EPS
Escola Politècnica Superior
UdG

Projecte/Treball Fi de Carrera

Estudi: Enginyeria Informàtica. Pla 1997

Títol: Automatització del procés de dissecció per la determinació de la qualitat de la carn de porc.

Document: Memòria

Alumne: Josep Serra i Busquet

Director/Tutor: Imma Boada
Departament: Informàtica i Matemàtica Aplicada
Àrea: LSI

Convocatòria (mes/any): Gener - 2007

Índex

Introducció i Objectius

1.1.- INTRODUCCIÓ

Un dels aspectes més importants en el procés de tractament del porc és com determinar la seva qualitat. En la Unió Europea, la metodologia utilitzada per determinar-la és la que s'anomena **dissecció**. La dissecció utilitza el contingut de magre en la carn com a referència comú per determinar la qualitat de la canal. Aquest procés és totalment manual per la qual cosa resulta laboriós i costós, tant pel que fa referència al temps que implica com per la seva depreciació.

La solució a aquests problemes es podria obtenir a partir de l'automatització de la dissecció. Per això seria necessari: (i) obtenir les dades del porc mitjançant tècniques d'adquisició no invasives, com la Tomografia Computeritzada (TC) i (ii) disposar de tècniques de segmentació i visualització d'imatge, que permetessin processar les dades per tal de determinar el contingut de magre en la carn amb una precisió similar a la dissecció.

D'aquesta manera, podríem conèixer la capacitat predictiva de la TC i la bondat d'ajust de les equacions de predicció que s'obtenen de mesures tals com la dissecció, i que al mateix temps s'utilitzen per calibrar equips que ens ajuden a quantificar la composició volumètrica de canals porcins i les seves peces.

L'objectiu d'aquest projecte és crear un entorn informàtic que integri les eines bàsiques que permetin automatitzar al màxim aquest procés de determinació de la qualitat porcina. El projecte forma part del desenvolupament d'una plataforma d'automatització de la dissecció. Concretament es centra en la implementació de tècniques de segmentació i visualització que permetin processar les dades porcines i determinar així la quantitat de grassa que conté.

1.2.- OBJECTIUS

L'objectiu principal d'aquest projecte és estudiar, dissenyar i implementar un entorn informàtic que integri les eines bàsiques que permetin automatitzar al màxim el procés de dissecció porcina. Per poder assolir aquest objectiu serà necessari:

- Estudiar el funcionament del procés de dissecció porcina per així determinar els diferents paràmetres que hi entren en joc.

- Interpretar les dades que s'obtenen dels dispositius de captació. En el nostre cas com a dispositiu de captació de dades s'utilitzarà la Tomografia Computeritzada.

- Definir un model de representació de les dades que permeti el seu processament informàtic. Com veurem més endavant, el model utilitzat per a la representació de dades serà el model de voxels.

- Estudiar, implementar i avaluar diferents tècniques de segmentació per veure quina s'adapta millor a les nostres necessitats. Les tècniques de segmentació ens permetran dividir o separar imatges en un conjunt d'àrees connectades anomenades regions.

- Estudiar i implementar tècniques de visualització 2D i 3D per facilitar la interpretació dels resultats.

- Desenvolupar els mòduls necessaris que ens permetin obtenir dades numèriques de les diferents estructures que formen el porc. D'aquesta manera, ens serà possible obtenir el volum i les àrees de regions concretes.

Aquest projecte forma part d'una plataforma de manipulació de dades porcines desenvolupada en el Grup d'Informàtica Gràfica de la Universitat de Girona. Així doncs, un altre dels objectius és implementar el projecte d'una manera modular i robusta de tal manera que faciliti la integració a aquesta plataforma. Es pretén implementar un entorn amigable i fàcil d'utilitzar, d'acord amb els requeriments de la plataforma. Cal tenir en compte que els usuaris finals de la plataforma no seran informàtics.

Els llenguatges de programació utilitzats per implementar l'aplicació són: ITK, VTK, OpenGL i Qt. En els quatre casos es tracta de llenguatges de lliure distribució i multiplataforma. Així doncs, també es compleix un dels objectius de l'aplicació final, de manera que es pugui executar sota diferents plataformes.

1.3.- METODOLOGIA DE TREBALL

La metodologia de treball utilitzada per a la realització d'aquest projecte ha estat la Extremme Programming (XP). Desenvolupar el projecte utilitzant aquesta metodologia vol dir dur a terme un procés de disseny, implementació i testing iteratiu i simultani. Així doncs, els diagrames de l'anàlisi i el disseny de l'aplicació, s'han construit i modificat a mesura que evolucionava el codi.

Aquesta metodologia ens defineix una sèrie de regles de construcció a partir d'uns requeriments ben definits al inici del projecte. Els canvis d'aquests requeriments poden variar al llarg del desenvolupament del projecte.

XP consta d'una sèrie de senzilles regles a seguir per obtenir un software fàcilment adaptable a entorns evolutius i millora la reutilització.

XP es recolza en el treball en equip. El projecte que he desenvolupat ha estat integrat en una plataforma on treballen diverses persones. La cohesió i el treball en equip de tots els membres que hi estan involucrats, proporciona sens dubte un millor resultat i una coherència més alta de l'aplicació final.

XP defineix un entorn senzill, ben definit, clar i concís, que acaba generant un software de qualitat construït en el temps més reduït.

Les fases de desenvolupament de la metodologia XP que he seguit són:

- Planificació.
- Disseny.
- Construcció.
- Proves.
- Manteniment.

1.3.1.- Planificació

A l'etapa de planificació marcarem els objectius del projecte que ens fixaran els requeriments del sistema. Aquesta fase de planificació pot variar al llarg del projecte. De fet, ens hi hem trobat diverses vegades en aquest projecte. Certs apartat que ja havíem implementat i integrat a la plataforma, veiem més endavant que era millor plantejar-los d'una manera diferent.

Així doncs, el pla no és estricte. Conforme els requeriments varien el pla ha de variar. Aquest sistema permet l'adaptació del projecte a un entorn canviant (principal filosofia XP).

Per realitzar aquesta planificació hem fixat reunions periòdiques amb els tutors del projecte. Aquestes reunions ens servien per fixar nous objectius i nous terminis d'entrega de certes parts del projecte.

1.3.2.- Disseny

En aquesta fase hem definit un conjunt de regles que es centren principalment en elaborar un disseny clar i senzill.

Aquest projecte ha estat integrat en una plataforma conjunta. Utilitzar una nomenclatura clara i entenedora de les classes i mètodes a crear perquè tothom entengui ràpidament a que fa referència el disseny a desenvolupar, pot facilitar molt el treball de tothom.

També hem intentat escollir un sistema de noms per els objectes que tothom pogués relacionar amb la part de segmentació i visualització i evitar els noms abstractes que fan més difícil l'enteniment del sistema global.

En aquest apartat de disseny també hem intentat eliminar redundàncies i funcionalitats no útils. Volem la màxima rapidesa possible dels mètodes implementats.

1.3.3.- Construcció

En la fase de construcció implementarem el nostres mòduls (segmentació i visualització), i els integrarem a la plataforma global. Aquesta plataforma disposa d'una sèrie de normes de programació i de regles a respectar per a poder afegir noves funcionalitats a l'aplicació. D'aquesta manera es pretén aconseguir una major cohesió i facilitat de comprensió de la plataforma.

1.3.4.- Proves

El codi implementat passa una fase de proves en cada iteració. En el nostre cas, en aquesta fase de proves hi participaven molt activament tots els tutors i persones que treballen també en la construcció de la plataforma conjunta. Tots ells ens marcaven els objectius a complir i es verificava si s'havien assolit correctament.

Cada vegada que hi havia algun apartat finalitzat havia de passar aquestes proves d'acceptació abans no s'integrava definitivament a la plataforma.

1.3.5.- Manteniment

La última fase d'aquesta metodologia de treball és l'apartat de manteniment. Aquesta fase encara no s'ha produït en la implementació d'aquest projecte ja que la plataforma conjunta encara no està en ús. S'espera que en un curt període de temps ja estigui disponible.

1.4.- ESTRUCTURA DEL DOCUMENT

Seguint aquestes etapes que acabem d'esmentar, hem estructurat aquest document en els capítols següents:

Capítol 2. La qualitat de la carn porcina.

En aquest capítol presentarem els diferents sistemes que s'utilitzen en l'actualitat per determinar la qualitat de la carn porcina. Veurem els avantatges i inconvenients de cada sistema, i finalment proposarem una solució per reduir aquests problemes.

Capítol 3. Tomografia Computeritzada per determinar la qualitat de la carn porcina.

En aquest capítol parlarem sobre com utilitzar la Tomografia Computeritzada per determinar la qualitat de la carn porcina. Veurem les diferents etapes que cal seguir per arribar aquest objectiu. Primer veurem com s'adquireixen les dades i el model de representació que s'usa per poder tractar-les informàticament. A continuació explicarem el funcionament de les tècniques de segmentació i visualització que ens permetran obtenir el resultat final i determinar així la qualitat de la carn.

Capítol 4. Implementació.

Aquest capítol el dividirem en 3 parts per explicar cada una de les etapes de la implementació.

Visió general de la plataforma

En aquest apartat donarem una visió general de la plataforma de la que partíem per implementar aquest projecte. Veurem els serveis bàsics que oferia. Per últim, explicarem breument les llibreries utilitzades per realitzar la implementació.

Requeriments del sistema

En aquest apartat descriurem els requeriments del sistema. Aquests requeriments ens definiran els objectius de l'aplicació juntament amb les funcionalitats que l'usuari pot realitzar. Per especificar aquestes funcionalitats ens ajudarem dels diagrames de cas d'ús i les fitxes de cas d'ús. Aquest apartat ens ha de permetre entendre els elements que composen el sistema informàtic que hem implementat.

Disseny del sistema

En aquest apartat descriurem la fase de disseny o modelatge conceptual, que ens servirà per explicar el disseny del sistema informàtic, és a dir, la nova estructura que hem construït per ampliar, amb les noves funcionalitats definides anteriorment, l'aplicació de la qual partim. Per fer aquesta descripció utilitzarem els diagrames de classe, d'activitat i de seqüència.

Capítol 5. Interfície gràfica.

En aquest capítol explicarem el funcionament de la interfície gràfica amb la que interactua l'usuari. Mitjançant aquesta interfície gràfica, l'usuari serà capaç entre altres coses, d'obrir un model porcí, executar una tècnica de segmentació, o visualitzar els resultats obtinguts.

Capítol 6. Resultats obtinguts.

En aquest capítol veurem els resultats que hem obtingut utilitzant les tècniques de segmentació i visualització implementades.

Capítol 7. Avaluació de resultats.

En aquest capítol compararem les diferents tècniques de segmentació implementades i realitzarem una primera avaluació per veure quina tècnica ens ofereix millors resultats per deduir el percentatge de magre en la carn en el models porcins.

Millores i treball futur.

En aquest capítol comentarem les possibles millores de les tècniques i funcionalitats implementades. També comentarem altres aspectes que ens interessaria desenvolupar en un futur per millorar les prestacions de la plataforma.

Conclusions.

En aquest capítol repassarem els objectius que ens havíem marcat al inici del projecte i comprovarem si els hem assolit correctament. També explicarem les conclusions més importants que hem arribat després de desenvolupar el treball.

La qualitat de la carn porcina

Un dels aspectes més importants en el procés de tractament del porc és com determinar la seva qualitat. En la Unió Europea, la metodologia utilitzada per determinar-la és la que s'anomena la dissecció. La dissecció utilitza el contingut de magre en la carn com a referència comú per determinar la qualitat de la canal. La canal és el cos d'un porc sacrificat, sagnat, sense vísceres, sense llengua, pèl, unglots, òrgans genitals, greix, ronyons ni diafragma. Desafortunadament aquesta metodologia necessita d'un procés molt laboriós i costós tant per el temps que implica com per la seva depreciació.

Una possible forma de resoldre els problemes que planteja el sistema actual és utilitzar una tècnica d'adquisició no invasiva, la Tomografia Computeritzada (TC). La TC ens permet obtenir informació del porc i representar-la en forma d'imatges. Sobre aquestes imatges es poden aplicar tècniques d'anàlisi i visualització d'imatge que ens ajudin a determinar el contingut de magre a la carn amb una precisió similar a la dissecció. Aquestes tècniques a més d'accelerar i automatitzar el procés que cal seguir per determinar la qualitat del porc, també podrien usar-se per mesurar la bondat d'ajust de les equacions de predicció que s'obtenen de mesures tals com la dissecció, i que al mateix temps s'utilitzen per calibrar equips que ens ajuden a quantificar la composició volumètrica de canals porcins i les seves peces.

2.1.- CLASSIFICACIÓ DE CANALS

El propòsit fonamental de la classificació de les canals porcines ha estat estructurar el sistema comercial entre productors i industrials de la carn mitjançant un mecanisme de regulació de preus. Aquest propòsit ha conduit a que administracions europees promulguin unes normes generals que garanteixin una classificació uniforme de les canals, a fi de poder garantir un pagament equitatiu no només per productors, sinó també per compradors.

El criteri adoptat per la classificació de les canals es basa en el contingut (percentatge) en carn magre de la canal que s'obté a partir de mesures objectives de una o diverses parts anatòmiques de la canal del porc. Aquest criteri determina en gran mesura el rendiment que es pot obtenir. El mètode d'aplicació es descriu en Reglament (CEE) 2967/85 (modificat per el Reglament

(CE) 3127/94) en el que es defineixen els criteris estadístics que han de complir els mètodes de classificació, així com la fórmula oficial per el càlcul del percentatge de magre d'una canal.

L'objectiu principal és establir la composició de la canal en percentatge de magre amb un error inferior al 2.5%. Cal definir exactament la presentació i les condicions a seguir en el moment de pesar les canals, de tal manera que els resultats que s'obtinguin es puguin comparar i permetin classificar la canal segons el contingut de magre per la seva posterior comercialització. Els principis en els que es basa la classificació d'una canal són:

- **Obligació:** Des de l'1 de gener de 1989 és obligat realitzar la classificació de les canals.

- **Presentació estàndard de la canal:** Per normativa es defineix a una canal com: el cos d'un porc sacrificat, sagnat, sense vísceres, sense llengua, pèl, unglots, òrgans genitals, greix, ronyons ni diafragma. En el cas del sistema de classificació porcina, es treballa amb mitges canals esquerres.

Figura 2.1: Presentació d'una canal

- **Marcatge de la canal:** Una vegada realitzada la classificació de les canals, aquestes s'han d'identificar segons la nomenclatura SEUROP o segons el seu percentatge de magre.

Percentatge de magre	Classe comercial
60 i més	S
55 fins a menys de 60	E
50 fins a menys de 55	U
45 fins a menys de 50	R
40 fins a menys de 45	O
menys de 40	P

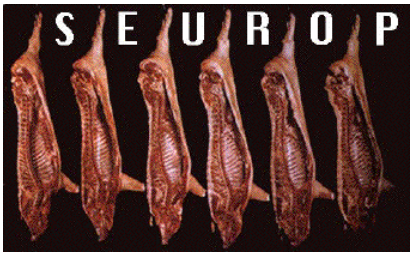


Figura 2.2: Classes comercials SEUROP.

- **Utilització de mesures objectives per predir el contingut de magre de la canal:** El contingut de carn magre només es podrà mesurar mitjançant equips autoritzats i que hagin estat estadísticament aprovats per el "Pig Meat Managenent Comitee" (PMMC) – DG VI Brussel·les.

2.2.- SISTEMES I EQUIPS DE CLASSIFICACIÓ ACTUALS

En la mesura que l'apreciació visual de la conformació d'una canal va deixar de ser un criteri per la seva classificació, hi ha hagut un considerable desenvolupament d'equips i tècniques que han fet possible una estimació més exacte del percentatge de magre en la canal. Les tècniques disponibles varien en sofisticació des d'una simple regla per a mesurar l'espessor de la grassa en un punt de referència, passant per sondes manuals i de registre automàtic per mesurar grassa i músculs, fins a sondes robòtiques i generadores d'imatge i de vídeo.

Actualment es classifiquen les canals a partir de mesures objectives d'una o diverses parts anatòmiques de la canal. Aquestes mesures es realitzen utilitzant equips amb captura automàtica de la informació. A Espanya els equips autoritzats són els que es detallen a continuació:

2.2.1.- FAT-O-MEAT'ER

El Fat-O-meater (FOM) (figura de continuació) és un equip de classificació semi objectiva, fabricat per l'empresa danesa SFK Technology A/S. El FOM es basa en la reflectància. Utilitza la seva punta òptica per mesurar la diferència en la llum reflectida de la grassa i la profunditat del múscul en diferents punts de la canal.

A Espanya, la mesura de l'espessor de grassa (g34fom) i múscul (m34fom) es realitza introduint la punta òptica a 6 cm de la línia mitja i entre la 3º i 4º costella contant des de la última. És un sistema de mesures lineals i es realitza en canals esquerres, mentre està penjada en la línia de l'escorxador. Amb aquestes mesures podem calcular el percentatge de magre segons la següent equació:

$$\%magre = 61,56 - 0,878\ g34fom + 0,517\ m34fom$$

LIMITACIONS

Tot i que el valor del percentatge de magre és calculat automàticament per l'equip, es tracta d'una mesura semi objectiva ja que **és necessari comptar amb un operador per a que la realitzi**. Aquest doncs és el principal problema que presenta aquesta tècnica.

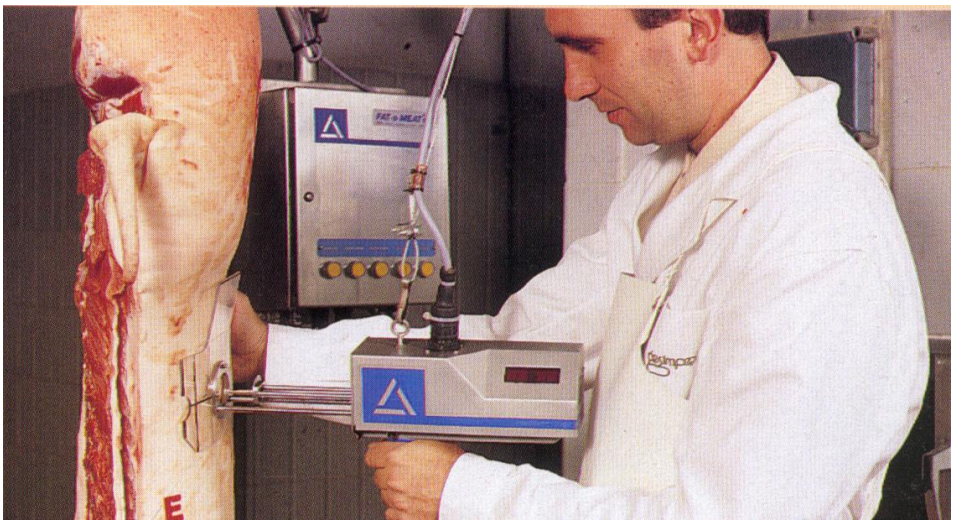


Figura 2.3: Sistema de classificació de canals Fat-O-Meat'er (FOM).

2.2.2.- AUTOFOM

El AUTOFOM (veure següent figura) és un sistema ultrasònic per la classificació de canals en els escorxadors porcins. Fabricat per l'empresa danesa SFK Technology A/S aquest equip utilitza la tècnica de l'escàner ultrasònic tridimensional per mesurar espessors de grassa i múscul en canals de manera totalment automàtica.

Les mesures es realitzen generalment quan l'animal surt de la depilació ja que encara es troba enter, calent i humit, permetent així una millor lectura. Posteriorment les dades registrades són processades amb la fi d'obtenir les prediccions més apropiades, per això s'utilitza una sèrie de models desenvolupats per mesurar el percentatge de magre en les canals, així com el pes i la composició dels principals talls.

A partir de les imatges obtingudes (unes 3200 per animal), es seleccionen automàticament 144 mesures d'espessors de grassa i múscul i a partir d'aquestes, mitjançant una equació aprovada a nivell europeu, s'obté el percentatge de magre de la canal i dels seus diferents talls.

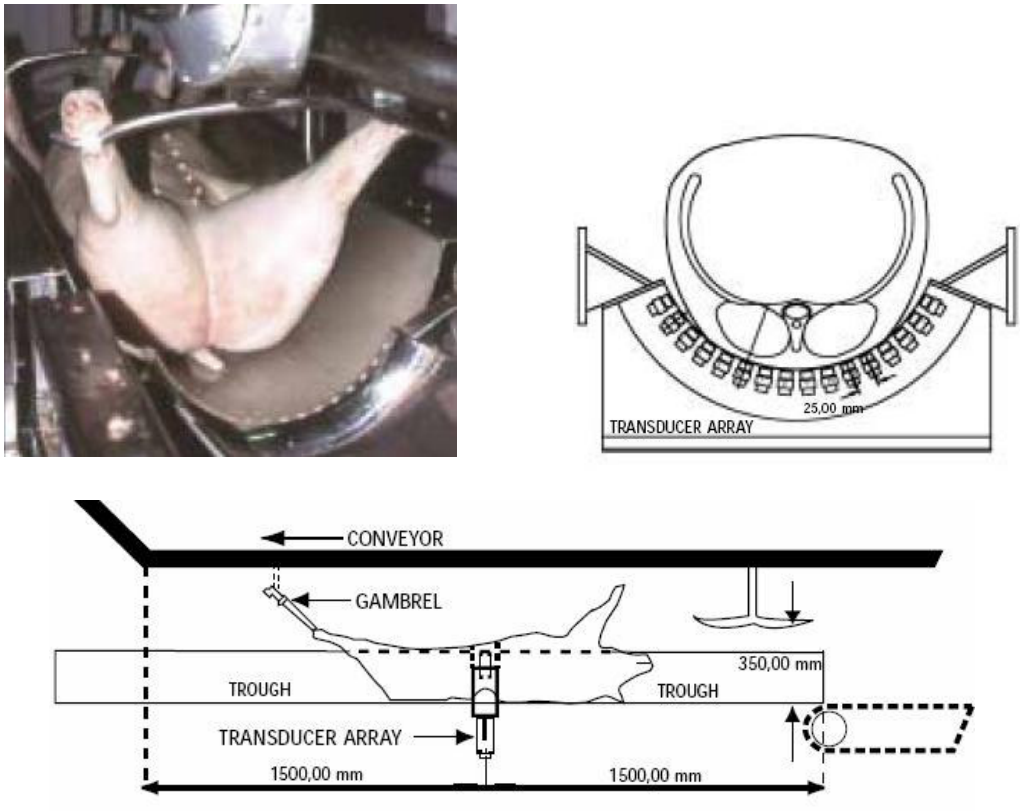


Figura 2.4: Sistema de classificació de canals AUTOFOM.

LIMITACIONS

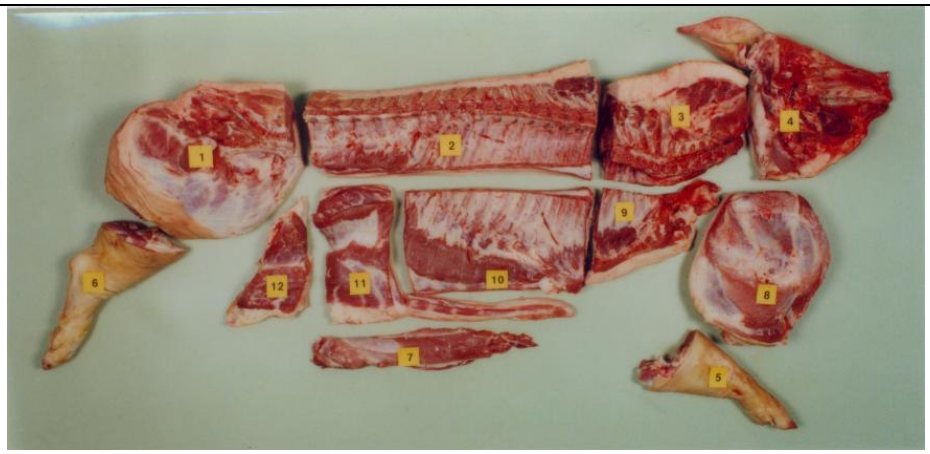
Malgrat que les mesures que es porten a terme amb aquest equip són totalment automàtiques, aquesta tècnica presenta el problema que s'han de realitzar uns **controls de validació per calibrar els equips**. A continuació es detallen aquests controls.

CALIBRATGE D'EQUIPS

Tots els equips s'han de calibrar i per això s'utilitza una fórmula de predicció del percentatge de magre que s'obté a partir de la dissecció. Com veurem a continuació per realitzar aquest calibratge i obtenir el percentatge de magre ens seran necessàries 3 etapes: **separació de peces**, **dissecció** i **càlcul del percentatge de magre**. El cost i temps necessaris per realitzar aquest procés és força elevat, d'aquí el principal problema de la tècnica actual AUTOFOM.

Com que el cost de la dissecció d'una canal és elevat tant per el temps que implica com per la seva depreciació, el treball d'avaluació de les canals es basa en l'estudi de les mitges canals. En el cas del sistema de classificació porcina, es consideren les mitges canals esquerres. Una vegada preparada la mitja

canal, el següent pas és la separació de peces que ens permet obtenir el pes de la canal. La separació de peces consisteix en dividir la canal en 12 peces comercials (següent figura). El pes de la canal es calcula com la suma dels 12 talls obtinguts.



Peces Comercials			
Peça 1	Pernil	Peça 7	Filet
Peça 2	Llom	Peça 8	Esquena
Peça 3	Llom d'agulles	Peça 9	Davant cansalada
Peça 4	Cap	Peça 10	Cansalada
Peça 5	Sofraja i mà	Peça 11	Retall mamelles
Peça 6	Sofraja i peu	Peça 12	Resta pernil

Figura 2.5: Separació de peces segons el sistema de referència europeu.

Llavors per obtenir el percentatge de magre es realitza el quocient entre el pes de magre respecte al pes total de la canal (suma de les 12 parts). El pes de magre es calcula de la dissecció de determinades peces comercials obtingudes anteriorment.

Aquesta tècnica es basa en la separació dels diferents teixits (grassa intramuscular, múscul, grassa subcutània més pell, ossos) que composen cada un dels talls que s'obtenen de la separació de peces (següent figura).

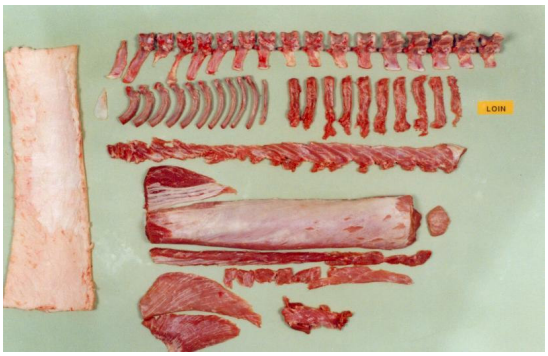


Figura 2.6: Dissecció completa de l'esquena.

En un principi el pes de la carn magre s'obtenia a través de la dissecció dels 12 talls de la canal. Aquest mètode es coneix com mètode de referència de dissecció completa (mètode de Kulmbach). Com que aquest treball implicava la disposició de 10 a 12 hores per canal, es va ajustar el mètode a un mètode simplificat en el qual la dissecció va quedar limitada a les 4 peces principals (pernil, llom, esquena i cansalada), que representa aproximadament el 75% del magre total. Per tant el percentatge de magre s'obté com:

$$\% \text{ magre} = \frac{1.3 * (\text{pes del magre del pernil + llom + cansalada + esquena + pes filet}) * 100}{\text{Pes de les 12 peces obtingudes de la separació de peces de referència}}$$

Per calibrar un equip s'han d'inspeccionar com a mínim 120 canals, les quals es seleccionen de manera que siguin representatives de la població porcina que es desitja mesurar. La fórmula és vàlida sempre i quan la població porcina no canviï. Per aquest motiu, es realitzen cada 5 anys comprovacions per determinar si hi ha hagut algun canvi en la població porcina i veure si encara es pot considerar vàlida l'equació.

Com acabem de veure a dia d'avui s'utilitzen bàsicament dues tècniques per realitzar la classificació de canals en funció del percentatge de magre en la carn.

D'una banda tenim el sistema FOM. Aquest sistema presenta el principal inconvenient que es necessita d'un operador per realitzar la tasca.

D'altra banda tenim el sistema AUTOFOM. El funcionament d'aquest sistema és eficient a excepció que necessita d'un procés de calibratge dels equips. Com hem vist, per realitzar aquest calibratge és necessari calcular el percentatge de magre en la carn. Actualment per efectuar aquest càlcul es realitzen 3 etapes: separació de peces, dissecció i finalment càlcul de percentatge de magre. Així doncs, el temps necessari és força elevat.

SOLUCIÓ QUE ES PROPOSA

L'objectiu d'aquest projecte és implementar un sistema automàtic de càlcul de percentatge de magre en la carn, de tal manera que es redueixi considerablement el temps necessari per efectuar el calibratge de la màquina en els sistemes AUTOFOM. Per efectuar aquesta automatització ens aprofitarem dels avantatges dels aparells de captació de dades de tomografia computeritzada.

Tomografia computeritzada per determinar la qualitat de la carn porcina

Com hem vist el procés de dissecció és totalment manual per la qual cosa resulta laboriós i costós, tant pel que fa referència al temps que implica com per la seva depreciació. La solució a aquests problemes es podria obtenir a partir de l'automatització de la dissecció. Per això seria necessari: (i) obtenir les dades del porc mitjançant tècniques d'adquisició no invasives, com la Tomografia Computeritzada (TC) i (ii) disposar de tècniques de segmentació i visualització d'imatge, que permetessin processar les dades per tal de determinar el contingut de magre en la carn amb una precisió similar a la dissecció.

El punt clau en tot aquest procés d'automatització de la dissecció està en l'obtenció i el posterior processament de les imatges del porc que s'obtenen a partir de la tomografia computeritzada. Totes les tècniques i mètodes que calen usar-se ens definiran un entorn informàtic. Independentment de les tècniques que s'implementin, la definició d'aquest entorn informàtic té uns passos comuns. El primer es **l'adquisició de dades** del porc mitjançant tècniques d'adquisició no invasives i **la definició del model** que s'usarà per mantenir aquestes dades en la memòria del computador.

Una vegada tenim el model, cal aplicar tècniques i mètodes per obtenir la informació que ens interessa. Particularment en el nostre cas, per calcular el percentatge de magre en la carn hem d'obtenir o segmentar només la regió de grassa. Per fer-ho, utilitzarem **tècniques de segmentació**. Posteriorment aplicarem un càlcul del volum segmentat junt amb **tècniques de visualització** per tal d'observar els resultats en la pantalla.

En aquest capítol ens centrarem en explicar el funcionament de cada una d'aquestes etapes. Començarem veient el procés d'adquisició de dades i la utilització del model de voxels. A continuació explicarem el funcionament de les tècniques de segmentació i visualització. Analitzarem les diferents opcions disponibles, i veurem les que finalment hem escollit per efectuar la implementació.

3.1.- ADQUISICIÓ DE DADES

La captació de dades del porc es realitza mitjançant algun dispositiu físic que permet mesurar un o més paràmetres determinats en diferents punts del cos del porc. Les dades obtingudes dependran de les prestacions de l'aparell: la seva resolució, fiabilitat, precisió, etc.

La tècnica de captació de dades més utilitzada és la **Tomografia Computeritzada (TC)**. La TC va ser la primera tècnica d'adquisició d'imatge que va permetre observar l'interior d'un organisme a través de talls mil·limètrics transversals (anomenats llesques o slices), mitjançant la utilització de raigs X.

Bàsicament el tomògraf, a través d'un feix de raigs X focalitzat en la seva amplada i en forma de ventall, travessa l'objecte i és registrat de forma digital per una fila de detectors col·locats enfront d'aquest.

Les diferents densitats de l'organisme als raigs X (ós, aire, aigua, grassa, ...) fan que el ventall que travessa el cos tingui un valor diferent en cada punt. Al girar el sistema 360º es genera una taula de densitats radiològiques, que analitzada per algoritmes matemàtics es representen en una imatge de la secció estudiada. D'aquesta manera es generen seccions "axials" (perpendiculars a l'eix longitudinal del cos), similars a llesques de pa, les quals poden tenir un grossor i una separació variable en funció de l'òrgan estudiat. A la figura següent podem observar el procés de captació i les imatges que podem obtenir utilitzant aquesta tècnica.

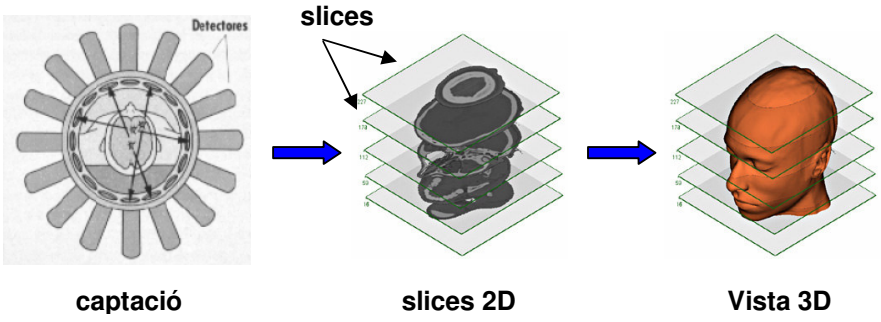


Figura 3.1: Procés d'adquisició de dades mitjançant la TC.

Actualment la TC es pot aplicar per estudiar y predir la composició d'animals vius o de canals. La capacitat predictiva de la TC depèn del propi equip i de les condicions de mesura utilitzades.

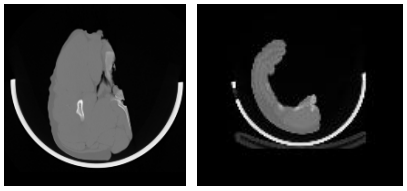
Les dades que s'obtenen del porc a través d'aquests dispositius es mantenen en un fitxer amb un format propi del fabricant de l'aparell, o bé, d'algun dels estàndards predefinits existents. Tot i que hi ha diversos estàndards, sembla que el DICOM és el que actualment està prenent més força. Malgrat la diversitat de format que existeixen, una de les característiques que tenen en comú el tipus de dades amb el que nosaltres treballem és que sempre segueixen una distribució espacial regular i venen distribuïdes sobre plans.

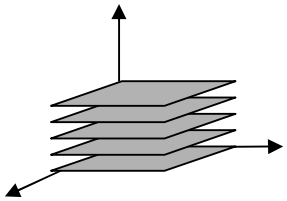
3.2.- DEFINICIÓ DEL MODEL DE VOXELS

Si volem tractar les dades del porc en el computador, cal definir un esquema de representació que ens permeti accedir fàcilment a aquestes dades.

El model més usat és el **model de voxels** proposat per Kaufman. Aquest model subdivideix l'espai en un conjunt de cubs o paral·lelepípedes de les mateixes dimensions, seguint una malla regular. Cadascun d'aquests cubs, o paral·lelepípedes, rep el nom de vòxel. Sobre cada vòxel s'hi representaran les dades que s'han obtingut del porc. En l'esquema següent podem veure una descripció gràfica de com es realitza aquest procés.

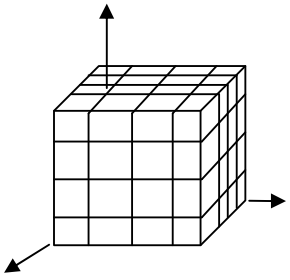
Partirem de les dades que s'han obtingut del porc. Observem que segueixen una distribució regular sobre un pla, cada un dels quals es pot interpretar com un tall de la part del cos que s'està examinant.





Els diferents plans o talls s'apilaran un sobre l'altre intentant reconstruir de nou la part 3D del cos.

Finalment es definirà una malla regular 3D formada per un conjunt de cel·les sobre el vèrtexs de les quals es representarà la informació de cada tall.



El model de voxels es caracteritza per:

- Ser un model aproximat: Només emmagatzema les propietats d'un número determinat de punts del volum.

- Requerir una gran quantitat de memòria: L'espai ocupat pel model té ordre O(n3) respecte a la resolució (n) .

A partir de les dades captades i distribuïdes sobre una malla regular es poden considerar dos models de voxels: el model tipus cel·la i el model tipus vòxel. Quan les mostres estan distribuïdes en els vèrtexs dels paral·lelepípedes que formen el model s'anomena model de cel·la. Si pel contrari, es considera que al interior del paral·lelepípede és homogeni és quan es parla de model de voxels.

Una vegada hem obtingut les dades i hem definit el model de voxels, aplicarem **tècniques de segmentació i visualització** per poder seleccionar i visualitzar la regió que ens interessa (en el nostre cas la grassa). A continuació explicarem el funcionament d'aquestes tècniques. Analitzarem les diferents opcions disponibles, i veurem les que finalment hem escollit per efectuar la implementació.

3.3.- TÈCNIQUES DE SEGMENTACIÓ

La **segmentació** és el procés d'identificar i classificar dades que es troben en una representació digital d'una mostra, és a dir, divideix una imatge 2D (o per extensió un model 3D) en un conjunt d'àrees connectades anomenades regions.

Cada regió està formada per grups de píxels (o voxels) que són uniformes respecte alguna propietat: nivell de gris, color, etc. Habitualment la representació de la mostra és una imatge obtinguda mitjançant dispositius de captació com ara escàners, TC o ressonàncies magnètiques.

La segmentació d'imatges és una tasca que requereix bastant d'esforç. Existeixen moltes tècniques de segmentació i no hi ha una única proposta general que permeti segmentar totes les modalitats d'imatges existents en l'actualitat. Els algoritmes de segmentació més efectius s'obtenen personalitzant combinacions dels components. Els paràmetres d'aquestes components s'adapten segons les característiques de la imatge utilitzada com entrada i les característiques de l'estructura que es pretén segmentar.

Les llibreries Itk ofereixen una sèrie de tècniques bàsiques que es poden utilitzar per desenvolupar i personalitzar una aplicació de segmentació complerta. Algunes de les tècniques més utilitzades són les que comentarem a continuació.

3.3.1.- REGION GROWING

Els algoritmes de Region Growing són una de les aproximacions més efectives per a la segmentació d'imatges. La idea principal d'aquest algoritme és començar a partir d'una zona llavor (habitualment un o més píxels) que estigui situada dins l'objecte a segmentar. Els píxels veïns d'aquesta regió són analitzats per determinar si s'han de considerar part de l'objecte. En cas afirmatiu, s'afegeixen a la regió i el procés contínua sempre hi quan es vagin afegint nous píxels a la regió.

Els algoritmes de Region Growing varien depenent del criteri que s'utilitzi per decidir si un píxel ha de ser inclòs a la regió o no, del criteri utilitzat per determinar els veïns, i de l'estratègia usada per visitar els píxels veïns.

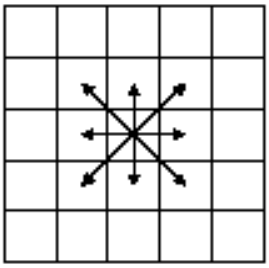


Figura 3.2: Direccions cap on avança el Region Growing.

Diferents implementacions de Region Growing estan disponibles en Itk. A continuació descriurem les més utilitzades.

Connected Threshold

Un criteri molt simple per incloure píxels a la regió creixent (Region Growing) és avaluar la intensitat dins un rang específic. La major complexitat d'un algoritme Region Growing ve al visitar els píxels veïns. El Connected Threshold disposa d'un iterador que assumeix aquesta responsabilitat i simplifica molt la implementació de l'algoritme. D'aquesta manera l'algoritme únicament ha de fixar el criteri per decidir si un píxel concret ha de ser inclòs o no dins la regió segmentada.

El criteri utilitzat per el Connected Threshold es basa en l'interval d'intensitat que selecciona l'usuari. Aquest escull un valor per el lower threshold i upper threshold. L'algoritme inclou aquells píxels que tenen una intensitat dins l'interval.

$$I(X) \in [lower,upper]$$

Si es seleccionen els dos valors massa pròxims no permetrà gaire flexibilitat a la regió per créixer. Per contra, si es seleccionen massa distants s'acabarà obtenint una regió que contindrà tota la imatge.

Una de les vulnerabilitats de les tècniques de Region Growing es produeix quan les estructures anatòmiques a segmentar no tenen una distribució estadística homogènia sobre l'espai de la imatge. Una opció per solucionar aquests problemes és la utilització de múltiples llavors. La tècnica de Connected Threshold admet la utilització de més d'una llavor.

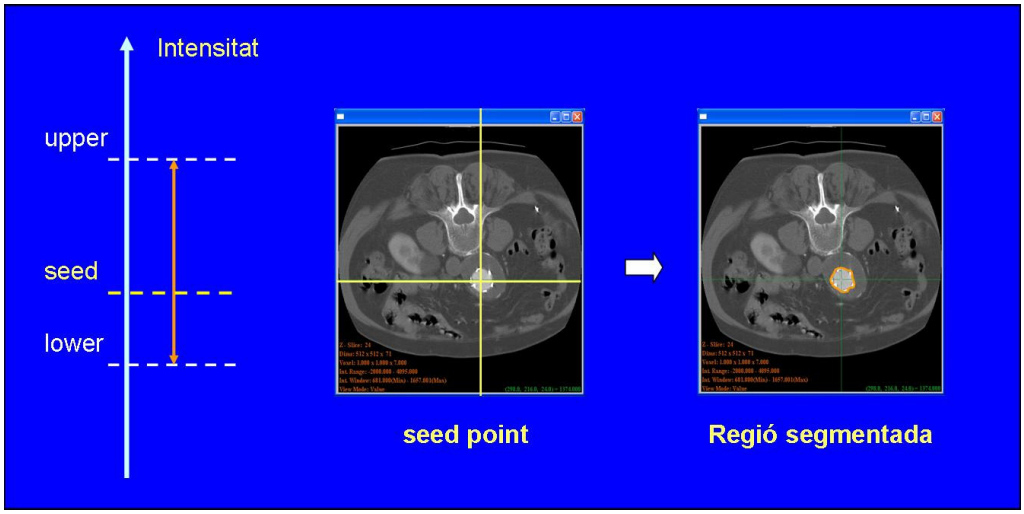


Figura 3.3: Esquema del funcionament de l'algoritme Connected Threshold.

Neighborhood Connected

Aquesta tècnica és una variant del Connected Threshold. En el cas anterior acceptàvem un píxel a la regió si la seva intensitat pertanyia dins l'interval definit per l'usuari mitjançant els dos valors de threshold. En el Neighborhood Connected, en canvi, només acceptarem un píxel si tots els seus veïns tenen intensitats que pertanyen a l'interval. La dimensió del "veïnat" a considerar al voltant del píxel és definit per l'usuari a través del paràmetre radi. Com més gran sigui el valor d'aquest paràmetre més estable serà el filtre contra imatges amb "soroll", però al mateix temps necessitarà un major temps de computació. La raó per considerar les intensitats dels veïns enlloc de únicament la intensitat del píxel és perquè d'aquesta manera estructures petites tenen menys possibilitats de ser acceptades en la regió. Al igual que succeïa amb la tècnica anterior, en aquesta ocasió també podem seleccionar més d'una llavor.

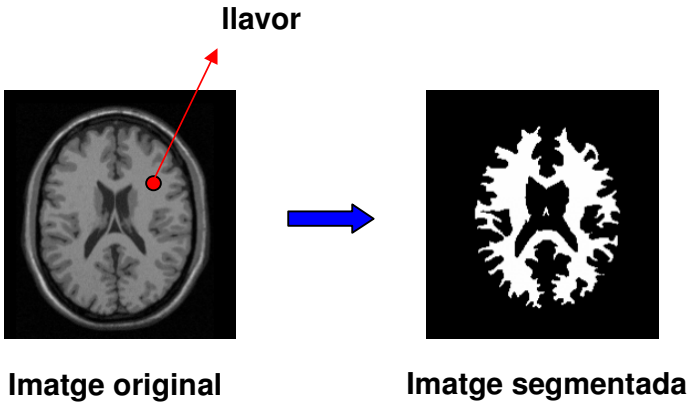


Figura 3.4: Exemple d'execució de la tècnica Neighborhood Connected.

Confidence Connected

El criteri utilitzat per el Confidence Connected està basat en una simple estadística de la regió creixent. Primer de tot, l'algoritme calcula la mitjana i la desviació estàndard dels valors d'intensitat per tots els píxels que en aquell moment pertanyen a la regió creixent.

Un factor que selecciona l'usuari s'utilitza per multiplicar la desviació estàndard i definir un rang al voltant de la mitjana. Els píxels veïns amb intensitat dins aquest rang són inclosos a la regió. Quan no hi ha més píxels veïns que satisfacin aquest criteri, es considera que l'algoritme ha acabat la primera iteració.

Arribat aquest punt, la mitjana i la desviació estàndard es tornen a calcular utilitzant tots els píxels que la regió conté en aquest moment. Aquesta mitjana i desviació estàndard defineixen un nou rang d'intensitats que s'utilitza per visitar els píxels veïns i avaluar si la seva intensitat està dins el rang.

Aquest procés iteratiu es repeteix fins que no existeixin més píxels a afegir, o bé, hagin arribat al nombre màxim d'iteracions seleccionat. L'equació de continuació mostra el criteri d'inclusió utilitzat per aquest filtre.

$$I(X) \in [m - f\sigma, m + f\sigma]$$

On m i σ són la mitjana i la desviació estàndard, f és el factor definit per l'usuari, I() és la imatge i X és la posició del píxel veí particular que s'està analitzant per decidir si pertany a la regió. Valors petits del factor f restringiran la inclusió de píxels amb intensitats molt similars als píxels de la regió creixent. Per contra, valors alts del multiplicador "relaxaran" la condició d'acceptació, i com a resultat obtindrem regions de dimensions més grans. Valors que siguin massa grans poden causar que la regió creixi a regions que potser pertanyen a estructures anatòmiques diferents.

Un altre dels paràmetres és el nombre d'iteracions. Habitualment regions homogènies només necessiten un parell d'iteracions. Regions amb desnivells com ara imatges MRI (Imatges de Ressonància Magnètica) amb camps heterogenis, possiblement requeriran més iteracions.

A la pràctica resulta més important seleccionar bé el factor multiplicador que no pas el nombre d'iteracions. A més a més, hem de tenir present que no hi ha cap raó per assumir que aquest algoritme convergeixi cap a una regió estable, és a dir, és possible que si deixem l'algoritme funcionar durant més iteracions ens acabi generant una regió segmentada que "ocupi" tota la imatge (enlloc de segmentar amb més precisió la regió que ens interessava).

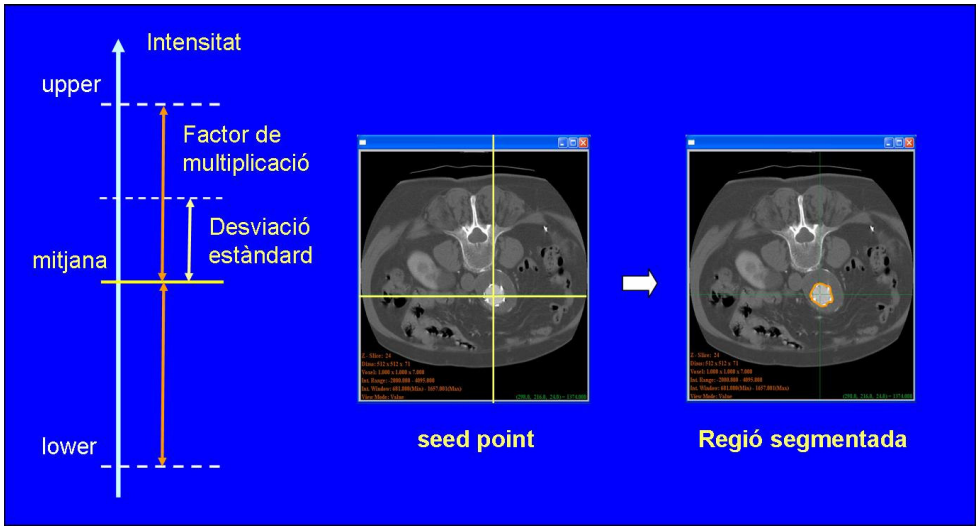


Figura 3.5: Esquema del funcionament de l'algoritme Confidence Connected.

Isolated Connected

Aquesta tècnica té com a paràmetres dues llavors i el lower threshold. Aquest filtre crearà una regió connectada amb la primera llavor i desconnectada de la segona. Per tal d'aconseguir-ho, el filtre busca un valor d'intensitat que pugui ser utilitzat com a upper threshold per la primera llavor. Una cerca binària s'utilitza per trobar el valor que separa les dues llavors.

Aquest filtre està pensat en ocasions on les estructures anatòmiques adjacents són difícils de separar. Seleccionant una llavor dins d'una estructura, i l'altre llavor en l'estructura adjacent, es calcularà el threshold que separarà les dues estructures i ens permetrà obtenir la regió segmentada amb més precisió.

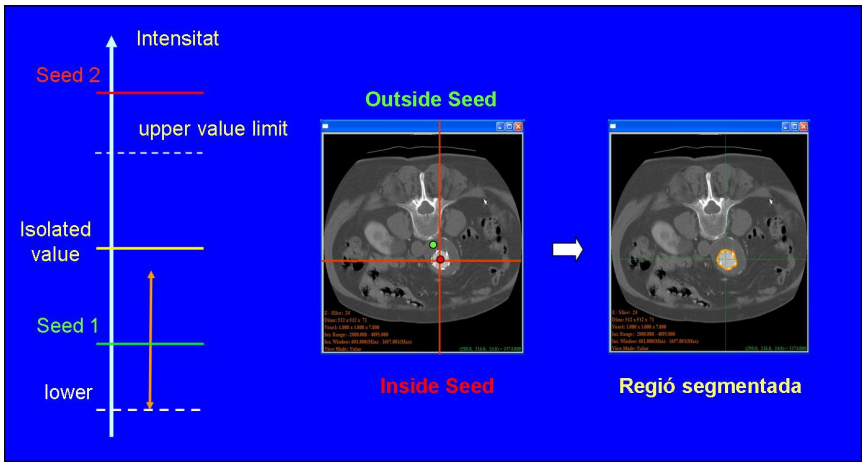


Figura 3.6: Esquema del funcionament de l'algoritme Isolated Connected.

3.3.2.- WATERSHED

La segmentació basada en Watersheds classifica els píxels en diferents regions usant descens de gradient en les imatges, i anàlisi dels punts "febles" en les regions frontera. Per explicar el seu funcionament es realitza una comparació amb la pluja. Imaginem quan plou, que la pluja s'acumula en un terreny i comença a formar petites basses. La dimensió d'aquestes basses creixerà com més precipitació hi hagi, fins que arribi el moment que la bassa vessi i s'acabi ajuntant amb una altre bassa, provocant d'aquesta manera basses més grans. En aquest exemple les línies Watershed serien les fronteres de separació entre les basses, és a dir, les que determinarien per a cada una de les basses la seva zona d'influència o conca. A més, cada bassa estaria associada a un mínim local de relleu. Aquests mínim locals serien les llavors del mètode de segmentació.

La transformació Watershed es pot aplicar a imatges en escala de grisos, tenint en compte que la intensitat d'un punt representa una altura equivalent en un relleu topogràfic associat. Així doncs, el valor numèric de cada píxel representa l'elevació en aquest punt.

La tècnica de Watershed és menys sensible als threshold definits per l'usuari, en comparació amb les tècniques de Region Growing. Al mateix temps és una tècnica més flexible, en el sentit que no produeix una única imatge de segmentació, sinó un grup de segmentacions, a través de la qual es pot segmentar una regió concreta especificant el threshold desitjat.

La estratègia de la segmentació Watershed és tractar una imatge *f* com una funció d'alçada. Sovint aquesta imatge no és l'original, sinó que s'hi aplica prèviament algun filtratge, extracció o fusió. Els valors alts d'aquesta imatge *f* indiquen la presència de fronteres o límits en les dades originals.

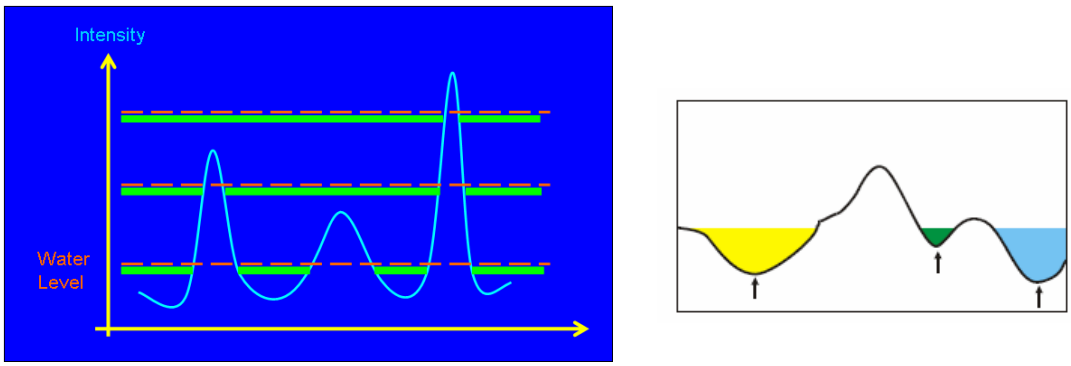


Figura 3.7: Esquema del funcionament de l'algoritme Watershed. Com s'observa a la imatge de la dreta, cada regió es pinta d'un color diferent.

Podem veure a les figures anteriors com la inundació comença a partir de les llavors situades als mínims locals (marcades amb unes fletxes a la imatge de la dreta), i com cada regió s'identifica amb un color diferent. A partir d'aquí s'agafen com a frontera els punts més alts d'aquest mapa topogràfic. El resultat final d'aquest tipus de segmentació és una imatge amb diferents colors per cada segment.

L'inconvenient de la segmentació Watershed és que produeix una regió per cada mínim local - en pràcticament totes les regions - i masses resultats de segmentació.

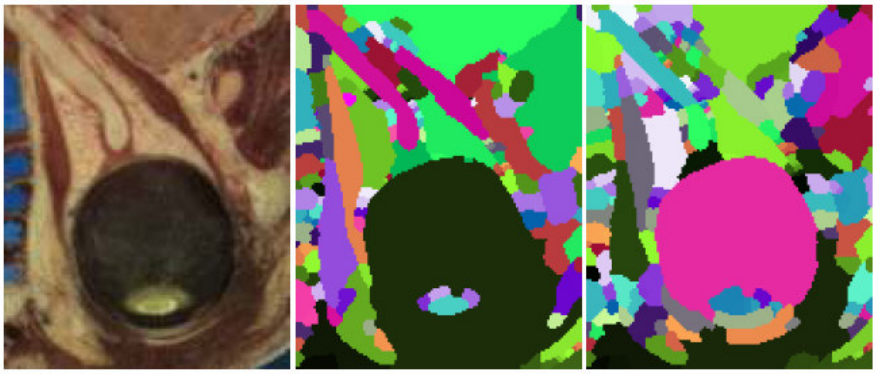


Figura 3.8: Exemple d'execució de la tècnica Watershed. La imatge de l'esquerra és la imatge original d'un cap humà. Les altres dues imatges són el resultat d'aplicar la tècnica Watershed amb diferents paràmetres.

3.3.3.- LEVEL SET

La segmentació Level Set és una tècnica numèrica per seguir l'evolució de contorns i de superfícies. Enlloc de manipular el contorn directament, es calcula mitjançant una funció anomenada funció de level set. Concretament el contorn es calcula a partir dels valors zeros d'aquesta funció. Els avantatges principals d'utilitzar aquesta tècnica, és que es poden modelar, associar, dividir o manejar formes complexes de les imatges a segmentar. Existeixen diferents variacions dins aquesta tècnica, com ara el Fast Marching, Geodesic Active o Shape Detection.

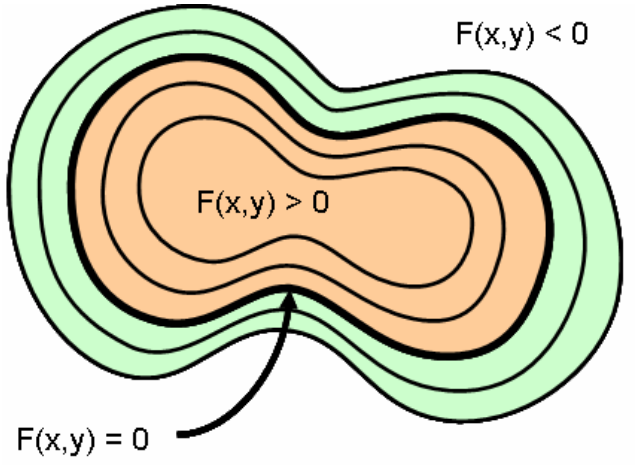


Figura 3.9: Diferents contorns que pot prendre la funció F(x,y).

La figura anterior ens mostra els diferents contorns que pot agafar la funció de level set. Com es pot veure tindrà un valor superior a 0 si està fora del contorn òptim, i inferior a 0 si està a dins. Amb això direm que l'encreuament amb el 0 serà el contorn.

De les diferents tècniques de segmentació que acabem de veure, **en aquest projecte hem implementat les 4 següents pertanyen a la metodologia de Region Growing:**

- **Connected Threshold**
- **Neighborhood Connected**
- **Confidence Connected**
- **Isolated Connected**

D'una banda hem descartat la implementació de les tècniques de Watershed ja que el resultat final d'aquest tipus de segmentació és una imatge amb diferents colors per cada segment. Així doncs, no ens permetia segmentar únicament la regió de magre o grassa.

D'altra banda també hem descartat la implementació de les tècniques de Level Set ja que els temps necessaris per executar aquest tipus de tècniques són molt elevats en comparació a les tècniques Region Growing.

3.4.- CÀLCUL DE VOLUMS

Mitjançant les tècniques de segmentació que acabem de comentar, obtindrem com a resultat la regió segmentada que ens interessa. Ara bé, també ens serà necessari calcular el volum d'aquesta regió.

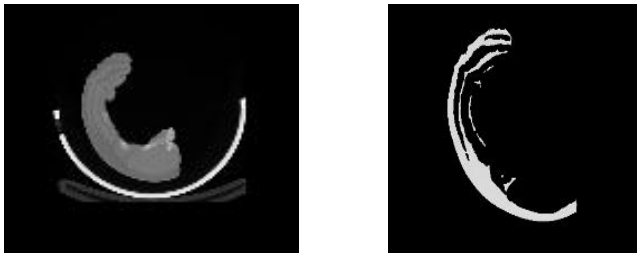
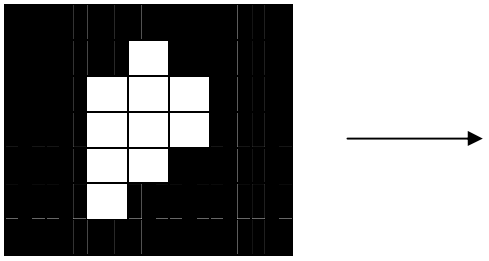


Figura 3.10: A l'esquerra observem la imatge original. A la dreta la imatge segmentada. Ara ens interessarà calcular el volum d'aquesta segona imatge.

La regió segmentada és en realitat una imatge binària on la regió d'interès està codificada amb el valor 1. La resta de la imatge no te cap interès i estarà codificada amb el valor 0.



0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	0	0	0
0	0	1	0	0	0	0
0	0	0	0	0	0	0

Figura 3.11: Codificació de la imatge segmentada.

La tècnica que hem utilitzat per a calcular el volum és força senzilla. Es tracte de recórrer el volum segmentat de principi a fi. Examinarem tots els píxels d'una llesca fins arribar al final. A continuació passarem a la següent llesca fins a finalitzar tot el model. A mesura que fem el recorregut contarem tant els píxels totals segmentats com els de cada llesca que tenen un valor de 1. D'aquesta manera obtindrem el volum total de l'objecte segmentat i l'àrea de cada llesca.

Per validar aquest càlcul varem implementar un petit programa per crear models sintètics, és a dir, models dels quals coneixíem les dimensions (amb el programa podíem crear models de les dimensions que volguéssim) i així comprovar que els resultats i càlculs realitzats eren els correctes.

Exemple: *Varem crear un model de dimensions 128x128x128. D'aquests varem pintar 120x120x120 voxels. Així doncs, els resultats que havíem d'obtenir eren els següents:*

Àrea de la llesca: 120x120 = 14400 mm²
Volum total: 120x120x120 = 1728000 mm³

Com podem observar en la següent figura els resultats concorden. L'àrea de la llesca és el que anomenem *"Slice Area*" i el volum total *"Total Volume"*.

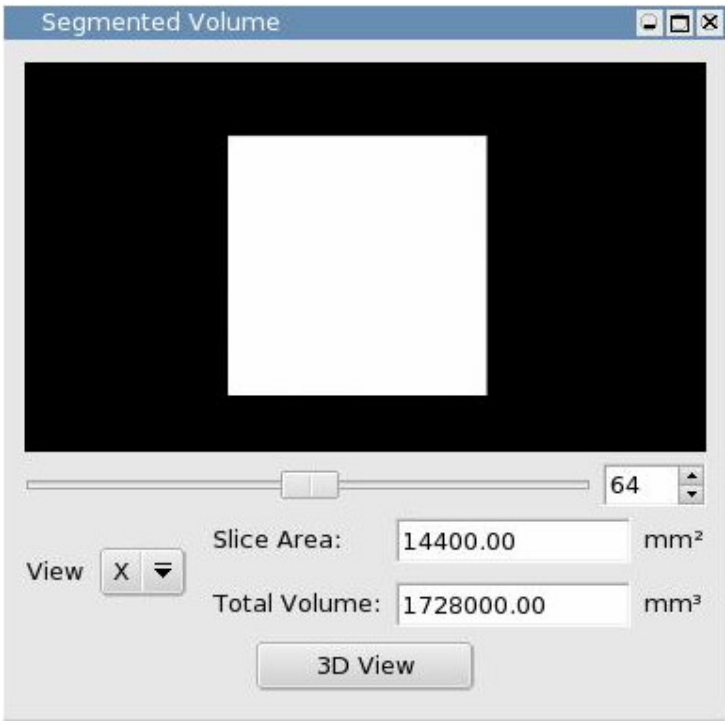


Figura 3.12: Prova realitzada sobre un model 128x128x128 del qual varem pintar 120x120x120.

3.5.- TÈCNIQUES DE VISUALITZACIÓ

Finalment l'últim pas que ens manca realitzar és aplicar alguna tècnica de visualització que ens permeti visualitzar la informació. La tècnica de visualització haurà de determinar quins atributs gràfics s'assignen als diferents valors de propietat representats en el model de voxels, per tal que es pugui obtenir una representació gràfica del mateix. El tipus d' atributs i les tècniques a aplicar, variaran en funció del tipus d'informació que volem visualitzar i del tipus d'informació que tinguem representada en el model.

En el nostre cas ens interessa una tècnica que ens permeti representar tot el volum (una opció de la plataforma és visualitzar tot el model) i al mateix temps també ens permeti visualitzar un model simplificat (en el moment de realitzar la segmentació disposarem d'un model simplificat). Per aquest motiu varem optar per implementar un **mètode de visualització directa de volums**.

Les tècniques de visualització directa de volum treballen directament amb totes les dades representades en el model de voxels. Per generar la imatge final s'assignen els atributs gràfics (color, transparència, textures,...) a tots els valors de propietat representats en el model de voxels. Els algorismes de visualització directa de volums ens permeten observar més d'una superfície al mateix temps. De fet, podríem observar tantes superfícies com valors diferents disposa el model que estem examinant.

Dels diferents algoritmes de visualització directa de volums varem optar per implementar l'anomenat **Ray Casting**. A continuació descriurem els punts claus que cal tenir en compte per definir un algorisme de visualització directe de volums. Finalment explicarem el funcionament de la tècnica escollida.

3.5.1.- ASSIGNACIÓ DELS ATRIBUTS GRÀFICS

Si tenim en compte que en el model de voxels hi tenim representats un conjunt de mesures que representen certa part del cos del porc, obtenir una representació visual d'aquesta model de voxels (d'aquesta part del cos) es pot considerar com un procés de mapeig en el que a cada mostra representada en el model, cal assignar-li un determinat color o atribut gràfic. Per exemple, les propietats que representen ós les volem pintar en pantalla de color blanquinós i totalment opaques, mentre que les que representen pell les volem rosades i amb certa transparència. Una vegada determinat el color que l'hi correspon a cada propietat, s'aplica un tècnica de visualització que indica com composar tots els atributs gràfics per tal d'obtenir la imatge. Així doncs, el primer factor clau del procés de visualització és l'assignació dels atributs gràfics a les mostres representades en el model de voxels.

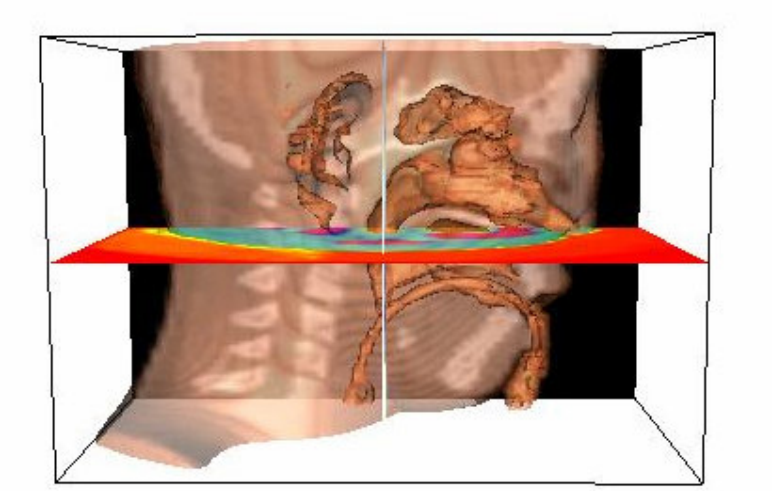


Figura 3.13: Imatge on es poden observar els efectes de la transparència.

Els atributs gràfics que s'usen per visualitzar les dades són el color i l'opacitat.

- **Color:** Aquest component vindrà representat per una combinació dels seus valors primaris: vermell, verd i blau (RGB).

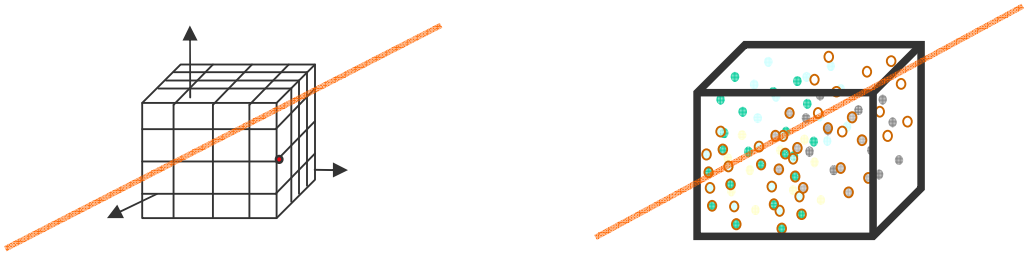
- **Opacitat:** La opacitat no és res més que el grau de transparència. És el que en direm a partir d'ara component alpha del color. El rang de alpha va de 0 a 1 essent 0 transparent i 1 totalment opac. El component alpha ens permetrà amagar parts que no ens interessin veure d'un volum determinat. Per exemple, si coneixem quines propietats són les que pertanyen a la pell, podríem fer que aquestes tinguessin un component alpha = 0 de manera que només veiéssim el que hi ha sota la pell (músculs, os...). El component opacitat es representa com Alpha (A) i es parla així del valor RGBA.

Per poder visualitzar el model aplicant una tècnica de visualització directa de volums, el primer que cal fer és assignar a cada mostra representada en el model de voxels un color i una opacitat (un RGBA). La estratègia que hem utilitzat per realitzar l'assignació RGBA és la que s'anomena **funció de transferència.**

En la funció de transferència l'usuari defineix per cada rang de propietat un color i una opacitat. En el moment de fer la visualització, s'analitzaran els diferents valors de propietat i en funció del color i la opacitat que haguem definit en la funció de transferència es realitzarà l'assignació de colors.

3.5.2.- COMPOSICIÓ DE COLORS

Una vegada sabem el valor RGBA que cal assignar a cada valor de propietat, per poder generar la imatge final ens cal aplicar un altre procés anomenat de composició de colors. En realitat, després de fer l'assignació dels atributs gràfics, el nostre model de voxels es pot veure com un conjunt de punts distribuïts en l'espai que tenen assignats un conjunt de colors i opacitats (veure figura següent) .



(a): model de voxels

(b): colors i opacitats assignats als diferents valors de propietat

Figura 3.14: Procés de composició de colors

Per determinar quin color correspon a cadascun dels píxels de la imatge final, haurem de calcular la contribució final dels diferents voxels que es veuen des del píxel. Aquesta contribució no es més que la composició dels diferents colors assignats als diferents voxels. Podem pensar que cada vòxel és com un vidre transparent i que el que ens interessa és saber quin és el resultat final de posar els vidres un a sobre de l'altre. El color final que es veu és el que s'anomena la composició dels colors.

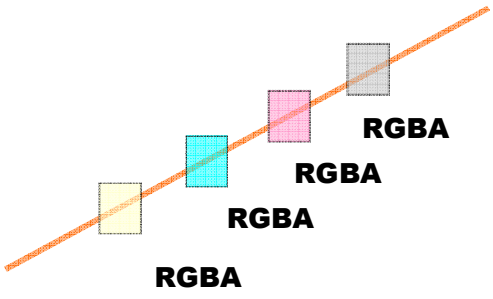


Figura 3.15: Composició de colors per determinar el color final dels píxels

Per fer la composició dels colors es poden seguir dues estratègies: de davant a darrera, o de darrera endavant, segons l'ordre en que es prenguin els voxels. Una vegada s'ha fixat l'ordre en que es prenen les mostres, llavors el color resultant es calcula aplicant la formula següent:

$$RGB_{final}=RGB_{mostra\_actual}+(1-A_{mostra\_actual})*RGBA_{anterior}$$

On RGB_{mostra_actual} és el color de la mostra que estem processant i $RGBA_{anterior}$ és el color que hem obtingut de la composició de les mostres anteriors. Per determinar l'opacitat de la mostra final s'aplica l'equació:

$$A_{final}=A_{mostra\_actual}+(1-A_{mostra\_actual})*A_{anterior}$$

Una vegada definides les tècniques d'assignació de colors i la forma en la que cal composar-los per determinar el color de la imatge final, passarem a veure el funcionament de l'algoritme de visualització de Ray Casting.

3.5.3.- RAY CASTING

El Ray Casting és l'algorisme més representatiu de la tècnica image-order. La idea bàsica d'aquests algorismes és llençar un raig per cada píxel de la imatge. El color final del píxel s'obté fent la composició dels valors RGBA dels voxels amb els que el raig ha intersecat (veure següent figura).

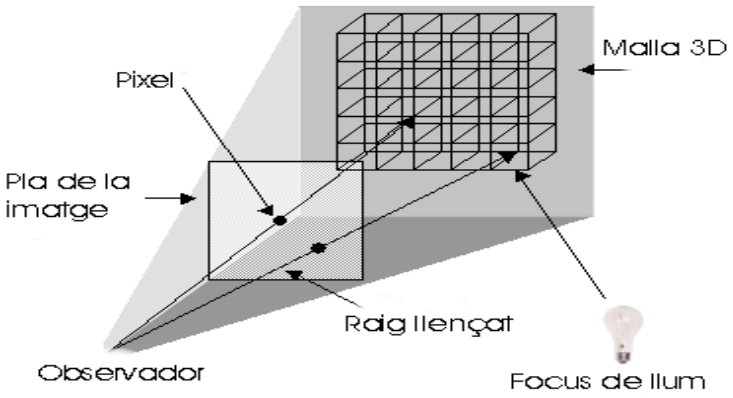


Figura 3.16: Algoritme image-order usant la tècnica de Ray Casting.

Els dos passos principals del Ray Casting consisteixen en determinar els valors que es troben al llarg dels raigs que hem llançat, i d'altra banda composar aquests valors. Les diferents variants que s'han proposat del Ray Casting varien en la forma en la que es prenen les mostres sobre el raig. Tal i com es representa en la figura de continuació, podem considerar el vòxel que interseca el raig com un valor únic, podem considerar un valor representatiu obtingut amb la mitjana dels valors de propietat representats en els vèrtexs del vòxel, o bé podem considerar diferents valors al interior del vòxel.

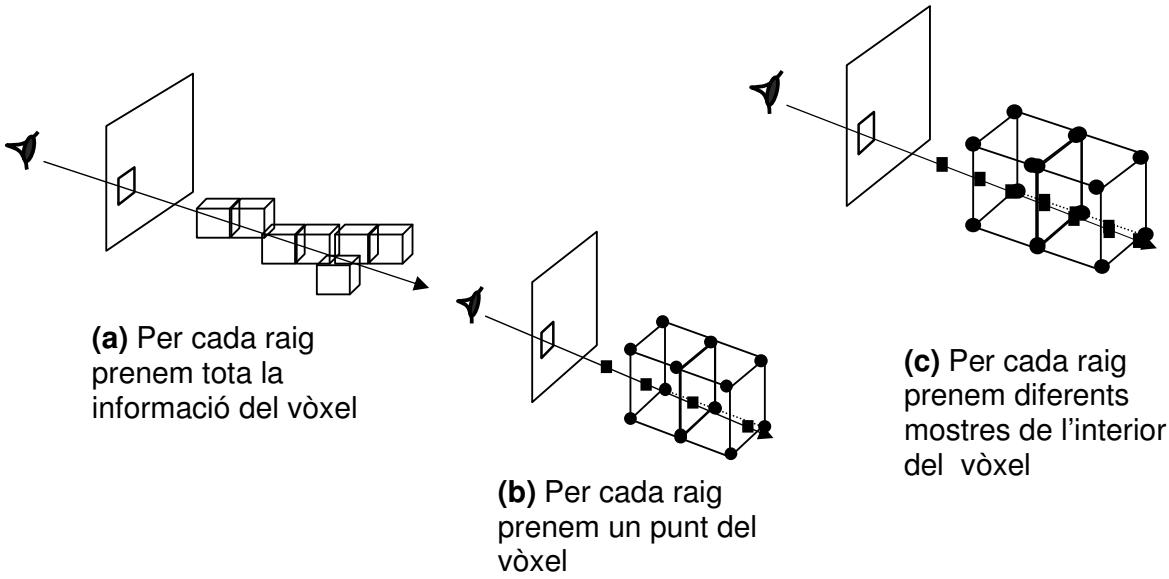


Figura 3.17: Diferents formes de prendre les mostres sobre el raig.

La funció d'interpolació més simple anomenada Nearest Neighbor Interpolation, retorna el valor de la mostra més propera al punt d'intersecció entre el voxel i el raig. El Nearest Neighbor Interpolation ens proporciona una manera una mica més ràpida d'aconseguir el resultat final. Malgrat tot, la imatge generada no és del tot precisa. L'altre tipus d'interpolació més usada i que proporciona una millor qualitat de la representació és la interpolació **Trilineal**. Per aquest motiu hem optat per utilitzar aquest tipus d'interpolació.

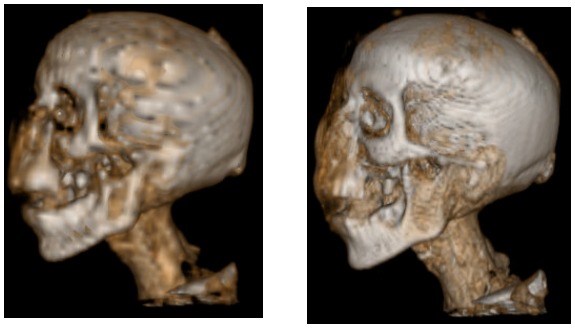


Figura 3.18: **(a):** Nearest Neighbor **(b):** Trilineal

Tal i com hem comentat, la metodologia usada per el Ray Casting consisteix en tirar raigs. Per tal de travessar les dades mitjançant un raig, haurem d'anar prenent mostres del volum en intervals uniformes, o bé, podem travessar una representació discreta del raig a través del volum, examinant cada vòxel que anem trobant. La selecció del mètode depèn de diversos factors com la tècnica d'interpolació, la funció de raig, o de si l'objectiu és la velocitat o el detall.

El raig es sol representar en paràmetres de la forma següent:

$$(x, y, z) = (x_0, y_0, z_0) + (a, b, c)t$$

On (x_0, y_0, z_0) és l'origen del raig (la posició de la càmera en el cas de vistes perspectives o un píxel del pla de vista en el cas de vistes paral·leles), i (a,b,c) és el vector normalitzat de la direcció del raig.

Un altre factor a tenir en compte és el número de mostres que prenen sobre cada raig. Quantes més mostres, més qualitat de la imatge i més lent el procés, mentre que menys mostres impliquen més rapidesa però menys qualitat en la imatge final. El número de mostres que es prenen ve definit pel **step size**. Si la distància escollida és massa llarga, llavors les mostres que prendrem deixaran força que desitjar. Si per contra seleccionem una distància massa curta llavors necessitarem bastant de temps fins no acabem de representar la imatge. En cas de seleccionar una distància massa gran, les imatges generades poden aparèixer com uns ratlles o anells que il·luminen massa certes regions del volum. Per reduir aquest efecte es podria fer avançar l'origen del raig que tirem al llarg de la direcció de la vista, una distància aleatòria. D'aquesta manera, obtindrem una imatge millor i reduirem el aliasing que ens apareixia anteriorment.

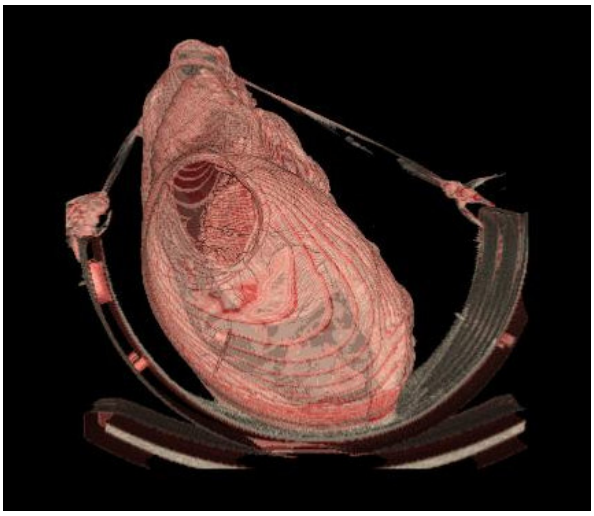


Figura 3.19: Exemple d'una imatge que hem obtingut visualitzant el model de porc amb el Ray Casting.

3.6.- TÈCNIQUES IMPLEMENTADES

Al llarg d'aquest capítol hem vist les diferents accions que haurem de realitzar per aconseguir determinar el percentatge de magre en la carn porcina. A continuació mostrem un esquema resumint les diferents etapes que hem vist d'aquest procés, junt amb les tècniques implementades.

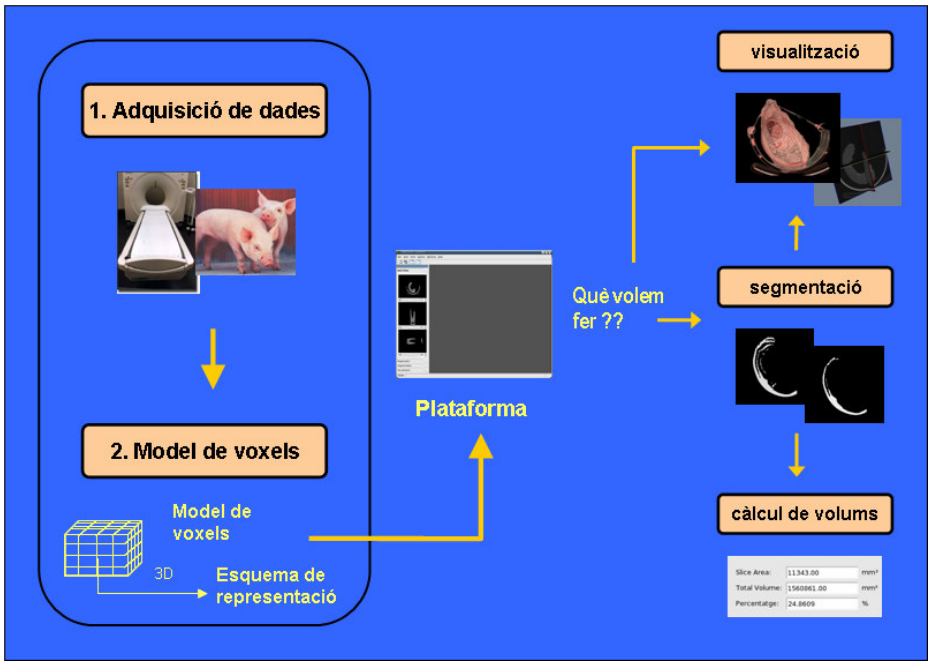


Figura 3.20: Diferents etapes en el procés d'automatització de la dissecció.

Adquisició de dades

Primer de tot disposarem d'un sistema d'adquisició de dades no invasiva com és la Tomografia Computeritzada. Mitjançant aquest sistema obtindrem les dades del porc.

Esquema de representació

Per poder efectuar un processament informàtic sobre aquestes dades serà necessari disposar d'un esquema de representació. En el nostre cas el model utilitzat és el model de voxels.

Tècniques de segmentació

A continuació aplicarem les tècniques de segmentació que ens permetran obtenir o separar les regions que ens interessin (en el nostre cas la regió de grassa). En aquest projecte hem implementat quatre tècniques de segmentació: Connected Threshold, Neighborhood Connected, Confidence Connected i Isolated Connected.

Tècniques de visualització

També disposarem de tècniques de visualització per obtenir una representació del model. En aquest projecte hem implementat la tècnica de visualització de Ray Casting, junt amb una altre tècnica de visualització alternativa anomenada Multiplanar. Aquesta última tècnica ens mostrarà 3 vistes del model (Axial, Sagital i Coronal) i ens permetrà desplaçar-nos per les diferents llesques de cada vista.

Càlcul de volums

Finalment també ens serà necessari implementar un sistema de càlcul de volums. Per calcular el volum hem implementat un algoritme molt senzill consistent en recórrer el volum segmentat de principi a fi i comptar els píxels que estiguin “plens”.

Implementació

Antecedents

El projecte que he implementat forma part del desenvolupament d'una plataforma d'automatització de la dissecció. En aquest capítol veurem una visió general de com és la plataforma de la qual partíem, així com les funcionalitats que oferia. D'altra banda, també descriurem breument les llibreries utilitzades per la implementació del projecte.

4.1.- LA PLATAFORMA INICIAL

Per tal de poder desenvolupar el meu projecte i integrar-lo en aquesta plataforma de visualització d'imatges porcines, se'm va facilitar una interfície inicial capaç de crear un model de voxels a partir de les dades en format DICOM d'un pacient mèdic.

Malgrat que les dades porcines amb les que treballàvem també estaven amb format DICOM, aquests tenien un format i unes característiques diferents respecte les imatges mèdiques. Per aquest motiu la primera acció que varem haver de realitzar va ser transformar les dades porcines a un format DICOM estàndard, i aconseguir així que es poguessin obrir correctament amb la plataforma inicial.

Una vegada realitzada aquesta conversió al format DICOM estàndard, la interfície inicial ens permetia obrir el model de dades de porc, i visualitzar les llesques Axial, Sagital i Coronal d'aquest volum. Aquestes llesques estaven situades en l'apartat anomenat *"Axis View"* de la part superior esquerre de la pantalla. Cada una d'aquestes vistes també disposa de 3 *"sliders"* o exploradors de llesca per poder avançar en la direcció de cada eix.

D'altra banda, la interfície disposava d'una regió gris situada a la dreta de la pantalla anomenada *"Working Area",* on no hi havia cap mena de model ni visualització oberta. Aquesta regió és la que utilitzarem per mostrar els resultats de les tècniques de segmentació i visualització que hem implementat.

En les imatges que mostrem a continuació podem observar aquesta interfície original a partir de la qual varem començar a treballar. En la primera imatge s'observa l'estructura d'aquesta interfície abans d'obrir cap model, mentre que en la segona imatge s'observa com s'actualitza la interfície al obrir un model porcí.

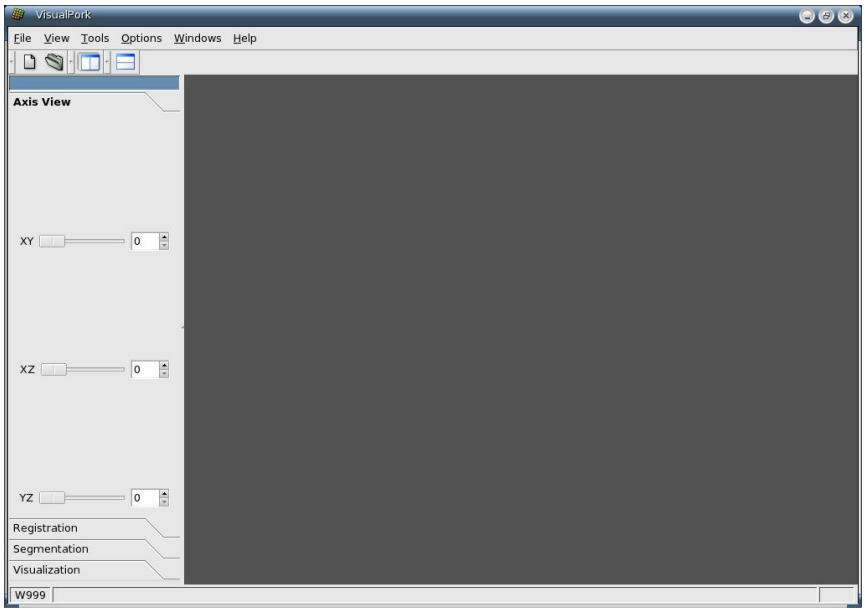


Figura 4.1: Interfície original de la plataforma de visualització.

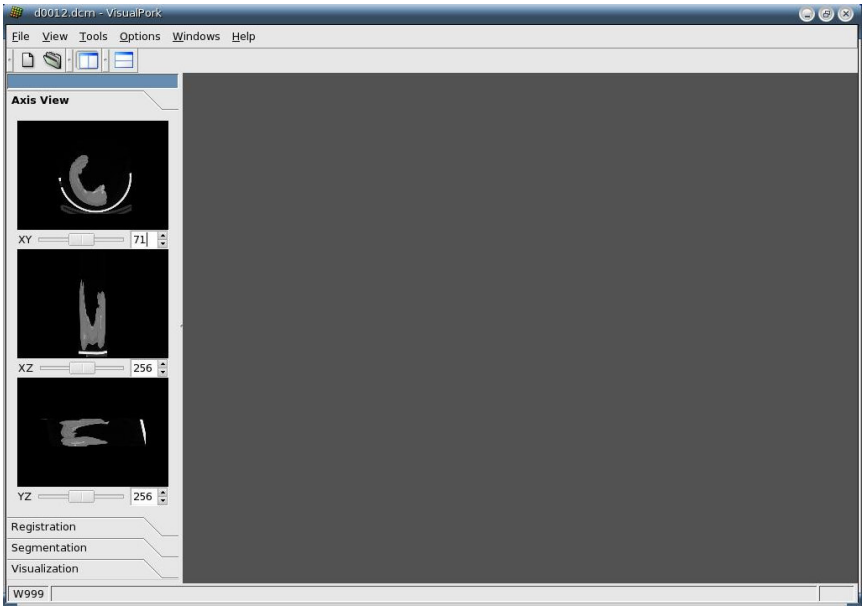


Figura 4.2: Una vegada obert el model, la plataforma ens mostra una llesca de cadascun dels tres eixos (Axial,Sagital i Coronal), i ens permet modificar el número de llesca.

4.2.- LLIBRERIES UTILITZADES

Per el desenvolupament de les nostres opcions de la plataforma (segmentació i visualització) únicament s'ha usat software de lliure distribució. Les eines i llibreries que he utilitzat per desenvolupar el projecte, i implementar aquestes tècniques han estat bàsicament:

- Llibreries Qt
- Llibreries OpenGL
- Llibreries VTK
- Llibreries ITK

4.2.1.- Llibreries Qt

Qt és una llibreria escrita en C++. Destaca per ser una llibreria molt completa i multi-plataforma. Permet desenvolupar amb un mateix codi versions per entorns Windows, Linux, Unix i Mac OS X. La versió per Unix és de lliure distribució sempre i quan no se'n comercialitzi el programa que la faci servir. La versió per Windows és plenament comercial però existeix una versió gratuïta. Qt destaca per ser la llibreria base de l'entorn de finestres KDE. Actualment existeixen varies eines per generar la interfície de forma visual. La pròpia llibreria en proporciona una, el *Qt Designer*. La totalitat de l'entorn gràfic que es pot observar en el programa ha estat implementat usant aquesta llibreria i més concretament fent servir el Qt Designer.

També disposàvem d'altres alternatives on escollir la eina de disseny de la interfície gràfica, malgrat tot, la majoria no són multi-plataforma. Si considerem indispensable que els programes generats es puguin executar tan en Windows com en sistemes Unix la nostre elecció queda limitada a tres eines: GLUT, GLUI i Qt.

- La llibreria GLUT disposa de molt pocs components gràfics. Bàsicament la llibreria està pensada per generar petites aplicacions de demostració de OpenGL.

- La llibreria GLUI complementa les carències de la llibreria GLUT però continua sent insuficient. Tot i ser molt senzilla de fer servir, resulta una llibreria poc optimitzada i només es pot integrar amb interfícies basades en GLUT.

- La llibreria Qt és una llibreria molt complerta. Proveeix de molts *widgets* i en permet la creació de nous. Té *widgets* específics per a la integració

amb OpenGL. Com a principals defectes té el fet de que no sigui completament de lliure distribució per a tots els sistemes operatius . Per altra banda està demostrat que és una llibreria robusta i fiable, amb un disseny completament orientat a objectes i que ofereix infinitat de possibilitats per crear interfícies. A més a més, disposa de nombrosos exemples, una documentació complerta i tutorials. Un cop escrit el codi, només cal recompil·lar-lo perquè funcioni en una altra plataforma. Tenint en compte totes aquestes consideracions la llibreria seleccionada ha estat la Qt.

L'eina principal de les Qt és el que anomenem *signals* i *slots*. Aquests ens serviran per establir les connexions entre els diversos elements gràfics que haguem dibuixat. És un sistema que serveix per a que els objectes de qualsevol tipus es comuniquin entre ells. Un *signal* és un avís que dóna un objecte quan ha passat algun succés (*event*), per exemple un botó pot emetre un *signal "clicked"* quan se'l prem. I un *slot* és aquella rutina que està esperant un senyal per executar-se, per exemple finalitzar l'execució d'una aplicació. De fet, la implementació d'un *slot* és com la d'un mètode qualsevol. Per connectar un *signal* amb un *slot* (o un altre *signal*) es crida la funció *connect()*. Aquest sistema de comunicació és probablement la característica que més diferencia les Qt de la resta de llibreries d'interfícies gràfiques. En l'esquema de la figura de continuació podem veure un exemple de comunicació *signal / slot*.

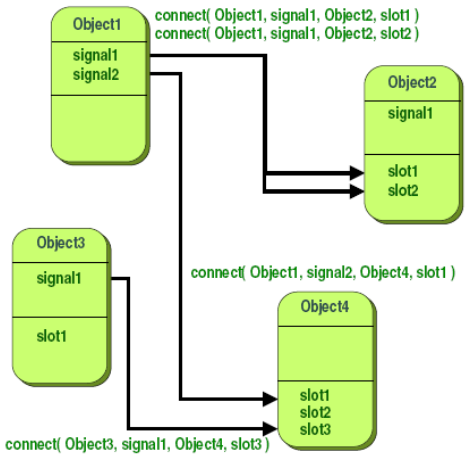


Figura 4.3: Vista abstracta d'algunes connexions entre signals i slots.

Es pot trobar més informació a: http://www.trolltech.com

4.2.2.- Llibreries OpenGL

OpenGL és la interfície software del hardware gràfic. Aquesta interfície consisteix en centenars de funcions que permeten als programadors gràfics especificar els objectes i les operacions que necessiten per produir imatges amb color d'alta qualitat d'objectes tridimensionals. Moltes d'aquestes funcions són simplement variacions de les altres, per tant, en realitat aquestes funcions es podrien dividir en unes 145.

OpenGL és un motor 3D les rutines del qual estan integrades en targetes gràfiques 3D. Disposa de totes les característiques necessàries per la representació mitjançant computadores de escenes 3D modelades amb polígons, des del pintat més bàsic de triangles, fins la representació de textures, il·luminació o NURBS.

Tal i com hem dit, el seu propòsit principal és representar objectes bidimensionals o tridimensionals en el *frame buffer*. Els objectes són descrits com una seqüència de vèrtexs (els quals defineixen objectes geomètrics) o píxels (defineixen imatges). OpenGL realitza una sèrie de passes sobre les dades per tal de convertir-les a píxels i formar la imatge que estem esperant en el frame buffer.

La OpenGL Utility Library (GLU) i la extensió de OpenGL al Sistema Window X (GLX) ens proporcionen eines força útils i complementen les funcions que ens ofereix OpenGL.

La companyia que desenvolupa aquesta llibreria és Sillicon Graphics Inc (SGI), en pro de fer un estàndard en la representació 3D gratuït i amb codi obert (open source). Està basat en els seus propis OS i llenguatges IRIS, de manera que és perfectament portable a altres llenguatges. Entre ells el C, C++, etc. i les llibreries dinàmiques permeten fer-lo servir sense problema en Visual Basic, Visual Fortran, Java, etc.

OpenGl suporta hardware 3D, i és altament recomanable disposar d'aquest tipus de hardware gràfic. Si no es té disposició de ell, les rutines de representació correran per software, enlloc de hardware, disminuint en gran mesura la seva velocitat.

Es pot trobar més informació a: http://www.opengl.org

4.2.3.- Llibreries VTK

VTK (Visualitzation Toolkit) és un kit "open-source", portable i orientat a objectes enfocat a gràfics 3D, visualització i processat d'imatges. Està implementat en C++ i ofereix la possibilitat de treballar-hi, a part de C++, amb Tcl, Python i Java. Està format per unes 700 llibreries escrites amb C++. Suporta molta varietat d'algoritmes de visualització incloent escalars, vector, tensor, texture, i mètodes volumètrics. Així com tècniques de modelització avançades com per exemple implicit modelling, cutting, contouring, Ray Casting, etc. A més a més, disposa de diversos algoritmes d'imatges integrats que permeten a l'usuari combinar imatges 2D, imatges 3D i dades. El disseny i la implementació de les llibreries ha estat seguint els principis bàsics de la metodologia orientada objectes. VTK ha estat instal·lat i testejat en tot tipus de plataformes existents actualment, com ara: Unix, PCs (Windows 98/ME/NT/2000/XP), i Mac OSX Jaguar o més recents.

Respecte el seu funcionament, disposa de 5 tipus bàsics d'objectes:

- **Dataset**: S'utilitza per guardar la informació inicial que volem visualitzar (per exemple, el fitxer d'entrada).

- **Filter**: Són les tècniques que s'apliquen sobre el volum. Aquests mètodes poden ser per exemple el Ray Casting, 2D Texture Mapping, Projecció de Màxima Intensitat, Splatting, etc. Aquest objecte és la part més important en tot aquest funcionament (és el que realment executa el procés).

- **Mapper**: S'utilitza per guardar la sortida del Filter. Per tant, el seu objectiu principal és el d'emmagatzemar les dades que llavors haurem de mostrar per pantalla.

- **Actor**: Una vegada construït el mapper el següent pas serà el de traslladar la informació en el actor. Sobre l'actor podem realitzar operacions com ara rotar, traslladar, escalar, etc.

- **Renderer**: Finalment cal representar els actors per la pantalla. Per tant, és l'objecte que realitza l'últim pas avanç de veure el resultat final.

Tots aquests objectes que acabem d'explicar es relacionen seguint l'ordre del mètode de treball. Aquest mètode de treball s'anomena processat **pipeline**, i consisteix en repartir el procés de visualització en vàries etapes:

1. Etapa Get Source.
2. Etapa Filtering.
3. Etapa Mapping.
4. Etapa Rendering.

La primera etapa, el **Get Source**, és el que converteix les dades originals del model a dades VTK. Per fer aquesta conversió ha de tenir present quin tipus de dades VTK s'utilitzaran.

La segona etapa, el **Filtering**, és la que aplica els mètodes sobre les dades (aplica el filter que hem vist anteriorment). Aquesta part és la més important dins el procés.

La tercera etapa, el **Mapping**, és on guardarem les dades transformades (resultat del Filtering) en mappers. El tipus de mapper varia depenent de les dades de sortida.

Finalment tenim l'etapa de **Rendering** on visualitzarem el resultat final per pantalla. Aspectes com la càmera, les llums, etc. també intervenen en aquesta etapa.

L'esquema d'aquestes etapes el podem veure a la figura següent.

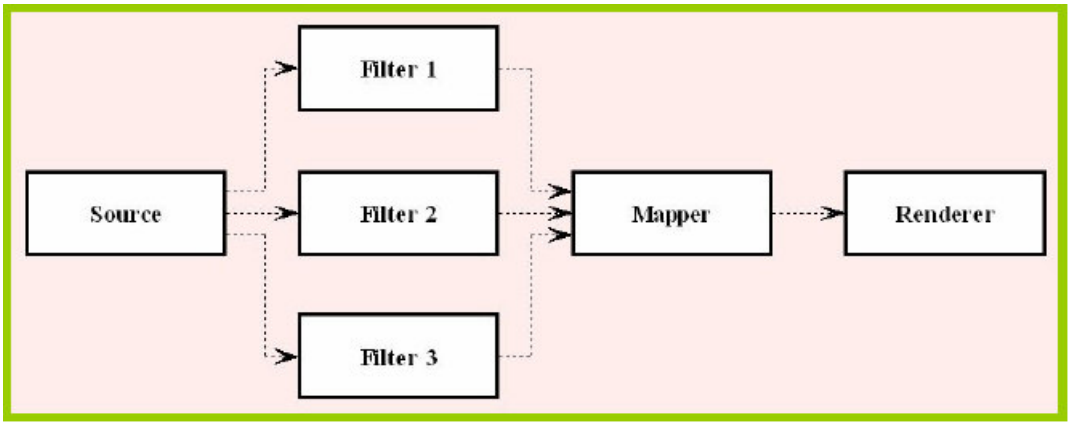


Figura 4.4: Procés visualització mitjançant mètode Pipeline.

Totes aquestes etapes de VTK disposen d'entrades i sortides, de tal manera que podem enllaçar-les i anar formant el processat pipeline. Per enllaçar dues classes, a la classe d'entrada hi apliquem la funció *SetInput()* i per obtenir la sortida utilitzem el *GetOutput().*

Per últim, un aspecte molt important que ens ofereixen les llibreries VTK és la seva connexió amb les llibreries Qt abans descrites. Aquesta integració es fa mitjançant la classe vtkRendererWindow. Bàsicament el que es fa és crear un widget amb Qt que rep les propietats de vtkRendererWindow. Així podrem tenir una aplicació dissenyada en Qt on podem visualitzar dades VTK.

Es pot trobar més informació a: http://public.kitware.com/VTK

4.2.4.- Llibreries ITK

L'eina utilitzada per a poder fer segmentacions, ha de poder oferir diferents mètodes de segmentació podent canviar els seus atributs per a obtenir un resultat idoni. Una de les eines més utilitzades per a segmentar tant imatges com volums, és el conjunt de llibreries ITK.

Les ITK (*Insight Toolkit*) són llibreries *open source* per a realitzar segmentacions i altres operacions com registre, etc. També conté eines per llegir imatges adquirides des de sistemes d'adquisició de dades, com CT o escàners MRI, i implementacions de diferents filtres amb la finalitat de, per exemple, reduir el soroll d'un model. A més a més està implementat amb C++ i conté gran quantitat d'exemples i documentació.

El funcionament d'aquestes llibreries és el mateix que les llibreries VTK explicades amb anterioritat.

Es pot trobar més informació a: http://www.itk.org

Implementació

Requeriments del sistema

En aquest capítol del projecte descriurem els requeriments del sistema. Aquests requeriments ens definiran els objectius de l'aplicació juntament amb les funcionalitats que l'usuari pot realitzar. Aquest apartat ens ha de permetre entendre els elements que "envoltaran" el sistema informàtic que hem implementat. Aquests elements fan referència a les persones que interactuaran amb el sistema, a les funcionalitats que oferirà la plataforma, al software o programari utilitzat per efectuar la implementació (afecta a la portabilitat de la plataforma), etc. Com veurem existeixen dues classes de requeriments:

- **Requeriments funcionals:** Descriuen els serveis o funcionalitats que ofereix l'aplicació independentment de la implementació. Per definir-los utilitzarem:
    - Diagrames de casos d'ús
    - Fitxes de casos d'ús

- **Requeriments no funcionals:** Descriuen les restriccions que venen imposades per el client o per el propi problema. Aquestes restriccions són els terminis, pressupostos, programari de desenvolupament, seguretat, portabilitat, etc.

Per descriure aquests requeriments usarem la metodologia UML que ens permetrà construir els diagrames necessaris, a partir dels requeriments funcionals de l'aplicació. Amb UML podrem indicar les funcionalitats de l'aplicació de manera entenedora.

4.3.- REQUERIMENTS FUNCIONALS

En aquest apartat descriurem els serveis o principals funcionalitats que ha d'oferir la plataforma.

Obertura d'un model porcí

El sistema ha d'oferir la possibilitat d'obrir el model de porc amb el que desitgem treballar i visualitzar-lo a la plataforma. A més a més ens ha de permetre desplaçar-nos per les diferents llesques que formen el model.

Segmentació d'un model porcí

El sistema ha d'oferir la possibilitat de segmentar aquest model mitjançant diferents tècniques de segmentació. Per cada una de les tècniques hem d'oferir la possibilitat a l'usuari de configurar els paràmetres, de tal manera que pugui segmentar la regió que més l'interessi. Concretament les tècniques implementades han estat: Connected Threshold, Neighborhood Connected, Confidence Connected i Isolated Connected.

Manipulació dels resultats de la segmentació

El sistema ha d'oferir la possibilitat de visualitzar i manipular els resultats de les tècniques de segmentació. D'una banda, hem de mostrar el model resultant amb 2 dimensions i disposar de la possibilitat de modificar el punt de vista (Axial, Sagital o Coronal). D'altra banda ha d'existir l'opció de visualitzar el resultat amb 3 dimensions mitjançant les tècniques de Ray Casting i Multiplanar.

Finalment també hem de mostrar les dades numèriques del resultat, de manera que permeti a l'usuari observar el volum de la part segmentada i el percentatge que representa aquest volum respecte la totalitat del model.

Visualització d'un model porcí

El sistema també ha d'oferir la possibilitat de visualitzar el model amb 3 dimensions mitjançant diferents tècniques de visualització. Al igual que amb la segmentació, també hem d'oferir la possibilitat a l'usuari de configurar els paràmetres d'aquestes tècniques. Les dues tècniques implementades han estat: Ray Casting i Multiplanar.

Manipulació dels resultats de la visualització

El sistema ha d'oferir la possibilitat de manipular els resultats de les tècniques de visualització Ray Casting i Multiplanar. Aquestes dues tècniques han de permetre una sèrie d'accions comunes com són:

- Realitzar zoom sobre el model.
- Rotar el model.

A més a més, en el cas de la visualització Ray Casting ha d'existir l'opció d'utilitzar una eina que s''anomena *Box Widget.* Aquesta utilitat permet que l'usuari pugui manipular el resultat i explorar el model amb més detall.

En el cas de la visualització Multiplanar ha d'existir l'opció de desplaçar-se per les diferents vistes del model. D'aquesta manera l'usuari podrà visualitzar qualsevol de les llesques de les vistes Axial, Sagital o Coronal.

4.3.1.- IDENTIFICACIÓ DELS ACTORS

Un actor és una entitat externa (persona, dispositiu, subsistema, ...) que interactua amb el sistema interpretant un determinat rol. Habitualment existeixen diferents preguntes per decidir i determinar els actors que intervindran en l'aplicació. En el nostre cas no ens és necessari realitzar distinció entre els possibles usuaris que la utilitzaran, ja que tant sols existeix un únic actor, l'usuari que interactua en tot moment amb el sistema.

4.3.2.- DIAGRAMES I FITXES DE CASOS D'ÚS

Els diagrames de casos d'ús permeten mostrar de manera visual com interactua l'usuari amb l'aplicació i les accions que pot realitzar. D'altra banda, les fitxes de casos d'ús especifiquen amb més detall el funcionament de cada cas d'ús. Aquestes estan formades per una breu explicació de què fa el cas d'ús analitzat, l'actor que pot utilitzar-lo, la precondició que ha de complir, el flux general que segueix l'aplicació per resoldre'l, la postcondició i si existeix algun flux alternatiu o observació a tenir en compte.

Primer de tot observarem el diagrama de cas d'ús de context on veurem totes les operacions que l'usuari pot realitzar amb l'aplicació. A continuació per a entendre millor els diferents requeriments, separarem els diagrames de casos d'ús i les fitxes, segons l'àmbit al qual pertanyen, és a dir, separarem segons accions que ens permeten obrir el model, seleccionar la tècnica i els paràmetres, o manipular els resultats.

4.3.2.1.- Diagrama de cas d'ús de context

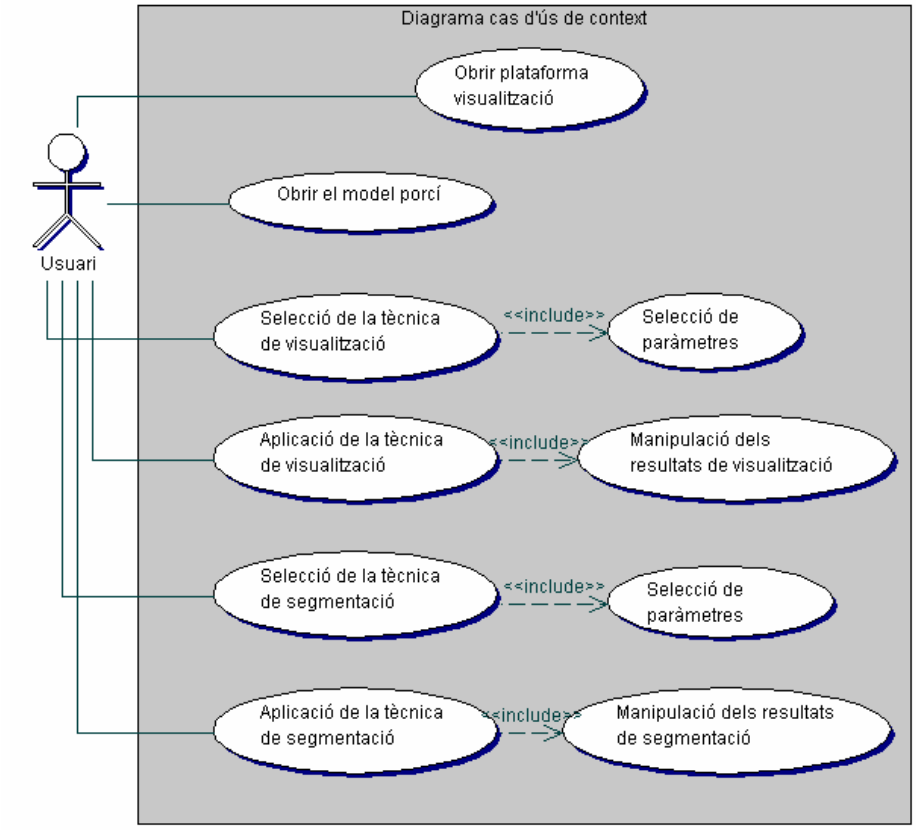


Figura 4.5: Diagrama de cas d'ús de context amb totes les operacions que pot realitzar l'usuari.

La primera acció que ha de realitzar l'usuari és obrir la plataforma de visualització, i tot seguit obrir el model porcí. A continuació disposa d'una sèrie de funcionalitats que pot dur a terme: aplicar una tècnica de segmentació, o bé, una tècnica de visualització. Per cadascun dels casos, primer haurà de seleccionar la tècnica corresponent i tot seguit introduir els paràmetres. Finalment ha d'escollir l'opció d'aplicar la tècnica, de manera que li apareixerà la nova funcionalitat de manipulació de resultats.

En els diagrames de continuació explicarem amb més detall els casos d'ús més destacats:

- Obertura del model porcí.
- Selecció de la tècnica de visualització, que inclourà selecció dels paràmetres de les tècniques de visualització: Ray Casting i Multiplanar.
- Manipulació dels resultats de la visualització, que inclourà: manipulació dels resultats de Ray Casting i manipulació dels resultats de Multiplanar.

- Selecció de la tècnica de segmentació, que inclourà selecció dels paràmetres de les tècniques de segmentació: Connected Threshold, Neighborhood Connected, Confidence Connected i Isolated Connected.

- Manipulació dels resultats de la segmentació.

4.3.2.2- Obertura del model porcí

Aquest primer requeriment és el que fa referència a com obrir a la plataforma el model de porc amb el que desitgem treballar.

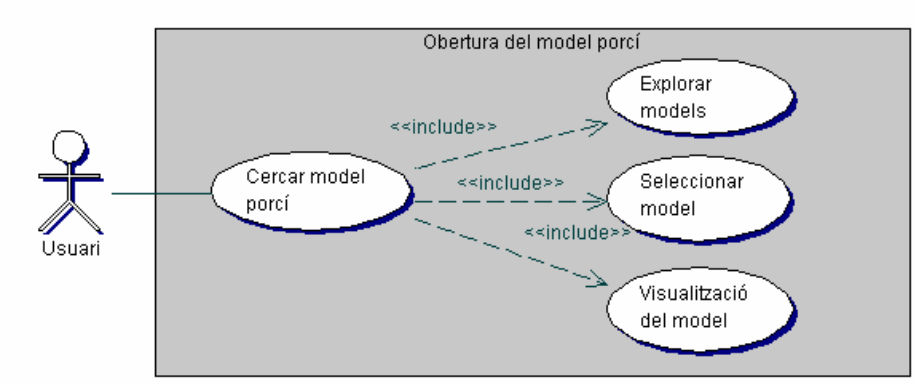


Figura 4.6: Cas d'ús d'obertura del model porcí.

Tal com es mostra en el diagrama anterior, les funcionalitats s'agrupen en una acció fonamental: cercar el model porcí, que agrupa les següents sub-accions:

- Explorar els diferents models disponibles.
- Seleccionar el model desitjat.
- Visualització del model.

Cas d'ús	**Obertura del model porcí**
Descripció:	Permet cercar el model desitjat, mitjançant un explorador gràfic.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada.
Flux principal:	1. Activem l'opció d'obertura de models. 2. Exploració dels diferents models existents dins del directori seleccionat. 3. Selecció del model porcí desitjat. 4. Una vegada seleccionat el model, aquest serà visualitzat a l'aplicació.
Flux alternatiu:	-
Postcondició:	El model ha estat seleccionat i visualitzat.
Observacions:	-

4.3.2.3.- Selecció de la tècnica de visualització

Aquest requeriment fa referència al procés per seleccionar la tècnica de visualització que l'usuari desitja executar.

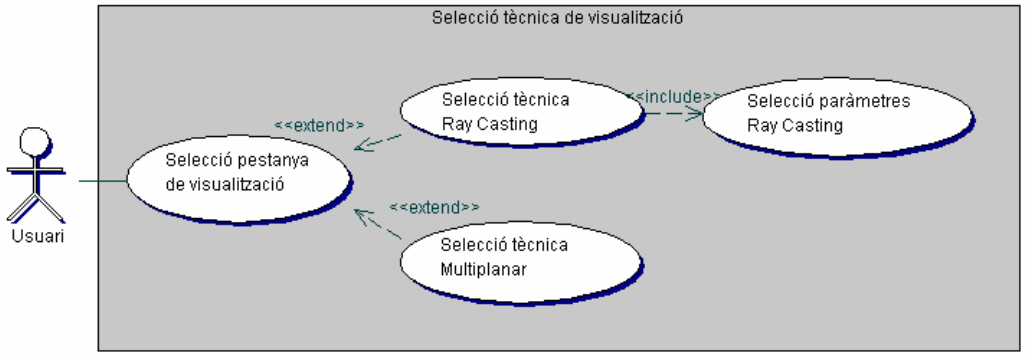


Figura 4.7: Cas d'ús de selecció de la tècnica de visualització.

Cas d'ús	**Selecció de la tècnica de visualització**
Descripció:	Permet seleccionar la tècnica de visualització a executar.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada.
Flux principal:	1. L'usuari selecciona la pestanya de visualització. 2. L'usuari disposa de 2 possibilitats: ▪ Seleccionar la tècnica de Ray Casting. (punt 3) ▪ Seleccionar la tècnica Multiplanar. 3. En el cas de la tècnica de Ray Casting l'usuari ha de seleccionar els paràmetres per executar la tècnica.
Flux alternatiu:	-
Postcondició:	L'usuari ha seleccionat la tècnica de visualització i ha escollit els paràmetres.
Observacions:	-

Tal com es detalla a la fitxa de cas d'ús anterior, la selecció de la tècnica de Ray Casting inclou també el cas d'ús de selecció de paràmetres. Per contra, no succeeix el mateix amb la tècnica Multiplanar. Aquesta no disposa de cap tipus de paràmetre, de manera que, per realitzar la seva execució únicament ens serà necessari seleccionar la tècnica dins la pestanya de visualització, i a continuació procedir a la seva execució.

Tot seguit explicarem amb més detall el cas d'ús corresponent a la selecció de paràmetres de la tècnica de Ray Casting.

Selecció paràmetres Ray Casting

Aquest requeriment fa referència al procés per seleccionar els paràmetres de la tècnica de visualització de Ray Casting.

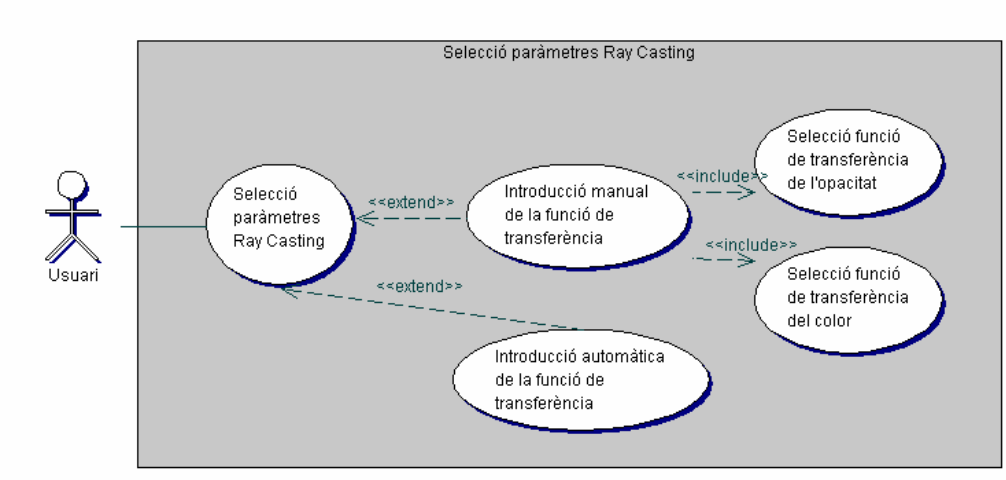


Figura 4.8: Cas d'ús selecció de paràmetres del Ray Casting.

Les funcionalitats s'agrupen en una acció fonamental: selecció dels paràmetres de Ray Casting, que agrupa les següents sub-accions:

- Introducció manual de la funció de transferència, que inclou: selecció de la funció de transferència de l'opacitat i selecció de la funció de transferència del color.

- Introducció automàtica de la funció de transferència.

Cas d'ús	**Selecció paràmetres Ray Casting**
Descripció:	Permet seleccionar els paràmetres de la tècnica Ray Casting.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i la tècnica de Ray Casting ha estat seleccionada.
Flux principal:	1. L'usuari disposa de 2 possibilitats: a. Utilitzar la funció de transferència automàtica. b. Introduir la funció de transferència manualment. (punt 2) 2. L'usuari selecciona la funció de transferència de l'opacitat, és a dir, associa rangs i opacitats. 3. L'usuari selecciona la funció de transferència del color, és a dir, associa rangs i colors.
Flux alternatiu:	-
Postcondició:	L'usuari ha especificat els paràmetres de la tècnica de Ray Casting.
Observacions:	-

4.3.2.4.- Manipulació dels resultats de la visualització

Aquest requeriment fa referència al procés de manipulació de resultats de les tècniques de visualització.

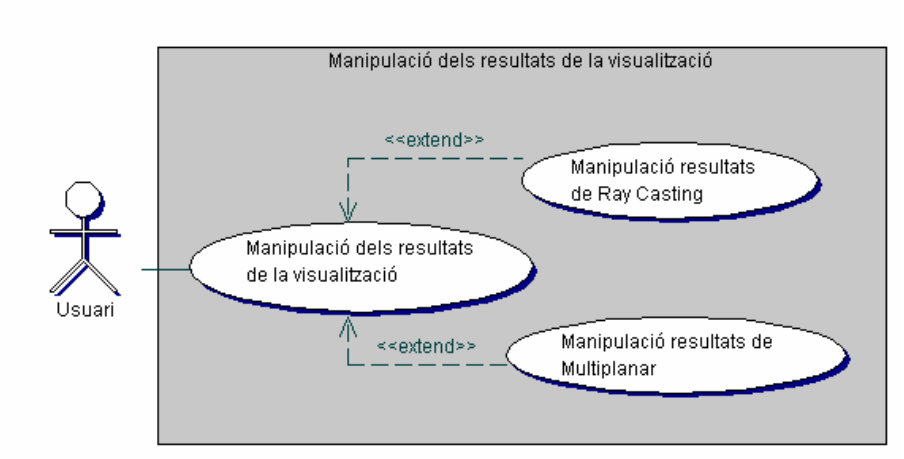


Figura 4.9: Cas d'ús manipulació dels resultats de la visualització.

Les funcionalitats s'agrupen en una acció fonamental: manipulació dels resultats de la visualització, que agrupa les següents sub-accions:

- Manipulació dels resultats de Ray Casting.
- Manipulació dels resultats de Multiplanar.

Cas d'ús	**Manipulació dels resultats de la visualització**
Descripció:	Permet manipular els resultats de les tècniques de visualització.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i s'ha d'haver executat una tècnica de visualització.
Flux principal:	1. Si l'usuari ha executat la tècnica Ray Casting s'aplicarà el cas d'ús manipulació de resultats de Ray Casting. 2. Si l'usuari ha executat la tècnica Multiplanar s'aplicarà el cas d'ús manipulació de resultats de Multiplanar.
Flux alternatiu:	-
Postcondició:	El model resultat ha estat manipulat.
Observacions:	-

A continuació explicarem amb més detall el funcionament dels casos d'ús manipulació de resultats de Ray Casting i manipulació de resultats de Multiplanar.

Manipulació de resultats de Ray Casting

Aquest requeriment fa referència al procés de manipulació dels resultats de la tècnica de visualització de Ray Casting.

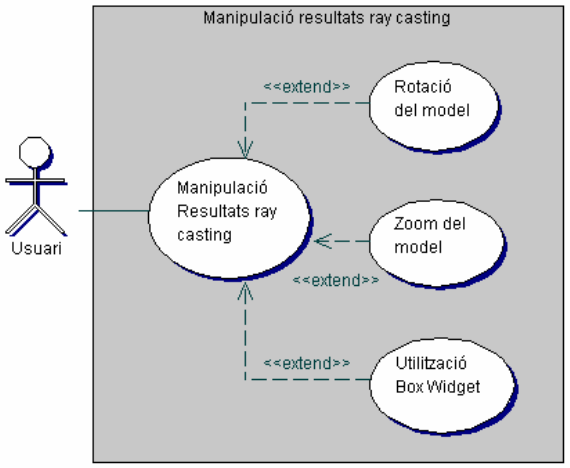


Figura 4.10: Cas d'ús manipulació dels resultats de Ray Casting.

Les funcionalitats s'agrupen en una acció fonamental: **manipulació dels resultats de Ray Casting**, que agrupa les següents sub-accions:

- Rotació del model.
- Zoom del model.
- Utilització del Box Widget.

Cas d'ús	**Manipulació resultats Ray Casting**
Descripció:	Permet manipular els resultats de la tècnica de visualització de Ray Casting.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i la tècnica de visualització de Ray Casting s'ha d'haver executat.
Flux principal:	1. L'usuari pot rotar el model 3D. 2. L'usuari pot realitzar zooms sobre el model 3D. 3. L'usuari pot utilitzar el Box Widget per explorar internament el model 3D.
Flux alternatiu:	-
Postcondició:	L'usuari ha pogut manipular el resultat de la tècnica de visualització Ray Casting.
Observacions:	-

Manipulació dels resultats de Multiplanar

Aquest requeriment fa referència al procés de manipulació dels resultats de la tècnica de visualització Multiplanar.

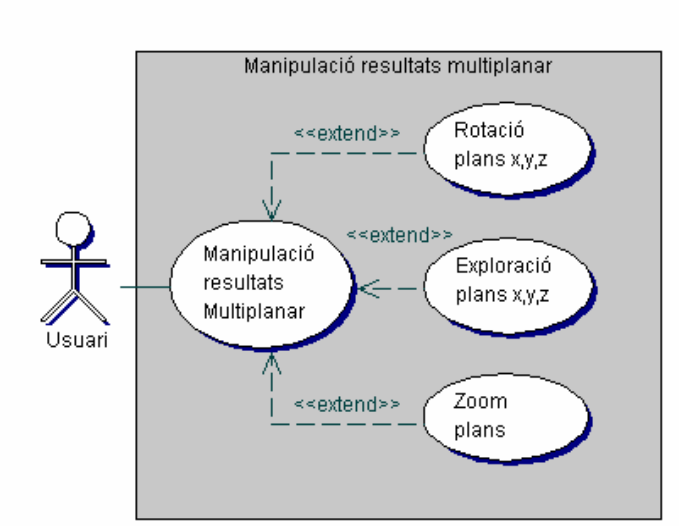


Figura 4.11: Cas d'ús manipulació dels resultats de Multiplanar.

Les funcionalitats s'agrupen en una acció fonamental: **manipulació dels resultats Multiplanar**, que agrupa les següents sub-accions:

- Rotació plans x,y,z.
- Exploració plans x,y,z.
- Zoom dels plans.

Cas d'ús	**Manipulació resultats Multiplanar**
Descripció:	Permet manipular els resultats de la tècnica de visualització Multiplanar.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i la tècnica de visualització Multiplanar s'ha d'haver executat.
Flux principal:	4. L'usuari pot rotar els plans x,y,z del model resultant. 5. L'usuari pot explorar els plans x,y,z del model resultant. 6. L'usuari pot realitzar zoom dels plans del model resultant.
Flux alternatiu:	-
Postcondició:	L'usuari ha pogut manipular el resultat de la tècnica de visualització Multiplanar.
Observacions:	-

4.3.2.5.- Selecció de la tècnica de segmentació

Aquest requeriment fa referència al procés per seleccionar la tècnica de segmentació que l'usuari desitja executar.

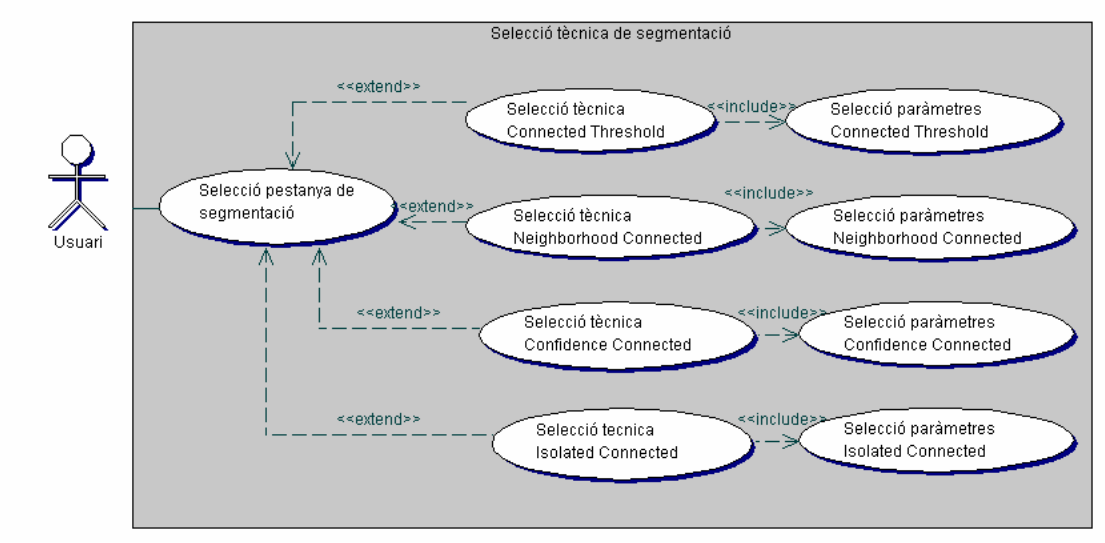


Figura 4.12: Cas d'ús de selecció de la tècnica de segmentació

Cas d'ús	**Selecció de la tècnica de segmentació**
Descripció:	Permet seleccionar la tècnica de segmentació a executar.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada.
Flux principal:	1. L'usuari selecciona la pestanya de segmentació. 2. L'usuari disposa de 4 possibilitats: a. Seleccionar la tècnica Connected Threshold. b. Seleccionar la tècnica Neighborhood Connected. c. Seleccionar la tècnica Confidence Connected. d. Seleccionar la tècnica Isolated Connected. 3. L'usuari ha de seleccionar els paràmetres corresponents per executar la tècnica.
Flux alternatiu:	-
Postcondició:	L'usuari ha seleccionat la tècnica de visualització i ha escollit els paràmetres corresponents.
Observacions:	-

A continuació detallarem el funcionament dels casos d'ús corresponents a la selecció dels paràmetres de cada tècnica de segmentació.

Selecció paràmetres Connected Threshold

Aquest requeriment fa referència al procés per seleccionar els paràmetres de la tècnica de segmentació del Connected Threshold.

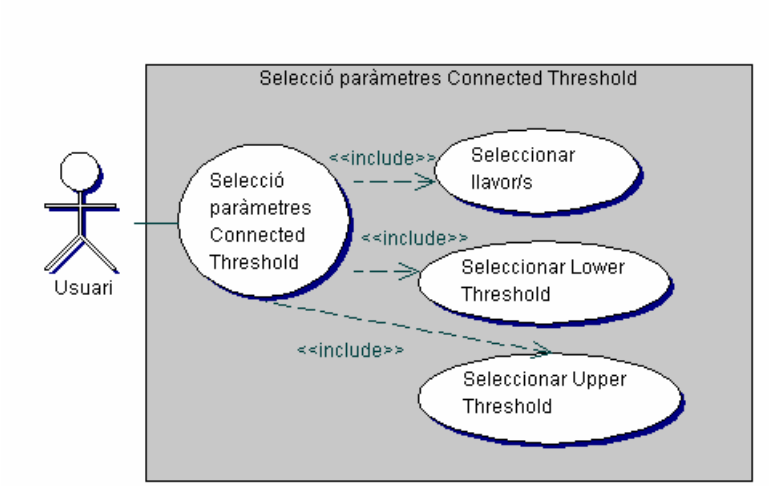


Figura 4.13: Cas d'ús selecció de paràmetres del Connected Threshold.

Les funcionalitats s'agrupen en una acció fonamental: **selecció dels paràmetres del Connected Threshold**, que agrupa les següents sub-accions:

- Selecció de la llavor/s a utilitzar.
- Selecció del Lower Threshold (llindar inferior).
- Selecció del Upper Threshold (llindar superior).

Cas d'ús	**Selecció paràmetres Connected Threshold**
Descripció:	Permet seleccionar els paràmetres de la tècnica Connected Threshold.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i la tècnica Connected Threshold ha estat seleccionada.
Flux principal:	1. L'usuari selecciona la llavor o llavors que vol utilitzar. 2. L'usuari selecciona el Lower Threshold o llindar inferior. 3. L'usuari selecciona el Upper Threshold o llindar superior.
Flux alternatiu:	-
Postcondició:	L'usuari ha especificat els paràmetres de la tècnica de Connected Threshold.
Observacions:	-

Selecció paràmetres Neighborhood Connected

Aquest requeriment fa referència al procés per seleccionar els paràmetres de la tècnica de segmentació de Neighborhood Connected.

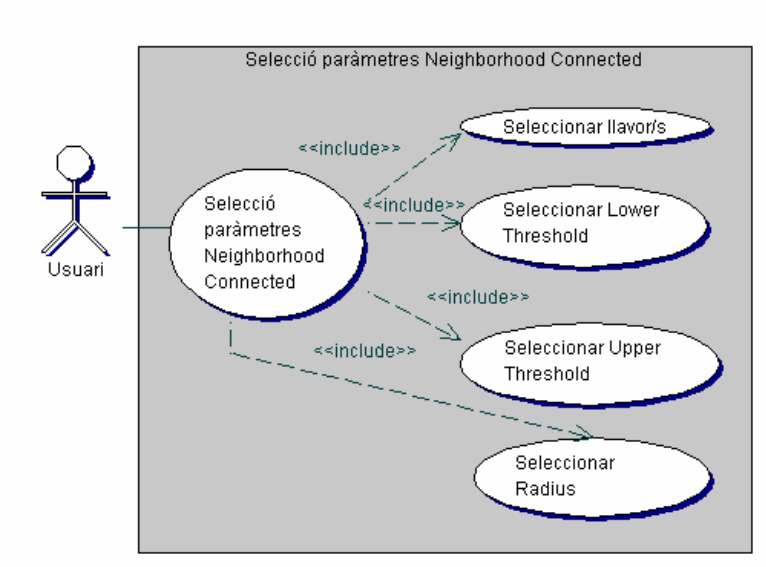


Figura 4.14: Cas d'ús selecció de paràmetres del Neighborhood Connected.

Les funcionalitats s'agrupen en una acció fonamental: **selecció dels paràmetres del Neighborhood Connected**, que agrupa les següents sub-accions:

- Selecció de la llavor/s a utilitzar.
- Selecció del Lower Threshold (llindar inferior).
- Selecció del Upper Threshold (llindar superior).
- Selecció del radi dels veïns.

Cas d'ús	**Selecció paràmetres Neighborhood Connected**
Descripció:	Permet seleccionar els paràmetres de la tècnica Neighborhood Connected.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i la tècnica Neighborhood Connected ha estat seleccionada.
Flux principal:	1. L'usuari selecciona la llavor o llavors que vol utilitzar. 2. L'usuari selecciona el Lower Threshold o llindar inferior. 3. L'usuari selecciona el Upper Threshold o llindar superior. 4. L'usuari selecciona el radi dels veïns.
Flux alternatiu:	-
Postcondició:	L'usuari ha especificat els paràmetres de la tècnica de Neighborhood Connected.
Observacions:	-

Selecció paràmetres Confidence Connected

Aquest requeriment fa referència al procés per seleccionar els paràmetres de la tècnica de segmentació de Confidence Connected.

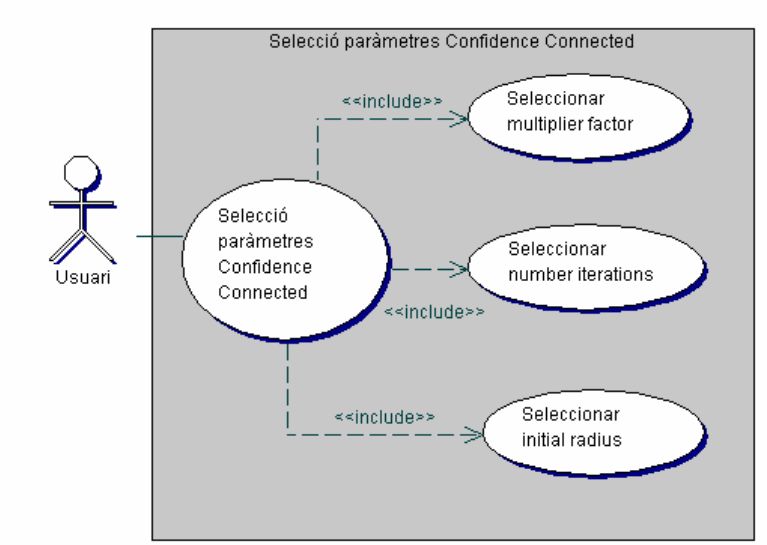


Figura 4.15: Cas d'ús selecció de paràmetres del Confidence Connected.

Les funcionalitats s'agrupen en una acció fonamental: **selecció dels paràmetres del Confidence Connected**, que agrupa les següents sub-accions:

- Selecció del factor multiplicador.
- Selecció del nombre d'iteracions.
- Selecció del radi inicial.

Cas d'ús	**Selecció paràmetres Confidence Connected**
Descripció:	Permet seleccionar els paràmetres de la tècnica Confidence Connected.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i la tècnica Confidence Connected ha estat seleccionada.
Flux principal:	1. L'usuari selecciona el factor multiplicador. 2. L'usuari selecciona el nombre d'iteracions. 3. L'usuari selecciona el radi inicial.
Flux alternatiu:	-
Postcondició:	L'usuari ha especificat els paràmetres de la tècnica de Confidence Connected.
Observacions:	-

Selecció paràmetres Isolated Connected

Aquest requeriment fa referència al procés per seleccionar els paràmetres de la tècnica de segmentació de Isolated Connected.

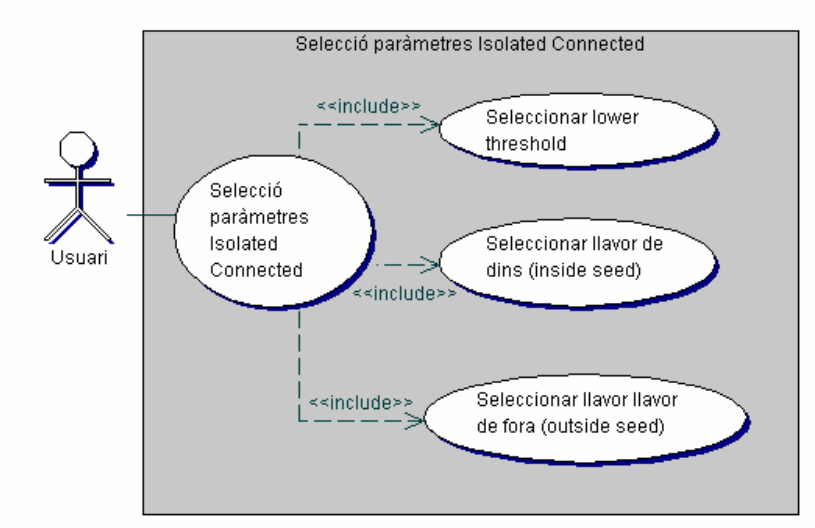


Figura 4.16: Cas d'ús selecció de paràmetres del Isolated Connected.

Les funcionalitats s'agrupen en una acció fonamental: **selecció dels paràmetres del Isolated Connected**, que agrupa les següents sub-accions:

- Selecció del Lower Threshold (llindar inferior).
- Selecció de la llavor interior (inside seed).
- Selecció de la llavor exterior (outside seed).

Cas d'ús	**Selecció paràmetres Isolated Connected**
Descripció:	Permet seleccionar els paràmetres de la tècnica Isolated Connected.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i la tècnica Isolated Connected ha estat seleccionada.
Flux principal:	1. L'usuari selecciona el Lower Threshold. 2. L'usuari selecciona la llavor interior. 3. L'usuari selecciona la llavor exterior.
Flux alternatiu:	-
Postcondició:	L'usuari ha especificat els paràmetres de la tècnica de Isolated Connected.
Observacions:	-

4.3.2.6.- Manipulació dels resultats de la segmentació

Aquest requeriment fa referència al procés de manipulació de resultats de les diferents tècniques de segmentació.

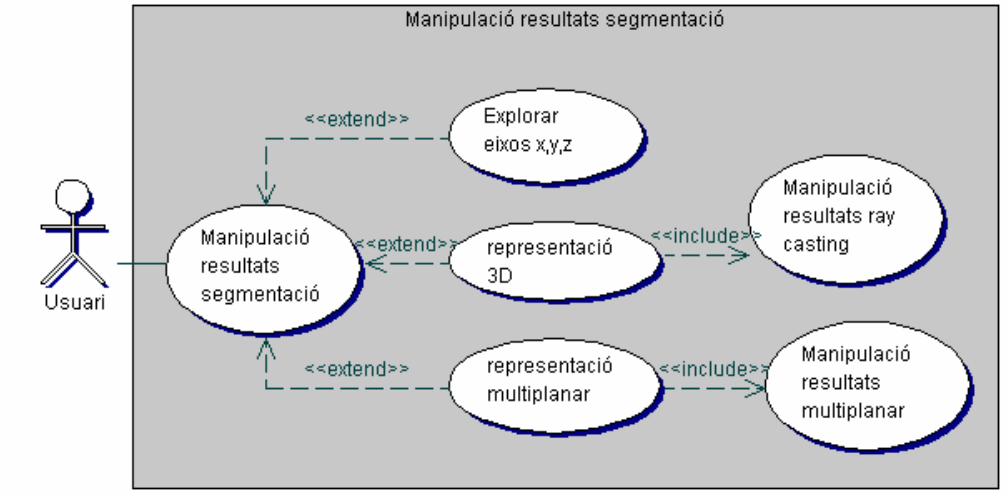


Figura 4.17: Cas d'ús manipulació dels resultats de segmentació.

Les funcionalitats s'agrupen en una acció fonamental: **manipulació dels resultats de segmentació**, que agrupa les següents sub-accions:

- Explorar els eixos x,y,z del model resultant.
- Representar el resultat en 3D mitjançant el Ray Casting.
- Representar el resultat mitjançant la visualització Multiplanar.

Cas d'ús	**Manipulació resultats segmentació**
Descripció:	Permet manipular els resultats de les tècniques de segmentació.
Actors:	Usuari.
Precondició:	L'aplicació ha d'estar iniciada i alguna tècnica de segmentació s'ha d'haver executat.
Flux principal:	1. L'usuari pot explorar els eixos x,y,z del volum resultant. 2. L'usuari pot obtenir una representació 3D amb Ray Casting del volum resultant. 3. L'usuari pot obtenir una representació Multiplanar del volum resultant.
Flux alternatiu:	-
Postcondició:	L'usuari ha pogut manipular el resultat de la tècnica de segmentació.
Observacions:	-

4.4.- REQUERIMENTS NO FUNCIONALS

En aquest apartat prestarem especial atenció a aquells aspectes que s'han de tenir en compte en el moment de dissenyar el sistema, més enllà de l'especificació funcional detallada en el punt anterior. En termes generals es diu que els requeriments no funcionals són restriccions, és a dir, són aspectes que fan referència a requisits de l'estil: temps de resposta, disponibilitat de recursos, seguretat, manteniment, software a utilitzar per la implementació, etc.

En el nostre cas únicament existirà un requeriment no funcional: la utilització de les llibreries explicades anteriorment en l'apartat 4.2 d'aquesta memòria. Un dels requeriments és que el projecte es desenvolupi amb software de lliure distribució i multiplataforma. Les 4 llibreries utilitzades (Qt, Itk, Vtk i OpenGL) compleixen ambdues condicions.

Implementació

Disseny del sistema

En aquest capítol tractarem la fase de disseny o modelatge conceptual, que ens servirà per explicar el disseny del sistema informàtic, és a dir, la nova estructura que hem construït per ampliar, amb les noves funcionalitats definides anteriorment, l'aplicació de la qual partim.

Com ja hem comentat, per a implementar aquest projecte, es parteix d'una aplicació de visualització i manipulació d'imatges. El fet d'haver d'integrar-ho a un sistema existent, implicarà ajustar-nos a unes normes i a una metodologia de programació, així com adaptar-nos a una estructura de classes.

El primer aspecte que veurem és **l'arquitectura del sistema del qual partim**. Analitzarem la seva estructura de classes i explicarem la funció de cada una d'aquestes. També mostrarem les **passes a seguir per afegir nous mètodes** en aquesta plataforma.

A continuació mostrarem el disseny i l'estructura de les classes encarregades d'executar funcions més concretes. D'aquesta manera, analitzarem els mòduls que hem creat que intervenen en la implementació de la **interfície gràfica,** així com els diferents mòduls que s'ocupen de la **implementació d'una tècnica** de segmentació o visualització.

Finalment ens centrarem amb els **diagrames d'activitat i de seqüència** que ens documentaran la relació entre classes amb la finalitat d'obtenir un disseny més detallat.

4.5.- ARQUITECTURA DE LA PLATAFORMA INICIAL

A continuació mostrarem la documentació necessària per a comprendre el funcionament de la plataforma de la qual partíem. D'aquesta manera tindrem una visió de la seva estructura i entendrem el perquè del seu disseny. En la figura de sota observem el disseny de classes de la plataforma inicial.

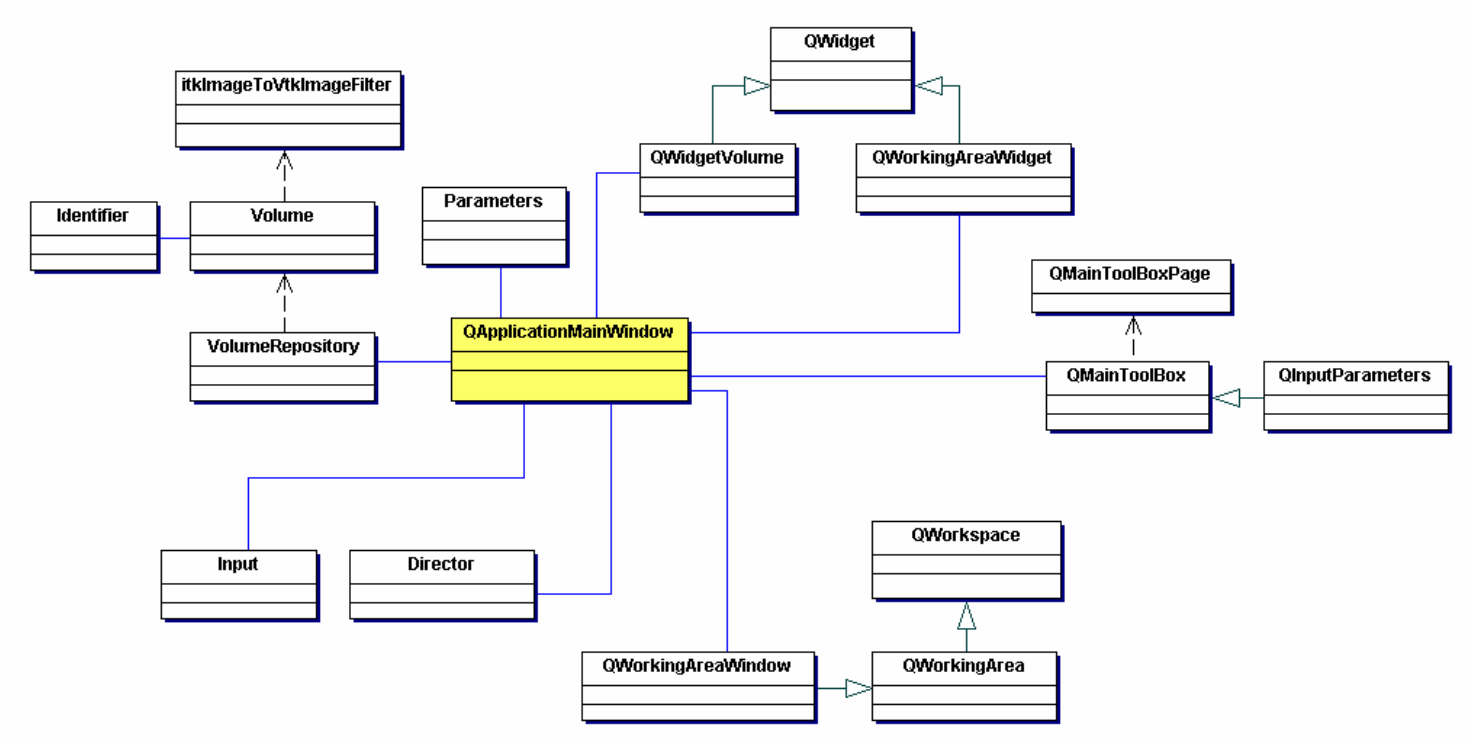


Figura 4.18: Disseny de classes de la plataforma inicial.

Classe principal i tractament de volums

La finestra principal de l'aplicació està definida en la classe QApplicationMainWindow. Aquesta conté tot el que es pot esperar d'una finestra principal: menús, barres d'eines, barres d'estat, una àrea de treball, etc. A més de tots els elements bàsics d'una interfície d'usuari, també conté les classes necessàries per obrir models, emmagatzemar-los, etc.

Els models queden encapsulats en la classe Volume. Aquesta classe ens pot proporcionar les dades del model en format Itk o Vtk segons necessitem, a través dels mètodes *getItkData()* i *getVtkData()*.

Per cada model que obrim tindrem una QApplicationMainWindow oberta. Això no vol dir que aquella finestra sigui la "propietària" exclusiva del model obert. Si volem podem accedir a qualsevol model des d'una altra finestra. Això és així perquè els models que s'obren s'emmagatzemen en un repositori comú. El repositori està implementat en la classe VolumeRepository. Quan obrim un nou model amb l'aplicació, aquest s'afegeix al repositori, el qual ens retorna un identificador per poder fer referència. Cada finestra doncs, guarda l'identificador del model que té obert, accessible des del mètode *getVolumeID()*. L'identificador està definit per la classe Identifier.

Per a totes les finestres que obrim només existirà una instància del repositori ja que s'aplica el patró de disseny *Singleton*, en el qual el repositori ve a ser com una "variable global" accessible en qualsevol moment. Si necessitem accedir al repositori, ja sigui per afegir un volum, obtenir-lo o eliminar-lo, l'únic que caldrà serà cridar el mètode *VolumeRepository::getRepository()* i obtindrem la instància de l'objecte VolumeRepository.

El ToolBox

El toolbox, implementat en la classe QMainToolBox, agrupa els formularis principals per escollir la tècnica de segmentació, visualització, etc. a aplicar sobre el model i els respectius paràmetres. El toolbox es divideix en pestanyes que agrupen segmentació, i visualització. A més a més també disposa d'un visor encastat de les vistes Axial, Coronal i Sagital del model.

El QMainToolBox s'estructura de la següent manera: Els widgets base de cada pestanya són anomenats "pàgines". Aquestes pàgines s'implementen en la classe QMainToolBoxPage i contenen únicament un *Combo Box*, un botó (Apply) i un QWidgetStack que conté els formularis amb els paràmetres de cada mètode. El QWidgetStack és, com el seu nom indica, una pila de widgets la qual només mostra un d'ells. Segons el mètode que escollim amb el *Combo Box*, el QWidgetStack mostrarà el formulari amb els paràmetres corresponents. Dins d'aquesta pila de formularis hi afegirem els nostres.

Malgrat que expliquem els detalls de l'estructura del QMainToolBox, no ens hem de preocupar per aquesta, ja que la nostra preocupació només és crear el formulari de paràmetres que anirà dins del QWidgetstack que conté un dels QMainToolBoxPage. QMainToolBox conté el mètode *addWidget()* pensat per donar-li el formulari de paràmetres, encarregant-se ell de col·locar-lo com cal.

Per fer aquests formularis disposem d'una plantilla que proporciona un conjunt de signals i slots que hauran de contenir tots els formularis que aniran dins del toolbox. L'objecte que es creï a partir d'aquesta plantilla heretarà de la classe QInputParameters.

L'àrea de treball

L'àrea de treball implementada a la classe QWorkingArea és el lloc indicat per a posar les noves finestres que obrim, tant de visualització, segmentació com de resultats numèrics. Aquesta classe s'encarrega del posicionament de les finestres entre altres coses.

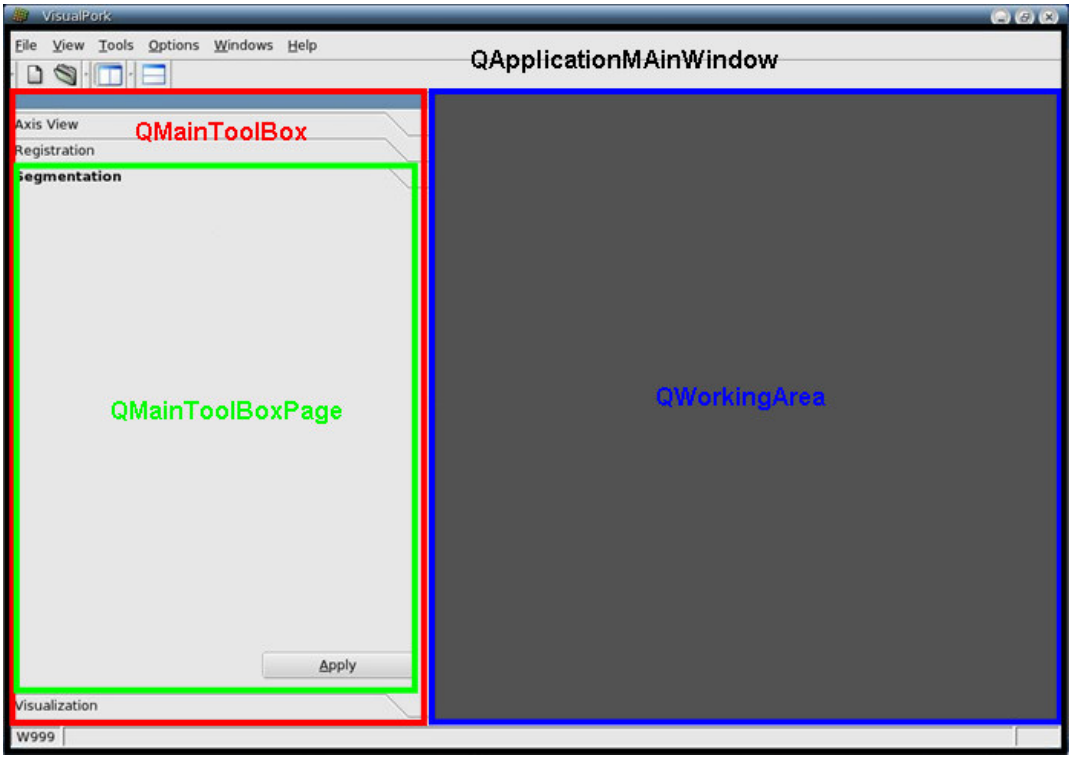


Figura 4.19: Interfície gràfica de la plataforma.

En la figura anterior observem la interfície gràfica de la plataforma. El requadre negre és la QApplicationMainWindow, la finestra mare, que contindrà tots els components gràfics de l'aplicació.

El requadre vermell és la QMainToolBox que, mitjançant pestanyes, adjuntarà formularis gràfics per a la introducció de dades. Cadascuna de les pestanyes de la QMainToolBox (requadre verd), equival a una pàgina de la mateixa, implementada per la classe QMainToolBoxPage.

Per últim, comentar que el requadre blau ens representa la QWorkingArea, que és la part de la finestra principal que incorporarà els QWorkingAreaWidgets.

Director, Parameters i InputParameters

Aquestes tres classes són les encarregades de l'execució de les tècniques de segmentació, visualització o registre. A grans trets la funció de cada una d'aquestes classes és la següent:

- Director: Enllaça la interfície amb la implementació .
- Parameters: Encapsula tots els paràmetres d'un determinat mètode centralitzant-los en una única instància
- QInputParameters: Element d'interfície (widget) base per a la introducció de dades a través de formularis, histogrames,etc.
- QAction: Enllaça la interfície amb el director per poder disparar l'execució del mètode.

Tant Director com QInputParameters contenen una referència (punter) a la respectiva classe de Parameters. La relació entre Parameters i QInputParameters és de 1:N. Hi pot haver un o més QInputParameters que ataquin a un mateix Parameters ja que la informació dels paràmetres es pot introduir des de més d'un element de la interfície.

La introducció de paràmetres és un aspecte que es pot realitzar d'una manera 'clàssica' amb caixes de text, desplegables i d'altres elements bàsics, o bé, amb elements més complexes com per exemple gràfiques, histogrames, etc depenent de les necessitats. És possible que la introducció de paràmetres d'un mètode es faci a través de més d'un formulari, els quals es poden fer servir alhora, per tant, caldrà tenir les dades en sincronisme entre tots els elements d'interfície. La classe base per a tots els elements d'interfície destinats a la introducció de paràmetres s'anomena QInputParameters.

4.6.- INTEGRACIÓ DE NOUS MÈTODES EN L'APLICACIÓ

Al partir d'una aplicació existent, cal conèixer les normes de programació i les regles a respectar per a poder afegir noves funcionalitats a l'aplicació. Per a cada funcionalitat que volem afegir a la plataforma, haurem de seguir els següents passos i crear les següents classes:

- **Implementació del mètode de segmentació o visualització** que volem integrar a la plataforma. Aquest mètode ha de quedar encapsulat dins una classe de nom Xxx.

- **Encapsulament dels paràmetres**: A continuació cal construir una classe que hereta de la classe abstracta Parameters i que anomenarem XxxParameters. En aquesta classe cal posar-hi tots els paràmetres (m_parametre1,m_parametre2,...), afegir una enumeració xxxParametersNames {Parametre1, Parametre2, ...} i crear dos mètodes per a cada paràmetre. Aquests han de ser els típics *set* i *get* per obtenir i modificar els seus valors. Tots els mètodes de tipus *set* emeten un signal *changed(int)* destinat a ser emès quan s'actualitza un paràmetre.

- **Interfície gràfica (introducció de paràmetres)**:

  - El primer a fer és crear un **formulari per a l'entrada de paràmetres**. Aquest formulari és el que es mostrarà en el toolbox. Per fer-ho, en principi s'utilitzarà el QtDesigner, l'eina gràfica de les Qt per a crear widgets gràfics. El formulari creat li posarem el nom de XxxBase, aquesta classe només conté els elements d'interfície. El QtDesigner crea un fitxer *.ui, aquest l'afegim a la zona d'interfície de l'aplicació.

    La classe XxxInputParameters hereta de QInputParameters, aquesta classe és una classe base per a tots els elements d'interfície destinats a introduir paràmetres i ens proporciona els signals i slots necessaris per a llegir-los.

  - A continuació **creem la classe filla** de la creada en el primer pas. Aquesta classe s'anomenarà XxxInputParametersForm i implementarà tota la interacció que pot haver-hi entre la interfície i l'usuari, és a dir que conté el codi relatiu a les accions associades a cada element.

    Cada vegada que canvia un paràmetre a la interfície s'actualitza a la classe de XxxParameters. La classe XxxInputParametersForm tindrà dos slots heretats de la classe Parameters: *readParameter(int)* que llegirà un paràmetre o un altre segons el signal que emeti la classe

XxxParameters, i *writeAllParameters()* que escriu tots els paràmetres al formulari.

- **La coordinació** de l'execució del mètode la durà la classe XxxDirector, classe que hereta de Director. Per poder portar el control, aquesta classe conté un punter a la QApplicationMainWindow, un punter a la classe de paràmetres XxxParameters i un últim punter al mètode en si, és a dir a la classe Xxx. Aquesta classe XxxDirector contindrà un o varis mètodes *setParameters* per indicar-li la instància dels paràmetres que ha d'utilitzar i reimplementarà el mètode *execute()* que hereta de Director, per executar l'algorisme i realitzar totes les interaccions necessàries.

- També cal **lligar la interfície amb el director**. Des d'algun element de la interfície cal activar la QAction associada al nostre mètode i notificar al director que iniciï l'execució. Aquest element (botó 'Apply' de la classe QMainToolBox) emet el signal *applyMethod()* del formulari i aquest signal està connectat a l'slot *execute()* de la classe XxxDirector.

- Finalment, per **afegir tota aquest estructura a la plataforma** cal:

  - Afegir a la classe QApplicationMainWindow la declaració dels nous atributs m_xxxAction, m_xxxDirector, m_xxxParameters i m_xxxInputParametersForm.

  - A continuació, cal afegir dins la implementació del constructor la construcció dels objectes del mètode que ja hem declarat, i connectar els signals i slots necessaris: entre QAction activada i XxxDirector *execute()* i entre XxxParameters *changed(int)* i XxxInputParametersForm *readParameter(int)*.

  - I per acabar afegir el formulari de paràmetres al QMainToolBox en el mètode createMainToolBox() de la mateixa classe QApplicationMainWindow, utilitzant el mètode addWidget(...).

4.7.- DISSENY DE CLASSES

En aquest apartat tractarem el disseny de les classes que hem hagut d'incorporar a la plataforma inicial, per a poder aconseguir els objectius d'aquest projecte. Ens centrarem en els dos aspectes principals. D'una banda veurem el disseny de classes dels diferents mòduls encarregats de la interfície gràfica. D'altra banda observarem l'estructura general que es crea per implementar i executar una tècnica de segmentació o visualització.

4.7.1.- DISSENY DE CLASSES DE LA INTERFÍCIE GRÀFICA

A continuació mostrarem els diagrames de classe més destacats que pertanyen a mòduls referents a la interfície gràfica. Aquests diagrames els dividirem en diferents seccions segons la funció que realitzen:

- Obertura d'un model porcí.
- Selecció de llavors.
- Selecció del color.
- Manipulació dels resultats de les tècniques de segmentació.
- Manipulació dels resultats de la visualització Ray Casting.
- Manipulació dels resultats de la visualització Multiplanar.

La interfície gràfica que permet introduir els paràmetres de les diferents tècniques de segmentació i visualització, l'explicarem en el següent apartat 4.7.2 d'implementació d'una tècnica.

4.7.1.1.- Interfície gràfica per obrir un model porcí

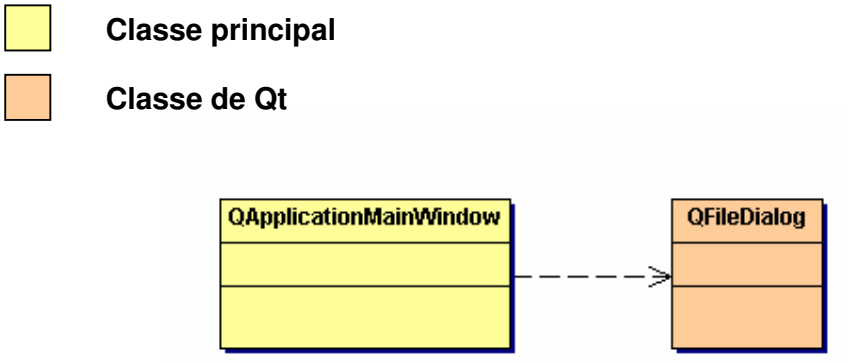


Figura 4.20: Disseny de classes de la interfície d'obertura del model porcí.

Aquesta estructura de classes és l'encarregada de la interfície que ens apareix per obrir un model porcí. La classe QApplicationMainWindow és la classe principal que s'ocuparà de realitzar les crides per dur a terme les accions que l'usuari selecciona. En el moment que l'usuari esculli l'opció d'obrir un model porcí, aquesta classe crearà una nova instància QFileDialog. La QFileDialog és una classe implementada en Qt i que actuarà d'interfície per navegar per els diferents directoris i seleccionar el model concret que volem obrir.

Aquest diagrama es pot veure més detallat en el diagrama de seqüència 4.8.1.

4.7.1.2.- Interfície gràfica per seleccionar les llavors

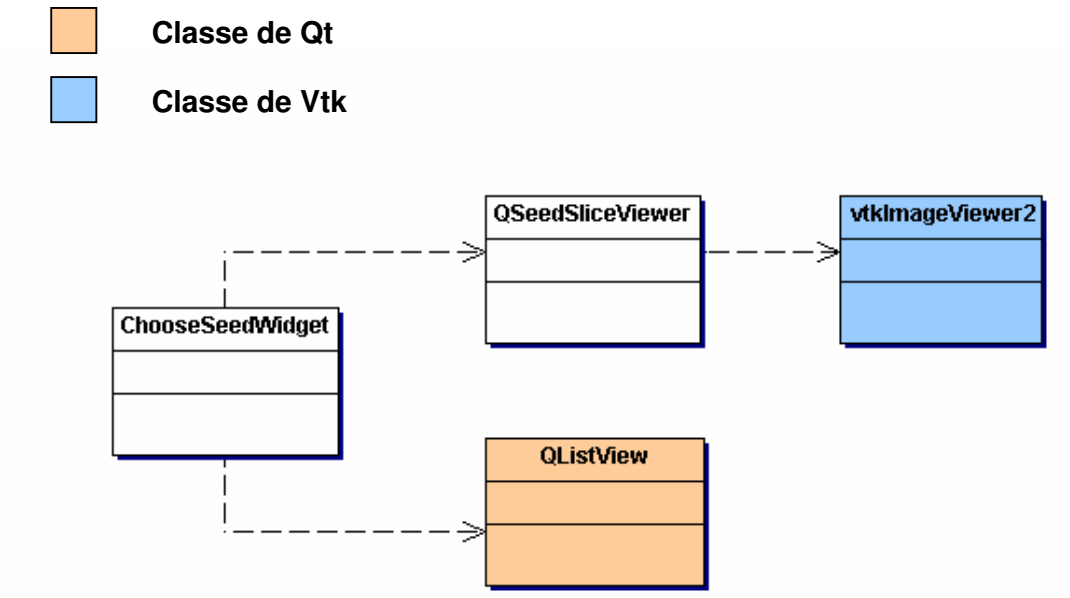


Figura 4.21: Disseny de classes de la interfície de selecció de llavors.

La classe principal d'aquest diagrama és la ChooseSeedWidget. Aquesta classe serà la interfície amb la que interactuarà l'usuari per seleccionar les llavors de les tècniques de segmentació. Aquesta interfície està composta per 3 vistes del model (Axial, Sagital i Coronal) més una taula on es guardaran les llavors que l'usuari vagi seleccionant.

Per crear aquestes vistes s'utilitza la classe QSeedSliceViewer. Aquesta classe disposa de 3 mètodes molt importants. El primer d'ells és el que s'anomena *signalSeed(int,double,double,double,double)*. Aquest mètode serà l'encarregat de captar els clics que l'usuari realitzi per seleccionar les llavors, i obtenir la

vista sobre la que ha pres, les coordenades i el valor d'intensitat d'aquest punt. Els altres 2 mètodes són el *drawSeed(double,double,double,double)* i *deleteSeed(int)*. El primer d'ells serà el que utilitzarem per dibuixar la llavor sobre les 3 vistes, mentre que el segons ens permetrà eliminar alguna llavor que haguéssim dibuixat prèviament.

Aquesta classe QSeedSliceViewer utilitza la classe vtkImageViewer2 que ens ofereixen les llibreries Vtk per poder representar les vistes Axial, Sagital i Coronal del model.

Finalment la última classe que intervé en aquest diagrama és la QListView. Aquesta és una classe de les llibreries Qt, que tal com hem dit permet guardar una llista amb totes les llavors seleccionades.

Aquest diagrama es pot veure més detallat en el diagrama de seqüència 4.8.2.

4.7.1.3.- Interfície gràfica per seleccionar el color

Classe de Qt

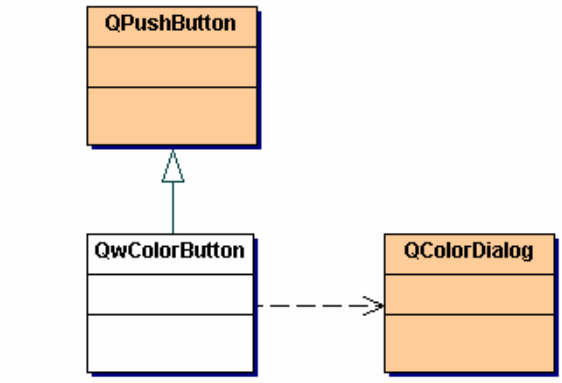


Figura 4.22: Disseny de classes de la interfície de selecció de colors.

Aquest diagrama de classes és el que s'encarrega de crear una interfície per poder seleccionar un color. Aquest selector de colors l'utilitzarem en el moment d'aplicar la tècnica de visualització de Ray Casting. La classe principal d'aquesta estructura és la QwColorButton. Aquesta classe hereta d'una classe de Qt anomenada QPushButton, i així permet que la interfície tingui forma de botó. Al prémer aquest botó és quan s'obrirà realment el selector de colors implementat mitjançant la classe de Qt anomenada QColorDialog.

4.7.1.4.- Interfície gràfica per manipular els resultats de la segmentació

Classe Itk

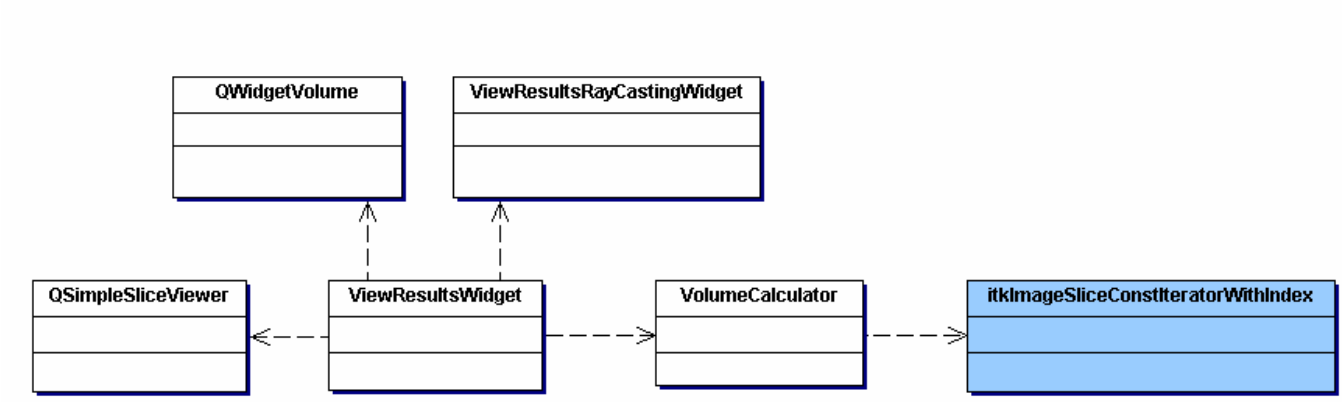


Figura 4.23: Disseny de classes encarregat de la manipulació de resultats.

Aquesta estructura de classes és la que ens permet visualitzar i manipular els resultats de les tècniques de segmentació. La classe principal entorn de la qual gira aquest disseny és la classe ViewResultsWidget. Aquesta és una classe d'interfície gràfica que s'encarregarà de realitzar les crides a la resta de classes i poder observar així la visualització dels resultats, el volum, percentatge i les àrees dels models resultants, així com la representació amb Ray Casting o Multiplanar del model segmentat, etc.

Visualització de la segmentació

Per tal de visualitzar amb 2D el resultat de la tècnica de segmentació, la classe ViewResultsWidget utilitzarà la QSimpleSliceViewer. Aquesta és una classe que té com entrada un volum i que ens ofereix un visor simple de les llesques (slices) que formen aquest volum. D'aquesta manera ens permet escollir la vista Axial, Coronal o Sagital del model.

Càlcul de mesures: volum i àrees

La classe encarregada de realitzar el càlcul del volum i de les àrees és la classe VolumeCalculator. Aquesta rebrà com entrada el model segmentat obtingut mitjançant la tècnica de segmentació que haguem aplicat. Per obtenir els resultats disposarà dels dos mètodes següents: *getTotalVolume()* que ens proporcionarà el volum total del model, i *getSliceArea(int slice)* que ens oferirà l'àrea de la llesca que indiquem com a paràmetre.

La tècnica que hem utilitzat per a calcular el volum és força senzilla. Es tracte de recórrer el volum segmentat de principi a fi. Examinarem tots els píxels d'una llesca fins arribar al final. A continuació passarem a la següent llesca fins a finalitzar tot el model. A mesura que fem el recorregut contarem tant els píxels totals segmentats com els de cada llesca que tenen un valor de 1. D'aquesta manera obtindrem el volum total de l'objecte segmentat i l'àrea de cada llesca. Per poder realitzar aquest recorregut sobre el model hem utilitzat la classe de Itk: itkImageSliceConstIteratorWithIndex.

Visualització 3D del model

Finalment, la última acció que podem realitzar sobre els resultats de les tècniques de segmentació és la visualització 3D del model. Per efectuar aquesta visualització disposem de dues tècniques: Ray Casting i Multiplanar. La primera d'elles la podem obtenir mitjançant la classe d'interfície gràfica ViewResultsRayCastingWidget. Respecte la tècnica Multiplanar disposem també d'una classe anomenada QWidgetVolume que l'implementa. En els diagrames 4.7.1.5 i 4.7.1.6 es pot veure amb més detall el funcionament de cada una d'aquestes visualitzacions.

Aquest diagrama es pot veure més detallat en el diagrama de seqüència 4.8.6.

4.7.1.5.- Interfície gràfica per manipular els resultats de la visualització Ray Casting

Classe Vtk

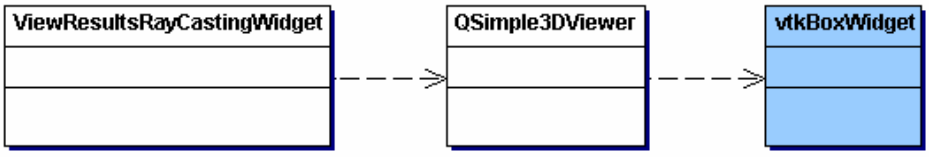


Figura 4.24: Disseny de classes encarregat de la manipulació dels resultats de la visualització Ray Casting.

Aquesta estructura de classes és la que ens permet visualitzar i manipular els resultats de la tècnica de visualització de Ray Casting. La classe principal d'aquest diagrama és la ViewResultsRayCastingWidget. Aquesta serà la que actuarà d'interfície per mostrar el resultat. Aquesta classe utilitza la QSimple3DViewer que s'encarregarà d'efectuar la representació del volum amb Vtk mitjançant l'algoritme de Ray Casting. Tal com hem vist en el capítol de definició dels requeriments, aquesta visualització també ha de disposar d'una eina que permeti explorar el model representat amb més detall. Aquesta eina és la que s'anomena *Box Widget* i està implementada amb les llibreries Vtk mitjançant la classe vtkBoxWidget.

4.7.1.6.- Interfície gràfica per manipular els resultats de la visualització Multiplanar

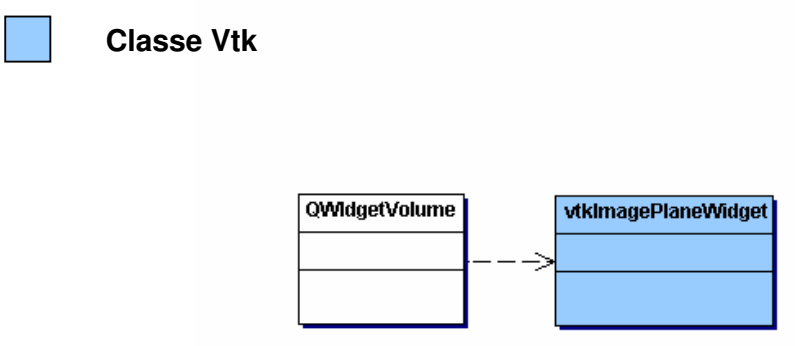


Figura 4.25: Disseny de classes encarregat de la manipulació dels resultats de la visualització Multiplanar.

Aquesta estructura de classes és la que ens permet visualitzar i manipular els resultats de la tècnica de visualització Multiplanar. La classe principal d'aquest diagrama és la QWidgetVolume que ja ens venia implementada amb la plataforma original. Aquesta serà la que actuarà d'interfície per mostrar el resultat. Aquesta classe utilitza la classe vtkImagePlaneWidget per mostrar les diferents vistes del model (vista Axial, Sagital i Coronal). El mètode principal de la classe QWidgetVolume és el *updateCoords()*. Aquest mètode estarà atent a les diferents accions que efectuï l'usuari sobre els plans, per així actualitzar el valor escalar del punt que l'usuari té seleccionat, la llesca seleccionada d'una vista concreta, etc.

4.7.2.- IMPLEMENTACIÓ D'UNA TÈCNICA

En aquest apartat mostrarem els diagrames de classe encarregats de la implementació de les tècniques. Primer veurem els mòduls utilitzats en el cas de la **visualització Ray Casting**. Referent a la tècnica Multiplanar, comentar que venia implementada amb la plataforma inicial i el seu diagrama de classes ja l'hem observat en l'apartat 4.7.6.1. A continuació ens centrarem amb els mòduls utilitzats en el cas de les **tècniques de segmentació.** Tal com hem vist en l'apartat 4.6, per implementar una nova tècnica a la plataforma cal crear una sèrie de classes molt concretes, d'aquí que expliquem el funcionament de les tècniques de segmentació d'una manera conjunta.

4.7.2.1.- Implementació de la tècnica Ray Casting

Classe principal

Classes d'interfície gràfica

Classes pròpies de la implementació de la tècnica

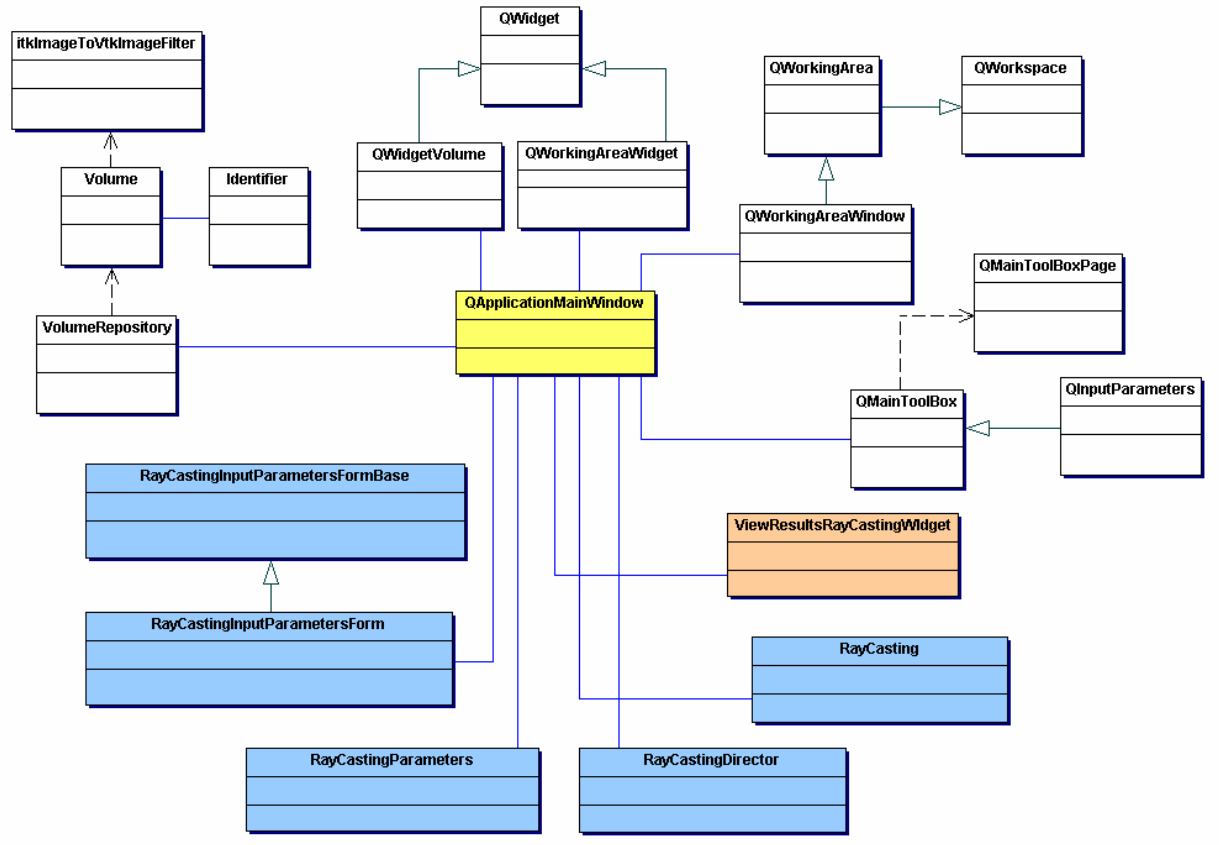


Figura 4.26: Disseny de classes de la tècnica Ray Casting.

Introducció de paràmetres

Per a poder introduir els paràmetres de la tècnica disposem de la classe RayCastingInputParametersForm que hereta de la classe base RayCastingInputParametersFormBase. Aquesta és una classe interfície que ens proporciona el formulari d'introducció de dades.

Emmagatzemant de paràmetres

És necessari disposar d'una estructura on emmagatzemar els paràmetres seleccionats. La classe encarregada en el cas de la tècnica Ray Casting és la RayCastingParameters. Aquesta heretarà de la classe Parameters i encapsularà els paràmetres de la tècnica en una sola instància.

Implementació de la tècnica

La classe RayCasting és l'encarregada d'implementar la tècnica de visualització. Aquesta classe rebrà com entrada els paràmetres seleccionats, i generarà com a sortida el volum amb 3D a través de la interfície de visualització de resultats. Aquesta doncs, serà la que aplicarà els diferents filtres de la tècnica i obtindrà el resultat final.

Enllaç interfície i implementació (director)

Com hem comentat anteriorment, la classe Director s'ocupa d'enllaçar la interfície amb la implementació. Serà necessari disposar d'un director per cadascuna de les tècniques de segmentació o visualització que implementem. En aquesta ocasió la classe RayCastingDirector s'ocuparà d'enllaçar la introducció de paràmetres des de RayCastingInputParametersForm amb l'execució de la tècnica i la posterior visualització del resultat.

Visualització de resultats

Tal com hem vist en el punt 4.7.1.5 la classe encarregada de mostrar els resultats de la visualització Ray Casting serà la ViewResultsRayCastingWidget. Aquesta és una classe d'interfície gràfica que s'encarregarà de realitzar les crides a la resta de classes i poder observar així la visualització dels resultats.

4.7.2.2.- Implementació de les tècniques de segmentació

En aquest apartat comentarem les classes utilitzades per la implementació de les tècniques de segmentació. Primer de tot veurem els mòduls creats per cada tècnica, i tot seguit comentarem la funció de cada classe.

Connected Threshold

Classe principal
Classes d'interfície gràfica
Classes pròpies de la implementació de la tècnica

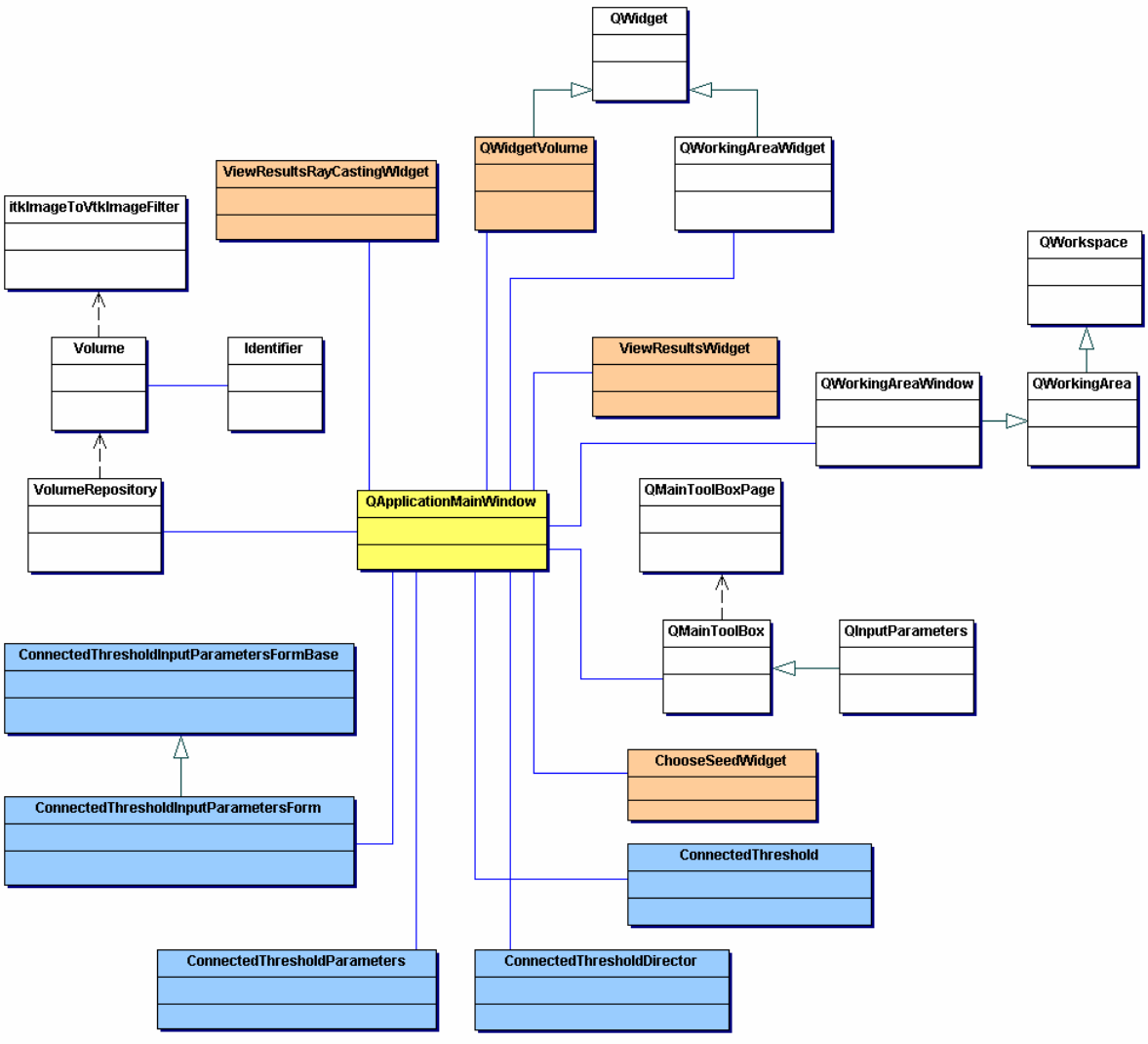


Figura 4.27: Disseny de classes de la tècnica Connected Threshold.

Neighborhood Connected

Classe principal
Classes d'interfície gràfica
Classes pròpies de la implementació de la tècnica

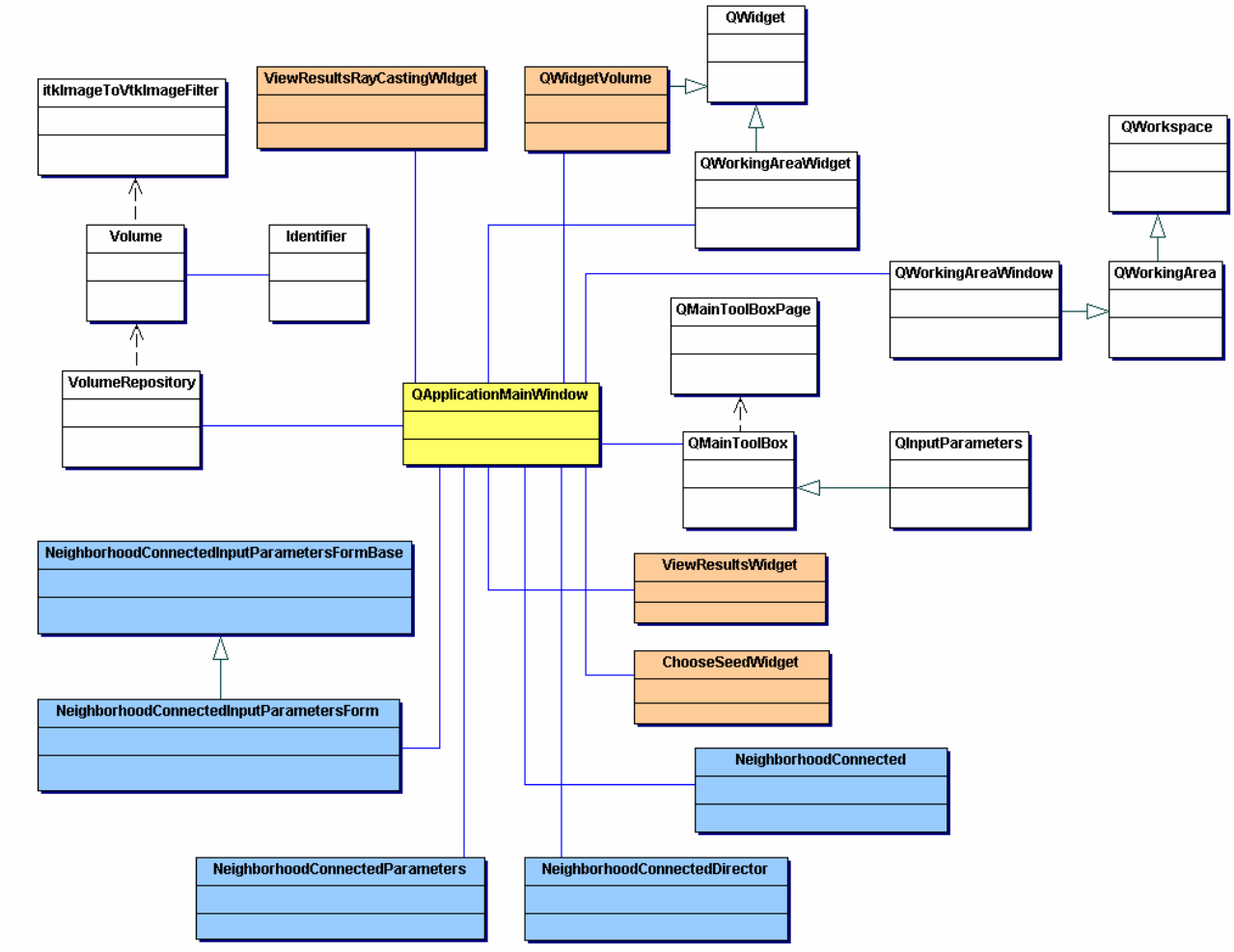


Figura 4.28: Disseny de classes de la tècnica Neighborhood Connected.

Confidence Connected

Classe principal
Classes d'interfície gràfica
Classes pròpies de la implementació de la tècnica

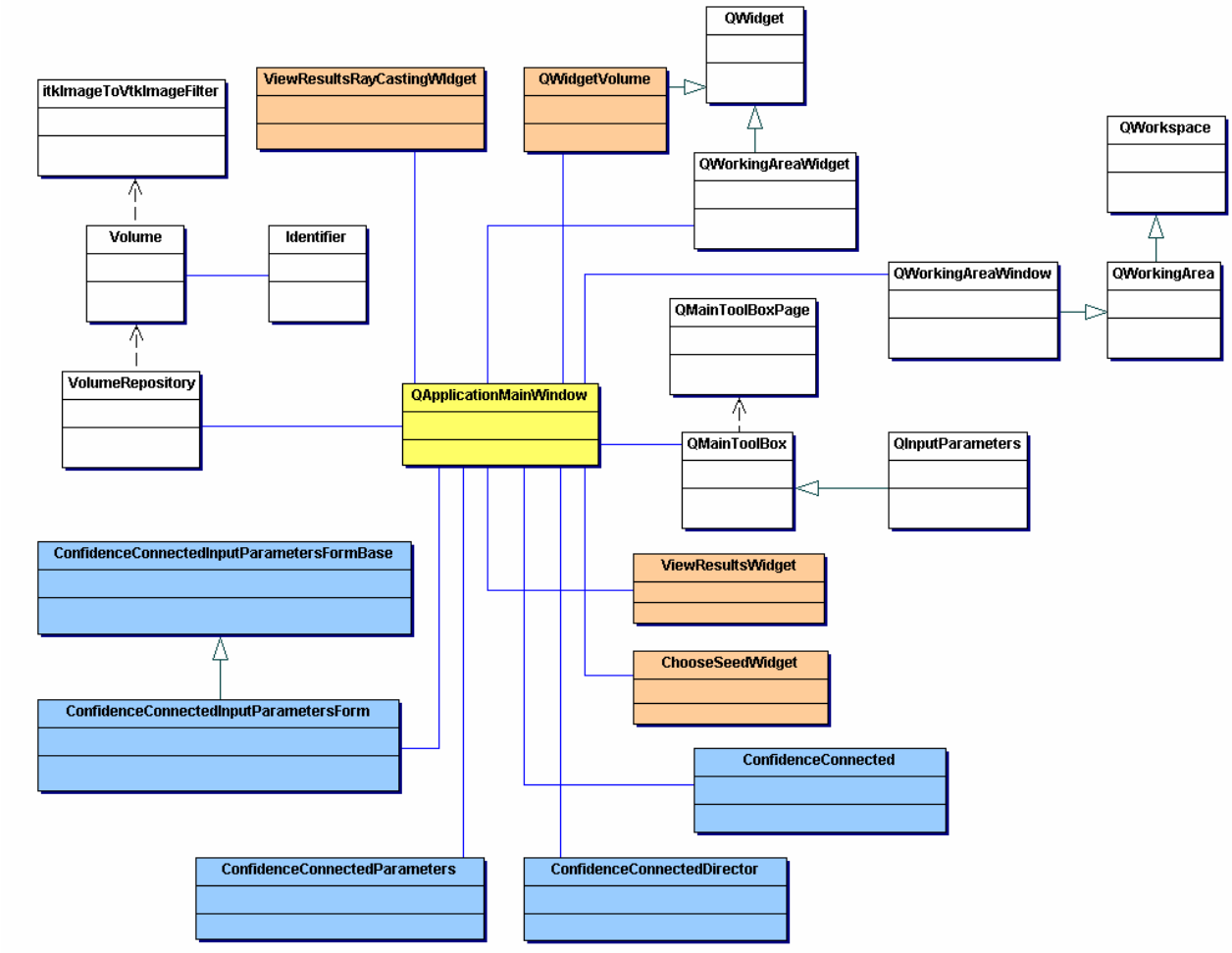


Figura 4.29: Disseny de classes de la tècnica Confidence Connected.

Isolated Connected

- **Classe principal**
- **Classes d'interfície gràfica**
- **Classes pròpies de la implementació de la tècnica**

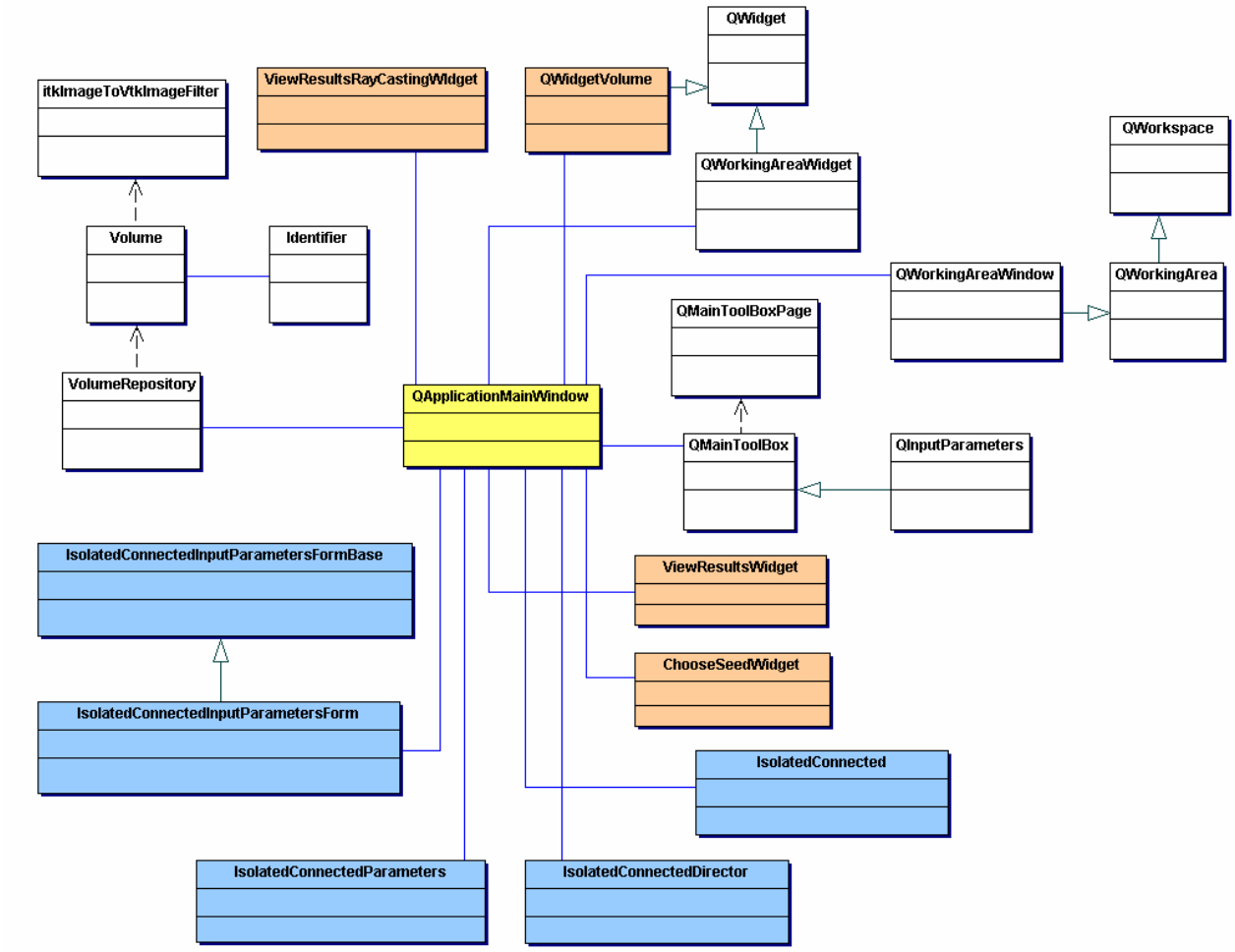


Figura 4.30: Disseny de classes de la tècnica Isolated Connected.

Introducció de paràmetres

Per a poder introduir els paràmetres d'aquestes tècniques disposem de les classes InputParametersForm (Ex:ConnectedThresholdInputParametersForm) que hereten totes de la corresponent classe base InputParametersFormBase. Aquestes seran les classes d'interfície que ens proporcionaran els formularis d'introducció de dades.

Introducció de la llavor/s

Un altre dels aspectes destacats al introduir els paràmetres és seleccionar la llavor o llavors. La majoria de tècniques de segmentació necessiten d'aquest paràmetre per a poder realitzar l'execució. Per efectuar aquesta selecció disposem de la classe interfície ChooseSeedWidget comentada anteriorment en l'apartat 4.7.1.2.

Emmagatzemant de paràmetres

És necessari disposar d'una estructura on emmagatzemar els paràmetres seleccionats. Cada tècnica ha de disposar de la seva classe de paràmetres (Ex: en el cas del Connected Threshold és la ConnectedThersholdParameters). Aquestes classes heretaran de la classe Parameters i encapsularan els paràmetres de les tècniques en una sola instància.

Implementació de la tècnica

Les classes amb el nom propi de cada tècnica són les encarregades d'efectuar la implementació (Ex: ConnectedThreshold). Aquestes classes rebran com entrada els paràmetres seleccionats, i generaran com a sortida el volum segmentat. Aquestes doncs, seran les que aplicaran els diferents filtres de la tècnica i obtindrà el resultat final.

Enllaç interfície i implementació (director)

Com hem comentat anteriorment, la classe Director s'ocupa d'enllaçar la interfície amb la implementació. Serà necessari disposar d'un director per cadascuna de les tècniques de segmentació que implementem.

Visualització de resultats

Tal com hem vist en el punt 4.7.1.4 la classe encarregada de mostrar els resultats de les tècniques de segmentació serà la ViewResultsWidget. Aquesta és una classe d'interfície gràfica que s'encarregarà de realitzar les crides a la resta de classes i poder observar així la visualització dels resultats, el volum, percentatge i les àrees dels models resultants, així com la representació amb Ray Casting o Multiplanar del model segmentat. Per poder fer aquestes representacions utilitzarà les classes ViewResultsRayCastingWidget i QWidgetVolume respectivament (veure punts 4.7.1.5 i 4.7.1.6).

El funcionament d'aquests mòduls es pot veure amb més detallat en els diagrames de seqüència 4.8.3, 4.8.4 i 4.8.5.

4.8- DIAGRAMES D'ACTIVITAT I DE SEQÜÈNCIA

Els diagrames d'activitat són una variació de les màquines d'estats. En aquests, els estats són les accions que es porten a terme i les transicions són activades al finalitzar l'acció. Són de fet, una màquina d'estat dels procediments o casos d'ús.

A continuació podem observar el diagrama d'activitat general per a l'execució d'una tècnica de segmentació o visualització amb la plataforma.

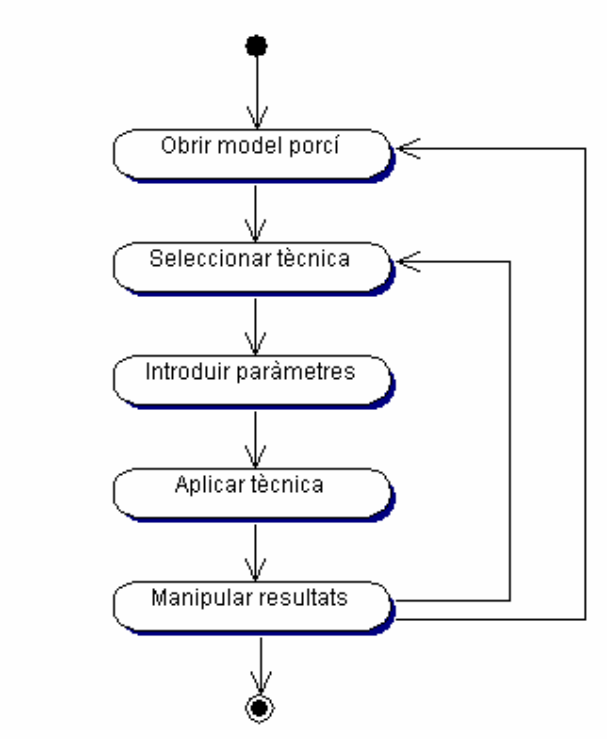


Figura 4.31: Diagrama d'activitat de l'execució d'una tècnica.

Tot seguit explicarem amb més detall mitjançant els diagrames de seqüència, les relacions entre classes que es produeixen al executar les diferents passes del diagrama d'activitat anterior. En els diagrames de seqüència que mostrarem només intervindran com actors els dels casos d'ús, que en el nostre cas únicament serà l'usuari.

Cada diagrama de seqüència mostrarà els events (un o més) que envia un actor al sistema per un cas d'ús concret.

4.8.1.- OBERTURA DEL MODEL PORCÍ

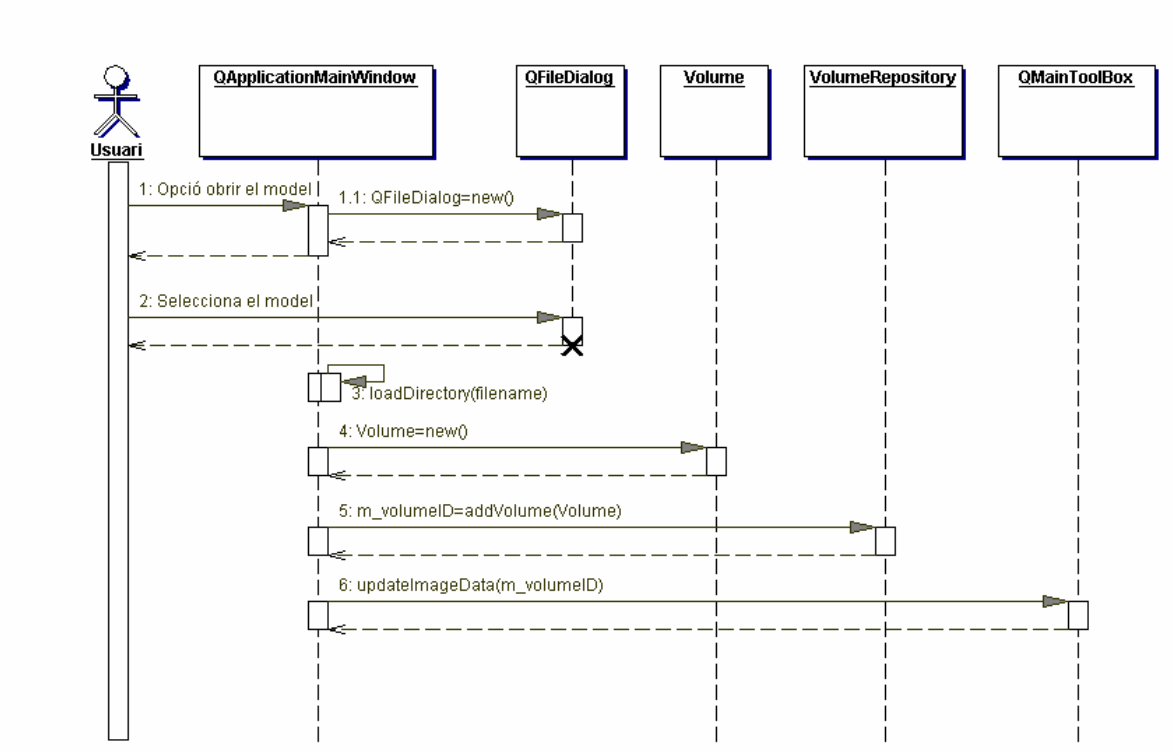


Figura 4.32: Diagrama de seqüència d'obertura del model porcí.

En el diagrama de seqüència anterior observem el procés d'obrir un model porcí. Aquest procés comença quan l'usuari indica que desitja obrir un model (pas 1). Al rebre aquesta petició, la classe *QApplicationMainWindow* crea un nou objecte d'interfície *QFileDialog* que s'encarregarà d'interactuar amb l'usuari per a determinar el model concret a obrir.

Una vegada l'usuari seleccioni el model desitjat (pas 2), la classe *QApplicationMainWindow* obtindrà el nom identificatiu del model (anomenat *filename*) i realitzarà l'acció de carregar-lo (pas 3). A continuació crearà un nou objecte *Volume* a partir de la informació carregada en el pas anterior. Tot seguit afegirà aquest nou *Volume* al repositori *VolumeRepository* (espai on guardem tots els models oberts). Al moment d'afegir el volum al repositori aquest ens retornarà l'identificador amb el que poder fer referència. Finalment la última acció que manca és actualitzar el *QMainToolBox* amb el nou model que acabem d'obrir. Per fer-ho, aplicarem el mètode *updateImageData* passant com a paràmetre l'identificador del volum obtingut amb anterioritat. D'aquesta manera haurem carregat el model i el podrem visualitzar a través de la *QMainToolBox*.

4.8.2.- SELECTOR DE LLAVORS

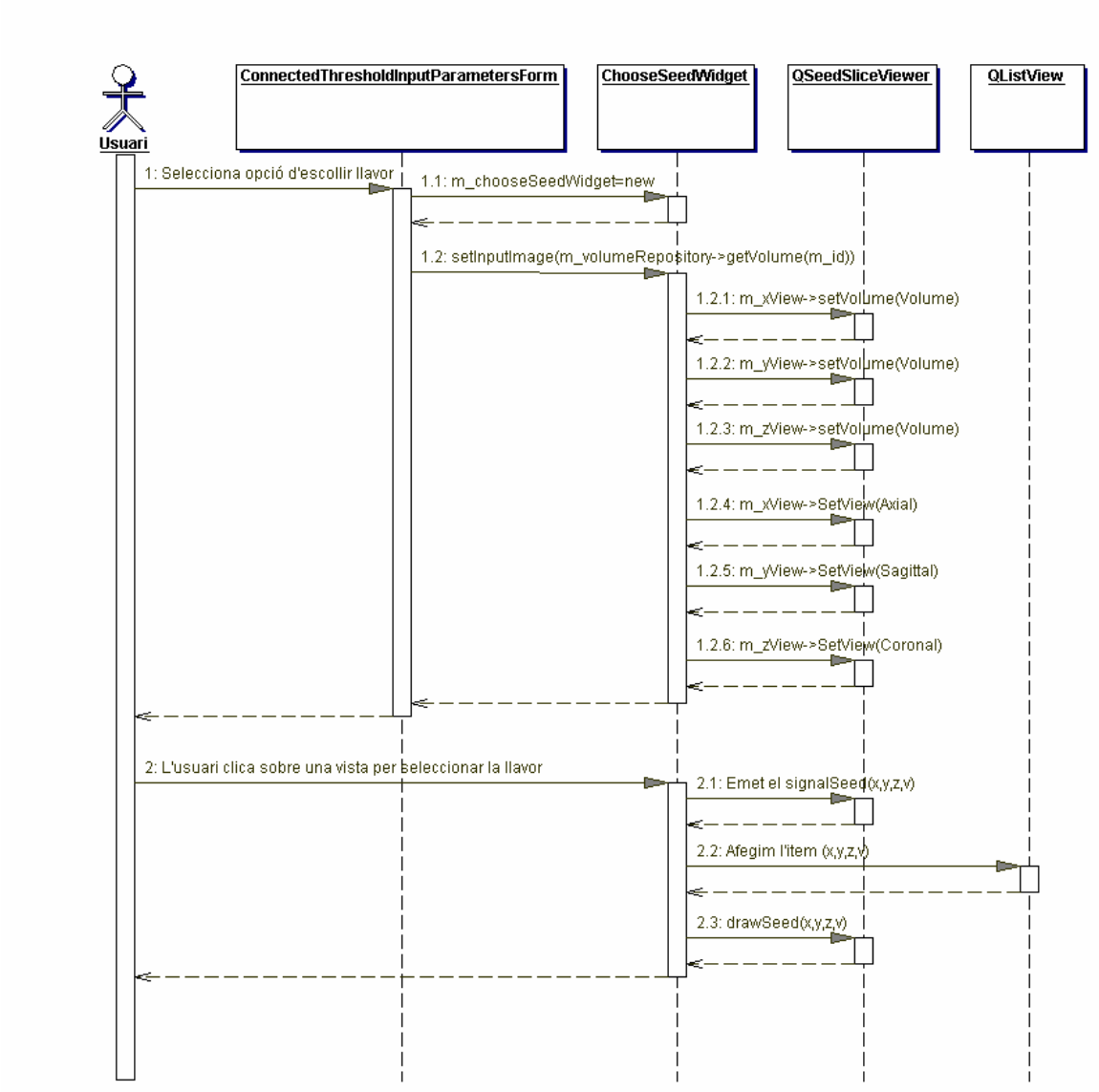


Figura 4.33: Diagrama de seqüència del selector de llavors.

En el diagrama de seqüència anterior observem el procés de selecció de la llavor/s de la tècnica de segmentació ConnectedThreshold. La resta de tècniques de segmentació tenen un funcionament idèntic, amb la única variació que el formulari d'introducció de paràmetres no es realitzarà amb la classe ConnectedThresholdInputParametersForm sinó amb el de la tècnica corresponent.

El procés s'inicia quan l'usuari selecciona a través de la interfície d'introducció de paràmetres l'opció d'escollir la llavor. En aquest moment la *ConnectedThresholdInputParametersForm* crea una nova instància de la classe *ChooseSeedWidget.* Aquesta segona classe és una interfície gràfica que interactuarà amb l'usuari per tal que aquest seleccioni la llavor o llavors que pretén utilitzar.

Aquesta interfície *ChooseSeedWidget* disposa de 3 vistes del model que permeten efectuar aquesta selecció. Aquestes vistes s'implementen mitjançant la classe *QSeedSliceViewer* que actuarà com un visor simple de llesques. Així doncs, el primer pas a realitzar és carregar el model a les vistes. Per aquest motiu apliquem el pas 1.2 *setInputImage* tot indicant com a paràmetre el volum o model que tenim obert. Al rebre aquesta crida, la classe *ChooseSeedWidget* carregarà a les *QSeedSliceViewer* el volum mitjançant el mètode *setVolume(Volume)* corresponent als punts 1.2.1, 1.2.2 i 1.2.3 del diagrama anterior.

Finalment cal indicar la visualització que es desitja efectuar a cada vista. Per aquest motiu es disposa dels punts 1.2.4, 1.2.5 i 1.2.6 on s'especifiquen les vistes Axial, Sagital i Coronal.

Una vegada efectuades aquestes passes l'usuari ja pot interactuar amb la interfície de selecció de llavors. D'aquesta manera, quan l'usuari premi sobre un punt d'una de les vistes es procedirà a calcular les propietats de la llavor que ha seleccionat. La primera acció que realitza la classe *QSeedSliceViewer* és emetre un signal (senyal) tot indicant les coordenades x,y,z i el valor escalar del punt que ha seleccionat l'usuari. Aquest procés correspon al pas 2.1. Al rebre aquest senyal la classe *ChooseSeedWidget* afegirà un nou ítem a la taula de llavors seleccionades. Aquesta taula està implementada mitjançant la classe *QListView*.

Per últim, també representarem a les diferents vistes de la interfície la llavor que l'usuari acaba de seleccionar. Per fer aquesta representació utilitzarem la crida *drawSeed(x,y,z,v)* sobre la classe *QSeedSliceViewer.* Com a resultat d'aquesta representació observarem un punt vermell en la localització especificada.

4.8.3.- COORDINACIÓ DE PARÀMETRES

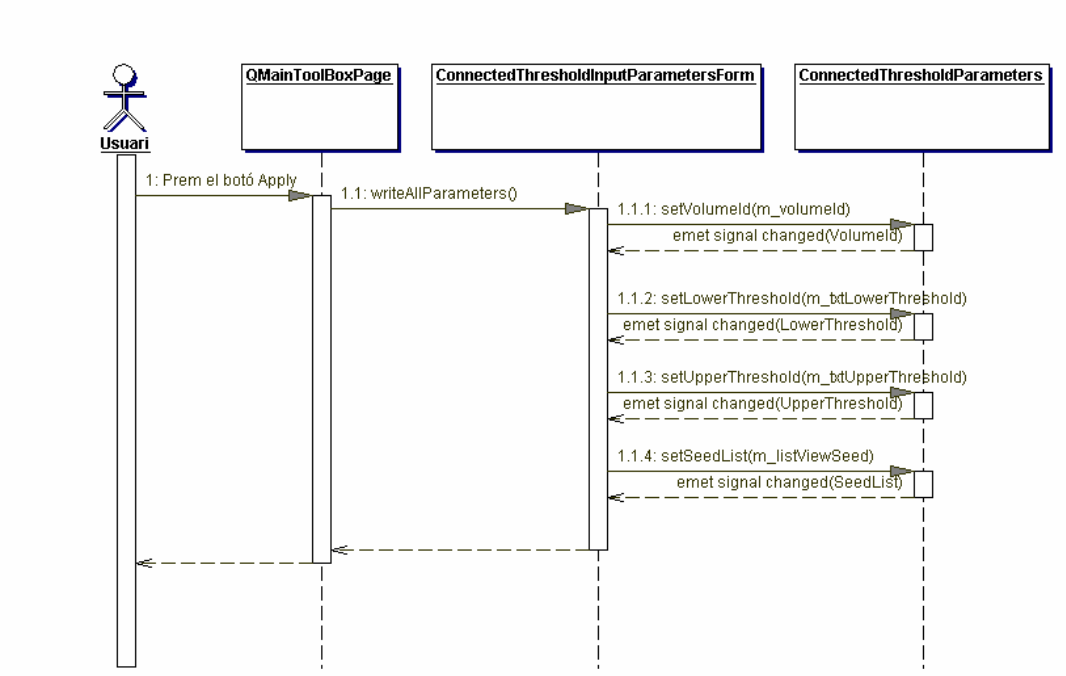


Figura 4.34: Diagrama de seqüència coordinació de paràmetres.

En el diagrama de seqüència anterior observem com es realitza la coordinació de paràmetres de la tècnica de segmentació ConnectedThreshold. La resta de tècniques de segmentació i visualització tenen un comportament molt similar.

El procés s'inicia quan l'usuari prem el botó *Apply* de la interfície *QMainToolBoxPage*. A continuació aquesta classe *QMainToolBoxPage* invoca el mètode *writeAllParameters()* de *ConnectedThresholdInputParameters* (Pas 1.1). Aquest mètode s'encarregarà d'emmagatzemar en la instància de paràmetres els valors que actualment hi ha en el formulari d'entrada de dades, és a dir, els valors que l'usuari ha seleccionat. Per realitzar aquest emmagatzematge s'utilitzen els mètodes *set* de la classe de paràmetres, passant com a paràmetre del mètode el valor actual de les caixes de text o de la llista de llavors del formulari. Així doncs, per guardar per exemple el paràmetre *LowerThreshold* efectuarem la següent crida: *setLowerThreshold(m_txtLowerThreshold)*.

4.8.4.- IMPLEMENTACIÓ D'UNA TÈCNICA: Coordinació

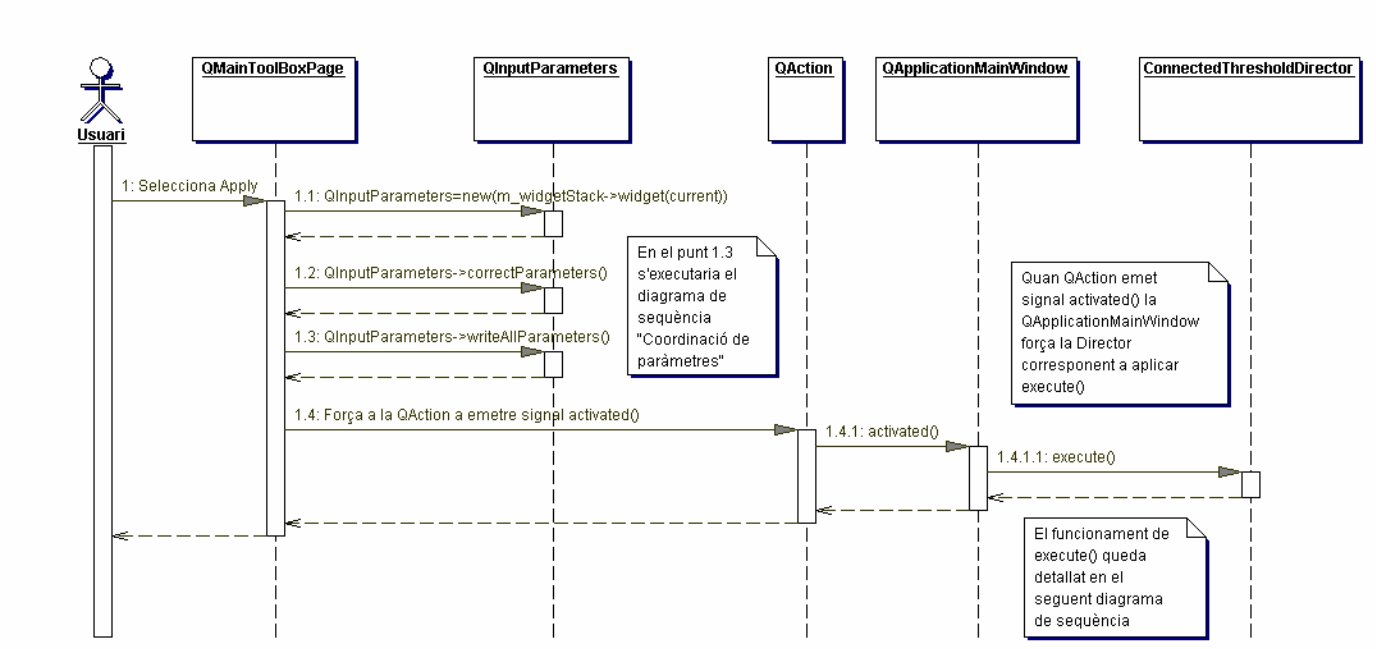


Figura 4.35: Diagrama de seqüència del procés de coordinació per realitzar l'execució d'una tècnica.

En el diagrama de seqüència anterior observem les diferents crides que es realitzen durant l'execució d'una tècnica de segmentació o de visualització. En la figura es mostra el cas concret de la tècnica *ConnectedThreshold*. La resta de tècniques tenen un comportament idèntic.

El procés s'inicia quan l'usuari prem el botó *Apply* de la classe *QMainToolBoxPage.* A continuació es crea un nou objecte *QInputParameters* amb els paràmetres que hi ha en el formulari actual. Per a recuperar el formulari actual que hi ha seleccionat en el *ToolBox,* s'utilitza la funció *m_widgetStack->widget(current)* corresponent al pas 1.1.

Una vegada disposem dels paràmetres, el primer que comprovem és que siguin correctes (pas 1.2). Per aquest motiu utilitzem la funció *correctParameters()* que comprova que els diferents paràmetres estiguin plens i que els valors especificats estiguin dins el rang de valors acceptables.

A continuació realitzem l'acció de coordinació de paràmetres (pas 1.3). En aquest punt apliquem el mètode *writeAllParameters()* per tal d'emmagatzemar els valors en una instància de paràmetres. El funcionament més detallat d'aquesta crida es pot observar en el diagrama de seqüència 4.8.3 "Coordinació de paràmetres".

Quan haguem finalitzat de realitzar aquestes accions, l'objecte *QAction* emetrà un senyal *activated()* indicant que l'execució definitiva de la tècnica es pot dur a terme. Al rebre aquest signal, la *QApplicationMainWindow* forçarà a la classe *ConnectedThresholdDirector* a executar l'slot *execute()*. És a dins d'aquest procés on s'aplicarà realment la tècnica de segmentació i s'obtindran els resultats definitius. En el diagrama de seqüència de continuació explicarem amb més detall les diferents passes que intervenen per aplicar el mètode *execute()*.

4.8.5.- IMPLEMENTACIÓ D'UNA TÈCNICA: Execució

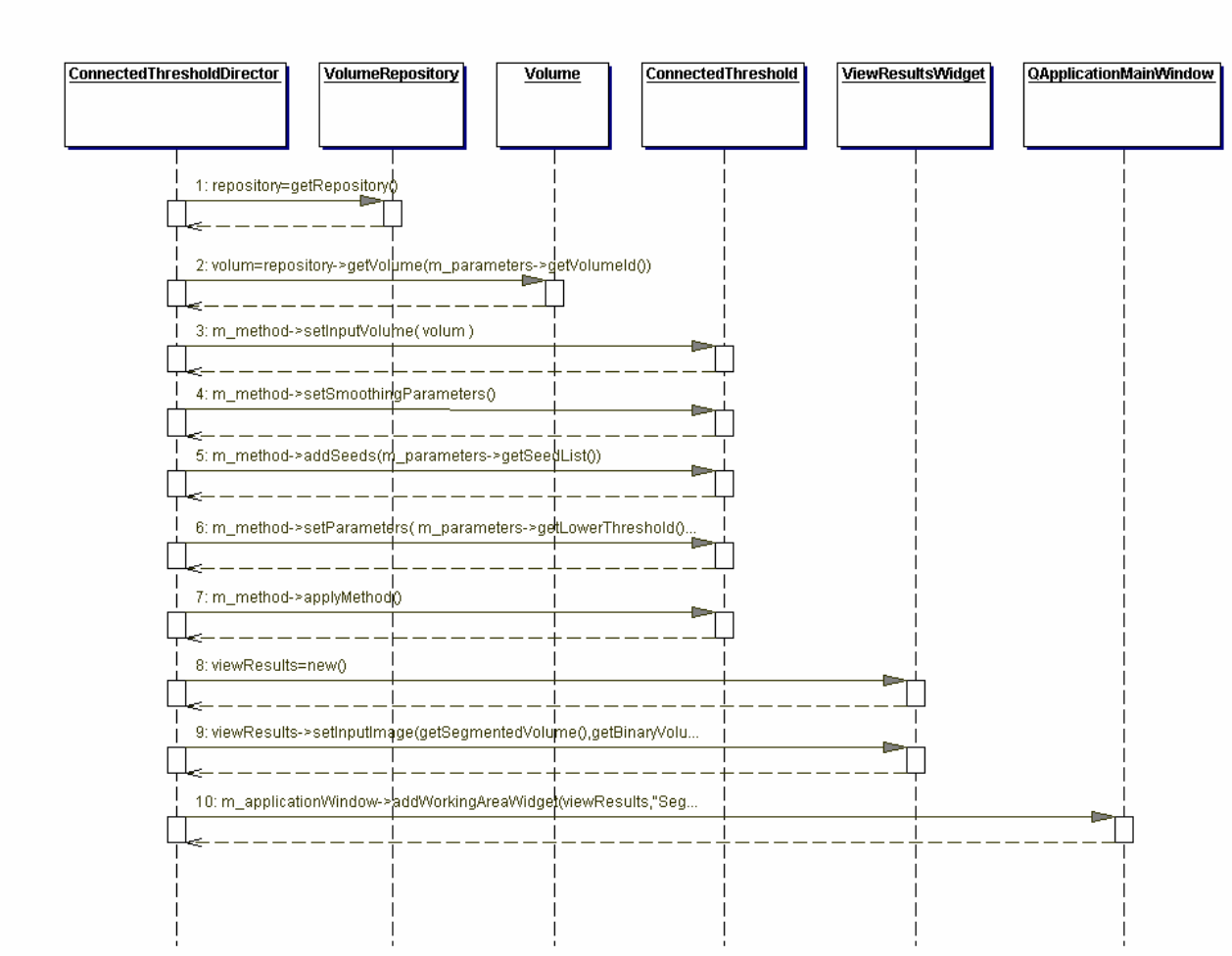


Figura 4.36: Diagrama de seqüència del funcionament intern de l'execució del mètode.

En el diagrama de seqüència anterior es pot observar el funcionament intern del mètode *execute()* de la classe *ConnectedThresholdDirector*. La resta de tècniques de segmentació tenen un comportament molt similar.

El primer pas que realitzem és recuperar el repositori i obtenir el volum amb el que estem treballant. Aquestes accions s'efectuen en les passes 1 i 2. A continuació procedim a indicar els paràmetres a la classe que implementa la tècnica, és a dir, a la classe *ConnectedThreshold*. D'aquesta manera comencem especificant el volum d'entrada mitjançant el mètode s*etInputVolume(volum)*. Tot seguit apliquem el filtre de *smoothing* mitjançant la crida *setSmoothingParameters()* i indiquem amb el mètode *addSeeds(m_parameters->getSeedList())* la llavor o llavors que utilitzarem. Per últim, especifiquem els paràmetres propis de la tècnica mitjançant la crida *setParameters(...)*. Una vegada especificats aquests valors ja podem aplicar els diferents filtres de segmentació. Aquest procés es realitza amb el mètode *applyMethod()* del pas 7 del diagrama anterior.

Finalment hem de mostrar el resultat de la segmentació per pantalla. Per fer-ho creem una nova instància de la classe *ViewResults* (pas 8) i li posem com a input el resultat de la tècnica de segmentació. Per últim cal situar aquest widget a l'àrea de treball. Això ho durem a terme amb el mètode *addWorkingAreaWidget* de la classe *QApplicationMainWindow*.

4.8.6.- MANIPULACIÓ DELS RESULTATS DE LA SEGMENTACIÓ

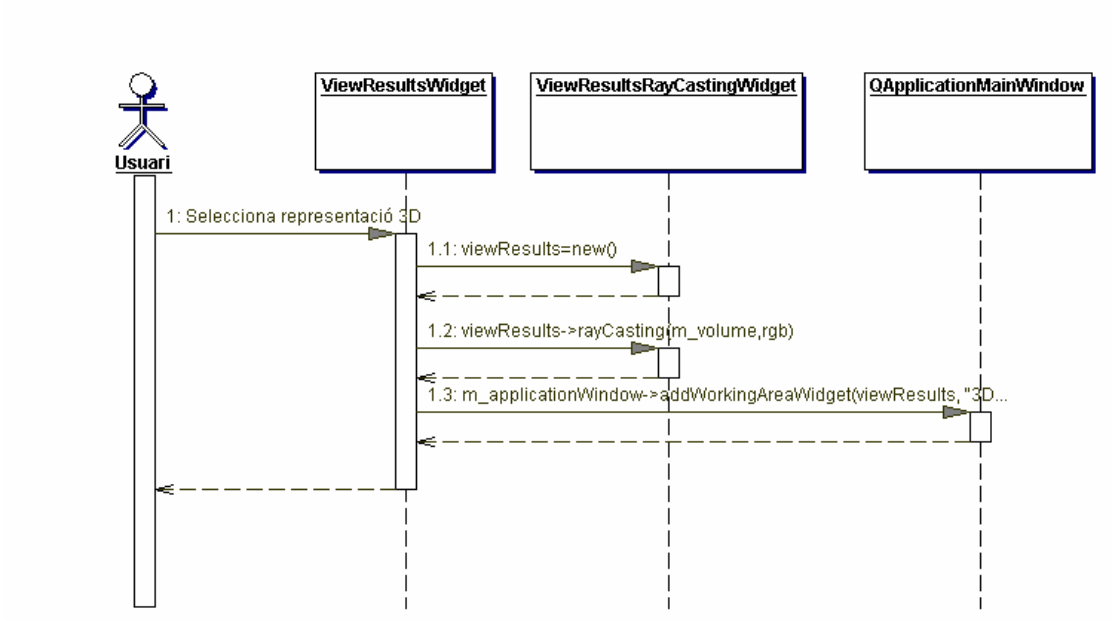


Figura 4.37: Diagrama de seqüència de la manipulació de resultats.

Aquest diagrama de seqüència és el referent a la manipulació dels resultats de segmentació. En aquest diagrama veurem les passes que es realitzen per obtenir mitjançant la tècnica de Ray Casting una representació de 3 dimensions del model segmentat. L'altre visualització disponible anomenada Multiplanar té un funcionament molt similar al que explicarem a continuació.

El procés s'inicia quan l'usuari prem el botó de representació 3D. A partir d'aquest moment la classe *ViewResultsWidget* crea una nova instància de la classe *ViewResultsRayCastingWidget* que serà la interfície a través de la qual observarem la representació. A continuació li indica a aquesta classe que apliqui el mètode de Ray Casting tenint com a paràmetres el volum i el color (rgb) que ha seleccionat l'usuari. Aquesta crida es realitzarà de la següent manera: *rayCasting(m_volume,rgb)*.

Per últim, cal situar aquest widget a l'àrea de treball. Això ho durem a terme amb el mètode *addWorkingAreaWidget* de la classe *QApplicationMainWindow*.

Interfície gràfica

En aquest capítol presentarem la **interfície gràfica** amb la que interactua l'usuari.

Per dissenyar aquesta interfície hem intentat aplicar els principis bàsics de disseny, és a dir, que sigui una interfície consistent, simple, flexible, robusta i que sigui compatible amb els usuaris que la utilitzaran i amb la tasca que ha de desenvolupar.

Un altre aspecte important de la interfície gràfica era disminuir el màxim possible els precipicis d'execució i d'avaluació. És a dir, que resulti senzill per l'usuari traduir els seus objectius a accions sobre la interfície (precipici execució), i que resulti senzill per l'usuari percebre l'estat del sistema i associar-ho als seus objectius psicològics (precipici d'avaluació).

D'acord amb aquests principis i amb aquestes premisses mostrem a continuació les diferents interfícies associades a cada acció.

5.1.- OBERTURA D'UN MODEL

Aquesta interfície serà la que ens permetrà obrir un model porcí i visualitzar-lo a través de la plataforma. La primera acció que durem a terme serà indicar que desitgem obrir un model. Per aquest motiu disposem del menú "*File/Open Directory"* tal com s'observa a la següent figura. També disposem d'un accés directe a la barra de menús de la part superior.

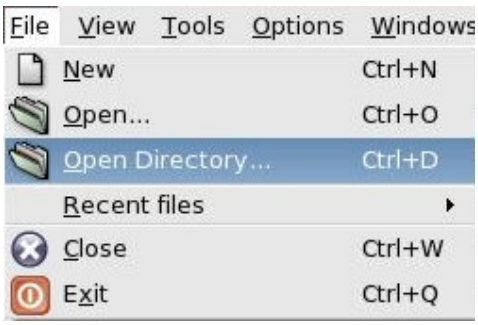


Figura 5.1: Interfície gràfica del menú per obrir un model.

Una vegada haguem seleccionat aquesta opció, s'obrirà una nova interfície a través de la qual podrem seleccionar el model. Aquesta finestra ens permetrà navegar per els diferents directoris i seleccionar el model concret que volem obrir.

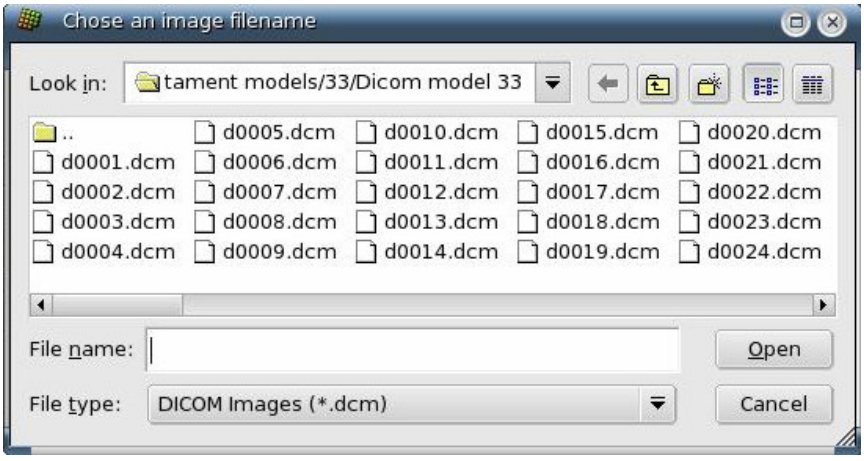


Figura 5.2: Interfície gràfica per seleccionar el model a obrir.

Una vegada haguem efectuat aquestes passes el model es carregarà en la plataforma i el podrem observar mitjançant el menú *"Axis View"* de la part esquerre de la pantalla.

5.2.- SELECCIÓ DE LA TÈCNICA

El següent pas que ha de realitzar l'usuari és seleccionar la tècnica que pretén executar. Per aquest motiu primer ha de seleccionar la pestanya corresponent, i a continuació la tècnica concreta mitjançant un *Combo Box*. En les imatges següents podem observar les diferents pestanyes disponibles i les diferents tècniques implementades dins de la pestanya de segmentació.

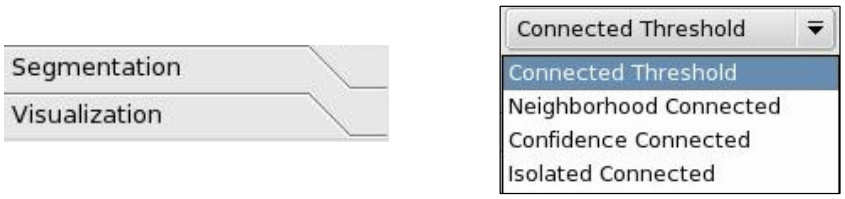


Figura 5.3: Pestanyes que permeten seleccionar l'acció a realitzar (esquerre) i *Combo Box* per seleccionar la tècnica de segmentació (dreta).

5.3.- MENÚ D'AJUDA

Dins de cada tècnica de segmentació i visualització disposem d'un menú d'ajuda (*Help)* que ens permetrà entendre el funcionament de la tècnica i ens indicarà el tipus de paràmetres que s'han d'introduir. Aquest menú està disponible amb català, castellà i anglès. Aquesta opció pot resultar útil per usuaris que no coneguin el funcionament de l'aplicació, permetent així reduir així el precipici d'execució. A la figura següent en podem observar un exemple.

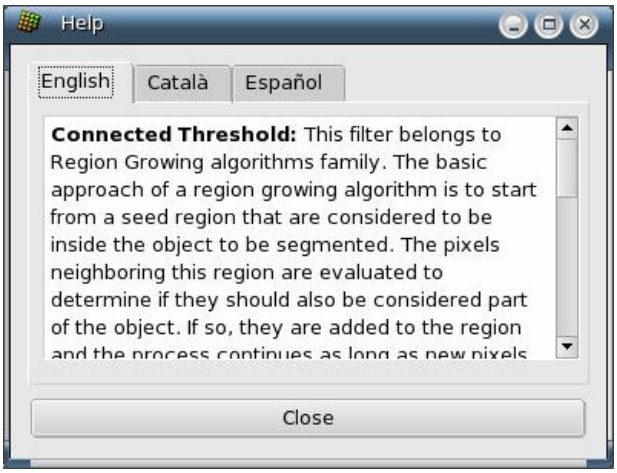


Figura 5.4: Menú d'ajuda en el cas de la tècnica Connected Threshold.

5.4.- COMPROVACIÓ DE PARÀMETRES

En els apartats de continuació mostrarem les interfícies gràfiques per introduir els paràmetres de cada una de les tècniques. Per evitar una introducció errònia, disposem d'un sistema de validació. Tot seguit mostrem un exemple del missatge que apareix al intentar executar una tècnica amb algun paràmetre incorrecte.

Figura 5.5: Missatge informatiu indicant que algun paràmetre és erroni.

Per complementar la informació també ressaltarem amb color vermell la part del formulari que conté l'error o paràmetre equivocat. A la figura següent podem veure'n un exemple.

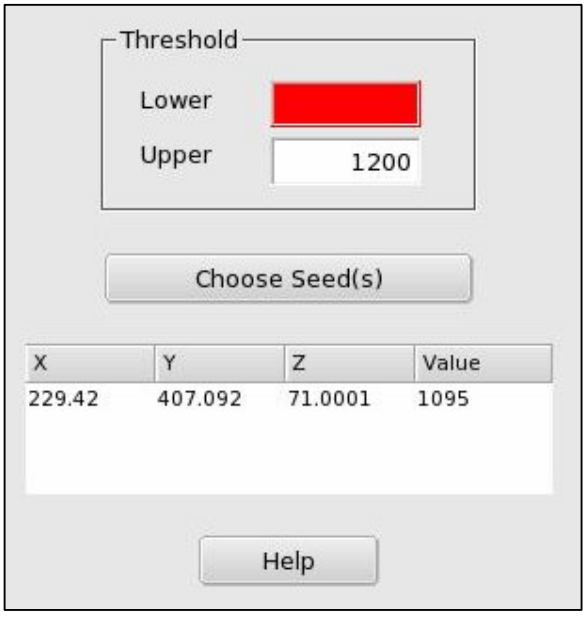


Figura 5.6: Formulari amb el paràmetre Lower Threshold incorrecte.

5.5.- TÈCNIQUES DE VISUALITZACIÓ

En aquest apartat ens centrarem amb les interfícies gràfiques de les tècniques de visualització: Ray Casting i Multiplanar. Pel que fa referència a la tècnica de Ray Casting observarem el formulari d'introducció de paràmetres i comentarem el funcionament de cada un d'ells. Per contra, la tècnica de Multiplanar no disposa de cap paràmetre, així doncs, únicament ens limitarem a mostrar la interfície gràfica que s'obté com a resultat d'executar la tècnica.

5.5.1- RAY CASTING

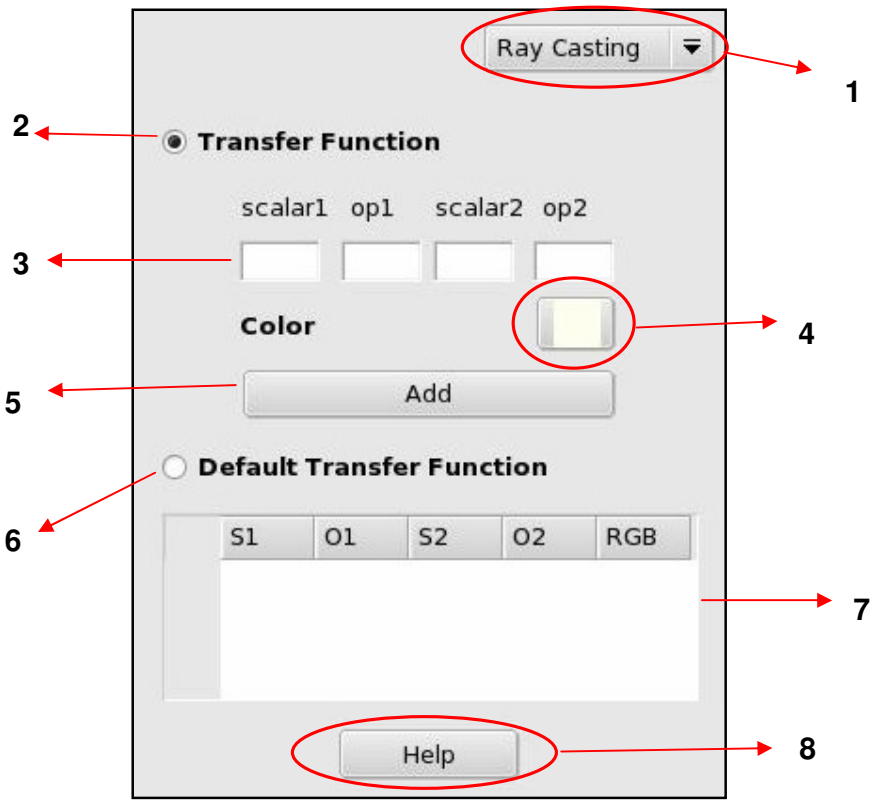


Figura 5.7: Formulari d'introducció de paràmetres de la tècnica de Ray Casting.

1.- Elecció de la tècnica de visualització: *Combo Box* que ens permet seleccionar el mètode de visualització que volem aplicar.

2.- Opció de funció de transferència manual: Seleccionant aquesta opció l'usuari podrà introduir la funció de transferència de manera manual mitjançant les utilitats corresponents als punts 3,4 i 5.

3.- Definició de la funció de transferència: L'usuari introdueix els rangs i les opacitats de la funció de transferència. Els rangs han de ser superiors a 0, mentre que les opacitats han d'oscil·lar entre 0 i 1. Si per exemple ens interessa que els punts amb valor de propietat 900 tinguin opacitat 0.4 i els de 1000 tinguin 0.5 llavors ompliríem les caixes amb els següents valors:

(900, 0.4, 1000, 0.5)

Per finalitzar la funció de transferència encara falta definir el color de cada rang.

4.- Selecció del color del rang: Mitjançant aquesta opció seleccionarem el color del rang entrat prèviament (punt 3). Ens permet escollir qualsevol color. A la figura següent podem observar la pantalla de selecció de color que ens apareix.

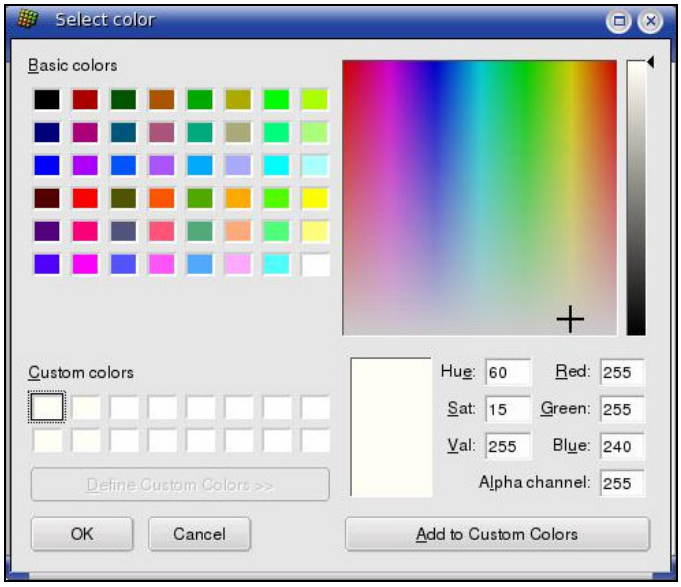


Figura 5.8: Interfície que permet la selecció del color.

5.- Introducció del rang: Una vegada seleccionats els rangs, opacitats i color hem de fer efectius aquests valors. Fent un clic sobre aquest botó veurem com ens apareix una nova entrada a la taula de sota. Aquest procés el podem repetir tantes vegades com rangs diferents ens interessi definir per visualitzar el model. A la figura 5.9 en veiem un exemple.

	S1	O1	S2	O2	RGB
1	900	0.4	1000	0.5	

Figura 5.9: Aspecte de la taula una vegada introduït un rang.

6.- Opció de funció de transferència automàtica: Utilitzant aquesta opció la taula de valors s'omplirà amb una funció de transferència per defecte tal com s'observa en la següent figura. D'aquesta manera l'usuari pot executar la tècnica més ràpidament.

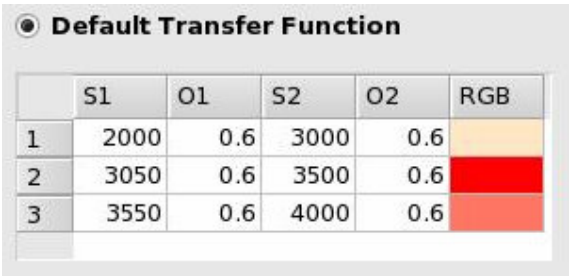


Figura 5.10: Aspecte de la taula una vegada introduïda la funció de transferència automàtica.

7.- Taula de valors: En aquesta taula s'hi afegiran els diferents rangs entrats de la funció de transferència. Com hem vist, és possible introduir els rangs de manera manual, o bé, utilitzar els rangs per defecte.

8.- Opció d'ajuda: Totes les tècniques implementades disposen d'una opció d'ajuda que ens mostra el funcionament de la tècnica i de cada un dels seus paràmetres. Aquest menú està disponible amb català, anglès i castellà.

.

Una vegada introduïts aquests paràmetres, i sempre hi quan els valors introduïts siguin correctes ja podem procedir a la seva execució. Per tal d'executar el mètode hem de fer un clic en el botó *Apply*. Quan seleccionem aquest botó és quan recuperarem els valors dels paràmetres i executarem la tècnica. El resultat final ens apareixerà en una nova pantalla. A continuació mostrem la interfície gràfica que ens ofereix els resultats del Ray Casting.

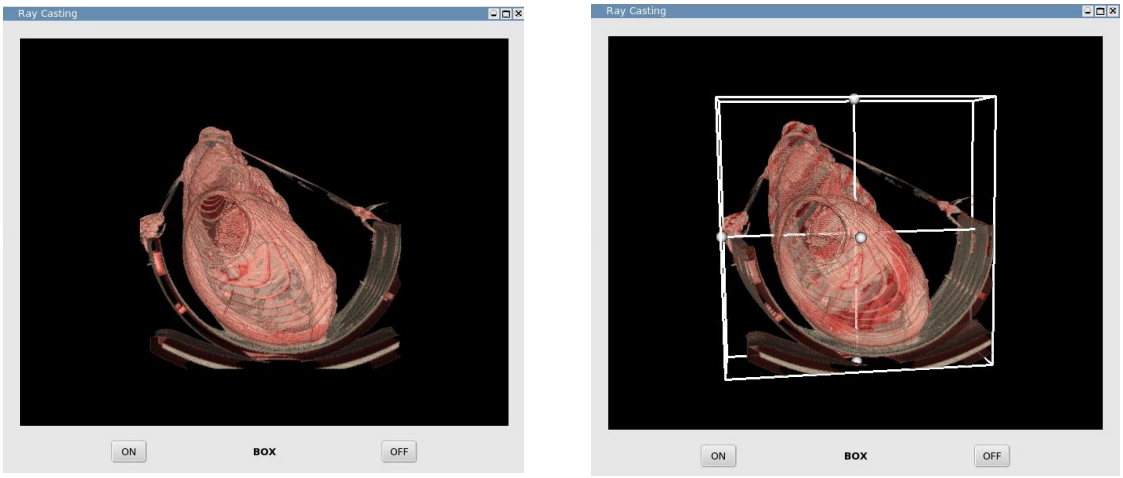


Figura 5.11: Interfície gràfica que mostra el resultat de la tècnica de visualització de Ray Casting. A l'esquerra observem el model sense utilitzar el *Box Widget* . A la dreta observem el model utilitzant el widget.

Aquesta interfície ens ofereix diverses accions. D'una banda tenim l'opció del zoom tant per apropar el model com per allunyar-lo. Aquesta acció es realitza mitjançant el ratolí. D'altra banda, disposem d'una opció per visualitzar el que anomenem un *Box Widget* sobre el model. Aquest *Box Widget* ens permet "endinsar-nos" dins el model i explorar-lo amb més detall. Per utilitzar-lo disposem de 5 direccions diferents (veure següent figura): una opció per explorar la part superior del model, una per la part inferior, una per l'esquerra, una per la dreta i finalment una per moure'ns en la profunditat del model.

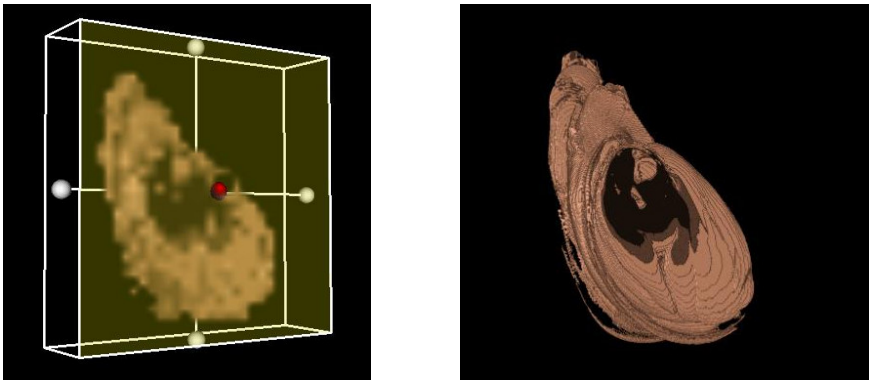


Figura 5.12: Utilització del *Box Widget* sobre el model (esquerra) i resultat d'aplicar el *Box Widget* sobre el model (dreta).

5.5.2- MULTIPLANAR

La tècnica de visualització Multiplanar no disposa de cap paràmetre. Per aquest motiu únicament hem de prémer sobre el botó *Apply* per executar-la. A continuació mostrem la interfície gràfica que ens apareix per mostrar els resultats.

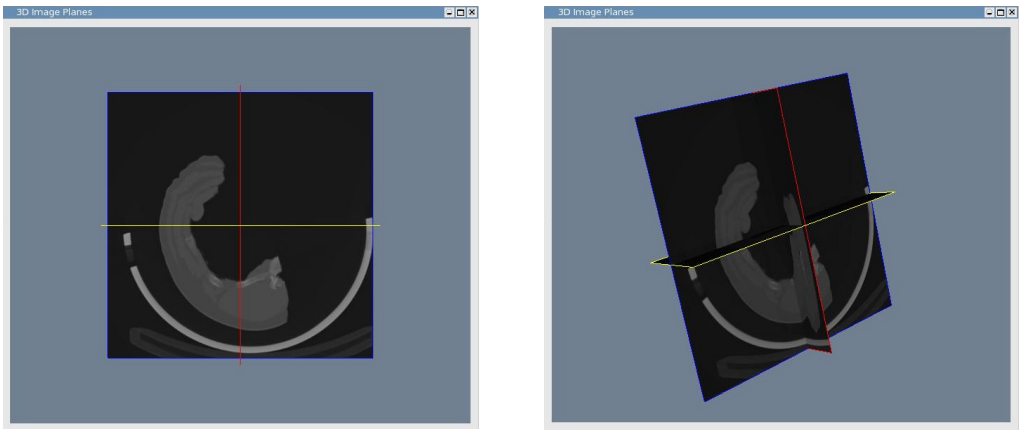


Figura 5.13: Interfície gràfica per observar els resultats de la tècnica Multiplanar.

Aquesta interfície ens permet avançar i retrocedir en les 3 vistes del model resultant (Axial, Sagital i Coronal). Per fer-ho simplement hem de seleccionar el pla desitjat amb el ratolí i efectuar el moviment. Al igual que el cas anterior també podem realitzar zoom sobre el model. Finalment, la interfície també ens ofereix informació sobre el punt que tenim seleccionat, és a dir, en el moment que situem el ratolí sobre un punt ens apareix un text informatiu indicant el valor d'intensitat i les coordenades.

5.6.- TÈCNIQUES DE SEGMENTACIÓ

En aquest apartat ens centrarem amb les interfícies gràfiques de les tècniques de segmentació. Primerament veurem la interfície que ens permet seleccionar la llavor o llavors a utilitzar. A continuació analitzarem les interfícies d'introducció de paràmetres de cada una de les tècniques. Finalment mostrarem la interfície gràfica per mostrar els resultats.

5.6.1- SELECTOR DE LLAVORS

La interfície de selecció de llavors està formada per les 3 vistes del model (Axial, Sagital i Coronal) més una taula on s'emmagatzemaran les llavors que l'usuari hagi seleccionat. A continuació podem observar una imatge de la interfície.

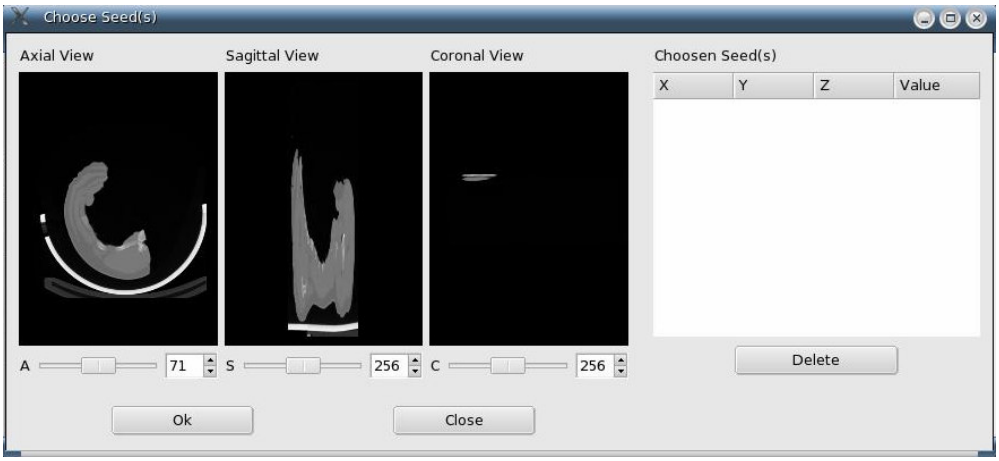


Figura 5.14: Interfície gràfica per seleccionar les llavors de la segmentació.

L'usuari pot explorar cada una d'aquestes vistes utilitzant els *sliders* (barres per desplaçar-se per les diverses llesques) col·locats a la part inferior de cada vista. Per seleccionar una llavor l'usuari únicament ha de prémer amb el ratolí sobre el punt desitjat. En aquest moment el nou punt s'afegirà a la taula de llavors, i també quedarà representat a les diverses vistes mitjançant un punt de color vermell (observar figura següent). Si en algun moment l'usuari vol modificar o esborrar alguna de les llavors afegides, únicament l'ha de seleccionar a la taula i prémer el botó Delete.

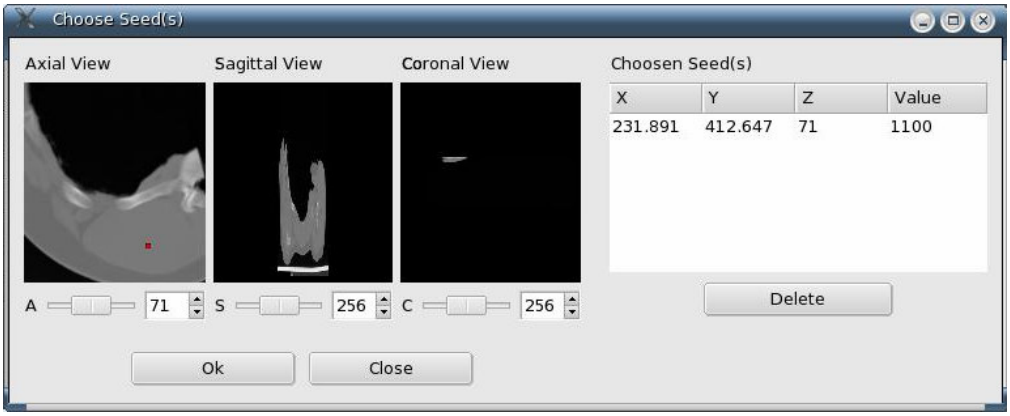


Figura 5.15: Selecció d'una llavor mitjançant la interfície gràfica.

A continuació mostrarem les interfície gràfiques per introduir els paràmetres de les tècniques de segmentació. Aquestes interfícies estan dividides principalment en dues regions. D'una banda disposen d'un apartat de selecció dels paràmetres "numèrics" com el threshold, el factor multiplicador, etc. D'altra banda disposen d'una taula per guardar les llavors que l'usuari hagi seleccionat mitjançant la interfície de selecció de llavors de l'apartat anterior.

5.6.2- CONNECTED THRESHOLD

El formulari d'introducció de paràmetres de la tècnica de Connected Threshold disposa de dues caixes de text per introduir els paràmetres lower threshold i upper threshold. També disposa d'una taula per guardar les llavors que l'usuari hagi seleccionat.

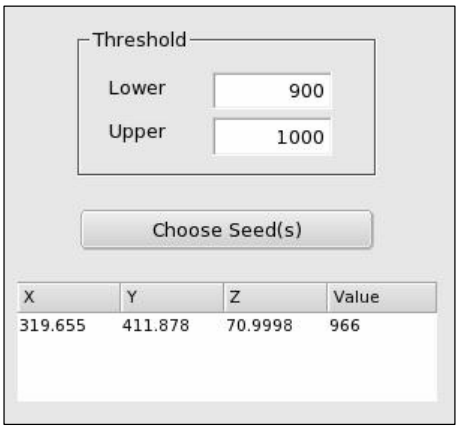


Figura 5.16: Interfície de selecció de paràmetres del Connected Threshold.

5.6.3- NEIGHBORHOOD CONNECTED

El formulari d'introducció de paràmetres de la tècnica de Neighborhood Connected disposa de tres caixes de text per introduir els paràmetres lower threshold, upper threshold i radi dels veïns a considerar. També disposa d'una taula per guardar les llavors que l'usuari hagi seleccionat.

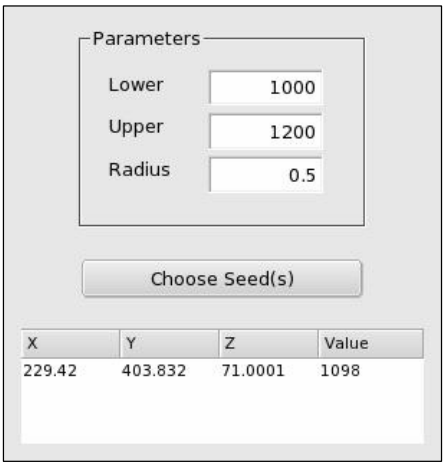


Figura 5.17: Interfície de selecció de paràmetres del Neighborhood Connected.

5.6.4- CONFIDENCE CONNECTED

El formulari d'introducció de paràmetres de la tècnica de Confidence Connected disposa de tres caixes de text per introduir els paràmetres del factor multiplicador, el número d'iteracions que realitzarà la tècnica i el radi inicial a considerar. També disposa d'una taula per guardar les llavors que l'usuari hagi seleccionat.

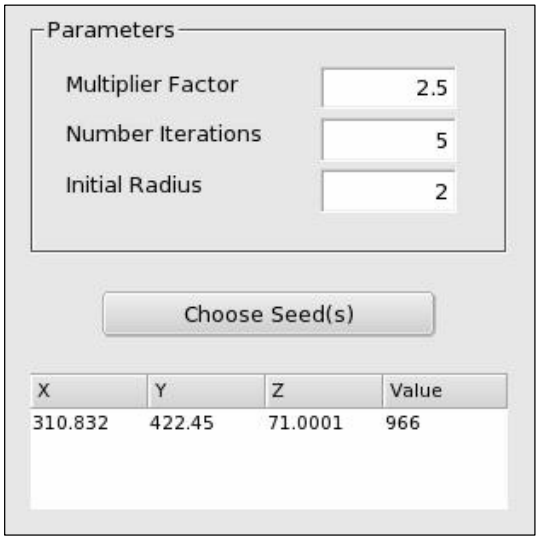


Figura 5.18: Interfície de selecció de paràmetres del Confidence Connected.

5.6.5- ISOLATED CONNECTED

El formulari d'introducció de paràmetres de la tècnica Isolated Connected disposa d'una caixa de text per seleccionar el paràmetre lower threshold. D'altra banda, disposa de dues taula per guardar la llavor situada dins el model a segmentar, i la llavor situada a l'exterior d'aquest model.

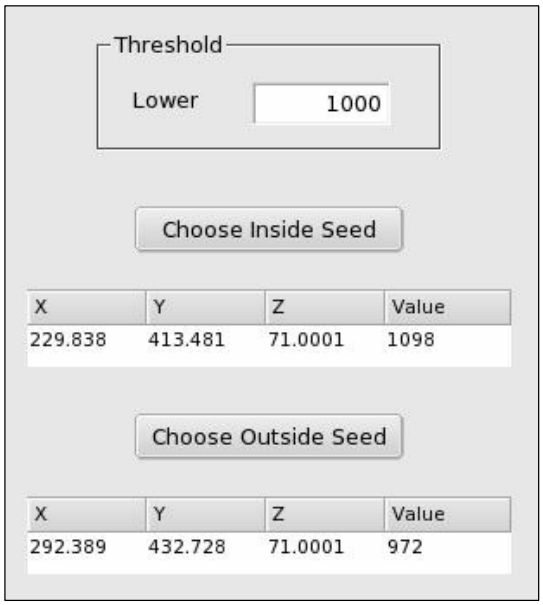


Figura 5.19: Interfície de selecció de paràmetres del Isolated Connected.

5.6.6- VISUALITZACIÓ DE RESULTATS

Totes les tècniques de segmentació implementades en aquest projecte utilitzen la mateixa interfície gràfica per mostrar els resultats. A la figura de continuació mostrem el funcionament i les opcions d'aquesta interfície.

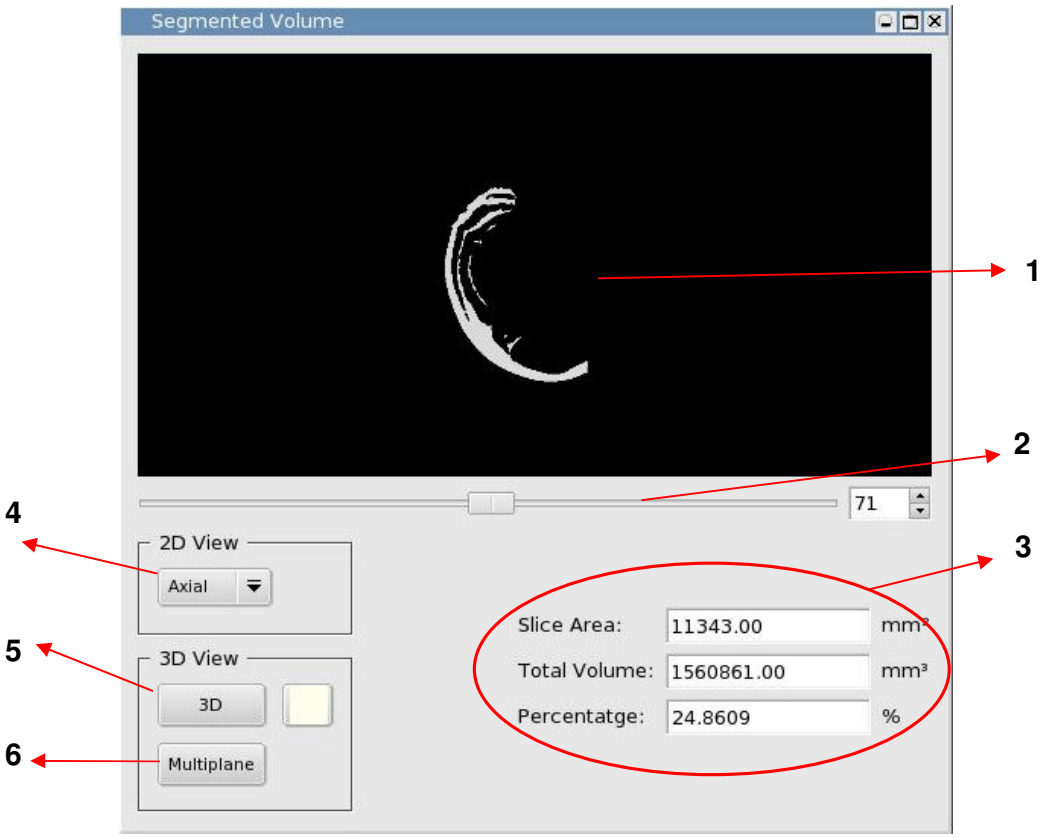


Figura 5.20: Interfície per visualitzar els resultats de les tècniques de segmentació.

1.- Visualització de la segmentació amb 2D: Aquesta regió és la que ens mostra el resultat de la segmentació amb dues dimensions.

2.- Slider per explorar les llesques: Permet desplaçar-nos per les diferents llesques del model de la vista que estigui seleccionada en aquell moment. Per seleccionar la vista veure el punt 4.

3.- Àrea de la llesca, volum total i percentatge: Aquesta part de la interfície mostra l'àrea en mm² de la llesca que estigui seleccionada en aquell moment, així com el volum total del model segmentat en mm³ i el percentatge d'aquest

volum segmentat respecte el volum total del model. Aquest últim aspecte és el més important, doncs ens permetrà saber el percentatge de grassa en la carn.

4.- Selecció de la vista: Aquesta opció ens permet seleccionar la vista 2D amb la que volem visualitzar el resultat. Les opcions possibles són Axial, Sagital o Coronal.

5.- Visualització 3D amb Ray Casting: Disposem de l'opció de visualitzar la segmentació obtinguda amb 3 dimensions mitjançant la tècnica de Ray Casting. Per fer-ho, únicament hem de seleccionar el color amb el que volem representar-ho i prémer el botó 3D. La interfície que ens apareixerà per manipular aquesta representació serà la mateixa que havíem observat anteriorment en al figura 5.11.

6.- Visualització Multiplanar: També disposem de l'opció de visualitzar la segmentació obtinguda amb la tècnica de Multiplanar. La interfície que ens apareixerà per manipular aquesta representació serà la mateixa que havíem observat anteriorment en l'apartat 5.5.2.

Resultats obtinguts

En aquest capítol veurem els resultats que hem obtingut aplicant les tècniques de segmentació i visualització que hem implementat. Per mostrar aquests resultats, dividirem el capítol en dos apartats. Primer de tot observarem les imatges obtingudes mitjançant les **tècniques de visualització**: Ray Casting i Multiplanar. A continuació mostrarem les imatges i resultats obtinguts aplicant les **tècniques de segmentació**: Connected Threshold, Neighborhood Connected, Confidence Connected i Isolated Connected.

Per realitzar aquestes proves hem utilitzat un model porcí que se'ns va subministrar. Es tracta d'un model de dimensions 512x512x146 adquirit mitjançant la Tomografia Computeritzada.

6.1.- TÈCNIQUES DE VISUALITZACIÓ

6.1.1.- Imatges obtingudes mitjançant el Ray Casting

Les imatges que mostrarem en aquest apartat estan obtingudes aplicant la tècnica de visualització de Ray Casting amb la funció de transferència de la figura 6.1.

	S1	O1	S2	O2	RGB
1	2000	0.6	3000	0.6	
2	3050	0.6	3500	0.6	
3	3550	0.6	4000	0.6	

Figura 6.1: Funció de transferència utilitzada per el Ray Casting.

Aquesta tècnica ens permetrà observar una representació 3D de tot el model porcí. Mitjançant aquesta representació podrem observar amb diferents tonalitats les diferents regions que formen el porc. A la figura de continuació podem observar el resultat de la tècnica.

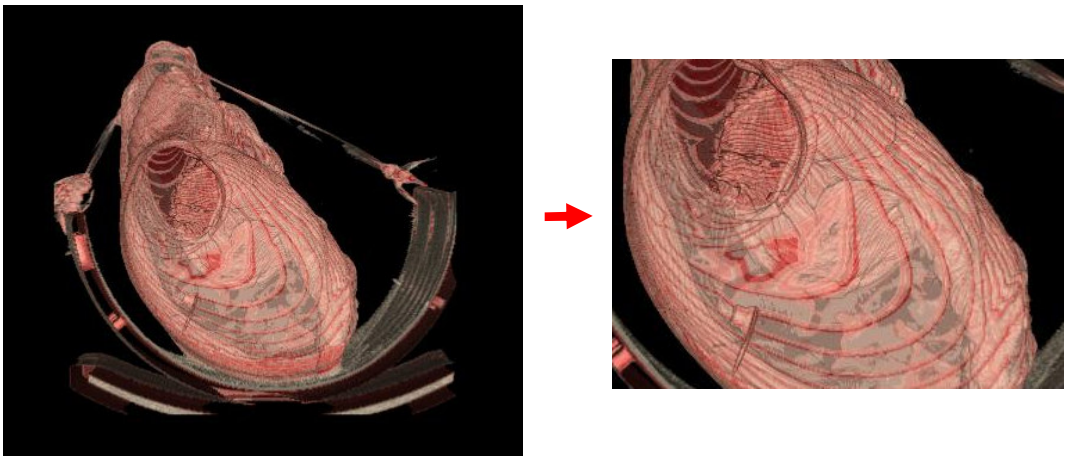


Figura 6.2: A l'esquerra observem el resultat d'aplicar Ray Casting sobre tot el model. A la dreta hem aplicat zoom sobre la imatge anterior.

Una vegada aplicada la tècnica de visualització Ray Casting podrem manipular el model obtingut. Tal com hem comentat, aquesta manipulació inclou una eina anomenada *Box Widget* que ens permet explorar el model des de diferents direccions (cada direcció que podem explorar correspon a una de les "boles" de la figura següent). Una vegada haguem finalitzat aquesta acció, podem deixar d'utilitzar aquesta eina i tornar a veure únicament el model resultant.

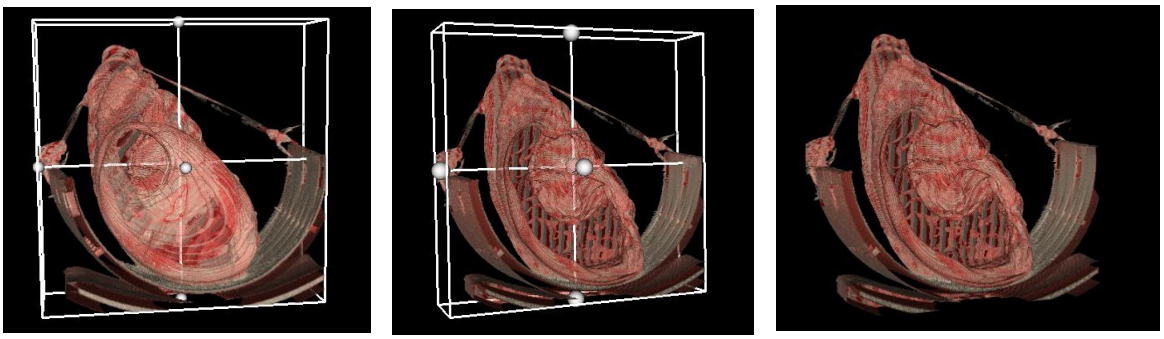


Figura 6.3: A la imatge de l'esquerra observem l'eina *Box Widget* sobre el model. Al mig observem com apliquem el *Box Widget*. A la dreta observem la imatge resultant d'aquesta acció.

6.1.2.- Imatges obtingudes mitjançant Multiplanar

Les imatges que observem a continuació estan obtingudes aplicant la tècnica de visualització Multiplanar sobre tot el model porcí. Aquesta tècnica ens representa les vistes Axial, Sagital i Coronal del model i ens permet desplaçar per cada una d'elles.

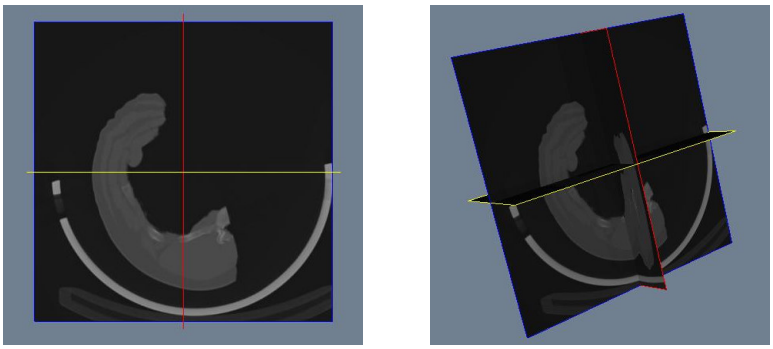


Figura 6.4: A l'esquerra observem la tècnica Multiplanar des d'un punt de vista centrat. A la dreta observem la mateixa imatge però rotada un cert angle.

Aquesta tècnica també ens informa sobre les propietats de cada punt del model. D'aquesta manera, quan l'usuari prem el ratolí sobre un punt de la pantalla, la tècnica ofereix a la part inferior de la pantalla la informació sobre les coordenades i el valor d'intensitat d'aquest punt. A la figura de continuació en podem veure una mostra.

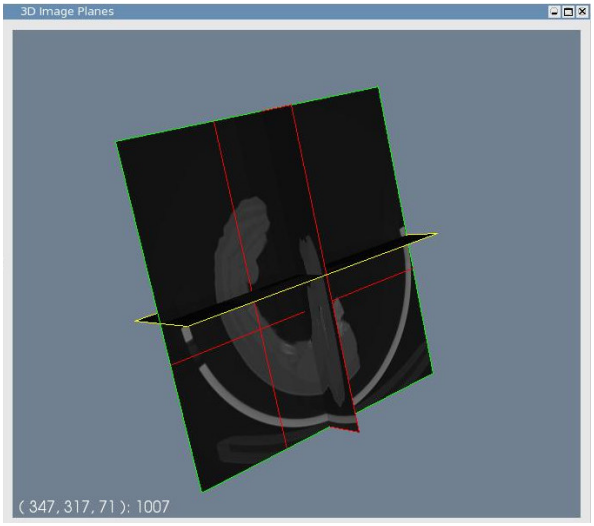


Figura 6.5: Al prémer sobre un punt la tècnica Multiplanar ens mostra les coordenades i el valor d'intensitat a la part inferior esquerra.

La tècnica també permet desplaçar cada una de les vistes per les diferents llesques que la composen. Per fer-ho l'usuari únicament ha de prémer sobre una vista i arrastrar el ratolí fins la llesca desitjada. També disposem de la possibilitat de realitzar zoom sobre alguna regió que ens interessi observar amb més detall i precisió. A continuació observem un exemple amb aquestes accions.

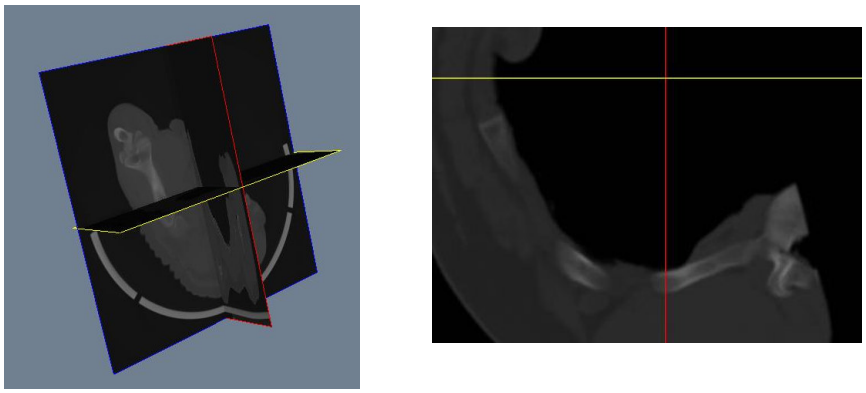


Figura 6.6: A l'esquerra observem com podem modificar i moure les llesques de les diferents vistes. A la dreta observem com podem efectuar zoom sobre les imatges.

6.2.- TÈCNIQUES DE SEGMENTACIÓ

6.2.1.- Connected Threshold

Els resultats que mostrem a continuació utilitzen com a paràmetres un *Lower Threshold* o llindar inferior de 900 i un *Upper Threshold* o llindar superior de 1000. D'aquesta manera segmentarem la regió corresponent a la grassa.

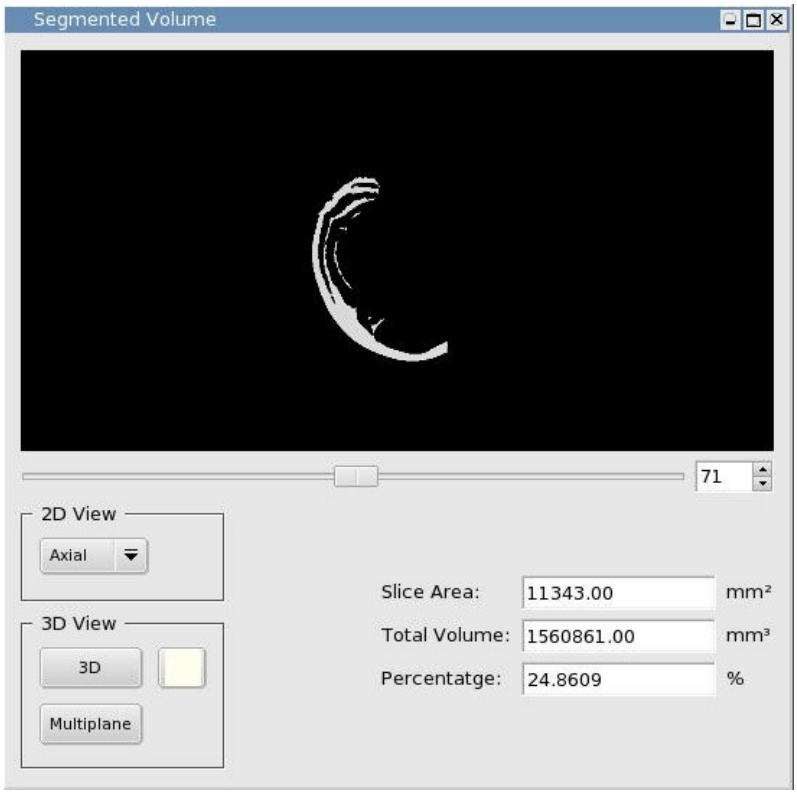


Figura 6.7: Resultat de la segmentació Connected Threshold.

Tal i com es veu a la figura anterior, com a resultat ens apareix el model segmentat i la informació numèrica d'aquest model (àrea de la llesca actual, volum i percentatge respecte el volum total). En aquesta ocasió observem com el percentatge de magre o grassa en la carn és del 24.8609 %. A més a més també tenim la possibilitat de visualitzar aquest model de dues maneres diferents: 3D corresponent al Ray Casting, o bé, utilitzant la visualització Multiplanar. A continuació mostrarem imatges obtingudes aplicant aquestes tècniques.

Al aplicar la visualització Multiplanar ens apareixerà el model segmentat amb les 3 vistes diferents (Axial, Sagital i Coronal). A partir d'aquí les opcions disponibles seran les mateixes que hem comentat en l'apartat 6.1.2.

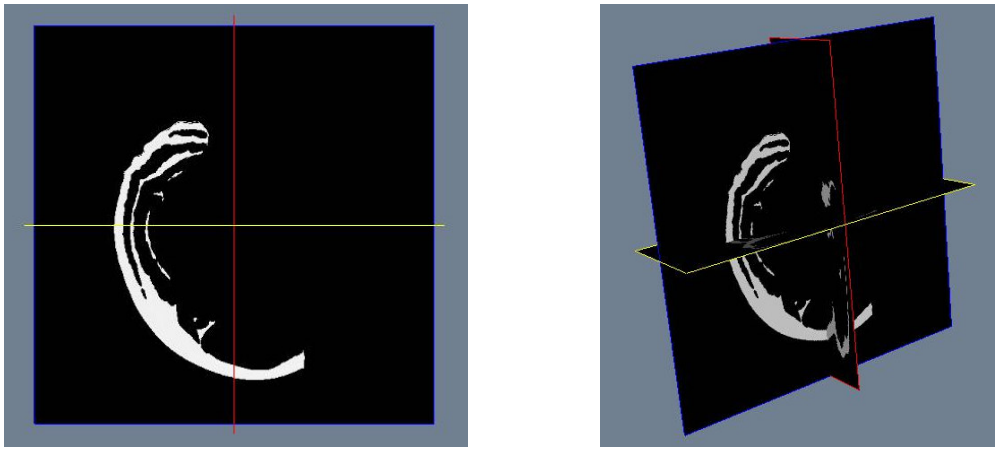


Figura 6.8: Resultat d'aplicar la visualització Multiplanar sobre la imatge segmentada.

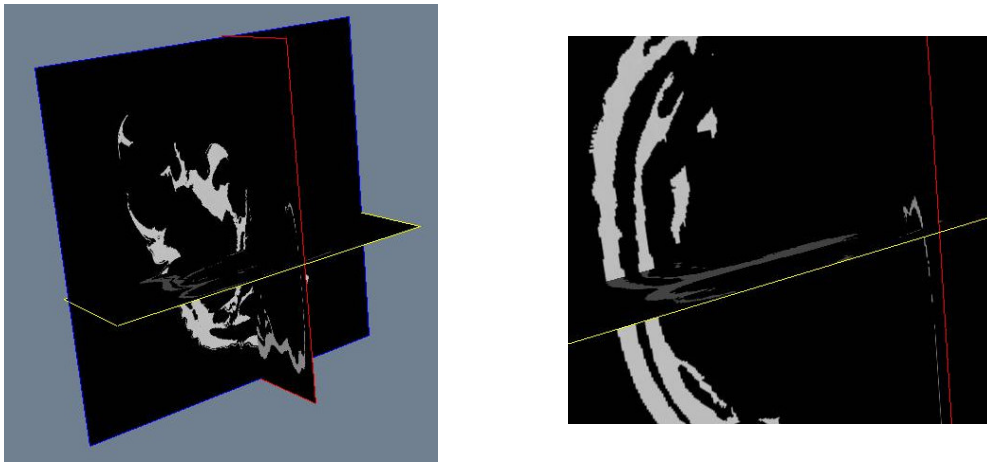


Figura 6.9: En la visualització Multiplanar podem desplaçar els plans per explorar el model (esquerra) i realitzar zoom (dreta).

Quan apliquem la visualització 3D amb Ray Casting del model segmentat disposarem de les mateixes opcions que hem comentat en l'apartat 6.1.1.

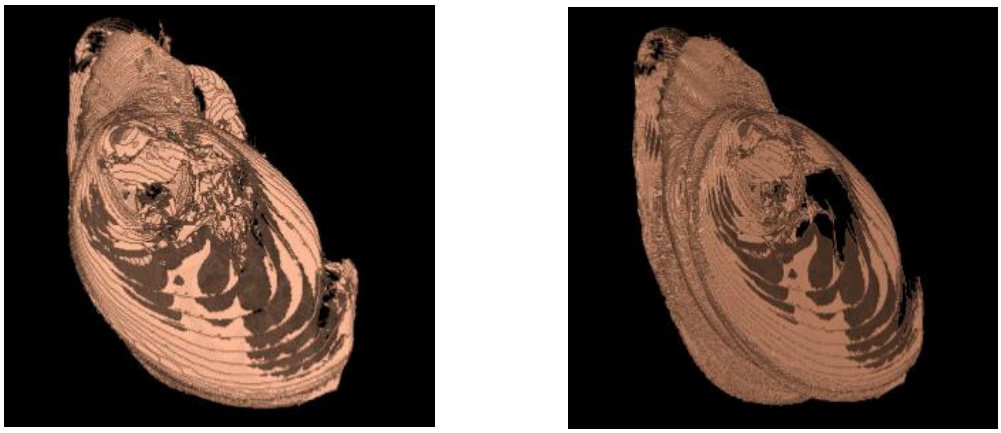


Figura 6.10: Resultat d'aplicar la visualització 3D Ray Casting sobre la imatge segmentada. A l'esquerra s'observa la imatge centrada i a la dreta la imatge rotada un cert angle.

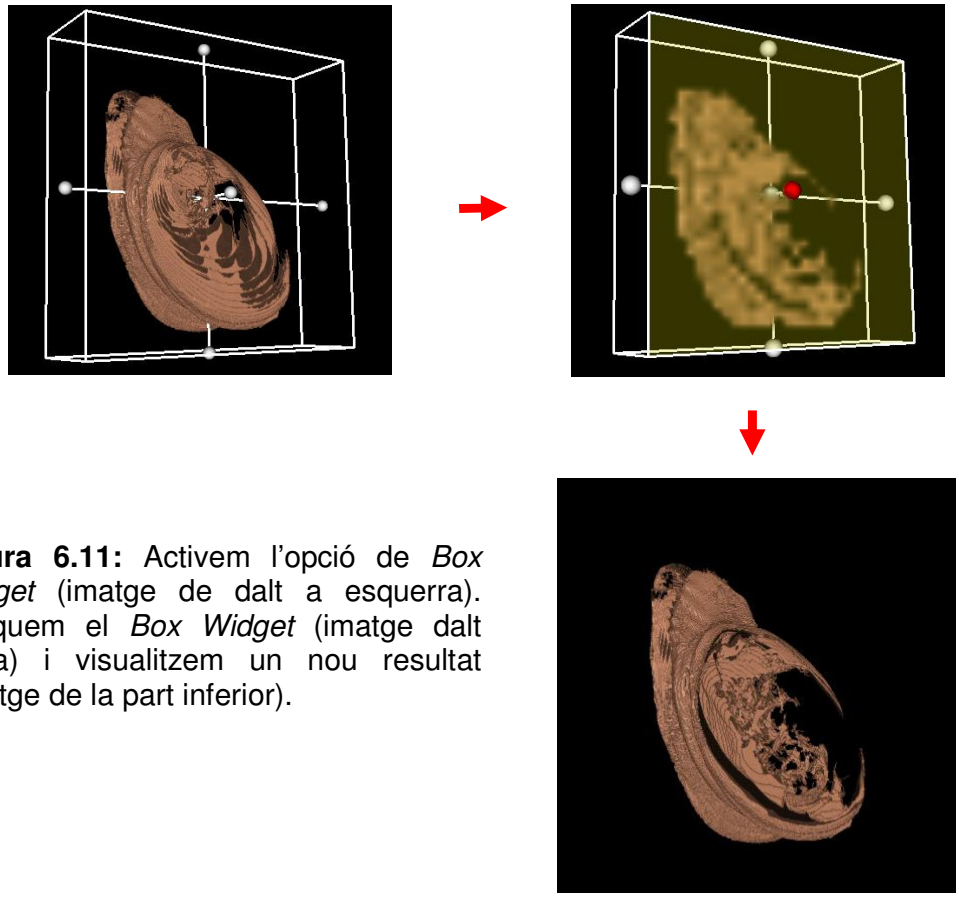


Figura 6.11: Activem l'opció de *Box Widget* (imatge de dalt a esquerra). Apliquem el *Box Widget* (imatge dalt dreta) i visualitzem un nou resultat (imatge de la part inferior).

6.2.2.- Neighborhood Connected

Els resultats que mostrem a continuació utilitzen com a paràmetres un *Lower Threshold* o llindar inferior de 900, un *Upper Threshold* o llindar superior de 1000 i un radi de veïns de 1. Si aquest últim paràmetre fos inferior a 1 els resultats d'aquesta tècnica serien els mateixos que el Connected Threshold. Això seria així ja que la distància mínima entre voxels és de 1, i per tant, al avaluar els veïns de distància < 1 no en trobaria cap.

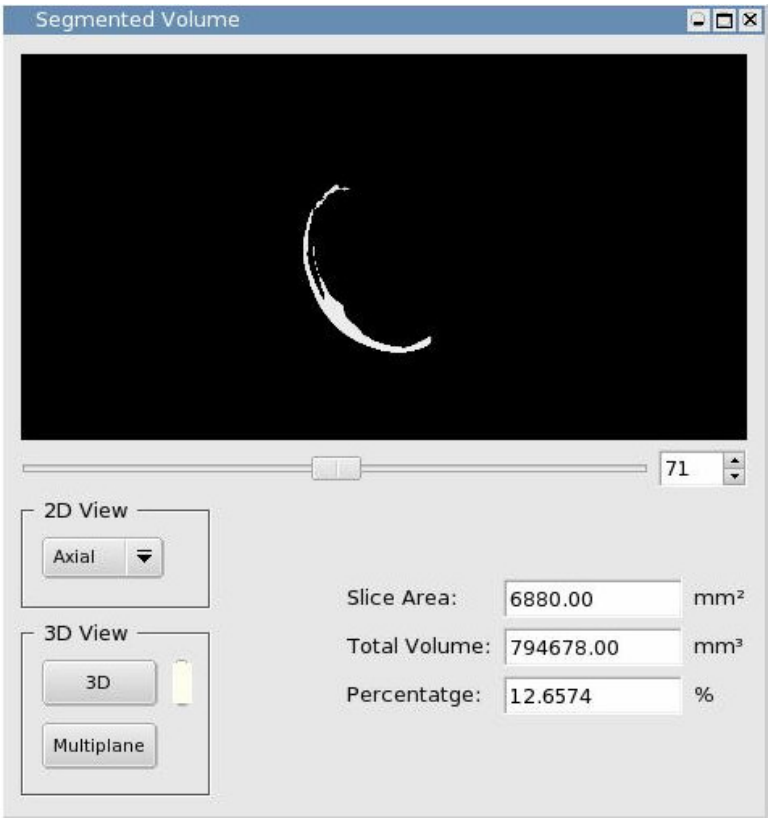


Figura 6.12: Resultat de la segmentació Neighborhood Connected.

En aquesta ocasió observem com el volum total segmentat és de 794678 mm^3, el qual representa un 12.6574% respecte el volum total del model. En el següent capítol de la memòria compararem aquests resultats amb els de les altres tècniques per veure quin s'ajusta millor.

A continuació mostrarem imatges obtingudes aplicant les tècniques de visualització sobre aquest model.

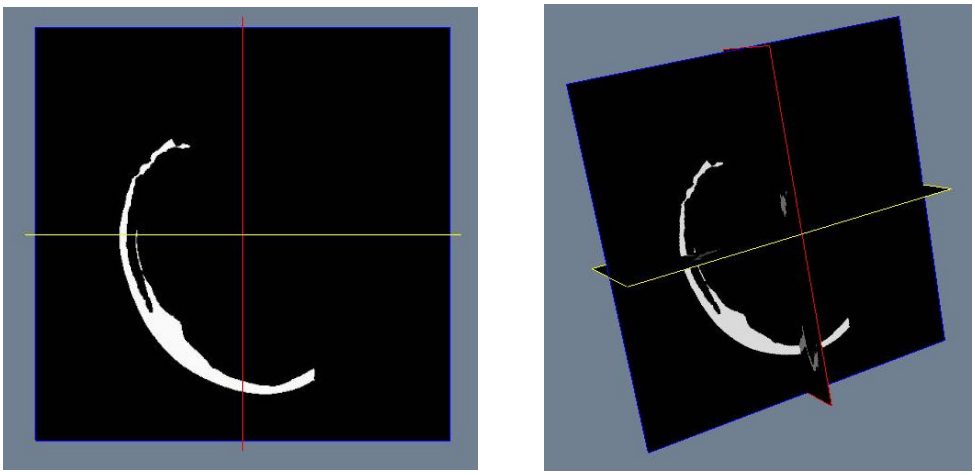


Figura 6.13: Resultat d'aplicar la visualització Multiplanar sobre la imatge segmentada amb el Neighborhood Connected.

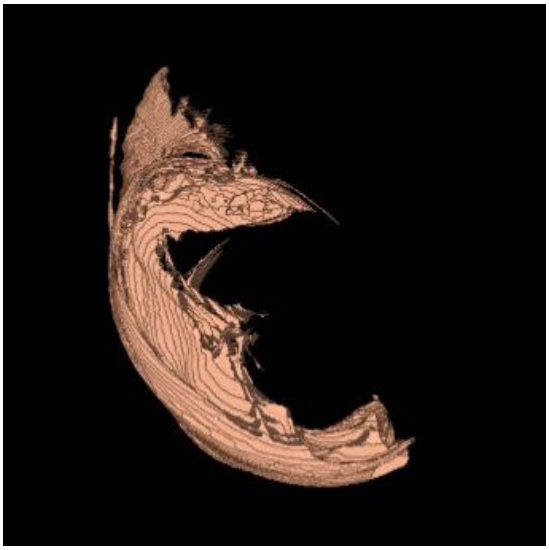


Figura 6.14: Resultat d'aplicar la visualització 3D Ray Casting sobre la imatge segmentada amb el Neighborhood Connected.

6.2.3.- Confidence Connected

Els resultats que mostrem a continuació utilitzen com a paràmetres un *factor multiplicador* de 2.5, 5 *iteracions* i un *radi inicial* de 2. D'aquesta manera segmentarem la regió corresponent a la grassa.

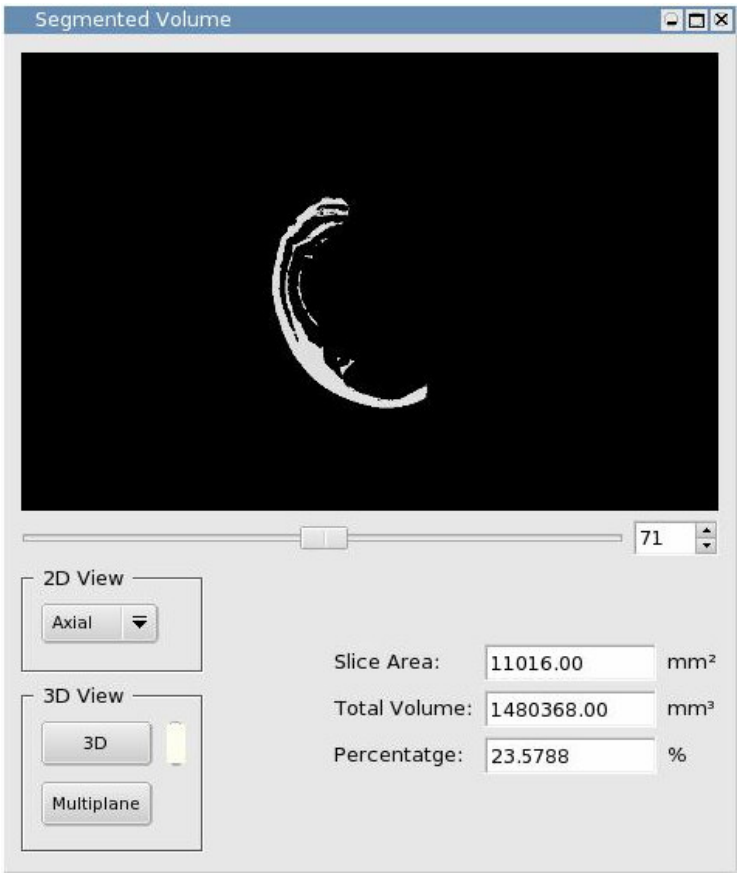


Figura 6.15: Resultat de la segmentació Confidence Connected.

En aquesta ocasió observem com el volum total segmentat és de 1480368 mm^3, el qual representa un 23.5788 % respecte el volum total del model. A continuació mostrarem imatges obtingudes aplicant les tècniques de visualització sobre aquest model.

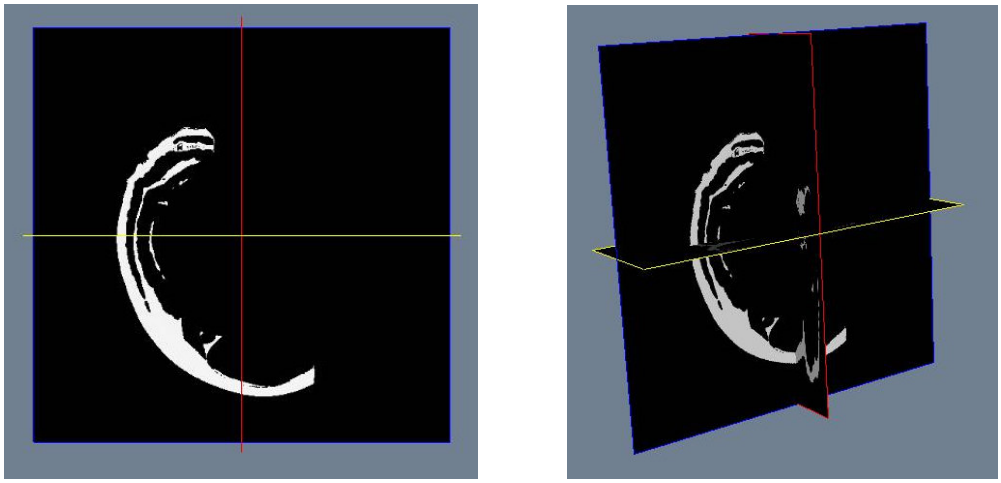


Figura 6.16: Resultat d'aplicar la visualització Multiplanar sobre la imatge segmentada amb el Confidence Connected.

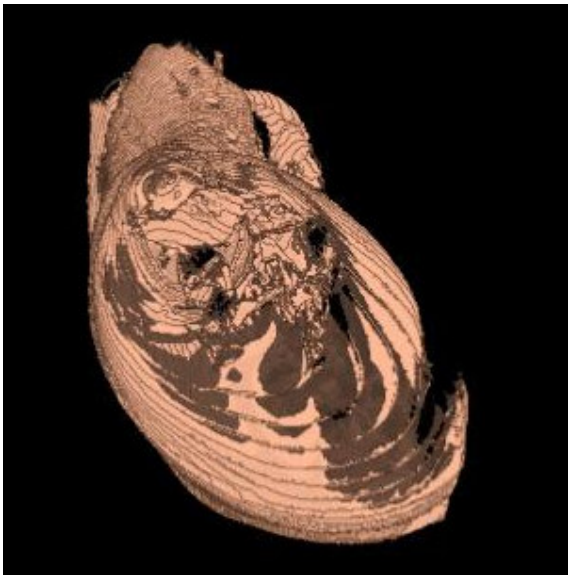


Figura 6.17: Resultat d'aplicar la visualització 3D Ray Casting sobre la imatge segmentada amb el Confidence Connected.

6.2.4.- Isolated Connected

Aquesta tècnica ha estat **incapaç de segmentar la regió de grassa**. Els paràmetres que rep com entrada són una llavor situada a l'interior de la regió a segmentar, una altre llavor situada a l'exterior d'aquesta regió i finalment un llindar inferior. La tècnica calcula el valor d'intensitat que pugui ser utilitzat com a llindar superior per la primera llavor. Malauradament en aquesta ocasió no ha pogut calcular aquest llindar, i per conseqüència no ha segmentat absolutament res. Així doncs, hem comprovat com aquesta tècnica no ens servirà per poder predir el percentatge de magre en la carn.

Hem realitzat una prova intentant segmentar altres regions i el resultat en aquesta ocasió si ha estat satisfactori. A continuació mostrem una figura amb la segmentació de la regió de carn i ós.

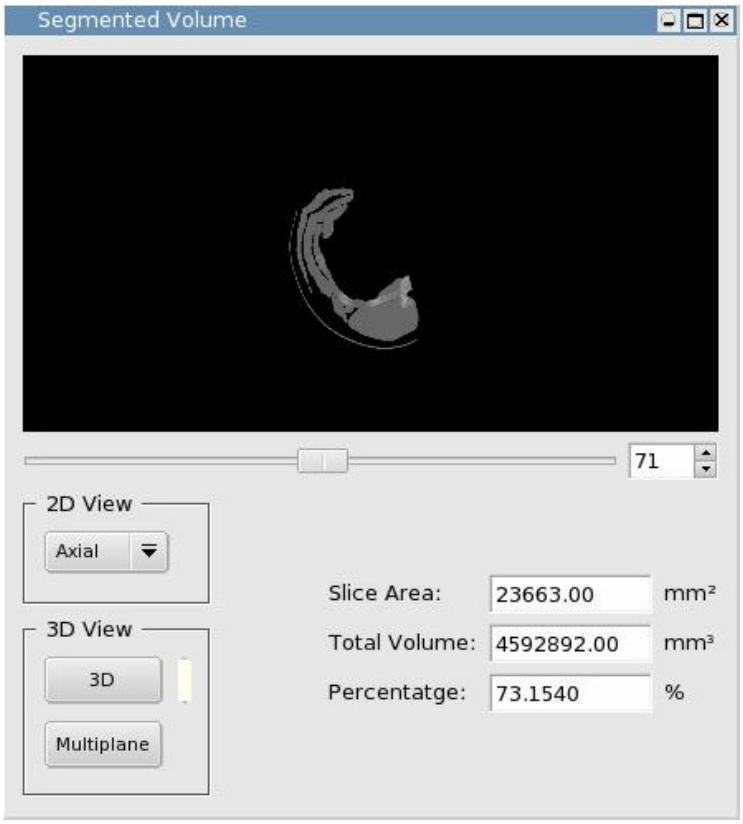


Figura 6.18: Resultat de la segmentació Isolated Connected.

Avaluació de resultats

Aquest capítol el dividirem en dos apartats. Primer de tot **avaluarem** el comportament de les diferents tècniques de segmentació utilitzant un **model sintètic**. Aquest és un model perfectament segmentat del qual coneixem els resultats que s'haurien d'obtenir.

A continuació **avaluarem** el comportament de les tècniques de segmentació utilitzant un **model porcí**. Compararem el resultat i els temps d'execució i decidirem quina tècnica ens ofereix millors solucions.

Comentar que l'ordinador amb el qual hem realitzat les simulacions no era el més adequat. Es tracta d'un *Intel Pentium 4 1700 MHz*, amb *512 Mg de RAM,* una targeta gràfica *Radeon 7000* i sota el sistema operatiu *Mandriva 2006*. Un ordinador amb unes característiques més avançades (targeta gràfica més potent, més memòria RAM, etc.) hauria reduït aquests temps d'execució. Hem de tenir present que l'ordinador sobre el qual acabarà treballant aquesta aplicació ja reuneix totes aquestes característiques.

Per últim destacar també que **aquest és un procés d'avaluació inicial**, doncs en un futur es comptarà amb el suport d'un expert en el tema per validar i determinar quina de les tècniques s'adapta millor a les necessitats.

7.1.- AVALUACIÓ UTILITZANT MODEL SINTÈTIC

Per efectuar aquesta avaluació utilitzarem el model sintètic *Brain Web* que simula l'estructura d'un cervell humà. Aquest model es pot aconseguir via web (veure apartat bibliografia). Existeixen diferents versions del model que es poden descarregar:

- D'una banda existeix l'anomenada **versió phantom**. Aquesta versió és la perfectament segmentada, és a dir, cada regió està formada per voxels d'igual valor d'intensitat. Així per exemple la regió de matèria blanca està composta únicament per voxels de valor 3. D'aquesta manera no hi haurà dificultat al considerar si el vòxel pertany o no a la regió (si és 3 pertany i si no és 3 no pertany), i per tant, podrem conèixer amb exactitud els resultats que haurien de donar.

- D'altra banda existeixen les anomenades **versions amb soroll**. A diferència del cas anterior aquestes versions no són "perfectes", és a dir, en aquestes les regions estan formades per voxels de diferents valors d'intensitat. Aquí per exemple la regió de matèria blanca pot estar formada per els valors d'intensitat del rang [130,160] aproximadament. Així per exemple els valors d'intensitat de 161 els descartarem, quan potser en realitat aquell valor també pertanyia a la regió. Existeixen diferents versions depenent del soroll del model: soroll 0, soroll 1, soroll 3, soroll 5, soroll 7 i soroll 9. Per realitzar les comparacions utilitzarem els models de soroll 0, soroll 3 i soroll 7. Evidentment com més soroll disposi el model més dificultat representarà en el moment de segmentar.

El que farem serà aplicar les 4 tècniques de segmentació amb els models amb soroll (soroll 0,3 i 7), i comparar aquests resultats amb els obtinguts amb el model *phantom*. D'aquesta manera podrem avaluar quina tècnica realitza una segmentació més precisa.

Primer de tot veurem l'estructura del model *phantom* i calcularem els resultats que ens hauria d'oferir.

7.1.1.- MODEL PHANTOM

Com hem vist aquest model està perfectament segmentat, de manera que, cada regió disposa únicament de voxels d'un valor d'intensitat. Concretament ens centrarem en segmentar la regió de matèria blanca corresponent als voxels amb valor d'intensitat 3. A la següent figura podem observar l'esquema d'aquest model *phantom*.

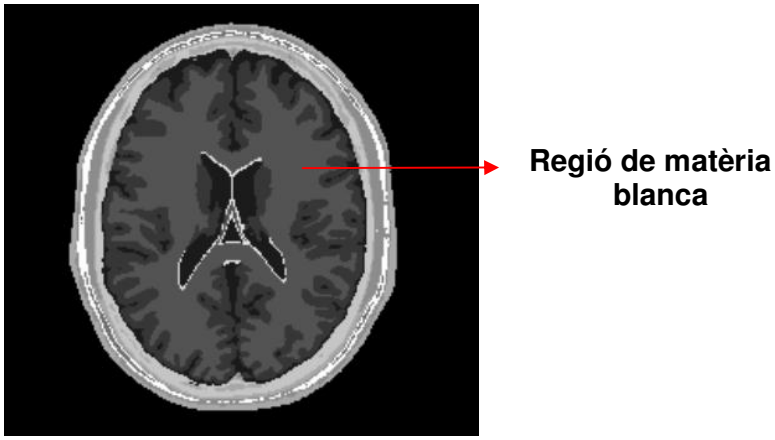


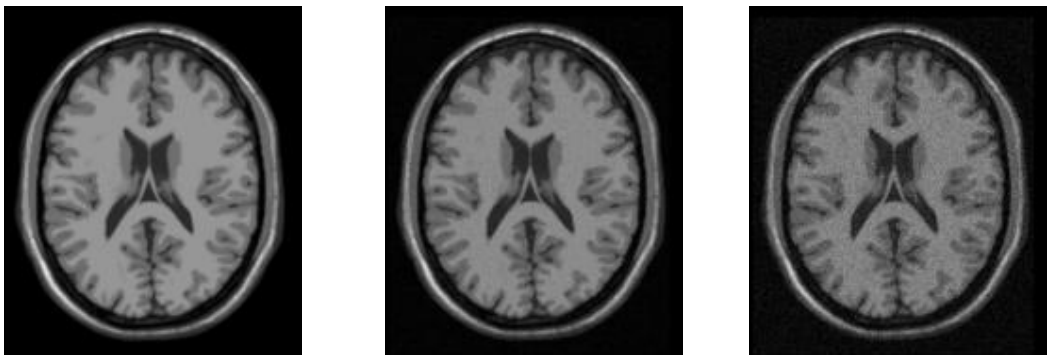
Figura 7.1: Estructura del model *phantom*.

Tal com està definit el model per calcular el volum de la regió de matèria blanca només hem hagut de realitzar un recorregut i comptabilitzar els voxels amb valor d'intensitat 3. El resultat calculat ha estat de **674777 mm³.**

Volum phantom	674777 **mm³**

7.1.2.- COMPARACIÓ DE RESULTATS

Per efectuar una avaluació aplicarem les tècniques amb els 3 models amb soroll i compararem els resultats amb el calculat amb el model *phantom*. A continuació veiem una figura amb els 3 models amb soroll que utilitzarem.



Soroll = 0 ***Soroll = 3*** ***Soroll = 7***

Figura 7.2: Diferents models amb soroll que utilitzarem.

Al fer l'avaluació de les diferents tècniques tindrem en compte **4 paràmetres:**

- Volum total segmentat.
- El nombre de voxels que no haurien d'haver estat segmentats però que la tècnica ha segmentat.
- El nombre de voxels que si haurien d'haver estat segmentats però que la tècnica no ha segmentat.
- El temps d'execució de la tècnica.

És important ressaltar que les comparacions que més ens interessaran seran les efectuades amb el model amb soroll 0. Els models amb més soroll (3 i 7) són molt complicats de segmentar, i per tant, és lògic si els resultats obtinguts varien considerablement respecte els esperats.

Connected Threshold

Hem utilitzat com a paràmetres un llindar inferior o *lower threshold* de 130 i un llindar superior o *upper threshold* de 160. A continuació mostrem una taula amb els resultats.

Connected Threshold	Soroll 0	Soroll 3	Soroll 7
Volum segmentat (mm^3)	663167	519365	209461
No eren però ha segmentat	8791	12	1
Si eren però no ha segmentat	20401	155424	465317
Temps	42 seg.	43 seg.	42 seg.

Com podem observar amb el model amb soroll 0 la tècnica ofereix uns resultats força bons. El volum total que segmenta és de **663167 mm^3**, enfront els 674777 mm^3 que hauria de donar. Malgrat això, no tots els voxels comptabilitzats dins aquest càlcul són correctes: 8791 dels que ha segmentat no ho haurien d'haver estat, mentre que 20401 no els ha segmentat quan realment si ho haurien de ser.

Per contra amb els models amb més soroll els resultats empitjoren considerablement. Amb el model de soroll 3 el volum segmentat únicament és de 519365, deixant-ne 155424 sense segmentar. Mentre que amb el model amb soroll 7 el nombre de voxels no segmentats encara és superior.

Pel que fa referència als temps d'execució veiem com en els 3 casos són similars i molt acceptables. Més endavant veurem com en el cas dels models porcins aquests temps són bastant superiors.

Neighborhood Connected

Hem utilitzat com a paràmetres un llindar inferior o *lower threshold* de 130, un llindar superior o *upper threshold* de 160 i un radi de 1. A continuació mostrem una taula amb els resultats.

Neighborhood Connected	Soroll 0	Soroll 3	Soroll 7
Volum segmentat (mm^3)	337854	244656	1
No eren però ha segmentat	180	0	0
Si eren però no ha segmentat	337103	430121	674776
Temps	55 seg.	49 seg.	47 seg.

En aquesta ocasió observem com els resultats són força més dolents comparats amb els que oferia la tècnica Connected Threshold. Amb el model soroll 0 la tècnica únicament ha segmentat **337854 mm^3**, deixant sense segmentar 337103 dels que ho eren, i segmentant-ne 180 que no haurien de ser.

En el cas del model amb soroll 3 també veiem com els resultats encara són lleugerament pitjors. La tècnica no ha estat de segmentar 430121 dels que pertanyien.

Finalment amb el model amb més soroll el resultat obtingut ha estat molt dolent ja que únicament ha segmentat 1 mm^3 (el corresponent a la llavor).

Confidence Connected

Hem utilitzat com a paràmetre un llindar inferior o *lower threshold* de 130. A continuació mostrem la taula amb els resultats.

Confidence Connected	Soroll 0	Soroll 3	Soroll 7
Volum segmentat (mm^3)	262046	618124	796272
No eren però ha segmentat	0	2467	125313
Si eren però no ha segmentat	412731	59120	3818
Temps	41 seg.	1 min 10 seg.	1 min 13 seg.

En aquesta ocasió observem com els resultats amb el model amb soroll 0 són bastant dolents. La tècnica únicament segmenta **262046 mm^3**.

Per contra amb els models amb soroll 3 i soroll 7 els resultats són força millors. Aquest és una dada molt destacada ja que la majoria de les tècniques són incapaces de segmentar correctament models amb gaire soroll. En el cas del model soroll 3 el volum segmentat ha estat de 618127 mm^3, enfront els 519369 mm^3 i 244656 mm^3 que oferien les tècniques Connected Threshold i Neighborhood Connected respectivament. Referent al model amb soroll 7 el volum segmentat ha estat de 796272 mm^3.

Destacar també que la segmentació dels models amb més soroll necessita d'un temps d'execució més elevat.

Isolated Connected

Els resultats obtinguts per aquesta tècnica han estat idèntics als obtinguts mitjançant el Connected Threshold. Tal com hem vist en el capítol 3 d'aquesta memòria, la tècnica Isolated Connected està pensada per segmentar regions petites i difícils de diferenciar. En el cas de regions com les que estem tractant aquesta tècnica ofereix resultats molt similars o idèntics al Connected Threshold (malgrat que tal com hem vist en el capítol anterior és incapaç de segmentar la regió de grassa en els models porcins).

Com a resum final d'aquesta comparació podríem dir que en línies generals les tècniques que ens ofereixen millors resultats amb aquest model sintètic són la **Connected Threshold i la Isolated Connected**. Per contra, en el cas de **models amb soroll** la millor tècnica és la **Confidence Connected**. A continuació realitzarem una comparació de les tècniques utilitzant el model porcí, i veurem si es continuen mantenint aquests resultats.

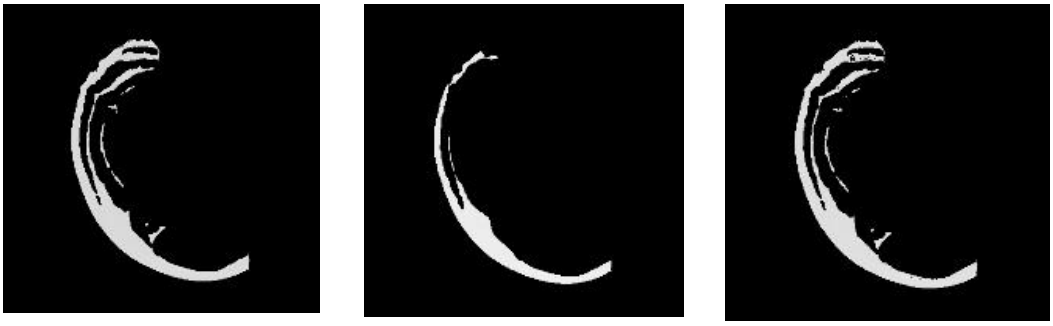
7.2.- AVALUACIÓ UTILITZANT MODEL PORCÍ

Per efectuar aquesta avaluació utilitzarem el model porcí que se'ns va subministrar. Es tracta d'un model de dimensions 512x512x146 adquirit mitjançant la Tomografia Computeritzada.

7.2.1.- COMPARACIÓ DE RESULTATS

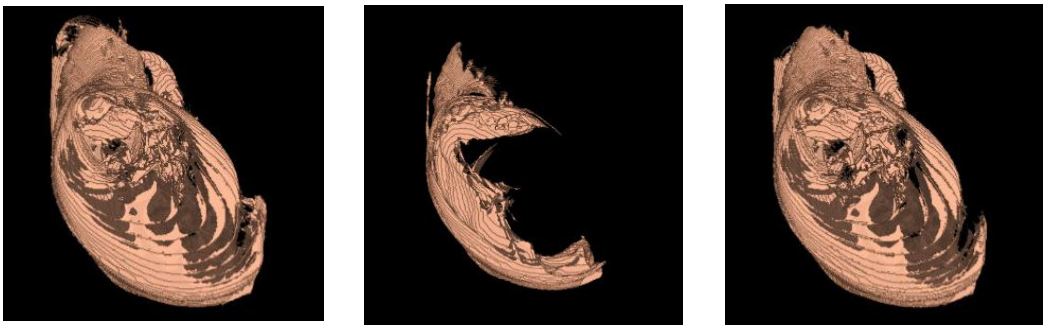
A continuació mostrarem una taula comparativa i unes imatges amb els resultats obtinguts amb cada tècnica. Tal com hem comentat en el capítol anterior de la memòria, la tècnica Isolated Connected no ha estat capaç de segmentar la regió de grassa.

Segmentació grassa	Volum segmentat (mm^3)	Percentatge (%)	Temps
Connected Threshold	1560861	24.8609	4 min 10 seg.
Neighborhood Connected	794678	12.6574	4 min 25 seg.
Confidence Connected	1480368	23.5788	5 min. 20 seg.
Isolated Connected	-	-	-



Connected Threshold **Neighborhood Connected** **Confidence Connected**

Figura 7.3: Comparació de les imatges obtingudes aplicant les tècniques de segmentació.



Connected Threshold **Neighborhood Connected** **Confidence Connected**

Figura 7.4: Comparació de les imatges obtingudes aplicant la visualització 3D sobre els models segmentats de la figura anterior.

Tal com s'observa a la taula i a les imatges anteriors, les tècniques que ofereixen una millor segmentació són la Connected Threshold i la Confidence Connected. En el cas del Connected Threshold la regió segmentada té un volum de 1560861 mm^3 que representa un 24.8609 % del total. Amb el Confidence Connected la regió segmentada té un volum de 1480368 mm^3 que representa un 23.5788 % del total. Al igual que havíem vist en l'apartat anterior el Connected Threshold segueix essent una de les tècniques que millor segmentació ofereix.

En el cas de la tècnica Neighborhood Connected observem la mateixa tendència que en el cas del model sintètic: la tècnica segmenta molt menys volum del que hauria de segmentar.

Pel que fa referència als temps d'execució observem com són força elevats. Per segmentar el model porcí totes les tècniques necessiten més de 4 minuts. L'ordinador que hem utilitzat per efectuar les proves, així com les dimensions del model porcí són causa d'aquests resultats.

Com es pot veure a la taula també existeix una diferència de temps considerable entre l'execució del Connected Threshold i del Confidence Connected. En el primer cas els temps són de 4 minuts 10 segons, mentre que amb el Confidence Connected els temps són de 5 minuts 20 segons. D'aquesta manera, podem concloure que **la tècnica que ens ofereix millors resultats és la Connected Threshold**.

Millores i treball futur

Tot i que hem assolit els objectius fixats al inici del projecte, som conscients que l'aplicació desenvolupada no està complerta, calen certes ampliacions i millores en aspectes concrets. En aquest capítol volem deixar constància d'aquest fet i per això proposem algunes millores i treballs futurs que ens interessaria integrar a la plataforma.

- **Editor de contorns**

  Seria interessant disposar d'una eina que permetés editar els contorns de les segmentacions. D'aquesta manera, una vegada apliquéssim una tècnica de segmentació, l'usuari disposaria de la possibilitat de modificar/editar aquest resultat. Això seria interessant ja que l'automatització total del procés de segmentació pot ser que no arribi a satisfer les necessitats.

- **Reduir el temps d'execució de les tècniques de segmentació**

  Actualment els temps d'execució de les tècniques de segmentació amb models porcins són força elevats. Resultaria interessant trobar millores i tècniques que ens permetessin disminuir encara més aquests temps necessaris.

- **Guardar els resultats**

  Estaria bé disposar de la possibilitat de guardar els resultats de les segmentacions. D'aquesta manera si l'usuari desitja en un futur consultar les dades obtingudes, disposaria d'una manera senzilla i ràpida de poder-ho fer.

- **Avaluació de les tècniques**

  Està pensat en un futur aprofundir en l'apartat d'avaluació de les tècniques, i veure quina tècnica s'ajusta millor a les necessitats, mitjançant l'ajut d'un expert en el tema.

Conclusions

Un dels aspectes més importants en el procés de tractament del porc és com determinar la seva qualitat. En la Unió Europea, la metodologia utilitzada per determinar-la és la que s'anomena la dissecció. La dissecció utilitza el contingut de magre en la carn com a referència comú per determinar la qualitat de la canal. Aquest procés és totalment manual per la qual cosa resulta laboriós i costós, tant pel que fa referència al temps que implica com per la seva depreciació.

L'objectiu d'aquest projecte ha estat crear un entorn informàtic que integri les eines bàsiques que permetin automatitzar al màxim aquest procés de determinació de la qualitat porcina. Per fer-ho ha estat necessari: (i) obtenir les dades del porc mitjançant tècniques d'adquisició no invasives, com la Tomografia Computeritzada i (ii) disposar de tècniques de segmentació i visualització d'imatge, que permetessin processar les dades per tal de determinar el contingut de magre o grassa en la carn amb una precisió similar a la dissecció.

Les tècniques de segmentació són les que ens permetran separar o extreure la regió de grassa. Posteriorment aplicarem un algoritme de càlcul de volums per poder deduir el percentatge de magre en la carn. Existeixen diverses classificacions d'algoritmes de segmentació: Region Growing, Watershed, Level Set, etc. Concretament en aquest projecte hem implementat 4 tècniques de Region Growing: Connected Threshold, Neighborhood Connected, Confidence Connected i Isolated Connected.

D'altra banda, les tècniques de visualització ens permetran representar amb 3 dimensions els model porcins i els resultats de la segmentació. En aquest projecte hem utilitzat la tècnica de visualització directa de volums anomenada Ray Casting. També disposem d'una altre tècnica de visualització anomenada Multiplanar que mostrarà les 3 vistes del model (vista Axial, Sagital i Coronal) i ens permetrà desplaçar-nos per les diferents llesques.

Finalment hem realitzat un últim apartat de comparació per determinar quina tècnica de segmentació ens oferia millors resultats. Després de l'avaluació inicial realitzada (en un futur es realitzarà una avaluació més detallada amb l'ajuda d'un expert), hem vist com la tècnica de Connected Threshold era la que efectuava una segmentació més precisa en un menor temps.

Per últim, també m'agradaria comentar el treball en equip que hem hagut de realitzar per aconseguir els objectius d'aquest projecte. Com hem dit, aquesta aplicació ha estat integrada en una plataforma on hi treballen més persones. La bona comunicació entre tots els membres que hi estem implicats també ha afectat positivament en el resultat final.

Índex figures

Bibliografia

[Dal02] "Programming with QT" 2nd Edition by Matthias Kalle Dalheimer. Ed. O'REILLY, 2002, ISBN: 0-596-00064-2.

[Ope97] "Open GL Reference Manual" Second Edition by Renate Kempf and Chris Frazier. Ed. Addison-Wesley, 1997, ISBN: 0-201-46140-4.

[Woo] "OpenGL Programming Guide" Second Edition by Mason Woo, Jackie Neider, Tom Davis, Addison-Wesley. ISBN: 0-201-46138-2.

[Sch02] "VisualizationToolkit" Third Edition by Will Schroeder, Ken Martin and Bill Lorensen. ISBN: 1-930934-07-6.

"The ITK software guide" Second Edition Updated for ITK version 2.4 by Luis Ibáñez, Will Schroeder, Lydia Ng, Josh Cates and the *Insight Software Consortium.*

[Sch03] "VTK User's Guide" by Will Schroeder, Ken Martin and Bill Lorensen. ISBN: 1-930934-08-4.

[Bja98] "El lenguaje de programación C++" by Bjarne Stroustrup. Turpial, Addison Wesley 1998.

[Fol94] "Introduction to computer Graphics" by James D.Foley, Andries van Dam, Steven K.Feiner, John F. Hughes John F Hughes, Addison-Wesley. ISBN: 0-201-60921-5.

[Gla89] "An introduction to Ray tracing" by A.S.Glassner. Academic Press.

Adreces d'Internet:

Informació llibreries Qt:
http://www.trolltech.com

Informació llibreries OpenGL:
http://www.opengl.org

Informació llibreries VTK:
http://public.kitware.com/VTK

Informació llibreries ITK:
http://www.itk.org

Informació sobre tot tipus de dispositius de captació (CT, MRI,UI ,SPECT, etc):
http://www.bodyscan.md/

Informació sobre els equips de classificació porcina AUTOFOM:
http://www.sfktech.com/pdf-filer/AutoFom-spansk_ver1.pdf
http://www.sfktech.com/products/measuring.fom.html

Informació sobre les tècniques de visualització directa:
http://www.cs.umbc.edu/~ebert/693/NLiao/693.html
http://www.cs.rug.nl/~michel/waveletsplatting/wavelets.html

Informació sobre el Ray Casting:
http://www.cs.wpi.edu/~matt/courses/cs563/talks/powwie/p1/ray-cast.htm