EPS
Escola Politècnica Superior
UdG

Projecte/Treball Fi de Carrera

Estudi: Eng. Tècn. Informàtica de Sistemes. Pla 2001

Títol: Pre-procés i segmentació en imatges d'ultrasons 2D i 3D de la pròstata

Document: Memòria

Alumne: Cristina Ríos Robles

Director/Tutor: Robert Martí Marly
Departament: Arquitectura i Tecnologia de Computadors
Àrea: ATC

Convocatòria (mes/any): 09/08

Índex

Índex de figures

Capítol 1

Introducció

Projecte s'ha dut a terme amb el Grup de visió per computador del departament d'Arquitectura i Tecnologia de computadors (ATC) de la Universitat de Girona. Cal esmentar que aquest PFC s'emmarca dins del projecte de recerca anomenat PROSCAN, que té com a finalitat el desenvolupar una aplicació informàtica per tal de guiar la biòpsia en el càncer de pròstata. Per situar-nos en antecedents en aquest capítol es farà una breu introducció a l'entorn del projecte, en que consisteix el càncer de pròstata, eines utilitzades per tal de desenvolupar el projecte i finalitat d'aquest.

1.1 Càncer de pròstata

1.1.1 La pròstata

El càncer de pròstata és el segon càncer més comú en els homes després del de pell i també és el segon més mortífer després del càncer de pulmó. Concretament la pròstata és una de les glàndules sexuals dels homes, produeix el líquid seminal que forma part de l'esperma. Tal i com mostra la figura 1.1 està situada a sobre del recte i a sota de la bufeta de l'orina, envoltant la uretra fins al punt que s'ajunta amb la bufeta. Els problemes sorgeixen quan la pròstata creix ja que exerceix pressió sobre la bufeta i la uretra produint que la orina no surti amb facilitat.

1.1.2 Factors de risc i símptomes

Avui en dia es desconeixen les causes que produeixen el càncer de pròstata tot i que s'han pogut analitzar alguns factors de risc que fan augmentar les possibilitats de patir la malaltia. Un dels factors més importants és **l'edat** del pacient, ja que

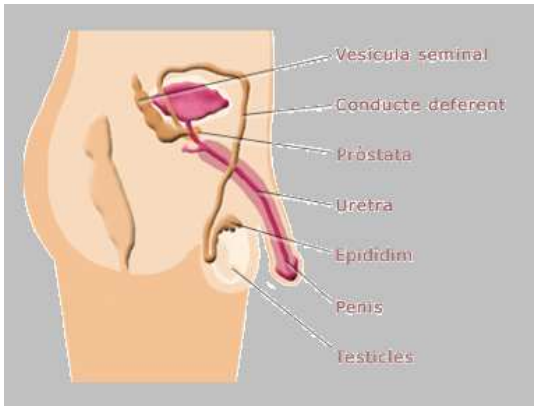

Figura 1.1: Situació de la pròstata

rarament es detecten casos abans dels 40 anys però l'estadística augmenta a partir dels 50 anys i el 80% dels casos es detecta passats els 65 anys d'edat.

Els homes de **raça** negra són més propensos a desenvolupar el càncer i fixant-nos en les **zones geogràfiques** les estadístiques mostren que és més freqüent a l'Amèrica del Nord i Europa Nord-Occidental que a d'altres continents com Àsia, Àfrica o zones com l'Amèrica Central o del Sud. La **dieta** i **l'activitat física** són factors importants en la prevenció. Segons alguns estudis el consum de fruites i verdures afavoreix la prevenció així com disminuir els greixos, tot i que no està comprovat. Com en tot tipus de malalties realitzar activitat física i mantenir un pes correcte disminueix el risc de patir el càncer. S'han relacionat diferents gens que provoquen el càncer de pròstata tot i que no s'ha pogut determinar quins són importants alhora de desenvolupar la malaltia, així doncs no es pot determinar que l'herència genètica tingui un paper essencial tot i que es cert que els familiars de primer grau d'un malalt (fills o germans) tenen el doble de possibilitats tenir càncer.

Per últim citarem que la testosterona (principal hormona masculina) en la fase inicial de la malaltia dispara el creixement de la pròstata tot i que a mesura que la malaltia es troba en una fase avançada aquesta mateixa hormona provoca que s'aturi aquest creixement. No es coneixen els motius que fan que la testosterona es comporti d'aquesta manera i s'està estudiant la possibilitat d'utilitzar-la en la lluita contra el càncer.

Pel que fa els símptomes, són més difícils de detectar en les fases inicials, però els més freqüents són:

- Necessitat d'orinar freqüentment, sobretot a la nit i cada vegada menys quantitat.

- Dificultat d'orinar.
- Interrupció en el fluix d'orina o fluix feble.
- Dolor durant l'erecció i ejaculació.
- Sang a la orina o al semen.
- Dolor en la part inferior de l'abdomen, a la pelvis o a la part superior de les cuixes.

1.1.3 Diagnosi

Existeixen diferents proves mèdiques per tal de detectar i diagnosticar el càncer de pròstata. El tacte rectal, la ressonància magnètica o l'estudi de la concentració de l'antígen prostàtic específic són proves que poden evidenciar la possible presència de càncer a la pròstata tot i que la prova definitiva per determinar el càncer és la biòpsia prostàtica.

Tacte Rectal

Tradicionalment aquesta detecció precoç s'ha fet a través del **tacte rectal** que consisteix en l'introducció per part del metge d'un dit en el recte i comprovar que no hi ha nòduls sospitosos en el volum de la pròstata.

Antigen prostàtic específic

Consiteix en un analisi de sang que determina la concentració de **l'antígen prostàtic específic**, conegut com a PSA. El PSA no es pot considerar un indicador exclusiu de càncer de pròstata, ja que pot ser degut a patologies no necessàriament malignes. Segons el tacte rectal i el nivell de PSA es determinarà si es necessària la biòpsia.

Biòpsia prostàtica

Quan els símptomes o els resultats del PSA facin sospitar la presència d'un càncer de pròstata caldrà fer una **biòpsia** del teixit que sigui sospitós guiada per ultrasons (US). Aquesta serà la prova definitiva en la detecció del càncer ja que indicarà del cert si el teixit extret és maligne o no. Aquest procés és complicat i dolorós ja que per tal que els teixits extrets siguin una mostra fiable de la pròstata s'han de fer diverses puncions directament a la pròstata. Per tal de guiar l'agulla (Figura 1.2) s'utilitza l'ecografia transrectal. També es pot fer una biòpsia amb una intervenció quirúrgica.

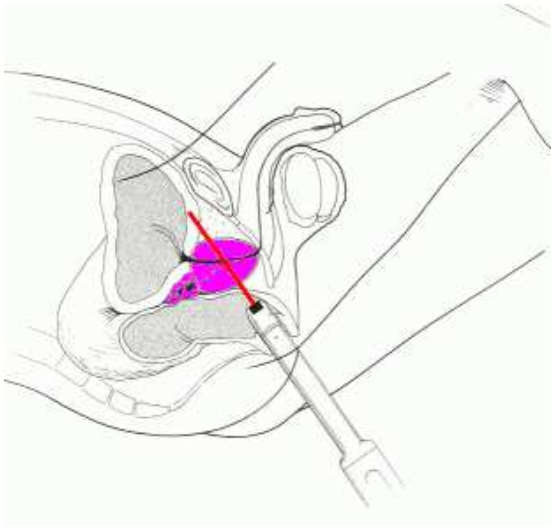

Figura 1.2: Biòpsia de la pròstata

Ecografia transrectal

Consisteix en l'introducció d'una sonda a través del recte la qual emet unes ones d'ultrasò que genera un eco quan xoca contra els diferents teixits. Aquests ecos són captats per la sonda la qual els transmet a un ordinador que genera la imatge de la pròstata. Els tumors de pròstata i el teixit prostàtic generen sons diferents per la qual cosa permet determinar on està situada la pròstata. Aquesta prova es utilitzada en la biòpsia per tal de guiar l'agulla fins a la pròstata.

1.1.4 Etapes el càncer

A continuació es descriuran les diferents etapes de evolució del càncer de pròstata, com sempre el fet de detectar el càncer en les primeres fases ajudarà en la futura recuperació. A partir de la biòpsia (Secció 1.1.3) es confirma l'existència de càncer, en aquest moment s'ha de determinar el grau del tumor (denominat grau de Gleason); a major grau, més elevat és el risc de creixement i extensió tumoral. Existeixen vàries classificacions diferents ABCD, Gleason i TNM.

Classificació ABCD

Etapa A El càncer està localitzat dintre de la pròstata i es diagnostica de forma casual.

Etapa B El càncer encara està localitzat exclusivament a la pròstata però ara ja té la grandària suficient per ser detectat mitjançant per exemple el tacte rectal.

Etapa C El càncer traspassa la càpsula de la pròstata i infecta teixits veïns tot i que encara no ha produït metàstasi.

Etapa D El càncer ha produït metàstasi, en aquesta etapa el càncer ja no es pot curar tot i que existeixen tractaments pal·liatius.

Classificació de Gleason

És una classificació númerica de 1 a 10, simple i precisa en la qual s'etiqueten les cèl·lules microscòpicament. Els graus 1, 2 i 3 són cèl·lules normals, el grau 4 són cèl·lules amb càncer les quals són complicades de diferenciar de les de grau 3 i finalment el grau 5 serà el més gran possible. Resumint, en termes de Gleason es classificarà el càncer entre 2 i 10, sent el 10 el nivell més greu.

Classificació TNM

Aquest mètode poc a poc està substituint al **ABCD** després de la classificació de Gleason per tal de valorar l'extensió del càncer en l'organisme.

(T) Correspon a tumor primari i representa l'extensió del càncer dintre de la pròstata i dels teixits directament veïns.

- (T0) - Sense evidència de càncer.
- (T1)/(T2) - Càncer localitzat a la pròstata.
- (T3) - El càncer s'ha estès als veïns més propers de la pròstata o vesícules seminals.
- (T4) - El càncer s'ha estès a organs veïns com la bufeta.

(N) Significa nòduls limfàtics i representa si el càncer s'ha estès a altres nòduls limfàtics propers.

- (N0) - El càncer no s'ha estès a cap del ganglis limfàtics.
- (N1) - El càncer s'ha estès a un o més ganglis limfàtics propers a la pròstata (zona de la pelvis).

(M) És metàstasi, si el càncer ha arribat a teixits propers com els ossos o els pulmons.

- (M0) - No hi ha metàstasi a distància.
- (M1) - Presencia de metàstasi fóra de la zona de la pelvis, per exemple als ossos, o en organs com el fetge o els pulmons.

1.2 Objectius del projecte

Una de les problemàtiques dintre del diagnosi del càncer de pròstata és el fet de que les diferents proves que es duen a terme no tenen una transformació directe, es a dir, si tenim una ressonància magnètica i volem localitzar un punt a biopsiar amb ultrasò no podem saber amb exactitud que sigui el mateix punt.

Com ja hem dit anteriorment aquest projecte està inclòs dintre del projecte ProSCAN el qual a grans trets busca crear una correspondència entre l'ecografia transrectal (ET) que es du a terme durant la biòpsia i la informació extreta de la ressonància magnètica i les proves dels nivells de PSA prèvies a la biòpsia.

El tenir una eina que genera a temps real aquest tipus de correspondència amb els nivells de PSA podria permetre atacar directament el lloc exacte de la biòpsia en comptes d'haver de fer la biòpsia de zones on realment no fa falta.

Dintre d'aquesta eina visual és important el tractament de la imatge ja que moltes vegades les interferències alhora de realitzar les proves són grans, és aquí on incideix la primera part d'aquest projecte que comença, la reducció i possible eliminació del soroll en proves d'ultrasons (US) per tal que el posterior anàlisi de les imatges sigui més fiable.

Una vegada reduït el soroll el següent pas és localitzar elements dintre de la imatge, en el nostre cas volem situar la pròstata dintre d'una imatge i que el volum de la pròstata quedi segmentat de la imatge.

Així doncs s'afegirà a l'entorn gràfic del projecte ProSCAN dos mètodes de preprocessat d'imatge. Aquests dos mètodes consisteixen en dos filtres aplicats al volum que es visualitza en el programa que rep el nom de Visual, els dos mètodes escollits són el mètode de Gauss, el de la mediana i l'anisotropic diffusion. (Apartat 2).

Una vegada tractada la imatge s'afegirà un mòdul de segmentació basat en el mètode ja implementat en llibreries ITK, Active Shape Model, en apartats posteriors s'explicarà detalladament en que consisteix el mètode. Inicialment s'intentarà obtenir resultats en 2D per després poder aplicar-los a 3D.

1.2.1 Objectius del PFC

Els objectius del PFC són els següents:

- Eliminació del soroll en imatges d'US.
  - Filtre de Gauss.
  - Filtre de la Mediana.
  - Anisotropic diffusion.

- Segmentar imatges d'US i MRI amb el mètode Active Shape Models. Aplicar segmentacions en imatges 2D i 3D.
- Integrar els dos mòduls dintre de l'eina VISUAL.

1.3 Eines utilitzades

El projecte està implementat en C++ amb el programa Visual Studio 2005 C++ sota la plataforma Windows. Com ja s'ha mencionat s'utilitzen llibreries ITK que consisteixen en unes llibreries específiques de C++ orientades al processat d'imatges mèdiques.

1.3.1 ITK

Les ITK *(National Library of Medicine Insight Segmentation and Registration Toolkit (ITK))* són unes llibreries *open-source* implementades en C++ amb una programació orientada a objectes. Estan adaptades per tal de poder utilitzar-se en les diferents plataformes (Unix, Windows, MacOXS). Les ITK estan implementades amb un estil anomenat genèric utilitzant *templates*, es a dir, el mateix codi es pot utilitzar en qualsevol classe. Paral·lelament és un codi en constant evolució ja que molts programadors arreu del món l'utilitzen.

Els objectes ITK estan estructurats de la següent manera:

Els *data-objects* representen les dades les quals es manipulen a través dels *process-objects*. Aquests últims es divideixen en 3 tipus , *source*, *filter objects* i *mappeds* (llegir dades, processar i extreure resultats).

La manera de programar i utilitzar els *data-objects* i els *process-objects* és a través de les *pipelines* les quals reben una entrada i són com un recorregut a través dels mètodes de les classes ITK fins a obtenir una sortida. Per tal d'enllaçar els diferents objectes s'utilitzen dos mètodes SetInput() i GetOutput().

Al capítol 4 d'aquesta memòria trobareu més informació sobre com programar i com les dades es manipulen.

De les diferents versions del software ITK en aquest projecte s'ha utlitzat la versió 3.0.1.

1.3.2 VTK i Qt

Aquestes llibreries no han estat directament utilitzades en aquest projecte però si que formen part del Visual (part visual de projecte ProSCAN), així doncs ha sigut necessari el instal·lar i alhora de inserir el projecte en el Visual si que hi han hagut

algunes petites modificacions així com entendre i poder programar amb aquest tipus de llibreries.

Les VTK són llibreries de visualització, així doncs la utilitat ha estat el poder visualitzar els objectes ITK que s'han processat anteriorment. Igual que les altres, les VTK també són llibreries *open-source* el que facilita tant l'adaptació a les necessitats com la gestió dels problemes.

Les Qt igual que les VTK formen part de la visualització final del codi, així doncs, aquestes Qt han servit per crear la interfície gràfica del Visual i per tant també s'han hagut de fer petites modificacions per tal d'adaptar la part de filtratge i segmentació d'aquest projecte.

La versió de les llibreries VTK compilada pel projecte ha sigut la 5.0. Pel que fa les llibreries Qt s'ha utilitzat la versió 3.3.3

1.3.3 CMake

CMake és un *open-source* per tal de generar i gestionar projectes. En el nostre cas ha permès la interacció de totes les llibreries utilitzades per qualsevol plataforma o compilador, a part que el programa ja et genera tots els arxius necessaris per tal de implementar i compilar correctament el codi.

El primer pas abans de crear qualsevol projecte és instal·lar tots els paquets de llibreries i compilar-los amb el CMake. Una vegada fet això ja es pot iniciar un projecte que utilitzi operacions de qualsevol de les tres llibreries i l'única cosa que s'ha de fer és crear un arxiu CMakeLists.txt on quedarà definida la ubicació de les llibreries que s'utilitzaran al projecte que s'inicia. Una vegada creat aquest CMakeLists només cal generar els arxius i començar a programar.

La versió utilitzada en aquest projecte ha sigut la 2.4.3, tot i que al principi es van provar altres versions ja que hi havia problemes alhora de compilar les classes ITK i posteriorment les VTK.

1.3.4 Altres eines

Visualitzar imatges

Altres eines utilitzades han sigut per exemple visualitzadors d'imatges. El format de les imatges utilitzat al llarg del projecte ha sigut bàsicament DICOM quan es tractava de MRI. Les imatges dicom corresponen a volums 3D que estan formats per talls (*slice*) d'imatges en 2D. Així doncs, aquestes imatges s'han de poder visualitzar generant el volum quan ens interessa la vista 3D o per separat quan volem extraure informació d'un sol tall. Un dels visualitzadors utilitzats ha sigut SViewer que permet explorar tall a tall la imatge.

Matlab

El Matlab és un eina que permet fer processat d'imatges així com manipular-les i realitzar multitud d'operacions. Aquesta eina ha estat utilitzada en primera instància per analitzar resultats de segmentació en les primeres fases ja que de forma ràpida i simple permet sobreposar imatges i vectors que com es veurà més endavant és necessari alhora de visualitzar resultats. També és molt útil alhora de treballar amb un tipus d'imatges que ja s'han comentat prèviament, les dicom.

Capítol 2

Pre-processat de la imatge

Com hem introduït prèviament per un bon processat de les imatges és important la seva qualitat. Aquest és l'aspecte atacat en aquesta primera part del projecte, el preparar la imatge per poder utilitzar-la en bones condicions. Les classes ITK ofereixen molts tipus de filtratge entre els quals s'han escollit 3 per implementar i després afegir el mòdul al Visual (entorn gràfic del ProSCAN).

2.1 Processat de la imatge

Dur a terme el processat d'una imatge consisteix en aplicar tècniques per tal de millorar-ne la qualitat, remarcar algun aspecte concret o poder analitzar les seves característiques que facilitaran la cerca d'un objecte dintre de la imatge. Les aplicacions directes quan és fa el processat de la imatge poden ser:

- Detectar objectes dintre de la imatge
- Inspecció visual automàtica, per exemple, si a la imatge hi ha un foc activar alarma incendis.
- Mesura de característiques geomètriques o objectes de color.
- Classificació d'objecte (cadena de fabricació d'una fàbrica).
- Restauració d'imatges.
- Mesura de la qualitat de les imatges.

En aquest apartat ens centrem en els filtres, els quals s'utilitzen per modificar les imatges ja sigui per detectar contorns, per modificar l'aspecte de la imatge o per eliminar el soroll que hi pugui existir. Aplicar un filtre vol dir produir una

convolució entre un píxel i els seus veïns respecte a una funció. El terme convolució és el que engloba les operacions matemàtiques que es generen quan s'aplica el filtre (funció) a la imatge. Aquests filtres es poden representar com matrius de dimensió NxM, sempre molt més petites que la matriu de la imatge a processar. La matriu de convolució es desplaça per tota la imatge calculant el nou valor de cada píxel.A la figura 2.1 podem veure una matriu de convolució.

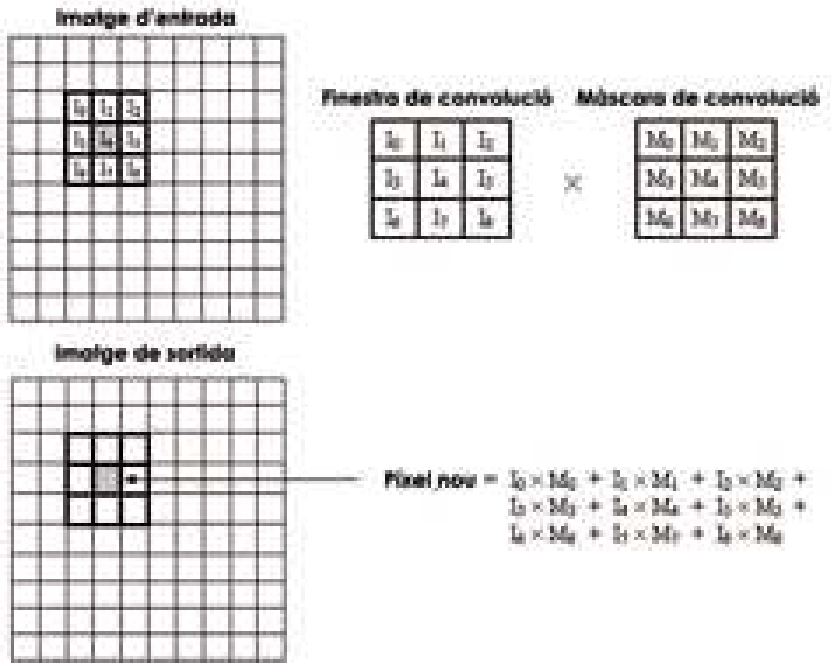

Figura 2.1: Convolució d'una imatge

Durant la convolució s'ha de decidir com tractar els píxels del voltant de la imatge ja que si el píxel central es col·loca al (0,0) mig filtre queda fóra. Veurem ràpidament 4 maneres d'actuar:

- Considerar els píxels fóra de la imatge com a 0, el problema està quan els límits de la imatge també són importants, és a dir, que no hi ha un objecte al mig el qual és l'objectiu sinó que és tota la imatge important.
- Començar la convolució on tot el filtre cau dintre de la imatge, llavors els límits es copien igual que la imatge original.
- Duplicar tot el límit de la imatge així el filtre serà aplicable en tots els píxels.
- El límit de l'esquerra utilitza com a veïns els píxels del límit de la dreta i viceversa, així com els píxels superiors utilitzen els inferiors.

Existeixen diferents tipus de filtres, primer de tot farem una distinció depenent de si son espaials, les operacions es fan directament sobre la imatge segons els píxels veïns, o si són freqüencials, en els quals les operacions sobre la transformada de Fourier de la imatge, per tant prèviament s'ha calculat la transformada de la imatge i després d'aplicar el filtre tornant a fer la transformada inversa per tal de recuperar la imatge.

Fixant-nos en els filtres espaials, distingim dos tipus, els passa baixes (ex. Gaussià, mediana, mitjana) i els passa altes (ex. Laplacià), terme que fa referència a si filtren les freqüències altes o la gama de freqüències baixes. El primer serveix per reduir el soroll que puguin tenir les imatges però com a inconvenient retorna una imatge més borrosa i difusa, en canvi, el passa altes augmenta el contrast i en conseqüència el que provoca és que els contorns quedin més accentuats. Els filtres ja implementats en les classes ITK triats per formar part del projecte han estat el filtre Gaussià, el filtre de la mediana i el anisotropic diffusion.

Aquests filtres tenen com a objectiu eliminar el soroll de les imatges. Els dos primers són filtres passa baixes, amb els inconvenients que això comporta que es generen contorns difosos i pèrdua de detall de formes, en canvi l'anisotropic diffusion alhora que redueix el soroll conserva els contorns, veiem cada filtre més detalladament.

2.2 Filtre Gaussià

El filtre Gaussià, ja esmentat de tipus espaial s'aplica directament a la imatge. Fent una petita introducció comentar que les funcions de Gauss aplicades al tractament de senyals (so, imatges, senyals biològiques, radars, etc.), són les que permeten definir els filtres Gaussians. Aquestes mateixes funcions són les que en temes d'estadística descriuen les distribucions normals, però ens centrem en la vessant del processat d'imatges.

El filtre rep el nom de *Gaussian blur* ja que descriu el procés en que la imatge es difumina en aplicar-li una funció Gaussiana per tal de reduir el soroll que hi pugui haver. El concepte soroll dintre d'una imatge és el fet que un píxel tingui un valor que realment no és el que li correspon, per tant, l'objectiu d'aplicar alguna d'aquestes funcions és estimar valor original del píxel. El soroll pot ser causa de diversos factors, un d'ells és l'equip d'adquisició de la senyal utilitzat, aquest pot estar que mal calibrat ó també es pot produir soroll durant la transmissió la imatge, possibles interferències o pèrdues de bits durant aquesta.

El soroll es pot classificar segons si és Gaussià o impulsiu:

Soroll Gaussià: Es caracteritza per tenir un espectre d'energia constant per totes les freqüències. Quan tenim aquest tipus de soroll i capturem el mateix píxel diferents vegades sempre té valors diferents.

Soroll impulsiu: També rep el nom de sal i pebre (Salt & Pepper) Es caracteritza per l'aparició de píxels amb valors arbitraris molt diferents a la resta de píxels veïns, per tant aquest tipus és fàcilment detectable.

El soroll Gaussià té un efecte general en tota la imatge, és a dir, la intensitat de cada píxel es veu afectada respecte a la intensitat de la imatge original, en canvi, el soroll impulsiu està més centrat en punts concrets de la imatge que prendran valors de 0 a 255 però sense cap tipus de lògica.

Ja s'ha explicat a la introducció la manera d'aplicar un filtre, la diferència entre el filtre Gaussià i per exemple altres filtres existents com el de la mitjana, mediana, etc. és que el Gaussià utilitza un a màscara especial segons la funció de gauss. La manera amb que és calcula la màscara (kernel) és la següent, la distribució de Gauss en 2D té aquesta forma:

$$G(x,y) = \frac{1}{2\Pi\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.1}$$

on la σ és la desviació estàndard de la distribució i genera la gràfica de la figura2.2.

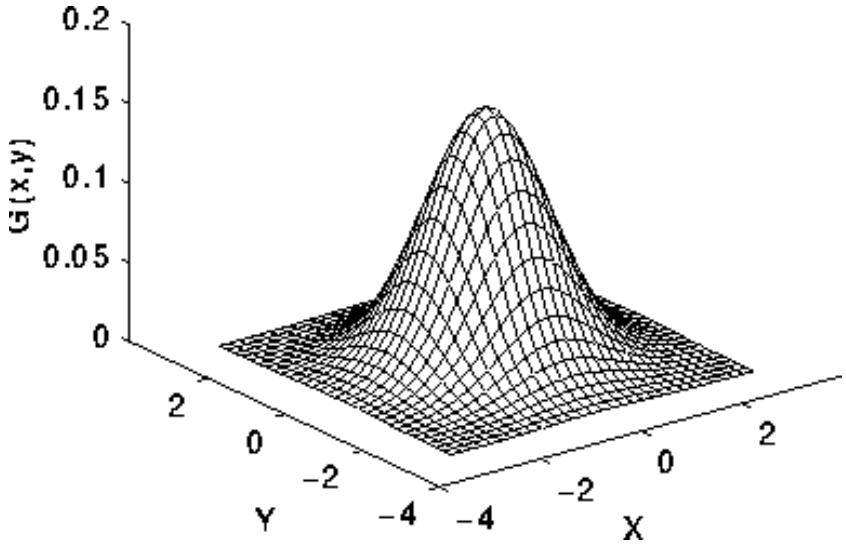

Figura 2.2: Distribució Gaussiana 2D

Els valors que s'extreuen de la distribució són els que s'utilitzen per crear el kernel. Normalment alhora d'aplicar aquest filtre no s'utilitza la distribució 2D sinó que es separa i tractar la imatge com dos independents d'una sola dimensió.

2.2.1 Pseudocodi

```
FUNCIÓ GAUSS

    var
        sigma tipus doble;
        volumEntrada, volumSortida tipus imatge3D;
        filtreGaussX  tipus  filtre;
    fvar

    filtreGaussX = crearNouFiltre();

    volumEntrada = llegir_volum();
    sigma = capturar_valor();

    filtreGaussX.dimensions(0);
    filtreGaussX.ordre(zero);
    filtreGaussX.normalitzarescala(fals);
    filtreGaussX.conf_imatge(volumEntrada);
    filtreGaussX.parametres(sigma);
    kernel = filtreGaussX.calcularMàscara(); //segons el valor de sigma i la fòrmula
    que hem vist prèviament.

    per píxels=0 fins que píxels = totalPíxels fer
        volumSortida = filtreGaussX.convolucionar(kernel,píxel);
    fi per

    retornar volumSortida;


FI FUNCIÓ GAUSS
```

Figura 2.3: Pseudocodi Filtre Gaussià

En el cas de filtre Gaussià per estructura de les ITK s'ha d'especificar la direcció en la qual s'aplicarà el filtre. En el pseudocodi s'ha deixat especificat només com es configura una direcció (la direcció x) però l'estructura mostrada s'ha de fer també per la direcció Y i per la Z. La manera amb que després es calcula la imatge final de sortida és crear una pipeline entre els tres filtres, és a dir, el filtre X esdevé entrada del filtre Y i aquesta últim serà entrada del filtre Z que serà el últim i per tant la seva sortida serà la sortida final del mètode. (filtreX->filtreY->filtreZ->imatgeSortida). En la figura 2.3 podeu veure el pseudocodi.

2.3 Filtre de la mediana

Igual que el filtre citat en l'apartat anterior, el filtre de la mediana correspon a un filtre espaial. El filtre de la mediana al igual que el Gaussià redueix el soroll tot i que preserva més la informació dels contorns. Aquest filtre que estem tractant ara és molt adequat per eliminar el soroll de tipus impulsiu pel tipus de funcionament de la mediana.

Fent un recordatori la mediana consisteix en el valor el qual té el mateix nombre de valors a esquerra que a dreta, és a dir, ordenar tots els valors de més petit a més gran o viceversa i agafar el que està situat al mig de tots. Veient quin és el procés a seguir queda clar perquè elimina de manera eficaç el soroll impulsiu, ja que el que fa és fer que tots els píxels tinguin un valor semblant als píxels veïns, per exemple, si tinguéssim una sèrie de píxels amb valors entre 0 i 20 i tinguéssim un sol píxel amb valor 50 aquest últim quedaria al final de l'ordenació i per tant no tornaria a ser escollit com el valor final del nou píxel.

El filtre de la mediana igual que el filtre gaussià ha de tenir un tractament especial pels píxels del voltant de la imatge segons la matriu de convolució també anomenat finestra ó màscara. Diferents formes de finestra, són el que determinaran quins píxels seran utilitzats per realitzar la mediana, així doncs, els models típics de màscares són el quadrat NxN, en X o en creu. Alhora d'escollir quin tipus de màscara s'utilitza és convenient provar per visualitzar resultats i així fer una tria adient.

$$\begin{pmatrix} x & x & x \\ x & x & x \\ x & x & x \end{pmatrix}$$

La matriu quadrada és la que millor elimina el soroll de la imatge, però a la vegada és la que genera també una distorsió més gran al resultat final. Normalment els filtres de la mediana utilitzen aquest tipus de finestra.

$$\begin{pmatrix} & x & \\ x & x & x \\ & x & \end{pmatrix}$$

La matriu amb forma de creu és eficient quan la imatge conté moltes cantonades, molts contorns

$$\begin{pmatrix} x & & x \\ & x & \\ x & & x \end{pmatrix}$$

Per últim la matriu en forma X és un terme mig entre les dues anteriors.

Un dels inconvenients d'aquest mètode és el cost de rendiment que té al haver d'ordenar les dades per tal de fer el càlcul, ja que, els mètodes per ordenar valors són relativament lents.

2.3.1 Pseudocodi

```
FUNCIÓ MEDIANA

    var
        vectorIndex tipus SizeExtenció;
        vecorMediana tipus vector;
        imatgeEntrada, imatgeSortida tipus volum3D;
        mediana, numPixelsImatge tipus enter;
        i tipus iteradorImatge;
    fvar

    vectorIndex[1] = 1; //per x
    vectorIndex[2] = 1; //per y
    vectorIndex[3] = 1; //per z

    imatgeEntrada =  llegir_imatge();
    numPixelsImatge = obtenirNumPixels( imatgeEntrada );
    i = inici;

    mentre i < numPixelsImatge fer

        vectorMediana = obtenirPixelsVeins( i );
        mediana = calcularMediana( vectorMediana );
        imatgeSortida( i ) = mediana;
        i++;
    fi mentre

    retornar imatgeSortida;

FI FUNCIÓ MEDIANA
```

Figura 2.4: Pseudocodi Filtre Mediana

El filtre de la mediana el procés és més simple, la convolució es fa a través d'una màscara de la qual podem escollir les dimensions al iniciar el mètode. Després de definir la màscara simplement el mètode la convoluciona amb tots els píxels de la imatge per obtenir la imatge de sortida. La figura 2.4 mostra aquest algoritme.

2.4 Anisotropic Diffusion

L'Anisotropic Diffusion és el darrer filtre implementat en el PFC, va ser ideat per Perona and Malik [2] amb l'objectiu de complementar els filtres lineals. Els filtres anisotropics també reben el nom de *no uniformes* o *variable conductance diffusion filters*. La diferència entre aquest i per exemple el Gaussià resideix en que l'anisotropic respecta els contorns alhora que redueix el soroll de la imatge.

El fet de suavitzar una imatge (ex. amb una Gaussiana) és pot implementar de manera iterativa on a un instant t de la imatge I_t és el resultat d'aplicar t gaussianes a la imatge original. Aquesta representació es pot també expressar amb la següent equació(equació 2.2), l'equació de difusió.

La idea és expressar la intensitat de la imatge com a una distribució de temperatura. En l'instant t l'equació decideix la transferència o difussió d'aquella temperatura en la imatge.

$$I_t(x, y, t) = \nabla(c(x, y, t)\nabla I(x, y, t)) \tag{2.2}$$

si c(x,y,t) és constant la difusió serà de tipus isotròpic, és a dir es comporta similar a una Gaussiana però calculant a través d'un procés iteratiu en comptes de amb una simple convolució. En el cas de l'anisotropic diffusion es pretén suavitzar de diferent manera la imatge per tal de preservar millor els contorns suavitzant més les regions homogènies. Així aquesta c(x,y,t), la conductància, serà variable, propera a 0 no hi haurà difusió (no suavitzarà). En el cas d'un píxels del contorn, si c = 1 la difusió serà màxima, per exemple en una regió homogènia. Aquest criteri vindrà donat per la funció g i es calcularà a partir del gradient de la imatge. (eq. 2.3).

$$c(x, y, t) = g(|\nabla I(x, y, t)|) \tag{2.3}$$

Existeixen diferents equacions per dur a terme aquest càlcul, un exemple són:

Leclerc:

$$g(|\nabla I|) = e^{-\frac{|\nabla I|^2}{k^2}} \tag{2.4}$$

Lorentz:

$$g(|\nabla I|) = \frac{1}{1 + \frac{|\nabla I|^2}{k^2}} \tag{2.5}$$

La diferència que trobarem entre utilitzar una o l'altre serà que l'equació Lorentz afavoreix més mantenir els contorns mentre que Lorentz afavoreix el filtratge de regions.

En ITK aquesta conductància es representa amb una k (també es coneix amb el nom de kappa) i és configurable per l'usuari abans d'iniciar el procés de filtratge ja que finalment és la que controla la sensibilitat del mètode a preservar els contorns.

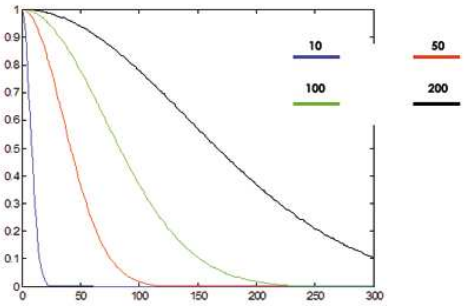

Figura 2.5: Gràfica de de la funció Leclerc

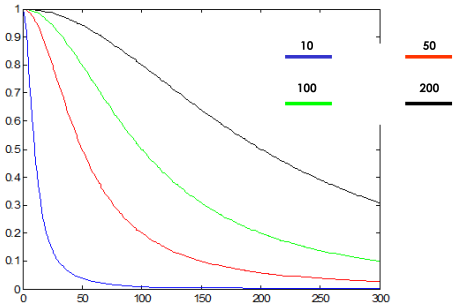

Figura 2.6: Gràfica de de la funció Lorentz

El temps o delta (t) és l'altre paràmetre del filtre i també s'ha de configurar prèviament. Correspon a l'increment de temps que triga a fer cada pas. Aquests dos paràmetres equivalen a la σ quan utilitzem un filtre Gaussià. Per últim, ja hem dit que el mètode no era una convolució lineal sinó un mètode iteratiu, llavors es necessari especificar el número d'iteracions que desitgem que faci. En imatges 2D, generalment, per obtenir bons resultats aquest paràmetre no ha de ser gaire gran. A capítol de resultats trobareu tota la informació sobre configuracions.

L'anisotropic Diffusion genera a la seva sortida un o diversos talls d'imatge (slices) amb la corresponent reducció del soroll i de textura però amb la conservació dels contorns.

2.4.1 Pseudocodi Anisotropic Diffusion

A la figura 2.7 veiem el pseudocodi de l'Anisotropic Diffusion on segons els paràmetres introduïts es duen a terme els càlculs tal i com s'ha explicat en aquest apartat per tal que la imatge redueixi el soroll però preservant els contorns. El procés és el següent, es crea una copia de la imatge d'entrada afegint dues columnes i dues files al voltant d'aquesta amb valor de píxel igual a 0. Llavors es dur a terme un rastreig de la imatge fent la diferencia entre la original i la nova de tal manera que produeix el

```
FUNCIÓ ANISOTROPIC DIFFUSION
    var
        imatgeEntrada, imatgeSortida tipus volum3D;
        numIteracions,i tipus enter;
        delta, k tipus doble;
    fvar

    imatgeEntrada = llegir_imatge();
    numIteracions = llegir();
    temps = llegir();
    k = llegir();

    //imatgeAux amb contorn de 0's
    per it=0 fins que it = numIteracions fer
        per cada direcció
            delta(direcció) = calcularDif(imaAux, imatgeEntrada,direcció);
            si (leclerc) llavors
                c(direcció)=aplicarLeclerec(delta,k)
            else if (lorentz) llavors
                c(direcció)=aplicarLorentz(delta,k)
            fi si
        fi per
        imatgeSortida = imatgeEntrada + temps*(c*delta(nord)+ c*delta(sud)+
                        c*delta(est)+c*delta(oest));
    fi per

    retornar imatgeSortida;

FI FUNCIÓ ANISOTROPIC DIFFUSION
```

Figura 2.7: Pseudocodi Filtre Anisotropic Diffusion

gradient per cada direcció (nord, sud, est i oest) després es calcula amb les fòrmules ja vistes (Leclerc i Lorentz equacions 2.4 i 2.5) obtenint una nova imatge suavitzada.

2.5 Altres filtres ITK

Segons les necessitats es pot considerar la opció d'implementar algun altre tipus de filtre. ITK ofereix un gran ventall de filtres específics segons l'objectiu que es persegueixi, els quals poden actuar sobre una sola imatge d'entrada o sobre un volum 3D (conjunt de talls o slice). Els filtres ITK degut a l'estructura de programació dels objectes la sortida d'un filtre esdevé entrada pel següent element del pipeline ja sigui un connector per tal de visualitzar o qualsevol altre tractament que se li pot fer a la imatge.

- Thresholding
- Detecció de contorns
- Casting i mapeig
- Mètodes basats en gradients
- Laplacians
- Mètodes amb veïns
- Mètodes basats en gradients

2.6 Exemples d'aplicació de filtres

Exemples Gauss

La eina per visualitzar resultats sobre els filtres ha sigut el Matlab, ja que fàcilment permet filtrar les imatges i generar la sortida. Així doncs s'han provat filtres Gaussians amb els dos tipus de soroll esmentats. La introducció de soroll a les imatges es fa a través de l'instrucció *imnoise*, en el nostre cas amb paràmetres 'gaussian' o 'salt & pepper' especificant també quin nivell de soroll volíem introduir. Per tal de filtrar les imatges, el Gauss és calcula a través de la instrucció *fspecial* que té com a paràmetres la dimensió de la màscara a aplicar i la sigma (σ), la instrucció seria de l'estil *fspecial('gaussian',3,0.5)* (dimensió 3x3 sigma=0.5).

Figura 2.8: Phantom generat amb Matlab per fer les proves de filtratge

Tenim una imatge original com la de la figura 2.8. El primer que hem fet ha estat introduir soroll del tipus gaussià, figura 2.9 i en la següent imatge (Figura 2.10) es

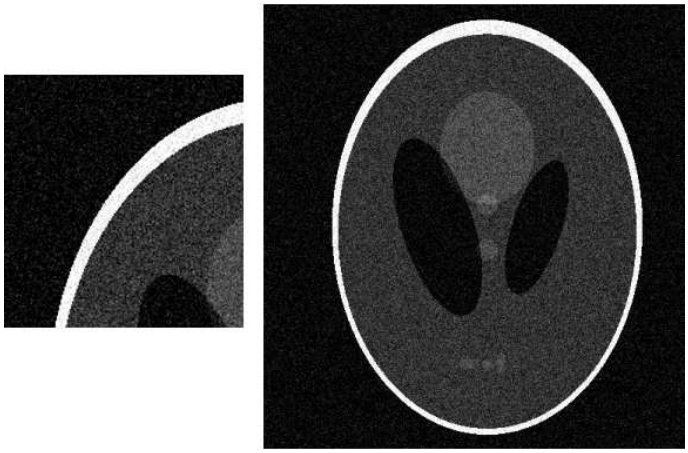

Figura 2.9: Imatge phantom amb soroll del tipus Gaussià

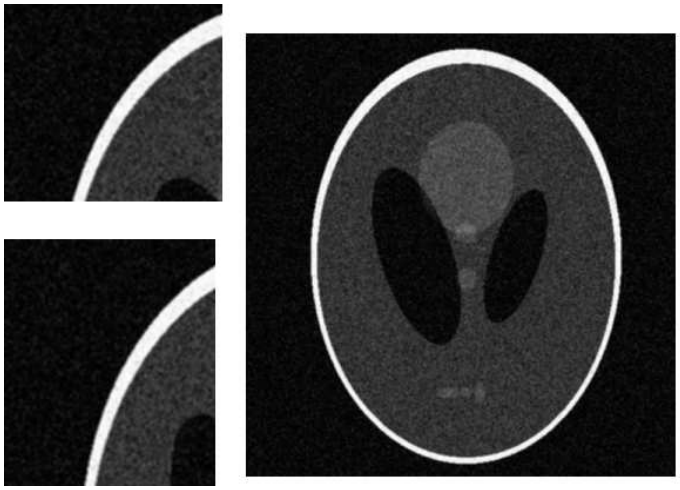

Figura 2.10: Imatge phantom amb soroll Gaussià filtrada

mostra el resultat de filtrar amb dos valors de sigma, la finestra superior esquerra σ = 1.0 i la inferior esquerra σ = 6.0 igual que la imatge gran. En aquestes imatges es pot apreciar que els contorns estan més difuminats.

Ja sabem que existeix un altre tipus de soroll el qual exemplifiquem amb una imatge (Figura 2.11) i el seu filtrat a la figura 2.12 tot i que com ja veurem la mediana elimina millor el soroll sal i pebre. Es pot apreciar que a la vegada que creix el valor de sigma l'eliminació del soroll és més gran però també els contorns són més difosos.

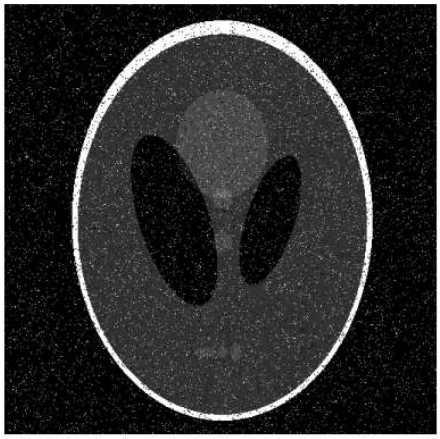

Figura 2.11: Imatge phantom amb soroll tipus Salt and Pepper

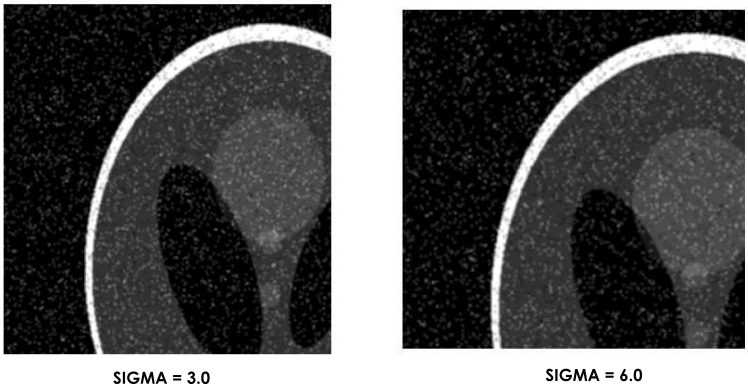

Figura 2.12: Imatge phantom amb soroll tipus Salt and Pepper filtrada amb un filtre Gaussià

Exemples mediana

Igual que en el cas de Gauss s'ha utilitzat l'aplicació Matlab per tal de testejar el filtre de la mediana. L'instrucció que ens permet aplicar aquest filtre és la *medfilt2(imatge)*. Utilitzarem aquest filtre per veure com redueix el soroll tipus salt i pebre amb la mateixa imatge d'entrada, figura 2.11. Així doncs amb la imatge d'entrada de la figura 2.11 tenim que la sortida és sense cap tipus de soroll, l'ha eliminat completament com mostra la figura 2.13. El següent pas és introduir més soroll ja que el Matlab permet amb una variable escollir el nivell de soroll introduït.

Es pot apreciar que a mesura que s'introdueix més soroll resten alguns píxels els quals no pot reduir-lo del tot i també ja no són tan exactes els contorns(figura 2.14 que conté un nivell de soroll de 0.07 tal i com genera el Matlab).

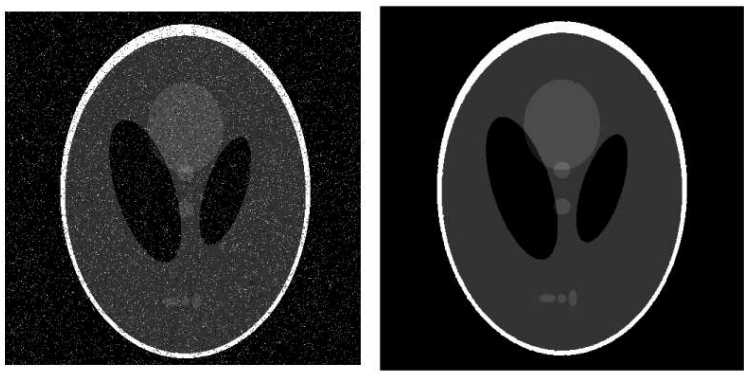

Figura 2.13: Imatge phantom amb soroll tipus Salt and Pepper (0.07) i resultat de filtre mediana

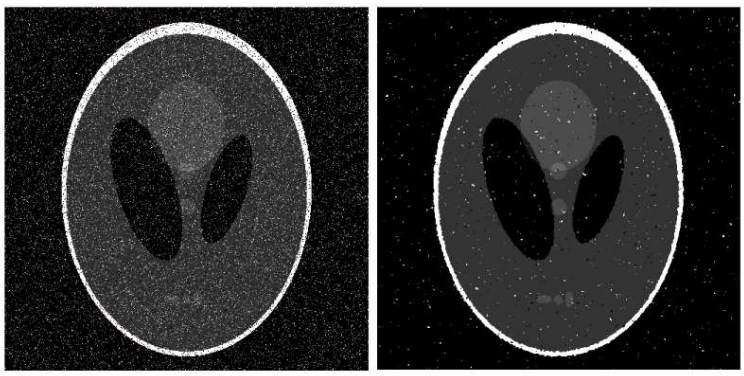

Figura 2.14: Imatge phantom amb soroll tipus Salt and Pepper (0.15) i resultat de filtre mediana

Exemples anisotropic diffusion

L'anisotropic diffusion s'ha provat amb un codi Matlab que executa el pseudocodi de la figura 2.7, el qual pot escollir l'equació i tots els paràmetres d'entrada. Veiem algun exemple, establint uns paràmetres fixes, el número d'iteracions utilitzat en els exemples és 80, el que variarà serà la conductància i el temps. A la figura **??** veiem la diferència quan disminuim el temps, a l'esquerra $\lambda = 0.20s$ mentre que la figura de la dreta $\lambda = 0.05s$, la figura en la qual el temps és menor, és molt més clara i definida mentre que en augmentar el temps es perd contrast.

Pel que fa la conductància, a la figura 2.16 s'aprecia (sobretot en el contorn exterior) que es va difuminant més.

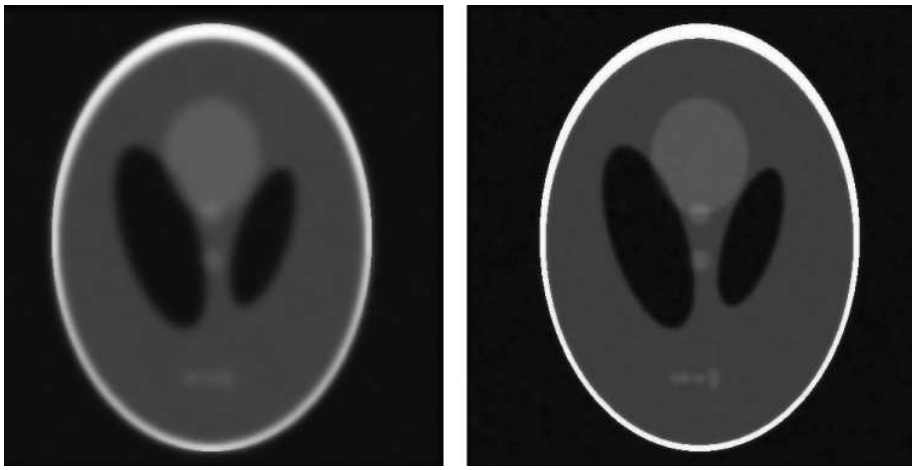

Figura 2.15: Filtratge Anisotropic Diffusion variació de temps (0.20 - 0.05)

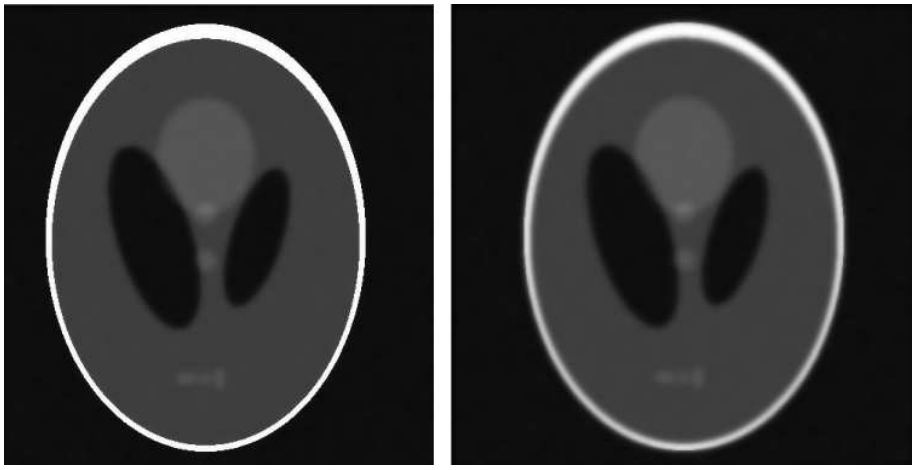

Figura 2.16: Filtratge Anisotropic Diffusion variació de temps (20 - 80)

Per acabar ja amb el tema del filtratge es mostra a la figura 2.17 la diferència que existeix quan s'utilitzen les diferents equacions esmentades en l'apartat de l'Anisotropic Diffusion. Els paràmetres utilitzats per dur a terme aquest exemple han estat:

- Número d'iteracions = 80.
- Conductància (k) = 60.
- Temps (λ) = 0.10.

Els resultats són que tal i com s'havia comentat Leclerc preserva contorns més que Lorentz.

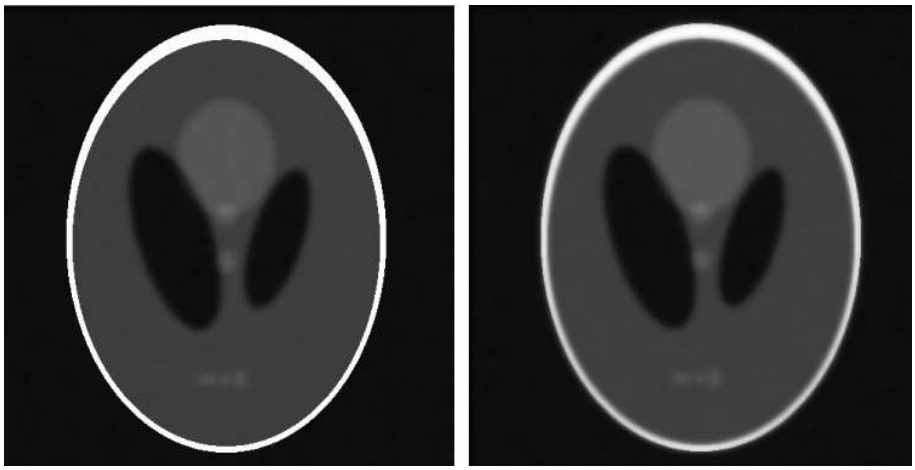

Figura 2.17: Resultats Anisotropic Diffusion segons equació Leclerc (esquerra) - Lorentz (dreta)

2.7 Sumari

En aquest capítol s'han introduït els filtres que estan dintre de la part de pre-processat del PFC. S'han analitzat individualment, donant pautes d'utilització i característiques principals. En l'apartat de resultats s'analitzaran en detall els paràmetres que els conformen per la correcta aplicació a les imatges de pròstata.

Capítol 3

Segmentació de la pròstata

Dintre de processat d'imatges la segmentació és una part important alhora que complicada. Consisteix en extreure informació concreta de la imatge, dividint-la en regions i objectes.

La necessitat d'automatitzar el procés sorgeix per facilitar la classificació imatges segons uns paràmetres concrets. L'ull humà no és capaç de captar tots els detalls existents en una imatge i menys si són volums que contenen 150 imatges, aquí és on la segmentació esdevé important.

Com ja s'ha esmentat abans, previ a qualsevol processament d'imatges cal seguir uns passos, així doncs enumerem els passos bàsics de la segmentació:

1. Eliminació del soroll (Capítol 2).
2. Segmentació.
3. Extracció de regions i objectes.
4. Extracció de característiques.
5. Classificació de la imatge o regió.

La segmentació pot ser generalista o específica, generalista vol dir que no es busca un objecte concret sinó que es segmenten totes les regions que es trobin a la imatge, en canvi si és específica si que hi ha un objecte dintre de la imatge i per tant només s'extraurà la zona desitjada deixant la resta (el fons) com tota una regió. Aquesta feina d'extracció, no és tan fàcil com sembla a priori, el fet de traslladar aquestes necessitats a una aplicació informàtica requereix unes certes condicions inicials si volem que les segmentacions siguin òptimes. Aquests requeriments són per exemple que les regions de la imatge siguin uniformes i homogènies respecte algunes característiques, nivell de gris, color o textura, les regions adjacents han de tenir valors diferents per tal de poder-les diferenciar o que estiguin separades per contorns ben definits.

3.1 Mètodes existents

Una vegada hem definit clarament en que consisteix segmentar veiem quines tècniques existeixen i perquè s'ha escollit el testejar l'Active Shape Model.

Existeixen molts mètodes enfocats a la segmentació d'imatges:

Mètodes basats en el creixement de regions Conegut com a *region growing*, consisteix en situar llavors dintre de la imatge i fer créixer aquestes llavors segons la informació dels píxels veïns, si són semblants en característiques llavors creix en aquella direcció.

Mètodes basats en *clustering* o en histograma El clustering permet agrupar píxels amb característiques semblants (color, textura, nivell de gris, etc.) i l'histograma agrupa píxels amb característiques de color semblants, és a dir, l'histograma seria un tipus concret del mètodes de clustering

Mètodes basats en contorns Aquest tipus de segmentació s'utilitza coma base d'altres. Els contorns generen molta informació respecte als diferents objectes que es poden trobar a la imatge, si es detecten aquestes fronteres entre regions de manera senzilla es podran obtenir bons resultats de segmentació.

Altres mètodes Gran quantitat de variants respecte aquests mètodes o d'altres que introdueixen nous paràmetres a tenir en compte com per exemple:

- Mètodes *Level Set*.
- Mètodes *Watershed*.
- Mètodes basats en models (ex. Active Shape Models).
- Segmentació semi-automàtica.
- Segmentació amb xarxes neuronals.

Descrits els principals mètodes de segmentació només queda introduir-nos a fons en el mètode escollit i implementat en ITK per tal d'obtenir resultats de segmentació de proves de pròstata.

3.2 Eines

En aquest apartat presentarem una eina específica per fer segmentacions d'imatges. Es tracta el ITK-SNAP, en aquest apartat ha estat utilitzada per tal de generar les segmentacions que posteriorment seran utilitzades per fer l'entrenament dels models.

L'ITK-SNAP és una utilitat que segmenta estructures en imatges mèdiques en 3D. Aquesta segmentació es pot dur a terme de dos maneres diferents:

- Manual: A través dels slice de la imatge es marca amb punts el contorn del que volem segmentar fins a obtenir tot el volum segmentat i el podem visualitzar separat de la resta de la imatge. Fins i tot SNAP genera un volum 3D de la forma extreta per tal de fer-se una idea del resultat.

- Semiautomàtica: A través de 3 passos senzills com és, delimitar la zona en la qual es troba la part que volem segmentar. Després binaritzar la imatge per tal de remarcar contorns i escollir uns punts de inici de l'algorisme que fa avançar la segmentació. La zona retallada creix segons el nivell d'intensitat dels píxels veïns o el contorn marcat de la zona.

Finalment s'ha de guardar els resultats obtinguts, existeixen diferents opcions però la que s'ha escollit ha sigut el format VTK ja que després s'ha pogut obrir fàcilment des de ITK o com en el nostre cas que ens interessaven talls específics de la imatge guardar en format DICOM i organitzar les segmentacions segons els nostres interessos. En la web de l'ITK-SNAP trobareu tota la informació.([3])

Figura 3.1: Interfície de l'ITK-SNAP

3.3 Mètode Active Shape Models (ASM)

Com ja sabem, l'objectiu és localitzar la pròstata i ho estudiarem a partir de la implementació del mètode Active Shape Model (ASM) formulat per Tim Cootes([4] "The Use of Active Shape Models for Locating Structures in Medical Images." July 1994).

3.3.1 Introducció

La idea principal resideix en el fet que tot i saber quin és l'objecte que es busca dintre de la imatge i la forma que té, aquest pot aparèixer amb alguna deformació,

no tots els objectes tenen una forma rígida. En imatges mèdiques que són les que nosaltres manipulem els òrgans que hi apareixen poden variar amb el temps i també depenent de cada pacient. Així doncs es defineixen uns punts que donen la forma a l'objecte els quals accepten una petita variació segons cap on evoluciona la figura.

3.3.2 Etapes de creació de models

Diferenciem dues parts que són les que després tractarem al codi d'ITK, la primera és l'entrenament (training) per construir el model i la segona la cerca (search) del model en una nova imatge.

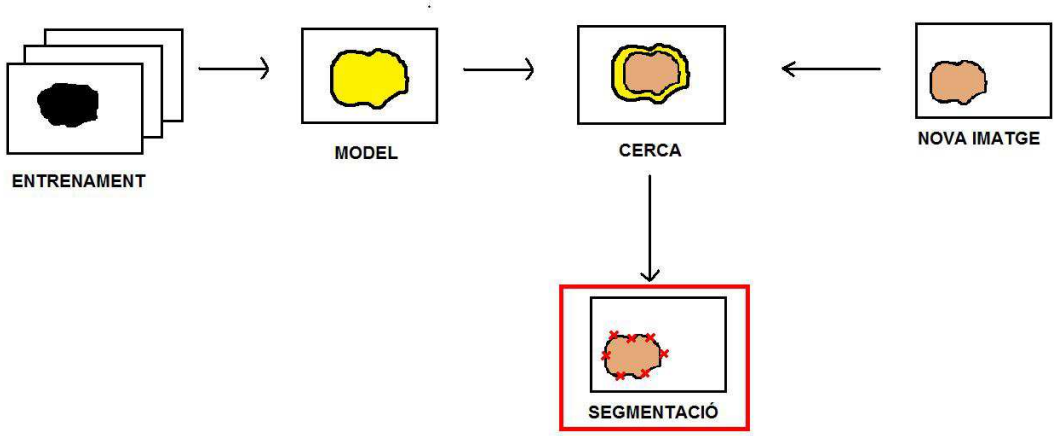

Figura 3.2: Esquema de l'Active Shape Models

Training

La forma de l'objecte es pot definir segons punts que delimiten el contorn d'allò que volem segmentar, aquests punts han d'acceptar una variabilitat, és a dir, han de "moure's"per tal d'adaptar-se a la forma de la figura. Aquesta forma base que posteriorment s'utilitzarà per fer les variacions rep el nom de **forma mitjana** o ***mean shape***. Les variacions del mean shape venen donades per la variació dels punts de les imatges d'entrenament (expressades per la matriu de covariàncies, vegeu més endavant).

En altres termes, el training calcula un model estadístic sobre com varia la forma a través d'unes imatges de mostra. Tot seguit s'analitza com es du a terme el càlcul d'aquest model.

La forma que trobem de l'objecte a les imatges d'entrenament es delimita a través de *n landmarks* distribuïts al llarg del contorn de l'objecte. Les variacions en les

coordenades d'aquests punts característics seran les que marcaran les variacions de la forma de l'objecte. Tenint aquests punts podem definir la forma de l'objecte com un vector.

$$x = (x1, y1, x2, y2, ..., xn, yn) \tag{3.1}$$

El següent pas, l'avaluació i organització dels punt obtinguts a l'entrenament es fa aplicant l'anàlisi de components principals (*Principal Component Analysis PCA*) que calcula els eixos principals en els quals estan les dades distribuïdes contant que estem en un sistema sense dimensions (nd-D space). Els passos del PCA són:

Calcular la mitjana on *s* són sèries de punts x_i

$$\bar{x} = \frac{1}{s}\sum_{i=0}^{s} x_i \tag{3.2}$$

Calcular la matriu de covariança

$$S = \frac{1}{s-1}\sum_{i=0}^{s}(x_i - \bar{x})(x_i - \bar{x}^T) \tag{3.3}$$

Calcular vectors propis ϕ_i i valors propis λ_i

Una vegada tenint aquests paràmetres contem que Φ conté els t vectors propis corresponents als valors propis amb més correlació, això vol dir que podrem aproximar qualsevol entrenament utilitzant:

$$x \approx \bar{x} + \Phi b \tag{3.4}$$

El vector b defineix el conjunt de paràmetres del model deformable, és a dir, posa límits per tal que la forma deformada sigui semblant a la inicial. El nombre de vectors propis es pot escollir/descartar segons que representi una proporció de les dades (ex. 98 %) la resta es considera soroll. La figura 3.3 mostra visualment el resultat dels càlculs.

La variança de les dades respecte la forma mitjana bé definida pels valors propis, la suma dels quals ens proporciona la variança total de l'enternament. $V_T = \sum \lambda_i$.

Nosaltres podem definir aquest valor per tal d'assolir una correlació determinada en el model, això es fa de la següent manera:

$$\sum_{i=1}^{t} \lambda_i \geq f_v V_T \tag{3.5}$$

El número t defineix el total dels diferents modes que podrà adoptar la forma. Quan es vulgui escollir un en concret contindrà una proporció (f_v) de la variança total obtinguda en l'entrenament (V_T).

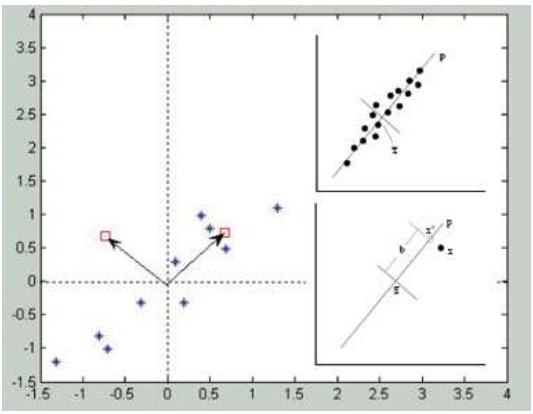

Figura 3.3: Exemples PCA: Núvol de punts i vectors propis sobreposats

Figura 3.4: Punts característics o landmarks

L'exemple classic per il·lustrar l'evolució dels models és el proposat per l'autor sobre l'expressió de les cares humanes, en la figura 3.4 es mostra un exemple de com es marquen els punts característics. També us mostrem un exemple del mateix autor Tim Cootes al seu article sobre l'Active Shape Models ([5]), en l'exemple (figura 3.5 es mostra la forma mitjana d'una cara en el punt inicial per començar l'adaptació a la forma, les iteracions són un paràmetre configurable, tot i que no sempre més iteracions correspon a més convergència, és important el punt inicial on es col·loca el mean shape.

Figura 3.5: Exemple d'evolució d'un mean shape facial

Cerca (Search)

La segona part del mètode és la de fer variar el model obtingut al training fins obtenir el resultat final que s'adapti a la forma d'una nova imatge. Tenint en compte els paràmetres es crea un model inicial $\mathbf{X}$ on es defineix la posició, orientació i escala i es van aproximant aquests valors amb el següent procés iteratiu:

1. Examinar la regió de la imatge al voltant de cada punt del model $\mathbf{X}$ per trobar el millor X_i'.
2. S'actualitzen els paràmetres amb el punt nou que és millor que l'anterior.
3. Repetir 1 i 2 fins a la convergència final.

Cada punt del model és examinat segons la imatge (figura 3.6).

Tot i aquesta transformació tan ideal no sempre els punts característics estan col·locats en el lloc correcte, a la imatge poden haver-hi més regions o formes i que, per qüestions del training un punt s'hagi fixat dintre d'algun d'aquests altres perfils existents.

Es presenten dos aproximacions per tal de localitzar punts correctes del contorn. Un dels mètodes consisteix en construir un model estadístic de l'estructura de la

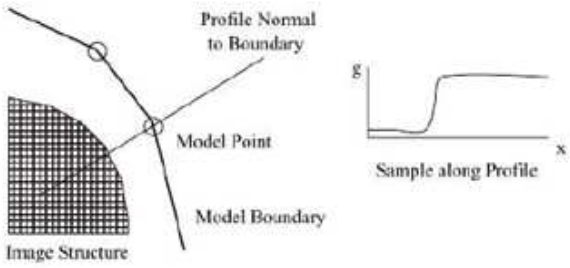

Figura 3.6: Procés d'adaptació de l'ASM a la imatge

imatge de manera que quan es cerquin els punts s'acceptaran els que més s'ajustin al model.

Aquest primer mètode per tal d'analitzar els punts de l'estructura rep el nom de *Modeling local structure*. El que fa és calcular la distància Mahalanobis de cada punt, aquesta mesura és utilitzada en estadística i es basa en correlacions entre les variables que intervenen en el càlcul per tal d'extraure característiques que les pugin identificar i analitzar. En el nostre cas el procediment és agafar un punt del contorn i els píxels veïns creant un model estadístic del nivell de gris del contorn de la imatge. Cada vegada que tinguem un punt a analitzar serà comparat amb aquest model si hi ha correspondència s'acceptarà creant així un nou punt a analitzar, d'aquesta manera la forma mitja anirà evolucionant fins a trobar la forma o fins que el procés iteratiu arribi al final.

El segon mètode contempla la possibilitat de tractar la tasca d'evolució com una classificació. Es recullen exemples que se saben que són certs, a la vegada que es recullen característiques de la imatge que estiguin situades a prop del contorn i que se sàpiga que són incorrectes, és a dir, que de cap manera pertanyen al contorn. Una vegada tenim les mostres es creen unes classes que permetran diferenciar punts correctes de punts incorrectes i alhora de buscar punt o analitzar-los s'escolliran els que es corresponguin als models certs i els que més difereixin i continguin menys característiques de les classificades com a negatives.

La idea principal de l'Active Shape Models ja la tenim, així doncs el mètode que modela definitivament l'estructura el que busca es efectivitat i bàsicament el que farà serà recorre els punts de la normal del perfil en els contorns del training per finalment construir el model estadístic en nivell de gris.

3.3.3 Pseudocodi del ASM

Igual que en capítol dels filtres veurem com s'estructura el codi de l'ASM en la figura 3.7. Primer amb el main principal desglosat en funcions.

```
FUNCIÓ ASM

    var
        Training, imatgeSortida,imatgeTest3D tipus volum3D;
        imatgeTest tipus imatge2D;
        Tolerancia tipus doble;
        númeroSlice,pacients tipus enter;
        meanShape, vectorPunts tipus vector;
    fvar

    Training = Creació_volum_training(pacients);
    meanShape = Entrenament_imatges(Training , tolerancia);
    imatgeTest = Llegir_imatge_test();

    si (imatgeTest és 3D) llavors
        per tots Slice de la imatge fer
            imatgeSortida = BuscarNovaForma(imatgeTest(numSlice),
                                            meanShape);
        fi per
    else
        imatgeSortida = BuscarNovaForma(imatgeTest, meanShape);
    fi si

    RETORNAR imatgeSortida

FI FUNCIÓ ASM
```

Figura 3.7: Pseudocodi Active Shape Model

El mètode du a terme tots els passos descrits en aquest apartat, el *training* i el *search* de la següent manera, es pot escollir si es calcula tot el volum 3D o només un tall d'imatge 2D i depenent de la decisió s'introdueixen uns paràmetres o uns altres al mètode. La implementació al visual té petites modificacions a aquest algorisme, ja que, aquest conté la idea bàsica de com es calcula l'ASM, en el capítol 4 es mostra amb més detall com s'ha adaptat el codi ITK.

```
FUNCIÓ CREACIÓ_VOLUM_TRAINING(pacients)

    var
        volumPròstata  TIPUS volum3D
    fvar
    per i desde 0 fins pacients fer

        slice=obir_slice(pacient);
        volumPròstata=afegir_slice(slice);

    fi per

    RETORNAR volumPròstata


FI FUNCIÓ CREACIÓ_VOLUM_TRAINING
```

Figura 3.8: Pseudocodi funció que crea el volum a partir de slice de diferents pròstates

3.4 Sumari

En aquest capítol hem analitzat el mètode Active Shape Models per segmentar imatges, calculant un model mitjà a través de mostres(mateix objecte que el volem segmentar) i posteriorment adaptant aquest a la forma real de la imatge a segmentar.

```
FUNCIÓ ENTRENAMENT_IMATGES(volum)
//meanShape engloba els punts, vectors propis i valors propis

    var
        vector tipus llista de punts x,y
        vectors_propis tipus matriu //conté vectors de vectors
        valors_propis tipus vector
    fvar
    mentre (no landmarksOK) fer
        vector(x,y)=assignar_landmarks();
        estadístiques = calcular_PCA(vector(x,y);
        vectors_propis = calcular_vector_propi(estadístiques);
        valors_propis = calcular_valor_propi(estadístiques);
        landmarksOK = verificar_landmarks();
    fi mentre

    meanShape = (vector(x,y), vectors_propis, valors_propis);

    RETURN (meanShape)

FI FUNCIÓ ENTRENAMENT_IMATGES
```

Figura 3.9: Pseudocodi funció Entrenament de models

```
FUNCIO BUSCAR_NOVA_FORMA(imatge, meanShape)
//meanShape engloba els punts, vectors propis i valors propis

    var
        nousPunts tipus llista de punts x,y
        imatgeFinal tipus imatge2D
    fvar
    mentre (i < numIteracions i no formaTrobat ) fer
        nousPunts = calcularMahalanobis(meanShape);
        formaTrobat = evaluar(nousPunts, modelEstadístic);
        //evalua si els nous punts pertanyen o no al contorn real segons e
        //mètodes descrits a la documentació
        i++;
    fi mentre

    imatgeFinal = marcar_punts(imatge, nousPunts);

    RETURN imatgeFinal

FI FUNCIÓ BUSCAR_NOVA_FORMA
```

Figura 3.10: Pseudocodi del search de l'ASM

Capítol 4

Anàlisi, disseny i implementació

Comença l'últim capítol abans d'extreure resultats dels mètodes implementats en el PFC. Entrarem una mica en detall en implementació del projecte per tal d'entendre com s'estructura. Igual que a la resta dels capítols dividirem el procés en pre-processat i segmentació i finalment s'unirà tot per donar una visió global de l'estructura.

4.1 Requeriments i anàlisi

Els objectius del projecte ja els hem anat citat al llarg de tota la documentació, s'han creat dos blocs de processat d'imatge els qual tenen la finalitat de filtrar i segmentar. Aquests dos blocs han de ser independents entre ells i respecte a qualsevol aplicació.

Per tal de visualitzar resultats necessitem d'una eina que sigui capaç d'obrir imatges del tipus US i MRI, tant volums 3D com imatges 2D, que són les que manipulem. Aquest tipus d'imatges les trobem amb format DICOM, aquest tipus de format correspon a un estàndard reconegut mundialment per l'intercanvi d'imatges mèdiques. A part de permetre totes aquestes accions que hem esmentat també ha de permetre afegir codi per tal d'ajuntar els dos blocs, així doncs aquesta serà la funció de la eina Visual, ja creada en un anterior PFC del mateix departament, dut a terme per A.Gubern [7].

El primer bloc, el de pre-processat ha d'executar diferents filtres sobre imatges d'ultrasons (Gauss, mediana, Anisotropic Diffusion) fent que l'usuari pugui escollir quin mètode utilitzar i els paràmetres de configuració que aquest pugui contenir.

Pel segon bloc, existeixen dues subparts, la de l'entrenament s'ha de saber sobre quines imatges es vol fer, en aquest aspecte es donaran dues opcions una és la d'utilitzar models ja definits de tal manera que la tasca sigui més automàtica i l'altre opció és que l'usuari pugi escollir quin volum s'utilitzarà per entrenar la forma mitjana.

La primera opció és per quan es vulgui segmentar un volum sencer, en canvi la segona opció serà únicament per segmentacions d'un sol tall o slice, el motiu és que quan es vulgui segmentar un slice no sabrem a quina part de la pròstata pertany el tall, per tant no es pot escollir automàticament el volum d'entrenament. Com ja veurem aquest volum utilitzat pel training està format per *slices* de diferents pròstates però que corresponen a una regió similar d'aquesta, llavors quan es vol segmentar tot el volum el procediment és generar diverses formes mitjanes que seran escollides alhora del *search* fent una estimació de la part de la pròstata a la que correspon el tall 2D que s'estigui analitzant en aquell moment.

Quan es segmenta un sol slice no se sap la regió de la pròstata, per tant no se sabria quina forma mitja escollir en el plantejament automàtic del training.

L'objectiu final es trobar bons resultats en imatges d'US on obtenir una correcta segmentació és més complexe. Tot i així també es pot treballar amb segmentacions d'imatges de MRI.

4.2 Classes i objectes

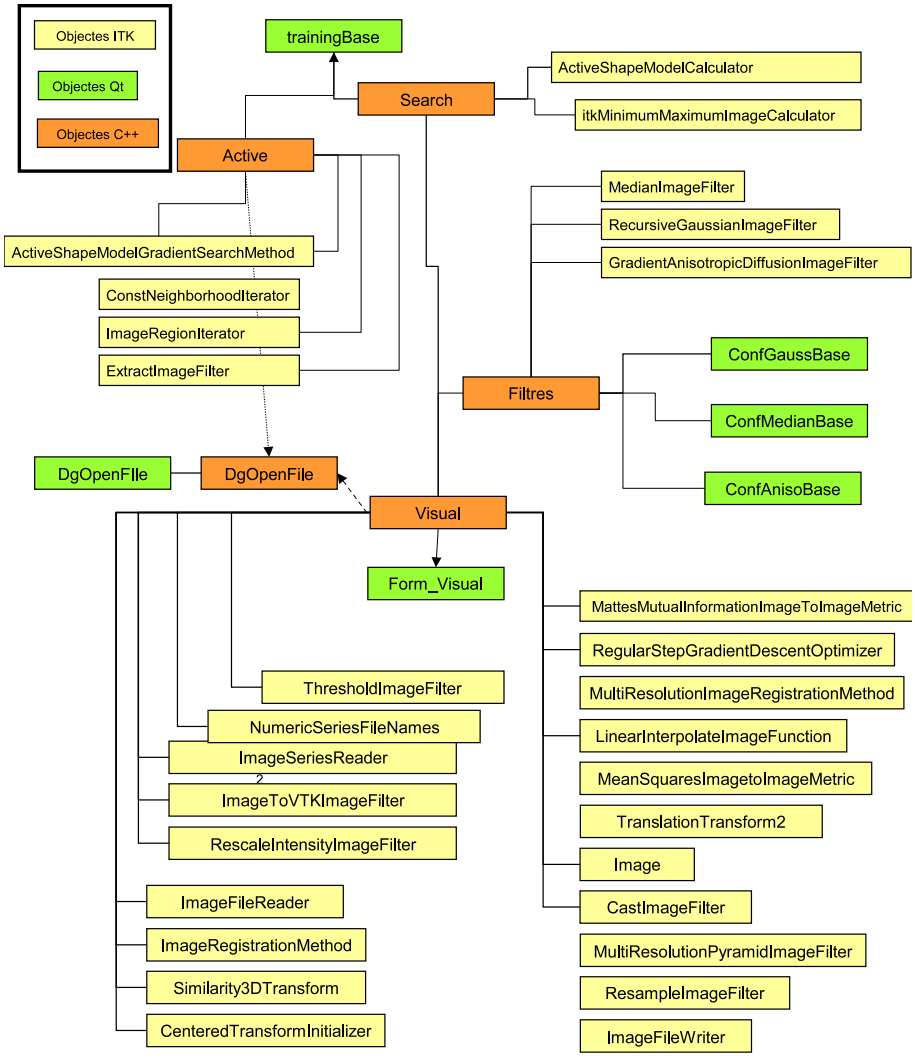

Figura 4.1: Diagrama de classes del PFC

En el diagrama de classes (figura 4.1) podem veure l'estructura general de les classes implementades en el PFC, juntament amb la del visual que serà la que instanciarà a la resta contenint l'estructura general de l'eina Visual.

El diagrama el podem entendre com classes i també com objectes ja que per cada classe tindrem un objecte creat amb el qual treballarem. Els diferents colors corresponen tal i com ens diu la llegenda depenent de si són objectes de c++ (color

taronja) on es duran a terme els càlculs corresponents, si són objectes ITK (groc) propi de les classes específiques amb una funció diferent cadascun d'ells, o si són objectes Qt (verd), part visual amb la qual el codi inicialment recupera els paràmetres que l'usuari introdueix i posteriorment mostra resultats.

Les classes i objectes que s'han generat durant aquest projecte són:

- Mòdul de filtratge: ***filtre.h***
- Mòdul de segmentació: ***active.h*** i ***search.h***

I els objectes Qt específics creats per capturar els paràmetres que es necessiten en les classes citades anteriorment han sigut:

- Mòdul de filtratge: ***confGaussBase.ui***, ***confMedianBase.ui*** i ***confAnisoBase.ui***
- Mòdul de segmentació: ***trainingBase.ui***

4.3 Estructura global de les llibreries ITK

Com ja hem citat en altres apartats la manera d'enllaçar tot un procés en ITK, des de l'entrada fins que obtenim la sortida final rep el nom de *pipeline*.

Existeixen diferents objectes a manipular dintre d'aquests processos, al llarg del capítol anirem mostrant els que han sigut necessaris en els nostres mètodes però hi ha un objecte bàsic que manipulem en totes les accions que duem a terme, les imatges.

En ITK les imatges són conjunts d'una o més regions. Aquestes regions són porcions de la imatge que poden ser processades per unes altres classes de forma independent. La regió més comú és la denominada *LargestPossibleRegion* que engloba tota la imatge. Un altre regió molt utilitzada és la *BufferedRegion* és la regió que es troba en memòria en el moment del procés i també la *RequestedRegion* la qual és cridada per un filtre o un altre classe per ser processada.

La manera de crear una imatge és indicant-li el tipus de píxel que contindrà i la dimensió que té, per exemple una del tipus char i 3D.

```
typedef itk::Image<unsigned char,3>ImageType;

ImageType::Pointer imatge;
```

Ja havíem dit que les classes ITK estan programades a través de *templates* per tant fent una crida a la classe ïtkImage.h"fàcilment creem un punter a una imatge amb les instruccions que tenim adalt.

Ja tenim creades les imatges formades per regions però hem de saber com cridar-les. Les regions es defineixen amb dos classes que són l'*Index* i el *Size*. La primera defineix l'origen de la regió mentre que la segona l'extensió que aquesta té. Per exemple si nosaltres volguéssim treballar només amb una regió específica de la imatge que abans hem cridat necessitaríem el següent codi:

1. Definim l'origen.
   ImageType::IndexType start;
   start[0] = 0; // origen index en X
   start[1] = 0; // origen index en Y
   start[2] = 0; // origen index en Z

2. Definir l'extensió
   ImageType::SizeType size;
   size[0] = 200; // extensió en píxels de X
   size[1] = 200; // extensió en píxels de Y
   size[2] = 200; // extensió en píxels de Z

3. Crear regió
   ImageType::RegionType region;
   region.SetSize(size);
   region.SetIndex(start);
   image->SetRegions(region); //Fa que la nova regió sigui la *largest, buffered i requested* de manera simultània
   image->Allocate(); // situar a memòria, si més endavant es modifica la regió s'ha de tornar a invocar aquest mètode

Una vegada sabem manipular les imatges veiem un exemple de pipeline, concretament el que fa es obrir una imatge i tornar-la a guardar en un fitxer. Creem el *reader* i el *writer* que són dos classes d'ITK que com el seu nom indica llegeixen i escriuen. Quan creem qualsevol objecte al ser classes *templates* s'especifica amb quin tipus d'imatge es treballarà <ImageType> on *ImageType* nosaltres ja ho tenim definit.

```
typedef itk::ImageFileReader <ImageType> ReaderType;
ReaderType::Pointer reader = ReaderType::New();
typedef itk::ImageFileWriter <ImageType>ReaderType;
ReaderType::Pointer writer = WriterType::New();

reader->SetFileName("nom del fitxer entrada");
writer->SetFileName("nom del fitxer de sortida");
writer->SetInput(reader->GetOutput);
writer->Update();
```

Si es volgués rescalar la imatge abans de guardar s'introduiria un mòdul del tipus Rescaler<ImageType> i el pipeline quedaria

rescaler->SetInput(reader->GetOutput) i *writer->SetInput(rescaler->GetOutput)*

4.4 Filtratge

Els filtres conformen el primer bloc del PFC, el qual esta programat amb classes tipus *template* per tal de poder-lo inserir no només al Visual de manera senzilla, també a futures aplicacions que necessitin un bloc de pre-processat.La utilització de *templates* resideix en que el C++ és un llenguatge de programació genèric, és a dir, es fixa més en els algorismes que en en les dades. La idea principal és crear funcions genèriques parametritzades i simples les quals es poden introduir en altres programes fàcilment. Resumint, els templates són plantilles les quals formen part dels paràmetres d'una classe o funció. Alhora de cridar el mètode o crear l'objecte de la classe és farà indicant el tipus amb el qual es treballarà dintre d'aquella classe, per exemple en el nostre cas el mòdul filtratge al estar programat amb *templates* permet que canviant una sola línia de codi es pugui canviar el tipus d'imatge amb les quals es treballa (*int*, *char*, *double*, etc.).

El fet que aquest bloc hagi esdevingut part de la interfície gràfica *visual* ha afavorit que es conegui i es treballi l'estructura d'aquest entorn gràfic que conté moltes més classes i per tant, és més elaborada ja que no utilitza només ITK sinó també VTK i Qt. La part que analitza la integració al visual la trobareu a l'apartat 4.6.

El mòdul de pre-processat està programat dintre de la classe filtre.h, creant un objecte tipus filtre. En aquesta classe filtre trobem els tres filtres, Gaussià, mediana i Anisotropic Diffusion que passem a analitzar seguidament, però abans esmentar que en aquesta classe tenim 4 mètodes, un *SetInput(Imatge)* amb el qual es configura la imatge a processar, també un *GetOutput()* que retorna la imatge després de ser filtrada i per últims els dos mètodes que donen forma als filtres pròpiament dits. Un d'aquests mètodes és el *aplicarFiltre(booleà)*, en el qual el booleà ens serveix per escollir entre filtre Gaussià i el filtre de la mediana. El filtre Anisotropic Diffusion s'aplica a través del mètode *aplicarAnisotropic()*.

4.4.1 Filtre Gaussià

En el pseudocodi de la funció de gauss que trobareu a la figura **??** del capítol **??** ja hem introduït una part del disseny d'aquest filtre esmentant que s'havia de repetir tres vegades el codi per tal de generar un filtre per cada direcció. El mètode per indicar aquesta direcció de filtratge és el filtre->SetDirection(enter) on el valor enter varia 0 per la x, 1 per la y i 2 per la z. L'altre paràmetre era l'ordre que fa referència

a que el mètode pot aproximar-se a la funció Gaussiana, a la primera derivada o a la segona utilitzant els paràmetres *ZeroOrder*, *FirstOrder* o *SecondOrder*.

La normalització Across Space-Scale consisteix en preservar al màxim els valors inicials de la imatge i es pot escollir aquest tipus de normalització fent cert el booleà que té com a paràmetre el següent mètode filtre->SetNormalizeAcrossScale(booleà). L'equació que segueix per tal de dur a terme la normalització és la que es mostra en les equacions 4.1 sent la primera l'Across space-scale.

$$\frac{1}{\sigma\sqrt{2\pi}} \qquad\qquad \frac{1}{\sigma^2\sqrt{2\pi}} \tag{4.1}$$

s'introdueix el valor de la sigma amb el mètode *filtre->SetSigma(doble)* i ja podem configurar l'entrada. La manera de configurar l'entrada per tal que agafi les tres direccions esmentades serà creant un pipeline entre elles de la següent manera:

```
filtreX->SetInput( imatgeEntrada );
filtreY->SetInput( filtreX->GetOutput() );
filtreZ->SetInput( filtreY->GetOutput() );
```

Finalment per recuperar la sortida només hem d'invocar el mètode *imatgeSortida = filtreZ->GetOutput()* fixant-nos en que la sortida es crida a través de la última direcció creada, ja que és el final del pipeline. Així doncs l'objecte filtre crear el volum 3D o el tall 2D amb el filtratge generat, l'usuari a través de la interfície pot escollir el valor de la sigma, la resta (ordre i normalització) són valors que estan configurats per defecte.

La classe que conté tota l'estructura interna del filtre i aplica finalment la funció és la itkRecursiveGaussianImageFilter.h.

4.4.2 Filtre Mediana

El filtre de la mediana trobem el seu pseudocodi al capítol 2 a l'apartat **??** però no és exactament el que hem hagut d'implementar en ITK ja que simplement s'han hagut de configurar els paràmetres, tot i això, per poder veure el funcionament del filtre el pseudocodi mostra la configuració i també el funcionament intern del mètode que es troba dintre de la classe itkMedianImageFilter.h.

Aquest filtre es resumeix bàsicament en escollir el kernel que s'utilitzarà per fer la convolució amb la imatge d'entrada. La instrucció que ens permet escollir la màscara és la *filtre->SetRadius(vectorSize)* a la qual se li passa un vector tipus *size* que indica la quantitat de píxels que es tindran en compte per fer la màscara, és a dir, si el vector "radius"es configura de la següent manera per una imatge 2D:

```
InputImageType::SizeType indexRadius;
indexRadius[0] = 1; // radius along x
indexRadius[1] = 1; // radius along y
filtre->SetRadius( indexRadius );
```

executarà amb una finestra 3x3, l'explicació és senzilla, el valor que se li assigna són el número de píxels que s'agafaran per cada costat del píxel central, si en canvi assignéssim 1 i 2 la finestra seria de 3x5.

Amb aquest mètode ja hem vist tot el que s'ha de fer per configurar el filtre de la mediana simplement faríem un *filtre->Update()* i un *imatge Sortida = filtre->GetOutput()* igual que amb la resta de filtres.

En el nostre mètode es possible filtrar amb 2 tipus de finestres, la 3x3x3 (ja que són imatges amb 3D) o amb un 5x5x5.

4.4.3 Filtre Anisotropic Diffusion

El filtre anisotropic consta de 3 paràmetres que ja hem explicat en el capítol 2 a que fan referència. Ara ens centrem en la seva configuració. Els mètodes que permeten introduir-los són:

```
filtre->SetNumberOfIterations( iteracions );
filtre->SetTimeStep( temps );
filtre->SetConductanceParameter( conductance );
```

Com sempre després és fa un Update() i GetOutput() per recuperar la sortida.

Aquests mètodes els trobem a dintre de la classe itkGradientAnisotropicDiffusionImageFilter.h.

4.5 Active Shape Models

Igual que en mòdul de pre-processat la segmentació s'ha programat amb classes tipus *template* i s'ha dividit la implementació en dos classes : *"active.h"* i *"search.h"*. Tal i com el seu nom indica la primera fa tot el procés de calcular la forma mitjana de la pròstata, s'encarrega del *training*, per altre banda el search fa la cerca al volum i la visualització final d'aquest.

L'active necessita de la classe itkActiveShapeModelCalculator.h per crear l'objecte tipus active i tenir els mètodes que donen forma al training. Com que nosaltres treballarem amb imatges 3D es crearà un objecte tipus active amb la següent instrucció Active<ImageType> active; que prèviament ja teníem definida amb quina tipus d'imatge es treballava (píxel tipus char, 3 dimensions),amb això ja es podrà

operar amb els mètodes de la classe que calcula la forma mitjana tal i com s'ha vist al capítol 3. L'active conté 7 mètodes que passem a descriure per ordre d'execució:

SetInputTraining(format, primer, últim, pas) Configuració del training al qual li passem la ubicació de les imatges que utilitzarà per entrenar tenint el format del path per obrir el volum. Aquest volum es llegirà amb un reader del tipus sèries que obrirà totes les imatges que continguin al nom el *format* començant per el *primer* fins *últim* contant de *pas* en *pas*. Per exemple tenim una sèrie de imatges DICOM que tenen per nom *dicom1.dcm, dicom2.dcm, dicom3.dcm*, etc. fins al 9 llavors li passaríem al sèries
active.SetInputTraining(dicomAquest mètode retorna un volum, en el nostre cas 3D

SetInput(Imatge,tolerància) En aquest mètode li passarem el volum que hem obtingut del training. Aquesta imatge forma part de les variables del mètode que després es farà servir per calcular la forma mitjana amb una "tolerància"que també es una variable global de la classe.

CalculateASM() Calcula la forma mitjana i guarda els punts obtinguts com un vector així com els vectors de valors propis, vectors propis.

GetOutputASM() La classe que calcula l'active té un mètode que mostra per pantalla tots els valors calculats així com les característiques inicials de configuració; número de imatges de training (talls 2D que conté el volum), tolerància, punts de la forma mitjana trobada, valors propis, vectors propis. El nostre getOutput mostra tots aquests valors per pantalla.

GetEigenValues() Retorna el vector dels valors propis.

GetEigenVectors() Retorna la matriu que té els vectors propis.

GetMeanShape() Retorna vector amb la forma mitjana del model.

Una vegada hem vist els mètodes l'estructura a seguir es clara:

```
imtrain= active.SetInputTraining(formato,primer,ultim,pas);
active.SetInputActive(imtrain, tolerance);
active.CalculateASM();
```

i una vegada vulguem utilitzar els vectors resultants obtinguts per calcular el search farem:

```
MatriuVectorsPropis = active.GetEigenVectors();
VectorValorsPropis = active.GetEigenValues();
VectorFormaMitja = active.GetMeanShape();
```

Existeixen uns requisits per tal que els càlculs de la forma mitjana sigui correcte, un d'ells es que el volum que se li passa al training no és una pròstata sencera sinó que li passem un volum creat manualment, és a dir, s'agafen talls de diferents pacients per tal de donar varietat a la forma de la pròstata. Quan s'explicaven els fonaments de l'Active Shape Models vam dir que un òrgan humà pot adoptar formes lleugerament diferents no només en diferents pacients sinó també en un mateix pacient com a conseqüència de l'evolució de la malaltia. Així doncs s'ha de donar al mètode aquesta possible variabilitat. En el nostre cas com veurem en apartats futurs no es disposa d'una gran base de dades per poder generar models variats però si que aconseguírem fer-nos una idea del funcionament del mètode així com els seus possibles resultats.

L'altre requisit de la classe *ModelCalculator* és que l'entrada és a dir les imatges de training han d'estar binaritzades per tal que es pugui diferenciar clarament la zona que es vol modelar, en el nostre cas la pròstata. Aquest procés és el que nosaltres hem dut a terme amb l'eina ITK-SNAP (Capítol 3, apartat 3.2).

El search.h és la segona classe implementada i necessita la classe ITK itkActiveShapeModelGradientSearchMethod.h, així com d'altres que ja ens anirem trobant en el seu anàlisis.

Mostrem primer els mètodes que té la classe i després analitzem l'estructura. En el cas del search farem servir dos objectes un de tipus 2D i l'altre de tipus 3D, com que les classes són *template* una vegada creat l'objecte treballa sempre amb el mateix tipus d'imatge i si no ho fem així tenim problemes de compilació.

Així doncs els mètodes són els següents:

SetInputSearch(Imatge, profile, numIter, vectorsPropis, valorsPropis, formaMitja) Els últims tres paràmetres ja els coneixem ja que corresponen a la sortida de la classe active.h. La resta de paràmetres són de configuració de la cerca, número d'iteracions que farà el mètode per buscar la forma i la grandària dels contorns que volem processar.

SetInputAux(Imatge) És l'input que farem servir per l'objecte *search* que durà a terme la part de mostrar els resultats, només cal introduir-li la imatge ja que la resta d'informació ja l'obtindrà a través d'un fitxer.

GenerateSearch() Fa la cerca a través del volum amb la informació que li hem passat amb el primer mètode input que hem vist, tenint en compte cap a on creix la imatge, la forma mitja que té i la resta de paràmetres.

GetOutputSearch() Igual que amb l'*active* el *search* té un mètode de printout (mostrar) els valors obtinguts i amb aquest GetOutputSearch() a part de retornar el vector amb els punts finals de la segmentació el que fem és mostrar aquests valors per pantalla.

Mostrar() Quan s'han generat els punts de la segmentació en el mètode GenerateSearch() els punts resultants es guarden en un fitxer de sortida el qual es manipula amb aquest mètode que el prepara per ser tractat alhora de marcar els punts a la imatge o al volum de sortida, diem que el fitxer s'ha de tractar ja que per qüestions d'implementació al principi de fitxer s'introdueixen el número total de punts que es pintaran, així amb els mètodes de manipulació de la imatge (marcarPunts() i addPoints()) el treball és més senzill.

marcarPunts(slice) Una de les formes de visualització final de la segmentació serà marcar els punts dintre de la imatge i el tall corresponent per tal de localitzar la pròstata. Aquest és el mètode encarregat de fer-ho sobre el tall 2D que se li indica a través del paràmetre "slice". Retorna la imatge resultant. Els punts els treu del fitxer que com hem dit ha organitzat el mètode Mostrar().

AddPoints(slice) Aquest mètode gestiona un altre manera de visualitzar el resultat fent que els punts obtinguts s'uneixin per mitjà duna recta, així doncs obtenim un perfil sencer. Aquest mètode es complementa amb el que explicarem a continuació el bresenham(). També retorna la imatge resultant.

Bresenham(xinici,yinici,xfinal,yfinal,slice) Algorisme per dibuixar una línia entre el punt (xinici,yinici) i el (xfinal,yfinal) amb l'algorisme que porta el nom del mètode. Fent una petita descripció només esmentar que és un mètode que només utilitza sumes i restes per fer els càlculs per la qual cosa és efectiu i ràpid cosa molt valorada quan es treballa amb gràfics o imatges ja que els costos de temps de computació solen ser elevats.([6] Jack E. Bresenham , "Algorithm for computer control of a digital plotter", January 1965).

extraureProstata(Imatge, primer, últim) Aquest mètode crea una màscara per tal de calcular el volum final de la pròstata, de manera que conta tots els punts que hi han dintre del contorn definit anteriorment pel bresenham i els hi fa el càlcul del volum segons l'spacing (proporció real del píxel en mm) que tenen i com que sabem quants slice tenim, podem fer un càlcul aproximat del volum total de la pròstata.

Igual que abans farem un recorregut a l'estructura que es seguirà per tal d'executar el search.

```
Search<Image2DType> search;
Search<ImageType> searchAux;
searchAux.SetInputAux(volum); //Per mostrar
search.SetInputSearch(imatge2D,profile, númeroIteracions, vectorsPropis, valorsPropis,formaMitjana);
search.GenerateSearch();
search.GetOutputSearch();
searchAux.Mostrar();
```

imatgeSortida = searchAux.addPoints(slice) o searchAux.marcarPoints(slice);

ja sabem que es poden enllaçar els mètodes a través dels pipelines, l'active i el search ho fan de la següent manera, alhora de cridar el SetInputSearch() els paràmetres de "vectorsPropis", "valorsPropisï "formaMitjana"es passen

search.SetInputSearch(imatge2D,profile, iteration, active.GetEigenVectors(), active.GetEigenValues(),active.GetMeanShape());

4.6 Integració al visual

Ja hem vist tota la implementació individual del projecte ara només falta integrar-ho dintre de l'aplicació Visual.

Com ja hem citat anteriorment aquesta interfície gràfica on s'integra aquest projecte correspon a un altre PFC ([7]) que tenia com objectiu relacionar les proves de MRI i proves d'US a través d'una eina visual que fos senzilla d'utilitzar a temps real. Aquesta correspondència nosaltres no la farem servir, en el PFC simplement s'insertaran els mòduls que hem vist anteriorment per tal de crear una eina més robusta, amb més opcions per tal d'analitzar les imatges mèdiques amb més qualitat. El Visual com s'anomena la interfície utilitza les llibreries VTK per visualitzar les imatges mèdiques i les QT per implementar la interfície gràfica on s'utilitzen ITK i VTK. Els dos mòduls s'han afegit com a dos menús en la part superior de la

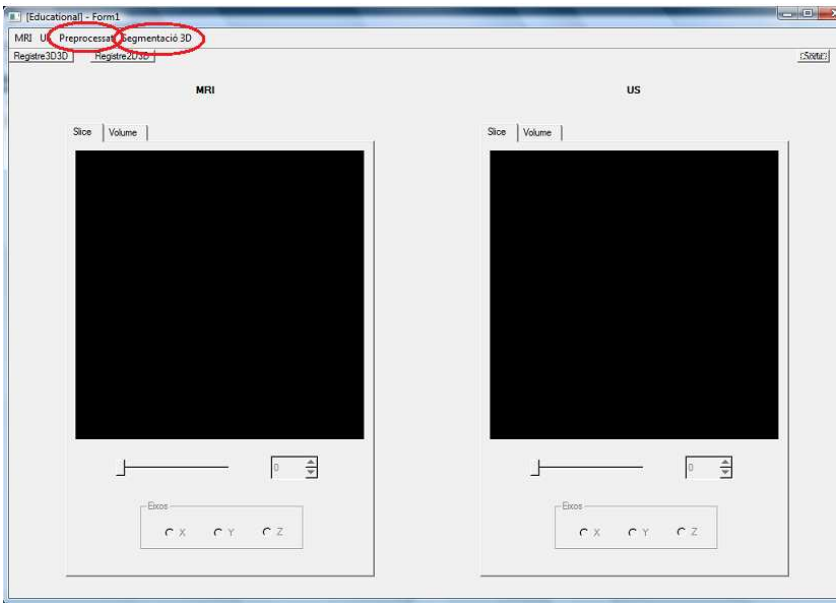

Figura 4.2: Pantalla inicial Visual

pantalla inicial. No entrarem gaire en detall en el funcionament del visual però si

donarem les pautes necessàries per tal de poder manipular la informació i utilitzar els mètodes de la part de pre-processat de la imatge i segmentació. A la figura 4.2 veiem la pantalla inicial de l'aplicació.

També està marcat en color vermell els dos ítems del menú que s'han afegit en el PFC. El procés és senzill segons si volem treballar amb imatges MRI o amb US, dintre dels menus corresponents et permet obrir com un volum o com una imatge 2D. Com ja havíem dit abans s'ha treballat a alt nivell amb les Qt ja que s'han hagut de crear els objectes que recullen els paràmetres dels nostres mètodes. A la figura 4.1 on tenim el diagrama de classes i a la figura **??** hem vist els objectes que intervenen en el procés, ara veurem com fer les crides per tal d'enllaçar els objectes amb els mètodes.

Quan hem mostrat la figura del visual (figura 4.2) hem dit que s'han creat dos menús això en Qt comporta crear dos events de click associats a una acció que finalment fa la crida a un mètode, és a dir, quan es produeixi un event del tipus click s'enviarà una senyal la qual tindrà un mètode associat en el que hem configurat el codi a executar. Per exemple, el filtre Gaussià és el primer del menú

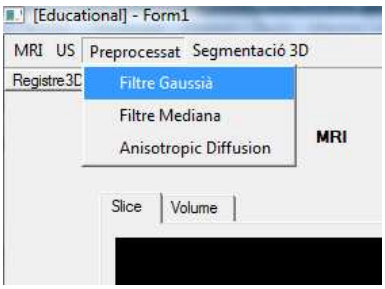

Figura 4.3: Menú pre-processat

pre-processat(figura 4.3) i té com a nom preprocessatFiltreGaussiAction llavors el que hem de fer a la classe visual que controla aquests events és configurar-lo per tal que executi el codi Gaussià:

```
connect(preprocessatFiltreGaussiAction,SIGNAL(activated()),this, SLOT(GaussUS()));
```

el codi és fàcil d'entendre, connecta l'event del menú Gaussià amb el mètode GaussUS() en el qual es crea un objecte filtre, que operarà amb els mètodes que s'han vist anteriorment corresponents a la classe filtre.h. El codi del filtre ja l'hem vist prèviament a la figura 2.3.

Una vegada estem dintre del mètode corresponent i s'ha creat l'objecte amb el qual es treballarà s'han de configurar els paràmetres específics de cada classe. Aquest procés de configuració es fa a través d'objectes Qt que s'han creat a mida per cada mètode, els més senzills són dels filtres, que recullen valors numèrics, veiem les tres

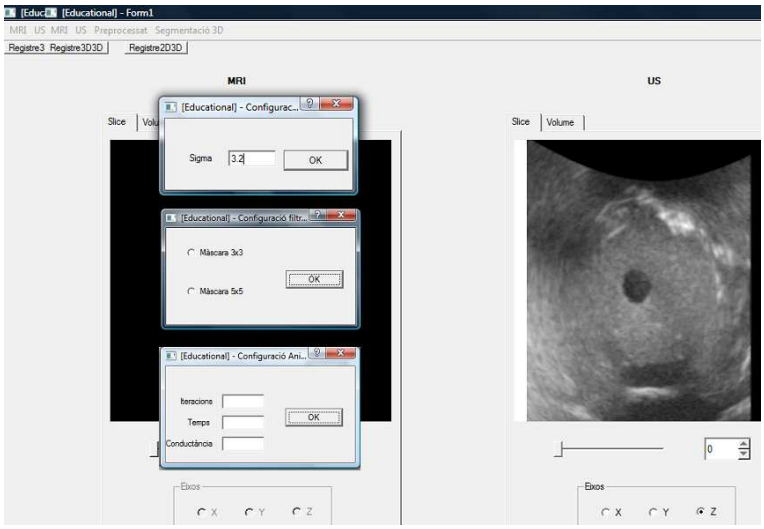

Figura 4.4: Finestres de configuració dels mètodes

captures de les configuracions a la figura 4.4. Veiem la pantalla inicial del Visual i hem sobreposat les tres finestres de configuració de filtres.

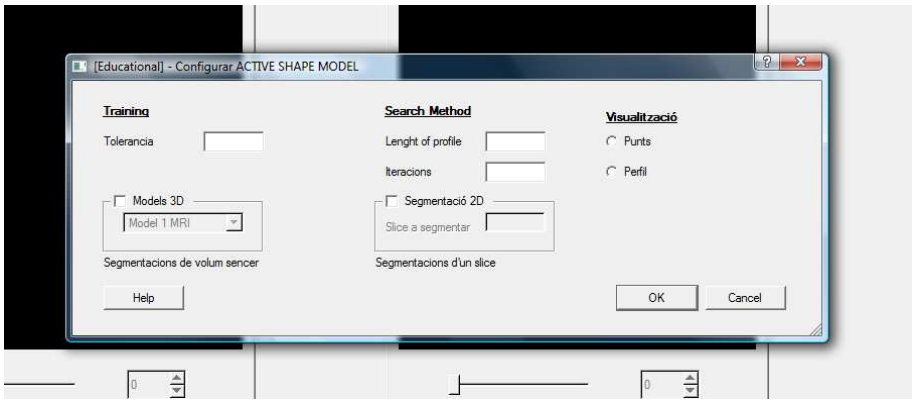

Figura 4.5: Finestra de configuració del mòdul de segmentació

L'altre objecte Qt creat ha sigut l'específic de la segmentació, aquest és una mica mes extens perquè recull totes les possibilitats de segmentació, tenim la captura a la figura 4.5. Ràpidament veiem la part del training on s'escull la tolerància que tindrà l'Active Shape Models alhora de permetre les variacions de contorn (valors propis, vectors propis). A la part del search escollim el número d'iteracions per tal de trobar la forma de pròstata i la grandària del perfil. La part de visualització correspon a que com veurem en el capítol de resultats s'han introduït dues maneres de visualitzar els punts finals de la segmentació, una només marca els punts de color blanc a la imatge original i l'altre a part de marcar els punts també els uneix per mitjà d'una recta creant així tot el perfil de la segmentació, aquesta última és

una evolució del programa original Visual ja que també s'està treballant en altres mètodes de segmentació.

Per últim escollim si la segmentació serà només d'un tall de la imatge o de tot el volum que ja veurem que implica el poder escollir el volum d'entrenament o no. La idea és si es segmenta només un slice es pot triar amb quines imatges es genera la forma mitjana, però si es vol segmentar tot el volum el que es fa es tenir uns volums genèrics on el mètode ja divideix el volum final en slice i fa els càlculs. El fet de no tenir moltes imatges fa que s'hagi generat un volum per segmentar imatges MRI, un per US i un per phantoms, però que això pogués anar creixent tenint una base de dades amb diferents models per escollir l'entrenament més adient a cada pacient.

4.7 Sumari

En aquest capítol hem vist d'estructura general del codi del projecte, quines classes conté, quines han estat creades per dur a terme les accions del projecte i quines ja existien en l'eina. També hem vist de quina es relacionen l'aplicació existent amb els nous mètodes.

Capítol 5

Resultats

En aquest capítol es presenten els resultats d'eliminació de soroll i segmentació dels mètodes presentats en els capítols anteriors. A l'apartat 2.6 del capítol 2 vam examinar els paràmetres dels filtres implementats i al capítol on es tractava la segmentació a través de l'Active Shape Models també vam mostrar alguna imatge com la figura 3.6 on es veia l'evolució del *mean shape* alhora de fer el *search* del mètode. Així doncs, analitzem ara els mateixos paràmetres configurables però dintre de les classes creades en el PFC.

5.1 Pre-processat

El mòdul de pre-processat està preparat per treballar amb volums 3D d'imatges d'US, és a dir, conjunt d'imatges com la que es mostra en la figura 5.1, on es pot apreciar que els contorns no estan ben definits i que existeix soroll a la imatge el qual volem reduir. Depenent del mètode treballarem amb imatges tipus DICOM de phantoms on s'ha introduït per mitjà del Matlab soroll de tipus Gaussià o Sal i Pebre ja que podrem apreciar els efectes dels filtres més clarament.

Ho farem diferenciant entre els tres mètodes implementats per tal de veure els diferents resultats segons si s'utilitza un valor de paràmetre o un altre.

Filtre Gaussià

El paràmetre configurable en aquest mètode és la sigma (σ) la qual farem variar per tal de comprovar que el mètode realitza el filtratge correctament. Per tal de calcular el kernel, el mètode concret ITK utilitza l'expressió 5.1.

$$\frac{1}{\sigma\sqrt{2\pi}}\left(-\frac{x^2}{2\sigma^2}\right) \tag{5.1}$$

Figura 5.1: Exemple imatge real d'ultrasons

El rang de valors a testejar han sigut des de $\sigma = 0.5$ fins a $\sigma = 3.5$, seguint el següent procediment, hem anat augmentant la σ en intervals de 0.3 i comparant resultats amb l'anterior filtratge buscant diferencies significatives per determinar rangs de valors acceptables per utilitzar aquest mòdul. La manera de poder comparar de manera ràpida i visual els resultats el que farem serà a l'aplicació Visual obrir la imatge original com si fos MRI i i es veurà a la visualització de la part esquerra i el filtratge es durà a terme a la part dreta que correspon a la visualització d'ultrasons.

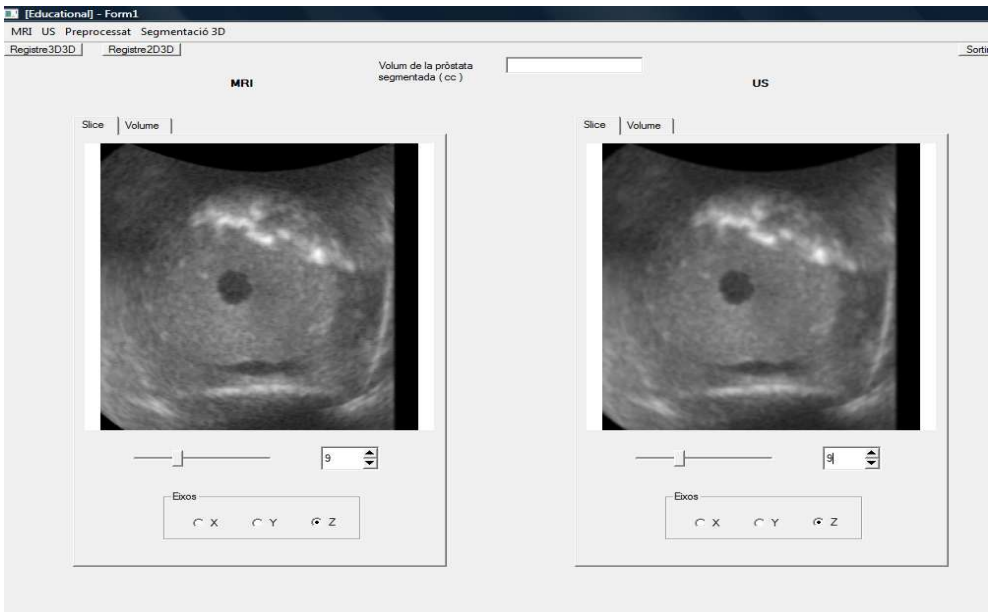

Figura 5.2: Filtratge de Gauss amb sigma = 0.8

Fent les proves inicials amb $\sigma < 1$ s'ha pogut comprovar que en imatges d'US no es tenen resultats apreciables a simple vista fins que no s'agafa un valor superior a 0.8 on ja es comença a apreciar la disminució de soroll a la vegada que la imatge

es torna més difosa. En l'exemple de la figura 5.2 pràcticament no es diferencien els resultats però augmentant la sigma fins a valors com per exemple amb $\sigma = 1.9$ veiem com el filtratge és molt més acusat (figura 5.3).

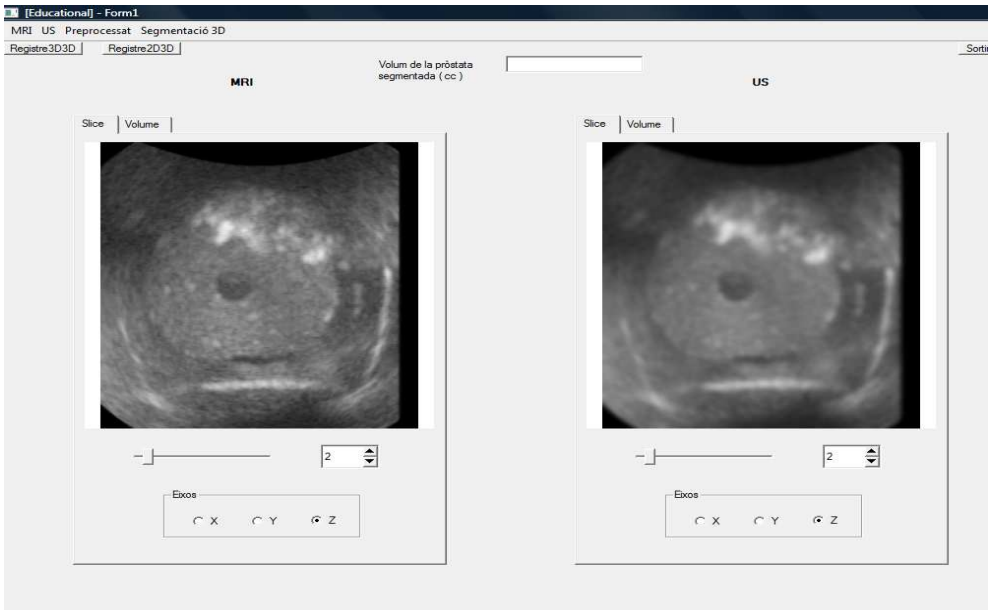

Figura 5.3: Filtratge de Gauss amb sigma = 1.9

Per últim quan els valor de $\sigma > 2.3$ (figura 5.4 els contorns de la pròstata comencen a no apreciar-se ja que la imatges es torna massa difosa, així podríem concloure que depenen de l'ús que posteriorment hagi de tenir la imatge s'hauria d'utilitzar el filtratge Gaussià amb un rang de valors $0.5 < \sigma < 2.5$.

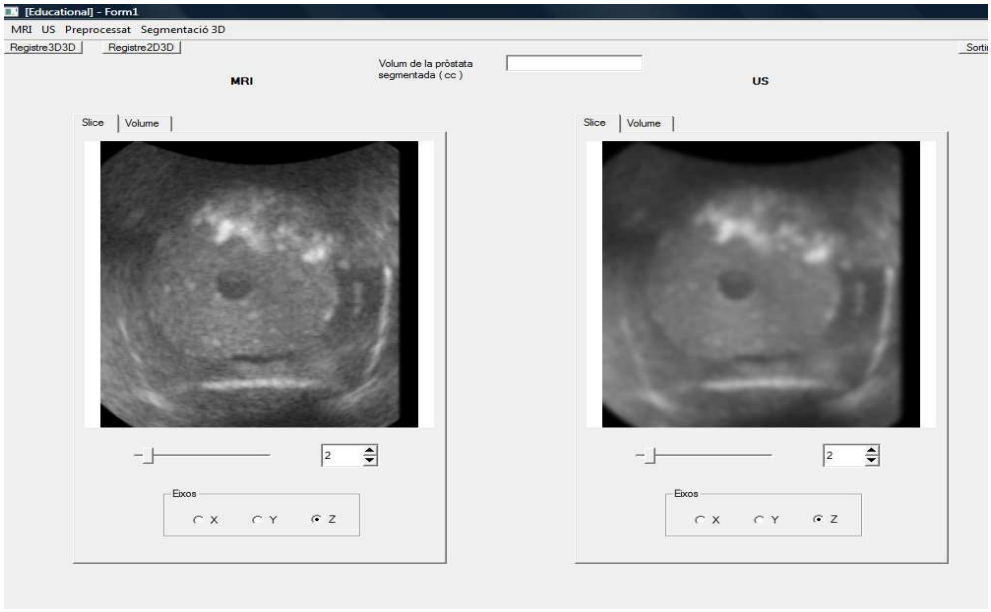

Figura 5.4: Filtratge de Gauss amb sigma = 2.3

Filtre de la mediana

El filtre de la mediana es pot utilitzar en aquesta aplicació convolucionant dos tamanys diferents de màscares, la 3x3 i la 5x5. En la primera figura 5.6 es mostra el resultat d'aplicar a la mateixa imatge que en l'apartat de Gauss però amb una màscara 3x3, es pot apreciar que la reducció del soroll és gran i que la relació eliminar del soroll / respectar contorns és major que amb el filtre de Gauss.

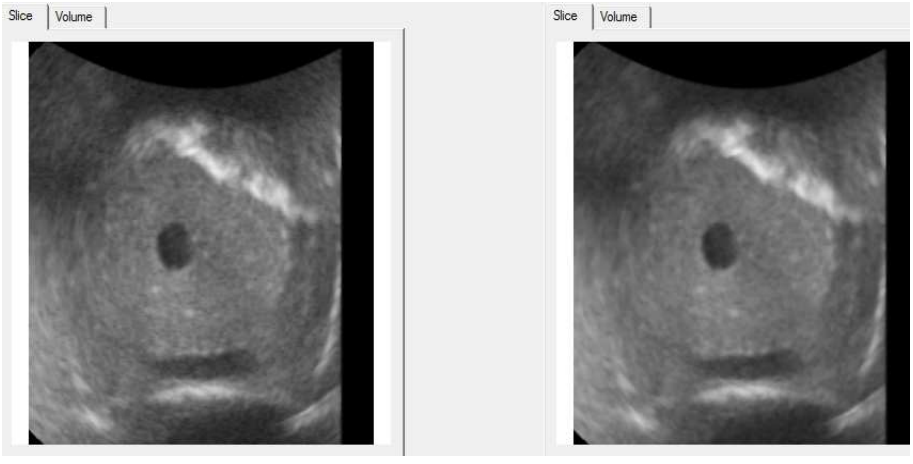

Figura 5.5: Filtratge de la mediana amb màscara 3x3

Les diferències de filtratge entre aplicar el filtre amb màscara 3x3 i màscara 5x5 no són gaire significatius amb aquest tipus d'imatges que són tan homogènies, recordem que aquest filtre era eficient sobretot per eliminar soroll del tipus salt i pebre.

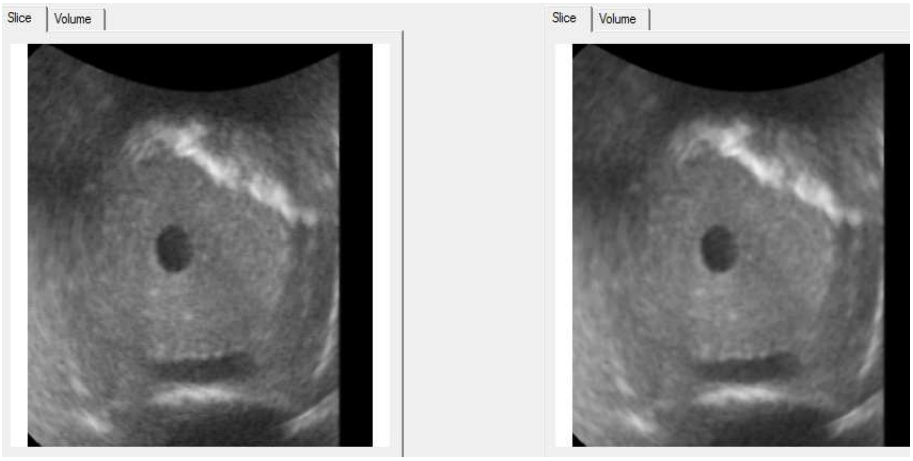

Figura 5.6: Filtratge de la mediana amb màscara 5x5

Filtre Anisotropic Diffusion

L'últim filtre del mòdul pre-processat és el filtre de Anisotropic Diffusion que consta de tres paràmetres(número d'iteracions, temps i conductància). Alhora de fer l'anàlisis del filtre en l'apartat 2 ja hem vist quines diferències hi han entre modificar els paràmetres, veiem ara aquest mostreig aplicat a les imatges d'US. Consultant la documentació de les ITK veiem que el rang de valors que proposen són més petits, amb imatges 3D proposen un temps de 0.125s, per iniciar el procés d'anàlisis executem l'exemple de la figura 5.7.

Pensem que els filtre Anisotropic tenia com objectiu reduir el soroll però preservant els contorns, en la imatge obtinguda no s'aprecia aquest resultats, un dels motius pot ser que la imatge original no té els contorns ben definits ja que el nivell de gris de tota la imatge és similar i realment és difícil diferenciar on es troba la pròstata. Un altre exemple (figura 5.8) s'ha executat utilitzant una conductància de 20 iterant 80 vegades. El fet que el mètode hagi de tenir un temps tant baix perquè sinó genera un warning (avís) en el qual diu que el mètode no esta treballant de manera estable i que pot produir errors fa que per produir efectes similars s'ha d'augmentar el número d'iteracions considerablement. A la figura s'han marcat dos zones ja que són les que inicialment tenen els contorns més definits i veiem que el comportament del mètode si que es preservar i accentuar els contorns.

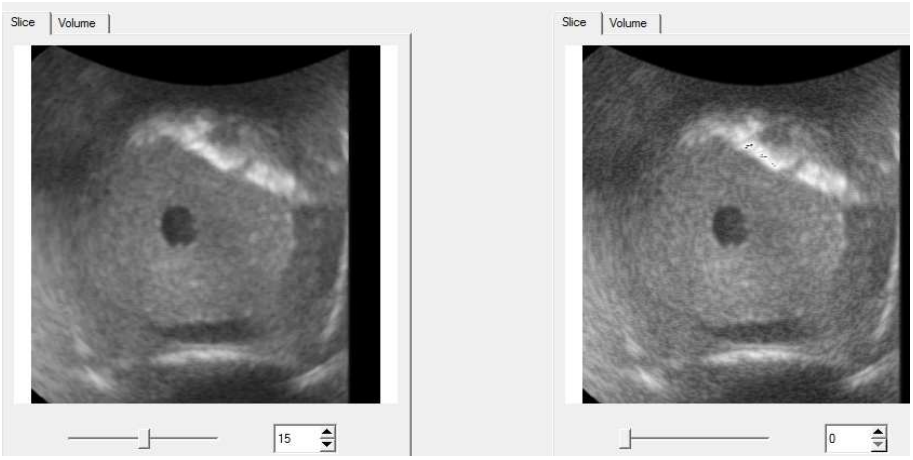

Figura 5.7: Anisotropic diffusion temps=0.03 iteracions=40 conductància=3.0

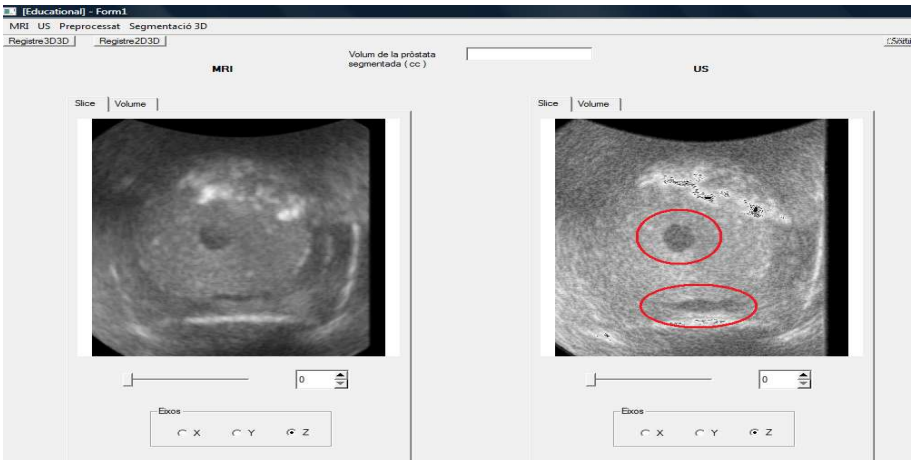

Figura 5.8: Anisotropic diffusion temps=0.06 iteracions=80 conductància=20.0

Resultat Globals filtratge

Després d'executar els tres filtres amb diferents paràmetres i veure els resultats que comportem ha de quedar constància que a part del resultat final que es pot observar s'ha de tenir en conte un altre factor que no s'ha comentat encara, el cost computacional que tenen aquests mètodes.

Les diferències s'han fet notòries entre el filtre Gaussià i la resta de filtres, sent el primer el més ràpid de tots. Inicialment les proves s'han realitzat amb un volum 3D que consta de 30 *slices*, que per extreure les captures de pantalla s'ha reduït aquest volum per tal d'agilitzar la sortida.

Ja sabem que la diferència entre el Gaussià i mediana respecte el Anisotropic és que els dos primer són filtres en els quals es convoluciona la imatge amb una màscara i l'últim és iteratiu el que comporta que com més iteracions hi han més lent és el

procés. Els dos primers la gran diferencia és en el tipus d'operacions que es realitzen alhora de dur a terme la convolució, el primer simplement a través de la funció amb σ escollida calcula la màscara i fa els càlculs per cada píxel el que no comporta un gran cost, en canvi el de la mediana, per la filosofia del filtre ha d'ordenar les dades i això en temps algorísmic té un cost elevat, el que provoca que el temps en obtenir resultat en el filtre de la mediana sigui molt més elevat que en el Gaussià.

5.2 Segmentació

Hem d'analitzar el mètode estudiat en la part principal del projecte, com ja sabem el mètode ITK inicialment està preparat per segmentar *slices* però en el PFC s'ha fet un tractament d'aquests slices per tal de crear models estàndards i ajustar els resultats a les imatges originals.

També com ja hem comentat un dels grans inconvenients alhora de fer proves i extreure conclusions sobre el mètode és la manca d'un número elevat d'imatges de prova, per poder generar models a partir de diversos pacients diferents, tot i això intentarem mostrar quines són les caràcterístiques del mètode per tal de millorar-lo en treballs futurs.

Una de les conseqüències d'aquesta limitació en les dades en alguns casos segmentem volums de pròstates que formen part del train. Aquestes limitacions, però no afecten al desenvolupament i la implementació del projecte si no que formen part del treball futur d'avaluació del sistema amb un conjunt de dades més gran. De totes maneres, en aquest apartat es mostraran també resultats d'utilitzar diferents tipus de pròstates (pacients reals, i phantoms).

Sobre els paràmetres de configuració, en el cas training trobem la tolerància alhora de buscar contorns. La diferència quan augmentem la tolerància resideix en els punts característics amb el que es genera la forma mitja que seran els que després anirà evolucionant per fer el *search*.

5.2.1 Segmentacions 2D

Per tal d'estructurar els resultats comencem fent segmentacions de tipus 2D, en aquestes tenim l'avantatge que escollim les imatges de training, veiem doncs a la figura 5.9 la segmentació una slice de phantom, un d'MRI real i un d'US. El paràmetre de tolerància al fer l'entrenament ha estat 2.5, mentre que els paràmetres del search per fer les proves inicials s'han utilitzat amb un valor de 3 de *length of profile* i número d'iteracions igual a 2 (són els valors estables del mètode). Amb la imatge phantom s'han obtingut un total de 12 punts característics que es mostren a la figura 5.10. Els punts obtinguts en la figura MRI real han sigut 19, per tal de dur a terme aquests entrenaments s'ha adoptat la mesura d'utilitzar totes les imatges del

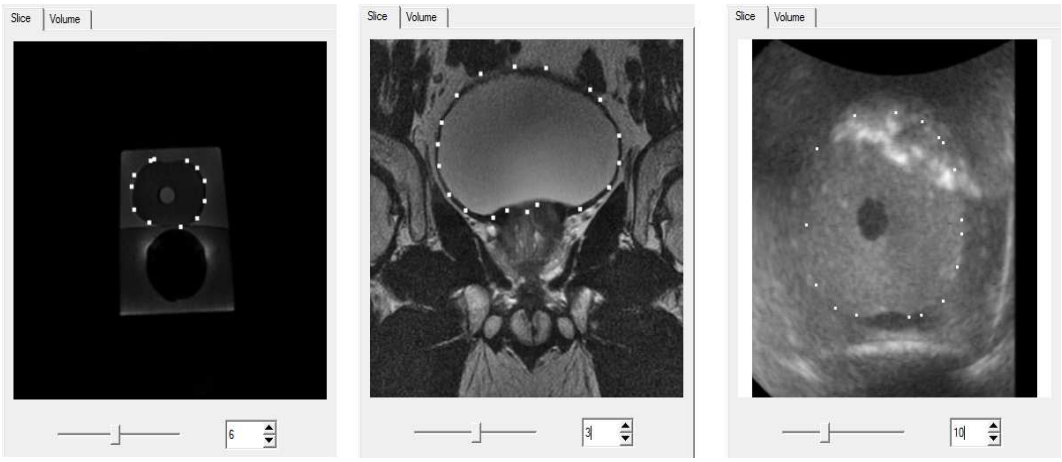

Figura 5.9: Segmentació d'imatges : Phantom MRI - MRI pacient real - Ultrasons

```
  none
  Valid : 1
  Mean Shape : 102.375 78.1042 89.375 88.1042 87.375 96.1042 89.375 112.104 101.
375 121.104 126.375 124.104 139.375 118.104 145.375 106.104 145.375 92.1042 139.
375 83.1042 131.375 78.1042 105.375 77.1042
  Eigen Vectors:
```

Figura 5.10: Exemple de sortida del mean shape d'un phantom MRI

volum de la pròstata excepte la que posteriorment es vol segmentar. I per últim els punts obtinguts en la segmentació dels ultrasons han sigut 17. Com es pot veure la segmentació sobre MRI és molt més clara degut a que els contorns de la imatge estan molt més definits.

Tot seguit durem a terme una segmentació mostrant quines imatges s'han utilitzat per fer l'entrenament, també ho farem sobre imatges MRI reals ja que de les imatges que disposem són les que tenen més varietat de forma i on podem apreciar si els resultats són correctes. A la figura 5.11 amb paràmetres de configuració tolerància = 3, *length* = 3, iteracions = 2.

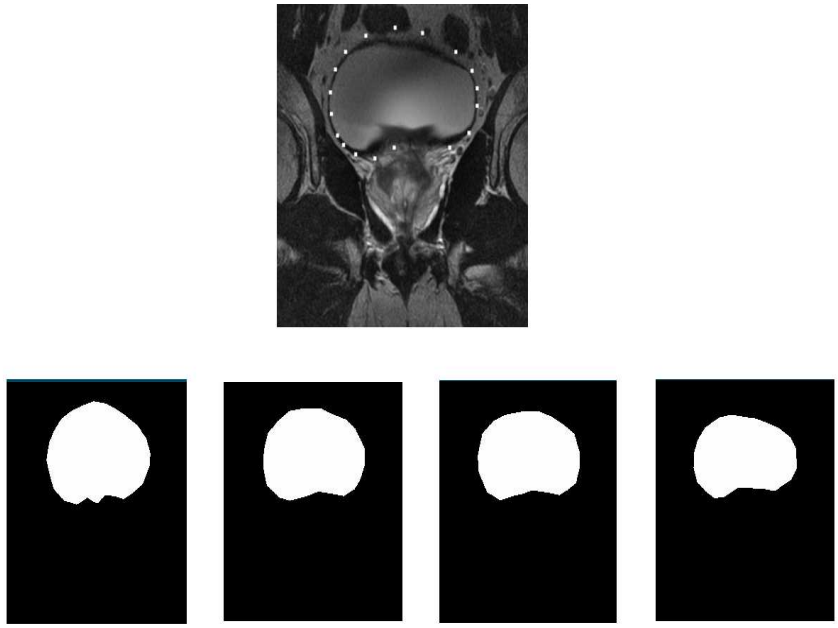

Figura 5.11: Segmentació MRI (dalt) amb les imatges de training (baix)

Realitzant proves podem valorar quins efectes tenen els paràmetres, la tolerància al training ja ho hem explicat anteriorment quants píxels es tindran en compte quan es troba el contorn analitzar el voltant. En el cas del *length* si el fem augmentar els *landmarks* es perden ja que li estem indicant que el contorn de la imatge és molt ampli i alhora d'analitzar-ho en les nostres en realitat és petit.

Per últim el número d'iteracions del mètode és el paràmetre que menys s'ajusta a la seva funció, si anem modificant aquest paràmetre la lògica ens diu que cada vegada s'hauria d'ajustar més al contorn, però en fer-ho obtenim resultats com els de la figura 5.12. Que les primeres iteracions s'adaptin a la forma és fàcil d'entendre les imatges d'ultrasons d'entrenament són del mateix estil de la que hem utilitzat per segmentar llavors la forma mitjana obtinguda és molt semblant en dimensions a la real llavors de primeres la segmentació és correcte però en arribar a fer 4 iteracions ja no és capaç de distingir contorns i la figura no s'adapta a de la posició real de la pròstata.

Amb aquest exemple de segmentació d'ultrasons de la figura 5.12 hem vist quina és l'altre manera de visualització de la segmentació on es genera el perfil total de la pròstata segmentada obtinguda fent la unió entre els punts. Aquest mètode utilitza el mètode bresenham ([6]) per fer les línies que uneixen els punts com ja havíem vist en el capítol d'anàlisi i disseny (4).

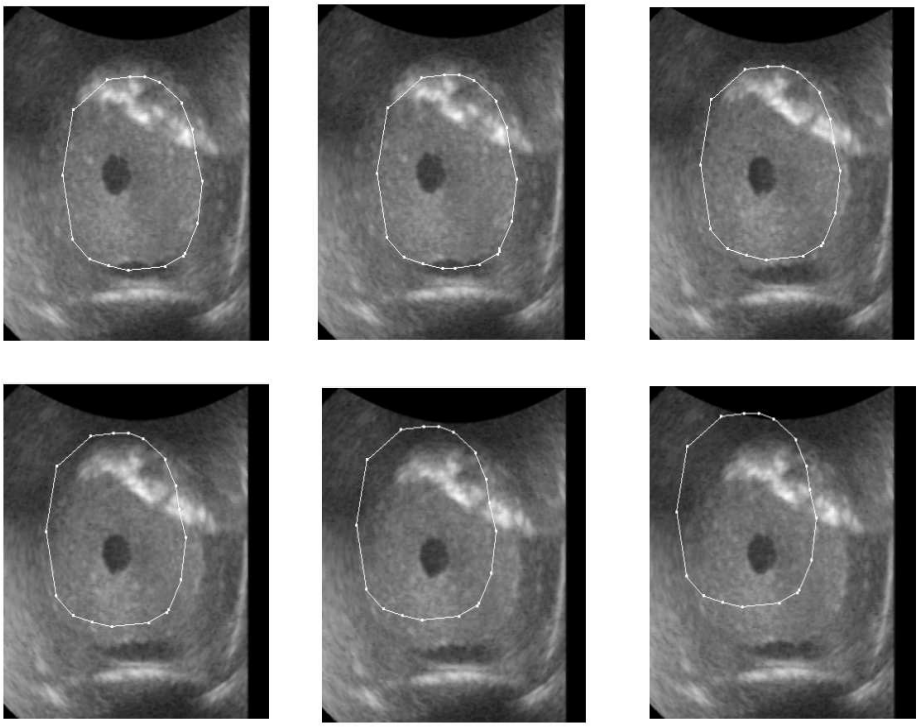

Figura 5.12: Evolució del paràmetre iteracions en segmentació d'ultrasons

5.2.2 Errors de cerca

Tot i aquestes imatges de segmentacions correctes que hem trobat no sempre les segmentacions són tan ajustades, a vegades el mètode es perd alhora de fer el *search* com en el cas que mostra la figura A.2 on el *mean shape* retorna 31 punts significatius però alhora de fer la cerca com que no troba el contorn no es capaç de fixar un punt de real del contorn i dels 31 inicials que tenia el *mean shape* es pinten només 17, i com es mostra gairebé la meitat de la pròstata ha quedat sense segmentar.

5.2.3 Segmentació 3D

Vistos els paràmetres veiem ara algun exemple de les segmentacions automàtiques de volum. Per realitzar aquesta tasca ja s'ha explicat anteriorment la metodologia a seguir, i s'han creat 3 models, els quals s'entrenaran amb els paràmetres estables del mètode.

Quan estem segmentant volums de pròstates es pot calcular el volum total de la segmentació a través de l'opció del menú segmentació>extraure pròstata. La manera en que es realitza aquest càlcul és senzilla, una vegada s'ha segmentat la imatge ja hem vist que es poden visualitzar els resultats de dos maneres diferents, una per

Figura 5.13: Error del search en una segmentació d'US

punts o l'altre per perfil. Així doncs quan es crea aquest perfil (marcant línies amb el mètode bresenham) no estem fent altre cosa que crear una regió tancada. Gràcies a la col·laboració de l'A.Gubern [7] amb aquesta regió es pot crear una màscara, és a dir, es llença una llavor dintre de la regió i es fa créixer fins a trobar els píxels blancs del contorn definit. Aquest procediment de creixement rep el nom de *region growing*. La màscara consisteix en una imatge binària, píxels blancs o negres, on es diferencien dues regions, la que conforma la pròstata i la resta, així doncs, fàcilment podrem calcular el número total de píxels que té el resultat de segmentació de la pròstata, fent que obtenint l'*spacing* de la imatge (mides reals en mm de cada píxel) es pot calcular fàcilment el volum d'un slice i fer un sumatori de tots els slices que conformen la pròstata obtenint el volum total.

El resultat d'aquest càlcul es mostra al quadre de text del visual, a part genera un fitxer del tipus vtk que conté la màscara que s'ha creat per calcular el volum, és a dir, una imatge binària on es distingeix entre pròstata i la resta de la imatge. A la figura 5.14 tenim els resultats de la segmentació automàtica MRI amb alguns dels slice resultants així com el volum calculat de la pròstata que en aquest cas és d'aproximadament 33 cc que es troba entre els valors reals (20cc - 50cc) de la pròstata. En el cas del phantom els resultats obtinguts han estat els de la figura 5.15, que com veiem no s'ajusten a l'slice, les formes mitjanes calculades són correctes però alhora d'associar quin model s'ha d'utilitzar amb cada slice no esta relacionat correctament. Això és perquè quan es fa una segmentació de volum es creen 3 models i per exemple, als 3 primers slice s'utlitza el primer model als 5 slices centrals el segon model i el tercer model pels 2 últims talls, això és així degut a que la pròstata no té les mateixes dimensions i llavors no s'han de cercar amb els mateixos models. Aquesta correlació és un dels punts a millorar quan es tinguin volums sencers per poder fer una estimació correcte.

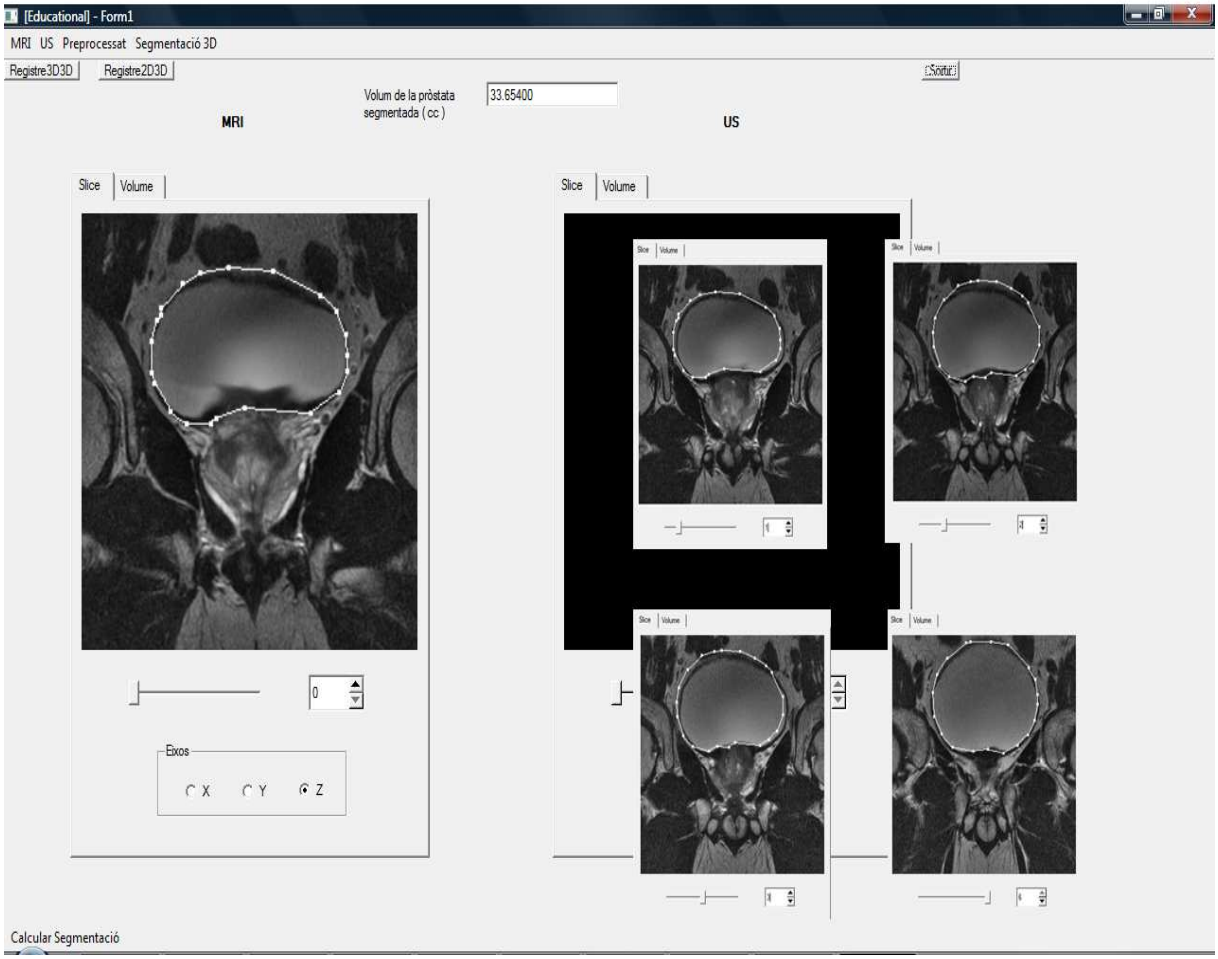

Figura 5.14: Segmentació volum 3D MRI

Per últim el volum de pròstata d'ultrasons per fer proves del que disposem conté 30 imatges (slices) però la majoria són molt semblants així que s'han triat 6 a segmentar per tal d'agilitzar el resultat. El resultat el trobem a la figura 5.16 on s'aprecia que com que no es controlen els paràmetres de manera tan exhaustiva com quan hem segmentat slice per slice, si hi sumem el fet que la qualitat de la imatge és baixa el mètode es perd quan fa el search.

Com ja hem dit quan es calcula el volum de la pròstata es crea una màscara binària que té un aspecte com el de la figura 5.17, la qual correspon a la segmentació de la imatge MRI de la figura 5.14

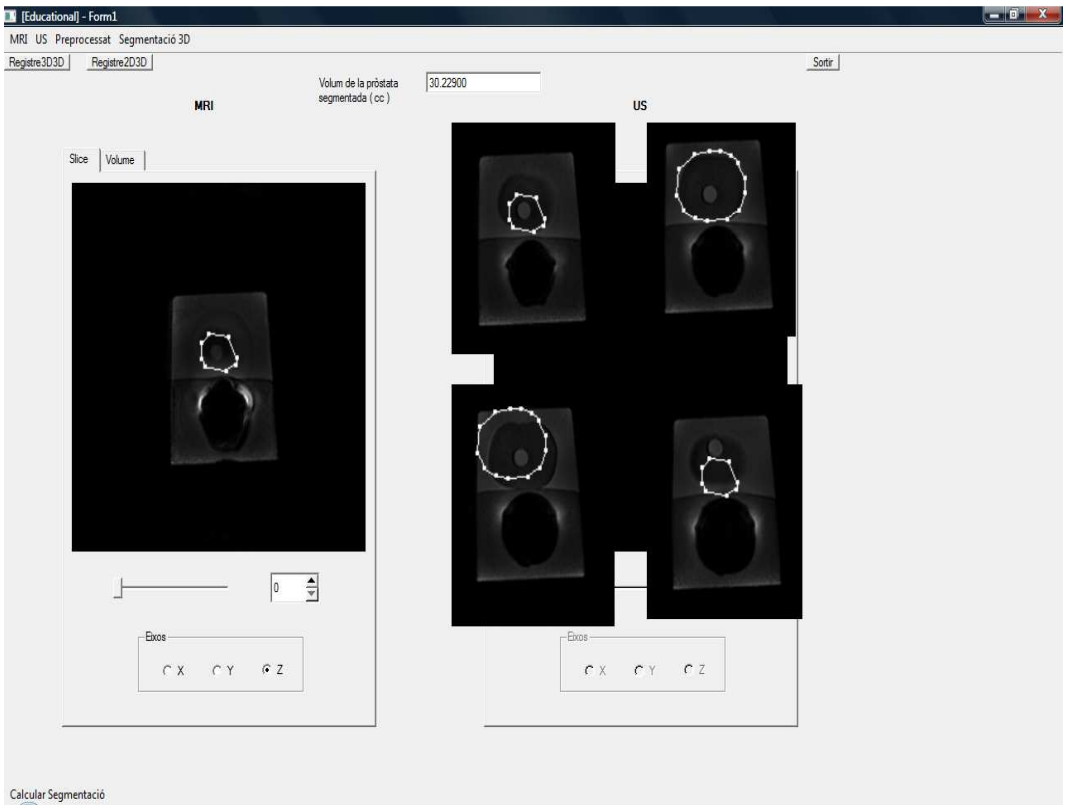

Figura 5.15: Segmentació volum 3D Phantom

Figura 5.16: Segmentació volum 3D ultrasons

Figura 5.17: Màscara binària d'una imatge MRI de pròstata

Capítol 6

Conclusions i treballs futurs

6.1 Conclusions

Arribem al final del PFC i és moment de valorar el treball realitzat, fins on s'ha arribat i quins són els treballs futurs per continuar el treball que ha iniciat aquest projecte.

Al inici teníem uns objectius fixats, per una part s'havia de crear un mòdul capaç de realitzar tasques de filtratge per tal d'eliminar el soroll en imatges mèdiques del tipus ultrasons, aquest tipus d'imatges contenen molt de soroll degut al tipus d'adquisició. S'han estudiat i implementat tres filtres amb característiques diferents per abordar les diferents necessitats, eliminar el màxim de soroll, mantenir els contorns, etc..

El segon objectiu era segmentar imatges de pròstata a través del mètode Active Shape Models, ITK constava de dos classes que duien a terme els càlculs propis de la teoria del l'Active Shape Model, s'havia d'estudiar la utilització d'aquestes classes per veure si s'obtenien resultats desitjables en la tasca de segmentació. Tenint en compte els objectius inicials creiem que els objectius s'han complert ja que s'han desenvolupat dos mòduls de processament d'imatges que són capaços de treballar de manera independent i generar resultats als objectius proposats.

Posteriorment i aprofitant la existència d'una aplicació per tractar imatges mèdiques s'han afegit aquests dos blocs per tal de testejar-los i analitzar.

Els resultats obtinguts en el mòdul dels filtres són correctes ja que depenen simplement de variables les quals generen uns càlculs que no comporten més complicació.

En el segon bloc s'han tingut més problemes, alhora d'iniciar el projecte es creia que les classes ITK estaven preparades per treballar amb volums 3D per tal de realitzar segmentacions no només d'slice en slice sinó directament de tota la pròstata, aquest fet ha fet variar una mica el plantejament inicial en el qual es començava directament segmentant 3D i després adaptant si feia falta el mètode als nostres

resultats ideals. Al no treballar exactament així es va haver de començar segmentant imatges 2D per provar el mètode i adaptar-ho a les necessitats, després s'ha estudiat la manera de fer-ho en 3D però la temporalitat ha fet que s'hagi plantejat la solució adoptada al projecte, no tractar directament el volum 3D sinó aprofitant que el mètode 2D funcionava tractar el 3D com conjunt d'imatges 2D.

A part dels objectius pròpiament especificats del projecte a mesura que s'anava implementant el codi font s'han hagut d'adquirir coneixements sobre les diferents eines utilitzades ITK, VTK, QT, l'editor Latex i les diferents aplicacions existents de segmentació, la qual cosa ha fet adquirir coneixement s'obre l'estructuració de projectes i serà útil en la realització de futures tasques ja sigui amb projectes similars o altres tipus de programació. També s'ha estudiat la teoria que permet generar els Active Shape Models per tal d'entendre que feia el mètode en cada moment ja que alhora de fer possibles canvis en l'estructura serà necessari el coneixement del funcionament.

En definitiva el projecte ha sigut el punt l'inici que ha servit per adquirir experiència en la realització de projectes informàtics que necessitan d'una planificació per tal de d'assolir uns objectius concrets com era el cas del PFC. No finalitzar sense agraïr l'ajuda rebuda per part del tutor del projecte i de la persona que va iniciar el projecte del visual i que esta treballant també en la recerca de la segmetnació de pròstata.

6.2 Treballs futurs

Al llarg del projecte ens hem anat adonant de les mancances dels mètodes utilitzats així com vies de millora a partir d'on s'ha finalitzat el PFC.

- De la part de filtratge per tal d'ajudar en la millora de les imatges US s'hauria de buscar mètodes o combinar els exitents per tal de produir resultats en la detecció de contorns, que facilités la tasca del *search* alhora de segmentar.
- La primera millora que s'hauria de realitzar ha d'estar enfocada als errors que el search de la classe ITK pugui tenir, modificar el codi font de la classe *itkActiveShapeModelGradientSearchMethod.h"* ja que com hem vist alhora d'augmentar els paràmetres no es comporta com s'esperava.
- Respecte a la segmentació 3D, si es vol treballar amb els Active Shape Models s'hauria d'idear la manera de que el mètode en comptes d'agafar un volum 3D que està format per diferents slices, pogues agafar inicialment diversos volums 3D de diferents pròstates, una vegada tiguès això s'haurien de generar un model estàndard de pròstata que servís com a forma mitjana, en definitiva, que el mètode realitzès el mateix treball però directament treballant amb 3D.

- El tema de l'incorporació a aplicacions gràfiques existents, aquí s'ha aprofitat una aplicació en la qual s'ha estat treballant però el fet que sigui independent podria fer que es provessin les classes en alguna altre aplicació per tal d'aprofitar la seva portabilitat.

- Per últim com ja s'ha esmentat és la necessitat de generar una base de dades amb diferents entrades de pròstata, és díficil ja que requereix de col·laboracions externes, ja que això és de l'àmbit informàtic, però ja que es treballa amb objectius aprofitables en l'àmbit mèdic s'hauria de poder realitzar proves per testejar el correcte funionament del mètodes, sinó el problema es que es limita molt ja que, quan s'han realitzat 50 proves del mètode amb les mateixes imatges realment no saps fins a quin punt els resultats són extrapolable o prou generals.

Apèndix A

Estructura de classes aplicació Visual

A l'apartat de anàlisi s'ha mostrat l'estructura de les classes implementades en aquest projecte, amb els objectes ITK que contenien cadascuna d'elles així com els QT creats. Tot així que finalment a esdevingut part de l'aplicació Visual, per tant és necessari conèixer també l'estructura de les seves classes.

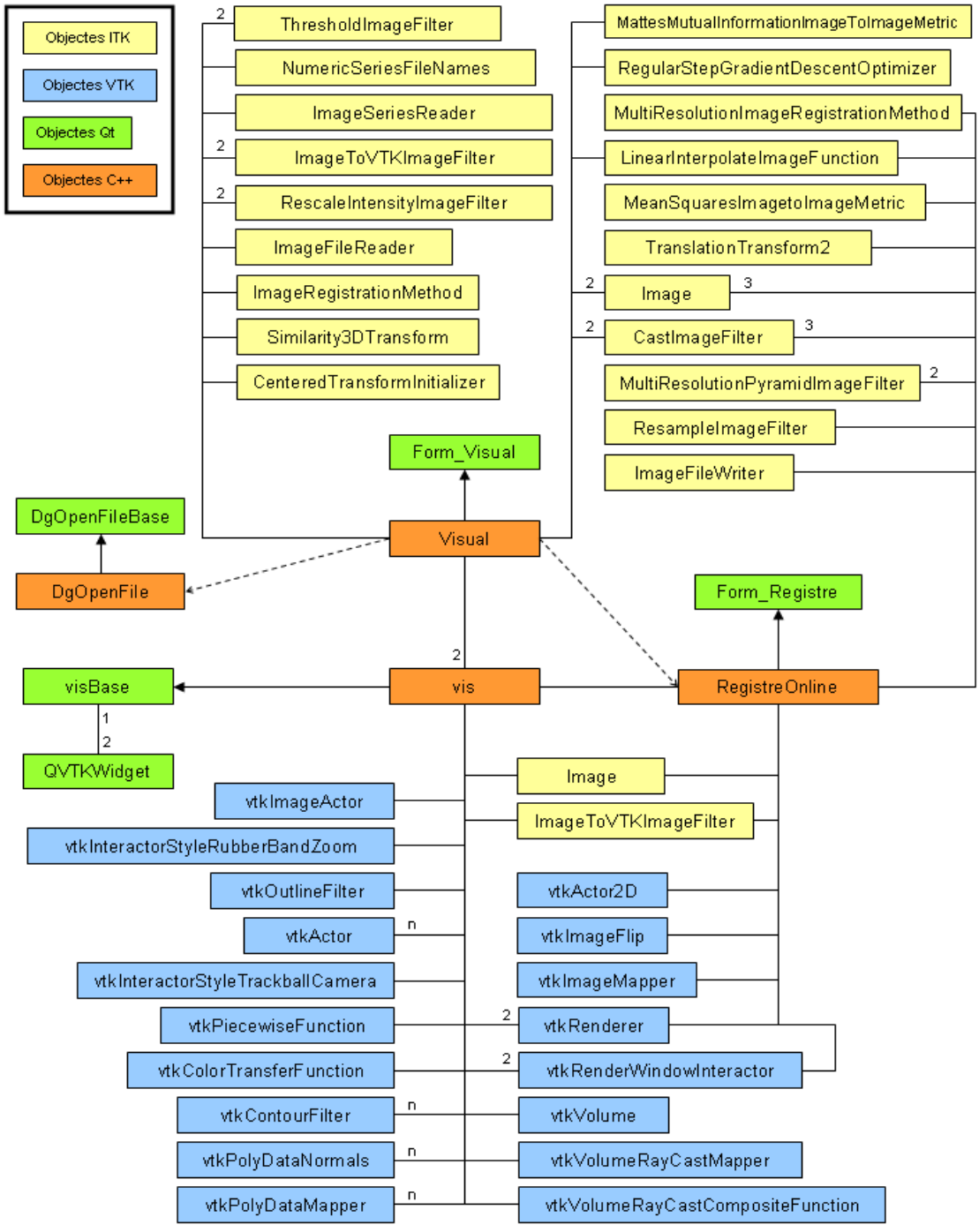

Figura A.1: Diagrama de classes del Visual

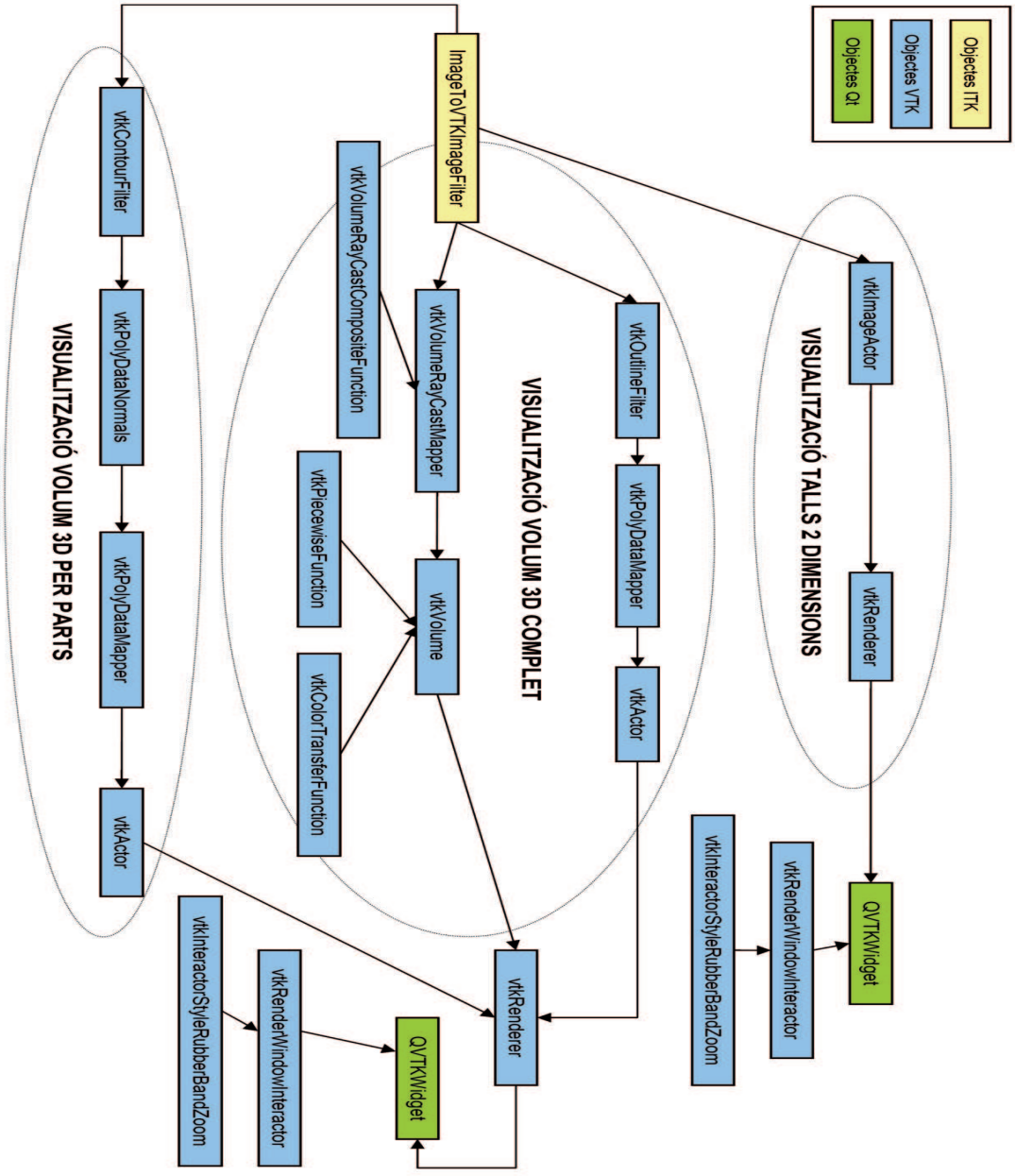

Figura A.2: Relació d'objectes Visual

Bibliografia

[1] Luis Ibáñez, Will Schroeder, Lydia Ng, Josh Cates and the *Insight Software Consortium*: *ITK Software Guide Second Edition* Updated for ITK version 2.4. November 21, 2005

[2] P. Perona, J. Malik *Scale-space and edge detection using anisotropic diffusion Pattern Analysis and Machine Intelligence* Juliol 1990

[3] *ITK- SNAP Segmentation Toolkit website.* http://www.itksnap.org/ (Juny 2008)

[4] T.F.Cootes, A.Hill, C.J.Taylor, J.Haslam: *The Use of Active Shape Models for Locating Structures in MedicalImages* Image and Vision Computing Vol.12, No.6 . July 1994

[5] T.F. Cootes, D. Cooper, C.J. Taylor and J. Graham: *Active Shape Models - Their Training and Application* Computer Vision and Image Understanding. January 1995

[6] Jack E. Bresenham: *"Algorithm for computer control of a digital plotter"* IBM Systems Journal, Vol. 4, No.1, January 1965

[7] A.Gubern *"Eina de recerca per la correspondència entre ecografia i ressonància magnètica per el guiatge en la biòpsia de pròstata"* PFC Setembre 2007