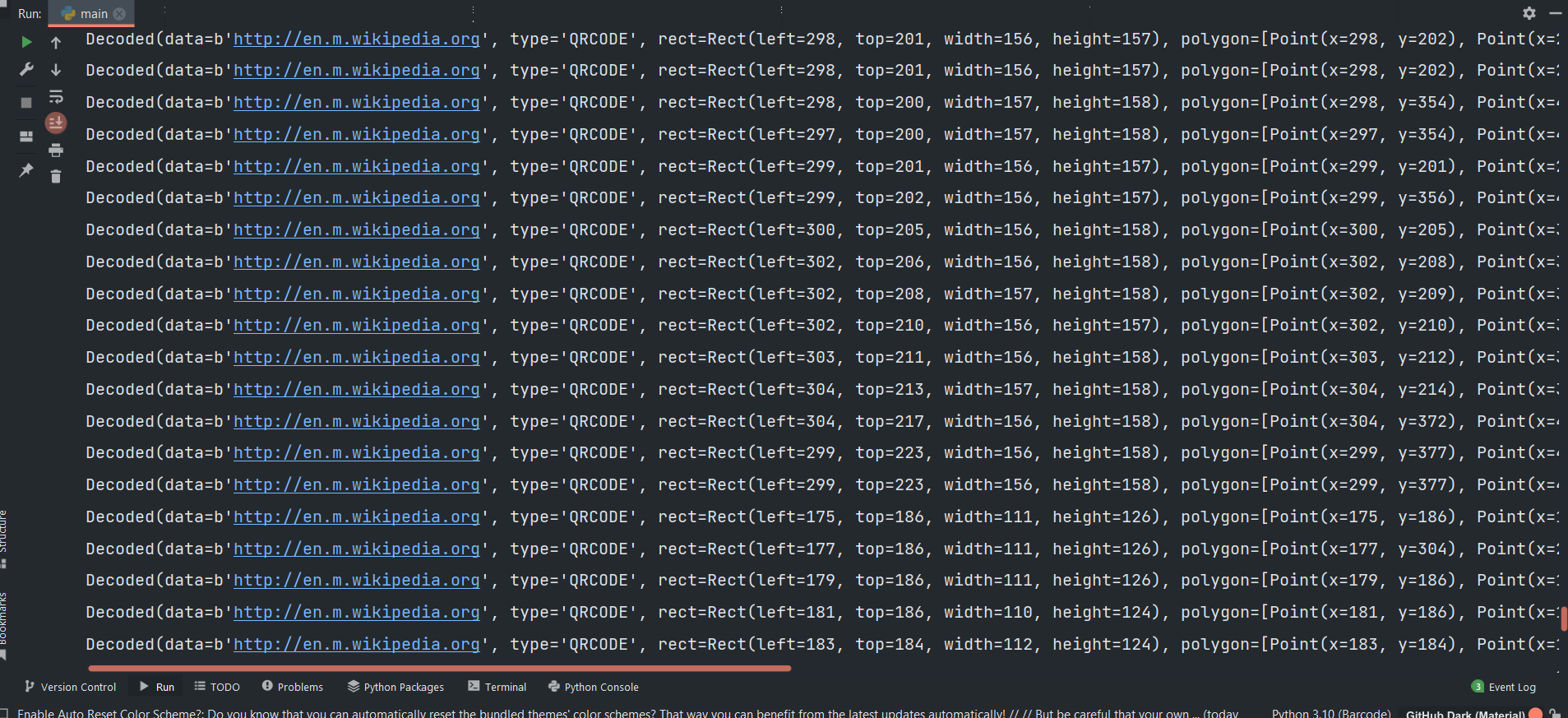
Рисунок 3 и Рисунок 4 показывают данные, тип, геометрическое расположение штрихкода при видеосъёмке.

Также программа может работать и с QR-кодами. Подадим на вход Quick Response code (Рисунок 5).



1. Обнаружение QR-кода

Теперь откроем консоль и увидим также информацию о данном QR-коде (Рисунок 6).



1. Вывод данных QR-кода в консоль

Рисунок 5 и Рисунок 6 также показывают нам данные, тип и геометрическое расположение QR-кода при видеосъёмке.

Заключение

При выполнении работы были рассмотрены основные понятия штрихового кодирования. Также была реализована программа, которая позволяет обнаружить в реальном времени штриховые и QR коды, а также декодировать их.

На данный момент разработаны следующие элементы программы:

* Простой графический интерфейс;
* Функция для выделения контуров штриховых кодов.

В дальнейшем можно улучшить графический интерфейс под конкретную область применения программы.

Список использованных источников

1. Дмитриченко М. И. Экспертиза качества и обнаружение фальсификации продовольственных товаров: Учебное пособие/ М. И. Дмитриченко, Л. А. Забодалова, В. Е. Куцакова – СПб.: Питер, 2003 – 160 с.
2. Колесников А.В. Гибридные интеллектуальные системы. Технология разработки / А.В. Колесников – СПб.: Изд-во СПбГТУ, 2011. – 111 с.
3. Галушкина А.И. Нейрокомпьютеры и их применение. / под общей ред. А.И. Галушкина, Я.З. Цыпкина. – М.: ИПРЖР, 2011. – 145 с
4. [Электронный ресурс]. URL: <https://www.jetbrains.com/ru-ru/pycharm/features/> (Дата обращения: 05.06.22)
5. [Электронный ресурс]. URL: <https://opencv.org/> (Дата обращения: 05.06.22)
6. [Электронный ресурс]. URL: <https://pypi.org/project/future/> (Дата обращения: 05.06.22, 06.06.22)
7. [Электронный ресурс]. URL: <https://numpy.org/> (Дата обращения: 05.06.22)
8. [Электронный ресурс]. URL: <https://pillow.readthedocs.io/en/stable/> (Дата обращения 04.06.22)
9. [Электронный ресурс]. URL: <https://pypi.org/project/pyzbar/> (Дата обращения 07.06.22)
10. [Электронный ресурс]. URL: <https://pypi.org/project/python-barcode/> (Дата обращения 07.06.22)

Приложение

Программный код

from pyzbar.pyzbar import decode  
import cv2  
import numpy as np  
  
  
def create\_border(barcode\_polygon):  
 pts = np.array([barcode\_polygon], np.int32)  
 pts = pts.reshape((-1, 1, 2))  
 return [pts]  
  
  
def main():  
 # Initialize videocapture  
 cap = cv2.VideoCapture(0) # 0 = open default camera  
 cap.set(3, 640)  
 cap.set(4, 680)  
  
 while True:  
  
 # wait for a new frame from camera and store it into 'image\_barcode'  
 success, image\_barcode = cap.read()  
  
 # walk through found barcodes  
 for barcode in decode(image\_barcode):  
 print(barcode)  
  
 # selection by polygon barcode  
 cv2.polylines(image\_barcode, create\_border(barcode.polygon), True, (255, 0, 255), 2)  
  
 # getting data from barcode  
 data = barcode.data.decode('utf-8')  
   
 # add data to image\_barcode  
 cv2.putText(image\_barcode, data, (barcode.rect[0], barcode.rect[1]), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9,  
 (255, 0, 255), 2)  
  
 # result display  
 cv2.imshow('Result', image\_barcode)  
 width = cv2.getWindowImageRect('Result')[2]  
 height = cv2.getWindowImageRect('Result')[3]  
 cv2.resizeWindow('Result', width, height)  
  
 # exit  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
  
 # When everything done, release the capture  
 cap.release()  
 cv2.destroyAllWindows()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()