Documentul de Proiectare a Soluţiei Aplicaţiei Software
(Software Design Document)

Versiune 1.0
05 November, 2013

Aplicaţie de binarizare a unei imagini

Structura echipei:
Marin Andreea - Lavinia
Tănăsescu Oana - Adelina
Lipan Alin – Ionuţ
State Andra-Mihaela
Barahtian Oana Maria
Costea Cristina
Dragomir Andreea Alisa
Trifu Mihai Bogdan
Neacșu Mihai
Țuca Laurentiu-Ionuț
Popescu Andrei
Stamati Daniela
Godeanu Dragoș

Îndrumător: Octavian Saca

Facultatea de Automatică şi Calculatoare, Universitatea Politehnica, Bucureşti

Cuprins

1.Introducere

1.1.Scopul documentului

Acest document are rolul de a descrie acurat şi complet soluţia proiectată pentru aplicaţia de binarizare a unei imagini, care se presupune a fi o pagină (veche) de ziar.

Documentul serveşte drept ghid unic de construire a soluţiei pentru echipa de dezvoltare a proiectului.

Acest document va fi înțeles si utilizat de către echipa de dezvoltatori ai aplicației, precum și de către utilizatori.

1.2.Definirea termenilor tehnici

Pixel= Cel mai mic element (punctual) în care se poate descompune o imagine obținută prin fotografiere (inclusiv prin teledetecție), tipărire sau creată pe un ecran.

Digitalizare= trecerea unui document dintr-un format fizic in unul digital.

BAM= Binarization Algorithm Module

VBAM= Voting Binarization Algorithm Module

Binarizare = Această operaţie are ca obiectiv obţinerea unei imagini alb-negru dintr-o imagine care conţine şi alte nuanţe nedorite provenite din diverse motive tehnice (de exemplu copiere).

Testare white-box= strategie de generare a testelor pe baza structurii interne a codului.

Testare black-box(testare functionala) = strategie de testare care necesită cunoasterea comportamentului extern al programului pe baza specificațiilor.

IBS = Image Binarization System

1.3.Conţinutul documentului

In prima parte a documentului se realizează descrierea într-o manieră sumară a proiectului, conținutul și scopul acestuia.

Cea de-a doua parte cuprinde modelul datelor folosit in aplicație. Acest model prezinta principalele structuri de date și formatul fișierelor folosite pentru schițarea soluției.

Cea de-a treia parte a documentului cuprinde modelul arhitectural si cel al componentelor. Aceste modele prezintă şabloanele folosite, arhitectura sistemului şi descrie componentele acestei arhitecturi.

Urmatoarea parte va descrie modul de interacțiune al utilizatorului cu aplicația, iar partea a patra cuprinde elemente de testare care prezintă componentele critice şi alternative de proiectare ale acestora.

2.Obiective

2.1. Situația actuală

In decursul ultimelor decenii, crearea si depozitarea documentelor a trecut de la formatul fizic la cel electronic. Acest fapt a schimbat radical modul in care oamenii interacționeaza cu datele.

Deși documentele create recent beneficiază atât de suport fizic cât și electronic, cele din trecut se regasesc doar in forma lor fizica. Acest lucru poate fi un dezavantaj întrucat exemple de astfel de documente pot fi

manuscrise sau tiparituri ce conțin informatii importante ce s-ar dori sa fie procesate in mod automat.

2.2.Scopul aplicației

Aplicatia are drept scop digitalizarea unor documente prin supunerea lor unor algoritmi complexi de binarizare, avand drept rezultat un output cât mai lizibil, indiferent de gradul de iluminare sau degradare al documentului original.

2.3.Contextul aplicației

Acest proiect are drept scop dezvoltarea unui sistem IBS (de binarizare a imaginilor) care include două module:

i) Modulul de binarizare care transformă imaginea in una binară.
ii) Modulul de votare care compune o singură imagine binară folosind mai multe rezultate ale mai multor module de binarizare.

Binarizarea unui document este un proces complex ce presupune mai multi pași de procesare, dar care nu se mai aplică datorită posibilității mari de apariție a erorilor. De aceea, se pleacă de la o variantă alb-negru a imaginii pentru a diferenția cât mai bine fundalul de conținut prin binarizare.

3. Modelul datelor

3.1. Structuri de date globale

In program vom introduce și folosi o structură de date globală in care vom reține fișierul de intrare asupra căruia se vor aplica algoritmii de binarizare.

Beneficiul utilizarii acestei structuri de date ar fi acela ca incărcarea datelor din fișierul de intrare se realizează o singură dată si nu de mai multe ori in cazul rulării fiecarui algoritm in parte.

3.2. Structuri de date temporare

Pentru stocarea in memorie a imaginilor se vor folosi matrici alocate dinamic.

Pentru reținerea rezultatului obținut in urma citirii si prelucrarii datelor de intrare se va folosi o clasă sau o structură.

Pixelii vor lua valori in intervalul 0-255 si vor avea valoarea 0 la inițializare.

3.3. Structuri de date de legatură

Nu se vor folosi structuri de date de legatură.

3.4 Formatul fișierelor

Imaginile vor avea format de tip .PGM
Un fișier de tipul PGM va conține:

1) Un "număr magic" pentru identificarea fișierului. Exemplu: P2/P4.
2) Lungimea si lătimea reprezentate de numerele N si M unde N=numărul de linii a matricei de pixeli si M=numărul de coloane.
3) Matricea de pixeli propriu-zisă.
   Exemplu:

   P2/P4
   24 7

```
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  3  3  3  3  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15 15 15 15  0
 0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0 15  0
 0  3  3  3  0  0  0  7  7  7  0  0  0 11 11 11  0  0  0 15 15 15 15  0
 0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0  0  0
 0  3  0  0  0  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

4. Modelul arhitectural şi modelul componentelor

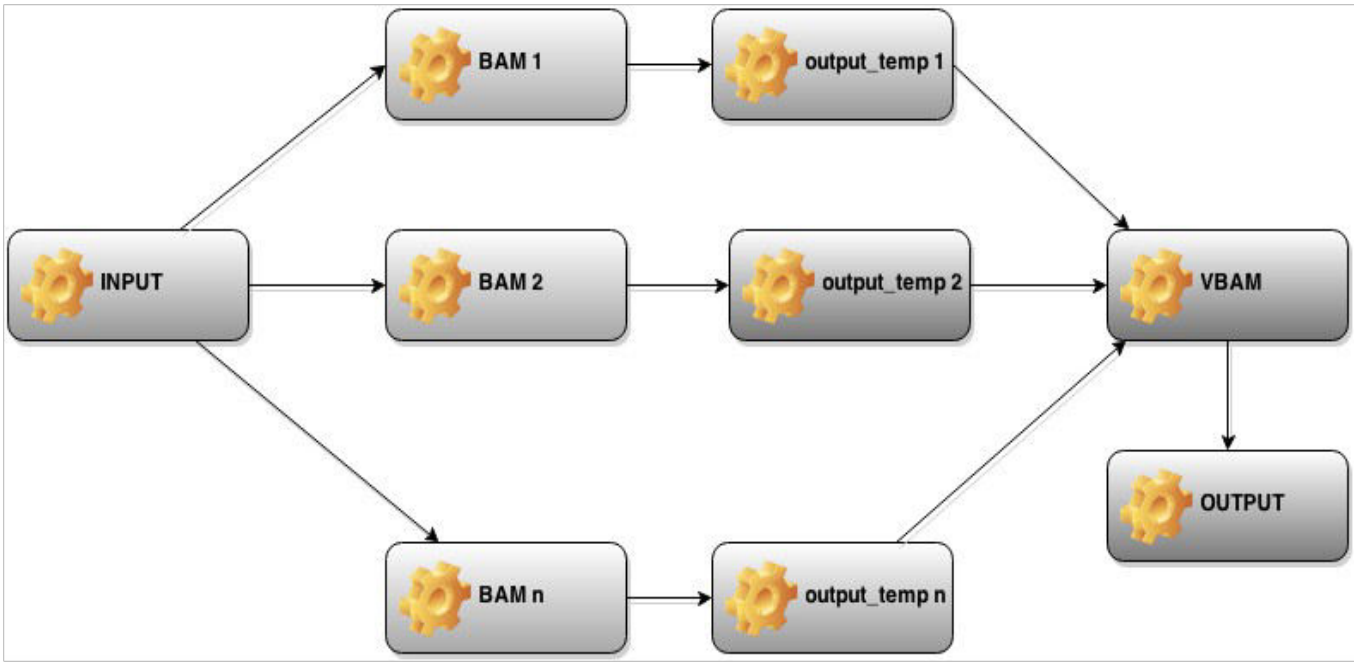
4.1 Șabloane arhitecturale folosite

Aplicatia contine doua tipuri de module:VBAM si BAM.

BAM-urile preiau ca input o imagine, aplică un algoritm de binarizare si intorc drept output două imagini.

Prima imagine reprezintă soluția după aplicarea algoritmului, iar cea de-a doua reprezinta factorul de incredere pentru corectitudinea imaginii output.

4.2 Diagrama de arhitectură



4.3 Descrierea componentelor

- **BAM**: aplică un algoritm de binarizare asupra imaginii primite ca input și realizează
imaginea binarizată și imaginea conținând factorul de corectitudine pentru fiecare pixel.

- **VBAM**: preia output-ul modulelor BAM (imaginile binarizate și imaginile factor de

corectitudine) și le supune unui algoritm de votare în urma căruia va desemna o singură
imagine, presupusă corectă (sau cea mai apropiată de corectitudine**).**

4.4 Restrictii de implementare

- Modulele BAM și VBAM vor fi dezvoltate utilizând limbajul de programare C/C++.
- Formatul imaginilor de intrare va fi .PGM.

4.5 Interactiunea dintre componente

Modulul VBAM va apela modulele BAM si va genera imaginea binarizată pe baza algoritmului de votare.

5. Elemente de testare

5.1 Componente critice

Pentru a se verifica corectitudinea aplicatiei se vor introduce parametrii de intrare gresit, se vor introduce fisiere de un alt format, altul decat cel suportat de aplicatie, imagini corupte, numar gresit de BAM-uri.

Pentru testare se va realiza de asemenea un script care sa preia automat datele din fișierele de intrare, sa le prelucreze și să afișeze imaginile corecte intr-un fisier de output.

Fiecare componentă va fi testată individual.

5.2 Tipuri de testare

Testarea se poate realiza ca:

1. Testare White-Box ce presupune ca testerul care acces in sursele

programului. De multe ori, testarea prin aceasta metodă implică scrierea de cod sau cel puțin, urmarirea celui existent. Se testează fiecare metodă in parte inițial, cât si interoperarea metodelor.

2. Testare Black-Box ce presupune testarea aplicatiei la nivelul user-ului, plecand de la premisa ca nu se cunoaste modul in care functionează. Pentru acest tip de testare, testerul trebuie sa cunoască rezultatele ce se așteaptă din partea aplicației.

5.3 Alternative

Modulul VBAM poate executa in paralel evaluarea fiecarui modul BAM prin pornirea mai multor procese.