**Введение**

В современном мире накопились большие объемы рукописных и печатных книг, не имеющих цифровых аналогов. А так как мы живем в мире, где большинство процессов представляют из себя автоматизированные системы, то важным компонентом нашего существования является обеспечение целостности связи между поколениями в виде сохранности существующей информации. У людей мало времени и сил на копирование вручную. Поэтому сейчас как можно скорее необходимо задуматься об оцифровке, т.е. о переводе научно-технической, художественной литературы, архивных документов и других бумажных носителей информации в электронный вид. Электронная версия информации – это гарантия сохранности, несмотря на возможные бедствия. Электронные копии книг могут образовывать [электронные библиотеки](https://ru.wikipedia.org/wiki/Электронные_библиотеки) и распространяться в глобальной сети Интернет. Цифровые книги можно легко распространять, воспроизводить и читать на экране. Во всем этом помогают новомодные гаджеты, которые стали неотъемлемой частью нашей жизни. Большой функционал, мобильность, аккуратность и многое другое делают эти устройства лидерами рынка. С их помощью люди экономят время на поиск необходимой информации в библиотеках и архивах, соответственно, усваивают быстрее и больше нужного материала.

На сегодняшний момент времени существует методика оцифровки текста, но она, к сожалению, имеет свои недостатки. Процесс оцифровки включает в себя два подхода.

1. Обязательный: получение копий страниц в виде графических изображений, путем сканирования или фотографирования с последующей обработкой и сохранение в одном из форматов графических файлов. При этом оригинальная верстка книги сохраняется, исключаются разного рода ошибки.
2. Опциональный: применение технологии «оптического распознавания символов – OCR» с последующим сохранением распознанного текста в одном из форматов электронных книг. Открывается возможность полнотекстового поиска по книге и индексация больших массивов электронных книг, но затрудняется воспроизведение оригинальной верстки, изображений, схем и формул.

Тема оцифровки информации с бумажных носителей является невероятно актуальной. Доставая том книги с полки, мы даже не задумываемся, что держим в руках не просто вещь, а результат многовековой эволюции человеческой культуры. С древнейших времен человек стремился оставить за собой след. Первыми носителями информации стали стены пещер и камни. Создавались наскальные рисунки и петроглифы. Далее были придуманы глиняные и восковые таблички. И только после появления папируса была изобретена бумага. Теперь же главной целью у человечества должно быть формирование электронного фонда цифровой информации.

В соответствии с вышеизложенным, **цель работы** –

Объект исследования –

Предмет исследования –

Основные задачи, намеченные для достижения цели:

**Глава 1. Технологии Оптического распознавания символов.**

Оптическое распознавание символов (англ. optical character recognition, OCR) — механический или электронный перевод изображений рукописного, машинописного или печатного текста в текстовые данные, использующиеся для представления символов в компьютере (например, в текстовом редакторе). Распознавание широко применяется для преобразования книг и документов в электронный вид, для автоматизации систем учёта в бизнесе или для публикации текста на веб-странице.

Существует несколько технологий, позволяющих реализовать данную задачу. Каждая имеет свои недостатки и преимущества. Далее мы одну за другой опишем из всех.

**Сопоставление изображений и шаблонов**

Эта группа методов основана на непосредственном сравнении изображений тестового и эталонного символов. При этом вычисляется степень сходства между образом и каждым из эталонов. Классификация тестируемого изображения символа происходит по методу ближайшего соседа. С практической точки зрения эти методы легко реализовать, и многие коммерческие системы OCR используют именно их. Однако при "лобовой" реализации корреляционных методов даже небольшое темное пятнышко, попавшее на внешний контур символа, может существенно повлиять на результат распознавания. Поэтому для достижения хорошего качества распознавания в системах, использующих сопоставление шаблонов, применяются другие, специальные способы сравнения изображений. Одна из основных модификаций алгоритма сравнения шаблонов использует представление шаблонов в виде набора логических правил.

**Статистические характеристики**

В данной группе методов выделение признаков осуществляется на основе анализа различных статистических распределений точек. Наиболее известные методики этой группы используют вычисление моментов и подсчет пересечений.

Моменты различных порядков с успехом используются в самых различных областях машинного зрения в качестве дескрипторов формы выделенных областей и объектов. В случае распознавания текстовых символов в качестве набора признаков используют значения моментов совокупности "черных" точек относительно некоторого выбранного центра. Наиболее общеупотребительными в приложениях такого рода являются построчные, центральные и нормированные моменты.

Для цифрового изображения, хранящегося в двумерном массиве, построчные моменты являются функциями координат каждой точки изображения следующего вида:

где p, q ∈ {0,1,…,∞}; MM и NN являются размерами изображения по горизонтали и вертикали и f(x, y) является яркостью пиксела в точке ⟨x, y⟩ на изображении.

Центральные моменты являются функцией расстояния точки от центра тяжести символа:

где x и y "с чертой" - координаты центра тяжести.

Нормированные центральные моменты получаются в результате деления центральных моментов на моменты нулевого порядка.

Следует отметить, что строковые моменты, как правило, обеспечивают более низкий уровень распознавания. Центральные и нормированные моменты более предпочтительны вследствие их большей инвариантности к преобразованиям изображений.

В методе пересечений признаки формируются путем подсчета того, сколько раз и каким образом произошло пересечение изображения символа с выбранными прямыми, проводимыми под определенными углами. Этот метод часто используется в коммерческих системах благодаря тому, что он инвариантен к дисторсии и небольшим стилистическим вариациям написания символов, а также обладает достаточно высокой скоростью и не требует высоких вычислительных затрат. На рис. 1 показано эталонное изображение символа R, система секущих прямых, а также вектор расстояний до эталонных векторов. На рис. 2 представлен пример реального изображения

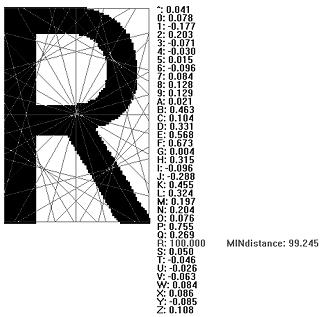
[](http://wiki.technicalvision.ru/index.php/%D0%A4%D0%B0%D0%B9%D0%BB:7-2-1.jpg)

Рисунок 1. Пример формирования набора пересечений для эталонного изображения символа R

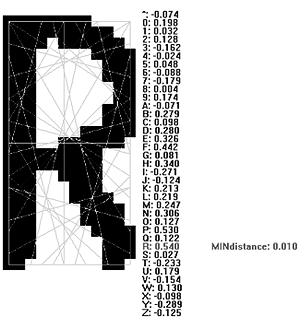
[](http://wiki.technicalvision.ru/index.php/%D0%A4%D0%B0%D0%B9%D0%BB:7-2-2.jpg)

Рисунок 2. Пример формирования набора пересечений для реального изображения символа R

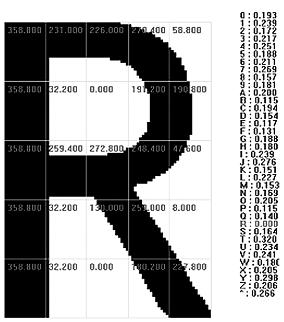
[](http://wiki.technicalvision.ru/index.php/%D0%A4%D0%B0%D0%B9%D0%BB:7-2-3.jpg)

Рисунок 3. Пример формирования зонного описания для эталонного изображения символа R

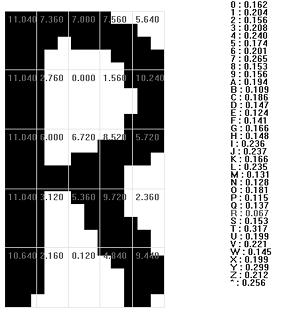
[](http://wiki.technicalvision.ru/index.php/%D0%A4%D0%B0%D0%B9%D0%BB:7-2-4.jpg)

Рисунок 4. Пример формирования зонного описания для реального изображения символа R

Метод зон предполагает разделение площади рамки, объемлющий символ, на области и последующее использование плотностей точек в различных областях в качестве набора характерных признаков. На рис. 3 показано эталонное изображение символа RR, а на рис. 4 - реальное изображение символа RR, полученное путем сканирования изображения документа. На обоих изображениях приводятся разбиение на зоны, пиксельные веса каждой зоны, а также вектор расстояний до эталонных векторов эталонных символов. Цветом помечена строка, соответствующая найденному ближайшему соседу.

В методе матриц смежности в качестве признаков рассматриваются частоты совместной встречаемости "черных" и "белых" элементов в различных геометрических комбинациях. Метод характеристических мест (characteristic-loci) использует в качестве признака число раз, которое вертикальный и горизонтальный векторы пересекают отрезки линий для каждой светлой точки в области фона символа.

Существует также множество других методов данной группы.

**Интегральные преобразования**

Среди современных технологий распознавания, основанных на преобразованиях, выделяются методы, использующие Фурье-дескрипторы символов, а также частотные дескрипторы границ. Преимущества методов, использующих преобразования Фурье - Меллина, связаны с тем, что они обладают инвариантностью к масштабированию, вращению и сдвигу символа. Основной недостаток этих методов заключается в нечувствительности к резким скачкам яркости на границах, к примеру, по спектру пространственных частот сложно отличить символ "O" от символа "Q" и т. п. В то же время, при фильтрации шума на границах символа, это свойство может оказаться полезным.

**Анализ структурных составляющих**

Структурные признаки обычно используются для выделения общей структуры образа. Они описывают геометрические и топологические свойства символа. Проще всего представить идею структурного распознавания символа текста применительно к задаче автоматического считывания почтовых индексов. В таких "трафаретных" шрифтах положение каждого возможного отрезка-штриха заранее известно, и один символ отличается от другого не менее чем наличием или отсутствием целого штриха. Аналогичная задача возникает и в случае контроля простых жидкокристаллических индикаторов. В таких системах выделение структурных составляющих сводится к анализу элементов заранее известного трафарета (набора отрезков, подлежащих обнаружению).

В системах структурного распознавания более сложных шрифтов часто используемыми признаками также являются штрихи, применяемые для определения следующих характерных особенностей изображения: концевых точек, точек пересечения отрезков, замкнутых циклов, а также их положения относительно рамки, объемлющей символ. Рассмотрим, например, следующий способ структурного описания символа. Пусть матрица, содержащая утонченный символ, разделена на девять прямоугольных областей (в виде сетки 3333), каждой из которых присвоен буквенный код от "A" до "I". Символ рассматривается как набор штрихов. При этом штрих, соединяющий некоторые две точки в начертании символа, может являться линией (L) или кривой (C). Штрих считается отрезком (дугой) кривой, если его точки удовлетворяют следующему выражению

в противном случае считается, что это прямолинейный отрезок. В данной формуле  является точкой, принадлежащей штриху;  - уравнение прямой, проходящей через концы штриха, коэффициент 0,69 получен опытным путем. Далее символ может быть описан набором своих отрезков и дуг.

Основное достоинство структурных методов распознавания определятся их устойчивостью к сдвигу, масштабированию и повороту символа на небольшой угол, а также - к возможным дисторсиям и различным стилевым вариациям и небольшим искажениям шрифтов.

**Классификация символов**

В существующих системах OCR используются разнообразные алгоритмы классификации, то есть отнесения признаков к различным классам. Они существенно различаются в зависимости от принятых наборов признаков и применяемой по отношению к ним стратегии классификации.

Для признаковой классификации символов необходимо, в первую очередь, сформировать набор эталонных векторов признаков по каждому из распознаваемых символов. Для этого на стадии обучения оператор или разработчик вводит в систему OCR большое количество образцов начертания символов, сопровождаемых указанием значения символа. Для каждого образца система выделяет признаки и сохраняет их в виде соответствующего вектора признаков. Набор векторов признаков, описывающих символ, называется классом, или кластером.

В процессе эксплуатации системы OCR может появиться необходимость расширить сформированную ранее базу знаний. В связи с этим некоторые системы обладают возможностью дообучения в реальном режиме времени.

Задачей собственно процедуры классификации или распознавания, выполняемой в момент предъявления системе тестового изображения символа, является определение того, к какому из ранее сформированных классов принадлежит вектор признаков, полученный для данного символа. Алгоритмы классификации основаны на определении степени близости набора признаков рассматриваемого символа к каждому из классов. Правдоподобие получаемого результата зависит от выбранной метрики пространства признаков. Наиболее известной метрикой признакового пространства является традиционное Евклидово расстояние

где  - -й признак из -го эталонного вектора;  - -й признак тестируемого изображения символа.

При классификации по методу ближайшего соседа символ будет отнесен к классу, вектор признаков которого наиболее близок к вектору признаков тестируемого символа. Следует учитывать, что затраты на вычисления в таких системах возрастают с увеличением количества используемых признаков и классов.

Одна из методик, позволяющих улучшить метрику сходства, основана на статистическом анализе эталонного набора признаков. При этом в процессе классификации более надежным признакам отдается больший приоритет:

где  - вес -го признака.

Другая методика классификации, требующая знания априорной информации о вероятностной модели текста, основана на использовании формулы Байеса. Из правила Байеса следует, что рассматриваемый вектор признаков принадлежит классу "", если отношение правдоподобия  больше, чем отношение априорной вероятности класса  к априорной вероятности класса .

**Глава 2. Способ реализации алгоритма распознавания символов.**

Итак, перед нами стоит задача распознать текст с изображения. Для обеспечения масштабируемости и читабельности имеет смысл разбить основной процесс на подзадачи, и реализовать каждую по-отдельности.

Исходя из нашей задачи можно выделить следующие этапы:

1. Перевод изображения в монохромный формат
2. Сегментация
3. Выравнивание
4. Распознавание

Разберем каждый из них.

**Перевод изображения в монохромный формат**

В основе этого этапа будет алгоритм, который анализирует каждый пиксель и перезаписывает его в одно из двух состояния: белый и черный (0 и 1 соответственно).

Разберем этот механизм подробнее. Известно, что стандартные форматы изображений, такие как “PNG” И “JPG” описывают каждый пиксель тремя каналами: красным, зеленым и синим. Каждый канал в свою очередь задается значением от 0 до 255. Это значение определяет меру присутствия соответствующего цвета в конечном цвете. Так, чтобы описать светлый оттенок, нам нужно установить как можно большее значение каждому каналу. Чистый белый цвет получается из значений 255, 255, 255 красного, зеленого и синего соответственно (R, G, B), а черный из значений 0, 0, 0. Пример цвета со значениями каналов приведены на рис. 1.

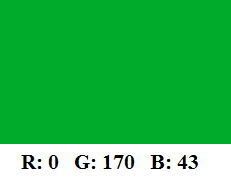


Рисунок 6. Пример цвета

Контуры символов соответствуют темным оттенкам, исходя из чего мы можем устанавливать каждому пикселю черный цвет (0,0,0), если значение каждого его канала будет меньше какого-либо заранее установленного предела и белый (255, 255, 255) в противном случае. Предел определяет границу между черным и белым и служит регулятором “чувствительности” алгоритма. Значение этого предела лежит в промежутке от 0 до 255. Целесообразней установить его в середине этого промежутка, но точное его значение определяется опытным путем при тестировании программы.

**Сегментация**

По итогу предыдущего алгоритма мы получим матрицу из нулей и единиц, описывающих черные и белые цвета соответственно. Перевод изображения в монохромный формат нужен прежде всего для корректной работы последующих этапов программы, одним из которых является сегментация.

Смысл этого этапа заключает в себя обрамление каждого символа и последующего сохранения этой информации в массив. Разберем этот процесс подробнее.

Одной из реализаций данного алгоритма является “раздувание” рамки при обнаружении закрашенных пикселей. Абстрагируемся от символов в пользу произвольных объектов и представим робота, который бегает по матрице и проверяет значение каждого пикселя. Обнаружив закрашенный, робот строит рамку вокруг этого пикселя (рис 2.).

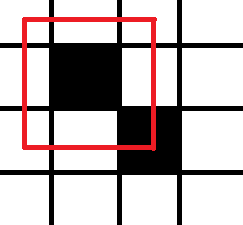


Рисунок 7. Построение рамки вокруг найденного пикселя

Далее робот проходит по периметру текущей рамки и анализирует каждый пиксель. Если на границе окажется закрашенный пиксель, робот расширит рамку, чтобы этот пиксель оказался внутри. Затем цикл повторяется, и так, пока на рамке не окажется закрашенных пикселей. В итоге для предыдущего примера картина окажется следующей (рис 3).

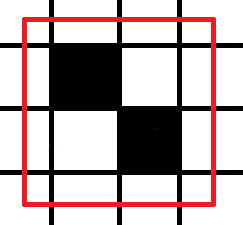


Рисунок 8. Результат работы алгоритма

Таким образом, каждый объект на изображении будет обрамлен. Вот живой пример заключения в рамки нескольких объектов на изображении (рис 4.)

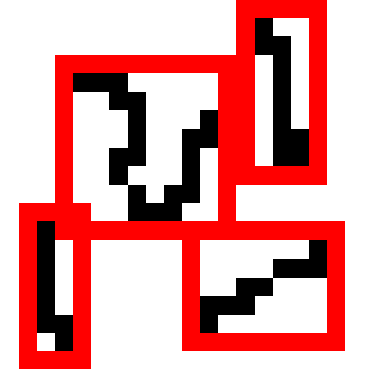


Рисунок 9. Пример работы алгоритма

Нам останется лишь сохранить информацию о каждом прямоугольнике, а именно позицию по ширине, позицию по высоте, ширину и высоту прямоугольника, в массив.

**Выравнивание**

В результате работы предыдущего этапа мы получаем карту изображения, которая описывает местоположение каждого объекта. Но давайте вернемся от объектов к нашей первоначальной задаче: символам. Тут нас ожидает проблема. Дело в том, что робот из предыдущего алгоритма, выполняя свою задачу, спускается вниз строка за строкой. И так как символы имеют разные размеры по высоте, то распознавание будет вестись хаотично, а если конкретней, в порядке уменьшения высоты символов. В результате ответ программы будет содержать набор букв, никак не связанный со словами на исходном изображении (рис 5.).

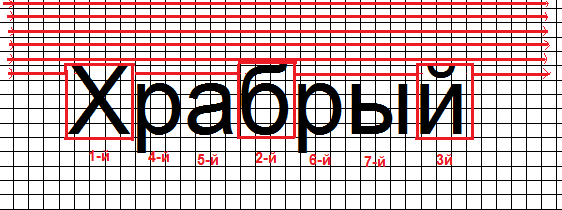


Рисунок 10. Результат программы без выравнивания

Так, для примера на рисунке 5 результатом программы будет “Хбйрары”, что, конечно же, неправильно.

Чтобы решить эту проблему нужно сделать выравнивание сначала построчно, затем посимвольно для каждой строки. Таким образом этот этап разбивается на две подзадачи. Разберем каждый по-отдельности.

Выравнивание по строкам можно реализовать путем сравнения уровней высоты двух идущих друг за другом букв. Рисунок 6 иллюстрирует данную методику.

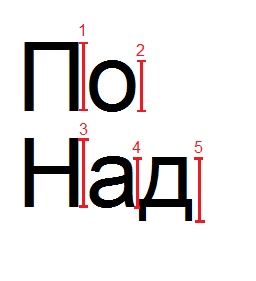


Рисунок 11. Способ выравнивания по строкам

Символы 1 и 2 располагаются на одном уровне, поэтому относим их к одной строке. А вот 2 и 3 находятся на разных уровнях, от чего мы образно переносим символ 3 на следующую строку и продолжаем сравнивать уровни следующих символов. В результате для примера из рисунка 6 мы получим две строки. Для структуризации всей собранной информации имеет смысл хранить контейнеры – строки, в которых мы и будем помещать символы с соответствующими номерами строк.

Когда первый под-этап отработал, и мы имеем структуру из строк нам не составит труда отсортировать символы по горизонтали в каждой строке. Для этого мы можем воспользоваться методом пузырька.

Алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов. Проходы по массиву повторяются {\displaystyle N-1} раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает — массив отсортирован. При каждом проходе алгоритма по внутреннему циклу, очередной наибольший элемент массива ставится на своё место в конце массива рядом с предыдущим «наибольшим элементом», а наименьший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции, как пузырёк в воде — отсюда и название алгоритма).

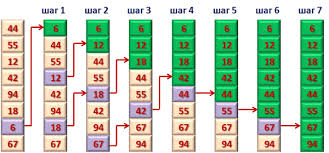


Рисунок 12. Сортировка методом пузырька

В нашем случае объектами сравнения будут позиции символов по горизонтали. По итогу этого этапа мы получим структуру строк и включенных в них в первозданном порядке символов.

**Распознавание**

Теперь, когда все готово, мы можем приступить к основному процессу, а именно к распознаванию каждого символа. Есть множество технологий, которые позволяют осуществить эту задачу.