MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

FUZIUNEA IMAGINILOR MEDICALE – STUDIU COMPARATIV

LUCRARE DE LICENȚĂ

Absolvent: **Arthur HENNING**

Coordonator ştiinţific: **Asist. univ. ing. Cornelia MELENTI**

2014

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

DECAN,
Prof. dr. ing. Liviu MICLEA

DIRECTOR DEPARTAMENT,
Prof. dr. ing. Rodica POTOLEA

Absolvent: **Arthur HENNING**

FUZIUNEA IMAGINILOR MEDICALE – STUDIU COMPARATIV

1. **Enunţul temei:** Studiu comparativ între metode de fuziune a imaginilor medicale în scopul de a extrage și a combina informațiile importante cât mai precis din mai multe surse.

2. **Conţinutul lucrării:** Cuprins, Listă de imagini, Listă de figuri, Introducere, Obiectivele proiectului, Studiu Bibliografic, Analiză și Fundamentare Teoretică, Proiectare de Detaliu și Implementare, Testare și Validare, Manual de Instalare și Utilizare, Concluzii, Bibliografie.

3. **Locul documentării**: Universitatea Tehnică din Cluj-Napoca, Departamentul de Calculatoare

4. **Consultanţi**: Cornelia Melenti

5. **Data emiterii temei:** 1 noiembrie 2013

6. **Data predării:** 3 Iulie 2014

Absolvent: _____

Coordonator ştiinţific: _____

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Declaraţie pe proprie răspundere privind autenticitatea lucrării de licenţă

Subsemnatul(a)_____
_____,
legitimat(ă) cu _____ seria _____ nr. _____
CNP _____, autorul lucrării
_____
_____
_____elaborată în vederea susţinerii examenului de finalizare a studiilor de licență la Facultatea de Automatică şi Calculatoare, Specializarea _____ din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea _____ a anului universitar _____, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării, şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă*.

Data

_____

Nume, Prenume

_____

Semnătura

Cuprins

Listă de figuri

Listă de tabele

Capitolul 1. Introducere

1.1. Contextul proiectului

Proiectul se dorește a oferi o modalitate eficientă de extragere și combinare a informației relevante din mai multe surse de imagini medicale prin fuziunea lor. Prin acest procedeu se poate vizualiza și diagnostica mai precis o eventuală boală, tumoare sau orice altă disfuncțiune a organismului.

Rolul principal al acestui procedeu este de a scuti pacientul de mai multe metode de imagistică medicală, invazive pentru corp, cum ar fi Rezonanța Magnetică (RMN), Computer Tomograf (CT). Aceste tehnologii, deși deseori oferă rezultate mai precise sau sunt mai recomandate pentru anumite părți ale corpului, cu utilizare repetată pot cauza probleme medicale pentru cel căruia i se aplică. Ideea generală este de a efectua o singură dată una din metodele amintite mai sus, la un stadiu inițial al stării de sănătate al pacientului, după care se pot face mai multe imagini non-invazive, cum ar fi ecografia. Combinând cele două tipuri de imagini, rezultă o imagine de calitate mai bună decât oricare cele două imagini sursă, și care va reprezenta zona corpului într-o stare cât mai recentă.

Această metodă de fuziunea al imaginilor medicale se practică deja în țările vestice, precum și Franța, în capitala căreia, Paris, se organizează anual conferința ICIFE (International Conference on Image Fusion Engineering), care are ca scop însumarea experiențelor și rezultatelor cercetării în domeniul fuziunii imaginilor.

Scopul proiectului este de a oferi o soluție pentru îmbunătățirea informației obținută din mai multe imagini medicale, care au fost făcute utilizând metode diferite, și au fost efectuate în stadii diferite al stării de sănătate al pacientului. S-au folosit diferiți algoritmi pentru fuziunea propriu-zisă, și s-au comparat rezultatele primite atât subiectiv, cu ajutorul a mai multor voluntari care au oferit un anumit punctaj pentru aspectul imaginilor, cât și obiectiv, cu ajutorul unor metode de măsurare a calității imaginilor.

1.2. Rezumatul capitolelor

În cele ce urmează se descrie pe scurt conținutul capitolelor din această lucrare. Informația este grupată în așa fel încât documentul să fie cât mai lizibil și ușor de înțeles.

În Capitolul 2, „Obiectivele proiectului” este descris domeniul din care face parte tema, problemele pe care le abordă și o propunere inițială de rezolvarea acestora. Sunt enumerate și cerințele funcționale și non-funcționale, pe care sistemul software trebuie să le îndeplinească. Mai departe sunt prezentate componentele importante ale sistemului, și datele de intrare și ieșire prin care comunică.

În Capitolul 3, „Studiu Bibliografic” sunt descrise separat și detaliat tehnologiile, protocoalele și domeniul în care se situează tema proiectului. Aici avem informații despre imagistica medicală, standardul DICOM, procesul de fuziune a imaginilor, aplicații similare și framework-uri candidate pentru implementarea proiectului.

În Capitolul 4, „Analiză și Fundamentare Teoretică”, se prezintă fluxurile generale ale sistemului, datele de intrare și ieșire detaliate, algoritmi folosiți pentru implementarea metodelor de fuziune și postprocesare și o posibilitate aleasă pentru măsurarea și compararea calității.

În Capitolul 5, „Proiectare de Detaliu și Implementare”, se descrie implementarea propriu-zisă a soluțiilor teoretice propuse precedent. Prin diagrame și secvențe de cod

s-a detaliat proiectarea si funcționarea aplicației, accentuând clasele și metodele principale precum și diferite secvențe de acțiuni pe care le face sistemul în funcție de comenzile primite de la utilizator.

În Capitolul 6, „Testare și Validare”, sunt prezentate modurile prin care s-a testat eficiența și precizia aplicației și algoritmilor. Aici se găsesc date concrete despre viteza procesării algoritmilor, structurate în tabele și diagrame pentru o mai bună lizibilitate. Sunt descrise și cele două metode de comparație a algoritmilor din punctul de vedere al calității fuziunii: comparația obiectivă și comparația subiectivă.

În Capitolul 7, „Manual de Instalare și Utilizare”, sunt prezentate cerințele minime de hardware si software pentru a instala și rula aplicația. Este descris în detaliu procesul de instalare, iar modurile de utilizare sunt explicate prin ajutorul screenshot-urilor însoțite de text care ajută utilizatorii în procesul de a înțelege cum funcționează aplicația, și cum poate ajunge la rezultatele dorite.

În Capitolul 8, „Concluzii”, este descrisă contribuția mea personală la acest proiect, analiza critică a rezultatelor obținute și o listă cu îmbunătățiri ulterioare posibile.

În capitolul cu titlul „Bibliografie” sunt enumerate sursele de inspirație pentru realizarea acestui proiect.

Capitolul 2. Obiectivele proiectului

2.1. Domeniul temei

Imagistica medicală este o ramură a ingineriei biomedicale. Prin diferite tehnici de înregistrare a imaginilor medicale asupra organelor și țesuturilor, se pot diagnostica mai ușor bolile în organismele vii. Se pot folosi atât în domeniul medical cât și în cel științific. Există mai multe instrumente pentru efectuarea acestor procedee care au propriile avantaje și dezavantaje. Se folosesc proprietăți chimice sau fizice care pot induce crearea imaginii medicale în scopul de a obține date importante. Datele pot fi codificate în mai multe moduri: imagini 2D, 3D, imagini de spectru sau liste de valori.

Acest sistem se dorește a ajuta tehnologia medicala să ajungă cel puțin la nivelul dispozitivelor mobile de astăzi, accentuând faptul că și domeniul medical necesită dezvoltare, avansare condusă de persoane pasionate. Obiectivul final este limitarea timpului pierdut, care de multe ori poate fi crucial în analizarea și diagnosticarea imaginilor medicale.

Țelul aplicației este de a crea, dintr-o imagine RMN sau CT și o imagine de tip ecografie, o a treia imagine, care conține informații mai relevante decât oricare din primele două. Pentru acesta se vor aborda următoarele clase de metode de fuziune și cinci implementări în total:

- Metode aritmetice
  - Valoarea minimă
  - Valoarea maximă
  - Valoarea medie
- Metode piramidale
  - Piramida Laplaciană
- Metode bazate pe transformata Wavelet
  - Transformata Discretă Wavelet Haar

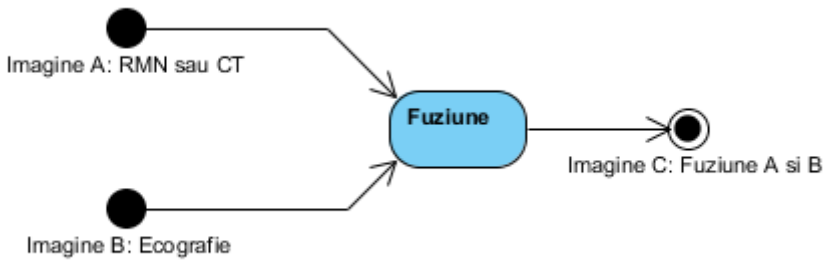

Figura 2.1.1: Fuziunea imaginilor

Pentru a compara aspectul vizual al rezultatelor, se va face un sondaj cu cele mai bune rezultate, pe care le vor puncta voluntarii, fără să știe în prealabil ce algoritm s-a folosit pentru calcularea acestuia. Imaginile rezultate se vor putea compara cu o imagine „perfectă”. Lângă această măsurare mai mult subiectivă, se vor aplica și doi metrici de măsurare a calității imaginilor:

- Eroarea medie pătratică (Mean squared error – MSE)
- Raportul între semnalul de vârf și zgomot (Peak signal to noise ratio – PSNR)

Proiectul face parte dintr-un proiect mai complex, de analiza, fuziunea și reconstrucția imaginilor medicale atât în 2D, cât și în 3D, din care abordează partea de fuziune a imaginilor (modulul de Image Fusion din imaginea de mai jos).

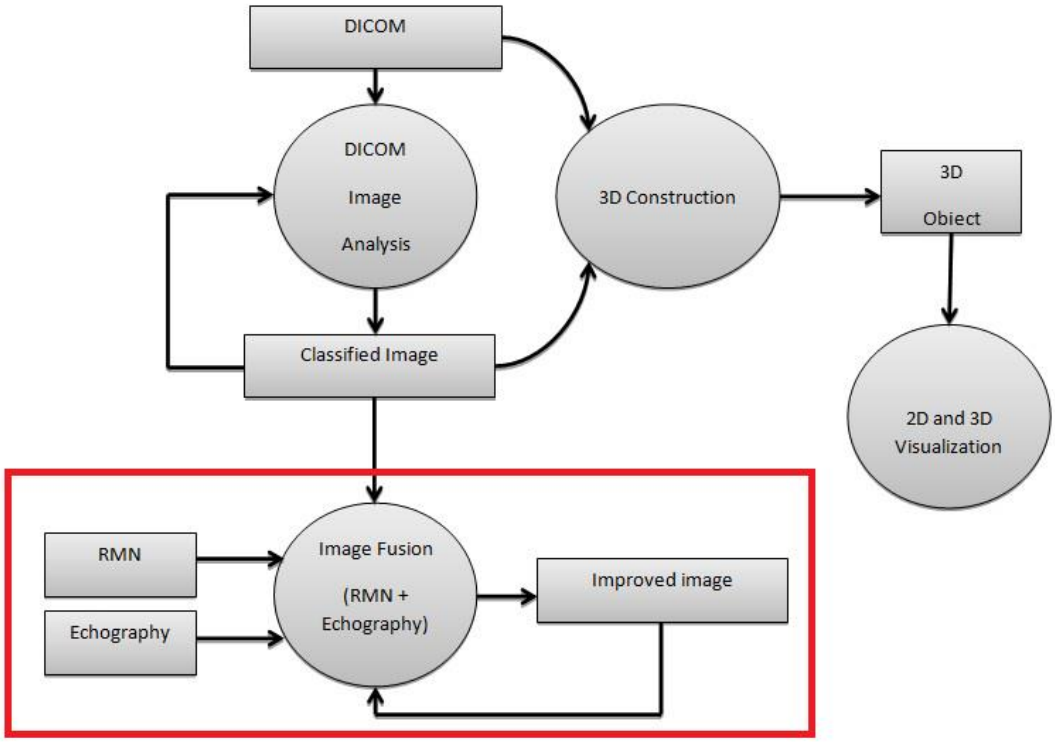

Figura 2.1 Proiectul de analiza, fuziunea și reconstrucția imaginilor medicale

Scopul acestui proiect este de crea o aplicație care servește la două funcționalități: fuziunea imaginilor medicale și studiul diferiților algoritmi pentru a efectua acest lucru. Aceste două părți nu sunt complet independente, compararea algoritmilor folosind modulul de fuziune, care conține și logica de procesare a imaginilor.

Necesitatea acestei aplicații constă în faptul că extragerea și combinarea informației din două surse diferite, dar care conțin date despre același obiect/subiect (în cazul nostru partea de corp al aceluiași pacient), poate ajuta la o diagnoză mai precisă. Pe de altă parte, minimizând expunerea unui pacient la metode cu raze intruzive, dar cu rezoluție mai bună (cum ar fi metoda RMN sau CT), este benefic din punctul de vedere al sănătății acestuia.

Conform ideii de mai sus, procesul de imagistică medicală s-ar transforma în aplicarea unei metode de rezoluție superioară, dar intruzivă, la începutul stagiului de consultanță, după care se fac mai multe imagini de calitate mai redusă, dar non-intruzive pentru corpul uman. După fuziunea imaginii inițiale cu imaginea făcută într-un stadiu mai avansat, va rezulta o a treia imagine, cu informații mai precise și mai actuale decât oricare din sursele precedente. Prin acest mod s-ar reduce efectul negativ cauzat de metodele intruzive, dar și timpul petrecut la centre medicale, metodele RMN și CT având durata de aplicare foarte mare (spre exemplu, un RMN pentru o parte relativ mică a corpului, umărul, durează 50 de minute, și este și inconfortabil).

Bazându-se pe modulul de fuziune propriu-zis de mai sus, se proiectează și un modul de comparare a diferiților algoritmi, din punctul de vedere al calității imaginii de ieșire, folosind mai mulți metrici de măsurare a calității. Această măsurare se dorește a fi atât obiectivă, creând un proces algoritmic care va avea ca ieșiri valorile pentru calitatea imaginilor, cât și subiectivă, care va consta din analizarea imaginilor rezultat de către mai mulți voluntari, acordându-le note.

Ca parametri de intrare, aplicația de fuziune folosește în principal fișiere DICOM, pentru că imaginile medicale moderne sunt codificate conform acestui protocol. Modulul de măsurare a calității algoritmilor va folosi ca intrare o listă de imagini JPEG deja incluse în proiect.

2.2. Cerințe funcționale

1. Încărcarea a două imagini de tip DICOM.
   Aplicația trebuie sa fie capabilă să încarce două fișiere de tip DICOM, care să conțină date despre imaginile medicale. Acestea vor fi extrase pentru a putea realiza fuziunea.
2. Afișarea imaginilor încărcate
   Aplicația trebuie să afișeze informațiile despre imaginile extrase din fișierele DICOM în ferestre noi. Pentru o stivă de imagini, trebuie să adauge o opțiune de a vizualiza toate imaginile din stivă.
3. Personalizarea parametrilor algoritmului de fuziune.
   Parametri algoritmilor de fuziune, care vor afecta rezultatul procesului, trebuie să fie modificabili din interfața grafică a utilizatorului. Parametri sunt limitați într-un interval care se consideră a fi utilizabil pentru scopul acestui proiect.
4. Personalizarea metodei de post-procesare.
   Utilizatorul trebuie să aibă posibilitatea să aleagă printre mai multe metode de postprocesare a rezultatului, printre care trebuie să fie prezenta și opțiunea de non-postprocesare (afișarea rezultatului „crud” după fuziune).
5. Afișarea rezultatelor în ferestre.
   Aplicația trebuie să afișeze rezultatele fuziunii în ferestre noi, păstrând ferestrele cu imaginile de intrare, precum și rezultatele precedente ale fuziunii.
6. Salvarea rezultatelor la alegerea utilizatorului.
   Aplicația trebuie să ofere posibilitatea de salvare a rezultatelor procesului de fuziune, în format de imagine obișnuit (.jpg).
7. Afișarea unei pagini de ajutor sau instrucțiuni.
   Aplicația trebuie sa afișeze o pagină de ajutor cu pașii principali și sfaturi asupra utilizării sistemului.
8. Rularea metodei de măsurare a calității algoritmilor.
   Aplicația trebuie să-i ofere posibilitatea utilizatorului să ruleze procesul de măsurarea calității a algoritmilor de fuziune.
9. Personalizarea modului de salvare a rezultatelor.
   Aplicația trebuie să ofere o posibilitate de a alege în ce format se dorește salvarea rezultatelor procesului de măsurare a calității.
10. Salvarea rezultatelor.
    Aplicația trebuie să salveze în fișiere separate, pe mașina locală, rezultatele proceselor.

2.3. Cerințe non-funcționale

1. Viteză
   Aplicația trebuie să efectueze pe un sistem performant, fuziunea imaginilor, în mai puțin de 2 secunde. Această viteză va depinde și de puterea de calcul a mașinii pe care se rulează programul. Procesul de măsurare a calității să nu dureze mai mult de 12 secunde, incluzând și salvarea rezultatelor în fișiere.
2. Precizie
   Aplicația trebuie să producă rezultate consistente, pentru aceleași valori de intrare, atât pentru fuziune cât și pentru măsurare a calității.
3. Funcționare independentă de platformă
   Aplicația trebuie să funcționeze pe orice sistem care are Java Runtime Environment instalat. Sistemul nu include componente care țin de o singură platformă.
4. Portabilitate
   Aplicația trebuie să fie portabilă. Prin acest lucru se înțelege faptul că ajunge ca să se copieze fișierul executabil al aplicației pe alt sistem, fără a fi nevoie de configurări specifice.
5. Funcționare offline
   Aplicația trebuie să funcționeze la orice moment dat, fără a avea nevoie de o conexiune la internet.

2.4. Componente

Partea cea mai complexă și importantă din punctul de vedere al funcționarii corecte a aplicației este partea algoritmică, care implementează algoritmii de fuziune. Aceștia sunt folosiți la fuziunea propriu-zisă a doi matrici de informații codificate folosind numere întregi, care în cazul nostru vor fi reprezentate de matrici de pixeli. Această componentă definește logica de bază a funcționării aplicației, și pe ea se bazează în mare parte și celelalte module.

Componenta de măsurare a calității este importantă din punct de vedere al comparării algoritmilor menționați mai devreme, folosind moduri de verificare a calității obiective. Ea va itera peste algoritmii implementați în modulul de mai sus, folosindu-le la fuziunea imaginilor cu mai multe opțiuni, pentru a primi o gamă largă de rezultate.

Componenta care leagă cele două amintite mai sus, este interfața grafică. Ea oferă o legătură între logica aplicației și utilizator, folosind un mod de vizualizare și interacțiune simplă și intuitivă. La rândul sau, și interfața expune diferitele opțiuni pentru rularea procesului de fuziune sau metodei de măsurare a calității.

2.5. Datele de intrare și ieșire

Aplicația va permite ca date de intrare trimise de către utilizator, fișiere de tip DICOM (.dcm sau alte formate specifice). E important de menționat faptul ca unele fișiere DICOM compuse dintr-o stivă de imagini, nu au nici o extensie de fișier. Aplicația acceptă și imagini de acest tip, verificându-le structura corectă pentru a fi procesate pe urmă. La fel, din motive de testare și depanare, se pot încărca și imagini de format obișnuit (.jpg, .png, .gif).

Ca date de ieșire, se primesc imagini fuzionate care se pot vizualiza în ferestre, și se pot salva în format .jpg. Rezultatele modulului de măsurarea calității vor fi salvate ca text (.txt) sau ca fișier Excel (.xls).

Capitolul 3. Studiu Bibliografic

3.1. Domeniul în care se situează tema

Aplicația având scopul îmbunătățirii metodelor de diagnoză medicală, trebuie studiată mai în detaliu acest domeniu.

3.1.1. Imagistica medicală

În articolul [1] scrie că imagistica medicală este un domeniu științific relativ recent, al cărui scop principal este extragerea informațiilor importante din organisme vii, folosind metode fizice sau chimice.

In capitolul 2, subcapitolul 2.1 din [2], sunt menționate următoarele metode de imagistică medicală: radioscopia, radiografia, tomografia, ultrasonografia, imagistica prin rezonanță magnetică și tomografia prin emisie de pozitroni. Dintre acestea, unele metode implică riscuri din punctul de vedere al sănătății pacientului.

Radioscopia, deși este metoda cea mai ieftină, poate iradia bolnavul. O alternativă mai bună la aceasta este radiografia, care însă necesită numeroase filme pentru a urmări precis funcționarea unor organe. Ultrasonografia deși este cea mai sigură, produce rezultate care sunt mai greu de interpretat, din cauza faptului că reprezintă organele interne într-o formă nenaturală. Tomografia este mai ieftina decât imagistica prin rezonanța magnetică, este superioară pentru imagini ale craniului uman, dar produce radiați, deci nu poate fi aplicată la pacienți cum ar fi femeile gravide sau bolnavi cu diabet. Rezonanța magnetică este probabil cea mai scumpă metodă, este foarte precisă. Procesul în sine poate cauza un sentiment de claustrofobie, este lung și sensibil la mișcări. La fel, undele magnetice sunt invazive pentru corpul uman.

Luând în considerare avantajele și dezavantajele prezentate mai sus, putem ajunge la concluzia că nu există o singură metodă care e perfectă din toate punctele de vedere. Din această cauză, fuziunea rezultatelor de la diferite surse de imagistica medicală ar putea fi soluția cea mai avantajoasă și pentru bugetul pacientului, dar și din punctul de vedere al diagnosticului corect.

3.1.2. Standardul DICOM

Din articolul [3], putem afla că standardul DICOM (Digital Imaging and Communications în Medicine) este folosit pentru stocarea, comprimarea și transmiterea datelor medicale. Acest standard definește modul de stocare fizic a informațiilor și un protocol de comunicare pe rețea, bazat pe TCP/IP. Fișierele DICOM conțin, de obicei, date în format textual despre pacient și mediul de efectuare al imaginii, și una sau mai multe imagini.

Datele sunt combinate în așa fel, încât numărul identificator al pacientului nu se poate separa de imaginea (imaginile) lui. Partea textuală a fișierului poate conține mai mult de 3300 de etichete (tag) specifice imaginii, pacientului sau instrumentului folosit, numite și atribute. Câteva exemple de atribute:

- Imagine
    - Datele conținute de pixeli
    - Lista de cadre (frame)
    - Începutul decupării
    - Sfârșitul decupării
    - Tipul cadrelor
    - Proprietăți volumetrice
- Pacient

  - Numele pacientului
  - Numărul său identificator
  - Vârsta
  - Grupa sanguină
  - Alergii
  - Grupa etnică
  - Fumător sau nefumător
  - Tipul terapiei
- Instrumentul folosit
  - Firma producătoare
  - Numărul identificator al aparatului
  - Modalitatea efectuării imaginii
  - Numele institutului
  - Adresa institutului

Câteva dintre etichete pot apărea de mai multe ori în fișier. Imaginea propriu-zisă este reprezentată în eticheta de datele pixelilor. Deși acest atribut poate fi prezent doar o dată în fișier, putem avea mai multe imagini asociate cu un DICOM. Diferența între imagini se face prin precizarea numărului și așezării cadrelor.

Pentru a ușura și a optimiza afișarea imaginilor pe dispozitive diferite, standardul definește un tabel de căutare după valoarea pixelilor DICOM, Grayscale Standard Display Function. Pentru a putea afișa corect datele conținute de pixeli, dispozitivele trebuie să conțină această funcție sau să fie calibrate pentru acest fel de afișare.

Pentru comunicare, standardul DICOM folosește TCP sau UDP și portul 104. Standardul definește și o serie de servicii pentru a ușura comunicarea pe rețea: stocare, căutare, listare și definirea unei operații automate de imagistică.

Unul din dezavantajele formatului DICOM posibilitatea definirii a unui număr prea mare de atribute opționale și completarea unor câmpuri cu informații greșite. Prin acest fel, se pierde din consistența combinației text-imagine.

3.1.3. Fuziunea imaginilor

In articolul [4], se găsește definiția acestei tehnologii teoretice. Fuziunea imaginilor este procesul prin care se combină datele relevante din două sau mai multe imagini sursă într-o singură imagine, în care rezultatul va conține informații mai complete și mai precise decât oricare din imaginile de intrare. În acest mod, fuziunea îmbunătățește calitatea rezultatelor pentru un anumit domeniu.

În medicină [5], fuziunea imaginilor devine din ce în ce mai folosită pentru a îmbunătăți efectele și a reduce durata tratamentelor. Imaginile fuzionate pot fi create folosind ca intrare rezultatele aceleiași tehnologii de imagistică, sau combinând modalitățile de achiziționare a datelor despre pacient. Imaginile rezultate, consistente în informațiile pe care le expun, sunt importante mai ales pentru detecția cancerului în corpul uman.

3.2. Aplicații similare

Aplicații prin care se poate efectua fuziunea imaginilor medicale:

Mirada Medical XD3 [6], care printre altele, oferă unelte pentru vizualizarea, înregistrarea, manipularea și segmentarea diferitelor imagini.

Velocity Medical [7], special conceput pentru oncologie, oferă integrarea ușoara a imaginilor de la tratamente diferite, cum este radioterapia și brachoterapia.

Keosys Imagys-Cloud Services [8], pune la dispoziție o aplicație de vizualizarea și fuziunea imaginilor medicale 3D.

În studiul [9], putem găsi o comparație între metode de fuziune bazate pe transformata Wavelet. Sunt comparate fuziunile pe baza regulii mediei aritmetice, valoarea maximă, pe baza contrastului, pe baza transformatei Wavelet pachet și pe baza contrastului combinată cu transformata Wavelet pachet. Din lucrare reiese că transformata Wavelet cu fuziunea pixelilor pe baza valorii medii produce cele mai bune rezultate dintre acești algoritmi. Din concluzia autorilor putem citi că metodele studiate prezervă structura importantă a organelor și tumorilor, însă se pierde din contrast și informația de pe muchii. Această problemă se poate remedia folosind transformata pe bază de contrast.

În studiul [4] s-au comparat metodele de valoarea medie aritmetice, valoarea maximă, piramida Laplaciană, piramida morfologică, PCA (Principal Component Analysis), transformata Wavelet Discretă, metoda anterioară combinată cu PCA și metoda combinației pixelilor cu regula fuziunii de energie. Dintre aceștia, primele trei operează în domeniul spațial, iar ultimele trei în domeniul transformatei. Metodele spațiale oferă o rezoluție spațială mare, dar introduc problema de estompare (blur) pe imagine. O soluție la această problemă este sugerată a fi folosirea transformatei Wavelet, care oferă rezultate de calitate spectrală ridicată. Autorii lucrării propun ca soluție ideală combinarea metodelor de transformare cu metode spațiale, și prin experimentele prezentate în articol susțin părerea că aceasta ar putea fi soluția pe viitor în domeniul fuziunii imaginilor.

3.3. Framework-uri propuse

In cele ce urmează, se prezintă lista framework-urilor dintre posibilitățile care s-au studiat, și care s-au dovedit cele mai potrivite pentru implementarea acestui proiect.

3.3.1. ImageJ

ImageJ este o aplicație de procesare și analizare a imaginilor, scrisă în limbajul Java. Acesta, conform [10], aduce cu sine și independența de platformă, programul putând fi rulat atât pe sisteme Linux, cât și pe Max OSX și Windows, 32 sau 64 de biți. Programul și codul său sursă este open-source, fără a avea nevoie de o licență pentru el. Este extensibil prin framework-ul robust și documentat, oferit de echipa dezvoltatoare, prin care se pot crea noi plugin-uri și programe de sine stătătoare folosind această tehnologie. În momentul de față există deja 500 de plugin-uri pentru ImageJ. Pe site-ul oficial, se susține faptul că ImageJ este cel mai rapid program de procesarea imaginilor scris în Java. Programul poate deschide și salva imagini de formatul GIF, JPEG, BMP, PNG, PGM, FITS și poate deschide fișiere DICOM, care este folositor pentru scopul nostru.

Programul suportă modificarea și filtrarea imaginilor deja deschise, punând la dispoziție câteva module deja definite, dar permițând și integrarea funcționalităților noi în aplicație. Poate procesa imagini pe 8, 16 sau 32 de biți. Se pot aplica operații de măsurare a diferitelor suprafețe, lungimi și unghiuri.

Printre altele, framework-ul oferă și funcționalitate de citirea imaginilor și stivelor de imagini în format DICOM, prelucrarea lor, extragerea informațiilor din aceste fișiere, și separarea lor în date despre pacient și imaginea efectivă.

3.3.2. JExcel API

Acesta este un modul Java integrabil [11], care pune la dispoziție funcții de citire, scriere și modificare a fișierelor Excel. Suportă formate de Excel 95, 97, 2000, XP și 2003. Poate citi și scrie și formule specifice documentelor excel. Generează foi de calcul și suportă formatul de literă, număr sau dată calendaristică. Pe lângă acestea, oferă o interfață logică și ușor de înțeles pentru dezvoltatori, includerea lui în proiecte fiind la fel ușoară.

3.3.3. Java Swing

Swing este un framework de proiectare a interfețelor grafice de utilizator pentru limbajul Java. Din articolul [12] găsim modurile în care se poate folosi acesta. Se alege un cadru, „JFrame”, care se poate popula cu diferite elemente vizuale, cum ar fi câmpuri de date de intrare, butoane, liste, meniuri și poze sau alte animații. Pe aceste elemente se pot scrie diferite acțiuni, care pot interacționa cu logica din spatele aplicației. Fiecare element vizual poate intercepta acțiunile utilizatorului în mai multe feluri, fie apăsarea unei taste, click-ul pe element sau scroll-ul asupra paginii. Acestea se procesează de către manipulanții de evenimente (EventHandler-uri), care sesizează acțiunea ce a avut loc și efectuează operațiile ce sunt scrise pe acțiunea respectivă.

3.3.4. JUnit

Pentru a asigura calitatea și funcționarea corectă a sistemului, se recomandă a folosi teste de unitate pentru verificarea răspunsului modulelor aplicației la diferite date de intrare și în diferite condiții. JUnit, 13, este un framework de testare pentru limbajul de programare Java. Este esențial în metodele de dezvoltare bazate pe teste (test drive development) [14]. Folosind aceasta unealtă, se dorește a automatiza testarea aplicației cât mai în detaliu, în acest mod garantând comportamentul corect al acesteia.

3.3.5. Maven

Maven [15], este o unealtă pentru administrarea, automatizarea și configurarea procesului de construire (build) a unei aplecații Java. Pe lângă astea, poate fi folosit și pentru a defini dependințele unei aplicații, pe care le poate descarcă de pe un server global, și le integrează în proiect. Unealta se configurează dintr-un fișier XML numit „pom.xml”, prin care se poate preciza ordinea proceselor de construire a codului aplicației, cum vor fi rulate testele, cum va arată rezultatul compilării și în ce structură de foldere va fi pus.

În cazul nostru, modulele ImageJ, JExcel API și JUnit vor fi declarate și descărcate folosind Maven, iar testele vor fi automatizate tot din script-ul scris în „pom.xml”. Aceasta oferă o independență a proiectului față de editorul de cod pe care-l folosim, procesul de build având comportament identic pe orice platformă sau editor.

3.4. Concluzie

Studiind aceste tehnologii științifice și tehnice, am ajuns la concluzia că ele sunt o soluție bună pentru aplicația software prin care se dorește susținerea afirmațiilor teoretice din această lucrare. Probabil se va pune întrebarea, de ce s-a ales limbajul Java pentru construirea aplicației, și nu limbajul C sau C++, care s-a dovedit a fi superior din punctul de vedere al vitezei de procesare în general, și mai ales în cazul procesării imaginilor.

În primul rând, framework-ul ImageJ ne pune la dispoziție o serie de module deja implementate, care ne ajută în citirea fișierelor DICOM, prelucrarea acestora și

procesarea imaginilor extrase din ele. Scopul aplicației nu este de a avea o structura unică, implementată de la zero, ci de a folosi modulele deja existente, cuplându-le în așa fel încât să ofere o soluție cât mai eficientă la rezolvarea unor anumite probleme, în cazul nostru, fuziunea imaginilor medicale. Aplicația în sine este scrisă având în vedere atributul de a fi reutilizabil în viitor, metodele și clasele fiind extensibile și configurabile ușor.

În al doilea rând, având interfața și modulele de procesare din spate în Java, aplicația poate fi rulată pe orice platformă. O soluție scrisă, de exemplu, în C++ cu domeniul de prezentare implementat în C#, ar fi avut constrângerea sa fie rulat doar pe sisteme Windows. Proiectul se dorește a fi open-source, scopul principal fiind ca alți dezvoltatori sau cercetători să-l integreze cu ușurința pe platforma lor în care lucrează cel mai eficient.

Un alt aspect, la fel relativ important, este experiența deja dobândită în cadrul facultății, în limbajul Java. Prin acest mod, dezvoltarea efectivă aplicației a rezultat într-un ritm mai productiv de scriere a codului, pe care s-au aplicat și practicile cele mai bune învățate la cursuri și la laboratoare. La fel, integrarea și folosirea framework-urilor alese a fost mai facilă, tot din cauza experienței dobândite deja la numeroase proiecte în Java.

Capitolul 4. Analiză și Fundamentare Teoretică

4.1. Flux general

Fuziunea imaginilor constă din mai mulți pași importanți, fiecare fiind grupat într-un modul separat cu eventuale submodule:

1. Citire imagini DICOM
2. Fuziune imagini
3. Postprocesare rezultat
4. Afișare rezultat

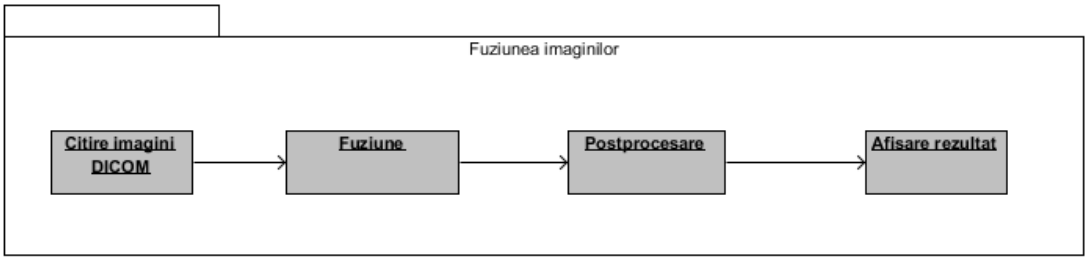

Figura 4.1 Fluxul general al procesului de fuziune

Un fir separat de execuție reprezintă rularea procesului de măsurarea calității a algoritmilor:

1. Încărcare imagini de test
2. Adăugare algoritmi care se doresc comparate
3. Calculare metrici de calitate
4. Formatare și scriere rezultate în fișiere

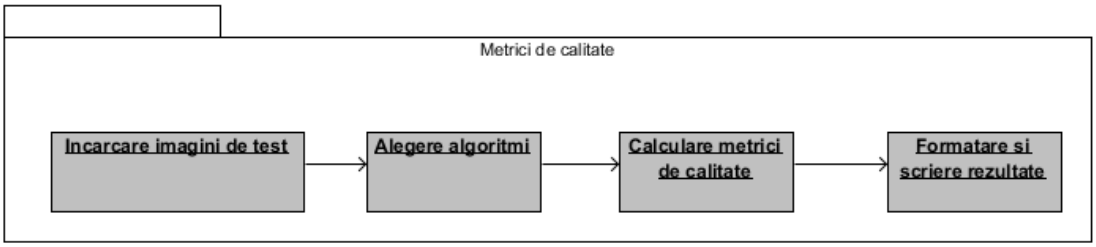

Figura 4.2 Fluxul general al procesului de măsurarea calității

Ambele procese folosesc același modul de algoritmi și metode de postprocesare. La fiecare proces, se pot alege algoritmii exacți care vor fi folosiți.

4.2. Date de intrare și ieșire

4.2.1. DICOM

Tipul de date de intrare cel mai important este fișierul DICOM, având de obicei extensia „.dcm”, dar unele informații stocate ca o stivă de imagini pot avea extensia goală. Din această cauză trebuie verificat fișierul încărcat din punctul de vedere al conținutului.

In articolul [16] putem citi că fișierul poate fi despărțit într-un grup de antet (header) și un set de date, care reprezintă de obicei datele despre imagine. Antetul este compus dintr-un preambul de 128 de octeți urmat de un prefix DICOM de 4 bytes.

Prefixul este compus din caracterele `D`, `I`, `C`, `M`, codate ca litere mari din ISO 8859 G0 Character Repertoire. Nu există constrângerea ca preambulul să fie structurat într-un anumit fel, este gândit doar pentru a facilita accesul la informațiile conținute de fișier.

Un set de date reprezintă un obiect care conține informații din lumea reala. Ele pot conține valorile codificate a atributelor obiectului. Aceste atribute sunt specificate în definițiile posibile ce pot fi incluse într-un set de date.

Un element de date este conținutul unui set de date. Este identificat unic printr-o etichetă, este ordonat crescător după această etichetă și poate fi prezent maxim o dată într-un set de date. Sunt două tipuri de elemente de date: standard și privat. Elementele standard au numărul de grup un număr par, iar elementele private număr impar.

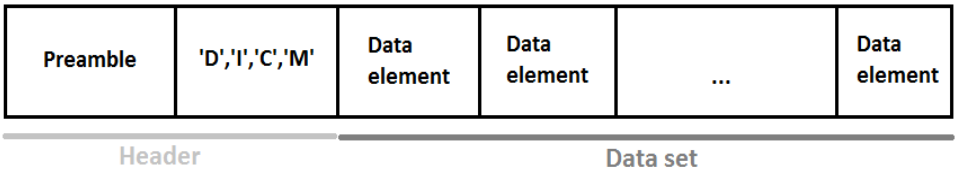

Figura 4.3 Structura fișierului DICOM

4.2.2. JPEG

Rezultatul procesului de fuziune va fi o imagine de format obișnuit, în prima fază afișată pe interfața grafică, și apoi putând fi salvată în format JPEG (.jpg).

În cursul [17] găsim că Joint Photographic Experts Group a dezvoltat acest format, care stochează informația în tip bitmapped, și poate avea următoarele formate: jpg, jpeg, jfif, jfl. JPEG folosește propriul algoritm de compresie, prin care se reduce semnificativ dimensiunea fizică a fișierului de imagine, dar se pierde și din calitate. Formatul suportă culori până în 24 de biți, și poate fi folosit pe orice platformă.

Acest format de imagine va fi și intrarea la procesul de măsurarea calității. Se folosește JPEG în acest modul din cauza faptului ca să se ofere mai multă libertate utilizatorului, în cazul în care acesta dorește sa experimenteze cu diferite set-uri de imagini de intrare, în scopul să vadă comportamentul algoritmilor de fuziune. Formatul DICOM fiind relativ greu de modificat fără software specific, s-a ales formatul JPEG care este ușor de manipulat cu orice aplicație de editare a imaginilor.

4.2.3. Text

O posibilă valoarea de ieșire a procesului de măsurare a calității este salvarea rezultatelor în format text, cu extensia .txt. În acest fel, informația se poate accesa cu orice editor de text, independent de platformă, însă are dezavantajul că nu poate fi ordonat după dorințele utilizatorului.

4.2.4. Excel

Cea de-a doua ieșire a măsurării calității este un fișier Excel, cu extensia .xls. În acesta se poate introduce informația culeasă într-o structură logică, curată și mai ales sortabilă după diferite coloane. Acesta, la fel este independentă de platformă, însă

trebuie deschis cu software special, cum ar fi Microsoft Excel pentru Windows, Libre Office Calc pentru Ubuntu/Linux și Office Web Apps pentru MacOS.

4.3. Algoritmi

Algoritmii propuși pentru comparația preciziei și corectitudinii a fuziunii imaginilor sunt:

- Transformata Discretă Haar Wavelet
- Piramida Laplaciană
- Metode aritmetice
    - Valoarea minimă
    - Valoarea maximă
    - Valoarea medie

4.3.1. Transformata Discreta Haar Wavelet

Acest algoritm este o implementare a transformatei Haar.

4.3.1.1. Transformata Wavelet Discretă

Acest tip de transformare împarte un semnal sursă în componente diferite de timp-frecvență [18]. Un semnal unidimensional de obicei se reprezintă în domeniul temporal, iar un semnal bidirecțional, cum sunt și imaginile, se reprezintă în majoritatea cazurilor în domeniul spațial. Acest al doilea tip de semnal are ca o reprezentare alternativă reprezentarea în domeniul frecvențial. Cele două moduri de reprezentare au propriile avantaje și dezavantaje. În domeniul spațial, reprezentarea este ușor de înțeles pentru percepția umană, filtrarea se aplică direct pe datele spațiale (nu necesită transformare), însă filtrele au nuclee mari și de obicei timpul de procesare este, la fel, mai mare. În domeniul frecvențial proiectarea nucleelor de filtrare este mai ușoară, filtrarea este mai rapidă însă reprezentarea în sine este non-intuitivă pentru ochiul uman și filtrările necesită o transformare în domeniul frecvenței și înapoi în domeniul spațial. Trecerea din primul domeniu în celălalt și invers se poate realiza cu transformata Fourier directă și inversă [19].

Transformata Wavelet combină cele două domenii (temporal și frecvențial), rezultând o aproximare atât în timp cât și în spațiu a semnalului . Se sacrifică o parte din precizia frecvențială a transformatei Fourier pentru a obține informații și despre componenta temporală a semnalului.

Există două metode de transformare Wavelet: transformata Wavelet continuă și discretă. Rezultatele obținute pentru prima au o precizie mai mare, efectuând operații redundate pe un semnal de intrare. A doua este mai rapidă, combinând perechile de date dintr-un semnal într-un mod mai eficient, însă cu pierderi minore de informație. Transformata Wavelet discretă este, din punct de vedere de procesare, mai puțin complex decât transformata Fourier, având timp de procesare O(n) fata de O(n * logn).

4.3.1.2. Transformata Haar Wavelet

Această transformare este o implementare a transformatei Wavelet discrete. A fost propus de către Alfréd Haar în 1909, este cea mai simplă implementare a wavelet-urilor. Dezavantajul este ca transformata nu e una continuă, deci nu este diferențiabilă (nu are derivate în oricare punct al domeniului). Această proprietate poate servi și în avantajul procesării, la analiza semnalelor cu o tranziție bruscă.

Transformata Haar poate fi descrisă utilizând următoarele două funcții:
Funcția wavelet $\psi(t)$:

$$\psi(t) = \begin{cases} 1, & 0 \le t < 1/2 \\ -1, & \frac{1}{2} \le t < 1 \\ 0\ in\ caz\ contrar \end{cases}$$

Funcția de scalare ϕ(t):

$$\phi(t) = \begin{cases} 1, & 0 \le t < 1 \\ 0\ in\ caz\ contrar \end{cases}$$

Unde t reprezintă componenta temporală.

4.3.1.3. Implementare

În lucrarea [20], găsim o soluție de implementare a tranformatei Haar Wavelet în Matlab. Transformata Haar Wavelet unidimensională desparte un semnal de intrare s(n) în două semnale, j(n) și i(n), unde j(n) reprezintă semnalele de frecvență joasă, iar i(n) semnalele de frecvență înaltă. Prima dată se filtrează semnalul cu un filtru trece jos și cu un filtru trece sus. Rezultatele filtrelor sunt sub eșantionate cu 2, și așa se obțin cele doua semnale j(n) și i(n). Transformata Wavelet Discreta Haar este invariant la deplasare.

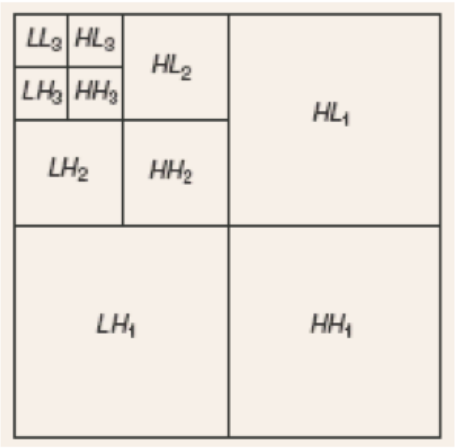

Figura 4.421, Haar Wavelet

Prima dată se parcurge matricea de pixeli pe rânduri, se calculează suma și diferența elementelor consecutive. Sumele vor fi stocate într-o jumătate a matricei, iar diferențele în cealaltă jumătate. Acest procedeu se repetă pe coloane. Acești doi pași reprezintă o iterație de transformare. Se pot aplica recursiv și pe matricea mai mică rămasă, simbolizând suma sumelor, în sensul ca e rezultatul sumelor de pe rânduri și de pe coloane.

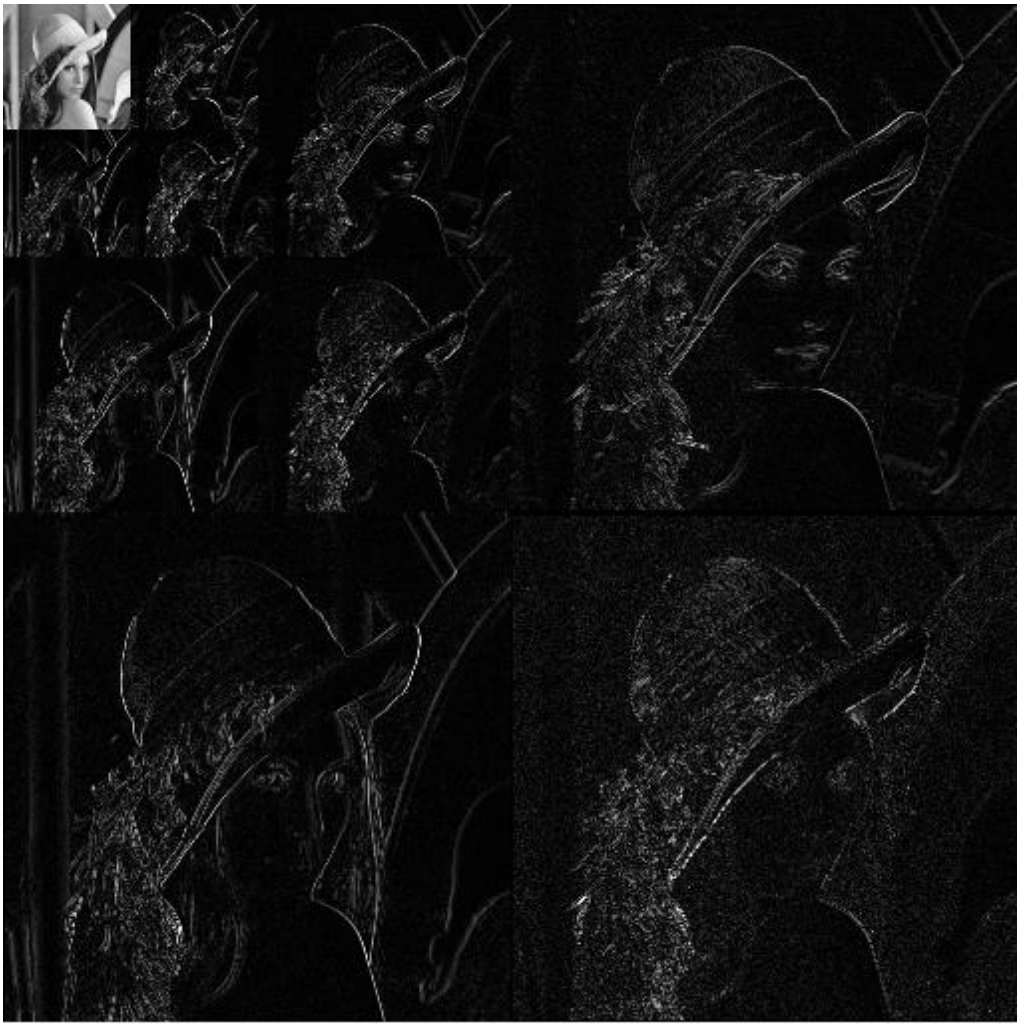

Figura 4.5 [22] Lena descompusa cu Haar Wavelet

Proprietăți ale transformatei Haar:

- Ortogonal
- Vectorii de bază sunt ordonate în timp
- Pierdere minimă de informație

4.3.2. Piramida Laplaciană

Metoda piramidală de fuziune oferă detalii mai precise în zone de contrast mare. Fuziunea propriu zisă are loc în domeniul transformatei.

Imaginea este descompusă în imagini mai mici, prin aplicarea filtrelor trece jos și trece sus, și scalarea imaginilor rezultate. Scalarea de regula divide dimensiunile imaginii de pe nivelul precedent cu 2. Imaginea de scala cea mai mică va conține frecvențele joase din imaginea sursă, iar celelalte imagini vor avea frecvențele înalte. Fuziunea se face combinând informațiile de pe fiecare nivel, iar imaginea rezultat se calculează aplicând transformata piramidală inversă pe acesta.

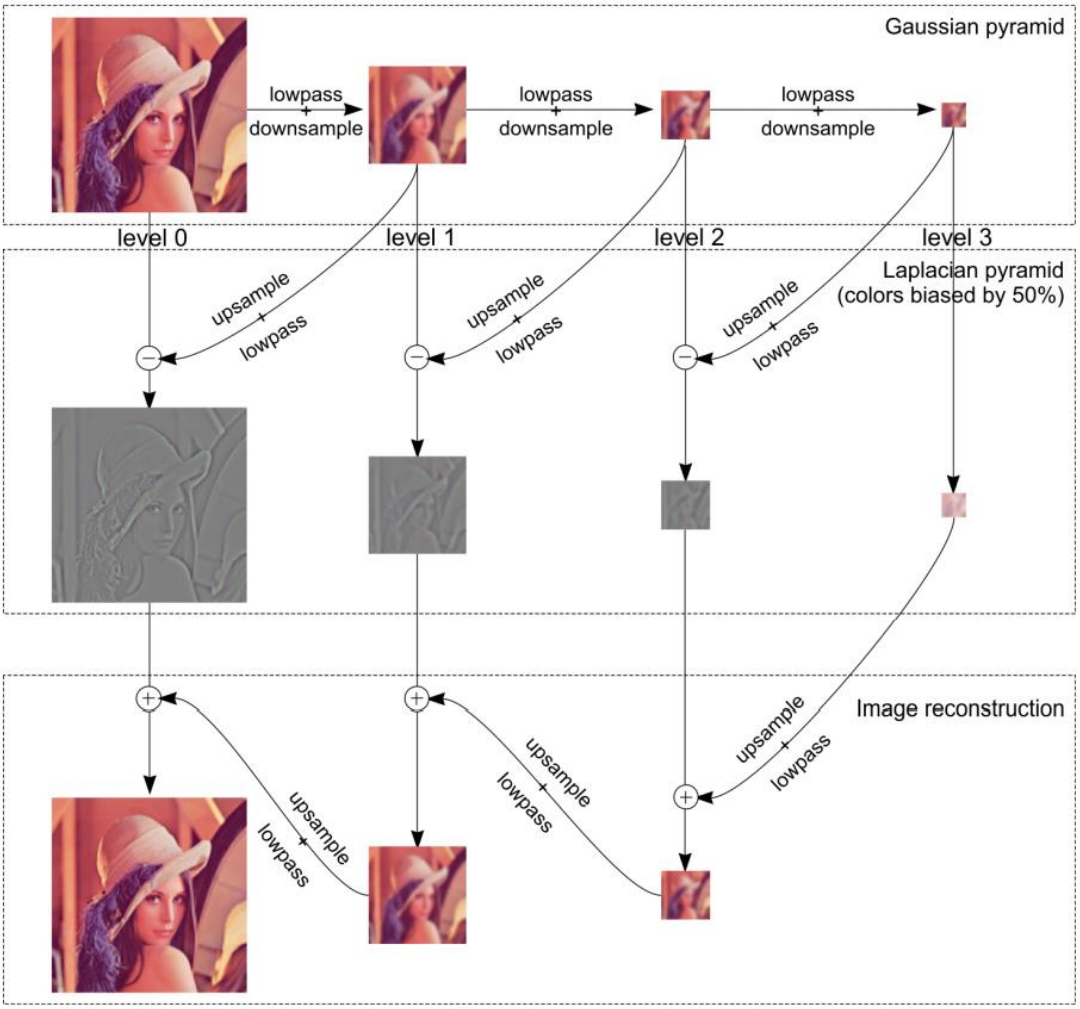

Figura 4.6 [23] Lena descompusă cu piramida Gaussiană și Laplaciană

Algoritmul de fuziune este compus din 3 părți:

1. Decompoziție: Aplicarea filtrelor trece jos și scalarea imaginilor până ce sa ajunge la un nivel dorit.
2. Fuziunea imaginilor: se combină informațiile de pe fiecare nivel folosind ori metoda aritmetică de calcularea mediei sau calcularea maximului, sau alternativ, se poate returna doar una dintre imagini, dacă se constată faptul că ar conține deja informații mai precise decât cealaltă imagine.
3. Reconstrucție: Aplicarea algoritmului piramidal invers pe nivelele de imagine rezultate. Constă din următorii subpași:
   3.1. Imaginea de pe nivelul cel mai de jos este redimensionată la 2 ori dimensiunea ei.
   3.2. Se aplică un filtru cu matricea de convoluție transpusă decât matricea aplicată la procesul de decompoziție, folosită în filtrul trece jos.
   3.3. Se fuzionează cu adunarea valorii pixelilor cu piramida de pe nivelul respectiv din decompoziție.
   3.4. Se repetă pasul acesta imaginea rezultată fiind imaginea de intrare pentru iterația următoare.

4.3.3. Metode aritmetice

Două imagini se pot fuziona și prin metode aritmetice simple, efectuând operații matematice între pixelii adiacenți.

Câteva metode aritmetice prezentate în această lucrare sunt metoda valorii minime, maxime și mediei între doi pixeli. Cum sugerează și numele metodelor, în prima se va asigna imaginii fuzionate pixelul cu intensitate mai mică dintre cele două imagini, aici vorbindu-se despre imagini greyscale. În cazul valorii maxime, se procedează invers, iar în cazul valorii medii, se va lua valoarea medie a celor doi pixeli. În cazul valorii mediei pixelul rezultat va lua valoarea mediei aritmetice a pixelilor de pe pozițiile aferente din imaginile de intrare.

4.3.4. Convertirea stivei de imagini în proiecție 2D

Presupunem că avem o stivă de imagini, care reprezintă același obiect la distantă sau adâncime diferită. Notăm dimensiunile stivei: x, y sunt lățimea și lungimea imaginilor individuale, iar z este lungimea stivei, sau numărul de imagini din stivă. In acest sistem, putem trage 3 axe imaginare, x, y și z. Proiecția imaginilor se va face pe axa z. În acest procedeu, se încearcă suprapunerea tuturor imaginilor, combinând valorile pixelilor în așa fel, încât să rezulte o singură imagine ca rezultat cu informația din toate celelalte imagini. Procedeul se poate face, de exemplu, luând valorile maxime din fiecare pixel dintre toate imaginile.

4.4. Metode de postprocesare

4.4.1. Dilatare

Din cursul [24], aflăm că dilatarea și eroziunea sunt baza operațiilor morfologice. Dacă avem doi matrici de pixeli, A și B, putem scrie formula dilatării binare A cu B:

$$A \oplus B = \{Z | (\hat{B})_Z \cap A \neq \emptyset\}$$

Sau

$$A \oplus B = \{Z | [(\hat{B})_Z \cap A] \subseteq A\}$$

Unde B – este un element structural

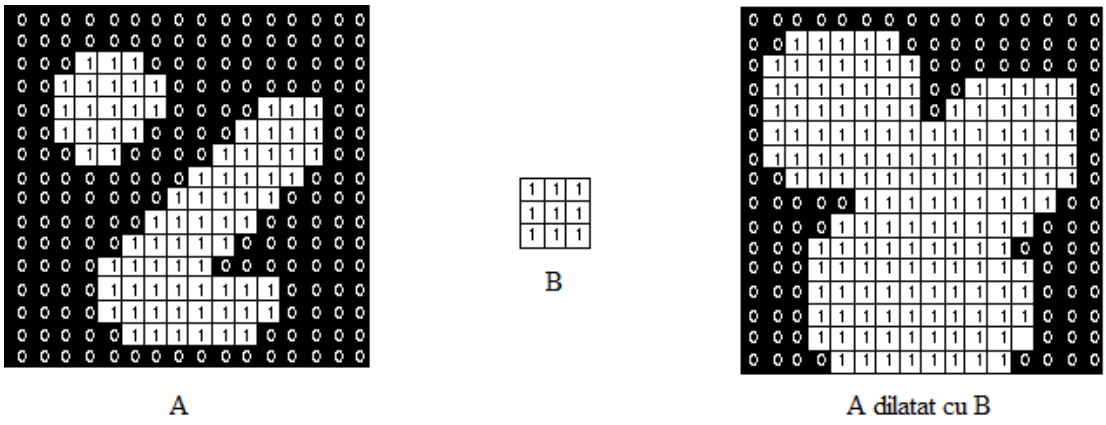

Figura 4.7 [25], Dilatare

Conform articolului [26], dilatarea imaginilor grayscale se poate scrie în felul următor:

$$(f \oplus b)(x) = \sup_{y \in E}[f(y) + b(x - y)]$$

Unde:

- f(x) este funcția de imagine
- b(x) este funcția de dilatare
- E este spațiul Eucledian
- „sup” este supremum

Practic, prin dilatare crește suprafața totală a obiectelor binare dintr-o imagine.

4.4.2. Eroziune

Tot din [26] aflăm că eroziunea și dilatarea sunt duale sau altfel spus complementare. Găsim formula pentru eroziunea binară:

$$A\Theta B = \{Z|(\hat{B})_Z \subseteq A\}$$

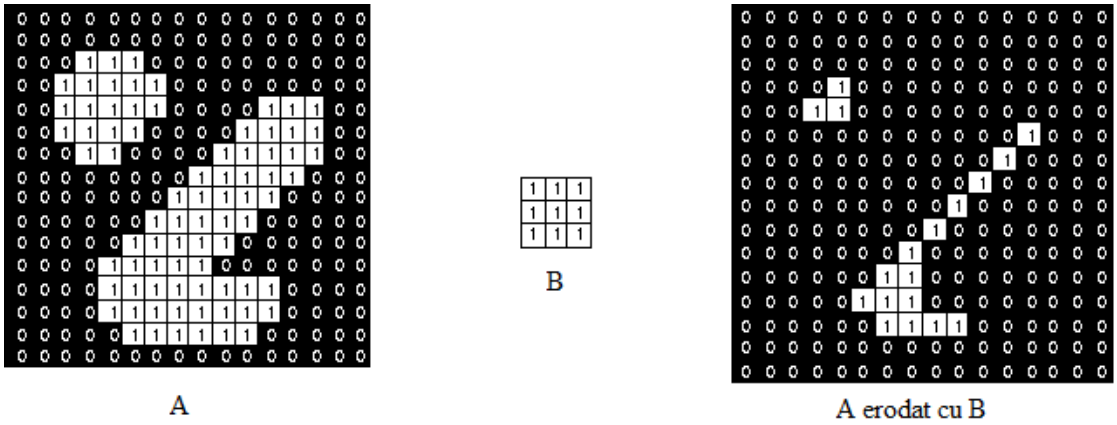

Figura 4.8 [27], Eroziune

Conform articolului [28], eroziunea imaginilor grayscale se poate scrie sub forma:

$$(f\Theta b)(x) = \inf_{y \in B}[f(x+y) - b(y)]$$

Unde B este spațiul pe care b(x) este definit și „inf” este infimum.
Practic, prin eroziune va scade aria totală a obiectelor dintr-o imagine.

4.4.3. Netezire (smoothing)

Din articolul [29], este formulată definiția procesului de netezire a imaginilor: este crearea unei aproximări în scopul de a captura șabloanele și modelele importante de date, în același timp eliminând zgomotul sau alte structuri de dimensiuni mici, cu conținut irelevant sau eronat. Se transformă semnalul de intrare într-unul mai neted la ieșire, reducând dimensiunea punctelor de zgomot, și mărind dimensiunea punctelor care sunt de valori cele mai mici între punctele adiacente.

Netezirea se poate face, de exemplu, cu un filtru Gaussian, care oferă și un efect de estompare (blur) pe imagine.

Figura 4.9 Lena, imaginea originală

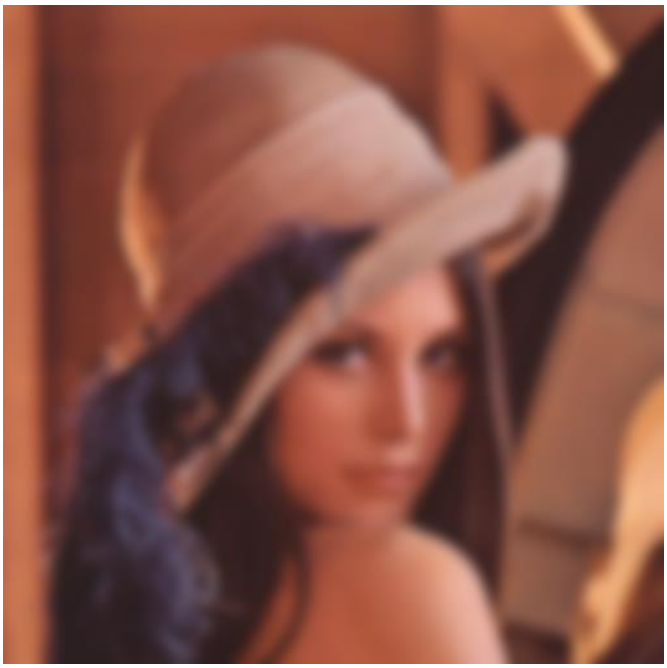

Figura 4.10 Lena, Gaussian blur cu rază de 5

4.4.4. Metode combinate

Pentru a ajunge la rezultate mai bune, metodele prezentate mai sus se pot combina între ele, dar trebuie ținut cont de ordinea în care acestea se execută.

De exemplu operațiunea de eroziune urmată de dilatare se numește deschidere, și se folosește pentru netezire de contururi, umplere goluri mici în obiecte și spargerea legăturilor slabe între obiecte, cum scrie și în [24].

Operațiunea de dilatare urmată de eroziune este numita închidere, și se folosește la fel ca deschiderea, cu diferența ultimului atribut, care acum este unirea legăturilor slabe între obiecte.

Pentru varietate și a ajunge la mai multe rezultate posibile, se pot folosi operațiuni de dilatare urmate de netezire.

4.5. Măsurarea calității

Pentru a putea compara obiectiv și analitic două imagini diferite, trebuie să folosim o metodă de măsurare a calității imaginii. Pentru a avea o imagine cu care să comparăm rezultatele fuziunii, vom folosi ca standard o "imagine perfectă". Cele două metode de măsurare implementate în această lucrare sunt:

- Eroarea medie pătratică (Mean squared error – MSE)
- Raportul între semnalul de vârf și zgomot (Peak signal to noise ratio – PSNR)

Presupunem următoarele notații:
P – imaginea perfectă
F – imaginea fuzionată
m – numărul de pixeli de pe rânduri
n – numărul de pixeli de pe coloane
Pentru a putea evalua, imaginile trebuie sa aibă aceleași dimensiuni.
Eroarea medie pătratica se poate calcula cu următoarea formula:

$$MSE = \frac{1}{m * n} \sum_{i=1}^{m} \sum_{j=1}^{n} (P_{ij} - F_{ij})^2$$

Raportul intre semnalul de vârf și zgomot:

$$PSNR = 10 * \log_{10}(\frac{v^2}{MSE})$$

Unde v este pixelul de valoare maximă.

O valoare cât mai mică reprezintă calitate mai bună în cazul primei metode de măsurare, iar în un raport semnal de vârf și zgomot cât mai mic semnifică la fel calitate mai bună.

Modul în care sunt implementate aceste măsurători constă în mai multe module decuplabile, în acest fel aspectele care se doresc a fi măsurate sunt personalizabile. Procedeul de măsurare a calității consta în următorii pași principali:

1. Se încarc grupuri de imagini de intrare.

Un grup este compus dintr-o combinație de trei imagini: o imagine perfectă, și două imagini care conțin informații parțiale, pe care se dorește fuziunea.

2. Se încarc metodele de fuziune pe care le dorim să analizăm

Aici se poate alege numărul de metode care se vor folosi pentru fuziune, și în acest fel și pentru compararea rezultatelor.

3. Se parcurge lista de imagini de intrare, pentru fiecare grup de imagini:
    a. Se parcurge lista de metode de fuziune, și se aplică algoritmul de fuziune asupra celor două imagini cu informații parțiale.
    b. Pentru fiecare metodă de fuziune:
        i. Se parcurg toate metodele de postprocesare, și se aplică pe imaginea rezultată din fuziune.
        ii. Pe această imagine se vor calcula metricile de calitate, comparând-o la imaginea perfectă de la intrare.
        iii. Rezultatele se păstrează ca grupuri de informație de ieșire în următorul format:
        ImagineRezultat, MSE, PSNR

Pentru afișarea rezultatelor am implementat un alt modul, care oferă utilizatorului posibilitatea să aleagă prin ce format vrea să se salveze rezultatele măsurătorilor de calitate. La momentul de față, se pot alege între trei feluri de scriere a rezultatelor: în format text (.txt) , în format Excel (.xls) sau în format de imagini. Toate se doresc să salveze rezultatele într-o formă cât mai lizibilă.

Avantajul metodei al doilea constă în faptul că rezultatele se pot ordona în funcție de coloană, iar pentru fiecare grup de imagini se creează o pagină noua (sheet). Dezavantajul este că nu toate sistemele au program de citire a fișierelor .xls.

Capitolul 5. Proiectare de Detaliu și Implementare

5.1. Prezentare generală

În proiectarea sistemului am urmărit despărțirea funcționalităților pe module, care la implementare vor fi reprezentate de pachete de cod sursă. Modulele au un rol bine definit în fluxul de date a aplicației, ele fiind independente una de alta, dar funcționarea corectă nu poate fi obținută fără acestea să coopereze și să se sincronizeze în procesele sistemului. Un controller principal este responsabil de coordonarea modulelor, și definirea ordinii pașilor care trebuie urmate pentru a obține rezultatele dorite și acestea sa ajungă la utilizator, fie prin interfață grafica sau prin salvarea lor pe hard disk.

Modulele principale și submodulele aferente sunt:

- Cititorul de imagini DICOM
- Modulul de fuziune
  - Convertirea din stivă de imagini în proiecție
  - Redimensionarea imaginilor
  - Fuziunea cu algoritmul și parametrii aleși
- Postprocesarea cu metoda aleasă
- Redenumirea imaginilor rezultate
- Afișarea rezultatului
- Salvarea rezultatului ca imagine JPEG

Pentru fluxul procesului de măsurarea calității, modulele sunt prezentate în capitolul anterior. Descrierea detaliată a modulelor se poate găsi în cele ce urmează.

5.2. Arhitectura generala a sistemului

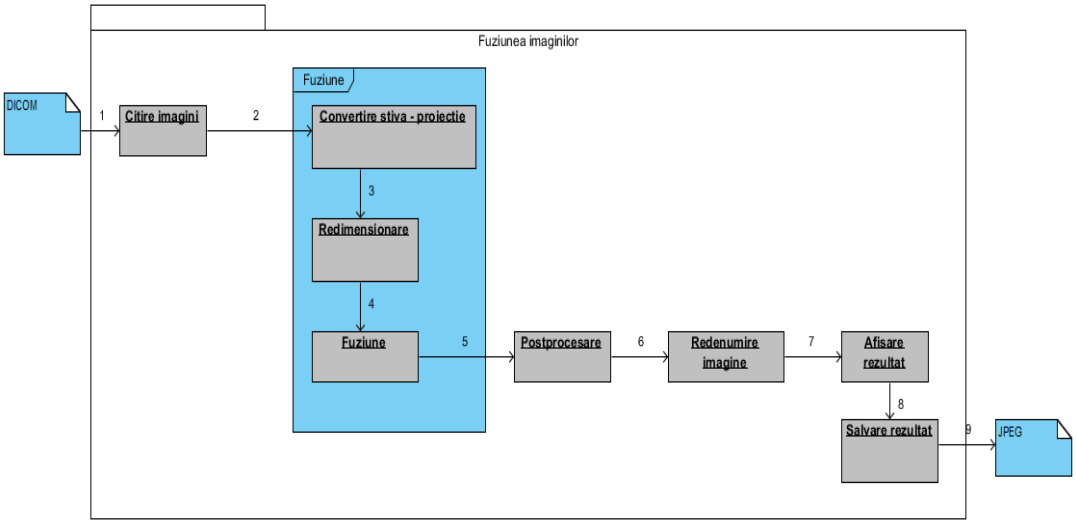

Figura 5.1 Fluxul detaliat al procesului de fuziune al imaginilor

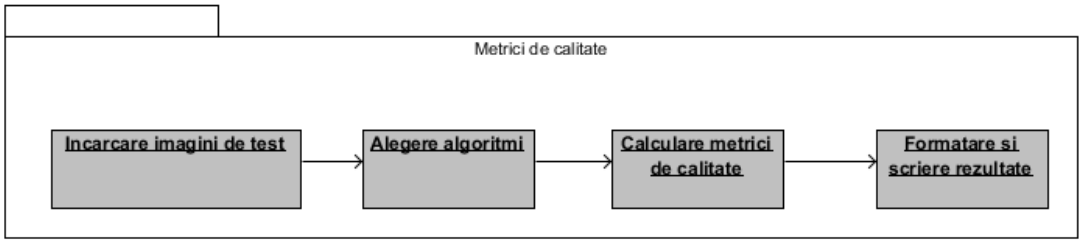

Figura 5.2 Fluxul detaliat al procesului de măsurare a calității

5.2.1. Cazuri de utilizare

Fluxurile generale ale informației sunt dictate de cerințele funcționale ale sistemului, prezentate anterior, și diagrama de use case.

Figura 5.3 Diagrama use-case a aplicației

Caz de utilizare: încărcarea imaginilor DICOM
Nivel: nivelul scopului utilizatorului
Actor principal: utilizator
Scenariul de succes principal: Utilizatorul încarcă fișierele care conțin imaginile și este notificat din interfața grafică în caz de succes.
Extensii: Un mesaj de eroare va apărea într-o fereastră nouă, în caz de eșec.

Caz de utilizare: vizualizarea imaginilor încărcate
Nivel: nivelul scopului utilizatorului
Actor principal: utilizator
Scenariul de succes principal: Utilizatorul poate să vizualizeze imaginile încărcate în ferestre noi, în cazul stivelor de imagini având posibilitatea să facă „scroll” peste ele.
Extensii: Un mesaj de eroare va apărea într-o fereastră nouă, în caz de eșec.

Caz de utilizare: rularea procesului de fuziune
Nivel: nivelul scopului utilizatorului
Actor principal: utilizator
Scenariul de succes principal: Utilizatorul, după ce alege algoritmul, parametri acestuia și metoda de postprocesare, poate să ruleze procesul de fuziune, a cărui rezultat este afișat în caz de succes.
Extensii: Un mesaj de eroare va apărea într-o fereastră nouă, în caz de parametri greșiți sau eroare la fuziune.

Caz de utilizare: vizualizarea rezultatelor
Nivel: nivelul scopului utilizatorului
Actor principal: utilizator
Scenariul de succes principal: Utilizatorul, la fel ca în cazul punctului doi, poate să vizualizeze rezultatele procesului de fuziune în ferestre noi.
Extensii: Imaginea nu va apărea în caz de eșec.

Caz de utilizare: salvarea rezultatelor
Nivel: nivelul scopului utilizatorului
Actor principal: utilizator
Scenariul de succes principal: Utilizatorul are posibilitatea să salveze rezultatele în fișiere externe, ca și imagini JPEG.
Extensii: Un mesaj de eroare va apărea în cazul în care nu s-a putut salva fișierul.

Caz de utilizare: calcularea metricilor de calitate
Nivel: nivelul scopului utilizatorului
Actor principal: utilizator
Scenariul de succes principal: Utilizatorul, după ce alege modul în care vrea să fie salvate rezultatele, poate să ruleze procesul de calcularea metricilor de calitate între algoritmi.
Extensii: Un mesaj de eroare va apărea într-o fereastră nouă, în caz de eșec.

5.2.2. Structura pachetelor codului sursă

Despărțirea modulelor în pachete de cod sursă a fost efectuat conform bunelor practici ale programării în limbajul Java. Structura arată în felul următor:

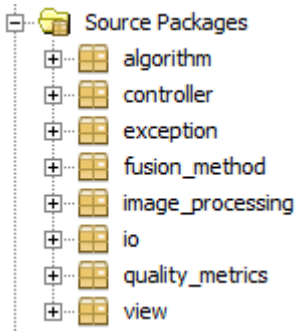

Figura 5.4 Structura de pachete a aplicației

Proiectarea aplicației a fost făcută folosind design pattern-ul MVC (Model View Controller), prin acest model separând clar nivelele de prezentare, model și control.

5.3. Descrierea modulelor

5.3.1. Modulul de algoritmi

Modulul poate cel mai important este modulul în care se află algoritmii care stau la baza metodelor de fuziune. În acesta sunt definiți cei doi algoritmi, Transformata Discretă Wavelet Haar și Piramida Laplaciană. Cei doi fac categorie din clase diferite de algoritmi de fuziune, primul fiind de tip Wavelet, iar cel de-al doilea fiind de tip piramidal. Implementarea lor este în pachetul „algorithm”, în clasele „HaarDWT” respectiv „LaplacianPyramid”. Se pot observa din numele metodelor, că acești algoritmi presupun și o modalitate de a inversa procesul de transformare, altfel zis, reproducerea semnalului original de intrare după transformarea acestuia.

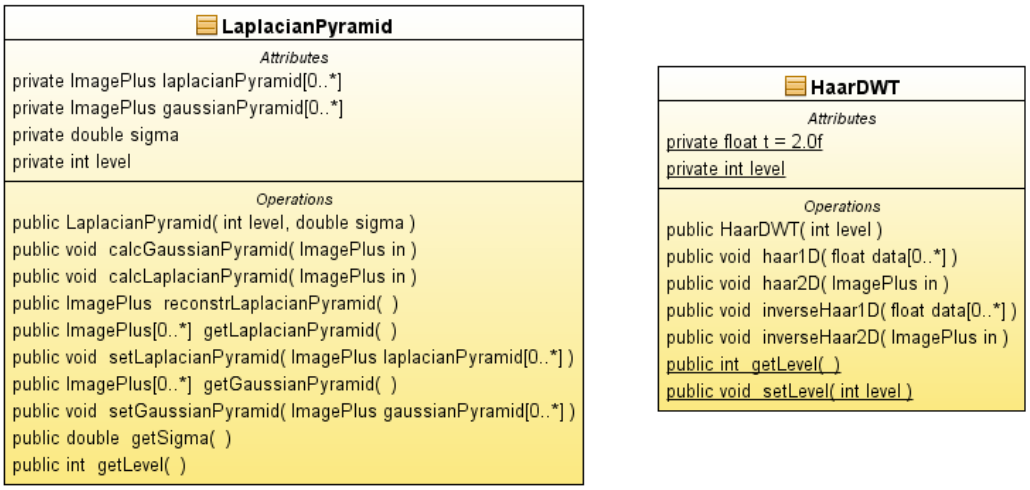

Figura 5.5 Diagrama de clasă a pachetului „algorithm”

5.3.1.1. Transformata Wavelet Discreta Haar

Transformata Wavelet Discreta Haar directa 1D:

```
int h = data.length / 2;
for (int i = 0; i < h; i++) {
        int k = i * 2;
        temp[i] = (data[k] + data[k + 1]) / t;
        temp[i + h] = (data[k] - data[k + 1]) / t;
}
```

- „data” este semnalul de intrare (imaginea reprezentata ca un vector), de exemplu un rând din imagine sau o coloană.
- „h” reprezintă mijlocul vectorului de intrare.
- „i” iterează de la începutul vectorului până la „h”
- „k” este folosit sa luăm elementele din două în două poziții
- „temp” este vectorul care va conține transformata wavelet la ieșire
- „t” este factorul de despărțire, care e de obicei 2

Transformata 1D inversă funcionează similar, cu diferența construirii vectorului de ieșire:

```
for (int i = 0; i < h; i++) {
        int k = i * 2;
        temp[k] = data[i] + data[i + h];
        temp[k + 1] = data[i] - data[i + h]
}
```

Transformatele 2D folosesc aceste metode, aplicându-le pe coloane și rânduri. Transformata Wavelet Haar este implementată ca fiind multirezoluțională, se poate preciza nivelul până la care se vor aplica transformatele 1D:

```
int levCols = cols / lev;
int levRows = rows / lev;
```

- „levCols” este nivelul coloanelor
- „cols” este numărul total al coloanelor
- „lev” este nivelul dorit de rezoluție
- Analog pentru rânduri, „levRows” și „rows”

Folosind aceste valori, se iterează prima dată peste rânduri, calculând transformatele Haar 1D:

```
row = new float[levRows];
for (int i = 0; i < levRows; i++) {
        for (int j = 0; j < row.length; j++) {
                row[j] = data[j][i];
        }
        haar1D(row);
        for (int j = 0; j < row.length; j++) {
                data[j][i] = row[j];
        }
}
```

- Se creează un nou rând „row” de dimensiunea rândurilor pentru nivelul de rezoluție aferent, „levRows”
- Se iterează în pătratul levCols * levRows din datele de intrare „data”
- Se extrag pixelii de pe pozițiile aferente din „data”, se pun în „row”
- Se aplică transformata Haar Wavelet directă pe „row”
- Se scrie rezultatul obținut în rândul din care s-a citit, înapoi în datele de intrare

Similar se aplica procedeul și pentru coloane. La sfârșitul procedeului vom avea în „data” transformata Haar Wavelet Discretă.

Analog se face și transformata inversă. Codul de mai jos exemplifică diferențele:

```
for (int i = 0; i < levRows; i++) {
    for (int j = 0; j < row.length; j++) {
            row[j] = data[j][i];
    }
    inverseHaar1D(row);
    for (int j = 0; j < row.length; j++) {
            data[j][i] = row[j];
    }
}
```

5.3.1.2. Piramida Laplaciană

Calcularea piramidei Laplaciane constă în doi pași importanți:

1. Calcularea Piramidei Gaussiene
2. Calcularea Piramidei Laplaciene

Cum sugerează și numele, semnalul de intrare este transformat într-o reprezentare de forma unei piramide, adică o să avem o listă de imagini dintr-una de intrare de lungimea nivelului dorit de transformare. Această listă o sa fie reprezentată ca un ArrayList<ImagePlus>.

Procesul de calculare a piramidei Gaussiene est următorul:

1. Se citește o imagine de intrare
2. Se adaugă în lista de imagini ArrayList<ImagePlus>
3. Pentru fiecare nivel de rezoluție
    a. Se aplica un filtru trece jos, sau filtru Gaussian pe imaginea precedentă, asigurând un efect de blur
    b. Se redimensionează imaginea precedentă divizând dimensiunile cu 2
    c. Se adaugă în lista de imagini pe poziția nivelului curent

```
iP.blurGaussian(sigma);
ImageProcessor iPResized = iP.resize(width / 2);
```

Procesul de calculare a piramidei Laplaciene:

1. Se ia piramida Gaussiană calculată anterior
2. Se începe de la nivelul cel mai inferior a piramidei
3. Se extrage imaginea de pe nivelul curent
4. Se măresc dimensiunile cu un factor de 2
5. Se aplică un filtru trece-jos, sau blur Gaussian
6. Se calculează diferențele de pixeli între imaginea pe care s-au efectuat aceste operații și imaginea precedentă din piramida Gaussiană
7. Se adaugă imaginea obținută în piramidă Laplaciana, pe nivelul aferent

```
for (int i = level - 1; i > 0; i--) {
        ImagePlus gaussian = gaussianPyramid.get(i).duplicate();
        ImageProcessor processor = gaussian.getProcessor();

        // size *= 2
        processor.setInterpolationMethod(ImageProcessor.BICUBIC);
        ImageProcessor processorResized = processor.resize(gaussian.getWidth() * 2);
        gaussian = new ImagePlus("GaussianResized " + i, processorResized);

        // low pass filter
        processorResized.blurGaussian(sigma);

        //difference between current image and previous
        ImagePlus current = gaussianPyramid.get(i - 1);
        ImageCalculator calculator = new ImageCalculator();
        ImagePlus difference = calculator.run("Subtract create", current, gaussian);
        difference.setTitle("Laplacian " + (i - 1));

        // store laplacian image
        laplacianPyramid.add(difference);
}
```

Reconstrucția imaginii originale din piramida Laplaciana este similară cu algoritmul prezentat mai înainte, cu diferența ca se adaugă valorile pixelilor la fiecare pas la o imagine ImagePlus. Suma totală va reprezenta imaginea retransformată.

5.3.2. Modulul de metode de fuziune

Acest modul conține metodele propriu-zise de fuziune. Câteva dintre clase folosesc ca algoritm de baza algoritmii prezentați mai sus în modulul precedent. Metodele de fuziune trebuie să implementeze interfață „FusionMethod”, și metoda acestuia „public ImagePlus fuse(ImagePlus image1, ImagePlus image2)”. Această metodă va conține toată logica din spate a fuziunii.

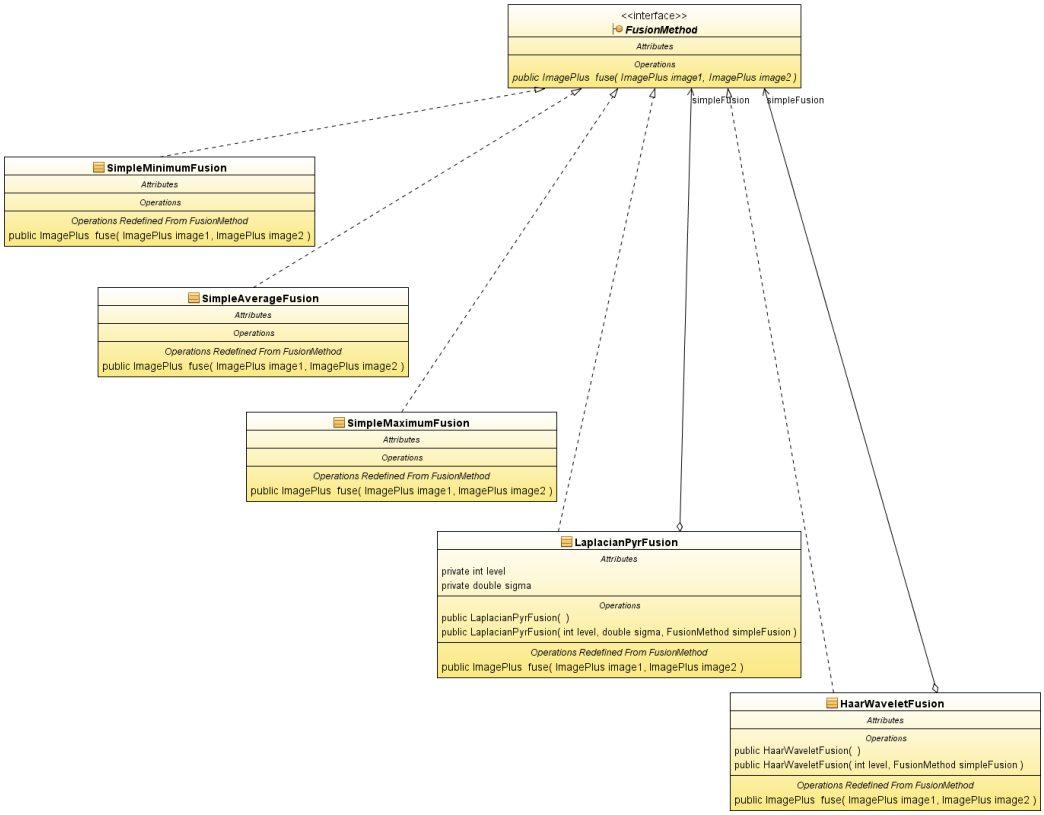

Figura 5.6 Diagrama de clasă a pachetului „fusion_method”

Clasa „ImagePlus” din pachetul „ij” (ImageJ), reprezintă o imagine cu metode de procesare și filtrare incluse [30]: conține un procesor de imagine de tip ImageProcessor, sau o stivă de imagini 3D, 4D sau 5D, de tip ImageStack. In afară de aceste obiecte, ImagePlus poate conține și metadate, cum ar fi calibrarea spațială, numele fișierului sau a directorului din care a fost citită.

Metodele de fuziune mai complexe, cum sunt clasele HaarDWT și LaplacianPyramid, conțin o referința la un obiect FusionMethod. Acesta este folosit la fuziunea propriu-zisă a rezultatelor obținute din aplicarea algoritmilor pe imaginile de intrare. Cum acestea transformă semnalul de intrare, procedeul general de fuziune în cazul lor este:

1. Transformarea imaginilor folosind algoritmul ales
2. Fuziunea rezultatelor cu o metoda de fuziune aritmetică
3. Transformarea rezultatului fuziunii înapoi în domeniul original al imaginii

Exemplu de implementare a metodei „fuse” în clasa „HaarWaveletFusion”:

```
public ImagePlus fuse(ImagePlus image1, ImagePlus image2) {
    ImagePlus haarImage1 = image1.duplicate();
    haarDwt.haar2D(haarImage1);
    ImagePlus haarImage2 = image2.duplicate();
    haarDwt.haar2D(haarImage2);
    // Fusion method used to fuse the two haar images
    ImagePlus result = simpleFusion.fuse(haarImage1, haarImage2);
    haarDwt.inverseHaar2D(result);
    result.setTitle("Haar_" + HaarDWT.getLevel() + " " + image1.getShortTitle() +
    " + " + image2.getShortTitle());
    return result;
```

}

- Se duplică imaginile de intrare, ca să nu se facă modificări direct pe ele
- „haarDwt” este algoritmul Haar de tipul „HaarDWT”
- Se calculează transformatele Haar pentru fiecare imagine
- Se face fuziunea între transformatele Haar cu o metodă simplă (aritmetică)
- Se aplică transformata inversă pe rezultatul fuziunii
- Se redenumește imaginea rezultat și se returnează

Analog funcționează și metoda piramidei Laplaciene, însă acolo se face fuziunea pe toate elementele piramidelor separat, rezultatul fiind o nouă piramidă. Pe aceasta se va aplica transformata inversă:

```
for (int i = 0; i < pyramid1.size(); i++) {
    result.add(simpleFusion.fuse(pyramid1.get(i), pyramid2.get(i)));
}
```

Iar imaginea rezultat se extrage cu:

```
ImagePlus resultImage = pyramidCalculator.reconstrLaplacianPyramid();
```

5.3.3. Modulul de procesare a imaginilor

În acest modul sunt definite operații generale de procesarea imaginilor, care sunt importante pentru aplicație de față, însă nu esențiale în funcționarea teoretică. Practic, ele se folosesc pentru a ajunge la un rezultat cât mai bun după aplicarea fuziunii.

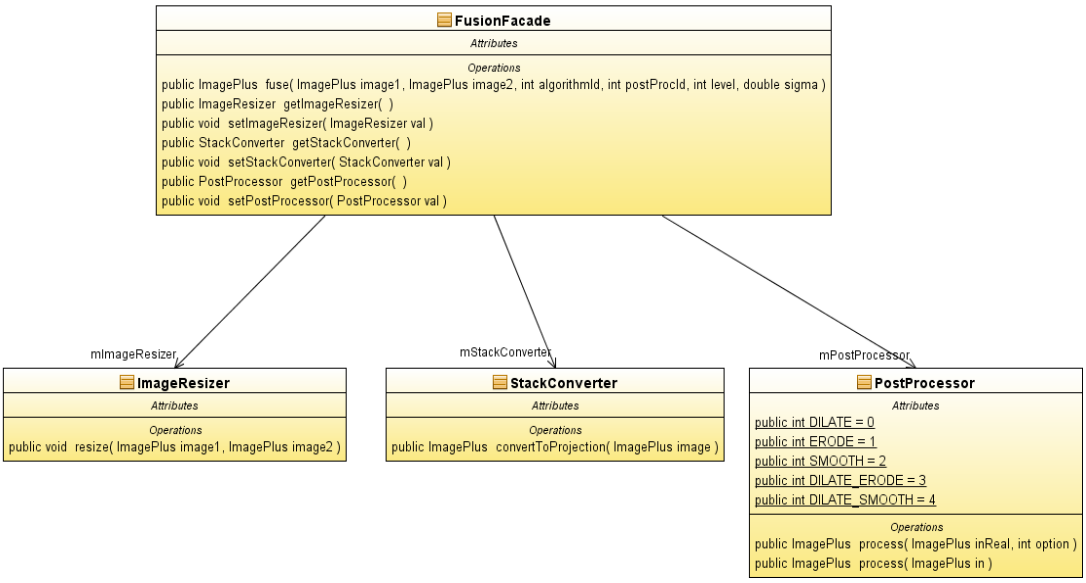

Figura 5.7 Diagrama de clase a pachetului „image_processing”

ImageResizer este folosit doar în cazul în care dimensiunile imaginilor de intrare diferă. În cazul acesta, se vor redimensiona la media dimensiunilor lor originale, în acest mod se minimizează cantitatea informației pierdute și cantitatea informației adăugate prin interpolarea pixelilor (în cazul în care se mărește dimensiunea unei imagini).

$$W_R = \frac{W_A + W_B}{2}$$

$$H_R = \frac{H_A + H_B}{2}$$

Unde:

- R este imaginea rezultat
- A și B sunt imaginile de intrare
- W este lățimea imaginilor
- H este înălțimea imaginilor

Clasa StackConverter converteste stivele de imagini în proiecția lor pe axa verticala imaginară Z. Pentru acest procedeu ne vom folosi de clasa deja implementată ca un plugin ImageJ, ZProjector, din pachetul „ij.plugin”.
Un exemplu de cod sursă sugerează funcționarea acestei clase:

```
ZProjector projector = new ZProjector(image);
projector.setMethod(ZProjector.MAX_METHOD);
projector.doProjection();
```

Clasa PostProcessor încapsulează logica de postprocesare a imaginilor. Pentru această operațiune vom folosi PostProcessor-ul din imaginea de intrare de tip ImagePlus. Exemplu de postprocesarea aplicând dilatarea:

```
ImageProcessor processor = in.getProcessor();
processor.dilate();
```

Aici „in” este imaginea de intrare de tip ImagePlus.

Cu un switch() se verifică opțiunea aleasă de utilizator și se alege metoda de postprocesare aferentă pentru aceasta:

```
in.setTitle(inReal.getTitle() + ">dilated");
```

Toată logica de procesarea a imaginilor este grupată într-o fațadă, FusionFacade, prin care se ascunde logica de implementare, și se oferă acces ușor la operațiile importante. Aceasta fațadă delegă responsabilitățile la obiectele enumerate mai sus.

5.3.4. Modulul de citire și scriere a datelor

Acest modul este responsabil de citirea datelor din fișiere și scrierea rezultatelor în forma de fișiere noi.

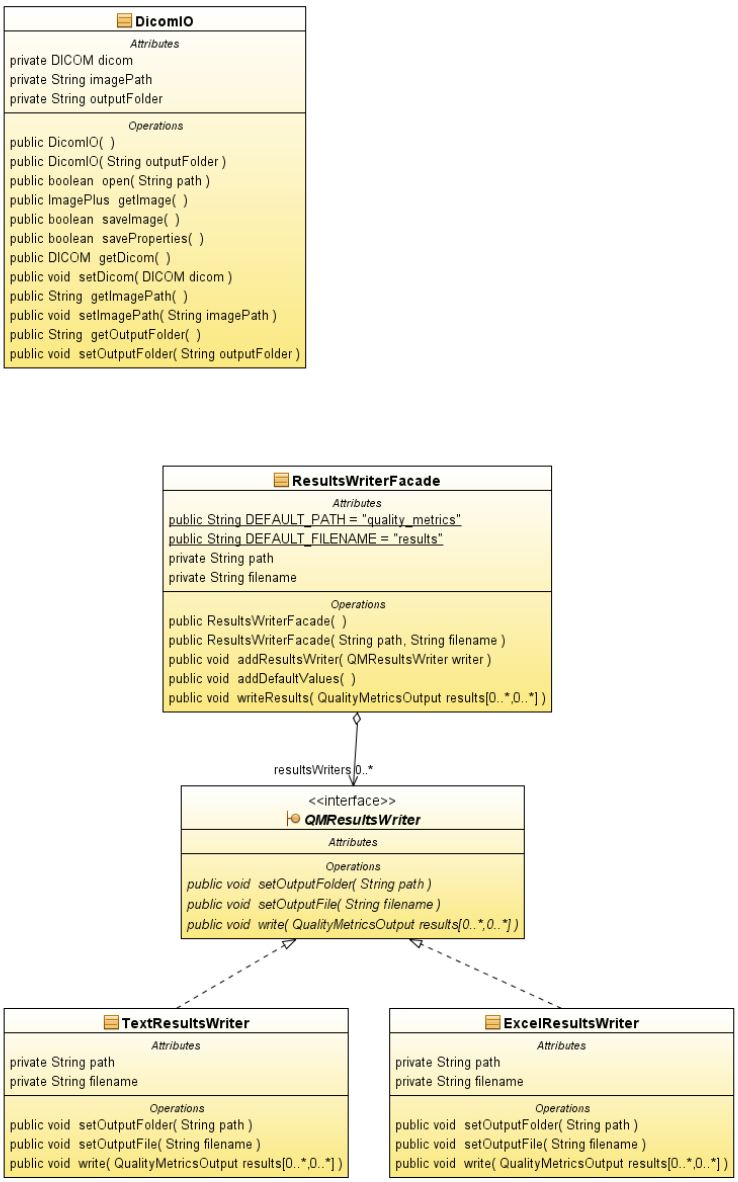

Figura 5.8 Diagrama de clase a pachetului „io”

Prin intermediul acestui modul putem deschide fișierele DICOM, extrăgând informațiile din ele în format ImagePlus. Acesta folosește clasa „DICOM” din ImageJ, care ne oferă o serie de metode deja implementate, cum ar fi deschiderea unui fișier DICOM, citirea dimensiunii stivei de imagini conținute în fișier, extragerea informațiilor din set-ul de date și citirea imaginii propriu-zise din fișier.

Exemple de operații:

- Inițializare obiect DICOM
  private DICOM dicom = new DICOM();
- Deschidere fișier DICOM cu calea „path”
  dicom.open(path);
- Citirea lățimii imaginii conținute în fișierul DICOM
  dicom.getWidth();
- Citirea dimensiunii stivei de imagini
  dicom.getStackSize();
- Citirea titlului scurt al fișierului deschis (fără calea absolută)

dicom.getShortTitle();
- Extragerea imaginii din DICOM
  dicom.getImage();
- Extragerea stivei de imagini
  dicom.getImageStack();

Al doilea format important de ieșire a sistemului sunt rezultatele măsurării calității. Acestea se pot scrie fie în fișier text, fie în fișier Excel sau ca imagini. Pentru a formata și persista fișierele Excel, am folosit framework-ul JExcel API, care este ușor de integrat și oferă metode folositoare pentru salvarea informațiilor în format lizibil și structurat logic. Exemplu de folosire JExcel:

- Importarea clasei „jxl.Workbook”
- Creare „caiet” excel
  WritableWorkbook workbook = Workbook.createWorkbook(new File(this.path + "/" + this.filename + ".xls"));
- Creare paginilor în interiorul cârtii excel
  WritableSheet sheet = workbook.createSheet("Results " + sheetIndex, sheetIndex);
  Aici sheetIndex este un număr întreg, care reprezintă numărul foii din carte.
- Crearea unui câmp nou care conține String
  Label label = new Label(0, rowIndex, output.getResultImage().getTitle());
- Adăugarea câmpului la foaia creata mai înainte
  sheet.addCell(label);
- Scrierea informațiilor adăugate în caiet
  workbook.write();
- Închiderea caietului de lucru
  workbook.close();

5.3.5. Modulul de metrici de calitate

Pentru calcularea metricilor de calitate am folosit două măsurători:

- Eroarea medie pătratica (Mean squared error – MSE)
- Raportul între semnalul de vârf și zgomot (Peak signal to noise ratio – PSNR)

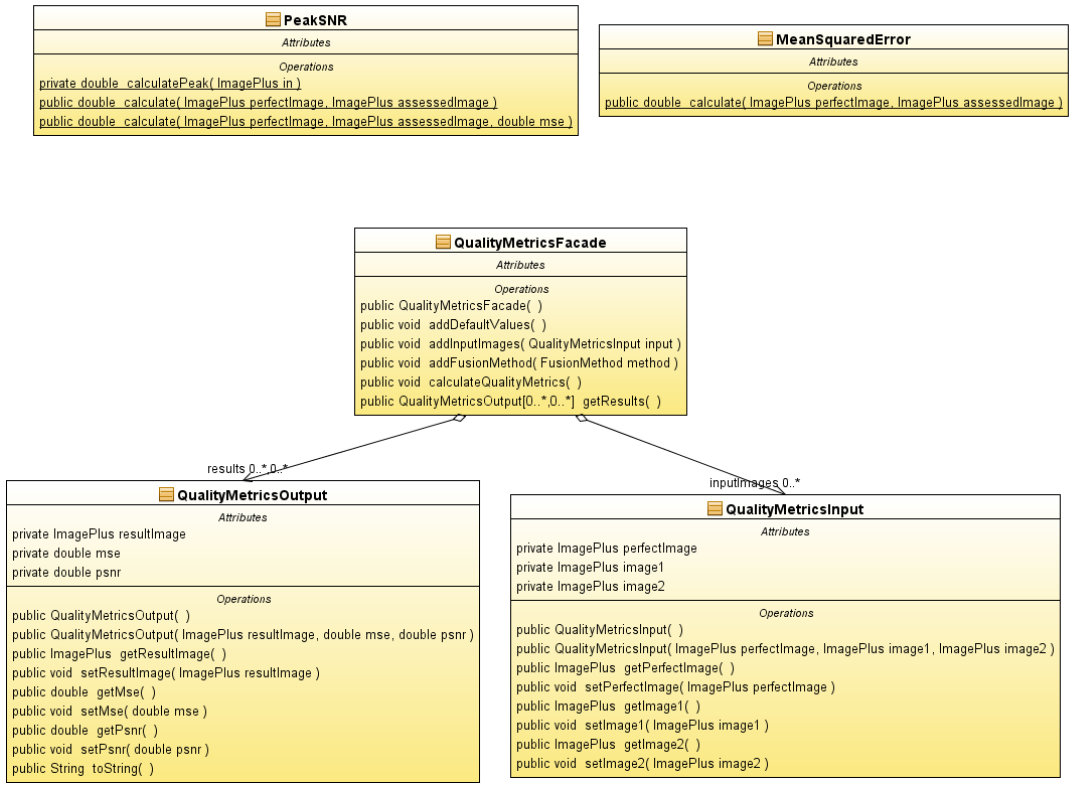

Figura 5.9 Diagrama de clase pentru pachetul „quality_metrics”

Funcționarea lor teoretică și formula sunt prezentate în Capitolul 3. Clasa MeanSquaredError conține metoda de calculare a erorii medii pătratice și este implementată în felul următor: pentru fiecare pixel al imaginii care se testează, se calculează diferența de intensitate între pixelul imaginii de intrare cu pixelul unei imagini „perfecte”, și numărul acesta se ridică la pătrat. Se adun toate valorile calculate și se divid la sfârșit cu numărul total al pixelilor din imagini.

```
for (int i = 0; i < perfectImage.getHeight(); i++) {
        for (int j = 0; j < perfectImage.getWidth(); j++) {
                error += Math.pow(perfectProc.get(i, j) - assessedProc.get(i, j), 2);
        }
}
return error / (perfectImage.getWidth() * perfectImage.getHeight());
```

Unde perfectProc este ImageProcessor-ul imaginii perfecte, iar assessedProc este ImageProcessor-ul imaginii supuse testului.

Clasa PeakSNR conține metodele pentru calcularea raportului între semnalul de vârf și zgomot. În prima fază se calculează intensitatea maximă a pixelilor din imaginea măsurata, după care se calculează MSE pentru imagine și se calculează PSNR cu următoarea formula:

$$PSNR = 10 * \log_{10} \frac{ValoreVarf^2}{MSE}$$

Codul care calculează acest număr:

```
double mse = MeanSquaredError.calculate(perfectImage, assessedImage);
double peak = calculatePeak(assessedImage);
double psnr = 10 * Math.log10(Math.pow(peak, 2) / mse);
```

Aceste două modele reprezintă precizia, sau mai bine zis similaritatea imaginii supuse testului cu o imagine „perfecta”. Eroarea medie pătratica mai mică reprezintă

similaritate mai mare, iar raportul intre semnalul de vârf și zgomot mai mare înseamnă la fel, calitate mai bună.

Pentru rularea procesului de măsurarea calității, modulul primește la intrare obiecte multiple de tip QualityMetricsInput, care încapsulează trei imagini ImagePlus: imaginea perfectă, și cele două imagini care urmează să fie fuzionate și rezultatul lor comparat cu prima imagine.

La ieșire, modulul produce un obiect de tipul QualityMetricsOutput, care va conține rezultatele procesului de măsurarea calității: imaginea rezultat a procesului de fuziune, și cele două valori de calitate, MSE și PSNR.

Toată logica de calcularea calității este încapsulată într-o fațadă, QualityMetricsFacade, care oferă acces ușor la funcționalități, și în același timp ascunde implementarea metodelor.

5.3.6. Modulul de vizualizare și interfață grafică

Acest modul conține interfață grafica a aplicației. Se pot găsi două clase în pachetul acesta: MainForm și MessageDialog.

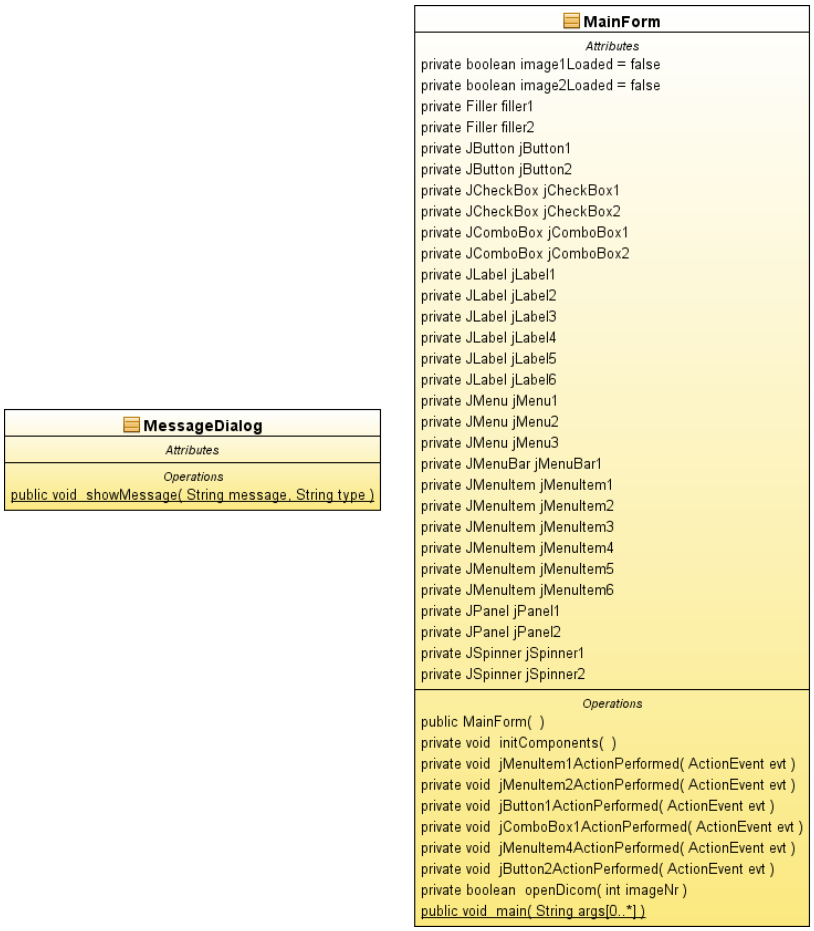

Figura 5.10 Diagrama de clase pentru pachetul „view”

MainForm conține elementele vizuale din Java Swing prin intermediul cărora utilizatorul poate interacționa cu aplicație. Interfața e proiectată în așa fel încât să fie intuibilă și ușor de folosit. Mai multe detalii despre interfață grafică și cum se poate interacționa cu aplicația se pot găsi în Capitolul 7.

Clasa MessageDialog este folosita la afișarea eventualelor mesaje, fie de eroare sau avertizare. Este proiectată în așa fel încât tipul și titlul mesajului poate fi precizat de către un parametru String:

```
if (type.toLowerCase().equals("warning")) {
        messageType = JOptionPane.WARNING_MESSAGE;
} else if (type.toLowerCase().equals("error")) {
        messageType = JOptionPane.ERROR_MESSAGE;
}
```

5.3.7. Modulul excepție

Aplicația are definit și un modul propriu de excepție. Această modalitate este preluată din bunele practici ale limbajului Java. Excepția este aruncată la nivel de procesare și este prinsă în controller, care ii afișează mesajul în interfața grafică.

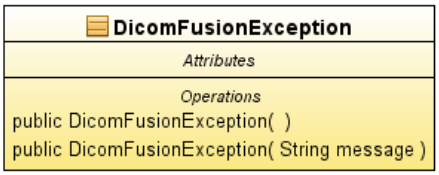

Figura 5.11 Diagrama clasei „Exception”

5.3.8. Modulul controller

Modulul de controller este responsabil de fluxul corect al datelor și operaților. El comunică cu stratul de vizualizare și modelele care conțin logica de procesare și algoritmii.

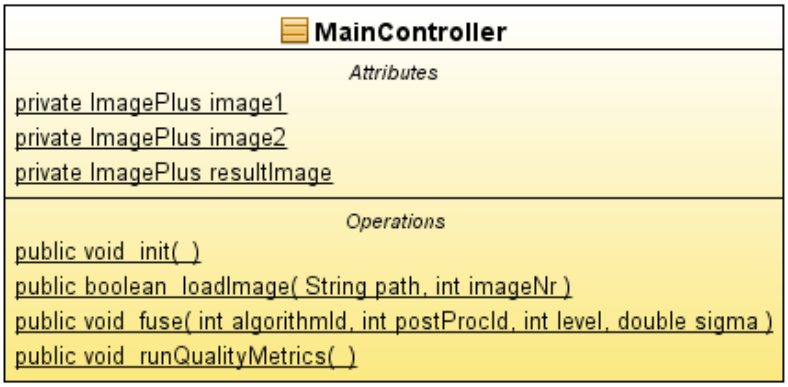

Figura 5.12 Diagrama clasei „MainController”

MainController-ul folosește printre altele fațadele definite în celelalte module. MainController este la fel responsabil să proceseze datele și stimulii primiți de la utilizator, să le „înțeleagă” și să transporte cerințele mai departe la modulele de algoritmi. Pe lângă asta, controller-ul interceptează mesajele de excepție și le afișează utilizatorului prin interfața grafică.

5.4. Vedere în ansamblu al sistemului și operațiile importante

Mai jos este prezentată diagrama de clase generală pentru întreaga aplicație. Se pot observa legăturile intre clase și interacțiunea părții de interfața grafică cu logica aplicației.

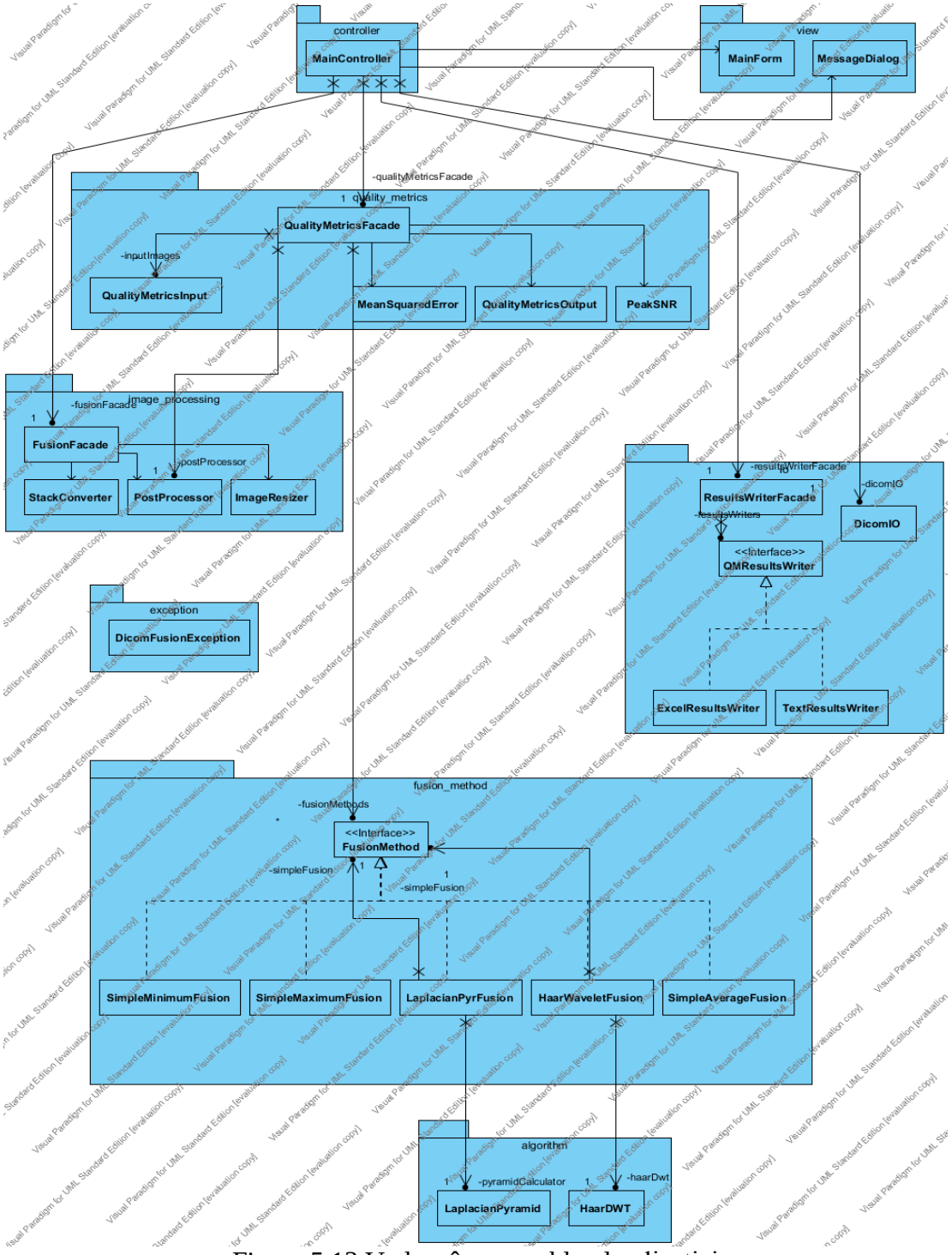

Figura 5.13 Vedere în ansamblu al aplicației

Mai jos este prezentata diagrama de secvență pentru deschiderea fișierelor DICOM. Pașii principali sunt:

- MainController primește comanda de la interfața grafică să citească un fișier

- Apelează metoda din dicomIO, cu care să deschidă fișierul
- După numărul transmis ca parametru, va ști care imagine a fost citită

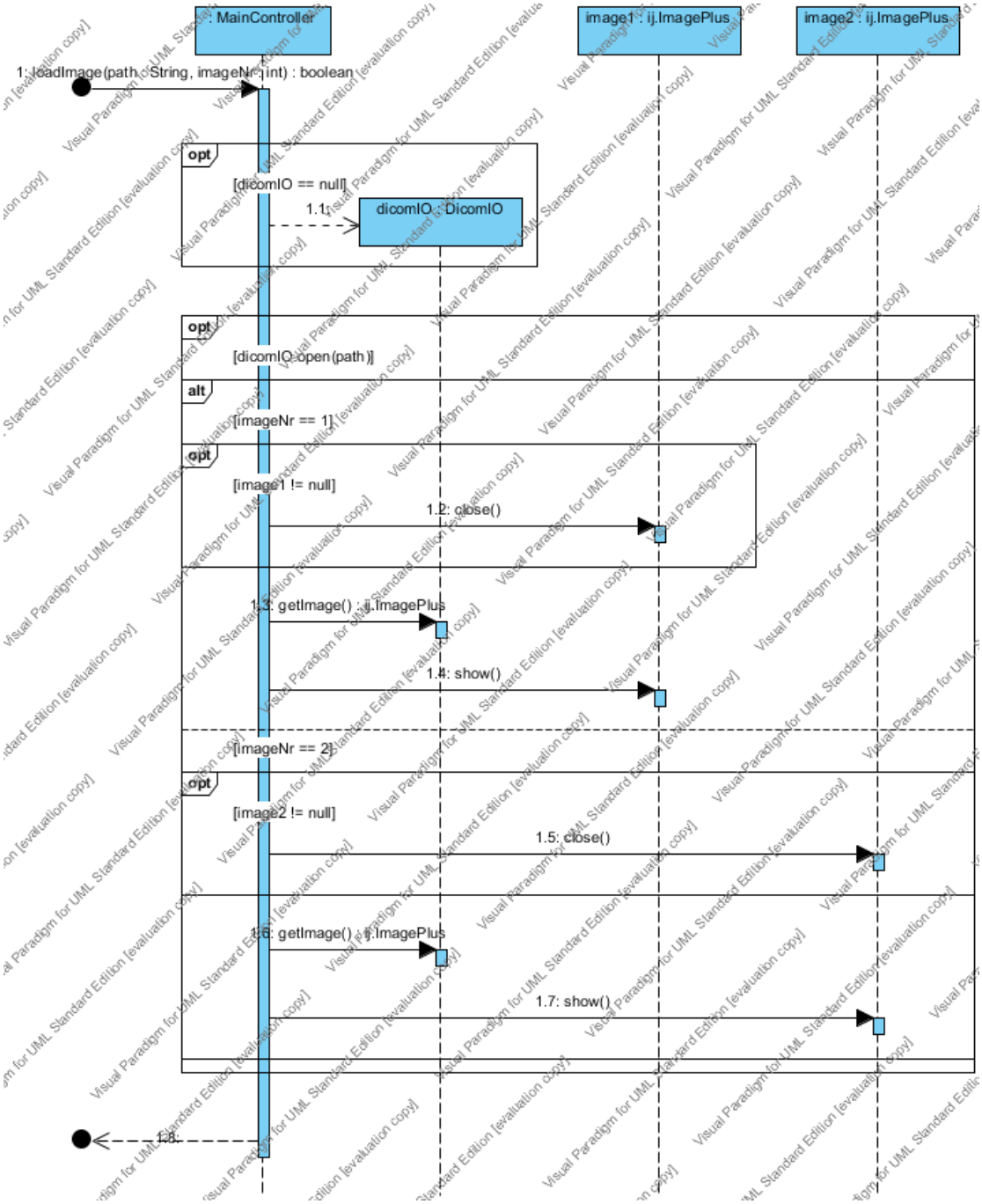

Figura 5.14 Diagrama de secvență a operației de deschidere fișierelor DICOM

Pașii principali a procesului de fuziune:

- MainController primește comanda de la interfața grafică
- Apelează metoda de fuziune din fusionFacade
- Acesta încapsulează logica de fuziune
- Va returna o imagine ImagePlus ca rezultat

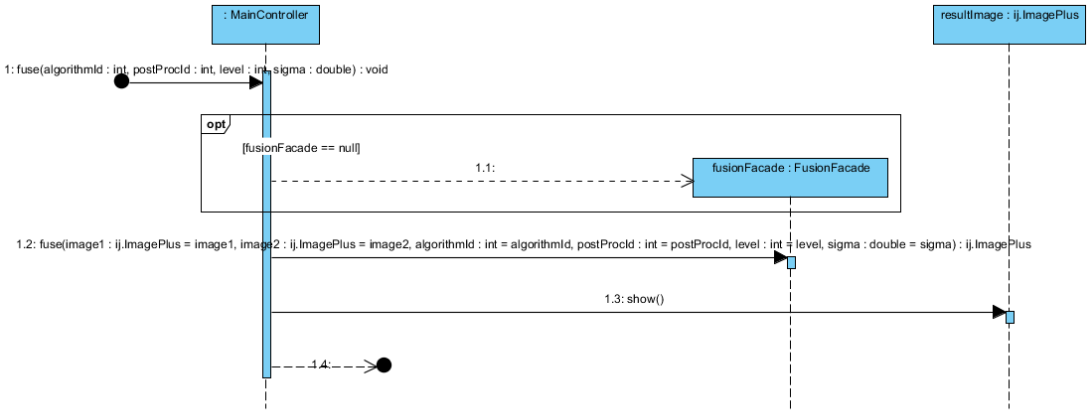

Figura 5.15 Diagrama de secvență a operației de fuziune, adâncime de 3 nivele

Pașii principali a procesului de măsurarea calității:

- MainController primește comanda de la interfața grafică
- Apelează metoda de calculare din qualityMetricsFacade
- Cu rezultatul obținut de la acesta, apelează resultsWriterFacade
- Se salvează rezultatele în funcție de formatul ales

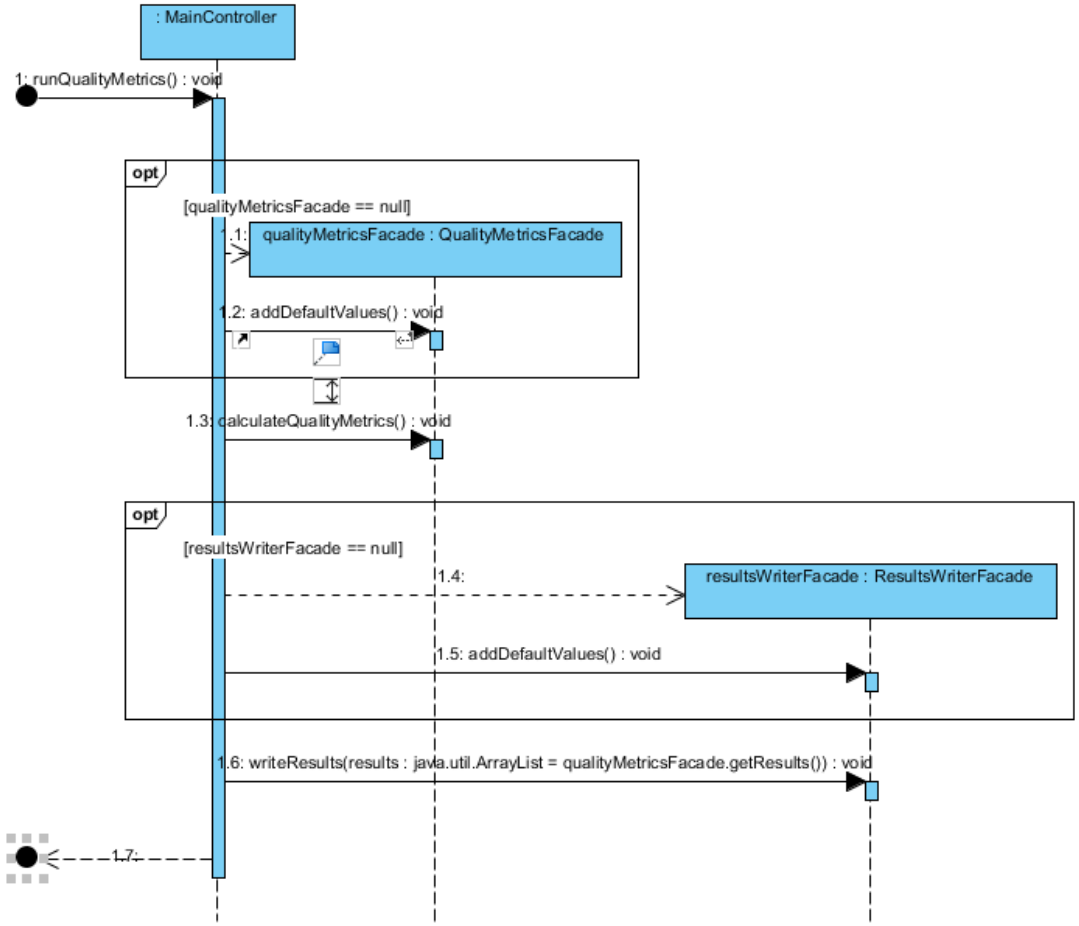

Figura 5.16 Diagrama de secvență a operației de rularea metricilor de calitate

Capitolul 6. Testare și Validare

6.1. Teste de performanță

Sistemul pe care s-au rulat testele are următoarele caracteristici:

- Procesor: Intel® Core™ i5 CPU M460 @ 2.53 GHz, 2 Cores, 4 Logical Processors
- Memorie RAM instalată: 8GB, 1333 MHz
- Sistem de operare: Microsoft Windows 7 Ultimate
- Arhitectură sistem: x64

6.1.1. Procesul de fuziune

Procesul de fuziune se măsoară în milisecunde pentru fiecare combinație de algoritm și metodă de postprocesare. Mai jos aveți un tabel cu valorile obținute.

Tabel 6.1 Viteza algoritmilor de fuziune și postprocesare

Algoritm fuziune	**Postprocesare**	**Parametri**	**Rezultat [ms]**
Minim	-	-	3
Medie	-	-	5
Medie	Dilatare	-	9
Medie	Eroziune	-	7
Medie	Netezire	-	7
Medie	Dilatare + Eroziune	-	11
Medie	Dilatare + Netezire		10
Maxim	-	-	4
Laplacian	-	Nivel=1, sigma=3	5
Laplacian	-	Nivel=2, sigma=3	56
Laplacian	-	Nivel=3, sigma=1	69
Laplacian	-	Nivel=3, sigma=2	66
Laplacian	-	Nivel=3, sigma=3	67
Laplacian	Dilatare	Nivel=3, sigma=3	74
Laplacian	Eroziune	Nivel=3, sigma=3	71
Laplacian	Netezire	Nivel=3, sigma=3	71
Laplacian	Dilatare + Eroziune	Nivel=3, sigma=3	74
Laplacian	Dilatare + Netezire	Nivel=3, sigma=3	74
Laplacian	-	Nivel=3, sigma=4	85
Laplacian	-	Nivel=4, sigma=3	76
Laplacian	-	Nivel=5, sigma=3	81
Haar	-	Nivel=1	12
Haar	-	Nivel=2	12
Haar	-	Nivel=3	13
Haar	Dilatare	Nivel=3	16
Haar	Eroziune	Nivel=3	16
Haar	Netezire	Nivel=3	15
Haar	Dilatare + Eroziune	Nivel=3	19
Haar	Dilatare + Netezire	Nivel=3	19
Haar	-	Nivel=4	13
Haar	-	Nivel=5	13

Haar	-	Nivel=6	13
Haar	-	Nivel=7	13

Din aceste valori, putem observa că metodele aritmetice sunt cele mai rapide, lucru care este ușor de înțeles, pentru că ele nu necesită operații complexe la fuziune. Referitor la metodele de postprocesare, am aflat că în general netezirea și eroziunea sunt cele mai rapide, urmate de dilatare, apoi de operații combinate, dilatare cu eroziune (închidere) și dilatare cu netezire.

La algoritmii multirezoluționali, timpul de procesare crește în general cu creșterea nivelului de rezoluție, mai semnificativ în cazul piramidei Laplaciene, mai puțin observabil în cazul transformatei Haar.

6.1.2. Procesul de măsurare a calității

Procesul de măsurare a calității nu poate fi parametrizat, deci cu multiple rulări încercăm să ajungem la o medie de milisecunde de procesare.

Tabel 6.2 Durata procesului de măsurarea calității

Numărul rulării	**Durata procesului [ms]**
1	9246
2	8507
3	8422
4	8448
5	8481

Deci în medie, procesul durează 8620 de milisecunde, sau aproximativ 8.5 de secunde. De menținut că procesul nu include și salvarea rezultatelor în fișiere. El rulează cele 5 metode, fiecare cu cele 5 moduri de postprocesare, deci în total 5 * 5 = 25 de rulări pentru fiecare set de imagini de intrare. Ca și parametru predefinit, avem 3 seturi de imagini de intrare, deci ne rezultă 3 * 25 = 75 de rulări de fuziune în total.

6.2. Metrici de calitate și comparația algoritmilor

După mai multe experimente, s-a observat că transformata Haar Wavelet de nivelul 2 și 3 produce rezultatele cele mai bune, iar piramida Laplaciană de nivelul 3 cu sigma de 3.0 rezultă, la fel, rezultate relativ mai precise. Din această cauză s-au folosit acești parametri la rularea acestui proces.

6.2.1. Comparație obiectivă

Comparația obiectivă a calității algoritmilor se efectuează cu ajutorul rezultatelor procesului de măsurarea calității, mai precis datele salvate în fișierul Excel. Pentru fiecare set de imagini de intrare vom afișa valorile cele mai bune din punctul de vedere al MSE și PSNR-ului.

6.2.1.1. Imaginea perfectă

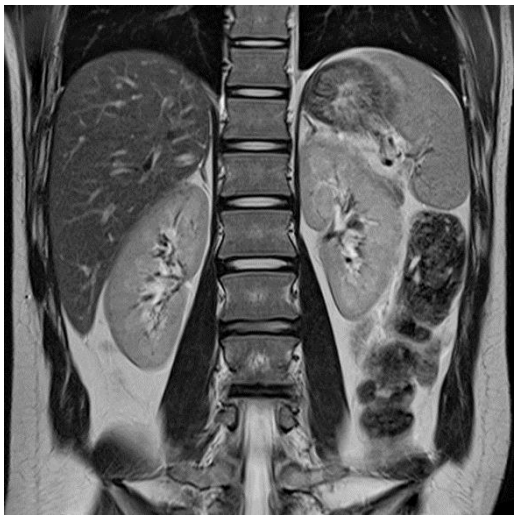

Figura 6.1 imaginea perfectă, de la sursa [31]

6.2.1.2. Setul de imagini 1

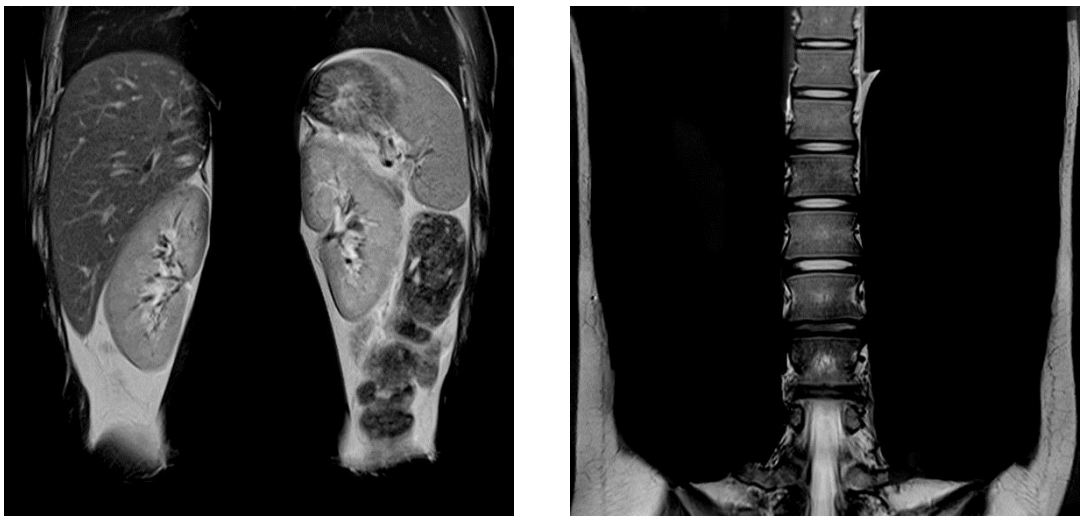

Figura 6.2 Stânga: mri_soft_2.jpg, dreapta: mri_hard_2.jpg

Rezultatele cele mai bune obținute din results.xls:

Tabel 6.3 Metrici de calitate pentru setul de imagini 1

Algoritm_numeimagini_postprocesare	**MSE**	**PSNR**
Maximum mri_soft_2 + mri_hard_2_eroded	609.8186	20.2788
Laplacian_3_3.0 mri_soft_2 + mri_hard_2_eroded	725.6997	19.52323
Haar_2 mri_soft_2 + mri_hard_2_smoothed	787.247	19.16969

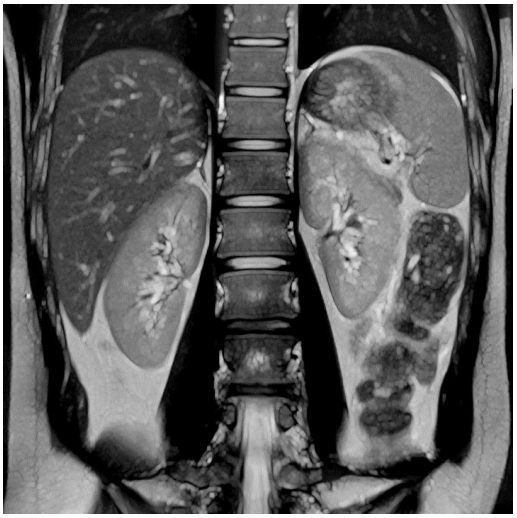

Figura 6.3 Maximum mri_soft_2 + mri_hard_2_eroded

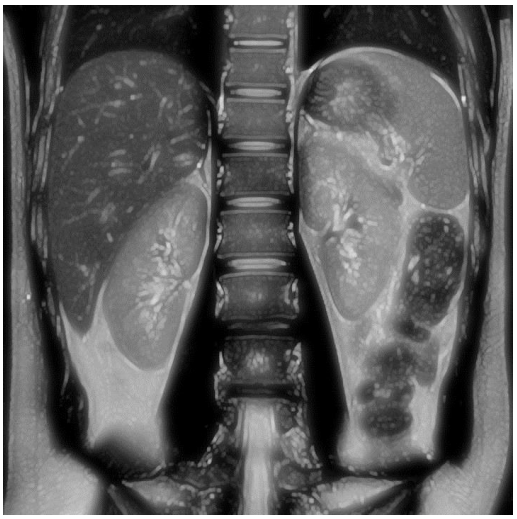

Figura 6.4 Laplacian_3_3.0 mri_soft_2 + mri_hard_2_eroded

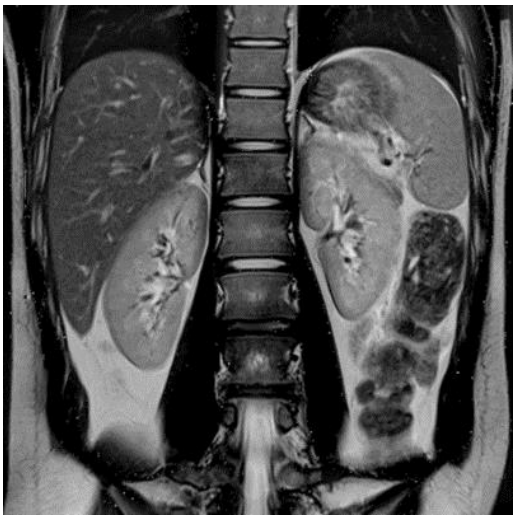

Figura 6.5 Haar_2 mri_soft_2 + mri_hard_2_smoothed

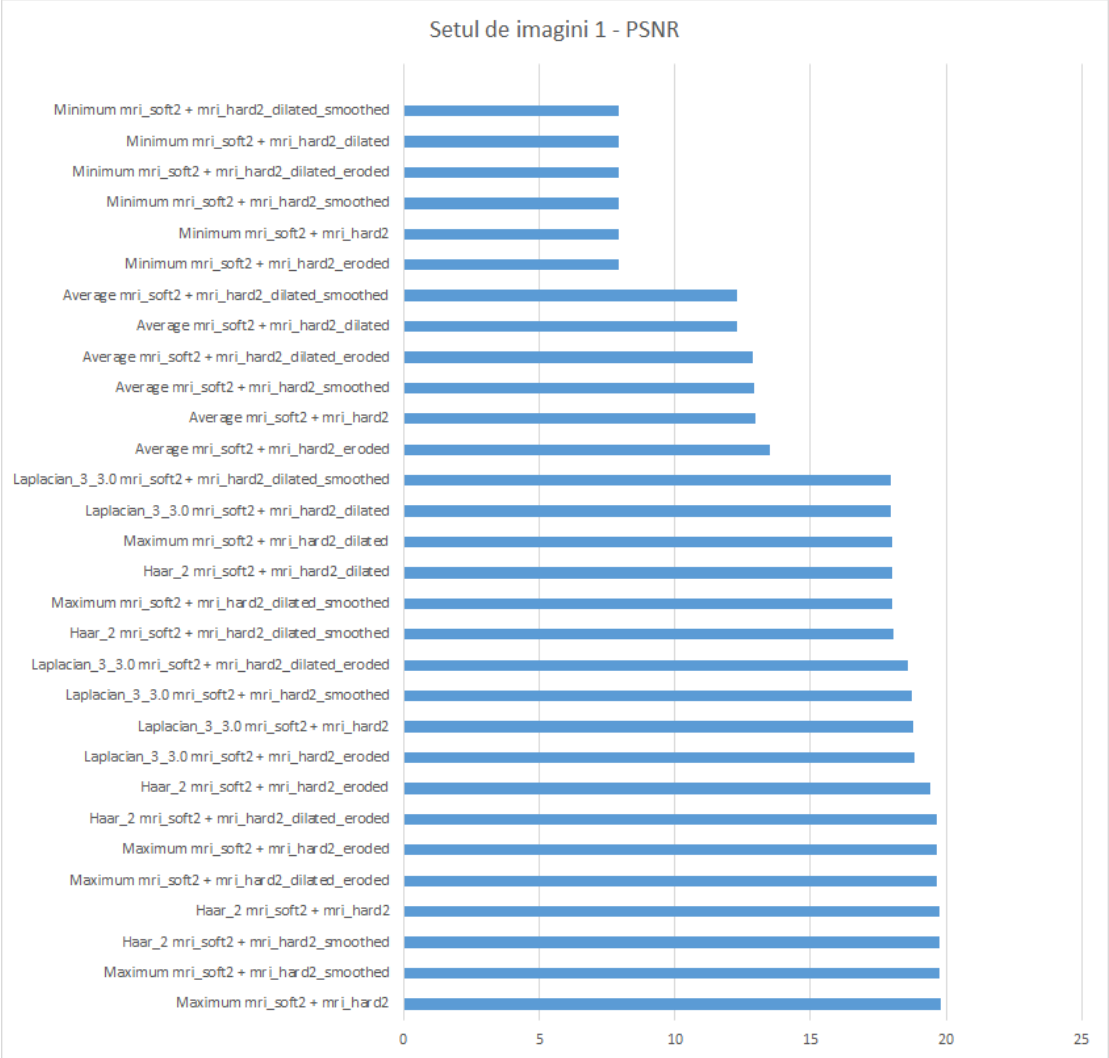

Figura 6.6 Diagrama de PSNR pentru setul de imagini 1

6.2.1.3. Setul de imagini 2

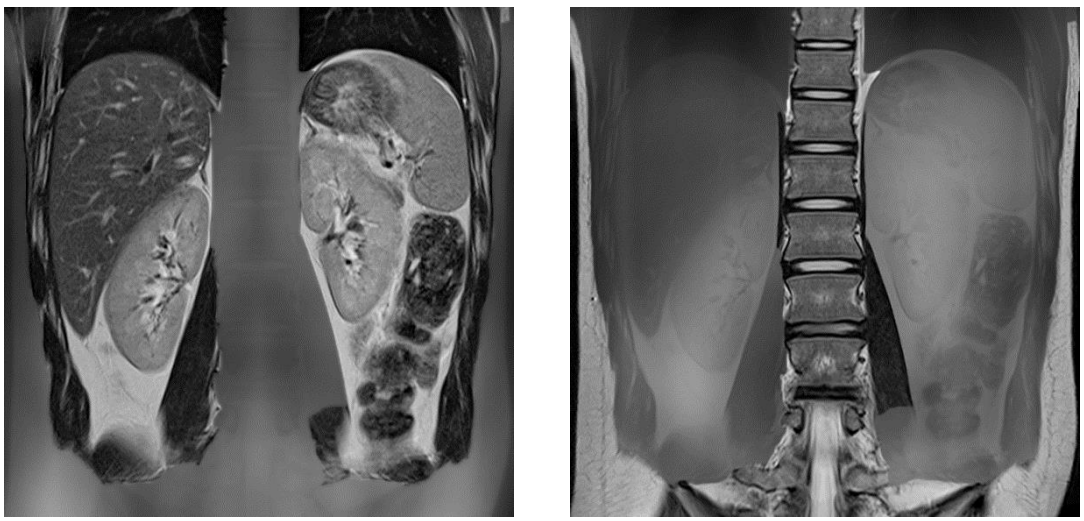

Figura 6.7 Stânga: mri_soft.jpg, dreapta: mri_hard.jpg

Tabel 6.4 Metrici de calitate pentru setul de imagini 2

Algoritm_numeimagini_postprocesare	**MSE**	**PSNR**
Average mri_soft + mri_hard	421.3322	21.88456
Haar_2 mri_soft + mri_hard_dilated_smoothed	531.9417	20.87216
Maximum mri_soft + mri_hard	640.0845	20.06843

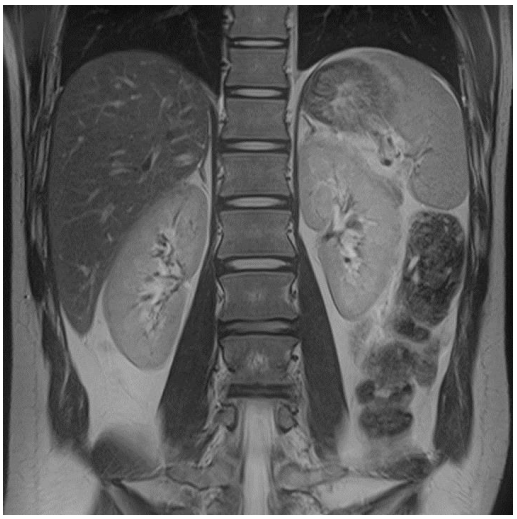

Figura 6.8 Average mri_soft + mri_hard

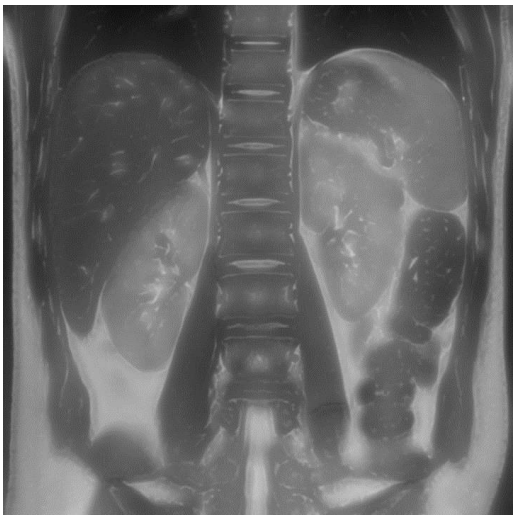

Figura 6.9 Haar_2 mri_soft + mri_hard_dilated_smoothed

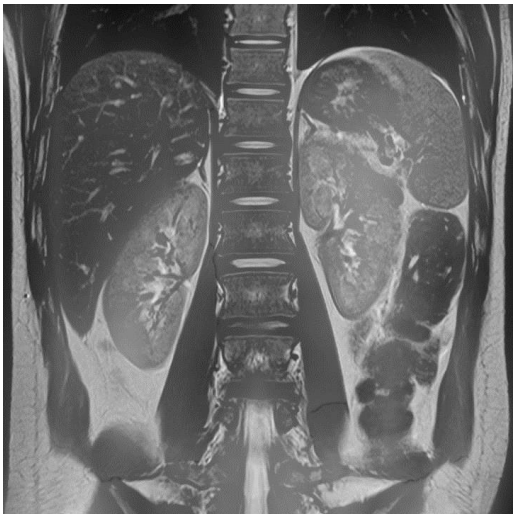

Figura 6.10 Maximum mri_soft + mri_hard

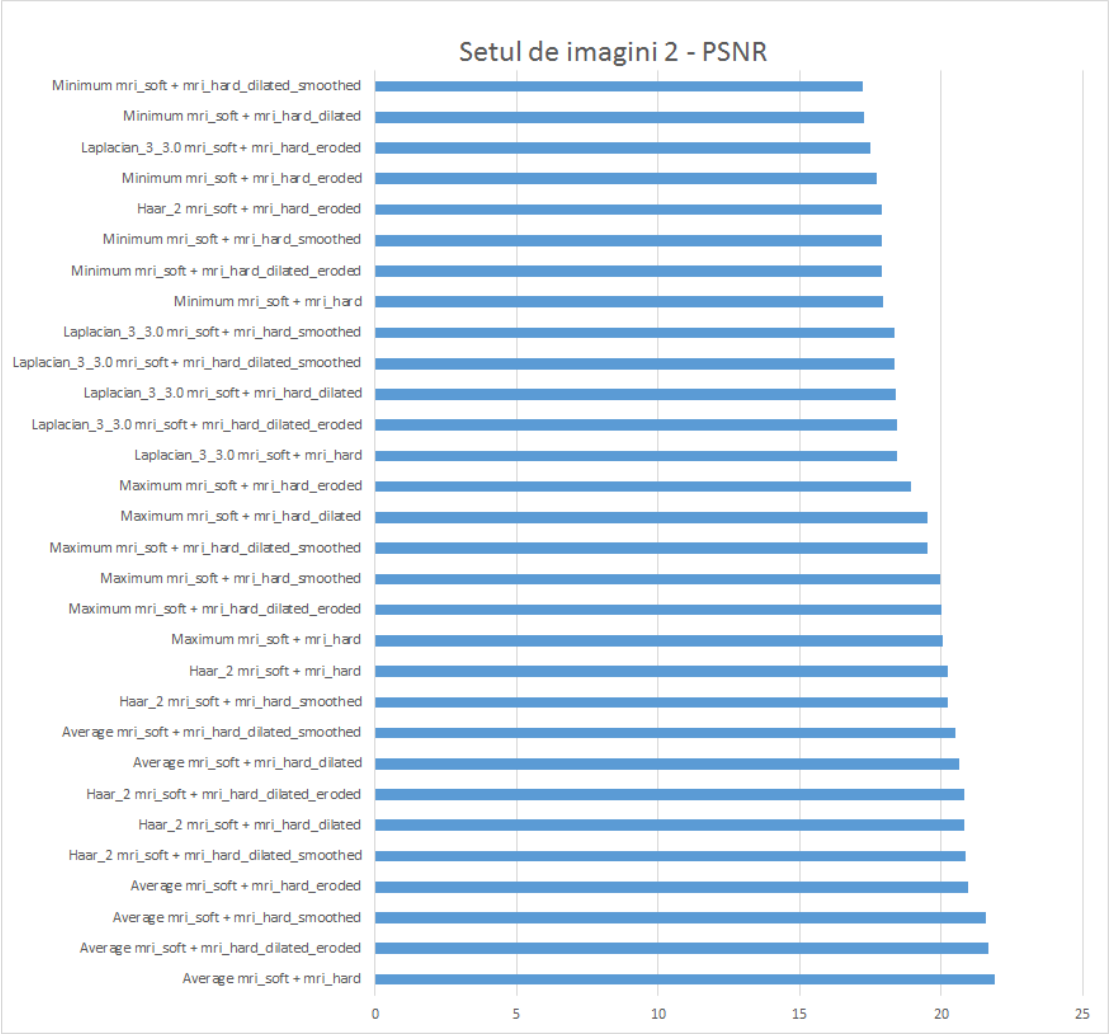

Figura 6.11 Diagrama de PSNR pentru setul de imagini 2

6.2.1.4. Setul de imagini 3

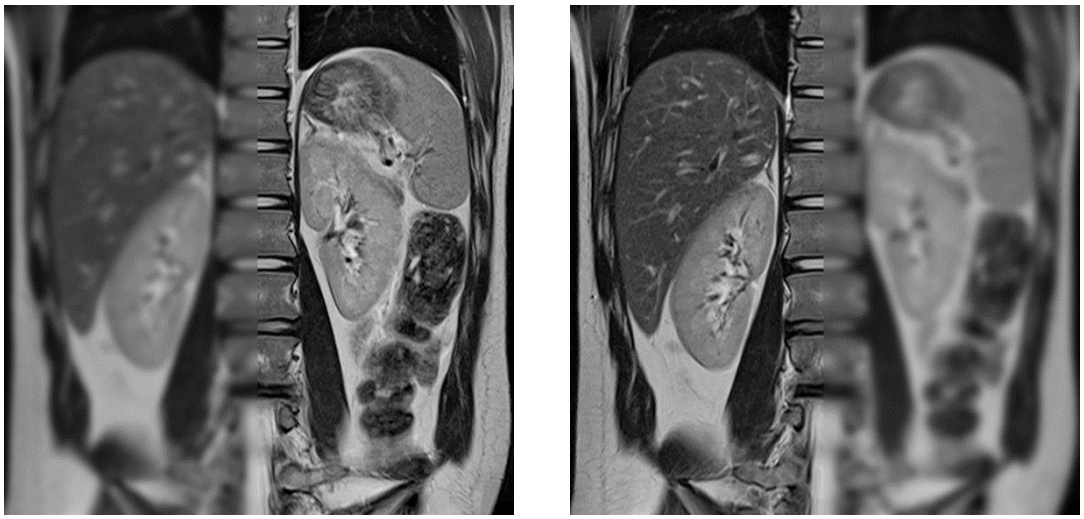

Figura 6.12 Stânga: mri_left_blurred.jpg, dreapta: mri_right_blurred.jpg

Tabel 6.5 Metrici de calitate pentru setul de imagini 3

Algoritm_numeimagini_postprocesare	**MSE**	**PSNR**
Average mri_left_blurred + mri_right_blurred	123.3515	27.21936
Haar_2 mri_left_blurred + mri_right_blurred_dilated_eroded	190.5589	25.33051
Maximum mri_left_blurred + mri_right_blurred	237.214	24.3794

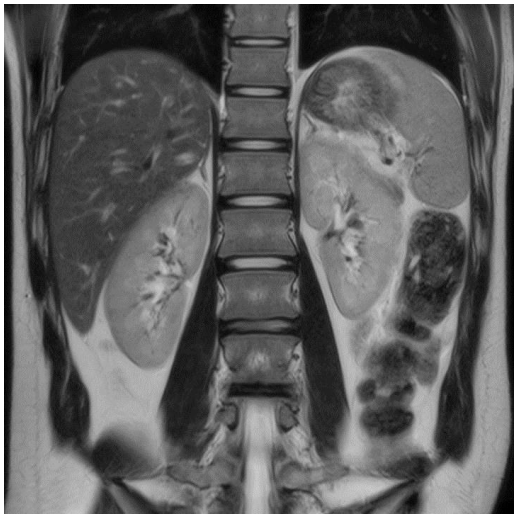

Figura 6.13 Average mri_left_blurred + mri_right_blurred

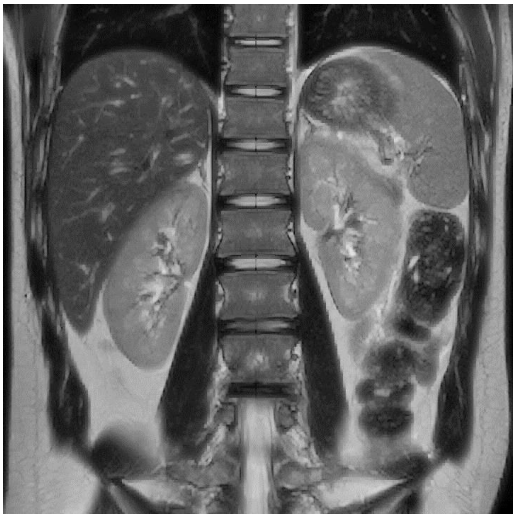

Figura 6.14 Haar_2 mri_left_blurred + mri_right_blurred_dilated_eroded

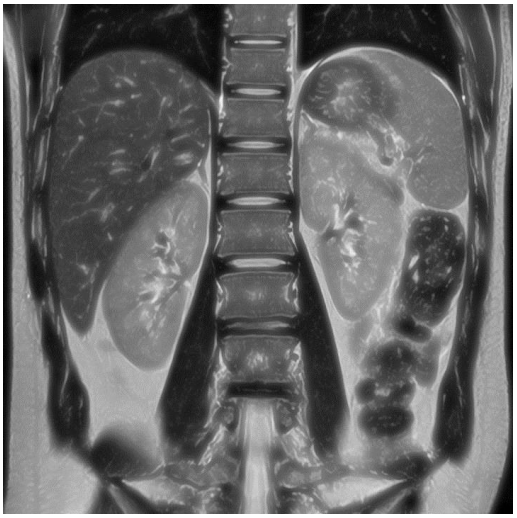

Figura 6.15 Maximum mri_left_blurred + mri_right_blurred

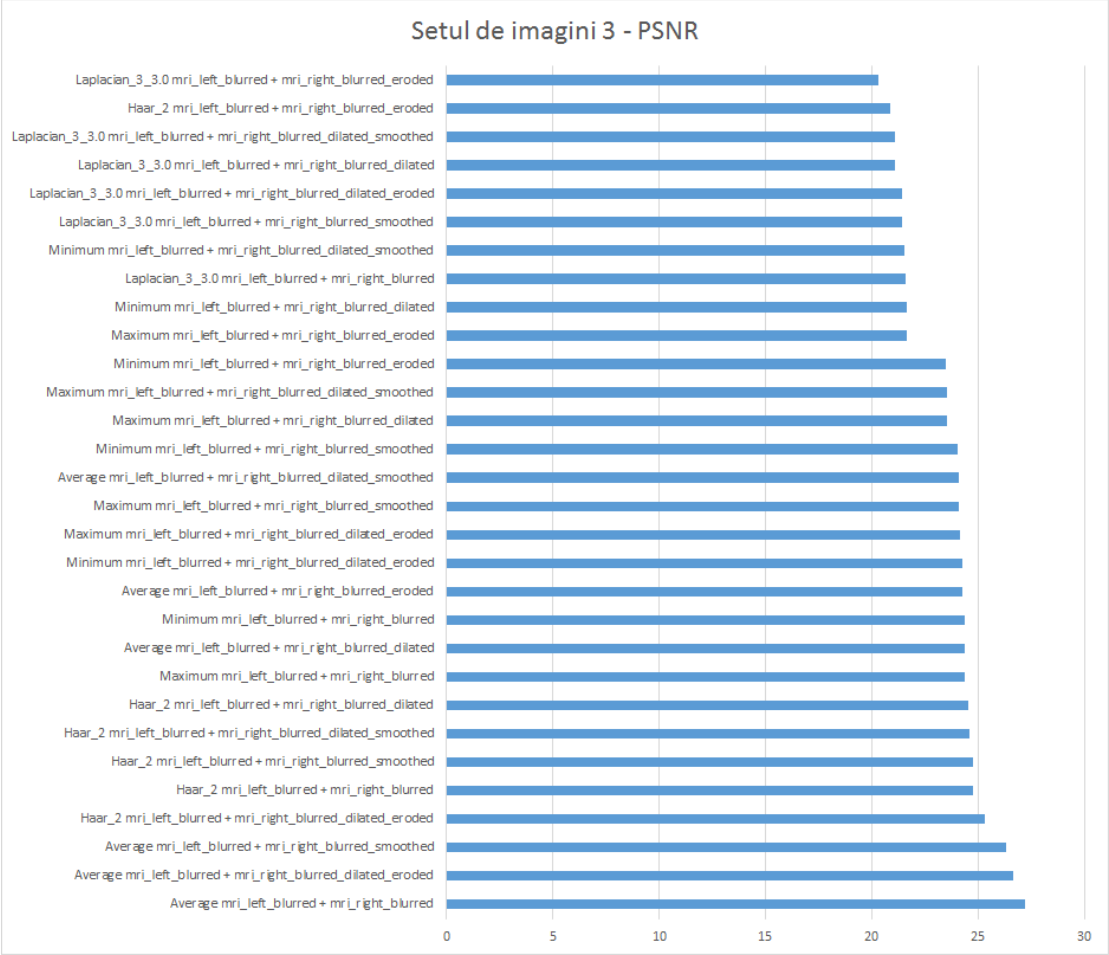

Figura 6.16 Diagrama de PSNR pentru setul de imagini 3

6.2.1.5. Media rezultatelor

Generalizând rezultatele după tipul de algoritmi, putem calcula media PSNR-ului:

1. Haar: 21.78
2. Maxim aritmetic: 21.56
3. Medie aritmetică: 20.68
4. Laplacian: 19.79
5. Minim aritmetic: 16.61

Observație importantă despre aceste rezultate: în cazul în care imaginile de intrare nu sunt complementare, cum este în cazul primului set de imagini, metodele aritmetice produc rezultate foarte proaste.

6.2.2. Comparație subiectivă

Am comparat aceste rezultate, creând un sondaj online cu ajutorul unui Google Docs Spreadsheet. Sondajul a conținut și imaginea perfectă, cu care trebuiau să compare voluntarii imaginile rezultate a fuziunii pentru cele 3 set-uri de imagini de intrare. Au participat 35 de persoane, iar rezultatele sunt următoare:

6.2.2.1. Setul de imagini 1:

1. Haar: 25 de voturi, 71%
2. Maxim aritmetic: 9 voturi, 26%

3. Laplacian: 1 vot, 3%

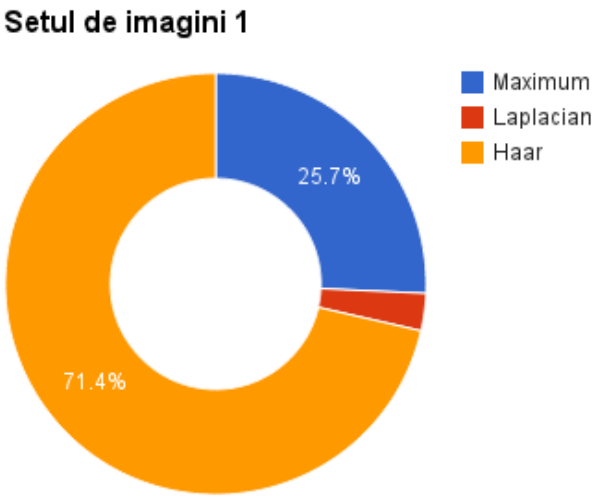

Figura 6.17 Diagrama de rezultate pentru setul de imagini 1

6.2.2.2. Setul de imagini 2:

1. Medie aritmetica: 19 voturi, 54%
2. Maxim aritmetic: 13 voturi, 37%
3. Haar: 3 voturi, 9%

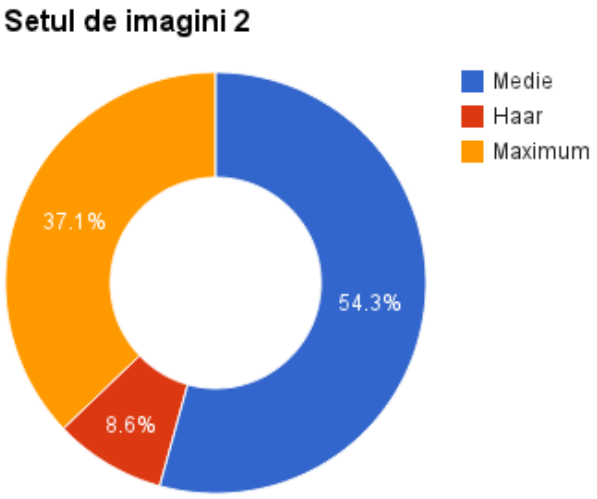

Figura 6.18 Diagrama de rezultate pentru setul de imagini 2

6.2.2.3. Setul de imagini 3:

1. Medie aritmetică: 13 voturi, 37%
2. Maxim aritmetic: 13 voturi, 37%
3. Haar: 9 voturi, 26%

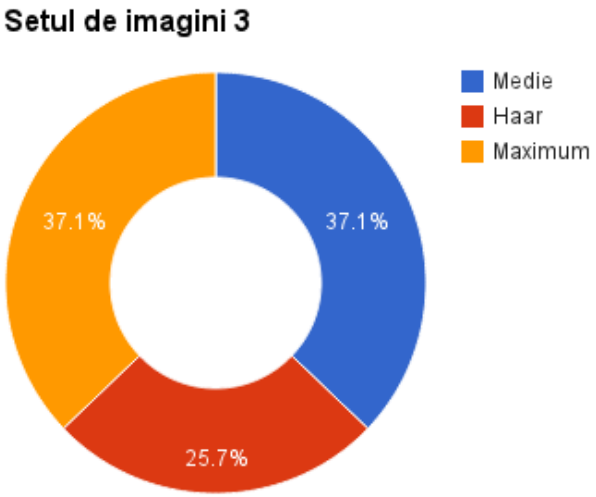

Figura 6.19 Diagrama de rezultate pentru setul de imagini 3

6.2.3. Rezultate finale

Din rezultatele sondajului putem calcula procentajul voturilor pentru fiecare algoritm:

1. Haar: 71% + 9% + 26% = 106%
   106 / 3 = 36%
2. Maxim aritmetic: 26% + 37 + 37% = 100%
   100 / 3 = 33%
3. Medie aritmetica: 54% + 37% = 91%
   91/3 = 30%
4. Laplacian: 3%
   3 / 3 = 1%

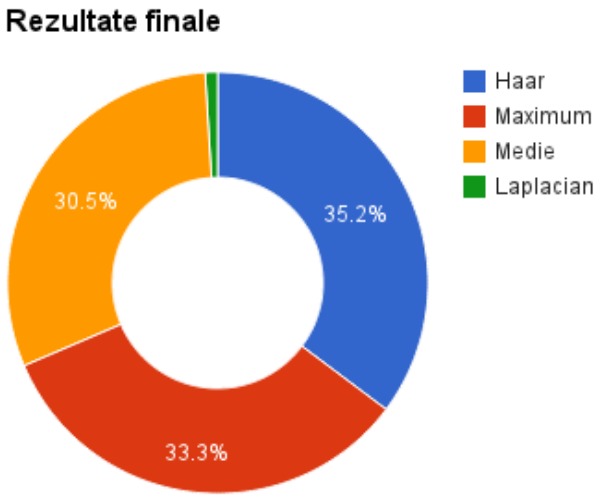

Figura 6.20 Diagrama de rezultate finale

Din cele 2 tipuri de comparații putem afirma că rezultatele au fost similare, care este dovada că metricile de calitate oferă rezultate automatizate similare a modului de gândire uman. Combinând rezultatele, primim lista finală a algoritmilor, sortate descrescător din punctul de vedere al calității:

1. Haar
2. Maxim aritmetic
3. Medie aritmetica
4. Laplacian
5. Minim aritmetic

De observat, din nou, rezultatele transformatei Haar în cazul în care imaginile de intrare nu sunt complementare una față de cealaltă. În aceste cazuri, de obicei, algoritmii aritmetici produc rezultate mult mai proaste.

Capitolul 7. Manual de Instalare și Utilizare

7.1. Instalare

Sistemul pe care se dorește rularea aplicației trebuie sa îndeplinească următoarele cerințe:

7.1.1. Cerințe hardware

Aplicația fiind relativ mică, procesele ei nu necesită putere foarte mare de calcul. O configurație minimă pe care rulează un sistem de operare modern este capabil să ruleze și aplicația fără probleme, singura chestie afectată ar putea fi durata proceselor de fuziune sau măsurarea calității. Totuși, orientativ, specificăm următoarele cerințe minime:

- Procesor Intel® Pentium™ de 2 GHz sau mai bun
- Memorie RAM 2GB sau mai mult
- Spațiu de stocare minim 10MB pentru aplicație, și 20 de MB pentru rezultate

7.1.2. Cerințe software

Independența de platformă a proiectului oferă posibilitatea folosirii oricărui sistem de operare preferat de către utilizator.

Aplicația fiind scrisă în limbajul de programare Java™, avem nevoie de un mediu Java Runtime Environment pentru a o rula. Acest JRE™ se poate instala de pe pagina web [32], selectând sistemul de operare și arhitectura aferentă. După instalare se poate deschide fișierul jar al aplicației.

Dacă se dorește dezvoltarea aplicației, construirea ei din codul sursă, trebuie instalat un Java Development Kit, sau JDK™. Acesta se poate instala de pe [33], selectând, la fel, opțiunile corecte la descărcare. Codul sursă al proiectului se poate clona local de pe GitHub, de pe adresa https://github.com/a-henning/DicomFusion.

De observat folosirea Maven-ului pentru descărcarea dependințelor proiectului și automatizarea procesului de construire. Această unealtă poate fi deja integrată în editorul de cod sursă, sau poate fi descărcată de pe pagina proiectului [15].

7.2. Utilizare

Dacă utilizatorul s-a asigurat ca sistemul lui îndeplinește cerințele de mai sus, poate să procedeze la pasul de pornirea aplicației. În funcție de ce cale a ales, dezvoltarea aplicației sau doar folosirea sa, el o sa ajungă la un fișier executabil cu extensia jar. Cu următoarele imagini se dorește ilustrarea pașilor de utilizare a sistemului.

Se pornește aplicația, și ni se arată următoarea interfață grafică:

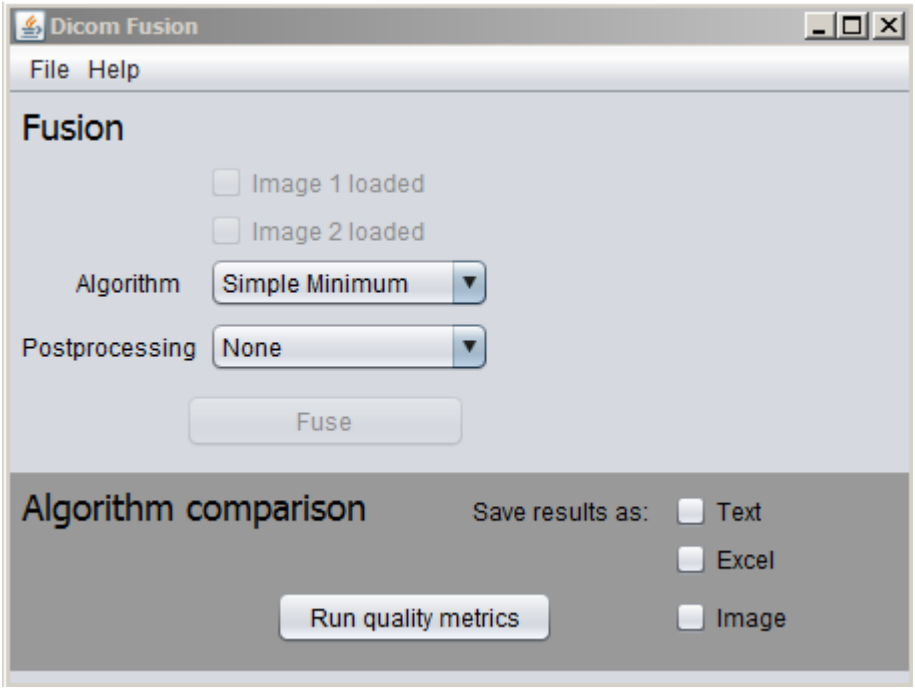

Figura 7.1 Interfața grafică a aplicației

Se deschid fișierele DICOM din meniul din stânga sus.

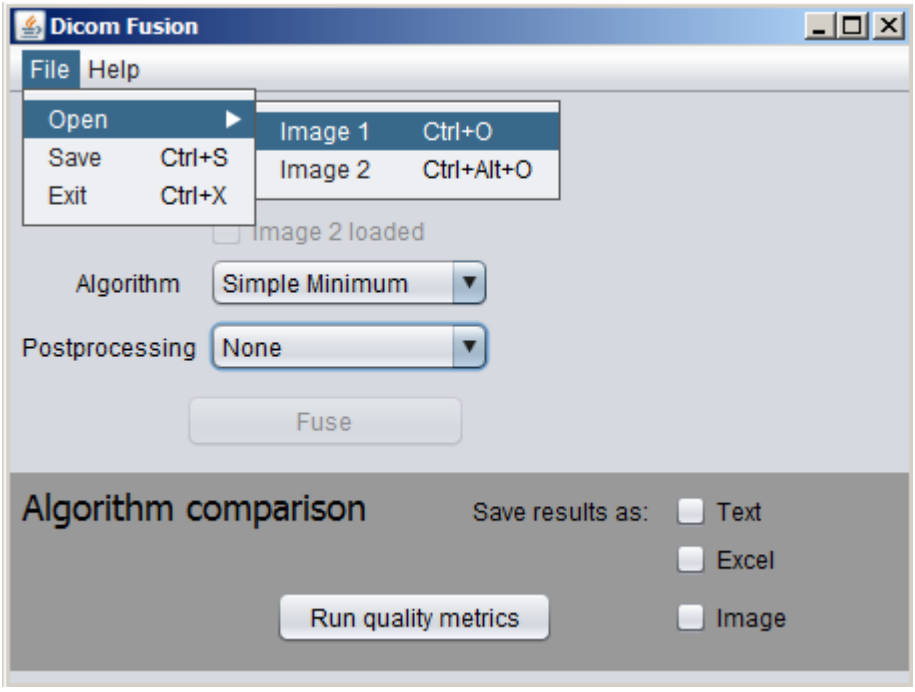

Figura 7.2 Deschiderea unui fișier DICOM

Se alege fișierul dorit de pe mașina locală. Pentru a schimba filtrul de extensie, se alege la „Files of Type” All în loc de DICOM.

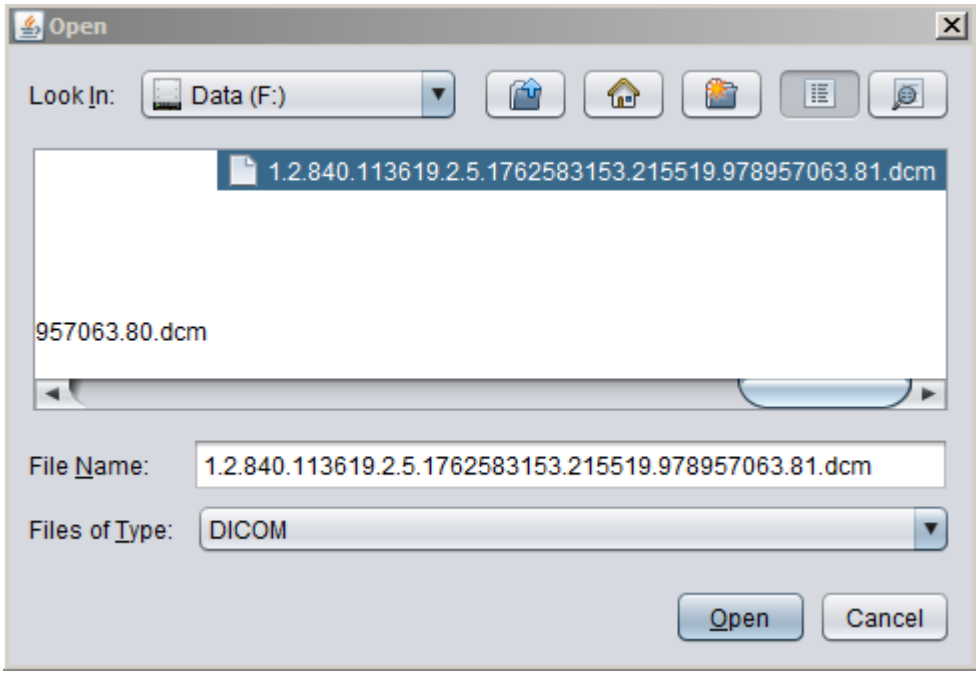

Figura 7.3 Alegerea fișierului

După ce am ales fișierul si am dat click pe open, aplicația deschide imaginea din înăuntrul fișierului într-un cadru nou.

Figura 7.4 Afișarea imaginii din fișier

În cazul stivelor de imagini, se va afișa o săgeată mică în coltul de stânga jos a cadrului, și un scroll-er prin care putem vizualiza imaginile din stivă.

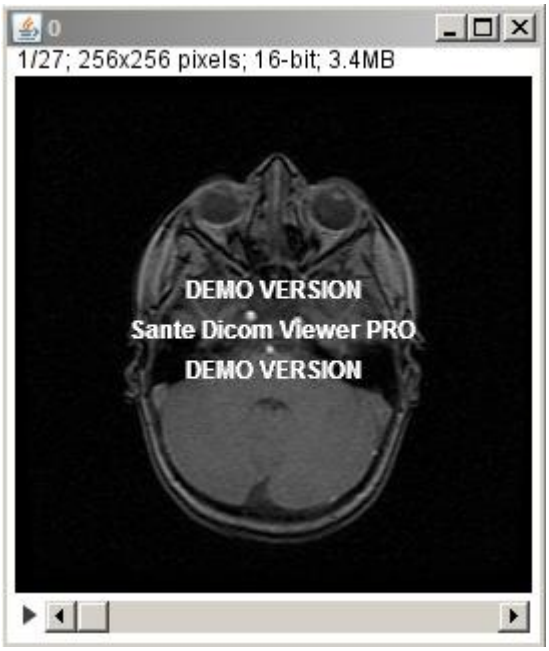

Figura 7.5 Afișarea stivei de imagine din fișier

Următorul pas constă in alegerea algoritmului de fuziune și eventualelor parametri a acestuia.

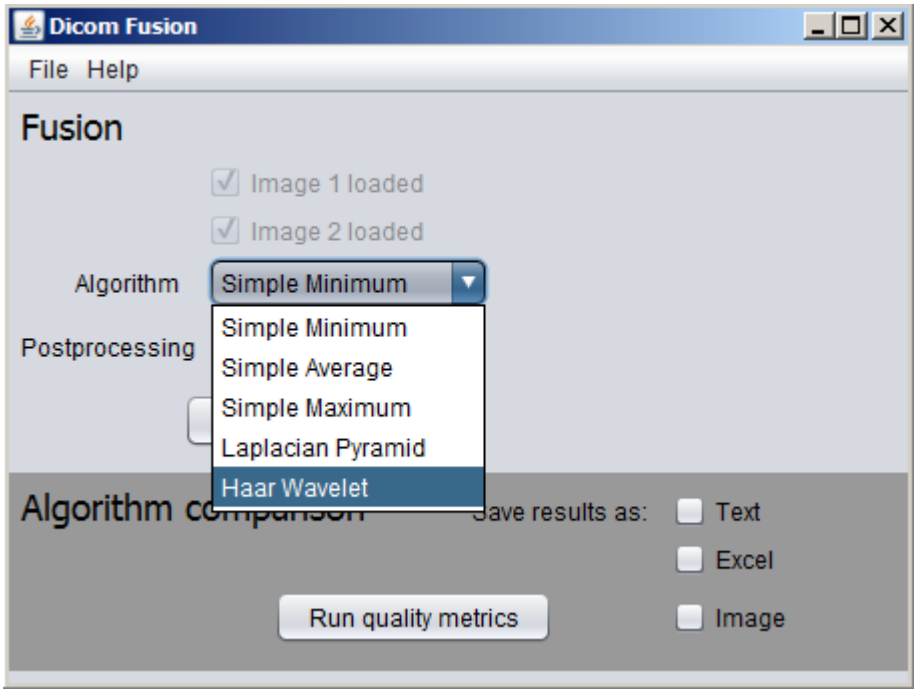

Figura 7.6 Alegerea algoritmului de fuziune

După acesta, se alege metoda de postprocesare dorită.

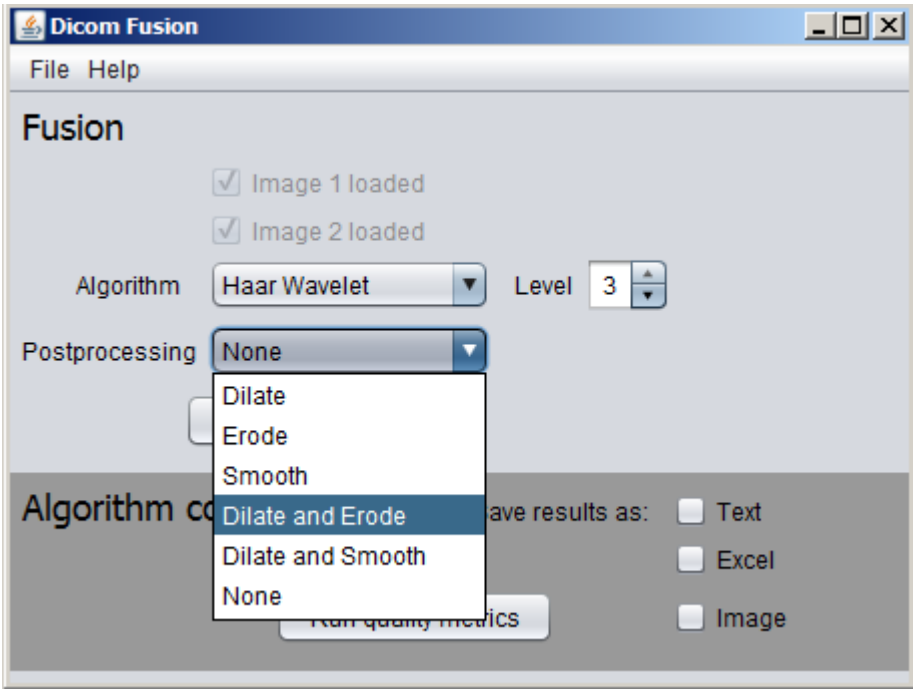

Figura 7.7 Alegerea parametrilor algoritmului și metodei de postprocesare

Se dă click pe butonul de fuziune „Fuse”, și după rularea procesului ni se va deschide imaginea rezultat într-o fereastra nouă. De observat titlul ferestrei, care este și titlul imaginii, alcătuit din algoritmul de fuziune, urmat de parametri a acestuia, imaginile sursă, metoda de postprocesare și timpul în milisecunde în care s-a terminat operația.

Figura 7.8 Rezultatul fuziunii

Opțional, după afișarea rezultatului acesta se și poate salva, din meniul din stânga sus.

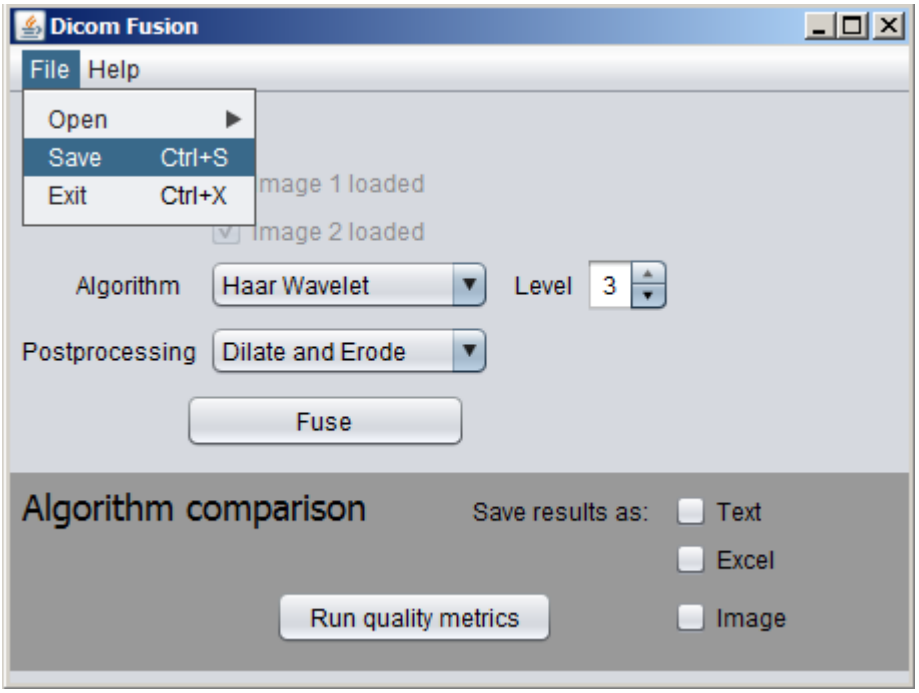

Figura 7.9 Salvarea rezultatului

În caz de succes va apărea un mesaj de informații, care ne spune unde a fost salvată imaginea și cu ce nume.

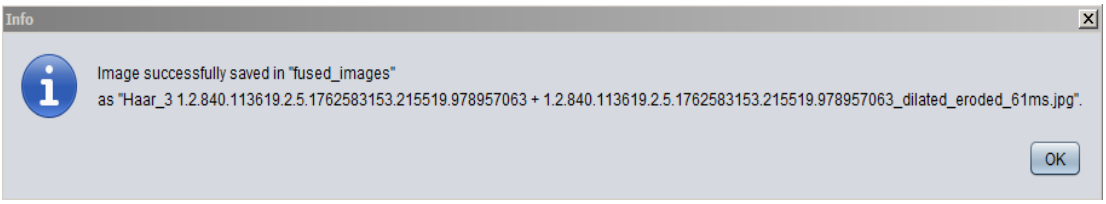

Figura 7.10 Mesajul de succes la salvare

La procesul de rularea metricilor de calitate, se pot alege modurile de salvare a rezultatelor din checkbox-urile de lângă titlu.

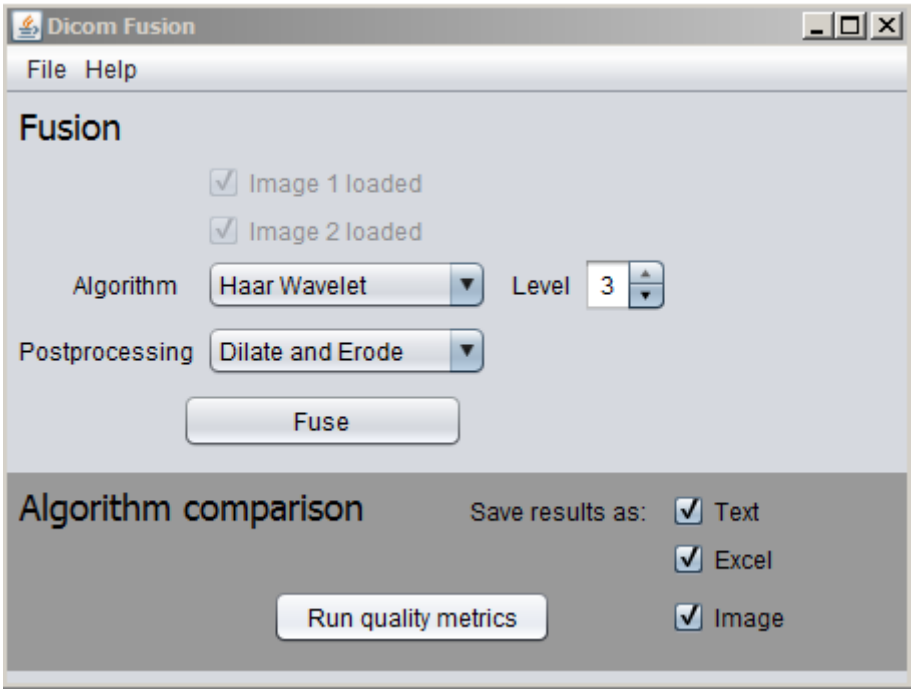

Figura 7.11 Alegerea modului de salvare a metricilor de calitate

În cazul în care operația a fost efectuată cu succes, va apărea un mesaj cu timpul în care s-a terminat aceasta. Se afișează timpul în care s-au calculat efectiv metricile de calitate, și timpul în care s-a terminat și tot procesul, care include și salvarea rezultatelor în funcție de opțiunile alese de la pasul anterior.

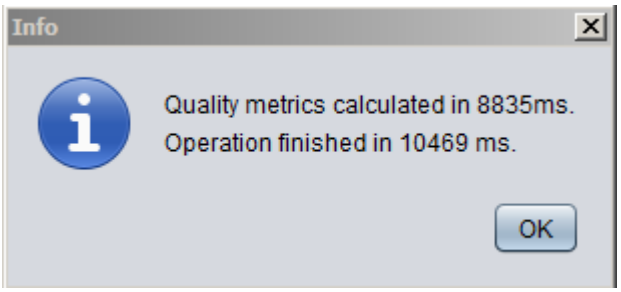

Figura 7.12 Mesajul de succes după rularea procesului de calcularea metricilor de calitate

În caz de eroare sau avertizare, se va deschide o nouă fereastră cu mesajul respectiv. Tipul mesajul poate fi de informație, de avertizare sau de eroare, pentru fiecare tip de mesaj modificându-se icoana ferestrei. Excepțiile tratate de aplicație sunt afișate și ele în acest fel.

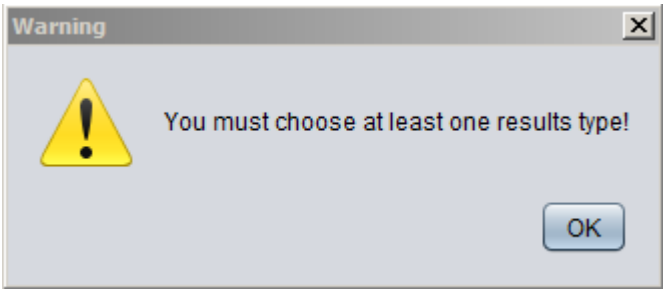

Figura 7.13 Mesaj de avertizare

Capitolul 8. Concluzii

8.1. Contribuția mea

Prin acest proiect am implementat o componentă a sistemului medical complex prezentat în capitolul 2, modulul de Fuziunea imaginilor medicale. Acest raport contribuie la evoluarea tehnologiei de fuziune a informației relevante din mai multe surse și duce cu un pas mai înainte posibilitatea ca sistemul de față să fie implementat și în centrele medicale, ajutând la diagnostice mai precise și mai rapide, în acest mod făcând vizita medicală o experiență un pic mai plăcută și confortabilă. Până acum, interacțiunea între componentele acestui sistem este doar teoretică, realizarea ei fiind o sarcină individuală.

Prin implementarea mai multor algoritmi de fuziune, am încercat să ofer posibilitatea de experimentare pentru utilizator din punctul de vedere al rezultatelor fuziunii. Iar, în cazul în care nu se poate decide care algoritm și metoda de postprocesare să folosească, poate apela la metricile de calitate, care oferă o comparație strict obiectiva între metode de fuziune. Modulele sunt create în așa fel, încât sa fie reutilizabile și extensibile, în acest mod aplicația fiind integrabilă și în alte proiecte, sau se pot folosi și dezvolta anumite componente din aplicație.

Am proiectat și am creat aplicația astfel încât ea să fie independentă de platformă, să fie utilizabilă pe sistemul de operare și arhitectura preferată fiecărui utilizator. Prin interfața grafică simplă și intuitivă atât rularea cât și folosirea aplicației este facilă.

Cred că aplicația este folositoare și este un progres în direcția bună, cea de a dezvolta și îmbunătății știința medicală, un domeniu a cărui evoluție nu ține pasul cu evoluția, de exemplu, a tehnologiei mobile sau smartphone-urilor. Deși mulți neglijează acest aspect a vieții, sănătatea este cel mai important lucru. Cred că tehnologia informaticii aplicată în medicină poate salva vieți.

8.2. Analiza critică a rezultatelor obținute

Prin acest studiu am ajuns asupra unor rezultate concrete în ceea ce privește metoda cea mai bună dintre cele studiate în această lucrare, pentru fuziunea imaginilor medicale atât din punctul de vedere al preciziei cât și din perspectiva vitezei de procesare. După testare și comparații am ajuns la concluzia că algoritmul Haar Wavelet oferă rezultatele cele mai bune din punct de vedere calitativ, însă nu este perfect. De cele mai multe ori introduce o anumită cantitate de zgomot în rezultat, din această cauză se recomandă a folosi întotdeauna cu o metodă de postprocesare. Acest zgomot apare, cel mai probabil, când se redimensionează imaginile la transformata Haar inversă, unde se măresc dimensiunile, iar pixelii noi se interpolează.

Din punctul de vedere al vitezei, metodele aritmetice sunt cele mai rapide. Însă acestea eșuează de regulă când imaginile de intrare nu sunt complementare una față de cealaltă, în sensul că nu se poate ajunge la „imaginea perfectă” doar luând valorile maxime ale pixelilor, de exemplu. Pentru fiecare caz particular, o metoda aritmetica poate produce rezultate foarte bune, dar per total, Haar Wavelet se comporta cel mai bine indiferent de situație.

Aplicația abordă o serie de scenarii de fișiere de intrare, cum ar fi stivele de imagini sau dimensiuni diferite ale acestora, însă este sensibilă la unghiul și adâncimea la care a fost făcută imaginea medicală.

Este inferioară aplicațiilor comerciale existente, dar are avantajul de a fi un proiect open-source și gratuit.

8.3. Dezvoltări și îmbunătățiri ulterioare posibile

Aplicația este funcțională și folositoare pentru mai multe cazuri de imagistică medicală, însă se poate îmbunătății pentru a fi și mai valoroasă și competitivă față de celelalte aplicații de pe piață:

- Fuziunea multi-imaginilor: aplicația să nu fie limitată la doar două imagini de intrare.
- Recunoașterea și calibrarea imaginilor în funcție unghiul și adâncimea în care au fost luate: dacă avem de-a face cu același organ, sau parte a corpului, imaginile se translatează și redimensionează astfel încât să se poată face fuziunea.
- Combinarea algoritmilor între ei: pentru a ajunge la un rezultat cât mai precis, s-ar putea combina algoritmii. De exemplu medie aritmetică și Haar Wavelet în cazul în care se detectează că imaginile sunt complementare.
- Detectarea dacă imaginile nu sunt făcute asupra aceluiași pacient, sau aceleași parte a corpului pacientului: din metadatele fișierelor DICOM putem extrage aceste informații, și putem face fuziunea doar în cazul în care vorbim de același subiect.
- Fuziunea și a altor fișiere de tip imagine: în prezent, modulul de fuziune accepta doar fișiere DICOM. Însă ar fi folositor și permiterea altor formate de imagine, pentru ca așa utilizatorul ar putea experimenta cu diferite date de intrare.
- Salvarea rezultatelor ca fișiere DICOM: acesta este posibil doar în cazul în care metadatele din fișierele de intrare sunt similare. Prin acest mod rezultatele etapei de fuziune s-ar putea refolosi într-o operație următoare.
- Modificarea parametrilor procesului de măsurarea calității din interfața grafică: utilizatorul ar putea alege ce algoritmi dorește să compare, ce metode de postprocesare respectiv pe ce seturi de imagini dorește verificarea.
- Afișarea mai în detaliu a zonelor interesante: aplicația ar putea identifica zonele mai importante, cum ar fi o tumoare, și să o afișeze mai în detaliu, fără fundal.

Bibliografie

[1] Imagistică Medicală, Wikipedia, http://ro.wikipedia.org/wiki/Imagistic%C4%83_medical%C4%83
[2] A. A. Feiler, A.-M. Ungureanu, „Manual de radiologie și imagistică medicală”
[3] DICOM, Wikipedia, http://en.wikipedia.org/wiki/DICOM
[4] Deepak Kumar Sahu, M.P. Parsai, “Different Image Fusion Techniques – A Critical Review”
[5] Image Fusion, Wikipeda, http://en.wikipedia.org/wiki/Image_fusion
[6] Mirada Medical XD3, http://www.mirada-medical.com/products/xd3/
[7] Velocity Medical, http://www.velocitymedical.com/solutions/
[8] Keosys Imagys-Cloud Services, http://www.keosys.com/eng/visualize/workstation_nm.php
[9] V. R. Gupta, V. K. Agarwal, S. L. Tade, „Comparison of Medical Image Fusion Algorithm for Preserving the Edge Information Based On Improved Wavelet Coefficient Contrast”, http://www.ijetae.com/files/Volume2Issue5/IJETAE_0512_55.pdf
[10] ImageJ, http://imagej.nih.gov/ij/features.html
[11] JExcel API, http://jexcelapi.sourceforge.net/
[12] Lesson: Learning Swing with the NetBeans IDE, http://docs.oracle.com/javase/tutorial/uiswing/learn/
[13] JUnit, http://junit.org/
[14] JUnit, Wikipedia, http://en.wikipedia.org/wiki/JUnit
[15] Apache Maven, http://maven.apache.org/
[16] DICOM Specification Overview: Basic DICOM File Structure, http://www.leadtools.com/sdk/medical/dicom-spec1.htm
[17] C. Melenti, “Formate de imagine”, curs Tehnologii Multimedia
[18] Discrete Wavelet Transfom, Wikipedia, http://en.wikipedia.org/wiki/Discrete_wavelet_transform
[19] Spatial domain, Frequency domain, Time domain and Temporal domain, Image Processing And Pattern Recognition, http://ippr-practical.blogspot.ro/2012/04/spatial-domain-frequency-domain-time.html
[20] G. Peyré, „2-D Haar Wavelets”, https://www.ceremade.dauphine.fr/~peyre/numerical-tour/tours/wavelet_2_haar2d/
[21] MATLAB Image Compression using Haar Wavelet Transform, http://www.eeweb.com/electronics-forum/matlab-image-compression-using-haar-wavelet-transform
[22] P. Jorgensen, „Image Decomposition using Haar Wavelet”, http://homepage.math.uiowa.edu/~jorgen/Haar.html
[23] S. Ludwig, „Implementation of a spatio-temporal Laplacian image pyramid on the GPU”, http://www.gazecom.eu/FILES/ludw08.pdf
[24] R. Dănescu, “Operatii morfologice”, curs Procesarea Imagnilior
[25] R. Fisher, S. Perkins, A. Walker and E. Wolfart, „Dilation”, http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm
[26] Dilation (morphology), Wikipedia, http://en.wikipedia.org/wiki/Dilation_(morphology)
[27] R. Fisher, S. Perkins, A. Walker and E. Wolfart, „Erosion”, http://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm
[28] Erosion (morphology), Wikipedia, http://en.wikipedia.org/wiki/Erosion_(morphology)
[29] Smoothing, Wikipedia, http://en.wikipedia.org/wiki/Smoothing
[30] ImagePlus, ImageJ API, http://imagej.nih.gov/ij/developer/api/ij/ImagePlus.html
[31] Siemens Healthcare, MAGNETOM ESSENZA – Abdomen, http://www.healthcare.siemens.de/magnetic-resonance-imaging/magnetom-world/toolkit/clinical-images/essenza-abdomen
[32] Oracle, „Java SE Runtime Environment 7 Downloads”, http://www.oracle.com/technetwork/java/javase/downloads/java-se-jre-7-download-432155.html
[33] Oracle, “Java SE Downloads”, http://www.oracle.com/technetwork/java/javase/downloads/index.html?ssSourceSiteId=otnjp