СОДЕРЖАНИЕ

[1. Введение 3](#_Toc419827992)

[2. Назначение и область применения 4](#_Toc419827993)

[2.1. Назначение программы 4](#_Toc419827994)

[2.2. Краткая характеристика области применения программы 4](#_Toc419827995)

[3. Технические характеристики 5](#_Toc419827996)

[3.1. Постановка задачи на разработку программы 5](#_Toc419827997)

[3.2. Описание алгоритма функционирования программы с обоснованием выбора схемы алгоритма решения задачи 5](#_Toc419827998)

[3.2.1. Блок предобработки изображения 5](#_Toc419827999)

[3.2.2. Блок сегментации изображения 9](#_Toc419828000)

[3.2.3. Блок выделения признаков 11](#_Toc419828001)

[3.2.4. Блок распознавания символов 11](#_Toc419828002)

[3.2.5. Блок постобработки результатов распознавания 12](#_Toc419828003)

[3.3. Описание и обоснование метода организации входных и выходных данных 12](#_Toc419828004)

[3.4. Описание и обоснование метода выбора технических и программных средств 12](#_Toc419828005)

[4. Ожидаемые технико-экономические показатели 13](#_Toc419828007)

[5. Список использованных источников 14](#_Toc419828008)

[Приложение 1 Описание и функциональное назначение классов 15](#_Toc419828009)

[Приложение 2 Описание и функциональное назначение методов, полей и свойств](#_Toc419828011) 16

[Приложение 3 Диаграммы UML 28](#_Toc419828013)

# ВВЕДЕНИЕ

Наименование программы: «Программа обнаружения и распознавания текста в цифровых изображениях».

Основание для разработки: приказ НИУ ВШЭ № \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ от \_\_\_\_\_\_\_\_\_\_\_\_ «О назначении тем курсовых работ и научных руководителей». Программа выполнена в рамках темы курсовой работы: «Программа обнаружения и распознавания текста в цифровых изображениях» (факультет компьютерных наук, департамент программной инженерии), в соответствии с учебным планом подготовки бакалавров по направлению 09.03.04 «Программная инженерия».

# 2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

# 2.1. Назначение программы

Основным назначением программы является решение задачи распознавания текста в цифровых изображениях. Программа выполняет оптическое распознавание символов в цифровых изображениях, полученных в результате сканирования печатных документов.

# 2.2. Краткая характеристика области применения программы

Данная программа предназначена для решения задачи оптического распознавания символов, которая является одной из наиболее актуальных задач компьютерного зрения. Программа может применяться для автоматизации документооборота, автоматизированной обработки больших объемов печатных документов, а также внедрения безбумажных технологий.

# 3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

# 3.1. Постановка задачи на разработку программы

Написать программу, выполняющую оптическое распознавание символов предварительно отсканированного изображения печатного текста.

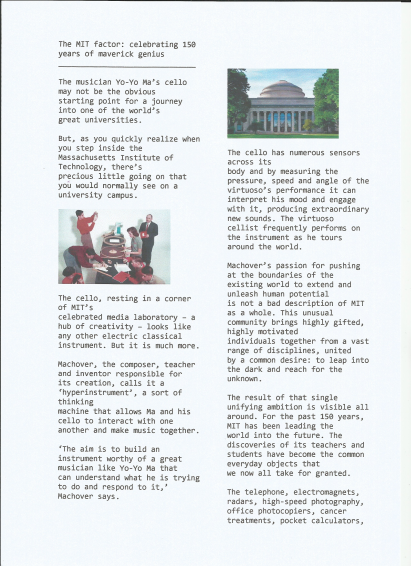
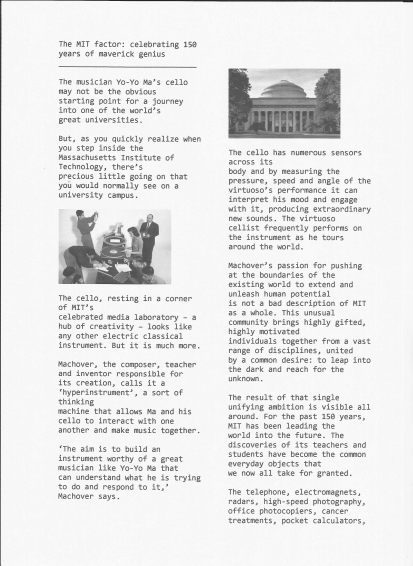
# 3.2. Описание алгоритма функционирования программы с обоснованием выбора схемы алгоритма решения задачи

Система оптического распознавания текстов состоит из нескольких основных блоков [1]:

* блок предобработки;
* блок сегментации;
* блок выделения признаков символов;
* блок распознавания символов;
* блок постобработки результатов распознавания.

# 3.2.1. Блок предобработки изображения

Предобработка играет важную роль в процессе сегментации и дальнейшего распознавания символов. Предобработка изображения выполняется в 4 шага.

   
 Рисунок 1: исходное изображение Рисунок 2: изображение в градациях серого

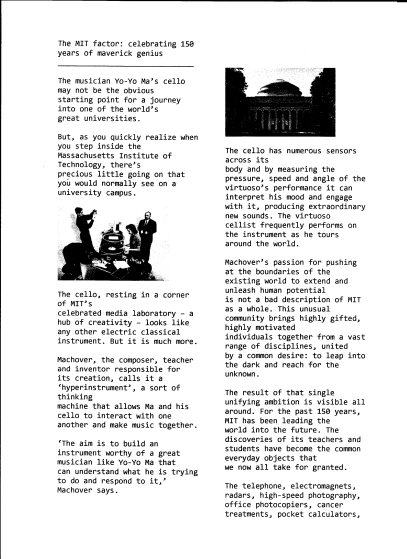
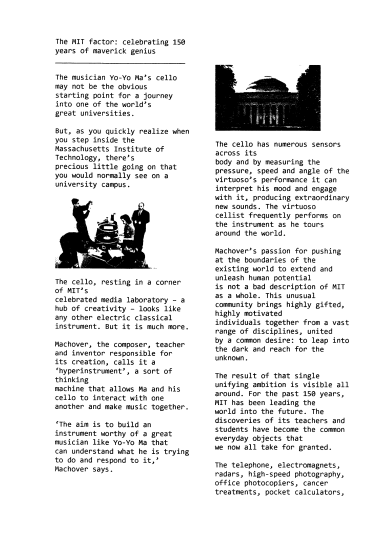
 

Рисунок 3: бинаризованное изображение Рисунок 4: убрать темные края

  
 Рисунок 5: открытие

Шаг 1. Перевод в градации серого

Для перевода изображения в оттенки серого (рис. 2) по формуле для каждого индекса считаем значение функции яркости [3]:

Таким образом, размер обрабатываемых данных (размер массива изображения уменьшается в 4 раза, т.к. вместо 4 байтов для каждого пикселя (RGBA), нужно хранить только одно значение функции яркости).

Шаг 2. Бинаризация изображения

Бинаризация – перевод изображения в градациях серого в изображения, в котором присутствуют только два типа пикселей (черные и белые) [4].

Выделяют два основных подхода к бинаризации: пороговый и адаптивный. Одним из быстрых и наиболее эффективных методов бинаризации является алгоритм вычисления порога бинаризации методом Оцу [5].

Был выбран именно этот алгоритм, потому что он позволяет разделить пиксели на 2 класса: «полезные» и «фоновые» [5], что хорошо применимо к изображениям с текстом, на которых в большинстве случаев есть два четко различающихся класса: фон и текст (рис. 3).

Согласно методу Оцу [6], минимизация внутриклассовой дисперсии эквивалентна максимизации межклассовой дисперсии, которая равна:

Где *w*1 и *w*2 — вероятности первого и второго классов соответственно, *a*1 и *a*2 — средние арифметические значения для каждого из классов [6].

Алгоритм Оцу [6]:

* вычисление гистограммы;
* проход через гистограмму, начиная с порога *t* = 1, на каждом шаге пересчитывая дисперсию. Если дисперсия оказалась больше максимума, то обновляем дисперсию и *T* = *t*;
* искомый порог равен *T*.

Шаг 3. Убирание темных краев

Некоторые сканеры в результате сканирования выдают результат – изображение, обрамленное черной рамкой. Это происходит в результате того, что размер сканируемой области обычно больше, чем стандартный размер листа A4.

По этой причине используется алгоритм убирания темных краев. Обработка происходит на бинаризованном изображении.

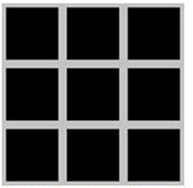
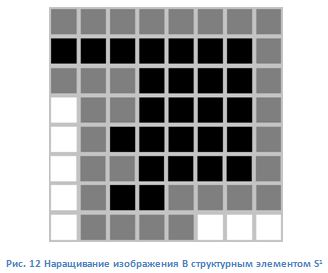
Алгоритм заключается в проходе по краям изображения и замене крайних черных пикселей и пикселей, которые к ним прилегают на белые (рис. 4).

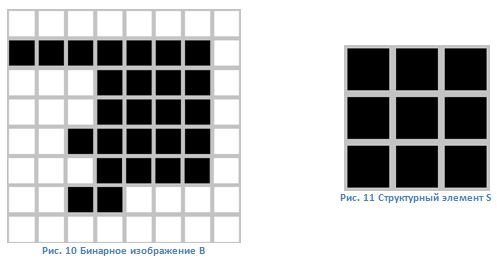
Шаг 4. Открытие

Один из наиболее важных этапов обработки – удаление шума. Широко известный способ устранения шума – используя операции математической морфологии, такие как сужение и расширение [7].

Есть два множества: объект обработки и структурный элемент.

Расширение аналогично логической операции «или». Структурный элемент накладывается поочередно на каждый пиксел изображения, если хотя бы 1 пиксел структурного элемента совпал с темным пикселем изображения, центр структурного элемента закрашивается (рис. 6) [7].

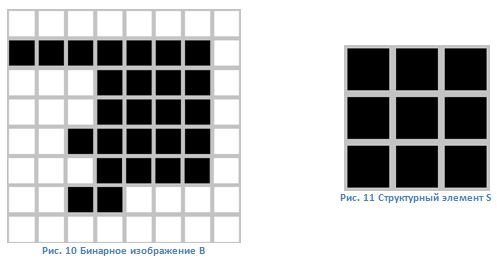
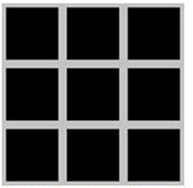
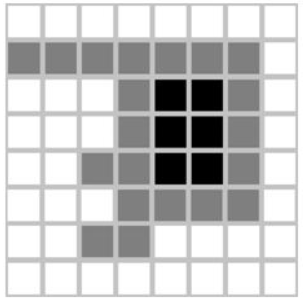
 



а) б) в)

Рисунок 6: а) исходное изображение, б) структурный элемент, в) результат операции расширение

При операции сужения структурный элемент накладывается поочередно на каждый пиксел изображения. Если весь структурный элемент совпадет с темными пикселами бинарного изображения, центр структурного элемента закрашивается (рис. 7) [7].

а) б) в)

Рисунок 7: а) исходное изображение, б) структурный элемент, в) результат операции сужение

Морфологическое открытие состоит в последовательном выполнении операций сужения и расширения, помогает удалить внешний шум, размеры которого меньше, чем диаметр структурного элемента (рис. 5) [7].

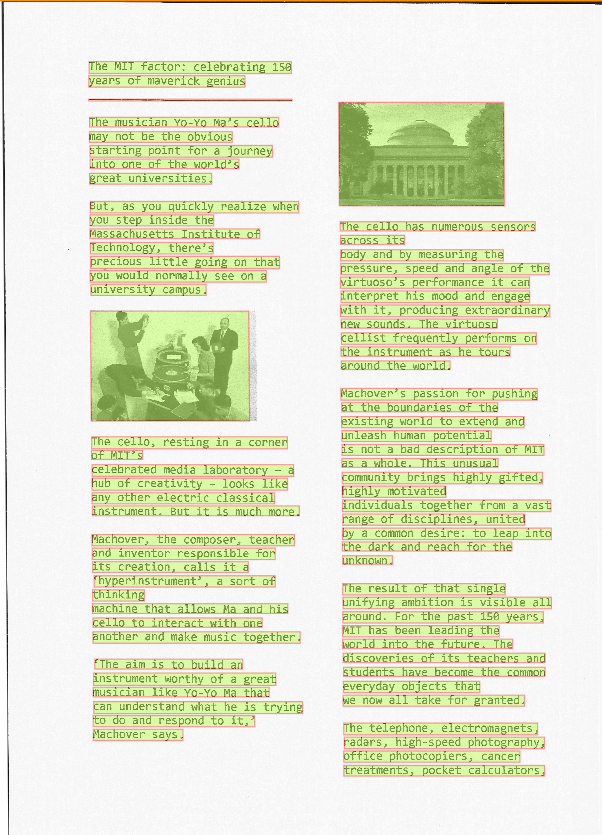
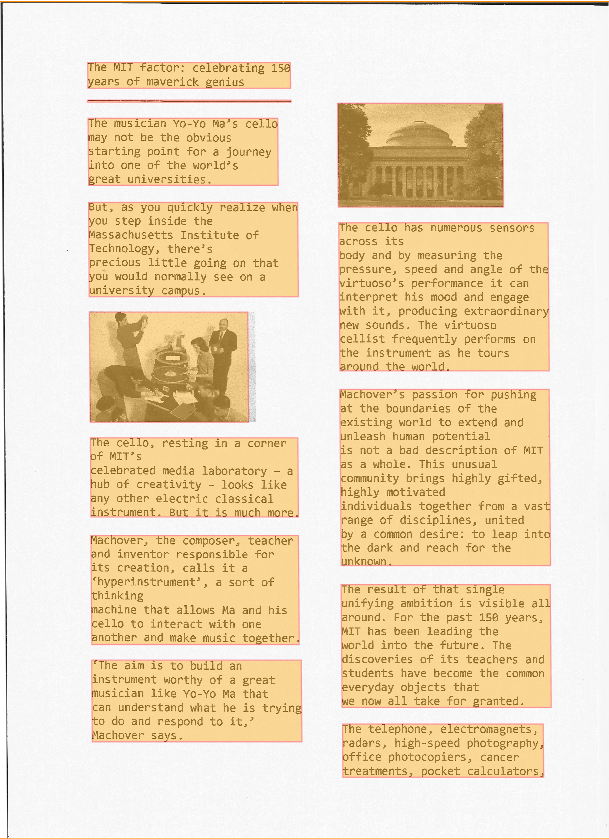
В программе был выбран структурный элемент вида «Кольцо» (рис. 8).

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

Рисунок 8: структурный элемент «Кольцо»

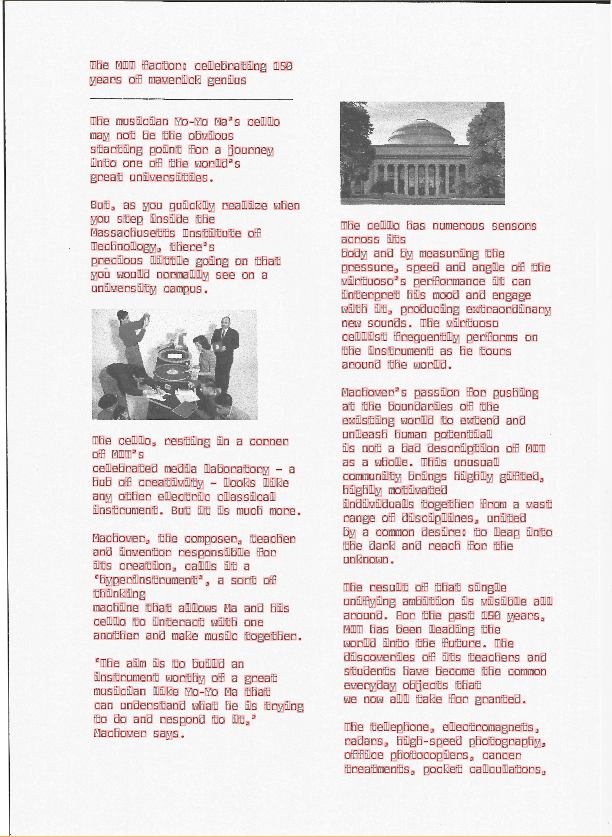
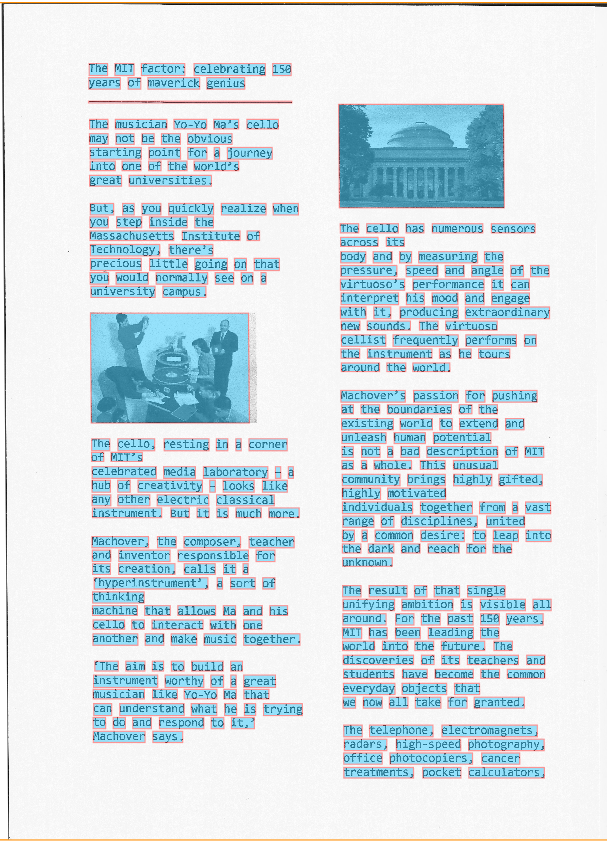
# 3.2.2. Блок сегментации изображения

В программе производится сегментация изображения на части, линии, слова и символы. Сегментация производится в предварительно обработанном изображении.



а) б)

Рисунок 9: а) части, б) линии



а) б)  
Рисунок 10: а) слова, б) символы

Шаг 1. Сегментация на части

При выделении частей изображения необходимо учитывать, что текст может быть расположен в несколько колонок (рис. 9(а)). Поэтому был выбран алгоритм XY-Cuts, который проходит все изображение вначале сверху вниз (Y-cut) в поисках сравнительно больших вертикальных пробелов, которые могли бы разделять части. Затем полученные части обрабатываются алгоритмом X-cut, и части разделяются на несколько других при нахождении больших вертикальных пробелов, которые могли бы разделять части.

Шаги 2 и 3 (сегментация частей на линии (рис. 9(б)) и линий на слова (рис. 10(а)) происходит аналогичным образом, используя алгоритмы X-cut и Y-cut.

Шаг 4. Сегментация на символы

В этом случае нельзя использовать вышеупомянутый алгоритм, потому что встречаются сочетания символов, между которыми нельзя провести вертикальную линию из белых пикселей (рис. 10(б)). Поэтому сегментация происходит по принципу поиска связанных компонент. Все символы английского языка состоят максимум из 2ух компонент связанности, так что задача сводится к поиску 1ой или 2ух компонент связанности.

Алгоритм:

* проходим по пикселам изображения слова слева направо, останавливаемся, если находим черный пиксел;
* двигаемся от найденного пиксела в 8-и возможных направлениях, перерисовывая матрицу символа, если следующий символ черный – рекурсивно продолжаем движение в 8-и направлениях, пока весь символ не будет найден;
* аналогично находим вторую компоненту связанности символа, если она есть;
* смещаемся относительно слова на ширину символа и продолжаем поиск.

# 3.2.3. Блок выделения признаков

Выделение признаков производится не по бинаризованным изображениям символов, а по изображениям в градациях серого с целью увеличить точность распознавания. В программе использовался алгоритм, позволяющий осуществить распознавание символа при помощи поиска по содержанию изображения [8]. Для распознаваемого изображения генерируется его индивидуальный хэш, который затем сравнивается с хэшами шаблонных изображений. В программе для поиска хэша изображения используется хэш-функция, которая отображает среднее значение низких частот [8].

Алгоритм [8]:

* уменьшение размера изображения в градациях серого до 8х8, чтобы избавится от высоких частот;
* рассчитывается среднее значение байтов в уменьшенном изображении;
* каждый байт заменить на 1 или 0, в зависимости от его значения, которое может быть больше или меньше среднего;
* последовательно записанные биты (нули и единицы) изображения образуют хэш.

Этот алгоритм был выбран т.к. итоговый хэш не изменяется при масштабировании, сжатии, растяжении, при небольшом изменении цвета [8].

# 3.2.4. Блок распознавания символов

В момент начала выполнения блока распознавания из файлов подгружаются предварительно рассчитанные хэши шаблонных изображений. Затем происходит побитовое сравнение хэша текущего изображения символа с каждым из хэшей шаблонов и вычисляется количество несовпадений (расстояние Хэмминга [9]). Распознанный символ соответствует шаблону, количество несовпадений с которым было минимальным.

# 3.2.5. Блок постобработки результатов распознавания

Одним из наиболее эффективных методов исправления ошибок распознавания является использование словаря [1]. После распознавания всех символов слова происходит поиск соответствующего слова в словаре. Если слово найдено, оно, скорее всего, распознано правильно. Если нет, находится самое близкое к нему слово (1 или 2 символа не совпадают) и выводится слово из словаря. Если такого слова не найдено, выводится первоначальный результат распознавания. Если найден более чем один кандидат из словаря, выбирается тот, замененный символ которого имеет наиболее близкий хэш с несовпадающим символом (минимальное расстояние Хэмминга).

# 3.3. Описание и обоснование метода организации входных и выходных данных

Формат входного файла: цифровое цветное изображение в формате «.jpg» или «.png», имеющее разрешение не более чем 2700х3700 пикселей. Для наилучшего качества распознавания рекомендуется при сканировании изображения задавать в параметрах разрешение, равное 300dpi. На входном файле может изображаться текст, напечатанный на английском языке шрифтом «Consolas» или «Calibri», от 12pt до 22pt и/или нетекстовые иллюстрации. Цвет текста должен быть более темным и должен заметно выделяться на фоне более светлого фона (при переводе в градации серого, функция яркости случайной точки фона и случайной точки текста должны отличаться не менее чем на 100). Текст не может располагаться в таблице/рамке, в противном случае будет распознан как нетекстовая иллюстрация.

Результат распознавания выводится в текстовое поле, представляет собой распознанный текст и нетекстовые иллюстрации. Результат распознавания можно сохранять в формате «.rtf».

# 3.4. Описание и обоснование метода выбора технических и программных средств

Для корректной работы программы рекомендуется компьютер, имеющий следующие   
технические характеристики:

- процессор с частотой 1,3 ГГц или более;  
 - 512 МБ ОЗУ или более;  
 - монитор с разрешением 800Х600 или более высоким;   
 - клавиатура и мышь или совместимые указывающие устройства.

# Требования к программным средствам, используемым программой:

- операционная система Windows XP/7/8;  
 - установленный Microsoft .NET Framework 4.0 .

# 4.ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

Задача оптического распознавания символов является одной из наиболее актуальных задач компьютерного зрения, решение которой активно используется для автоматизации документооборота, автоматизированной обработки больших объемов печатных документов, а также внедрения безбумажных технологий. Существует множество платных программ с аналогичным функционалом. Программа реализована в рамках выполнения курсовой работы за второй год обучения. Программа может свободно распространяться. Монетизация разработки не предусматривается.

# 5. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Обработка и анализ изображений в задачах машинного зрения: Курс лекций и практических занятий/ Ю.В. Визильтер, С.Ю. Желтов, А.В. Бондаренко, М.В. Ососков, А.В. Моржин. - М.: Физматкнига, 2010. - 672 с.
2. Szeliski R. Computer Vision: Algorithms and Applications/ R. Szeliski. - Springer, 2010.
3. Оттенки серого [Электронный ресурс] //URL: <https://ru.wikipedia.org/wiki/Оттенки_серого> (Дата обращения: 04.05.2015, режим доступа: свободный).
4. Бинаризация изображений [Электронный ресурс] //URL: <http://recog.ru/blog/applied/15.html> (Дата обращения: 04.05.2015, режим доступа: свободный).
5. Метод Оцу [Электронный ресурс] //URL: <https://ru.wikipedia.org/wiki/Метод_Оцу> (Дата обращения: 09.05.2015, режим доступа: свободный).
6. Обнаружение объектов методом Оцу [Электронный ресурс] //URL: <http://habrahabr.ru/post/112079/> (Дата обращения: 09.05.2015, режим доступа: свободный).
7. Математическая морфология [Электронный ресурс] //URL: <http://habrahabr.ru/post/113626/> (Дата обращения: 09.05.2015, режим доступа: свободный).
8. «Выглядит похоже». Как работает перцептивный хэш [Электронный ресурс] //URL: <http://habrahabr.ru/post/120562/> (Дата обращения: 15.05.2015, режим доступа: свободный).
9. Расстояние Хэмминга хэш [Электронный ресурс] //URL: <https://ru.wikipedia.org/wiki/Расстояние_Хэмминга> (Дата обращения: 15.05.2015, режим доступа: свободный).

# ПРИЛОЖЕНИЕ 1

ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| [ImageBytesClass](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\3eea08d3-38a7-9cc8-6ae6-45d47bd47c1b.htm) | Класс, реализующий байтовое представление изображения (массив/матрица) |
| [Line](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\ff599c43-2bf4-8132-42d1-8ec275438ac4.htm) | Линия изображения |
| [MainWindow](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\9e021649-b86a-3cdd-2804-14347d91a67a.htm) | Главное окно программы |
| [MessageBox](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\78eef74e-49b9-2c7c-0cf5-92fdcd0add9b.htm) | Окно для вывода сообщений пользователю |
| [Part](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\ba8e0f58-e25f-9fe8-d1bf-a80dc27d7cc8.htm) | Часть изображения (параграф) |
| [Pattern](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\713c285e-0ecd-79e5-6016-1c17406cd5bb.htm) | Класс - хэш шаблона изображения |
| [Postprocessing](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\6471d921-6637-6810-666a-e6a0f9405ef5.htm) | Постобработка результатов распознавания |
| [Preprocessing](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\52db370a-3a1c-d4a2-57e1-491564a9ff95.htm) | Изображение, полученное в результате предобработки |
| [ProcessingImage](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\75bf2463-f108-2777-21eb-29567f1606af.htm) | Обрабатываемое изображение |
| [ProcessStatus](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\c4c1f25d-6e36-3d84-93d7-b357d2203743.htm) | Класс, отображающий стадии распознавания |
| [Recognition](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\c28039b2-c08f-cb5d-b685-512127b48cd8.htm) | Распознавание символов |
| [Segmentation](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\ba028c5f-5658-e57a-0222-74837b5f2ca3.htm) | Сегментация изображения |
| [Word](file:///D:\STUDY\%D0%9B%D0%B5%D1%88%D0%B0\Help\Help\html\98cf1070-1340-a395-75f2-c87cd929f42e.htm) | Слово изображения |

# ПРИЛОЖЕНИЕ 2

ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ

# Preprocessing

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Preprocessing | Конструктор, инициализирующий изображение для последующей предобработки |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | bitmapImageToArr | Метод, возвращающий байтовое представление цветного изображения |
| - | bitmapImageToArrGS | Метод, возвращающий байтовое представление изображения в grayscale |
| - | checkDilatation | Проверка при наложении структурного элемента на изображение в заданной позиции |
| - | checkErosion | Проверка при наложении структурного элемента на изображение в заданной позиции |
| - | cleanXEdges | Удаление горизонтальных черных полос вверху и внизу изображения |
| - | cleanYEdges | Удаление вертикальных черных полос справа и слева изображения |
| - | doDilatation | Дилатация (расширение) изображения |
| - | doErosion | Эрозия (сужение) изображения |
| - | doOpening | Морфологическая операция открытия изображения |
| + | doPreprocessing | Метод, производящий предобработку изображения |
| - | findThreshold | Нахождение порога бинаризации методом Оцу |
| + | getHash | Метод, возвращающий строку, представляющую хэш текущего изображения |
| - | removePageEdge | Метод, удаляющий черные края |
| - | scaleCharacter | Метод, изменяющий размеры изображения |
| - | setBin | Бинаризация изображения |
| - | setGS | Метод, переводящий изображение из цветного в градации серого |
| + | toBitmapSource | Метод, переводящий массив нецветного изображения в BitmapSource |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | height | Высота изображения |
| - | imageBytesBin | Байтовое представление бинаризованного изображения |
| - | imageBytesGS | Байтовое представление изображения, переведенного в градации серого |
| - | width | Ширина изображения |
| - | x | Координата x текущего изображения относительно общего |
| - | y | Координата y текущего изображения относительно общего |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Height | Свойство приватного поля height |
| + | ImageBytesBin | Свойство приватного поля imageBytesBin |
| + | ImageBytesGS | Свойство приватного поля imageBytesGS |
| + | InitialImage | BitmapImage, предсталяющий исходное цветное изображение |
| + | Width | Свойство приватного поля width |
| + | X | Свойство приватного поля x |
| + | Y | Свойство приватного поля y |

# Pattern

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Pattern | Конструктор класса с параметрами |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Font | Шрифт текста |
| + | Hash | Хэш шаблона |
| + | Value | Символ - значение шаблона |

# MainWindow

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | MainWindow | Конструктор главного окна |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | addTextRichTextbox | Метод, заполняющий текстовое поле результатами распознавания |
| - | backgrWorker\_DoWork | Метод, выполняемый в фоновом потоке |
| - | backgrWorker\_ProgressChanged | Изменение статуса обработки |
| - | backgrWorker\_RunWorkerCompleted | Завершение распознавания в фоновом потоке |
| - | cutPartFromImage | Вырезать иллюстрацию (часть) из цветного изображения |
| - | drawChars | Метод, рисующий символы |
| - | drawLines | Метод, рисующий линии |
| - | drawParts | Метод, рисующий части |
| - | drawWords | Метод, рисующий слова |
| - | ExitButton\_Click | Обработчик нажатия на кнопку Закрыть |
| - | goRecognize | Метод, выполняющий обработку изображения и распознавание |
| - | loadDictionary | Загрузить список английских слов |
| - | MaximizeButton\_Click | Обработчик нажатия на кнопку Развернуть |
| - | MinimizeButton\_Click | Обработчик нажатия на кнопку свернуть |
| - | NextButton\_Click | Обработчик нажатия на кнопку вправо |
| - | OpenButton\_Click | Обработчик нажатия на кнопку Выберете файл |
| - | OpenImage | Открытие изображения |
| - | PreviousButton\_Click | Обработчик нажатия на стрелку влево |
| - | RecognizeButton\_Click | Обработчик нажатия на кнопку Распознать |
| - | reset | Сбросить все |
| - | RestoreButton\_Click | Обработчик нажатия на кнопку Восстановить |
| - | SaveButton\_Click | Обработчик нажатия на кнопку Сохранить |
| - | showImage | Метод, отображающий стадии обработки |
| - | Window\_MouseDown | Обработчик нажатия на кнопку мыши |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | backgrWorker | Обработчик фоновых событий |
| - | constBitmapSource | BitmapSource для отображения позиции курсора |
| - | currStatus | Перечисление - текущий статус обработки изображения на панели |
| - | dependencyObjectDictionary | Словарь объектов ричтекстбокса |
| - | englishWordsList | Список английских слов |
| - | postProcessing | Постобработка результатов |
| - | preprocImage | Предобработанное изображение |
| - | recognizeImage | Распознавание символов изображения |
| - | segmentImage | Сегментация изображения |

# ProcessStatus

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | ProcessStatus | Статический конструктор |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | toBitmapSource | Метод, переводящий массив нецветного изображения в BitmapSource |
| + | toBitmapSourceFull | Метод, переводящий массив цветного изображения в BitmapSource |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | statusDictionary | Статическое поле - словарь (статус - массив изображения) |

# Line

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Line | Конструктор с параметрами |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | doX\_Cut | Метод, сегментирующий линии на слова |
| + | segmentWords | Метод, сегментирующий линии на слова |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | x | Координата x линии относительно всего изображения |
| - | y | Координата y линии относительно всего изображения |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | LineImage | Изображения линии |
| + | Words | Список слов текущей линии |
| + | X | Свойство приватного поля x |
| + | Y | Свойство приватного поля y |

# ImageBytesClass

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | ImageBytesClass | Конструктор с параметром |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | arrayToMatrix | Статический метод, конвертирующий массив изображения в матрицу изображения |
| + | copyMatrixXY | Статический метод, копирующий часть матрицы от и до заданных индексов в другую |
| + | getArr | Метод, переводящий матрицу изображения в массив |
| + | GetLength | Длина матрицы по заданному измерению |
| + | matrixSetWhite | Статический метод, устанавливающий все значения заданной матрицы равными 255 (белый) |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | matrix | Матрица байтов изображения |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Item[([( Int32])]) | Индексатор, предоставляющий доступ к элементу матрицы изображения по одному индексу (обращение, как к элементу массива) |
| + | Item[([( Int32, Int32])]) | Индексатор, предоставляющий доступ к элементу матрицы изображения по двум индексам (обращение, как к элементу матрицы) |
| + | Length | Общая длина матрицы изображения |
| + | Matrix | Свойство приватного поля matrix, возвращающее и устанавливающее матрицу изображения |

# ProcessingImage

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | ProcessingImage(String, Boolean) | Конструктор с параметрами, принимающий строку - имя входного файла и логическую переменную, определяющую, нужно ли сохранять текущий статус изображения |
| + | ProcessingImage(MemoryStream, Int32, Int32) | Конструктор с параметрами, инициализирующий изображение заданной длины и ширины из потока |
| + | ProcessingImage(array<Byte,2>[,](,)[,][,], array<Byte,2>[,](,)[,][,], Int32, Int32, Int32, Int32) | Конструктор с параметрами, инициализирующий изображение |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | bitmapImageToArr | Метод, возвращающий байтовое представление цветного изображения |
| - | bitmapImageToArrGS | Метод, возвращающий байтовое представление изображения в grayscale |
| + | getHash | Метод, возвращающий строку, представляющую хэш текущего изображения |
| + | SaveImg |  |
| - | scaleCharacter | Метод, изменяющий размеры изображения |
| - | setGS | Метод, переводящий изображение из цветного в градации серого |
| + | toBitmapSource | Метод, переводящий массив не цветного изображения в BitmapSource |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | height | Высота изображения |
| - | imageBytesBin | Байтовое представление бинаризованного изображения |
| - | imageBytesGS | Байтовое представление изображения, переведенного в градации серого |
| - | width | Ширина изображения |
| - | x | Координата x текущего изображения относительно общего |
| - | y | Координата y текущего изображения относительно общего |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Height | Свойство приватного поля height |
| + | ImageBytesBin | Свойство приватного поля imageBytesBin |
| + | ImageBytesGS | Свойство приватного поля imageBytesGS |
| + | InitialImage | BitmapImage, представляющий исходное цветное изображение |
| + | Width | Свойство приватного поля width |
| + | X | Свойство приватного поля x |
| + | Y | Свойство приватного поля y |

# Part

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Part | Конструктор с параметрами |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | doY\_Cut | Метод, сегментирующий изображение части на линии |
| + | isText | Метод, проверяющий, является ли часть текстовой |
| - | segmentLines | Метод, сегментирующий часть на линии |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | x | Координата x части относительно всего изображения |
| - | y | Координата y части относительно всего изображения |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Lines | Список линий текущей части |
| + | PartImage | Изображение части |
| + | X | Свойство приватного поля x |
| + | Y | Свойство приватного поля y |

# Recognition

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Recognition | Конструктор с параметром |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | compareWithPattern | Метод, сравнивающий хэши текущего символа и шаблона |
| - | correctWord | Метод, корректирующий регистры символов распознанного слово |
| - | determineCategory | Метод, определяющий, к какой категории относится символ |
| + | doRecognition | Метод, производящий распознавание символов |
| - | findAverage | Метод, устанавливающий среднюю ширину и высоту символов одной линии |
| - | findWordLowerBaseLine | Метод, возвращающий нижнюю базовую линию слова |
| - | findWordUpperBaseLine | Метод, возвращающий верхнюю базовую линию слова |
| - | loadFromFile | Метод, загружающий из файлов хэши шаблонных изображений символов |
| - | loadPatterns | Метод, загружающий шаблонные хэши |
| - | recognizeCharacter | Метод, распознающий изображения символа |
| - | recognizeWord | Метод, производящий распознавание символов слова |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | averCharHeight | Средняя высота символа |
| - | averCharWidth | Средняя ширина символа |
| - | horizontalCalibr | Список шаблонов горизонтально вытянутых символов |
| - | horizontalCons | Список шаблонов горизонтально вытянутых символов |
| - | normalCalibr | Список шаблонов средних символок |
| - | normalCons | Список шаблонов средних символок |
| - | segmentImage | Сегментированное изображение |
| - | tinyCalibr | Список шаблонов маленьких символов |
| - | tinyCons | Список шаблонов маленьких символов |
| - | uncertain | Список символов, для которых заглавные и прописные буквы похожи по написанию |
| - | verticalCalibr | Список шаблонов вертикально вытянутых символов |
| - | verticalCons | Список шаблонов вертикально вытянутых символов |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | BigCalibr | Список шаблонов больших букв |
| + | BigCons | Список шаблонов больших букв |
| + | FontCalibri | Количество распознанных символов шрифта Calibri |
| + | FontConsolas | Количество распознанных символов шрифта Consolas |

# Segmentation

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Segmentation | Конструктор с параметром |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | checkHorizontalWhitespace | Статический метод, проверяющий, является ли пробел большим (отделяет ли части) |
| - | checkVerticalWhitespace | Статический метод, проверяющий, является ли пробел большим (отделяет ли части) |
| - | clearLists | Очистка списков |
| + | doHorizontalCut | Статический метод, обрезающий изображение по горизонтали |
| + | doSegmentation | Метод, производящий сегментацию изображения на части, линии, слова, символы |
| + | doVerticalCut | Статический метод, обрезающий изображение по вертикали |
| - | doX\_Cut | Метод, сегментирующий изобращение на части по вертикали |
| - | doY\_Cut | Метод, сегментирующий изобращение на части по горизонтали |
| + | findNextEmptColumnX | Статический метод, находящий конец части (координату x перед пробелом) |
| + | findNextEmptLineY | Статический метод, находящий конец части (координату y перед пробелом) |
| - | segmentParts | Сегментация изображения на части |
| + | totalCharacters | Метод, подсчитывающий общее число символов и слов |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | HOR\_WHITESPACE\_BTW\_PART | Расстояние между частями по горизонтали |
| + | LITTLE\_SIZE | Размер части, при котором она удаляется из списка |
| + | VERT\_WHITESPACE\_BTW\_LINES | Расстояние между строками /// |
| + | VERT\_WHITESPACE\_BTW\_PART | Расстояние между частями по вертикали |
| + | WORD\_MAXIMUM\_HEIGHT | Максимальная высота символа |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Parts | Список сегментированный частей изображения |
| + | PreprocImage | Предобработанное изображение |

# Word

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Word | Конструктор с параметрами |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | around | Рекурсивный метод, находящий 1 связанную компоненту - часть или весь символ слова |
| - | cut1Pixel | Метод, обрезающий 1 пиксел по всем сторонам изображения символа |
| - | cutCharacter | Метод, обрезающий изображения символа с четырех сторон |
| - | findCharacterAround | Метод, находящий символ слова |
| - | findSecondComponent | Метод, находящий вторую связанную компоненту символа |
| + | isText | Метод, проверяющий, является ли слово текстом |
| + | segmentCharacter | Метод, сегментирующий слово на символы |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | x | Координата x слова относительно всего изображения |
| - | y | Координата y слова относительно всего изображения |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Characters | Список символов текущего слова |
| + | Value | Значение слова |
| + | WordImage | Изображение слова |
| + | X | Свойство приватного поля x |
| + | Y | Свойство приватного поля y |

# Postprocessing

## Конструкторы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Postprocessing | Конструктор с параметром |

## Методы:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | checkWithWordsList | Метод, исправляющий распознанное слово и проверяющий его наличие в словаре |
| - | checkWordWithList | Метод, ищущий слово или его модификации в словаре |
| - | chooseBestCandidate | Метод, выбирающий лучшего кандидата на замену |
| - | countDifference | Метод, считающий количество несовпадающих символов двух слов |
| - | countDistance | Метод, считающий разницу хэшей изображений символов слов |
| + | doPostprocessing | Метод, выполняющий постобработку |

## Поля:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| - | dictionary | Список английских слов |
| - | patterns | Словарб символ-хэш |

## Свойства:

|  |  |  |
| --- | --- | --- |
| **Модиф. доступа** | **Название** | **Описание** |
| + | Segmentation | Результат сегментации |

# ПРИЛОЖЕНИЕ 3

ДИАГРАММЫ UML

Диагрмма классов

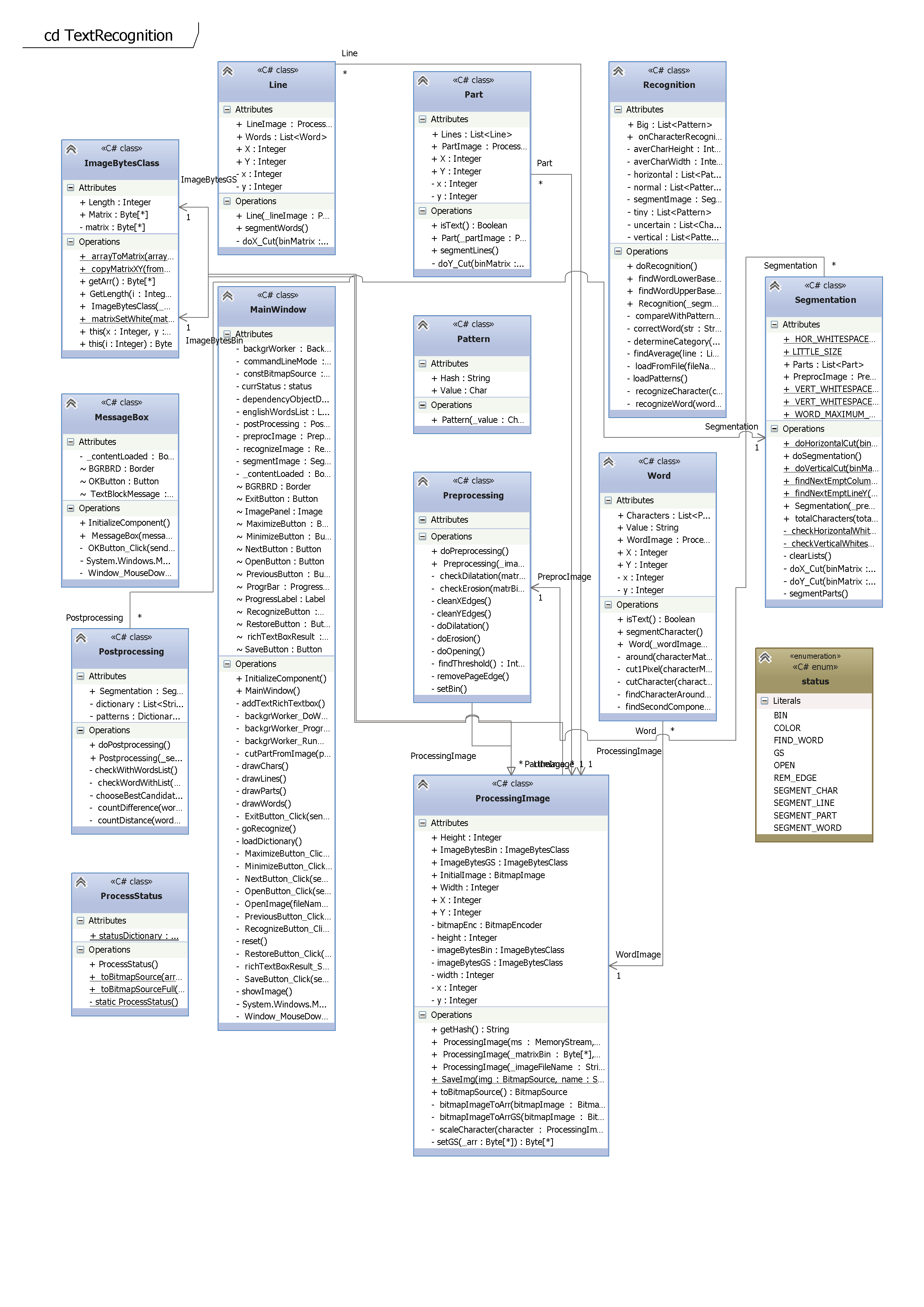


Диаграмма деятельности



|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц) в докум. | № документа | Входящий номер сопроводи-тельного докум. и дата | Под-пись | Дата |
| Изм. | изменен-ных | заменен-ных | новых | аннулиро-ванных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |