Бинарные изображения

Шокуров Антон В.
shokurov.anton.v@yandex.ru
http://машинноезрение.рф

1 октября 2018 г.

Версия: 0.09

Аннотация

Преобразование растровых изображений в бинарные. Пороговые функции. Связанные компоненты. Их подсчет.

1 Бинарные изображения

В данной заметке будет показано как преобразовывать изображения в бинарные. Подсчет, а точнее расстановка меток в изображение.

1.1 Бинаризация

Данное преобразование обозначает превращение обычного (цветного) изображения в бинарное, т.е. состоящее из двух значений пикселов: истина и лож. Числовое представление для данных сущностей может быть разным. Обычно всегда для лжи используется ноль, а вот для истины может быть по разному. Можно истину обозначать 1, а можно значением 255.

Простейший способ выполнения бинаризации заключается в преобразовании изображения в изображение с градациями серого цвета, с последующим применением пороговой функции.

Градации серого цвета Напомню, что считать изображения в градациях серого цвета можно как минимум двумя способами.

Можно сразу это указать при чтение изображения:

```
1 Mat gray;
2 gray = imread(argv[1], IMREAD_GRAYSCALE);
```

Такой способ полезен при считывании изображения из файла.

Если же изображение уже дано и оно цветное, то тогда его нужно будет преобразовать. Напомню, что это делается так:

```
1 //Mat -- это цветное изображение
2 Mat gray;
3 cvtColor( img, gray, COLOR_BGR2GRAY );
```

В обоих случаях мы в итоге получим изображение с градациями серого цвета.

Пороговая функция Далее необходимо изображение с градациями серого цвета превратить в бинарное. Для этого используется простейшее истинностное выражение, которое просто проверяет факт того что текущее значение больше на перед заданного. Данная функция называется пороговой.

В опенсв она вызывается следующем образом:

```
1 Mat bin;
2 threshold(gray, bin, 180, 255, THRESH_BINARY );
```

В данном случае, все что меньше значения 180 будет ложью, а что больше или равно 180, но меньше 255 будет истинной.

В некоторых случаях, когда на белом фоне нарисованы черные объекты, корректнее будет использовать ключ THRESH_BINARY_INV.

Если все сделано правильно, то объекты будут заполнены истинным значением, а фон ложью.

Поиск порогового значения Анализ гистограммы Otsu

1.2 Связанные компоненты

Входное изображение является бинарным. Считается, что фон заполнен ложным значением, а объекты истинным. Далее необходимо выделить на изображении связанные компоненты, как минимум научится считать их количество.

Связанность Понятие о 4х и 8 связанности. Считается, что фон 4ч связан, а объекты 8ми. Иначе будут противоречия.

Компоненты Для выделения на изображении связанных компоненты можно вызвать функцию из опенсв:

```
1 Mat labels;
2 int n = connectedComponents( bin, labels);
```

Данная функция возвращает количество найденных на бинарном изображении `bin` связанных компонент. На изображении `label` они будут отмечены путем заполнения компоненты значением равным метке.

Упражнение. Напишите собственную реализацию данной функции.

Упражнение. Как извлечь каждую связанную компоненты по отдельности? Так чтобы каждое изображение содержало свою компоненту: там где искомые пикселы было бы значение 255.

Простейшая характеризация Для каждой компоненты по отдельности можно вычислить некие числовые характеристики, которые её описывают. Например, площадь.

Можно было бы сразу вызвать функцию `connectedComponentsWithStats`, которая вернула бы для каждой связанной компоненты данные параметры. Для этого:

```
1 Mat labels;
2 Mat stats, centroids;
3 int n = connectedComponentsWithStats( bin, labels,
4     stats, centroids);
```

Аргументы `bin` и `labels` совпадают со стандартной функцией textttconnectedComponent
Аргументы `stats` и `cendroids` характеризуют каждую из компонент. Тип данных у `cendroids` является CV_64F (т.е. double), а у `stats` – CV_32S (т.е. int32_t).

Так, координаты центра iой компоненты (центройд) задаются, соответственно, как `centroids.at<double>(i, 0)`, `centroids.at<double>(i, 1)`.

Другие числовые характеристики хранятся в матрице `stats`. Ряд также указывает на номер компоненты, а колонка на извлекаемую информацию. Для удобства им в опенсв даны идентификаторы:

Идентификатор CC_STAT_AREA задает площадь данной компоненты, а CC_STAT
(диаметр?), т.е. максимальное расстояние между двумя точками. Идентификаторы CC_STAT_LEFT и CC_STAT_TOP задают левый верхний угол, а CC_STAT_WID
и CC_STAT_HEIGHT ширину и высоту описанного прямоугольника вокруг данной компоненты.

Например, для извлечения площади:

```
1 // Пусть i это номер компоненты.
```

```
cout << "area ~ " <<
    stats.at<int32_t>(i, CC_STAT_AREA) << endl;
```

Упражнение. Написать программу, которая выводит данные характеристики для всех связанных компонент.

Моменты Помимо площади можно было бы вычислить и другие подобные характеристики. В общем виде они называются моментами.

Для задания конкретной связанной компоненты необходимо сделать следующее (внимание, следующий код является решением одного из предыдущих упражнений):

```
// Пусть i это номер компоненты.
Mat obj = labels == i;
```

Теперь для вычисления моментов достаточно вызвать:

```
Moments mom = moments( obj, true);
```

Упражнение. Написать программу, которая выводит моменты для всех связанных областей.

Инварианты характеристик Площадь инварианта относительно поворота и перемещения связанной компоненты. В этом смысле она инвариантна. Для других компонент можно вычислить некий инвариант, который также не будет зависеть от сдвига и поворота.

Данные инварианты называются ХюМоментами. Они вычисляются по обычным моментами путем вызова функции.

```
// Пусть в mom моменты.
Mat hu;
HuMoments( m, hu);
```

Упражнение. Проверить свойство инвариантности. Вывести для объектов (квадрат, прямоугольник с фиксированным соотношением сторон, круг, эллипс с фиксированным соотношением сторон) моменты как стандартные, так и Хю моменты.

Упражнение. Проверить насколько численно устойчивы данные характеристики.

Сопоставление объектов Имея числовые характеристики инвариантные относительно тех или иных преобразований можно построить на их основе систему

распознавания объектов. Так, если числовые характеристики близки, то и объекты считать совпадающими.

Упражнение. Напишите программу использующая Хю инварианты для распознавания объектов. Например, подсчитывающая количество кружков, квадратов и их вариаций в изображении.