# Введение

Классификация объектов - это обширная область исследований в области компьютерного зрения и машинного обучения.

Решение проблемы категоризации автомобиля и его идентификации по регистрационному номерному знаку является важным аспектом безопасности и контроля. Использовать такой продукт можно в различных сферах применения, касающихся автотранспорта. Примером могут служить автотранспортные предприятия, заправочные станции, контроль скорости движения, автомобильные стоянки, контроль въезда на территорию предприятия и т.п.

В данном докладе будет рассмотрено распознавание регистрационного номерного знака транспорта.

# 3.2.1 Бинаризация изображения

Процедуры сегментации и распознавания работают с бинарным изображением, т.е только с черными и белыми пикселями. Поэтому прежде чем передавать работу этим про­цедурам, нужно исходное цветное изображение привести к бинарному виду. Эта задача решается в два этапа.

На первом этапе цветное изображение превращается в черно-белое и представляется в градациях серого . Для каждого пикселя вычисляется его яркость в пределах от нуля до 255. Уровню яркоццсти 0 соответствует черный цвет, уровню 255 - белый. Таким образом, для хранения изображения приходится один байт на пиксель. Яркость пикселя вычисляется по одной из следующих формул:

*I = R* + *G* + *B*

*I =* max( *R, G, B*) + min( *R, G, B*)

где *R, G, B* - нормированный на 256 (один байт) красный, зеленый и голубой компо­нент цвета пикселя соответственно.

Вторым этапом является собственно бинаризация. Результат бинаризации зависит от заранее заданного параметра - соотношения черных пикселей и общего их количества на изображении.

**Алгоритм бинаризации изображения**

1. Создаем одномерный массив *I* из 256 элементов (от 0 до 255). Заполняем его нулями.
2. Пробегаем попиксельно все изображение. Увеличиваем на единицу значение в ячейке массива *I*, соответствующей яркости пикселя *i (I[i]++)*. В итоге, значе­ние каждой ячейки массива *I[i]* будет равно количеству пикселей яркости уров­ня *i* на всем изображении.
3. На этом шаге определяется порог яркости *a*. Пусть *N* - общее количество пик­селей (высота умноженная на ширину), *k* - коэффициент, определяющий коли­чество черных пикселей. Тогда *kN* будет равно желаемому количеству черных пикселей на бинарном изображении. Суммируем значения ячеек массива, начи­ная с нулевой до тех пор, пока значение этой суммы не превысит *kN*. Индекс последней суммируемой ячейки и будет порогом *a*.
4. Повторно пробегаем попиксельно все изображение. Сравниваем уровень ярко­сти каждого пикселя с порогом *a*. Если этот уровень меньше или равен *a*, то пиксель становится черным, иначе - белым.  
     
   **Удаление обрамления**

Т.к. в подавляющем большинстве случаев большая часть самых темных пикселей на­ходится именно в обрамлении, то после бинаризации изображения они и становятся чер­ными. Символов номера становится практически не видно (рисунок 3.5). Эта проблема привела к необходимости данной процедуры.

Алгоритмы работают с бинарным изображением, но обрезают исходное. Первым де­лом удаляются черные вертикальные полосы слева и справа.

Затем обрезаем темные области сверху и снизу. Если на исходном изображении на номерной пластине есть тень (обычно это тень сверху от бампера, решетки радиатора,

«кенгурятника»), то часть символов, на которые падает тень, будет обрезана (рисунок 30). Эта проблема решается при распознавании, смещением эталона относительно символа на изображении.   
  
**Алгоритм обрезания вертикальной темной части изображения слева**

1. Начинаем с самой первой левой вертикальной линии пикселей.
2. Подсчитываем количество черных пикселей в линии.
3. Если это количество превышает 75% высоты изображения, то линия удаляется, текущей становится следующая полоса справа, переходим к шагу 2.

**Алгоритм обрезания горизонтальной темной части изображения сверху**

1. Исходной является линия пикселей, проходящая ровно через середину изобра­жения по горизонтали.
2. Подсчитываем количество черных пикселей в линии.
3. Если это количество превышает 75% ширины изображения, то это означает, что мы дошли до черного обрамления номерной пластины или тени. Часть изобра­жения выше этой линии, включая ее саму, обрезается, на этом алгоритм завер­шает свою работу.
4. Если мы не дошли до верхнего края изображения, то текущей становится сле­дующая линия по направлению вверх.

Обрезанное исходное изображение бинаризуется. Теперь символы становятся отчет­ливее, и появляется возможность найти горизонтальные линии, между которыми стоят символы и обрезать по ним изображение.

**Алгоритм поиска нижней границы символов**

1. Исходная горизонтальная линия пикселей проходит через середину изображе­ния.
2. Подсчитывается число слитных групп черных пикселей в линии *s*.
3. Начинаем с самого левого пикселя линии, *s* равно нулю.
4. Двигаемся влево, пока не встретим черный пиксель, *s* увеличиваем на единицу. При достижении правой границы изображения переходим на шаг 3.
5. Продолжаем движение влево до первого встретившегося белого пикселя, переходим на шаг b. Если достигнем правого края изображения, то пере­ходим на следующий шаг алгоритма
6. Если линия пересекает символы, то *s* будет порядка десяти, текущей становится следующая линия изображения по направлению вниз, переходим на шаг 2. Ина­че, алгоритм добрался до светлой полосы и *s* резко падает до 0-3, завершаем ра­боту алгоритма.

В итоге получаем изображение, готовое к обработке фильтром подчеркивания границ и сегментации.

Изображение подвергается фильтрации. Фильтр подчеркивания границ накладывается на каждый пиксель изображения, вычисляется сумма произведений значений матрицы и уровней яркости изображения.

Полученное число является яркостью соответствующего пикселя в новом изображе­нии (рисунок 3.1).

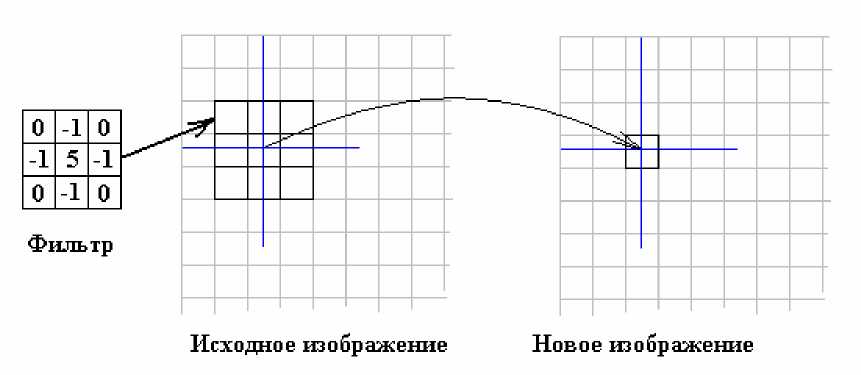


Рисунок 3.1 – Фильтрация

Профильтрованное изображение подвергается процедуре бинаризации и передается на сегментацию.

# 3.2.3 Сегментация символов

Сегментация проходит слева направо. Ищется точка, подозрительная на принадлеж­ность символу. Изображение сканируется вертикальными полосами. Имея ввиду вероят­ность тени, пропускаем точки, связанные с верхом и низом растра. Точка, не имеющая та­кой связи, передается алгоритму выделения символа (рисунок 3.2). Подробный алгоритм приведен ниже.



Рисунок 3.2. Поиск подозрительной точки

**Алгоритм поиска точки, подозрительной на принадлежность символу**

1. Стартовой становится самая левая вертикальная линия пикселей.
2. Начиная от верхнего пикселя, спускаемся вниз, пока не встретим белый пик­сель. Так мы получаем верхнюю границу области поиска.
3. Начиная от нижнего пикселя, поднимаемся вверх, пока не встретим белый пик­сель. Так мы получаем нижнюю границу области поиска.
4. В выделенной области поиска ищем первую попавшуюся черную точку. Если таковой не найдется, то текущей становится следующая линия по направлению вправо. Переходим на шаг 2.
5. Найденная точка передается алгоритму выделения символа.

От выбранной точки строится четырехсвязная область. Таким образом, выделяется символ. Накладываются ограничения на ширину области, учитывая возможные сливания символов тенью и грязью.

**Рекурсивный алгоритм выделения символа**

1. Точка помечается как принадлежащая символу.
2. Если верхняя соседняя точка является черной, то переход на шаг 1.
3. Если нижняя соседняя точка является черной, то переход на шаг 1.
4. Определяем возможность рассмотрения соседних точек слева и справа. Для это­го исследуется вертикальная линия пикселей, в которой находится текущая точ­ка.
5. Выход из рекурсии.

В итоге получаем набор обрамлений, не все из которых выделяют символ. Мелкие обрамления удаляются(Рисунок 3.3.).



Рисунок 3.3. Результат работы сегментации

Для этого вычисляется средняя ширина символов. Первый и последний обрезаются соответственно справа и слева, т.к. могут сливаться с тенью или грязью. Остальные сим­волы расширяются или сужаются до нужной ширины. В заключение, происходит вырав­нивание по высоте. В итоге получаем набор символов для распознавания.

# 3.2.3 Распознавание символов шаблонным методом

Из положения символа на номере становится ясно буква это или цифра. Поэтому для символа, из сравнения его растра с эталоном, получаем 10 или 13 оценок для цифры или буквы соответственно. Принимаем решение в пользу символа с максимальной оценкой.

Эталоны символов хранятся в матрицах. Черный пиксель отмечается единицей. Фоно­вые пикселям соответствуют ячейки с записанными в них расстояниями до ближайшего черного пикселя со знаком минус. Расстояние считается по одной из следующих формул.

*d* = max(*x*, *y*) , *d* = *x* + *y* ,

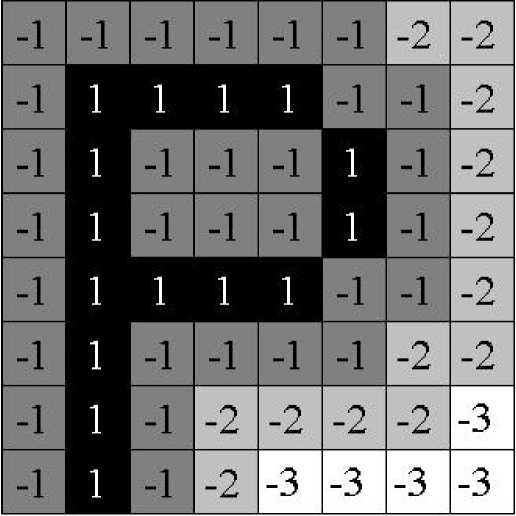
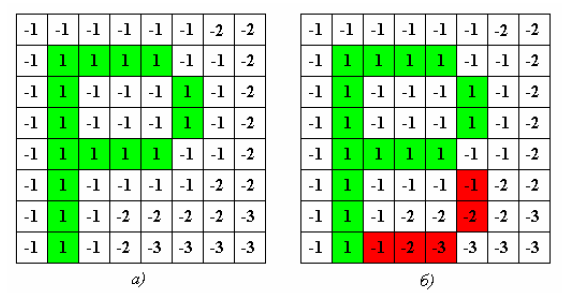
где *x, y* - модули смещений до ближайшей черной точки.

Рисунок 3.4 - Пример эталона буквы «Р»



а) *6)*

Рисунок 3.5 - а) оценка для символа «P» равна 15, б) оценка для символа «B» равна 6

Вычисление оценки происходит следующим образом. Растр символа трансформиру­ется до размеров эталона, и они накладываются друг на друга. Соответствующие черным пикселям символа значения в ячейках эталона суммируются. Появление точки в ячейке символа, далекой от ближайшей черной точки в эталоне, существенно снижает значение оценки.

Алгоритм сегментации не всегда выделяет символ минимальным охватывающим пря­моугольником. Поэтому при наложении растров возникает необходимость смещать эталон относительно растра символа. Оценка вычисляется для исходного положения эталона на растре и для всех случаев смещения эталона относительно растра символа в восьми на­правлениях на три пикселя.

Из положения, соответствующего максимальной полученной оценке, производится аналогичная процедура, но со сдвигом в один пиксель. В итоге максимальный сдвиг эта­лона относительно символа достигает четырех пикселей, что покрывает ошибки сегмента­ции.

В ходе работы были разработаны алгоритмы и на их основании реализованы методы предварительной обработки изображения автомобильного номерного знака, сегментации и распознавания символов.

# Список использованных источников

1. ГОСТ 50577-93. Знаки государственные регистрационные транспортных средств типы и основные размеры. Технические требования. Введен 01.01.94. - М.: Изда­тельство стандартов. - 33 стр.
2. Арлазаров В. Л., Троянкер В.В., Котович Н.В. Адаптивное распознавание симво­лов. [Электронный ресурс]. - <http://www.ocrai.narod.ru/adaptive.html>.
3. Лукошенко Г. Н. Распознавание скелетных образов. [Электронный ресурс]. - <http://www.ocrai.narod.ru/skeletrecognize.html>
4. Саймон Хайкин. Нейронные сети. Полный курс. - М.: Вильямс, 2005. - 1104 стр
5. Хорн Б.К.П. Зрение роботов. – М.: МИР, 1989.
6. Warren S. McCulloch and Walter Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. 1943.
7. Paul J. Werbos. Backpropagation Through Time: What It Does and How to Do It. 1990.
8. J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. “Man vs. computer: Bench marking machine learning algorithms for traffic sign recognition”, 2012.
9. Y. LeCun and M.A. Ranzato, ICML 2013 tutorial.
10. Библиотека машинного обучения Theano: https://github.com/Theano/