ОЛИМПИАДА ШКОЛЬНИКОВ «ШАГ В БУДУЩЕЕ»

НАУЧНО-ОБРАЗОВАТЕЛЬНОЕ СОРЕВНОВАНИЕ «ШАГ В БУДУЩЕЕ, МОСКВА»

1381

регистрационный номер

Информатика и системы управления
название факультета

Программное обеспечение ЭВМ и информационные технологии
название кафедры

Повышение скорости и точности распознавания одномерного штрихкода на изображении путем поиска области штрихкода

название работы

Автор: ***Боренко Анастасия Денисовна***
фамилия, имя, отчество
ГБОУ Школа №1564
наименование учебного заведения, класс

Научный руководитель: **Боренко Денис Борисович**
фамилия, имя, отчество
ООО «МитФиш»
место работы
бухгалтер-операционист
звание, должность

_____
подпись научного руководителя

Москва - 2019

Аннотация

Невозможно представить современную жизнь без использования штрихкода. Он применяется для идентификации товара на кассе в магазинах, в документах, на производстве. Обычно для его считывания используют сканеры, но иногда необходимо распознать штрихкод на изображении. Существуют программы, способные это сделать. Например, zbarimg, написанная для Linux. Но если штрихкод занимает не все изображение, то число ошибок таких программ возрастает, а скорость работы падает. Существенно улучшить их качество работы можно, выделив область штрихкода. В своем проекте я исследую проблему определения области одномерного штрихкода на изображении.

Оглавление

Введение

Штрихкод сегодня активно используется в торговле. Это представленный графически идентификатор товара, который наносят на поверхность или упаковку продукта. Его можно считывать специальными сканирующими устройствами.

Нтрихкод находит применение в системах архивного хранения. Документы, требующие длительного хранения, часто сканируют и хранят в цифровом формате. Сканированный документ можно привязать к объекту базы данных, для последующего поиска. Удобно сделать это, если на бланк документа предварительно нанести штрихкод. Вручную считывать штрихкод с каждой накладной перед сканированием трудоемко. Намного удобней и быстрее считывать штрихкоды с уже сканированных документов. У QR-кода есть специальные маркеры, которые позволяют определить его положение, угол поворота и перспективные искажения, у одномерных же штрихкодов таких маркеров нет, поэтому их распознавание на изображении может вызывать трудности. Однако одномерный штрихкод обладает большой устойчивостью к искжениям.

Причиной, побудившей меня провести данное исследование, стало знакомство с существующей архивной системой хранения сканированных копий документов у одной компании. В указанной системе для распознавания штрихкода с отсканированного документа используется программа zbarimg, которая умеет распознавать штрхкоды. Точность работы этой программы на неподготовленном изображении не очень велика, как и скорость. Программа работает долго и может допускать ошибки при повернутом на некоторый угол изображении. Написаны алгоритмы, которые повышают точность работы этой программы, осуществляющие многократное последовательное распознавание, чередуемое с поворотом изображения на ранее заданные углы. Процесс распознавания отнимает большое количество времени. Во многом это связано с размером изображения, которое приходится поворачивать и анализировать zbarimg, и с тем, какой процент изображения занимает штрихкод: если подавать программе изображение, содержащее только штрихкод, то она работает гораздо быстрее и точнее.

Учитывая выше сказанное, мною был предложен способ повышения качества и скорости распознавания штрихкода в отсканированном изображении. В его основе лежит предварительное выделение области штрихкода на изображении. Не трудно заметить признак штрихкода — это штрихи. Область изображения, содержащая такой же узор, с высокой долей вероятности не повторится. В ходе работы мною разработан метод статистического анализа фрагментов изображения на предмет нахождения во фрагменте штрихкода. Штрихкоду характерна очень частая смена цвета пикселей по ширине. Прототип программы, выделяющей штрихкод, был написан на языке Python, поскольку он относительно изучен мною по школьным проектам, что обеспечило скорость и простоту написания алгоритма. Прототип также визуализирует работу метода, с помощью библиотеки Qt с прослойкой для Python – PQt5 и библиотеки Pillow.

Метод статистического анализа фрагментов изображения

Метод состоит из следующих этапов:

- перевода изображения из цветного в черно-белое
- разбиения изображения на блоки
- подсчет математического ожидания для каждого блока
- подсчет дисперсии для каждого блока
- анализ результатов

Перевод изображения из цветного в черно-белое

Штрихкод изображается черно-белыми полосками, поэтому проще анализировать черно-белую картинку, для этого необходимо перевести изображение в черно-белое, в котором каждый пиксель изображения закодирован одним битом. Мною рассмотрено два способа конвертации изображения в черно-белое.

В обоих способах предварительно цветное изображение переводится по яркости в серое. В первом способе, по серой точке (128), которая выбирается в центре всего диапазона (0-255) градаций серого, пиксели разделяются на черно-белое. Но такой способ подходит только для правильно экспонированных фотографий и сканированных изображений (рис 1.)

Рисунок 1: Изображение с правильной экспозицией

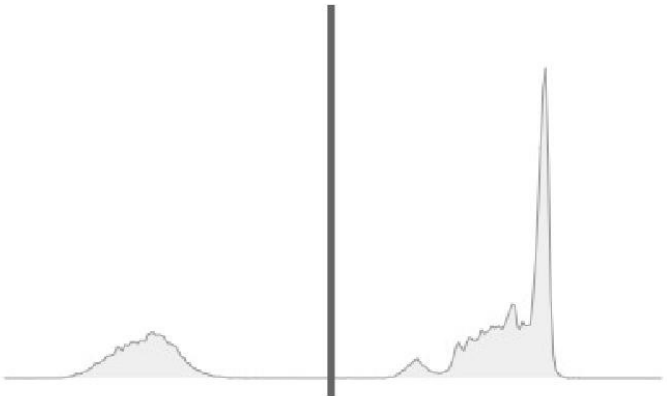

Рисунок 2: Гистограмма к рис.1

Гистограмма на рисунке 2 демонстрирует распределение пикселей на изображении 1 по яркости. При использовании первого способа, необходимо взять все пиксели с цветами левее медианы (жирной линии), и преобразовать в черный, все цвета справа от медианы надо преобразовать в белый.

Для фотографий с плохой экспозицией этот способ не подойдет, потому что самый белый цвет фотографии не будет лежать правее медианы графика или, наоборот, самый черный цвет окажется правее медианы как на гистограммах рис. 3 и рис. 4.

Рисунок 3: Гистограмма пересвеченного изображения

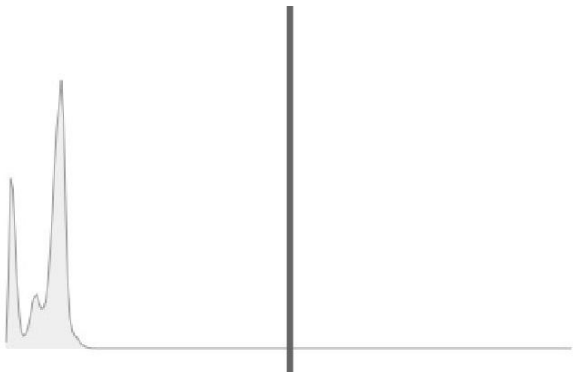

Рисунок 4: Гистограмма затемненного изображения

Второй способ позволяет избежать потери черного и белого цветов при переводе из серого в черно-белый, а значит, и потери штрихкода. Во-втором способе необходимо выбрать серую точку не из середины всего диапазона, а из середины только того диапазона, который присутствует на изображении, гистограмма рис.5 иллюстрирует этот метод.

Рисунок 5: Гистограмма затемненного изображения, 2-ой способ обработки

В написанной программе используется первый способ, потому что он быстрый и достаточно точный для сканированных накладных из-за постоянной и правильной экспозиции, однако если эту программу использовать на фотографиях, то необходимо применять 2-ой способ, в таком случае даже засвеченные или затемненные фотографии будут переводиться в черно-белый цвет без серьезных потерь, и штрихкод не будет преобразован в только белую или только черную область.

Разбиение изображения на блоки

Необходимо разделить изображение на фрагменты, высота которых будет не больше половины высоты штрихкода, а ширина будет в две высоты блока, такая ширина гарантированно будет меньше ширины штрихкода. Программа прототип иллюстрирует этот шаг на рис. 6.

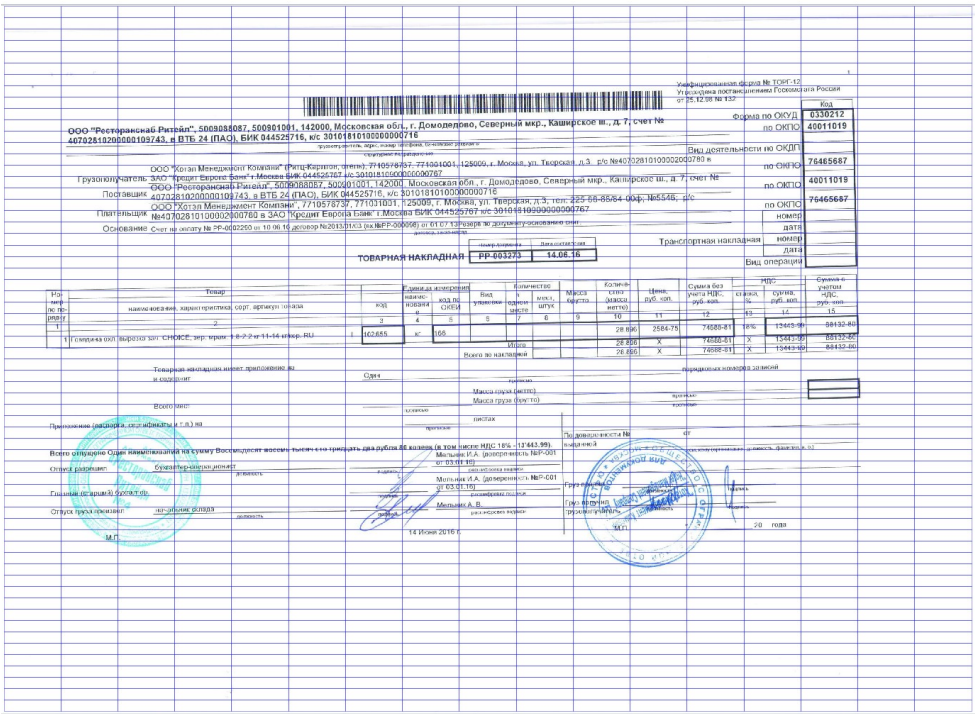

Рисунок 6: Результат работы программы-прототипа - деление изображения на фрагменты

Подсчет математического ожидания для каждого блока

В данном методе за случайную величину принимается количество групп пикселей одного цвета, идущих подряд, в строке одного блока.

На рисунке 7 показан один фрагмент изображения, не попавший в штрихкод.

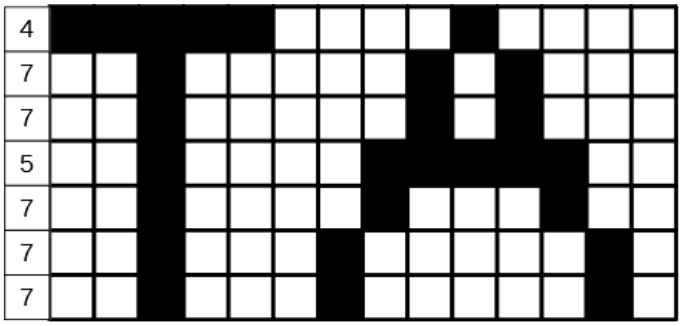

Рисунок 7: Блок вне штрихкода

В первой строке выбранного блока случайная величина составляет 4, т.е. в этой строке 4 группы пикселей.

Для выбранной случайной величины подсчитаем математическое ожидание в строках блока. В случаях когда блок не попадет в штрихкод, мат. ожидание будет мало, в блоке на рис. 7 мат ожидание будет равно 6.28. Если же выбранный блок попадет в штрихкод, то математическое ожидание будет значительно больше: 11(рис.8).

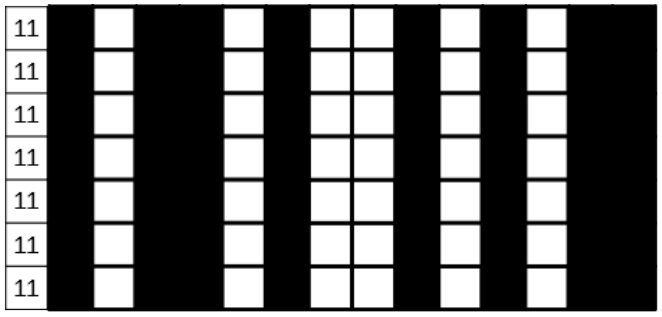

Рисунок 8: Блок в штрихкоде

Поэтому необходимо найти наибольшее мат. ожидание и сравнить с мат. ожиданием каждого блока, если они равны, то блок лежит в штрихкоде и подходит нам. Но необходимо учесть погрешность, возникающую из-за дефектов изображения, тогда мат. ожидание в блоках может немного изменяться, поэтому необходимо взять запас в 15%, чтобы все блоки, находящиеся внутри штрихкода, были выделены.

Однако при таком выделении будут выбраны только блоки, строго лежащие внутри штрихкода. На рисунке 9 продемонстрирована работа программы при анализе среднего мат. ожидания. Красным выделены блоки со слишком маленьким мат ожиданием. Отчетливо видно, что выделенная область занимает не весь

штрихкод. Чтобы выбрать весь штриход необходимо добавить по одному блоку с каждой стороны.

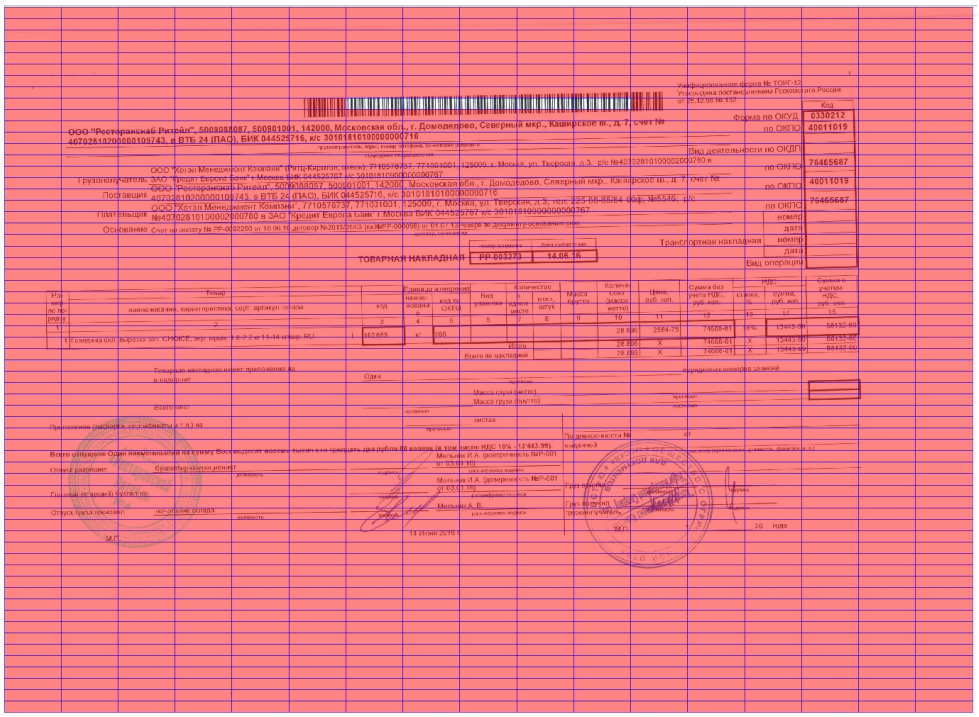

Рисунок 9: Результат работы программы при анализе сканированного документа

Для анализа и архивирования сканированных документов достаточно использовать только мат. ожидание случайной величины. Но анализ мат. ожидания может давать сбои при обработке изображения с зашумленными областями (рис. 10), потому что в зашумленных фрагментах мат. ожидание тоже будет большим.

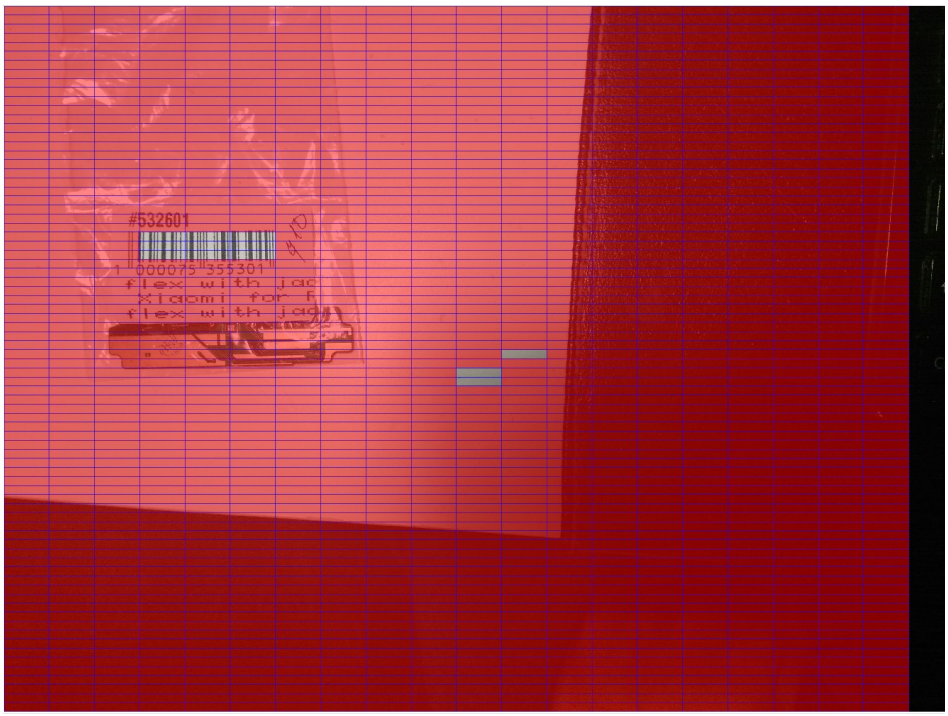

Рисунок 10: Результат работы программы по анализу мат. ожидания фотографии с зашумленными областями

Подсчет дисперсии для каждого блока

Для повышения точности алгоритма можно также посчитать дисперсию каждого блока. У блока в зашумленной области дисперсия будет больше, чем у блока в штрихкоде.

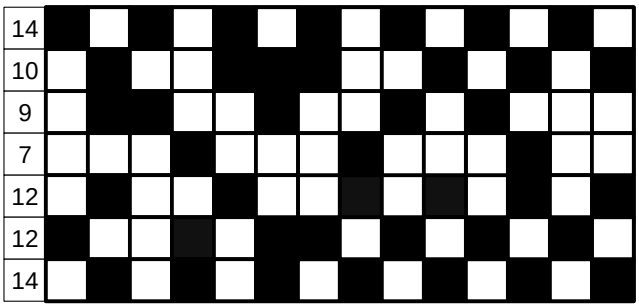

Рисунок 11: Блок в зашумленной области

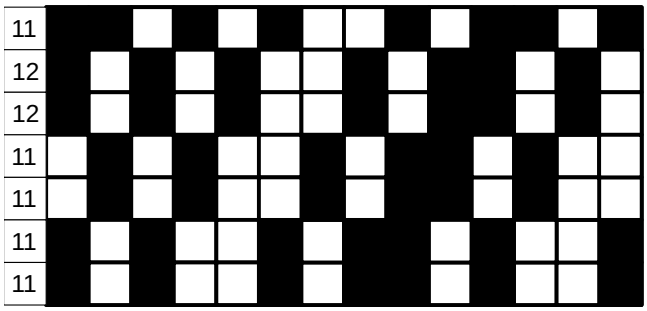

Рисунок 12: Блок в наклонненом штрихкоде

На рисунках 13 и 14 представлены графики кривой нормального распределения для блоков с рисунков 11 и 12. Для блока, представленного на рисунке 11 дисперсия равна 5.84. Для блока, представленного на рисунке 12, дисперсия равна 0.2. Очевидно, что в блоках зашумленных областей разброс случайной величины будет большой, а в штрихкоде разброс случайной величины будет стремиться к нулю.

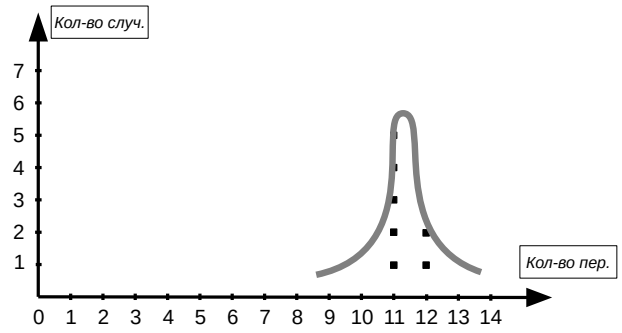

Рисунок 13: Кривая нормального распределения к рис.12

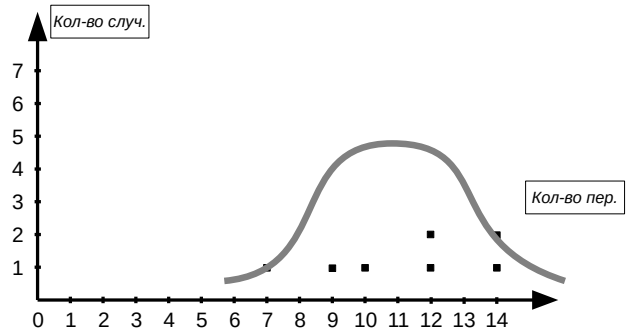

Рисунок 14: Кривая нормального распределения к рис.13

Рисунок 15 демонстрирует работу программы по подсчету дисперсии, программа выводит на экран значения дисперсии для каждого блока, а также блоки, неподходящие по дисперсии закрашивает синим.

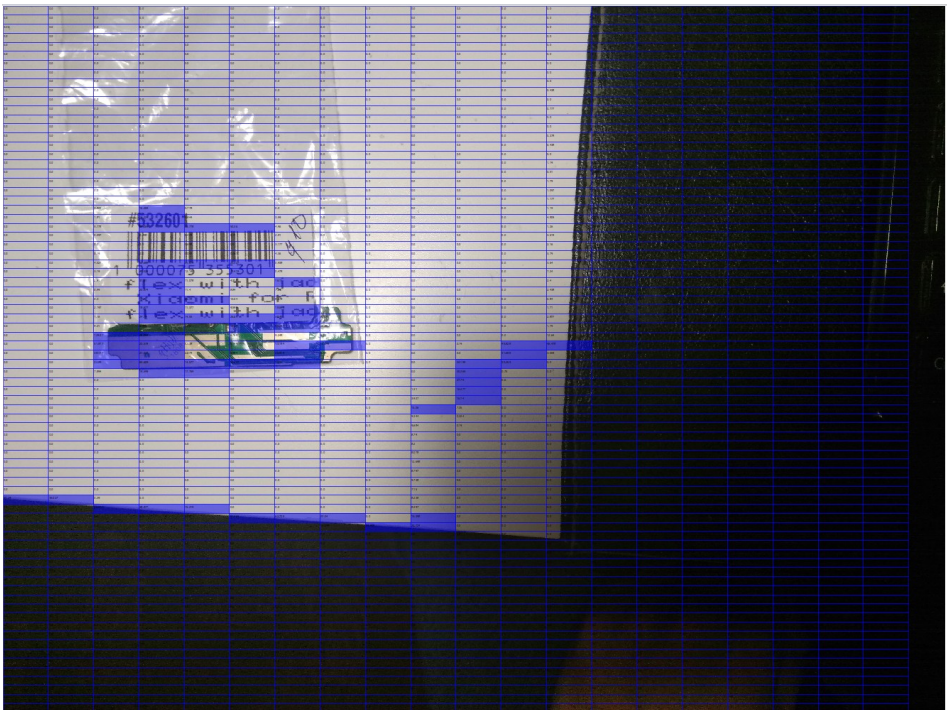

Рисунок 15: Вывод программы при подсчете дисперсии

Подсчет разности математических ожиданий

Для еще большего уточнения метода можно посчитать разность вертикального и горизонтального мат. ожидания блоков. Модуль разности горизонтального и вертикального мат. ожидания, нормированного по размеру блока, для блоков, содержащих штрихкод максимален. На рисунках 16, 17 показаны блок внутри штрихкода и блок вне штрихкода соответственно, вертикальное мат ожидание для первого блока 1/7, горизонтальное 11/14, их разность 0,9. У блока вне штрихкода вертикальное мат. ожидание 2.21/7, горизонтальное 6.29/14, их разность 0,13.

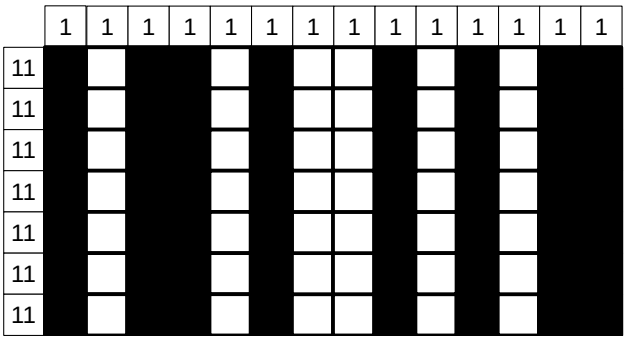

Рисунок 16: Блок внутри штрихкода

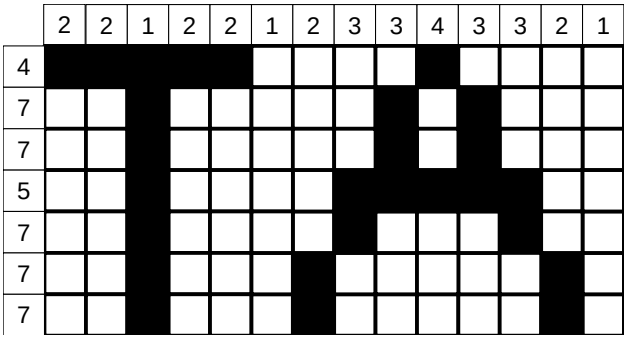

Рисунок 17: Блок вне штрихкода

Анализ результатов

Очевидно, что использовать дисперсию без математического ожидания нельзя, поэтому необходимо совмещать результаты расчетов мат. ожидания и дисперсии, тогда с высокой точностью будет выделен весь штрихкод. На рисунке 18 представлен результат работы программы после сложения результатов расчетов дисперсии и мат. ожидания.

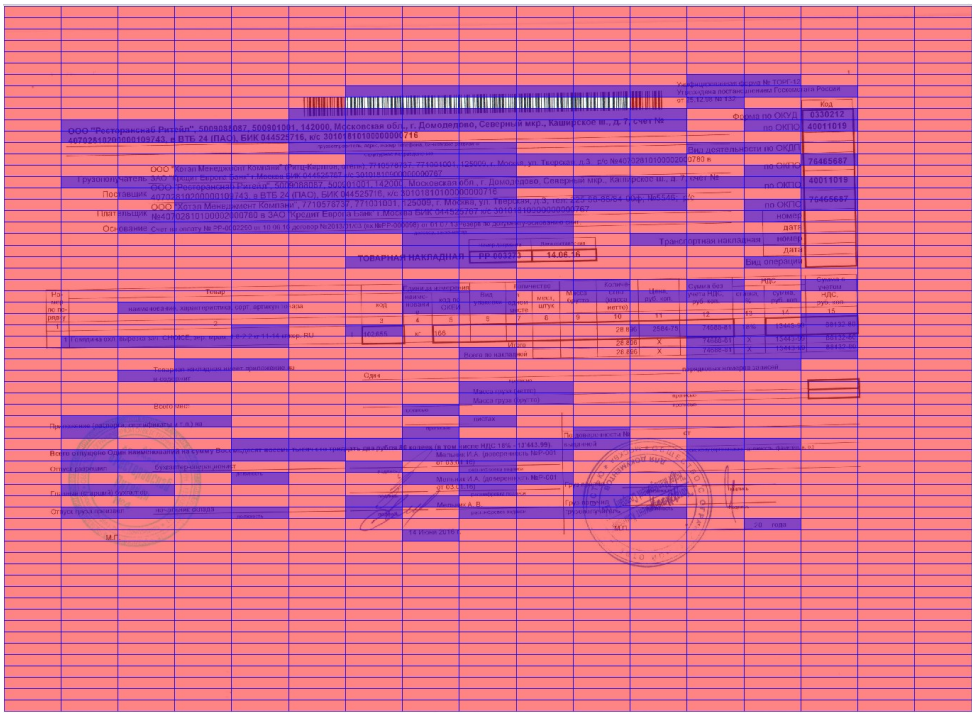

Рисунок 18: Вывод программы при анализе результатов

Демонстрационный прототип алгоритма

Программа написана на языке питон. Использован принцип декомпозиции, описанные классы вынесены в разные файлы для простоты восприятия, возможности быстрого внесения изменений и исправления ошибок. В программе используются модули itertools, Pillow, PyQt5, dotmap. Также использован класс из программы QtViewImage.py, выложенной в открытом доступе на GitHub. На рисунке 19 представлена структура программы.

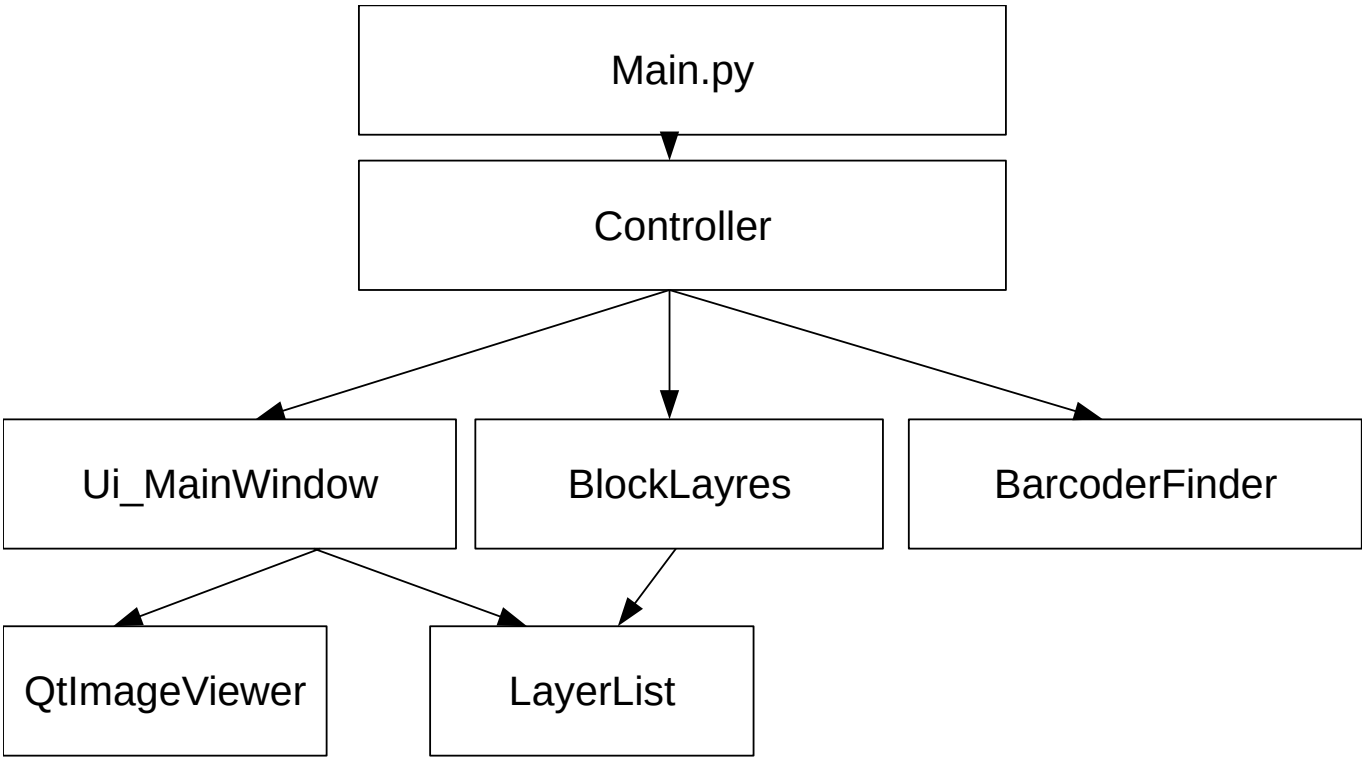

Рисунок 19: Схемотичное представление структуры программы

Загрузить изображение для анализа можно с помощью пункта «Открыть в меню «Файл» (рис. 20).

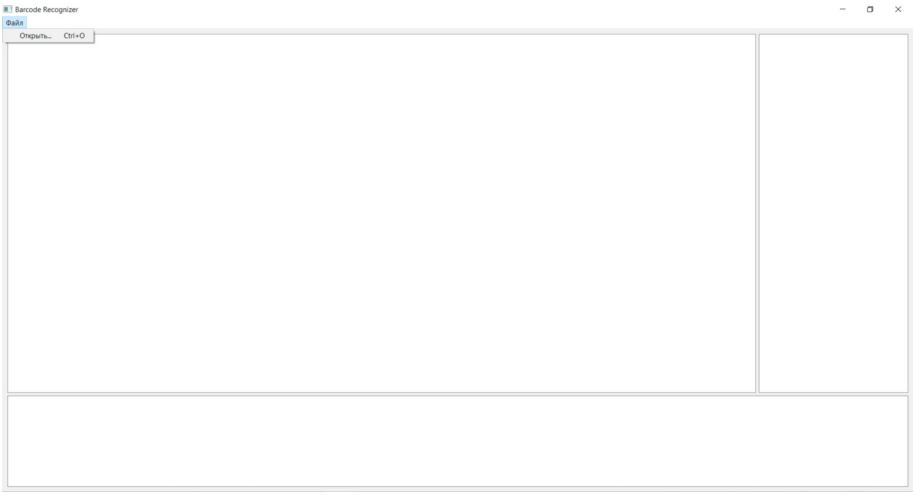

Рисунок 20: Вывод программы при запуске

После открытия изображения заполняются три окна. Верхнее левое окно показывает изображение, верхнее правое показывает слои, которые можно проявить и скрыть, а нижнее окно сообщает пользователю о ходе выполнения работы (рис. 21).

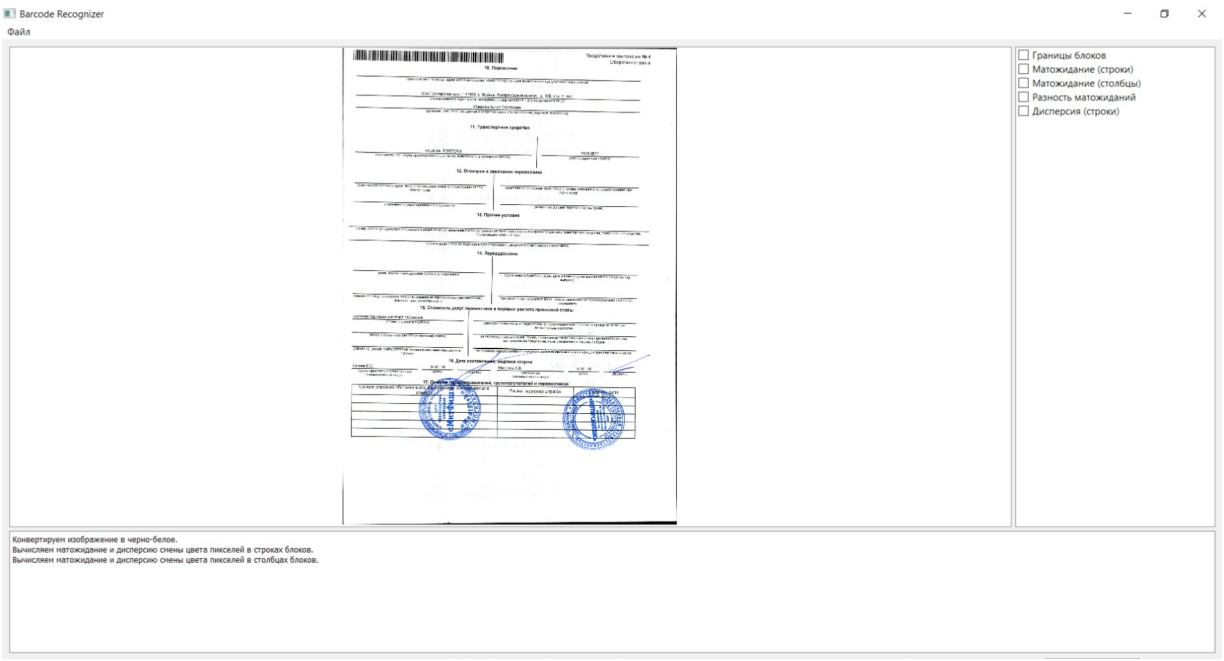

Рисунок 21: Вывод программы после вычислений

Для просмотра результатов вычислений пользователю необходимо отметить галочкой интересующие слои. (рис. 22)

Рисунок 22: Вывод слоя с дисперсией

При наложении слоев дисперсии, мат. ожидание (строки), разность мат. ожиданий будут не закрашены только блоки внутри штрихкода, что подтверждает эффективность применения статистического метода анализа фрагментов изображения. (рис. 23)

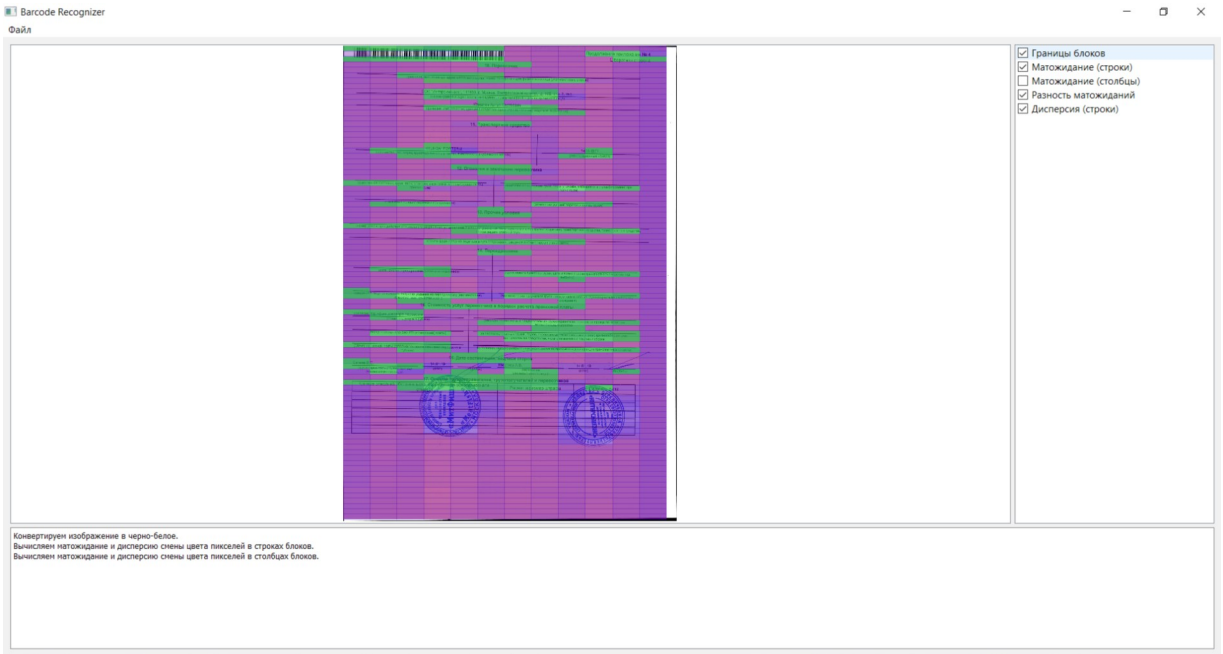

Рисунок 23: Вывод программы при наложении слоев мат. ожидания, дисперсии, разности мат. ожидания

Заключение

Предложенный мною статистический метод анализа фрагментов изображения достаточно точно и быстро работает при анализе сканированных изображений и может значительно ускорить архивацию документов в компании засчет повышения скорости распознавания штрихкода. Однако на некачественных фотографиях с зашумленными областями или неправильной экспозицией метод может давать сбои. Написанный демонстрационный прототип иллюстрирует предложенный метод и позволяет использовать его на разных изображениях.

Python, благодаря динамической типизации и простоте отладки, хорошо подходит для прототипирования, обеспечивая высокую скорость достижения результата. А библиотека Qt позволяет разрабатывать приложения с графическим интерфейсом легко и быстро.

Рассмотренный метод не чувствителен к незначительным поворотам изображения, что часто случается при сканировании документов.

Список справочной литературы

1) Википедия (https://ru.wikipedia.org/)
2) stackoverflow (https://stackoverflow.com/)
3) GitHab программа QTViewImage

   от автора Marcel Goldschen-Ohm <marcel.goldschen@gmail.com>
4) Qt Documentation (https://doc.qt.io/qt-5/index.html)
5) Учебник «Алгебра и начала математического анализа» для учащихся общеобразовательных учреждений (профильный уровень) 11 класса часть 1 (А. Г. Мордкович, П.В. Семенов)