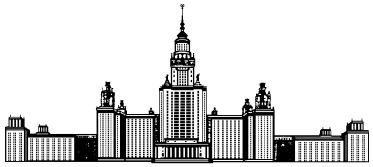
Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

Кафедра Математических Методов Прогнозирования

ОТЧЁТ

«Изучение и освоение методов обработки и сегментации изображений.»

Лабораторная работа №1 по курсу

«Обработка и распознавание изображений»

Выполнил:

студент 3 курса 317 группы

Зиннуров Артём Ринатович

Научный руководитель:

к.ф-м.н.

Китов Виктор Владимирович

Москва, 2022

Содержание

1 Постановка задачи

Требуется разработать и реализовать программу для работы с изображениями карточек игрового набора **Геометрика**.

Программа должна обеспечивать:

- ввод и отображение на экране изображений;
- сегментацию изображений на основе точечных и пространственных преобразований;
- поиск карточек на картинках;
- выделение и распознавание изображений фигур на карточках.

Реализованная программа решает следующие подзадачи на белом фоне:

1. Определяет количество карточек на изображении:
   (a) Карточки изолированные
   (b) Карточки накладываются друг на друга
2. Определяет фигуру на карточке:
   (a) Определить тип фигуры - многоугольник или фигура с гладкой границей
   (b) Для многоугольников определяет количество вершин
   (c) Определяет, является ли многоугольник выпуклым

2 Описание данных

Вход программы - изображения в формате **JPG**.

Выход программы – исходное изображение с нанесенной на нем разметкой результата маркерами PnC, Pn или S:
P – многоугольник, n – число вершин многоугольника, C – выпуклый многоугольник, S - гладкая фигура.

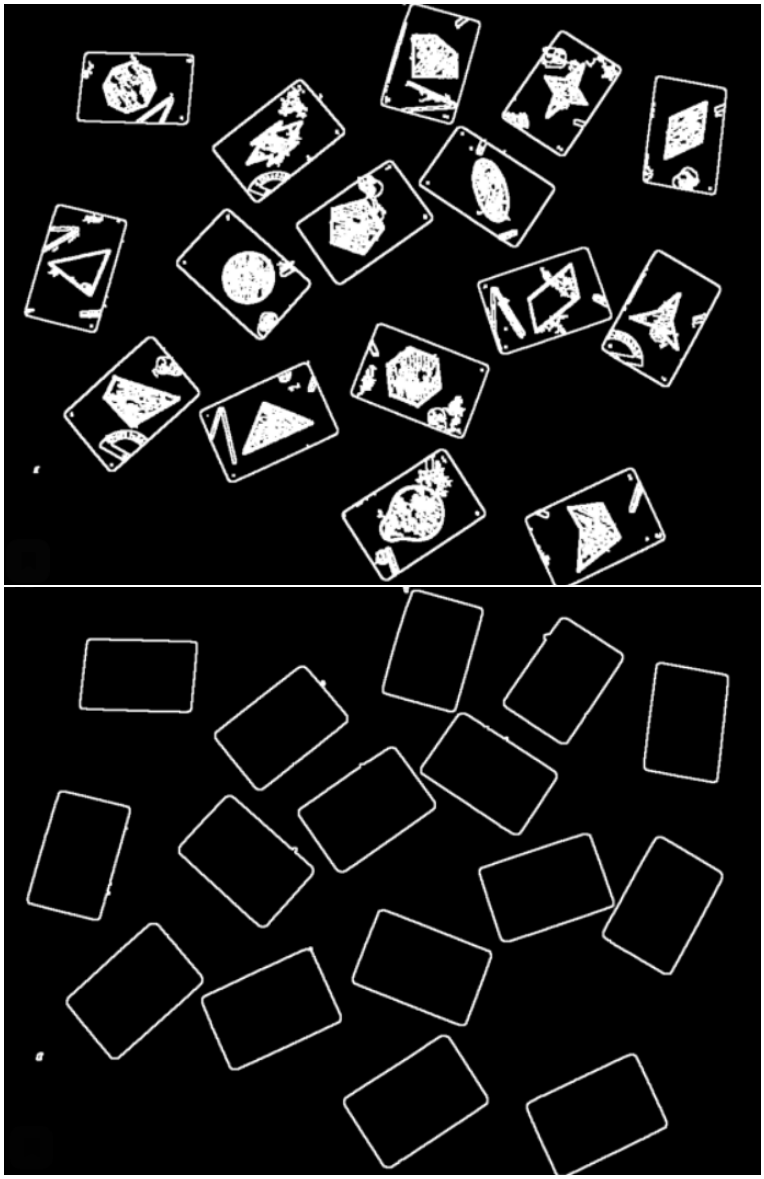
Рис. 1: Пример работы программы.

3 Описание метода решения

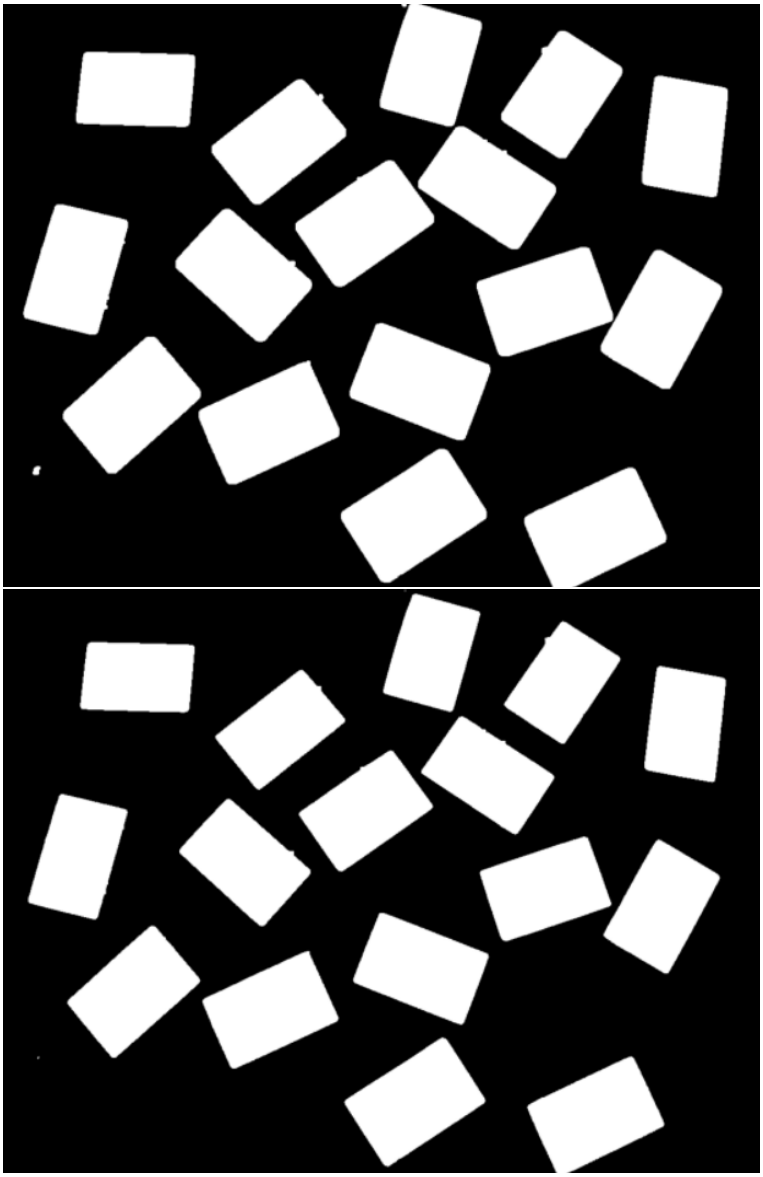
Данное решение опирается на тот факт, что фон изображений с белым фоном однотонный, без объектов, которые могли бы быть проинтерпретированы как границы различными градиентными методами выделения границ.

Изображение считывается и переводится в **градации серого**.

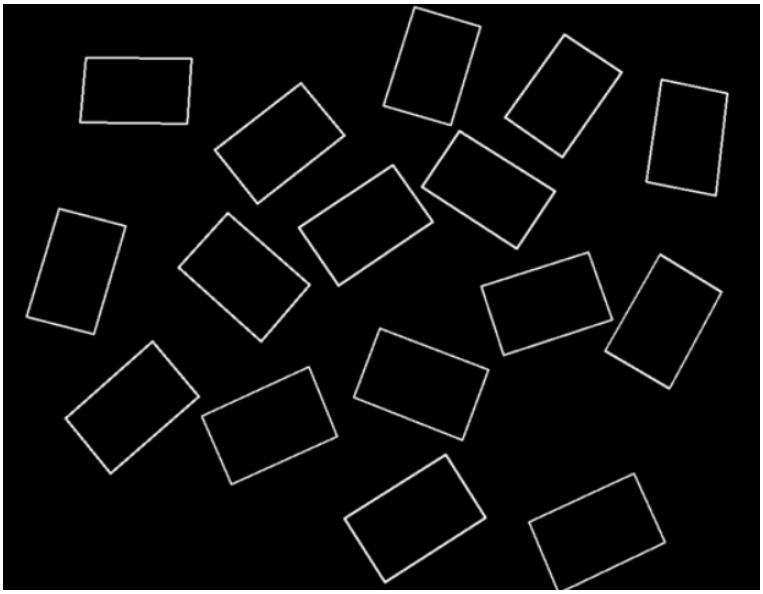
К получившемуся представлению применяется градиентный метод выделения границ **Canny**. Далее к чёрно-белому изображению применяется морфологическая операция **дилатации**, которая утолщает границы, соединяет возникающие пробелы и, затем, выделяются контуры, которые наносятся на чёрный фон белым цветом.



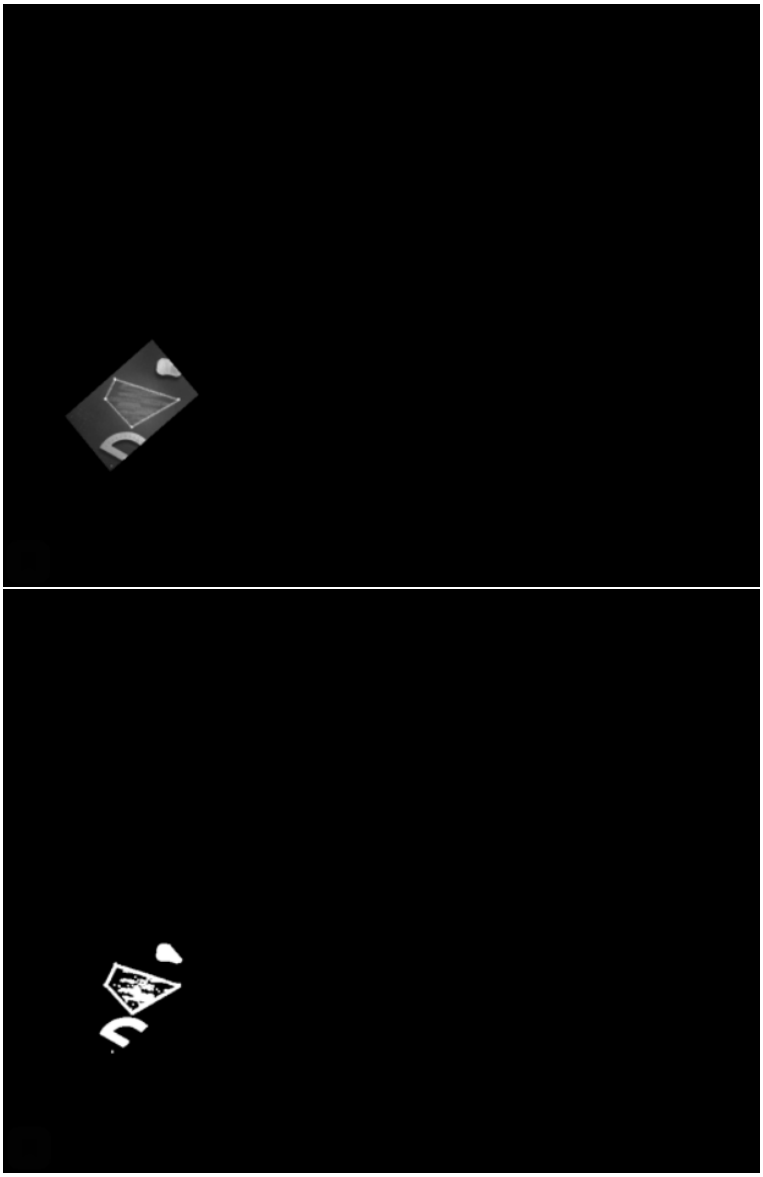
Это изображение заполняется цветом, выполняется **эрозия** для исключения границ карточек с фоном и производится повторное выделение контуров. Автором считается, что после такой обработки остаётся относительно небольшое число внешних контуров, среди которых, однако, по-прежнему много шумовых.



Следующим шагом является применение аппроксимации контура (сильного уменьшения количества вершин, образующих контур) с отбором получившихся контуров по занимаемой ими площади. Таким образом, в случае непересекающихся карточек, мы уже знаем их количество, а контур, им соответствующий, представляет собой четырёхугольник. При этом функция, реализующая данную функциональность, имеет возможность также определить случай пересекающихся карточек, сравнивая количество вершин контуров со значением 4. Так, если они не равны, то на данном изображении представлены пересекающиеся или близко расположенные карточки, и аппроксимация их контуров происходит с большей точностью для максимального сохранения информации.



Далее применяется некоторая эвристика: в дальнейшем мы также будем аппроксимировать контуры теперь уже фигур, поэтому важна точность, с которой это будет происходить. И точность должна быть тем выше, чем меньше размер карточек. Эвристика заключается в мультипликативной добавке отношения нового размера карточки к размеру, для которого некоторое значение показывает приемлемый результат. Далее в цикле по аппроксимированному контуру маскируются только отдельные карточки или связки из пересекающихся карт. Получившиеся вырезки приводятся в среднем к одинаковой яркости и применяется операция **бинаризации**, за которой следует **дилатация** для утолщения и связывания.



У получившегося изображения также ищутся контуры, они аппроксимируются, заливаются и ещё раз ищутся контуры и аппроксимируются получившееся изображение с фильтрацией по площади. Таким образом становится неважно сколько карточек будет наложено друг на друга. Главное - чтобы не перекрывались фигуры на них. Остаётся только вычислить место нанесения надписи, количество углов (количество вершин контура), проверить фигуру на **выпуклость** и нанести все полученные данные на исходное изображение.

4 Описание программной реализации

При решении данного задания используется библиотека компьютерного зрения **Open CV**:

- `cv.imread()` - считывает изображение;
- `cv.cvtColor()` - переводит из одного представления в другое (например в GrayScale);
- `cv.Canny()` - реализует градиентный алгоритм выделения границ Canny;

- `cv.morphologyEx()` - реализует морфологические операции (дилатация, эрозия, открытие, закрытие и др.);
- `cv.findContours()` - осуществляет поиск контуров изображения;
- `cv.drawContours()` - отображает контуры на изображении;
- `cv.fillPoly()` - заполняет замкнутые контуры;
- `cv.threshold()` - производит бинаризацию;
- `cv.isContourConvex()` - определяет, является ли данный контур выпуклой фигурой;
- `cv.putText()` - рисует текст на изображении;
- `cv.moments()` - вычисляет моменты изображения;
- `cv.arcLength()` - вычисляет периметр контура;
- `cv.contourArea()` - вычисляет площадь контура;
- `cv.approxPolyDP()` - производит аппроксимацию контура.

Собственно реализованные функции:

- `print_img(img)` - печатает изображение;
- `approx_contour(contours, alpha, max_card_side, thresh_area, no_area, use_overlay, use_smooth)` - аппроксимирует контур с заданной точностью с возможностью исключения контуров, площадь которых меньше указанного порога, а также с возможностью поиска пересечения карточек и гладких фигур;
- `mean_brightness(img, mask)` - вычисляет среднюю яркость изображения в пределах переданной маски;
- `change_brightness(img, mask)` - меняет среднюю яркость изображения в пределах переданной маски.

Для отображения используется библиотека **Matplotlib**.

5 Эксперименты

Проведены эксперименты на семи предложенных изображениях. Алгоритм не обрабатывает изображения с пёстрым фоном, поэтому они не будут входить в эксперименты.

Среди представленных изображений присутствуют два (IMG_6.jpg, IMG_7.jpg), карточки на которых получились размытыми, поэтому сегментация на таких изображениях получается хуже, чем на чётких.

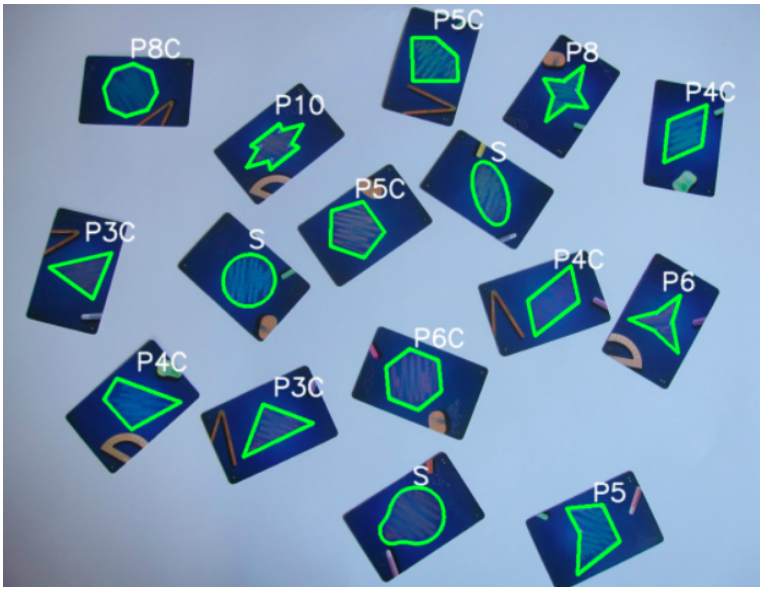


Рис. 2: Сегментация IMG_1.png

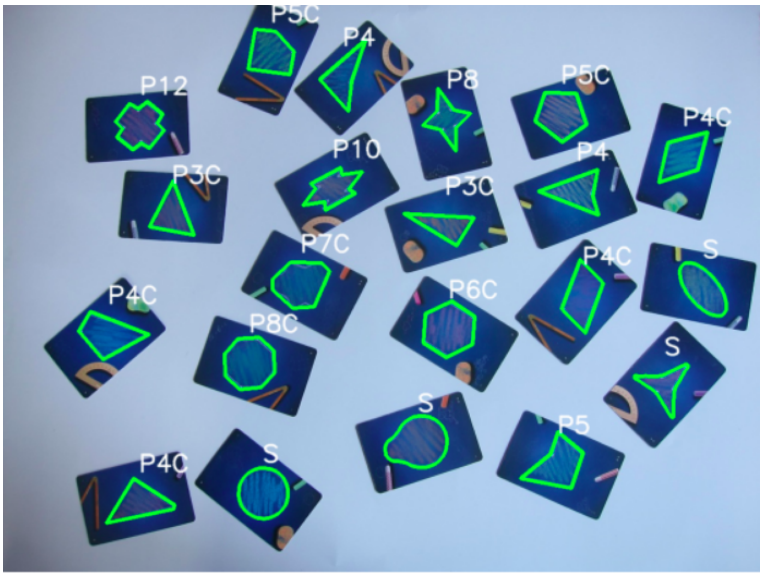


Рис. 3: Сегментация IMG_2.png

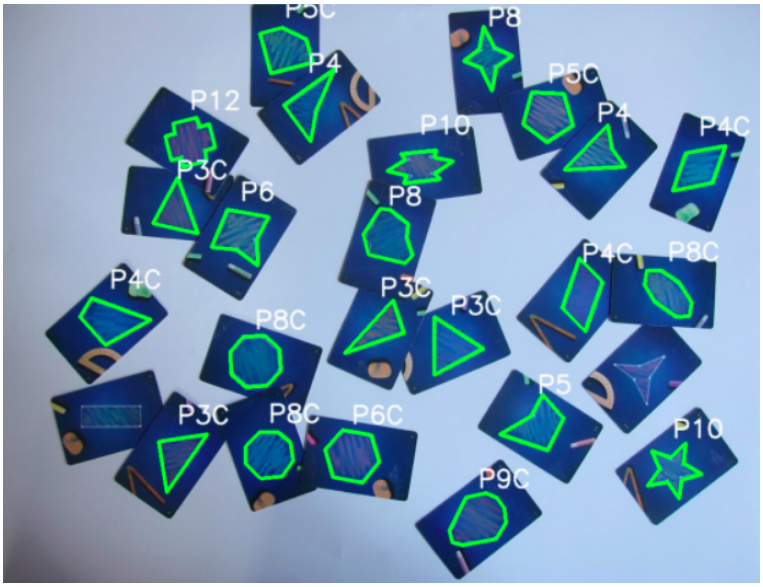


Рис. 4: Сегментация IMG_4.png

6 Выводы

Таким образом, в ходе выполнения данного задания изучены и освоены методы обработки и сегментации изображений, произведено знакомство с популярными библиотеками компьютерного зрения **OpenCV** и **Pillow**.