|  |
| --- |
| FILS |
| BAMOTSU |
| BAM - Java image binarization using Otsu’s algorithm |
|  |
| **Carmen Casaru, Andrei Moisei, Mihaita Tinta** |
| **11/5/2013** |

|  |
| --- |
| There are many binarization algorithms these days and almost all of them involve several steps or processing. The raw image is transformed in a black and white version. This step is critical because many errors can be propagated from here. In character recognition is very important to separate the background from the letters. |

Contents

[Software Design Diagram 2](#_Toc371413582)

[Document purpose 2](#_Toc371413583)

[Objectives 2](#_Toc371413584)

[Overview 2](#_Toc371413585)

[Data Design 2](#_Toc371413586)

[Architectural design 2](#_Toc371413587)

[System architecture 2](#_Toc371413588)

[Patterns 2](#_Toc371413589)

[Diagram 2](#_Toc371413590)

[Description 2](#_Toc371413591)

[Implementation requirements 2](#_Toc371413592)

[Component interaction 2](#_Toc371413593)

[User interface design 2](#_Toc371413594)

[Flow chart 2](#_Toc371413595)

[Screen images 2](#_Toc371413596)

[Testing issues 2](#_Toc371413597)

[Critical components 2](#_Toc371413598)

[Alternatives 2](#_Toc371413599)

[Workdown brearking structure 2](#_Toc371413600)

[Gant diagram 3](#_Toc371413601)

[Git 3](#_Toc371413602)

# Software Design Diagram

# Document purpose

This document is intended to describe the capabilities of the software product “Voting-Based Image Binarization” provided to its end-users. It also specifies all the non-functional requirements that the application should implement, regarding: performance, availability, reliability and security.

# Objectives

The main Goal of this project is to convert images into binary ones and also create an additional matrix representing the level of trust for every pixel of the initial image.

The purpose of the project is to develop an ”Image Binarization System” (IBS). The IBS will consist from two parts:

* A ”Binarization Algorithm Module” (BAM):

This will be an executable which will receive an input continuous-tone image and will produce an output binary image.

* A “Voting Binarization Algorithm Module” (VBAM): using more BAMs a “smart-voting” technology will be used to blend the independent BAMs results into a binary image.

# Overview

The remainder of this document is three chapters, the first offering a general description of the software product about the initial situation, the purpose of the project, the context and the benefits of the project.

The second chapter lists the functional requirements that the software product should meet. It describes the actors, the system boundary and the use cases.

The final chapter exposes the non-functional requirements of the application: performance, safety and security issues.

There are many binarization algorithms these days and almost all of them involve several steps or processing. The raw image is transformed in a black and white version. This step is critical because many errors can be propagated from here. In character recognition is very important to separate the background from the letters.

Data Design

A BAM will be an executable which will receive from command line two file names (input\_image and an output\_image). The BAM will return an error-code: zero for no error (results are valid and will be used for voting purposes) and nonzero in case of an error occurrence (the error code should specify the error type; in this case the result will not be considered).

The output of the BAM is a 1bpp image, output\_image and an 8bpp image, output\_image- confidence. The first image is the actual binarization and the second a gray-scale image containing the confidence for the binarization for every pixel. 0 means that the respective pixel was randomly assigned a color (black or white); 255 means that the algorithm is absolutely certain that the respective color of the pixel is correctly assigned.

The Otsu binarization method created by Nobuyuki Otsu assumes that the image to be thresholded contains two classes of pixels: foreground (something with interest for the user) and background (no interest for the user). It calculates the optimum threshold separating those two classes so that their combined spread (intra-class variance) is minimal (this represents the error).

The main advantage is the speed because we only need to compute the histograms and arrays of 256 and the easiness of the implementation.

Architectural design

System architecture

## Patterns

## Diagram

## Description

## Implementation requirements

## Component interaction

# User interface design

## Flow chart

## Screen images

## Testing issues

## Critical components

## Alternatives

# Workdown brearking structure

Leafs < 80 hours

Resources budget and time

Sum(leaf) = parent

Leaf1 intersect leaf2 = void

# Gant diagram

Git