Отчет по курсу

«Индустриальные распознающие системы»

Разработка системы распознавания двумерных штрих-кодов (этап нормализации)

Студентка: Позднякова А.А.

Группа: Б05-924

Преподаватель: Полевой Д.В.

Весна 2023г.

Содержание

Введение

Общую задачу распознавания двумерных штрих-кодов (Data Matrix и QR) на фотографии можно представить в виде последовательных шагов:

1. грубая локализация - выделение прямоугольной зоны на изображении, в которой расположен штрих-код;
2. точная локализация - нахождение точных границ штрих-кода на изображении;
3. нормализация – устранение искажений из-за дефектов поверхности и перспективы.;
4. декодирование - расшифровка данных штрих-кода.

Далее рассмотрено решение задачи нормализации.

Постановка задачи

Входные данные:

1. Цветное изображение (фотография), содержащее QR код или Data Matrix
2. Последовательность точек, задающих точную границу объекта

Выходные данные:

Цветное изображение, являющееся преобразованием исходного и содержащее нормализованное изображение штрих кода (без искажений из-за дефектов поверхности и перспективы)

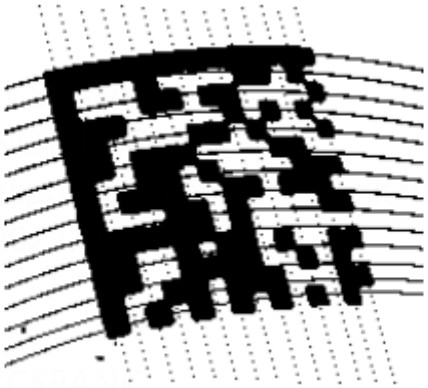
Способ решения задачи

Идея метода взята из статьи Lei Lei, Weiping He and Wei Zhang. Distortion Correction of Data Matrix Symbol Directly Marked on Cylinder Surface

Идея метода:

Рассмотрим штрих код, находящийся на цилиндрической поверхности.

1. Представим сетку, с одинаковыми квадратными ячейками на поверхности. Заметим, что по одной из осей линии сетки на фотографии будут параллельными прямыми.
2. Применим афинное преобразование, убирающее перспективное искажение. Параллельные прямые из п1 перейдут в вертикальные параллельные прямые.
3. Теперь пусть верхняя граница задается функцией f_1, а нижняя f_2. Тогда, поскольку деформаций по одной из осей нет, линии сетки будут иметь вид $f = f_1 + f_2$. Значит точка на QR коде с координатами (x, y), где y делит код по вертикали в отношении t к (1-t) на деформированно изображении находится в $(x, t*f_1(x) + t*f_2(x))$
4. Используя знания из п3, можем построить карту отображения в расправленное изображение, которое будет лучше декодироваться.



Сетка на поверхности datamatrix

Использование библиотеки

Проект лежит в github репозитории.

Зависимости:

```
numpy==1.21.5
matplotlib==3.7.1
opencv-python==4.7.0.72
python-json-logger==2.0.7
fastjsonschema==2.16.3
jsonpointer==2.3
jsonschema==4.17.3
scipy==1.8.0
```

Запуск из терминала (в корне директории):

```
python3 get_images.py <source_path>
```

Пример (для использования датасета):

```
python3 get_images.py data
```

Вход: папка с датасетом в формате COCO.segmentation

Выход: папка с преобразованными изображениями

В терминале: выводится статистика по результатам детектирования **OpenCV QRCodeDetector** на исходных и преобразованных данных

Результаты и оценка качества

Оценить успешность решения сложно, поскольку как показатель успеха стоит рассматривать улучшение способности некоторого декодера распознавать QR коды. Заметим, что декодеры бывают разные, а значит нельзя придумать метрику, которая будет отражать результаты для всех.

Улучшение работы декодера OpenCV QRCodeDetector.

(Рассматривается, поскольку был выбран в качестве используемого в проекте декодера в результате исследования)

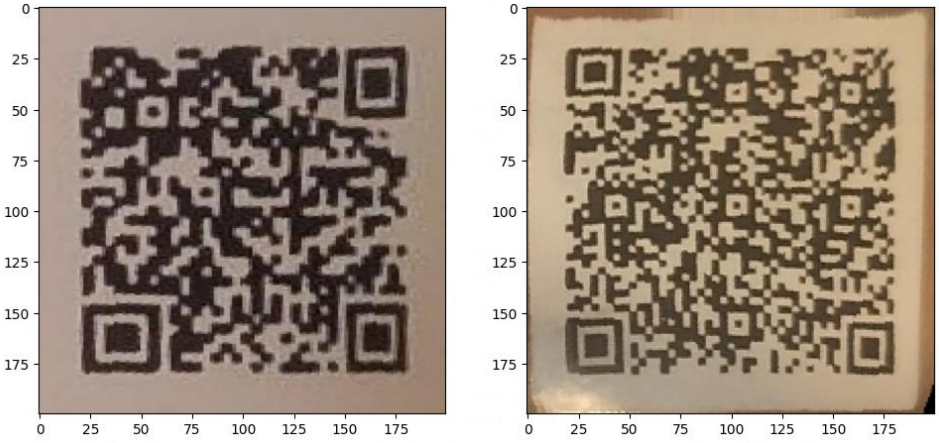
На датасете из 59 изображений сравнивается работа декодера из OpenCV на изначальных изображениях и полученных с помощью вышеописанного метода.

Результаты

Количество кодов	59
Количество распознанных без нормализации	11
Доля распознанных без нормализации	18.6%
Количество распознанных после нормализации	38
Доля распознанных без нормализации	64.4%
Количество ошибок после нормализации на кодах, распознаваемых без неё	1
Количество дополнительно распознанных кодов	28

Пример изображений, для которых декодер после нормализации отработал успешно, а до нормализации - нет

Изображения до применения метода



Изображения после применения метода

Изображения, не распознанные декодером и после нормализации

Заметим, что не изображения, которые распознать не удалось, визуально преобразовываются достаточно хорошо (сравнивая с изображениями выше).
Возможно, проблемы с распознаванием связаны с недостатками самого декодера.

Скорее всего, декодер чувствителен к засветкам, маленькой разнице между цветами фона и QR кода.

В качестве исключений можно выделить изображения с серьезными искажениями, как правило нанесённые на бумагу

Сильно искаженное изображение

Изображение на поверхности бумажного типа