ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ
ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

# Ανάπτυξη γραφικού εργαλείου ανακατασκευής τομογραφικής εικόνας μονοφωτονικής τομογραφίας εκπομπής SPECT

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ζώτος Παντελής

**Επιβλέπων :** Κωνσταντίνα Νικήτα
Καθηγήτρια Ε.Μ.Π

Αθήνα, Ιούνιος 2008

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ
ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

# Ανάπτυξη γραφικού εργαλείου ανακατασκευής τομογραφικής εικόνας μονοφωτονικής τομογραφίας εκπομπής SPECT

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ζώτος Παντελής

**Επιβλέπων :** Κωνσταντίνα Νικήτα
Καθηγήτρια Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 2 Ιούνη 2008.

| Κ.Νικήτα | Ν. Ουζούνογλου | Δ. Κουτσούρης |
|---|---|---|
| Καθηγήτρια Ε.Μ.Π. | Καθηγητής Ε.Μ.Π. | Καθηγητής Ε.Μ.Π. |

Αθήνα, Μάιος 2008

Παντελής Ζώτος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

# Ευχαριστίες

Θα ήθελα να εκφράσω την ευγνωμοσύνη μου στην επιβλέπουσα καθηγήτρια μου, Δρα Νικήτα Κωνσταντίνα, Καθηγήτρια της σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Ηλεκτρονικών Υπολογιστών του ΕΜΠ. Την ευχαριστώ για την ευκαιρία που μου έδωσε να συνεργαστώ μαζί της, την προθυμία της να παρέχει λύσεις σε ζητήματα που παρουσιάστηκαν και την εμπιστοσύνη που μου έδειξε για τη λήψη αποφάσεων και πρωτοβουλιών. Η διδασκαλία, η καθοδήγηση, η εμπειρία και οι γνώσεις της ήταν άκρως απαραίτητα για τη διεξαγωγή και ολοκλήρωση της διπλωματικής εργασίας.

Επίσης, ευχαριστώ τον υποψήφιο διδάκτορα κ. Μπούκη Σπύρο για την βοήθεια του στην γνωριμία μου και την εξοικείωσή μου με το προγραμματιστικό εργαλείο Qt με στόχο την ευκολότερη υλοποίηση του γραφικού περιβάλλοντος της εφαρμογής που αναπτύξαμε στην παρούσα εργασία.

Τέλος, την πραγματοποίηση της παρούσας εργασίας οφείλω κατά το μεγαλύτερο μέρος στον Δρα Λούντο Γιώργο. Τον ευχαριστώ για τη στήριξη, τις ατελείωτες ώρες που αφιέρωσε στην εργασία παρά το βαρυφορτωμένο του πρόγραμμα, την υπομονή του, τις γνώσεις που τόσο πρόθυμα μου μετέδωσε, τη φιλική του διάθεση και την καθοδήγηση του. Οι αστείρευτες γνώσεις, το ενδιαφέρον και ο ενθουσιασμός του για το αντικείμενο της ιατρικής απεικόνισης, αποτέλεσαν για μένα παράδειγμα προς μίμηση. Η εμπειρία και οι γνώσεις που αποκόμισα από την διεκπεραίωση της παρούσας εργασίας είναι καθοριστικής σημασίας για την μελλοντική μου ενασχόληση με το αντικείμενο της ιατρικής απεικόνισης και γενικότερα με το πεδίο της βιοϊατρικής.

# Περιεχόμενα


**Abstract** 3
**Περίληψη** 4
**1. Εισαγωγή** 5
**1.1 Ιατρική Απεικόνιση** 5
**1.2 Τομογραφική Απεικόνιση** 6
**1.2.1 Αρχές Τομογραφικής Απεικόνισης** 7
**1.3 Πυρηνική Ιατρική – SPECT** 10
**1.4 Στόχος Εργασίας** 14
**2. Τομογραφική ανακατασκευή σε SPECT** 15
**2.1 Οπισθοπροβολή/Backprojection** 15
**2.2 Φιλτραρισμένη Οπισθοπροβολή** 17
**2.3 Επαναληπτική ανακατασκευή και πίνακας συστήματος** 19
**2.3.1 Γεωμετρικός Πίνακας Συστήματος** 20
**2.4 Επαναληπτική ανακατασκευή με MLEM** 23
**2.5 Επαναληπτική ανακατασκευή με OSEM** 24
**3. Ανάπτυξη προγράμματος** 26
**3.1 Το εργαλείο Qt** 26
**3.1.1 Εισαγωγή στο Qt** 26
**3.1.2 Τα βασικότερα στοιχεία του Qt** 27
**3.1.3 Η λειτουργία Signals and Slots** 28
**3.1.4 Αντιμετώπιση των γεγονότων στο Qt** 29
**3.1.5 QtDesigner** 31
**3.2 Η εφαρμογή QSPECT** 32
**3.2.1 Δομή της εφαρμογής QSPECT** 34
**3.2.2 Οι λειτουργίες του προγράμματος** 39
**4. Παράδειγμα Εκτέλεσης** 54
**4.1 Εισαγωγή** 54
**4.1.1 Δημιουργία του πίνακα πιθανοτήτων του συστήματος** 55
**4.1.2 Δημιουργία της ανακατασκευής μιας τομής και εξαγωγή της είτε σε μορφή αρχείου κειμένου .txt είτε σε μορφή αρχείου εικόνας .bmp** 59
**4.1.3 Δημιουργία αρχείου εικόνας .bmp από αντίστοιχο αρχείο κειμένου .txt ανακατασκευασμένης τομής** 63
**4.1.4 Φόρτωση και προβολή εικόνων** 67
**5. Συμπεράσματα – Μελλοντικές Προοπτικές** 74
**5.1 Συμπεράσματα** 74
**5.2 Μελλοντικές Προοπτικές** 75
**Βιβλιογραφία** 77

# Abstract

The aim of the present project is the development of an Image Reconstruction application, which visualizes images from SPECT data. We reviewed nuclear medical imaging, methods of image reconstruction and particularly the SPECT imaging. We studied two of the image reconstruction algorithms for SPECT data, MLEM and OSEM, and we built a suitable GUI application. At the end, we made some performance tests of the final application.

For the needs of the project, we converted the original code of the image reconstruction algorithms, which were written in C, into C++, which is the language that the whole project's application is developed. In addition, we used the C++ toolkit Qt for GUI application development, which is the standard framework for developing high-performance cross-platform applications.

There were two image reconstruction methods developed, the MLEM algorithm and the OSEM algorithm. The first one (Maximum Likelihood Expectation Maximization) has precise results, but it needs sufficient time to complete the image reconstruction file. Whereas, the latter (Ordered Subsets Expectation Maximization) has the same base idea with the MLEM algorithm, but its convergence is much more faster.

Finally, the application’s functionality and performance was tested with clinic data, which were produced by SPECT/CT camera Varicam of GE Medical from the clinic Charite Verichow, Berlin.

# Περίληψη

Αντικείμενο της παρούσης εργασίας είναι η ανάπτυξη μιας εφαρμογής σε γραφικό περιβάλλον για την ανακατασκευή εικόνων τομών από δεδομένα τομογραφίας SPECT. Πραγματοποιήθηκε βιβλιογραφική επισκόπηση της ιατρικής απεικόνισης , των μεθόδων τομογραφικής ανακατασκευής εικόνας και ιδιαίτερα της πυρηνικής ιατρικής απεικόνισης. Δόθηκε έμφαση στους επαναληπτικούς αλγόριθμους MLEM και OSEM, για τους οποίους ήταν διαθέσιμη η υλοποίηση τους σε γλώσσα C. Μετατρέψαμε τα προγράμματα αυτών των αλγορίθμων ανακατασκευής σε C++ και υλοποιήσαμε τη σύνδεσή τους με το γραφικό περιβάλλον της εφαρμογής.

Για την ανάπτυξη της γραφικής διασύνδεσης του χρήστη με το πρόγραμμα χρησιμοποιήθηκε η πλατφόρμα ανάπτυξης γραφικού περιβάλλοντος σε γλώσσα προγραμματισμού C++, Qt. Το εργαλείο αυτό είναι ιδιαίτερα δημοφιλές στην ανάπτυξη διασύνδεσης χρήστη προγράμματος και χρησιμοποιείται ευρέως σε εφαρμογές ανοικτού αλλά και κλειστού κώδικα.

Επίσης υλοποιήσαμε κάποιες απλές λειτουργίες επεξεργασίας των εικόνων των τομών. Αυτές ήταν η δυνατότητα για zoom in, zoom out μιας εικόνας, η εισαγωγή χρωματικής απόχρωσης και η παραγωγή εικόνας δέκα χρωμάτων από μια μονόχρωμη εικόνα σύμφωνα με μια συγκεκριμένη παλέτα χρωμάτων. Το λογισμικό δοκιμάστηκε με κλινικά δεδομένα, τα οποία παρήχθησαν από SPECT/CT camera Varicam, του οίκου GE Medical από την κλινική Charite Verichow του Βερολίνου.

# ΚΕΦΑΛΑΙΟ 1

# ΕΙΣΑΓΩΓΗ

## 1.1 Ιατρική Απεικόνιση

Ο όρος ιατρική απεικόνιση περιλαμβάνει τις τεχνικές και τις διαδικασίες που χρησιμοποιούνται για να παραχθεί μια εικόνα του ανθρώπινου σώματος ή μέρος αυτού για κλινικούς σκοπούς (εξέταση, διάγνωση ή θεραπεία μιας ασθένειας) ή για επιστημονικούς λόγους (μελέτη της ανατομίας). Διάφορες μορφές ενέργειας χρησιμοποιούνται και μελετάται η αλληλεπίδραση αυτών με τους βιολογικούς ιστούς με στόχο την εξαγωγή κλινικά χρήσιμης πληροφορίας. Η πληροφορία αυτή καταγράφει σε τομογραφικούς χάρτες κάποια ιδιότητα των ιστών.

Στα αρχικά στάδια, για την απεικόνιση χρησιμοποιήθηκαν οι ακτίνες Χ που ανακαλύφθηκαν από τον (*Wilhelm Röntgen*) το 1895, έτσι η εφαρμογή τους στην Ιατρική ονομάστηκε Ακτινολογία, η οποία περιλαμβάνεται σε μια ευρύτερη θεώρηση του όρου ιατρική απεικόνιση. Με την πρόοδο της Ιατρικής και της Φυσικής, κυρίως τα τελευταία 30 χρόνια, ανακαλύφθηκαν και εφαρμόστηκαν και άλλες μέθοδοι απεικόνισης του σώματος. Με την παρουσίαση του Υπολογιστικού Αξονικού Τομογράφου ( Computed Axial Tomography – CAT ) στα 1972 από τον Hounsfield ξεκίνησε μια νέα εποχή για τη σύγχρονη ιατρική απεικόνιση. Μια πλειάδα απεικονιστικών συστημάτων, όπως η απεικόνιση με χρήση ραδιενεργών πυρήνων, οι υπέρηχοι και ο πυρηνικός μαγνητικός συντονισμός, έχουν εξελιχθεί και ενταχθεί στην καθημερινή χρήση της ιατρικής επιστήμης.

Σήμερα, η αλματώδης εξέλιξη στην ψηφιακή τεχνολογία και στην επιστήμη της πληροφορικής οδήγησε σε σημαντικές εξελίξεις στο χώρο της ιατρικής απεικόνισης. Η ταχύτατη απόκτηση, επεξεργασία και αποθήκευση μεγάλου όγκου

δεδομένων έδωσε μια νέα ώθηση και προσέφερε νέες δυνατότητες στις τεχνικές τομογραφικής απεικόνισης. Αυτό έχει οδηγήσει στην ανάπτυξη ενός τεράστιου πλήθους από ψηφιακές εφαρμογές με σκοπό την επίλυση ιατρικών προβλημάτων. Η ιατρική απεικόνιση έχει μεταβληθεί σε μια από τις σημαντικότερες μεθόδους οπτικοποίησης και εξαγωγής συμπερασμάτων στη βιολογία και την ιατρική. Ο σχεδιασμός, η ανάπτυξη και ο έλεγχος ολοκληρωμένων συστημάτων για κλινική χρήση απαιτεί τη στενή διεπιστημονική συνεργασία μεταξύ φυσικών, βιολόγων και μηχανικών.[1]

## 1.2 Τομογραφική Απεικόνιση

Η Τομογραφία είναι η απεικόνιση τομών ενός αντικειμένου. Τα δεδομένα, που λαμβάνονται, προέρχονται από τη διάδοση, εκπομπή ή την ανάκλαση ακτινοβολίας από το αντικείμενο προς εξέταση σε διαφορετικές διευθύνσεις και στη συνέχεια τροφοδοτούν έναν αλγόριθμο τομογραφικής ανακατασκευής εικόνας σε υπολογιστή. Η επίδραση αυτής της τεχνικής στη διαγνωστική ιατρική ήταν τεράστια, αφού έδωσε τη δυνατότητα σε γιατρούς και επιστήμονες να μελετούν εσωτερικά όργανα με ακρίβεια, που δεν ήταν εφικτή προηγουμένως, και επίσης με σχετική ασφάλεια για τους ασθενείς.

Στις πρώτες εφαρμογές της απεικονιστικής τομογραφίας χρησιμοποιούνταν ακτίνες Χ, ωστόσο σήμερα περισσότερα και διαφορετικά φυσικά φαινόμενα βρίσκουν εφαρμογή όπως ακτίνες γ, εκπομπές ποζιτρονίου, μαγνητικός συντονισμός (MRI), υπέρηχοι κ.ά.. Τα αντίστοιχα είδη τομογραφίας είναι CT, SPECT, PET, MRI, υπερηχογράφημα.

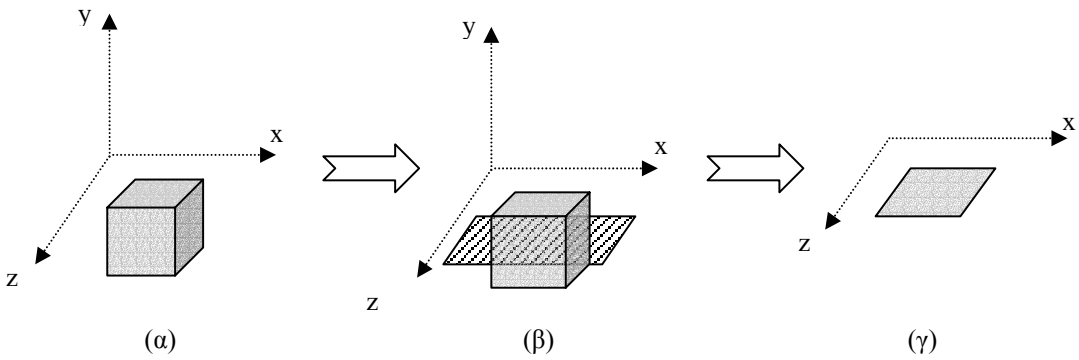
Η τομογραφική απεικόνιση σχετίζεται βασικά με την ανακατασκευή μιας εικόνας από τις προβολές της. Με την αυστηρή έννοια του όρου μια προβολή σε μια δοσμένη γωνία είναι το ολοκλήρωμα της εικόνας στην κατεύθυνση που ορίζει η γωνία αυτή. Μια προβολή εκφράζει την εξαγόμενη πληροφορία από την διερχόμενη ενέργεια όταν το αντικείμενο ακτινοβολείται υπό μια συγκεκριμένη γωνία.

Πολλοί διαφορετικοί αλγόριθμοι ανακατασκευής εικόνας υπάρχουν και οι περισσότεροι από αυτούς κατατάσσονται σε δυο κατηγορίες : φιλτραρισμένη οπισθοπροβολή (filtered back projection FBP) και επαναληπτικής ανακατασκευής

(iterative reconstruction IR). Οι δυο αυτές ομάδες αλγορίθμων ανακατασκευής διαφοροποιούνται όσον αφορά την ακρίβεια των αποτελεσμάτων και τον χρόνο υπολογισμού που χρειάζονται. Οι αλγόριθμοι φιλτραρισμένης οπισθοπροβολής έχουν μικρότερες υπολογιστικές απαιτήσεις, ενώ οι επαναληπτικοί αλγόριθμοι παράγουν γενικώς λιγότερα λάθη κατά την ανακατασκευή αλλά με μεγαλύτερο κόστος σε πόρους και χρόνο. Οι δυο αυτοί τύποι αλγορίθμων θα εξεταστούν εκτενέστερα στο επόμενο κεφάλαιο. [1, 2]

### 1.2.1 Αρχές Τομογραφικής Απεικόνισης

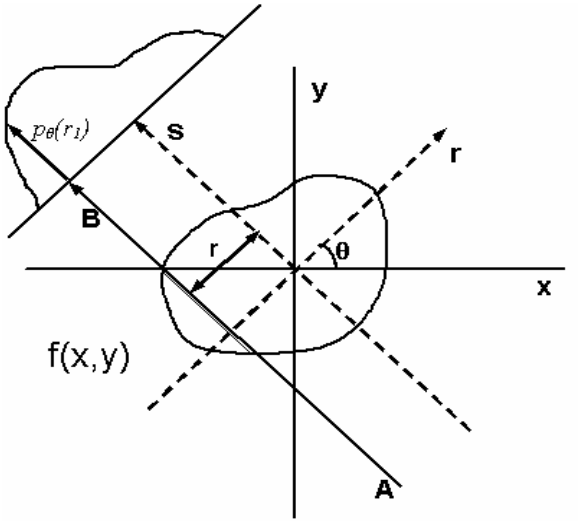
Αν έχουμε ένα αντικείμενο (π.χ. κύβος) σε ένα σύστημα συντεταγμένων (x, y, z) και θέλουμε να απεικονίσουμε μια τομή του σε ένα επίπεδο κάθετο σε αυτόν, η διαδικασία τομογραφικής απεικόνισης συνίσταται στην περιστροφή ενός ανιχνευτή γύρω από το αντικείμενο και στη συλλογή προβολών, από ένα αριθμό γωνιών. Με χρήση καταλλήλων αλγορίθμων ανακατασκευάζονται τομές σε επίπεδα, που είναι κάθετα στον άξονα περιστροφής.




Σχήμα 1.1: (α) Κύβος σε ορθοκανονικό σύστημα συντεταγμένων. (β) Τομογραφικό επίπεδο. (γ) Τομή.

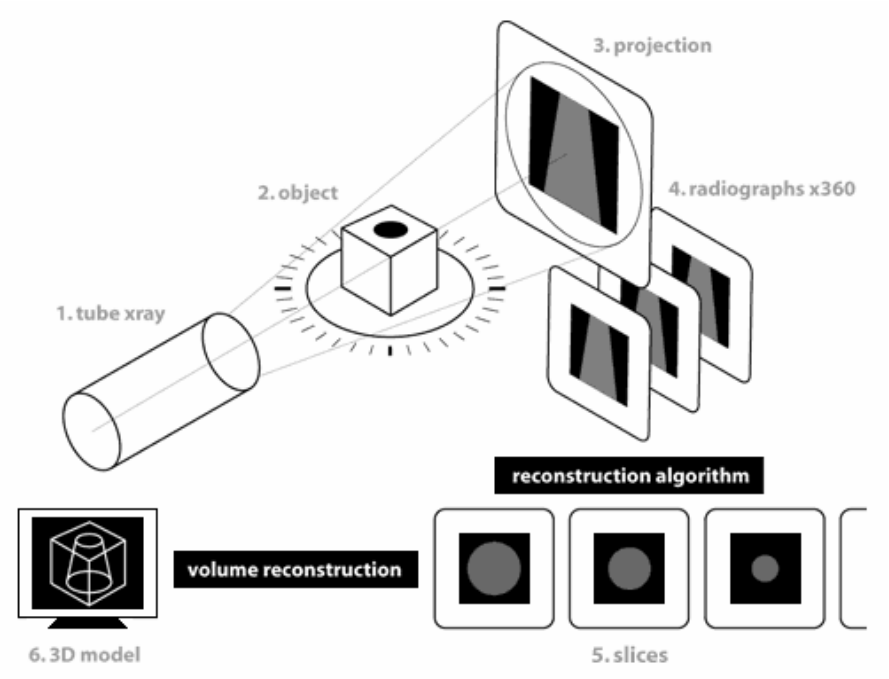
Η προβολή ενός αντικειμένου σε μια δοσμένη γωνία, θ, κατασκευάζεται από ένα σύνολο από γραμμικά ολοκληρώματα. Στην αξονική τομογραφία (CT) το γραμμικό ολοκλήρωμα αντιπροσωπεύει τη συνολική εξασθένηση της δέσμης ακτίνων X, καθώς αυτή διέρχεται μέσα από το αντικείμενο, ενώ στην τομογραφία SPECT την εκπεμπόμενη ακτινοβολία, που είναι ανάλογη της συγκέντρωσης ενός ραδιοφαρμάκου σε κάποιο ιστό ή σημείο του σώματος

Ο ευκολότερος και απλούστερος τρόπος οπτικοποίησης μιας περιοχής είναι το σύστημα παράλληλων προβολών, που είχε χρησιμοποιηθεί στους πρώτους τομογράφους. Εδώ, θεωρούμε ότι τα δεδομένα συλλέγονται από σειρές παράλληλων ακτίνων στη θέση r και κατά μήκος μιας προβολής σε γωνία θ, το οποίο επαναλαμβάνεται για διαφορετικές γωνίες.




Σχήμα 1.2 : Γεωμετρία παράλληλης δέσμης. Κάθε προβολή δημιουργείται από ένα σύνολο από ολοκληρώματα κατά μήκος του αντικειμένου.

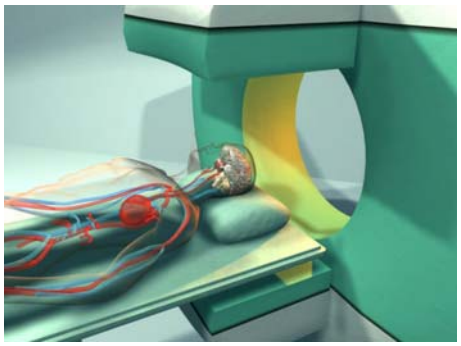
Θεωρούμε ένα δισδιάστατο αντικείμενο που παριστάνεται από τη συνάρτηση f(x,y) της χωρικής κατανομής μιας φυσικής ποσότητας (στην τομογραφία SPECT αυτή η φυσική ποσότητα είναι ο αριθμός των εκπεμπόμενων φωτονίων) και εργαζόμαστε στο ορθογώνιο σύστημα συντεταγμένων (t,s) που έχει στραφεί ως προς το σύστημα (x,y) κατά γωνία φ. Το επικαμπύλιο ολοκλήρωμα της f(x,y) κατά μήκος μιας γραμμής L παράλληλης ως προς τον άξονα s, η οποία ονομάζεται τομογραφική ακτίνα, αποτελεί τα δεδομένα προβολής (projection data) και ισούται με: $P(\varphi,t)=\int f(x,y)ds$.




Σχήμα 1.3: Προβολές από διαφορετικές γωνίες συλλέγονται έτσι ώστε να δημιουργηθούν οι εικόνες των τομών και ίσως μια τρισδιάστατη απεικόνιση.

Μια ομάδα δεδομένων προβολής κατά μήκος παράλληλων γραμμών δημιουργεί μια "όψη", η οποία είναι ένα μονοδιάστατο προφίλ της μετρούμενης ποσότητας σαν συνάρτηση της θέσης και αντιστοιχεί σε μια δεδομένη γωνία φ. Η συλλογή πολλών διαφορετικών όψεων μπορεί να παρασταθεί σ' ένα δισδιάστατο διάγραμμα ή σαν εικόνα, όπου ο ένας άξονας είναι η θέση t και ο άλλος η γωνία φ. Η εικόνα αυτή ονομάζεται ημιτονόγραμμα (Sinogram) ή μετασχηματισμός Radon του δισδιάστατου αντικειμένου (Radon Transform). Σκοπός των αλγορίθμων ανακατασκευής εικόνας είναι η επίλυση του αντίστροφου μετασχηματισμού Radon προκειμένου να προσδιοριστεί η εκτίμηση της προς ανακατασκευή εικόνας **f**(x,y) βάσει των δεδομένων προβολής.[1-3]

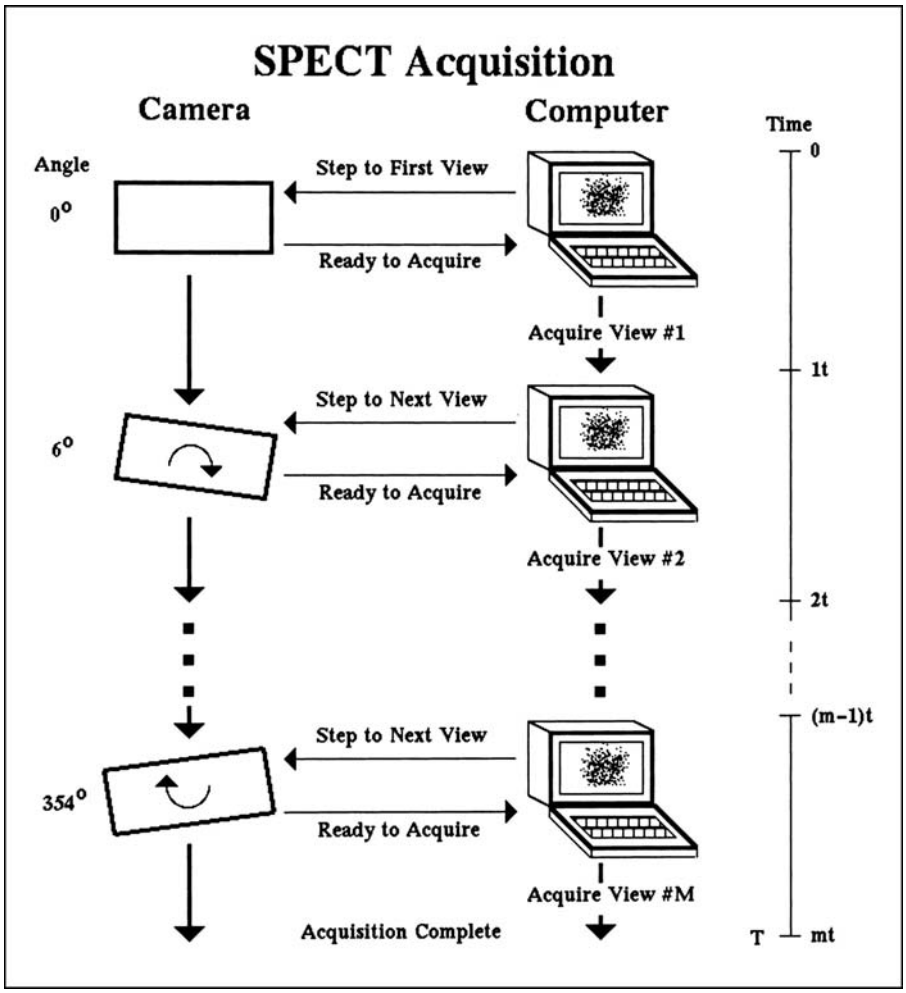
## 1.3 Πυρηνική Ιατρική – SPECT



Η τεχνική της μονοφωτονιακής τομογραφίας εκπομπής (Single Photon Emission Computerised Tomography-SPECT) είναι μια μέθοδος ιατρικής απεικόνισης που συνδυάζει τις συμβατικές τεχνικές απεικόνισης της πυρηνικής ιατρικής με τις μεθόδους ανακατασκευής της εικόνας. Στην τομογραφία SPECT χρησιμοποιούνται ακτίνες γ και μπορούμε να απεικονίσουμε λειτουργικές πληροφορίες για κάποιο συγκεκριμένο όργανο του ανθρώπινου σώματος. Η συνεχώς αυξανόμενη ποιότητα και ποσοτική ακρίβεια των εικόνων SPECT σε συνδυασμό με το χαμηλό κόστος τους, καθιστούν την τομογραφία SPECT ως ένα από τα χρησιμότερα διαγνωστικά εργαλεία.

Η διαδικασία της απεικόνισης SPECT αρχίζει με τη χορήγηση ενός ραδιοφαρμάκου στον εξεταζόμενο. Το ραδιοφάρμακο συγκεντρώνεται επιλεκτικά σε ορισμένους ιστούς ή όργανα ανάλογα με τις βιοκινητικές του ιδιότητες και την παθοφυσιολογία του ασθενούς. Τα χρησιμοποιούμενα ραδιονουκλίδια εκπέμπουν ακτίνες-γ χαμηλών σχετικά ενεργειών όπως τα φωτόνια 140keV του $^{99m}$Tc και τα φωτόνια 70keV του $^{201}$Tl. Οι εκπεμπόμενες ακτίνες-γ διαπερνούν τους ιστούς και καταγράφονται από μια ανιχνευτική διάταξη σε διάφορες γωνίες γύρω από τον ασθενή.

Μια ή περισσότερες γ-κάμερα περιστρέφονται γύρω από τον ασθενή με σκοπό να παραχθούν οι SPECT εικόνες. Συλλέγονται προβολές της κατανομής του ραδιοφαρμάκου στην υπό μελέτη περιοχή κατά τη διάρκεια της περιστροφής, συνήθως κάθε 3-6 μοίρες. Στις περισσότερες περιπτώσεις μια ολόκληρη περιστροφή 360 μοιρών χρησιμοποιείται για να επιτευχθεί η ιδανική ανακατασκευή. Ο χρόνος που απαιτείται για κάθε προβολή είναι επίσης μεταβλητός, αλλά 15-20 δευτερόλεπτα είναι μια τυπική τιμή, το οποίο σημαίνει συνολικό χρόνο για την συλλογή όλων των προβολών 15-20 λεπτά.

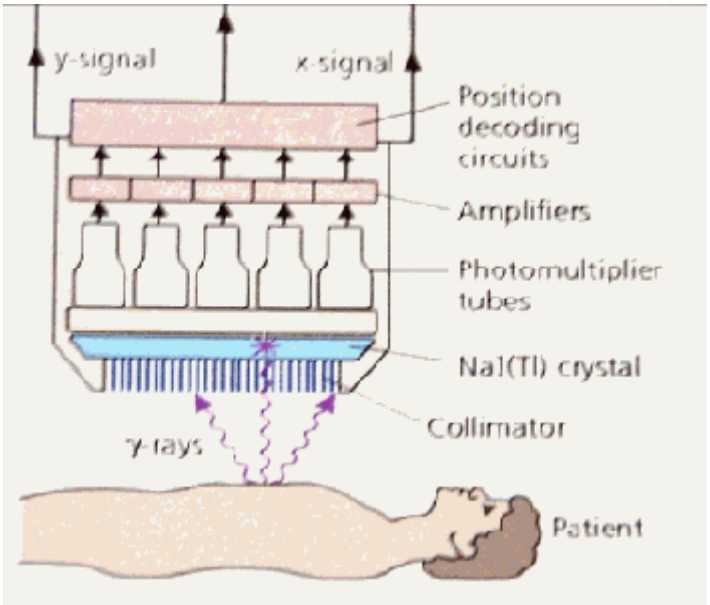



Σχήμα 1.4: Συλλογή δεδομένων SPECT μέσω μιας γ-κάμερας συνδεδεμένης με υπολογιστή.[4]

Το απεικονιζόμενο όργανο διαχωρίζεται με τη βοήθεια του υπολογιστή σε ξεχωριστές επιφάνειες (slices) και ανιχνεύονται τα φωτόνια τα προερχόμενα από κάθε επιφάνεια. Κάθε ανιχνευτής, συνδεδεμένος με κατευθυντήρα παράλληλων οπών, δίνει ένα σύνολο δισδιάστατων προβολών χρησιμοποιώντας μονοφωτονιακές τεχνικές μέτρησης (single photon counting). Αυτές οι προβολές, συνδυαζόμενες κατάλληλα στον υπολογιστή με την εφαρμογή ενός αλγόριθμου ανακατασκευής, επιτρέπουν την κατασκευή εγκάρσιας εικόνας η οποία αναπαριστά την κατανομή του ραδιοφαρμάκου στο υπό μελέτη όργανο ή δομή. Μ' αυτό τον τρόπο μπορούμε να μελετήσουμε τη μορφολογία και τη λειτουργικότητα του υπό μελέτη οργάνου ή δομής.

Στην κλινική πυρηνική ιατρική χρησιμοποιούνται ανιχνευτικές διατάξεις γ-camera τύπου Anger (Anger Scintillation Camera). Αποτελούνται από ένα μόνο κρύσταλλο NaI(Tl), αρκετά μεγάλης διαμέτρου, ώστε να είναι δυνατή η απεικόνιση

ενός μεγάλου τμήματος του ανθρώπινου σώματος και μια διάταξη φωτοπολλαπλασιαστών. Η ακτινοβολία η οποία περνά από τον κατευθυντήρα συναντά τον κρύσταλλο του σπινθηριστή, όπου έχουμε απορρόφηση φωτονίων και παραγωγή σπινθηρισμών στο σημείο απορρόφησης. Η ένταση των σπινθηρισμών είναι ανάλογη των ακτίνων-γ. Η φωτοκάθοδος των φωτοπολλαπλασιαστών, η οποία βρίσκεται πίσω από τον κρύσταλλο μαζί με το λογικό κύκλωμα θέσης, καθορίζουν τη θέση κάθε σπινθηρισμού που παράγεται στον κρύσταλλο, δημιουργώντας ένα αποτύπωμα των σπινθηρισμών το οποίο περιγράφει την κατανομή της ραδιενέργειας του υπό μελέτη δείγματος. Κάθε φωτοπολλαπλασιαστής παράγει ηλεκτρικό παλμό του οποίου το εύρος είναι ανάλογο προς την ένταση του φωτός που διεγείρει την φωτοκάθοδο. Ο αναλυτής ύψους παλμών (pulse-height analyser) καθορίζει τα γεγονότα που θα καταγραφούν. Με βάση την ενέργεια του παλμού (αν είναι εντός του προεπιλεγμένου "ενεργειακού παραθύρου" της συσκευής), επιτρέπεται η είσοδος του από το διευκρινιστή και μεταφέρεται σε μήτρα κωδικοποίησης. Η γ-camera μπορεί να χρησιμοποιηθεί για μελέτη είτε στατικής εικόνας, στην οποία η εικόνα μιας σταθερής κατανομής των ραδιοϊσοτόπων καταγράφεται για ένα παρατεταμένο χρόνο (π.χ. λεπτά), είτε δυναμικής εικόνας, στην οποία παρατηρούνται αλλαγές στην κατανομή των ραδιονουκλιδίων τόσο γρήγορα όσο αρκετές εικόνες ανά δευτερόλεπτο.



Σχήμα 1.5: Βασικές αρχές και τμήματα της Anger camera.

Τα συστήματα SPECT αποτελούνται από μια ή περισσότερες ομάδες ανιχνευτών, διατεταγμένους σύμφωνα με κάποια ειδική γεωμετρία και οι οποίοι κινούνται με κάποια μηχανισμό ώστε να λαμβάνονται δεδομένα από διαφορετικές γωνίες. Τα συστήματα SPECT μπορούν χωρισθούν στις ακόλουθες κατηγορίες.

- Συστήματα πολλών ανιχνευτών . Οι διατάξεις των ανιχνευτών μπορεί να είναι σε σχήμα τετραγώνου, κύκλου ή οι ανιχνευτές μπορούν να σαρώνουν τόσο ακτινικά όσο και εφαπτομενικά.
- Συστήματα βασισμένα στην Anger Camera. Αποτελούνται από μια ή περισσότερες Anger Cameras που μπορούν να περιστρέφονται γύρω από τον ασθενή και οι διατάξεις τους ποικίλουν. Ορισμένα παραδείγματα διατάξεων είναι τα εξής : μια κάμερα, δυο κάμερες τοποθετημένες αντιδιαμετρικά κατά τη διάρκεια της περιστροφής, δυο κάμερες που σχηματίζουν ορθή γωνία, τρεις κάμερες σε τρίγωνο και τέσσερις κάμερες σε τετράγωνο.
- Υβριδικά συστήματα. Είναι συστήματα που συνδυάζουν τις δυο προηγούμενες κατηγορίες. Παράδειγμα : Το CERESPECT brain SYSTEM που αποτελείται από ένα σταθερό κυκλικό κρύσταλλο NaI(TI) και έναν περιστρεφόμενο κυκλικό ευθυγραμμιστή.

Τα περισσότερα συστήματα απεικόνισης ακτίνων-γ που χρησιμοποιούνται στην πράξη βασίζονται στην αρχή της απορροφητικής κατευθυντικότητας (absorptive collimation) για την παραγωγή εικόνας. Ο κατευθυντήρας που χρησιμοποιείται, καθορίζει την περιοχή από όπου φωτόνια μπορούν να εισέλθουν στον ανιχνευτή σπινθηρισμού και είναι εκείνα που ταξιδεύουν κατά μήκος μόνο συγκεκριμένων διευθύνσεων, ενώ φωτόνια που δεν ταξιδεύουν στη σωστή κατεύθυνση απορροφώνται από τον κατευθυντήρα. Η χρήση του κατευθυντήρα συντελεί, επομένως, σε επιλογή των φωτονίων που ανιχνεύονται από τη γ-camera και ως εκ τούτου ρυθμίζει την ένταση της διερχόμενης ακτινοβολίας. Επίσης καθορίζει το γεωμετρικό οπτικό πεδίο και προσδιορίζει τη διακριτική ικανότητα και ευαισθησία του συστήματος.

Υπάρχουν τέσσερα βασικά είδη κατευθυντήρων τα οποία χρησιμοποιούνται στη γ-camera: α) Κατευθυντήρας τύπου καρφίδος (pinhole), β) Κατευθυντήρας με συστήματα παράλληλων οπών (parallel hole collimator), γ) Κατευθυντήρας με αποκλίνουσες οπές (diverging collimator) και δ) Κατευθυντήρας με συγκλίνουσες οπές (converging collimator).[1, 4-7]

## 1.4 Στόχος εργασίας

Στόχος της εργασίας ήταν η ανάπτυξη ενός γραφικού εργαλείου ανακατασκευής εικόνων τομών από δεδομένα τομογραφίας SPECT. Για το σκοπό αυτό ήταν απαραίτητη η εξοικείωση με το εργαλείο Qt, το οποίο είναι κατάλληλο για την ανάπτυξη γραφικού περιβάλλοντος με τη γλώσσα C++. Το Qt είναι cross-platform, δηλαδή λειτουργεί και είναι δυνατή η ανάπτυξη εφαρμογών σε διαφορετικές πλατφόρμες και διαφορετικά λειτουργικά συστήματα (Windows, Linux, Mac OS). Αρχικά, ήταν αναγκαία η απόκτηση εμπειρίας μέσα από τη μελέτη βιβλιογραφίας για το εργαλείο αυτό και τη συγγραφή μικρών προγραμμάτων.

Στη συνέχεια πραγματοποιήθηκε βιβλιογραφική επισκόπηση της πυρηνικής ιατρικής απεικόνισης και των μεθόδων τομογραφικής ανακατασκευής. Επικεντρωθήκαμε στους επαναληπτικούς αλγορίθμους ανακατασκευής MLEM και OSEM. Τα προγράμματα για την κατασκευή του πίνακα πιθανοτήτων συστήματος καθώς και για τους παραπάνω αλγορίθμους ανακατασκευής ήταν διαθέσιμα σε γλώσσα C. Ένας από τους στόχους της εργασίας ήταν η μετατροπή αυτών των προγραμμάτων σε γλώσσα C++ και η σύνδεσή τους με το γραφικό περιβάλλον της εφαρμογής.

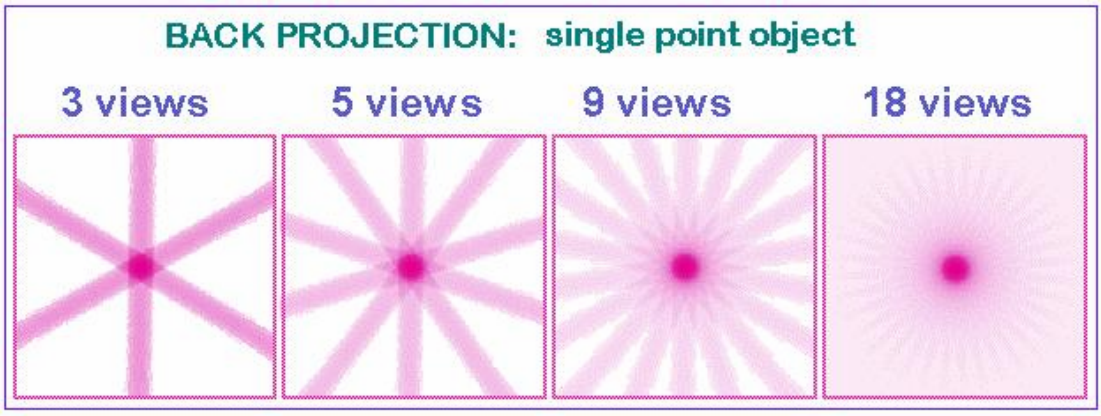
Τέλος, εισήχθησαν κάποιες βασικές λειτουργίες επεξεργασίας των εικόνων των τομών, όπως δυνατότητα για zoom in, zoom out, χρωματική απόχρωση και παραγωγή εικόνας δέκα χρωμάτων από μια μονόχρωμη εικόνα σύμφωνα με μια συγκεκριμένη παλέτα χρωμάτων. Το λογισμικό δοκιμάστηκε με κλινικά δεδομένα, τα οποία παρήχθησαν από SPECT/CT camera Varicam, του οίκου GE Medical από την κλινική Charite Verichow του Βερολίνου.

# ΚΕΦΑΛΑΙΟ 2

# Τομογραφική ανακατασκευή σε SPECT

## 2.1 Οπισθοπροβολή

Για την ανακατασκευή μίας τομογραφικής εικόνας από τις συλλεγόμενες προβολές, ο πιο απλός τρόπος είναι η οπισθοπροβολή. Δημιουργούμε μια εικόνα οπισθοπροβάλοντας μια προβολή σε κάθε γωνία σε όλα τα pixel στην κατεύθυνση της προβολής. Η διαδικασία αυτή φαίνεται στο σχήμα 2.1. Διαδοχικά οπισθοπροβάλλονται οι προβολές σε 3,5,9 και 18 γωνίες. Η εικόνα που σχηματίζεται από την άθροιση όλων των οπισθοπροβολών, προσεγγίζει σταδιακά το αντικείμενο.

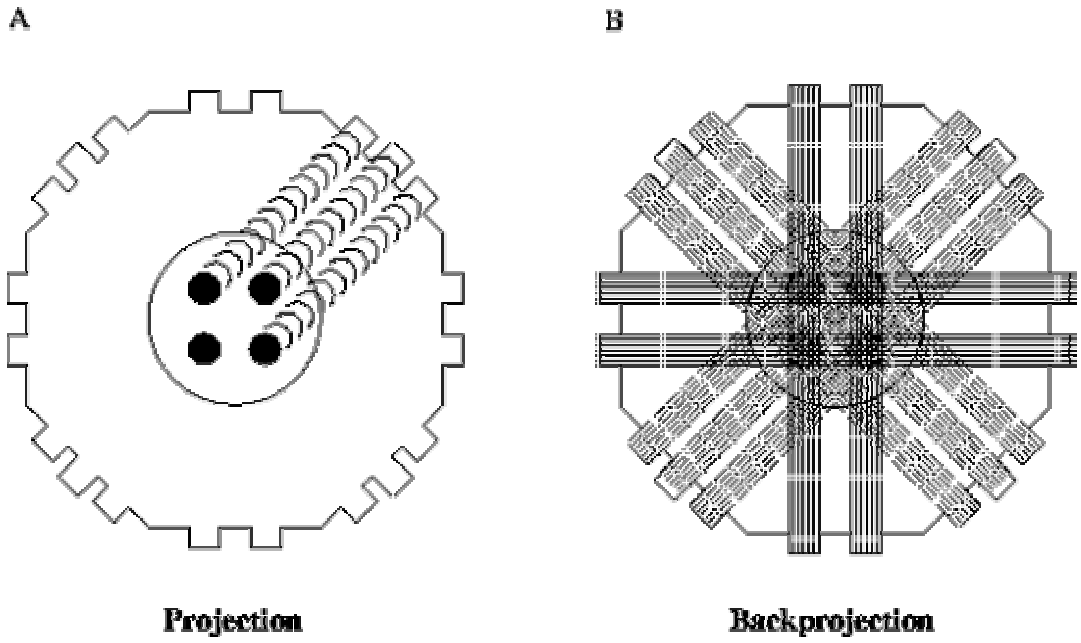



Σχήμα 2.1: Διαδικασία οπισθοπροβολής. Οπισθοπροβολή 3,5,9 και 18 προβολών.

Αυξάνοντας τις προβολές η ποιότητα της ανακατασκευής βελτιώνεται. Το μειονέκτημα της διαδικασίας αυτής είναι πως εκτός από το πραγματικό αντικείμενο στην ανακατασκευασμένη εικόνα υπάρχει υψηλός θόρυβος, ο οποίος οφείλεται στις οπισθοπροβολές και έχει την μορφή ακτινικών γραμμών (star artifacts). Επίσης, οι αιχμές της εικόνας δεν είναι σαφώς καθορισμένες και σε περιοχές ομοιομορφίας της εικόνας, η ανακατασκευασμένη εικόνα παρουσιάζει μη πραγματική ενίσχυση προς το κέντρο. Όσο πολυπλοκότερο είναι το προς ανακατασκευή αντικείμενο, τόσο

πολυπλοκότερες είναι οι προβολές και κατά συνέπεια εντονότερος ο θόρυβος, με αποτέλεσμα η ανακατασκευή να είναι σε πολλές περιπτώσεις ανεπιτυχής.

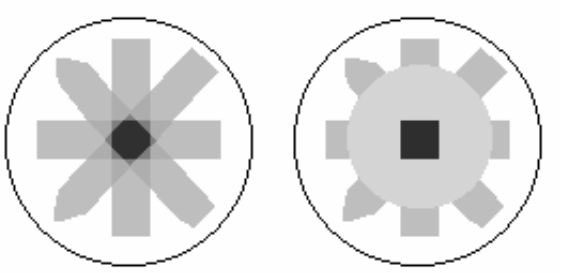
Θεωρητικά η οπισθοπροβολή (ο αντίστροφος μετασχηματισμός Radon) δίνει τέλειες ανακατασκευές, όταν ένας άπειρος αριθμός προβολών είναι γνωστός. Στην πράξη αυτό δεν είναι ποτέ δυνατό αφενός γιατί οι ανιχνευτικές διατάξεις έχουν ένα πεπερασμένο βήμα δειγματοληψίας και αφετέρου –στην ιατρική απεικόνιση- επειδή ο χρόνος ανίχνευσης/εξέτασης πρέπει να είναι όσο το δυνατό συντομότερος. Για το λόγο αυτό έχουν αναπτυχθεί αλγόριθμοι προεπεξεργασίας και φιλτραρίσματος των προβολών, ώστε η οπισθοπροβολή να μην συνοδεύεται από έντονο θόρυβο. Ο πιο σημαντικός είναι ο αλγόριθμος της φιλτραρισμένης οπισθοπροβολής (Filtered Backprojection – FBP). [1]




Σχήμα 2.2: Προβολές και οπισθοπροβολές σε διάφορες γωνίες τεσσάρων κύκλων.

## 2.2 Φιλτραρισμένη οπισθοπροβολή

Ο αλγόριθμος FBP χρησιμοποιείται σε όλα τα κλινικά συστήματα τομογραφίας και το σημαντικότατο πλεονέκτημά του είναι η εύκολη υλοποίηση του σε υπολογιστή, με τρόπο ώστε η ανακατασκευή των τομογραφικών εικόνων να γίνεται σε πραγματικό χρόνο. Αποτελείται από δύο βήματα: 1) Εφαρμογή ενός κατάλληλου φίλτρου h(t) στα δεδομένα προβολής και 2) Οπισθοπροβολή των διαμορφωμένων δεδομένων προβολής για τη δημιουργία της εικόνας. Το πρώτο βήμα υλοποιείται σε τρία στάδια: α) Μετασχηματισμός Fourier των δεδομένων προβολής στο χώρο των συχνοτήτων ω με τη χρήση μεθόδου FFT (Fast Fourier Transform), β) Πολλαπλασιασμός των κατά Fourier μετασχηματισμένων δεδομένων προβολής με μια κατάλληλη συνάρτηση H(ω), που είναι ο μετασχηματισμός Fourier της συνάρτησης φίλτρου h(t) και γ) Αντίστροφος μετασχηματισμός Fourier στο χώρο.



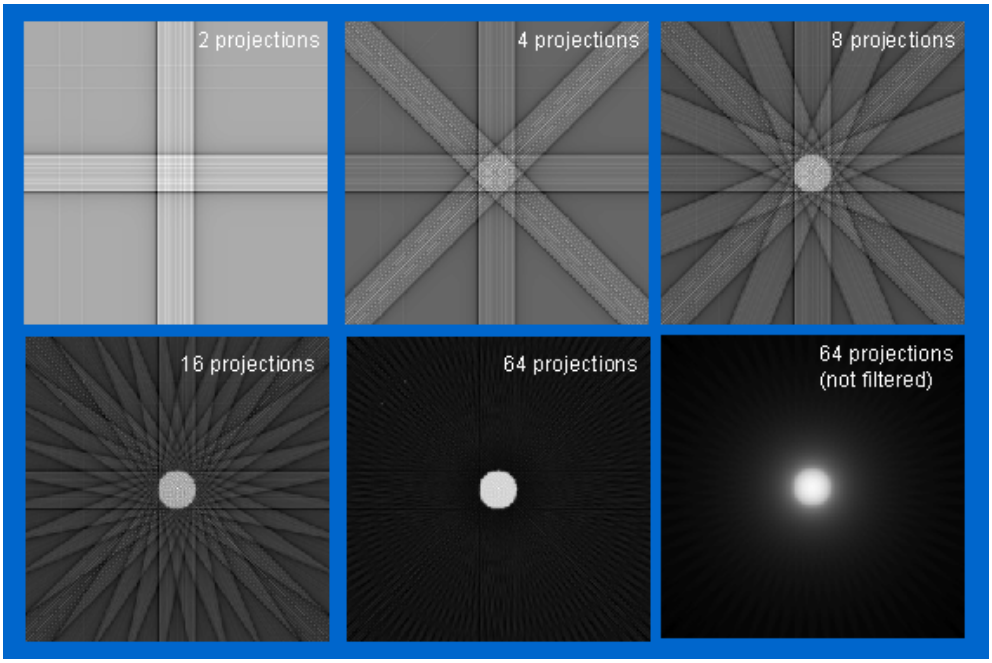
(α) (β)

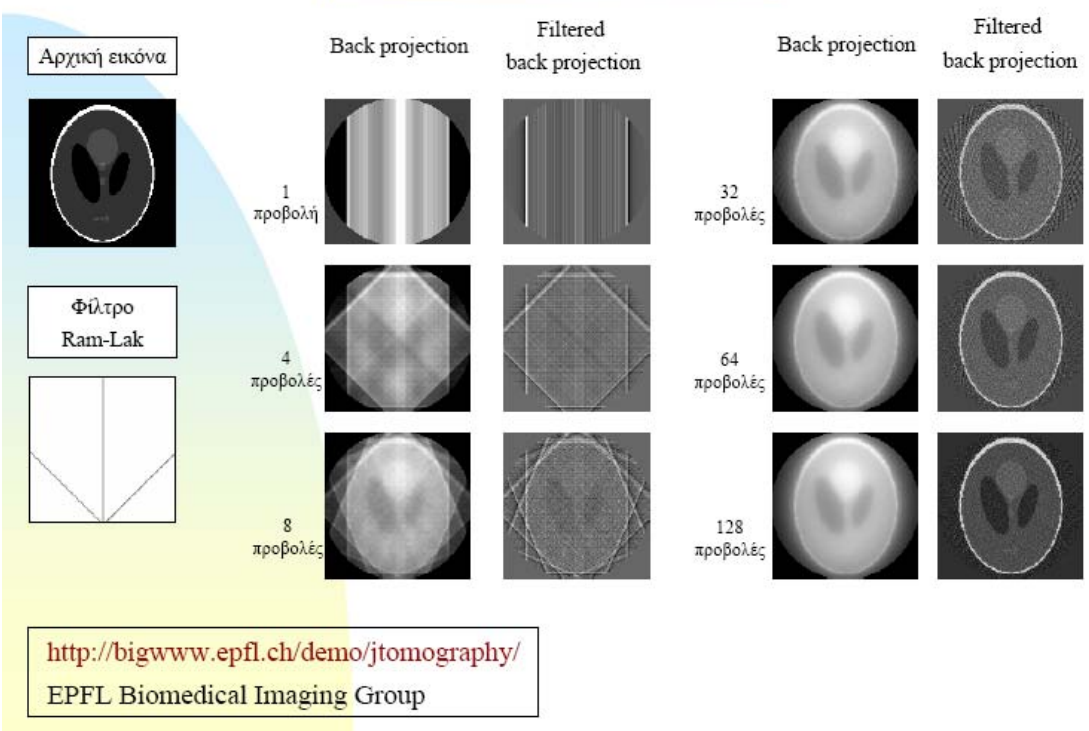
Σχήμα 2.3: Ανακατασκευή με οπισθοπροβολή: (α) χωρίς φιλτράρισμα και (β) με φιλτράρισμα των προβολών.

Η συνάρτηση H(ω) που χρησιμοποιείται είναι η απόλυτη τιμή της συχνότητας |ω|. Τα φιλτραρισμένα δεδομένα προβολής οπισθοπροβάλονται και προκύπτει η ανακατασκευασμένη εικόνα. Ο αλγόριθμος FBP δίνει τα ίδια αποτελέσματα με τη μέθοδο του αντίστροφου μετασχηματισμού Fourier, αλλά είναι πιο εύκολος να εφαρμοστεί. Δύο είναι τα βασικά πλεονεκτήματα του αλγόριθμου FBP, σε σχέση με την παρεμβολή στο χώρο της συχνότητας:

- Η ανακατασκευή μπορεί να αρχίσει από τη στιγμή, που λαμβάνεται η πρώτη προβολή. Αυτό έχει ως αποτέλεσμα να επιταχύνεται η διαδικασία και να μειώνεται ο όγκος της πληροφορίας, που αποθηκεύεται κάθε φορά.
- Είναι ευκολότερο να γίνεται παρεμβολή στον πραγματικό χώρο, παρά στο χώρο της συχνότητας. Παρεμβολή είναι αναγκαία και στην περίπτωση του αλγορίθμου

FBP. Στην περίπτωση αυτή η γραμμική παρεμβολή, παρέχει ικανοποιητικά αποτελέσματα, ενώ στην περίπτωση του μετασχηματισμού Fourier στο χώρο της συχνότητας, απαιτούνται πολυπλοκότερες μορφές παρεμβολής.[1, 8]




Σχήμα 2.4 και Σχήμα 2.5 : Σύγκριση των αποτελεσμάτων με φιλτραρισμένη και απλή οπισθοπροβολή για περισσότερες προβολές.[9]

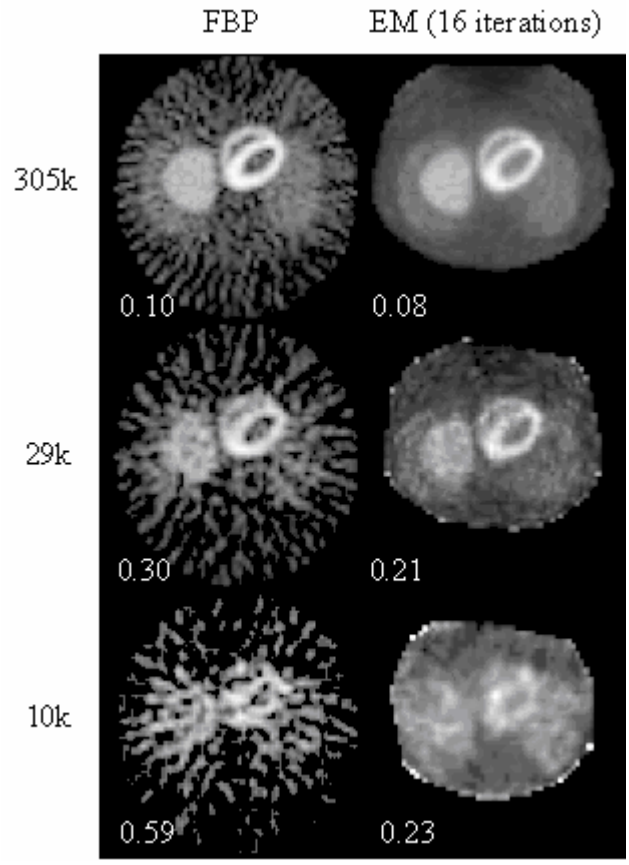
## 2.3 Επαναληπτική ανακατασκευή και πίνακας συστήματος

Η τεχνική των επαναληπτικών αλγορίθμων διαφέρει σε μεγάλο βαθμό από την τεχνική της οπισθοπροβολής (filtered back projection FBP). Οι βασικές αρχές των επαναληπτικών αλγορίθμων ανακατασκευής μπορούν να συνοψιστούν στα παρακάτω. Η χωρική συνάρτηση κατανομής της φυσικής ποσότητας f(x,y) από ένα πλέγμα αποτελούμενο από τετράγωνες περιοχές (pixels) σε κάθε μία από τις οποίες η τιμή της φωτεινότητας θεωρείται σταθερή. Η τομογραφική ακτίνα αντιστοιχεί σε μία παράλληλη ζώνη εύρους $\Delta r$ . Η τιμή κάθε μέτρησης προβολής καθορίζεται από το εμβαδόν και την φωτεινότητα των pixels που τέμνει η τομογραφική ακτίνα. Γίνεται μια αρχική εκτίμηση της συνάρτησης $f(x,y)$, υπολογίζονται οι προβολές και συγκρίνονται με τις υπάρχουσες μετρήσεις στις αντίστοιχες γωνίες. Εφαρμόζοντας ένα αλγόριθμο βασισμένο σε ειδικά στατιστικά κριτήρια οι διαφορές χρησιμοποιούνται για να διορθώσουν την αρχική εκτίμηση της εικόνας. Η διαδικασία επαναλαμβάνεται μέχρι η διαφορά της εκτίμησης και των μετρήσεων να γίνει αρκετά μικρή (μικρότερη μιας προκαθορισμένης μικρής τιμής). Τα στατιστικά κριτήρια που χρησιμοποιούνται περιλαμβάνουν προσεγγίσεις Minimum Mean Squares Error (MMSE), Weighted Least Squares (WLS), Maximum Entropy (ME), Maximum Likelihood (ML) και Maximum A Posteriori.

Τα μειονεκτήματα αυτών των αλγορίθμων είναι το αυξημένο κόστος σε υπολογιστικό χρόνο και η μεγάλη χρονική διάρκεια της επεξεργασίας. Επίσης, Η ύπαρξη θορύβου στις μετρήσεις ενός υπερκαθορισμένου συστήματος δεν επιτρέπει την σύγκλιση της μεθόδου. Ωστόσο, μπορεί να χρησιμοποιηθεί εκ των προτέρων γνώση σχετικά με την εικόνα και ειδικά για την φωτεινότητα και την διάταξη των pixels με σκοπό να αντισταθμιστούν τα φαινόμενα υποβάθμισης της εικόνας, μοντελοποιώντας τη διαδικασία απεικόνισης συμπεριλαμβανομένων και των παραγόντων υποβάθμισης. Έτσι για παράδειγμα, μπορούν να παράγουν αποδεκτές εικόνες από περιορισμένο αριθμό όψεων, σε περιπτώσεις όπου η συλλογή δεδομένων σε ορισμένες γωνίες δεν είναι εφικτή λόγω φυσικών περιορισμών.

Το βασικότερο πλεονέκτημα των επαναληπτικών αλγορίθμων είναι ότι επιτρέπουν να μοντελοποιηθούν πιστά οι διαδικασίες εκπομπής και ανίχνευσης. Σε

αντίθεση, στον αλγόριθμο οπισθοπροβολής δεν είναι δυνατή η μοντελοποίηση της εξασθένησης και του σκεδασμού των εκπεμπόμενων φωτονίων. Επίσης, η ποσότητα του θορύβου που περιέχει η τελική ανακατασκευή μιας εικόνας με επαναληπτικό αλγόριθμο είναι πολύ πιο αποδεκτή από αυτήν που έχει χρησιμοποιηθεί ο αλγόριθμος οπισθοπροβολής. Τέλος, η δυνατότητα ενσωμάτωσης πρόσθετης πληροφορίας, όπως η ενσωμάτωση ανατομικής πληροφορίας στην μαγνητική τομογραφία (MRI), στην ανακατασκευή με επαναληπτικούς αλγορίθμους είναι ένα ακόμη ισχυρό πλεονέκτημα των επαναληπτικών αλγορίθμων. [8, 10, 11]

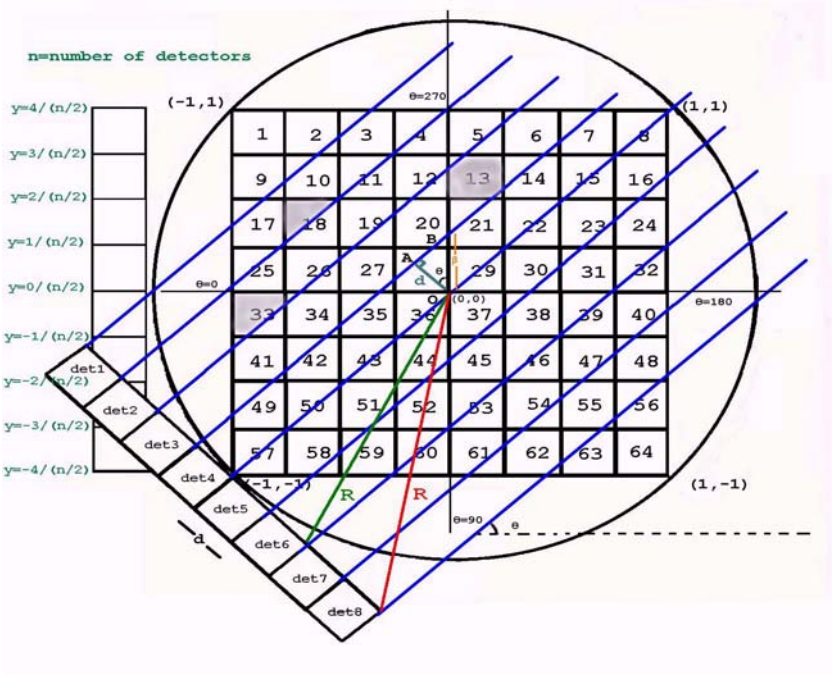



Σχήμα 2.6 : Σύγκριση ανακατασκευής με οπισθοπορβολή (FBP) και με επαναληπτικό αλγόριθμο (EM) για διαφορετικές συνολικές μετρήσεις. Ο θόρυβος για μικρό αριθμό μετρήσεων στον αλγόριθμο FBP είναι εμφανής.

### 2.3.1 Γεωμετρικός πίνακας συστήματος (SYSTEM MATRIX)

Η υλοποίηση όλων των επαναληπτικών αλγόριθμων στηρίζεται στη γνώση του γεωμετρικού πίνακα του συστήματος P. Ο υπολογισμός του πίνακα γίνεται γεωμετρικά. Για κάθε ανιχνευτή, γωνία περιστροφής και pixel εικόνας βρίσκονται τα σημεία τομής του οπτικού πεδίου κάθε ανιχνευτή με κάθε pixel, το είδος του

σχήματος που προκύπτει και τελικά υπολογίζεται το εμβαδόν του. Στις περισσότερες περιπτώσεις δεν υπάρχει τομή και το εμβαδόν αυτό είναι μηδενικό. Στις υπόλοιπες περιπτώσεις το προκύπτον σχήμα μπορεί να είναι τρίγωνο, τετράγωνο, τραπέζιο, πεντάγωνο ή εξάγωνο, όπως μπορεί να παρατηρηθεί και στο σχήμα 3.2, όπου έχει υποτεθεί μια περίπτωση εικόνας 8x8.

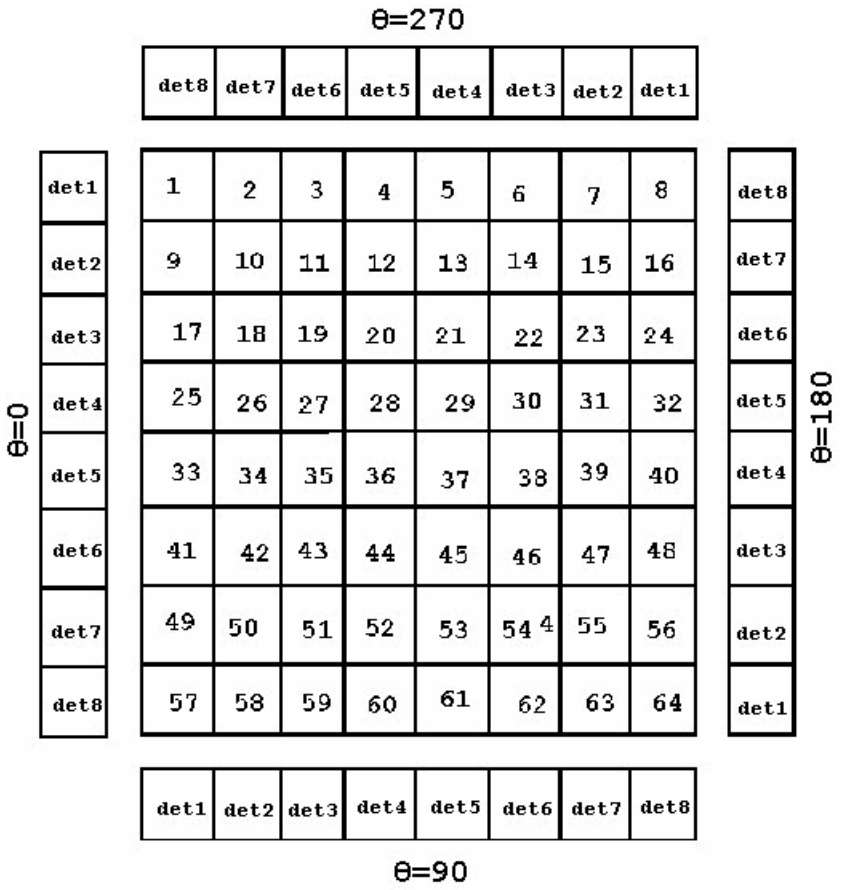



Σχήμα 2.7: Περιστροφή ενός ανιχνευτή αποτελούμενου από 8 κυψελίδες και τα εμβαδά που σχηματίζει η ζώνη ανίχνευσης με τα pixel της προς ανακατασκευή εικόνας.

Ο υπολογισμός των εμβαδών μπορεί να απλοποιηθεί αν ληφθούν υπόψη οι συμμετρίες που υπάρχουν στο σχήμα. Στην παρούσα υλοποίηση το κέρδος από την αξιοποίηση των συμμετριών είναι μόνο η μείωση του υπολογιστικού χρόνου εύρεσης του πίνακα. Όμως τροποποιώντας τον κώδικα του αλγόριθμου, είναι δυνατόν να αποθηκεύεται μόνο ένα μέρος του πίνακα με προφανή οφέλη κατά την ανακατασκευή μιας εικόνας μεγαλύτερων διαστάσεων, όπου ο χρόνος προσπέλασης του πίνακα είναι μεγαλύτερος. Οι συμμετρίες αυτές μπορούν να αξιοποιηθούν και σε μια υλοποίηση που θα υπολογίζει απευθείας τα στοιχεία $p_{ij}$, χωρίς να αποθηκεύει τον πίνακα. Ενδεικτικά, στην περίπτωση μιας εικόνας 128x128, όπου λαμβάνονται 36 γωνίες ο χρόνος υπολογισμού του πίνακα είναι περίπου 1 λεπτό. Ωστόσο, στην περίπτωση της τριασδιάστατης τομογραφικής ανακατασκευής ο χρόνος αυτός αυξάνεται σημαντικά, καθώς και οι απαιτήσεις αποθήκευσης και διαχείρισης του πίνακα αυτού.

Οι σημαντικότερες συμμετρίες είναι οι εξής:

1) Η πρώτη αφορά στη σειρά με την οποία κάθε ανιχνευτής βλέπει τα pixels στις 0º, 90º, 180º και 270º. Αξιοποιώντας τη συμμετρία αυτή είναι δυνατόν να γίνουν οι υπολογισμοί των εμβαδών μόνο στο διάστημα $0º<\theta<90º$. Έτσι οι απαιτούμενοι υπολογισμοί περιορίζονται στο ¼. Με δεξιόστροφη φορά περιστροφής οι θέσεις από τις οποίες περνά ο ανιχνευτής και οι σχετικές θέσεις του ως προς την εικόνα στις 0º, 90º, 180º και 270º φαίνονται στο σχήμα 3.3:




Σχήμα 2.8: Σχετικές θέσεις ανιχνευτών και pixels εικόνας στις 0º, 90º, 180º και 270º.

2) Η δεύτερη συμμετρία αφορά στη σχετική θέση των ανιχνευτών και των pixels σε

**detectors**

**pixels**

$$\begin{bmatrix} a\ b\ c\ \cdots\cdots\cdots \\ d\ e \\ f \\ \\ \cdots\cdots\cdots\cdots\cdots f \\ \cdots\cdots\cdots\cdots e\ d \\ \cdots\cdots\cdots\cdots c\ b\ a \end{bmatrix}$$

συγκεκριμένες γωνίες. Διαπιστώνεται ότι υπάρχει συμμετρία της μορφής:

Δηλαδή στην περίπτωση που υπάρχουν 8 ανιχνευτές ό,τι "βλέπει" ο ανιχνευτής 1 σε μια γωνία θ, το βλέπει με την αντίστροφη σειρά ο ανιχνευτής 8 στη γωνία 180+θ κ.ο.κ. Έτσι είναι δυνατόν οι υπολογισμοί να περιοριστούν στους μισούς ανιχνευτές με αποτέλεσμα τον υποδιπλασιασμό του απαιτούμενου χρόνου.

Ο γεωμετρικός πίνακας πρέπει να περιέχει τις πιθανότητες ανίχνευσης ενός φωτονίου που εκπέμπεται από ένα pixel σε κάθε ανιχνευτή. Για να γίνει αυτό πρέπει να διαιρεθεί το κάθε εμβαδόν με το άθροισμα των εμβαδών που σχηματίζει το pixel αυτό με όλες τις ζώνες ανίχνευσης όλων των ανιχνευτών. Με τον τρόπο που έχει οριστεί ο πίνακας του συστήματος, το μόνο που απαιτείται να γίνει είναι να διαιρεθεί κάθε στοιχείο μιας γραμμής με το άθροισμα των στοιχείων της γραμμής αυτής. Ο πίνακας που προκύπτει είναι ο πίνακας πιθανοτήτων του συστήματος. [10-12]

## 2.4 Επαναληπτική ανακατασκευή με MLEM

Το 1979 οι L. Shepp και Y. Vardi, παρουσίασαν ένα μαθηματικό μοντέλο για την τομογραφία εκπομπής, όπου μία άγνωστη πυκνότητα εκπομπής (κατανομή της συγκέντρωσης του ραδιοϊσοτόπου) λ=λ(x,y,z) μπορεί να υπολογιστεί όταν είναι γνωστός ένας αριθμός κρούσεων σε ένα πλήθος ανιχνευτών. Ο αλγόριθμος ονομάστηκε Maximum Likelihood Expectation Maximization (MLEM) και λαμβάνει υπόψη τη στατιστική φύση της εκπομπής ακτινοβολίας από μία πηγή. Σε κάθε επανάληψη υπάρχει ένα βήμα πρόβλεψης (expectation), το οποίο χρησιμοποιεί τις τρέχουσες εκτιμήτριες των τιμών των pixels και ακολουθείται από ένα βήμα μεγίστης πιθανοφάνειας (maximum likelihood), που ανανεώνει τις εκτιμήτριες αυτές.

Έστω ότι η προς απεικόνιση κατανομή διακριτοποιείται σε στοιχειώδεις όγκους (voxels) ή σε στοιχειώδεις επιφάνειες (pixels). Για απλότητα περιορίζεται η ανάλυση στις δύο διαστάσεις. Έστω ότι η εικόνα έχει διαστάσεις nxn και άρα αποτελείται από $n^2$ pixels. Έστω επίσης ότι για την ανακατασκευή μίας τομής είναι γνωστή η πληροφορία από D συνολικά ανιχνευτές. Στην περίπτωση του SPECT υποτίθεται ότι κάθε ανιχνευτής αποτελείται από n στοιχειώδεις ανιχνευτές και για θ γωνίες περιστροφής D=nxθ, αφου ο ίδιος ανιχνευτής δίνει ανεξάρτητες πληροφορίες σε διαφορετικές γωνίες. Επιπλέον απαιτείται ο υπολογισμός του πίνακα πιθανοτήτων

P του συστήματος. Ένα τυχαίο στοιχείο του πίνακα $p_{ij}$ δίνει την πιθανότητα ένα φωτόνιο, που εκπέμπεται από ένα pixel i να ανιχνευθεί από τον ανιχνευτή j.

Η εκκίνηση του αλγορίθμου γίνεται από μία τυχαία εικόνα, κάθε pixel της οποίας έχει την ίδια τιμή. Χρησιμοποιώντας τις μετρούμενες προβολές της εικόνας σε διάφορες γωνίες σε κάθε επανάληψη k γίνεται ανανέωση των εκτιμούμενων τιμών των pixels της εικόνας $I_k$ με βάση την ακόλουθη σχέση:

$$I^{k+1} = I^k \sum_{\det=1}^{D} \frac{p_{ij} d_j}{\sum_i p_{ij} I_i^k}$$

[8, 10, 11, 13, 14]

## 2.5 Επαναληπτική ανακατασκευή με OSEM

Ο αλγόριθμος διατεταγμένων υποσυνόλων (Ordered Subsets Expectation Maximization –OSEM) παρουσιάστηκε το 1994 από τους Hudson και Larkin. Βασίζεται στην ίδια ιδέα με τον ML-EM, αλλά συγκλίνει πολύ πιο γρήγορα. Η βασική του καινοτομία είναι ότι χωρίζει το σύνολο των προβολών σε υποσύνολα (subsets) και εφαρμόζει τον MLA, σε καθένα από αυτά ως εξής:

Έστω ένα σύνολο $T=\{T_1, T_2, T_3, ...\}$ προβολών το οποίο χωρίζεται σε n υποσύνολα. Η επιλογή των υποσυνόλων μπορεί να γίνει με διάφορους τρόπους και αυτά να είναι διαδοχικά ή επικαλυπτόμενα. Δηλαδή το πρώτο σύνολο $S_1=(T_1,T_2,\ldots T_\kappa)$, το δεύτερο $S_2=(T_{m+1}, T_{m+2}, \ldots T_{2m})$, …., το τελευταίο $S_k=(T_{(k-1)m+1}, T_{(k-1)m+2},\ldots T_{km})$ ή για επικαλυπτόμενα $S_1=(T_1,T_2,\ldots T_m)$, το δεύτερο $S_2=(T_1,T_2,\ldots T_m, T_{m+1}, T_{m+2}, \ldots T_{2m})$, …., το τελευταίο $S_k=( T_1,T_2,\ldots T_m, T_{m+1}, T_{m+2}, \ldots T_{2m}, T_{(k-1)m+1}, T_{(k-1)m+2},\ldots T_{km})$.

Η εκκίνηση του αλγορίθμου γίνεται και πάλι από ένα ομοιόμορφο πίνακα. Στο πρώτο υποσύνολο των προβολών εφαρμόζεται ο MLEM, το αποτέλεσμα που προκύπτει χρησιμοποιείται ως αρχική εικόνα για εφαρμογή του MLEM του δευτέρου υποσυνόλου, το αποτέλεσμα που προκύπτει χρησιμοποιείται ως αρχική εικόνα για τον MLEM του τρίτου υποσυνόλου κ.ο.κ. Η εφαρμογή του MLEM σε ένα τέτοιο υποσύνολο ονομάζεται επίπεδο (level). Μόλις συμπληρωθούν όλα τα επίπεδα έχει ολοκληρωθεί μία επανάληψη του OSEM, η οποία διαρκεί τελικά περίπου όσο και μία επανάληψη του MLEM, ενώ η ανακατασκευή συγκλίνει πολύ πιο γρήγορα. Κατά

αντιστοιχία με τον MLEM η ανανέωση των εκτιμούμενων τιμών των pixel της εικόνας $I_k$ γίνεται με βάση την ακόλουθη σχέση:

$$I^{k+1} = I^k \sum_{\det=1}^{Sp(n)} \frac{p_{ij} d_j}{\sum_i p_{ij} I_i^k}$$

όπου $S_p(n)$ είναι οι προβολές του υποσυνόλου n.

Είναι επίσης σημαντικό να επιλεγεί ο κατάλληλος αριθμός υποσυνόλων (άρα και levels) και ο αριθμός των επαναλήψεων, ώστε ποιοτικά αποδεκτές εικόνες να λαμβάνονται γρήγορα. Ο OSEM έφερε μία μικρή επανάσταση στον τομέα της επαναληπτικής ανακατασκευής τομογραφικών εικόνων και σήμερα οι περισσότερες εταιρίες κατασκευής τομογραφικών συστημάτων παρέχουν απλοποιημένες εκδόσεις του OSEM –συνήθως ως επιπλέον προσφορά- στο λογισμικό των συστημάτων SPECT και PET. [8, 15, 16]

# ΚΕΦΑΛΑΙΟ 3

# ΑΝΑΠΤΥΞΗ ΠΡΟΓΡΑΜΜΑΤΟΣ

## 3.1 Το εργαλείο Qt

### 3.1.1 Εισαγωγή στο Qt

Το Qt είναι ένα εργαλείο ανάπτυξης εφαρμογών με γραφικό περιβάλλον για τη γλώσσα C++. Το κύριο χαρακτηριστικό του είναι ότι δίνει τη δυνατότητα στον προγραμματιστή να αναπτύξει εφαρμογές, τις οποίες θα μπορεί να μεταγλωττίσει και να τρέξει σε διαφορετικά λειτουργικά συστήματα (Windows, Mac OS X, Linux, Unix). Ένα μεγάλο μέρος του Qt είναι προορισμένο στο να παράγει ανεξάρτητο πλατφόρμας γραφικό περιβάλλον σε οτιδήποτε από αναπαράσταση χαρακτήρων στη μνήμη μέχρι τη δημιουργία εφαρμογών με πολυνηματικού (multithread) γραφικό περιβάλλον. Οι εφαρμογές σε C++, που αναπτύσσονται με το Qt, τρέχουν σε όλα τα υποστηριζόμενα λειτουργικά συστήματα χωρίς αλλαγές στον κώδικα. Οι σημαντικότερες εφαρμογές που έχουν αναπτυχθεί με το Qt είναι το γραφικό περιβάλλον εργασίας για σταθμούς UNIX, KDE, ο web browser Opera, Google Earth, Skype κ.ά..

Το Qt δημοσιεύτηκε και εισήλθε στην αγορά αρχικά τον Μάιο του 1995. Οι αρχικές εκδόσεις του είχαν αναπτυχθεί από τους Νορβηγούς Haavard Nord και Eirik Chambe-Eng. Στις 20 Μαΐου του 1995 η ανέβηκε στο διαδίκτυο η πρώτη δημόσια έκδοση του Qt (0.90) . Η τελευταία έκδοση κυκλοφόρησε το καλοκαίρι του 2005 , το Qt 4, και περιέχει σχεδόν 500 κλάσεις και πάνω από 9000 συναρτήσεις οργανωμένες σε διαφορετικές βιβλιοθήκες έτσι ώστε να είναι ευκολότερη η ανάπτυξη μιας εφαρμογής. [17, 18]

### 3.1.2 Τα βασικότερα στοιχεία του Qt

Το Qt 4 αποτελείται από διαφορετικές ενότητες, η κάθε μια από τις οποίες ανήκει σε διαφορετική βιβλιοθήκη και οι οποίες υποστηρίζουν ανάπτυξη γραφικού περιβάλλοντος(GUI), Βάσεις Δεδομένων (Database), XML, προγραμματισμό δικτύων (networking), σύνδεση με τη βιβλιοθήκη Open GL, Multithreading και άλλα.

| | |
|---|---|
| QtCore | Core non-GUI classes used by other modules |
| QtGui | Graphical user interface components |
| QtNetwork | Classes for network programming |
| QtOpenGL | OpenGL support classes |
| QtSql | Classes for database integration using SQL |
| QtScript | Classes for evaluating Qt Scripts |
| QtSvg | Classes for displaying the contents of SVG files |
| QtXml | Classes for handling XML |
| QtDesigner | Classes for extending *Qt Designer* |
| QtUiTools | Classes for handling *Qt Designer* forms in applications |
| QtAssistant | Support for online help |
| Qt3Support | Qt 3 compatibility classes |
| QtTest | Tool classes for unit testing |

Σχήμα 3.1 : Οι ενότητες (modules) του Qt 4

Η βασικότερη ενότητα (module) είναι το QtCore, πάνω στην οποία βασίζονται οι υπόλοιπες ενότητες. Περιέχει γενικές κλάσεις, που μπορούν να χρησιμοποιηθούν από οποιαδήποτε εφαρμογή όπως κλάσεις για διάβασμα και γράψιμο αρχείων (QFile), για ροή δεδομένων (QDataStream), για λίστες ή συνδεδεμένες λίστες (QList , QlinkedList), για επεξεργασία συμβολοσειρών (Qstring) και πολλές άλλες.

Η ενότητα για το γραφικό περιβάλλον (QtGui module) περιέχει κλάσεις για την ανάπτυξη εφαρμογών με γραφικό περιβάλλον. Τέτοιες είναι κλάσεις για τον έλεγχο της ροής και των ρυθμίσεων της εφαρμογής (QApplication), για την εισαγωγή γραφικών στοιχείων για την επικοινωνία με τον χρήστη (QPushButton , QCheckButton, QDialog, QLabel κ.α.), για την επεξεργασία εικόνας (QImage, QPicture κ.α.).

Οι κλάσεις είναι ιεραρχικά δομημένες περνώντας από τις αφηρημένες γενικότερες κλάσεις σε συγκεκριμένες και ειδικότερες. Για παράδειγμα η κλάση

QPushButton, που αναπαριστά ένα κουμπί, κληρονομεί από την αφηρημένη κλάση QButton και η οποία με τη σειρά της κληρονομεί από την QWidget κ.ο.κ..

Πέρα από τα βασικά στοιχεία ενός γραφικού περιβάλλοντος, όπως κουμπιά, περιοχές κειμένου, λίστες κλπ. , τα οποία στο Qt κληρονομούν από μια γενική, την QWidget , θα ήταν ενδιαφέρον σε αυτό το σημείο να εξετάσουμε δυο άλλα σημεία, το πώς επικοινωνούν αυτά τα στοιχεία μεταξύ τους και πως δομούνται τα γεγονότα (events) στο Qt. [19]

### 3.1.3 Η λειτουργία Signals and Slots

Η λειτουργία Signals and Slots χρησιμοποιείται για την επικοινωνία μεταξύ αντικειμένων. Στο προγραμματισμό γραφικών περιβαλλόντων (GUI) όταν μεταβάλλουμε μια ιδιότητα ενός στοιχείου (π.χ. μιας ετικέτας) , τα οποία στο εξής θα τα λέμε widgets, συχνά θέλουμε ένα άλλο widget να ειδοποιηθεί για αυτό. Γενικότερα θέλουμε αντικείμενα κάθε είδους να μπορούν να επικοινωνήσουν μεταξύ τους. Για παράδειγμα όταν πατάμε το κουμπί Close σε ένα πρόγραμμα λογικά θέλουμε να κληθεί η συνάρτηση Close() του προγράμματός μας.

Ένα σήμα (signal) εκπέμπεται όταν συμβαίνει ένα συγκεκριμένο γεγονός. Τα widget του Qt έχουν αρκετά προδιαμορφωμένα σήματα, αλλά κάποιος μπορεί να φτιάξει υποκλάσεις αυτών για να προσθέσει τα δικά του σήματα. Το slot είναι η συνάρτηση που καλείται σαν απάντηση σε κάποιο συγκεκριμένο σήμα και υπάρχουν και εδώ πολλά προδιαμορφωμένα slot ,αν και κάποιος μπορεί αν προσθέσει τα δικά του σε υποκλάσεις για διαχειριστεί τα σήματα που τον ενδιαφέρουν.

Οι συναρτήσεις Slots είναι κανονικές συναρτήσεις μέλη της C++ και μπορούν να είναι εικονικές, να υπερφοτωθούν, να είναι δημόσιες, προστατευμένες ή ιδιωτικές, μπορούν να προσπελασθούν όπως κάθε άλλη συνάρτηση της C++ και μπορούν να έχουν παραμέτρους οποιαδήποτε τύπου. Η διαφορά έγκειται στο ότι ένα slot μπορεί να συνδεθεί με ένα signal, όπου και καλείται αυτόματα κάθε φορά που εκπέμπεται το signal. Η πρόταση connect(), η οποία συνδέει ένα signal με ένα slot έχει την παρακάτω μορφή:

```
connect(sender, SIGNAL(signal), receiver, SLOT(slot));
```

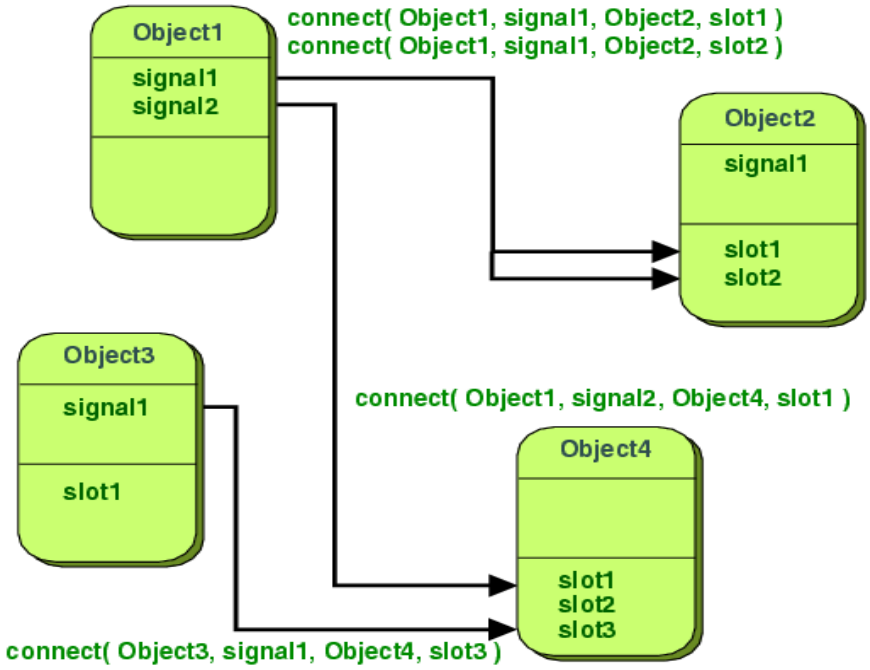
όπου τα sender και receiver είναι δείκτες σε αντικείμενα QObjects και τα signal και slot είναι ονόματα συναρτήσεων χωρίς τα ονόματα των παραμέτρων τους.

*Παράδειγμα:*

connect(slider, SIGNAL(valueChanged(int)),spinBox, SLOT(setValue(int)));

Με αυτήν την πρόταση συνδέουμε ένα slider με ένα spinbox. Όταν αλλάζει η τιμή στο slider εκπέμπεται ένα σήμα, το οποίο λαμβάνει ένα slot στο spinbox και αλλάζει την τιμή του spinbox.

Μπορεί κάποιος να συνδέσει όσα signals θέλει σε ένα slot και ένα signal μπορεί να συνδεθεί σε όσα slot χρειάζεται. Επίσης είναι δυνατό η εκπομπή ενός signal να πυροδοτεί την εκπομπή ενός άλλου signal αν συνδέσουμε το πρώτο με το δεύτερο. [19]



Σχήμα 3.2: Οι πιθανές συνδέσεις αντικειμένων με την τεχνική Signals and Slots.

### 3.1.4 Αντιμετώπιση των γεγονότων στο Qt

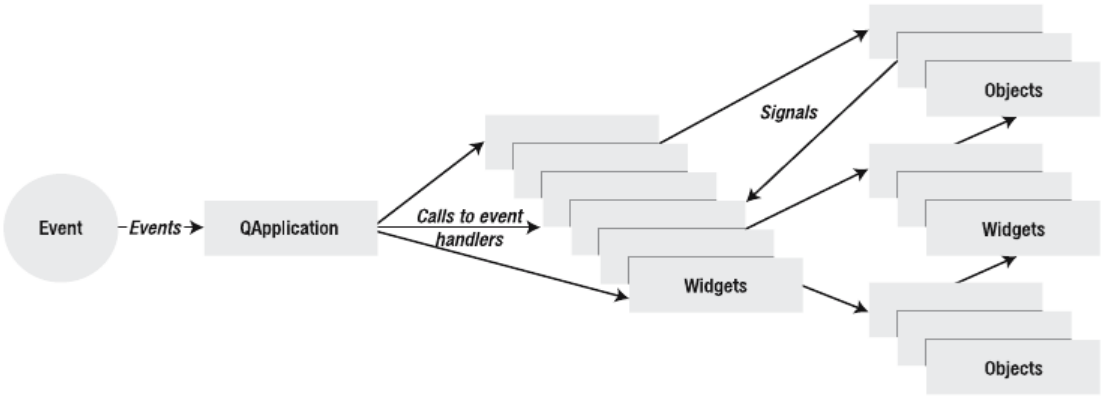
Όλες οι εφαρμογές, που κατασκευάζονται με το Qt είναι οδηγούμενες από γεγονότα(event-driven) και γι' αυτό το λόγο δεν μπορεί κάποιος να ακολουθήσει το μονοπάτι εκτέλεσης από συνάρτηση main προς όλα τα μέρη της εφαρμογής. Αντιθέτως, η εφαρμογή αρχικοποιείται από τη συνάρτηση main, η οποία καλεί την μέθοδο exec() σε ένα αντικείμενο QApplication και έτσι ξεκινάει ο βρόγχος γεγονότων του προγράμματος (event loop). Το αντικείμενο QApplication περιμένει

γεγονότα, τα οποία στη συνέχεια τα περνάει σε κάθε επηρεαζόμενο αντικείμενο QObject.

Τα γεγονότα (events) παράγονται από το σύστημα παραθύρων (window system) ή από το Qt ως απάντηση σε διάφορα συμβάντα. Όταν ο χρήστης πιέσει ή αφήσει ένα πλήκτρο, παράγεται ένα ανάλογο γεγονός (key event). Όταν ένα παράθυρο δημιουργείται για πρώτη φορά, παράγεται ένα αντίστοιχο γεγονός (paint event) έτσι ώστε το παράθυρο αυτό να σχεδιαστεί. Τα περισσότερα γεγονότα είναι απαντήσεις σε κάποιες ενέργειες του χρήστη, αλλά υπάρχουν και άλλα που παράγονται ανεξάρτητα από το σύστημα.

Όταν κάποιος προγραμματίζει με το Qt σπάνια χρειάζεται τα γεγονότα, καθώς υπάρχει η λειτουργία signals and slots. Τα γεγονότα χρησιμεύουν όταν γράφουμε τα δικά μας widget ή όταν θέλουμε να διαμορφώσουμε τη συμπεριφορά των υπαρχόντων widget.

Τα γεγονότα χειρίζονται από τους χειριστές γεγονότων (event handlers), οι οποίοι είναι εικονικές προστατευμένες μέθοδοι που οι κλάσεις των widget υπερβαίνουν όταν χρειάζεται να αντιδράσουν σε ένα γεγονός. Στο Qt τα γεγονότα είναι αντικείμενα, που κληρονομούν από την αφηρημένη κλάση QEvent, η οποία δίνει τη δυνατότητα στο δέκτη ενός γεγονότος ν αποδεχτεί ή να αγνοήσει ένα γεγονός.



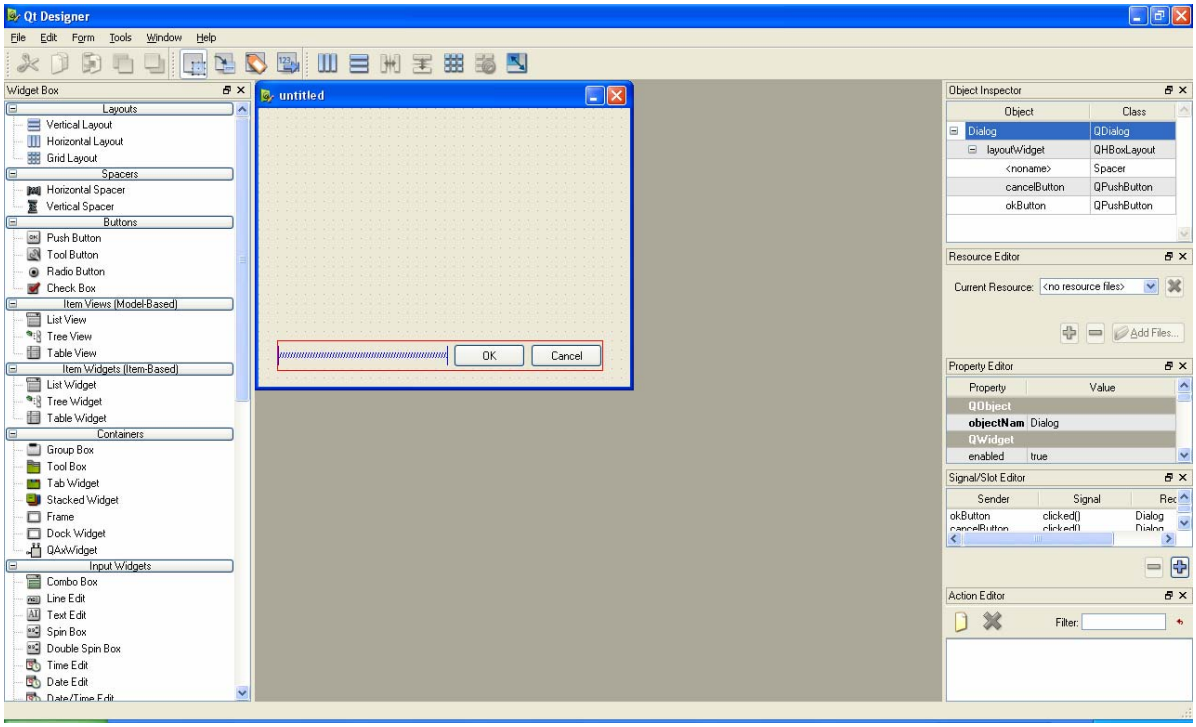

Σχήμα 3.3 : Τα γεγονότα που προκαλεί ο χρήστης περνούν μέσα από το αντικείμενο QApplication πριν φτάσουν στα widgets και πριν σταλθούν τα διάφορα σήματα.

Ένα πολύ ισχυρό συστατικό του μοντέλου γεγονότων του Qt, είναι το ότι ένα

αντικείμενο QObject μπορεί να τεθεί ώστε να παρακολουθεί τα γεγονότα ενός άλλου αντικειμένου QObject πριν το τελευταίο τα δει. Μια εφαρμογή της παραπάνω δυνατότητας είναι να υλοποιήσουμε την απαιτούμενη συμπεριφορά ενός αντικειμένου για τα γεγονότα των παιδιών του, που μας ενδιαφέρουν. Αυτή η συμπεριφορά μπορεί να επιτευχθεί με την χρήση των φίλτρων γεγονότων(event filters), τα οποία εγκαθίστανται σε δυο βήματα. Αρχικά συγχρονίζουμε το αντικείμενο παρατηρητή με το παρατηρούμενο αντικείμενο καλώντας την μέθοδο installEventFilter() στο παρατηρούμενο αντικείμενο και στη δεύτερον αντιμετωπίζουμε τα γεγονότα του παρατηρούμενου αντικειμένου με την μέθοδο eventFilter() του αντικειμένου παρατηρητή. [19]

### 3.1.5 Qt Designer

Το Qt Designer είναι ένα εργαλείο ανάπτυξης GUI, που επιτρέπει ταχεία και αποτελεσματική ανάπτυξη διεπαφών για όλες τις υποστηριζόμενες πλατφόρμες (Windows, Linux, MacOS). Δίνει τη δυνατότητα σε ένα προγραμματιστή να σχεδιάσει και να κατασκευάσει widgets και παράθυρα διαλόγου χρησιμοποιώντας φόρμες και widgets on-screen. Τα στοιχεία, που φτιάχνονται με το Qt Designer μπορούν να εκμεταλλευτούν τη λειτουργία signals and slots του Qt και μπορεί κάποιος να έχει οποιαδήποτε στιγμή εικόνα για το πώς είναι το στοιχείο που κατασκευάζει. [19]



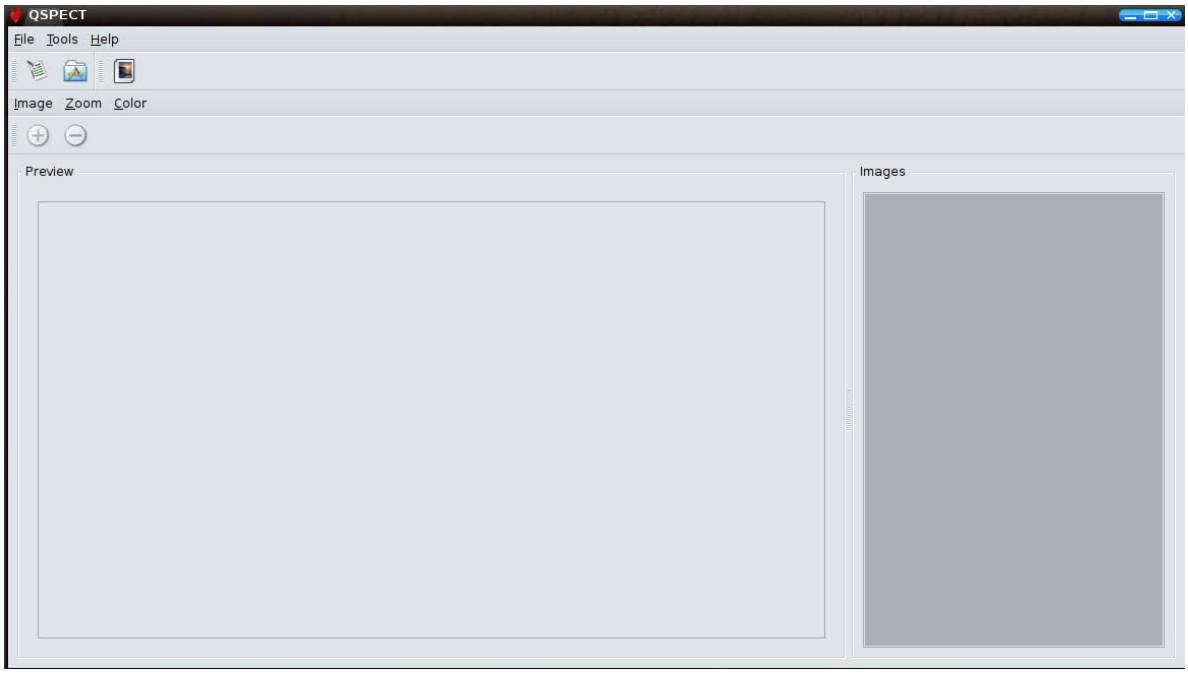

Σχήμα 3.4: Το γραφικό περιβάλλον του Qt Designer

## 3.2 Η εφαρμογή QSPECT

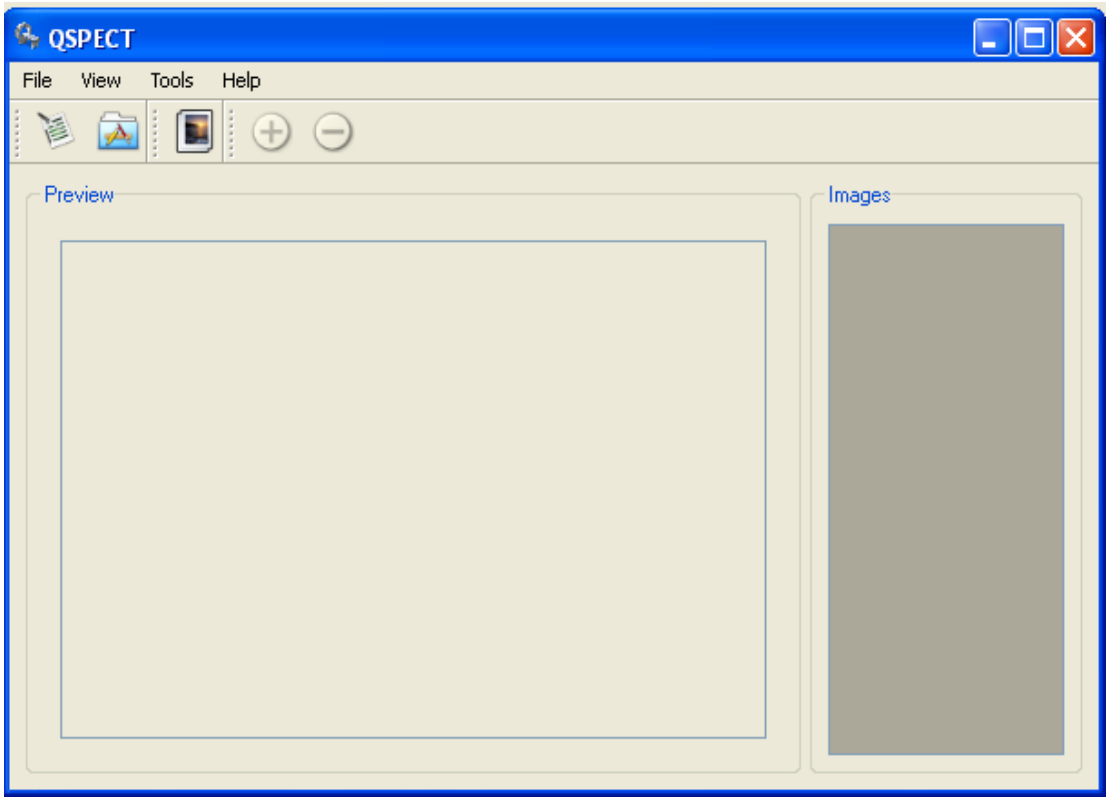
Σε αυτή την ενότητα θα αναλύσουμε την εφαρμογή, που αναπτύξαμε σε αυτήν την διπλωματική εργασία. Η εφαρμογή αυτή δέχεται δεδομένα που έχουν προέλθει από τομογραφία SPECT ενός αντικειμένου, στην περίπτωσή μας κάποιο μέρος από ανθρώπινο σώματος, και στόχος της είναι να ανακατασκευάζει τις τομές και να παράγει εικόνες σε κάποιο δημόσιο format (εδώ .bmp) και να τα εξάγει σε αρχείο txt.

Η εφαρμογή χωρίζεται σε 2 μέρη. Το πρώτο μέρος περιλαμβάνει τις λειτουργίες, που είναι σχετικές με την ανακατασκευή της τομής από τα SPECT δεδομένα, δηλαδή την κατασκευή του πίνακα πιθανοτήτων, την δημιουργία του αρχείου ανακατασκευής της τομής από το ημιτονόγραμμα (sinogram) με επιλογή ανάμεσα σε 2 αλγορίθμους ανακατασκευής και την παραγωγή της εικόνας από το προηγούμενο αρχείο ανακατασκευής. Το δεύτερο μέρος είναι ένας χώρος, όπου κάποιος μπορεί να ανοίξει ήδη κατασκευασμένες εικόνες και να κάνει κάποιες βασικές ενέργειες, όπως zoom κλπ.

Επίσης θα πρέπει να σημειώσουμε ότι το λογισμικό έχει δοκιμαστεί και τρέχει τόσο σε Windows όσο και σε Linux, καθώς έχει δημιουργηθεί με το Qt που είναι cross-platform εργαλείο.




Σχήμα 3.5 : Το πρόγραμμα QSPECT σε Linux.




Σχήμα 3.6 : Το πρόγραμμα QSPECT σε Windows XP.

### 3.2.1 Δομή της εφαρμογής QSPECT

Όπως είπαμε και παραπάνω το QSPECT αποτελείται από δυο μέρη, ένα μέρος με τις λειτουργίες που μπορεί να κάνει και ένα μέρος με τον χώρο όπου ο χρήστης μπορεί να προβάλλει εικόνες και να τις επεξεργαστεί σε ένα μικρό βαθμό.

Η κύρια κλάση της εφαρμογής είναι η simplewin, η οποία κληρονομεί από την κλάση του Qt, QMainWindow . Η κλάση QMainWindow παρέχει ένα παράθυρο για γενικές εφαρμογές με μπάρες μενού, εργαλείων, κατάστασης, dock widgets γύρω από ένα κεντρικό widget, που μπορεί να είναι περιοχή κειμένου, ζωγραφικής κλπ.. Στην περίπτωσή μας, η κλάση simplewin περιέχει ως μέλη όλες τις απαιτούμενες κλάσεις και μεθόδους για τις λειτουργίες του προγράμματος. Ακολουθούν οι δηλώσεις στο header file simplewin.h.

```
class simplewin : public QMainWindow
{
    Q_OBJECT
public:
    simplewin();
      bool eventFilter(QObject *target, QEvent *event);
```

Όταν ο χρήστης ανοίξει κάποιες εικόνες με το πρόγραμμα, οι προεπισκοπήσεις αυτών των εικόνων φαίνονται στο αριστερό μέρος του κύριου widget. Αυτές οι προεπισκοπήσεις (preview) μπορούν να έχουν μέγιστο μέγεθος 140 x 140 pixels, έτσι αν μια εικόνα είναι μεγαλύτερη δεν φαίνεται ολόκληρη. Στην περίπτωση αυτή, αν ο χρήστης πατήσει διπλό click πάνω σε μια από αυτές τις προεπισκοπήσεις, η αντίστοιχη εικόνα φορτώνεται σε πραγματικό μέγεθος στο δεξιό μέρος του προγράμματος, όπου χρήστης μπορεί να την μεγαλώσει, να την μικρύνει, να της δώσει μια χρωματική απόχρωση ή να την κάνει χρωματιστή αν αυτή είναι ασπρόμαυρη. Με την συνάρτηση `eventFilter` ελέγχουμε την συμπεριφορά ενός αντικειμένου simplewin, στην περίπτωση που ο χρήστης πατήσει διπλό click πάνω σε μια προεπισκόπηση(preview) εικόνας.

```
private slots:
      void open();
      void open(QStringList slist);
      void print();
      void save();
      void imgdelete();
      void zoomIn();
```

```
    void zoomOut();
    void normalSize();
    void setcolor();
    void multicolor();
    void fitToWindow();
    void Pij_Fill();
    void twoDM();
    void CreateImage();
    void about();
```

Οι μέθοδοι slots ενεργοποιούνται όταν εκπεμφθεί ένα σήμα που έχει οριστεί να τις ενεργοποιεί. Για παράδειγμα η μέθοδος about() ενεργοποιείται όταν ο χρήστης πατήσει από την μπάρα εργαλείων την επιλογή Help->about.

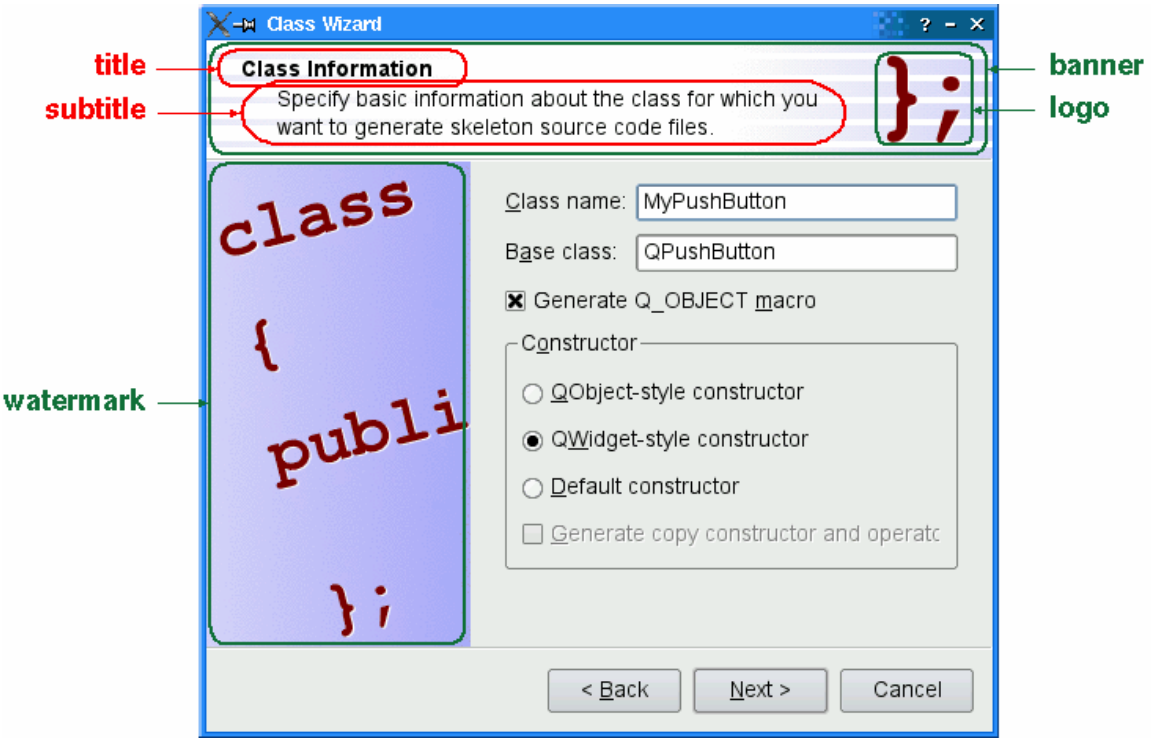
```
private:
    void createActions();
    void createMenus();
    void createToolBars();
      void createPreviewGroupBox();
      void createImagesGroupBox();
```

Οι παραπάνω 5 μέθοδοι κατασκευάζουν το γραφικό περιβάλλον του προγράμματος και καλούνται μια φορά στην αρχή του κατασκευαστή simplewin(). Η createActions() δημιουργεί τις ενέργειες μπορούν να γίνουν όταν ο χρήστης έχει κάποιου είδους αλληλεπίδραση με αυτό (π.χ. πατά ένα κουμπί ή κάνει μια επιλογή μέσα από ένα μενού). Οι ενέργειες αυτές ανήκουν σε μια κλάση του Qt, την QAction, η οποία παρέχει ένα αφηρημένο user interface για κάποια ενέργεια και μπορεί να εισαχθεί σε widgets. Γενικώς στις διάφορες εφαρμογές κάποιες κοινές εντολές μπορεί να καλούνται μέσα από μενού, μπάρες εργαλείων, συντομεύσεις μέσα από το πληκτρολόγιο (π.χ. σε μια εφαρμογή συνήθως υπάρχουν μια επιλογή στο menu, ένα κουμπί και μια συντόμευση πληκτρολογίου για το άνοιγμα ενός αρχείου). Από τη στιγμή που ο χρήστης θέλει αυτές οι εντολές να πραγματοποιούνται με τον ίδιο τρόπο είναι χρήσιμο κάθε τέτοια εντολή να αντιπροσωπεύεται ως μια ενέργεια, που θα μπορεί να προστεθεί σε μενού και μπάρες εργαλείων και να τα συγχρονίσει. Στην εφαρμογή μας τα διάφορα μενού και γραμμές εργαλείων καθώς και η πρόσθεση των ενεργειών σε αυτά γίνεται με τις μεθόδους createMenus(), createToolBars(). Οι μέθοδοι createPreviewGroupBox() και createImagesGroupBox() κατασκευάζουν δυο groupbox (συλλογές από widgets) για την απεικόνιση των προεπισκοπήσεων των εικόνων και των εικόνων που φορτώνει ο χρήστης αντίστοιχα.

```
  void updateActions();
  void scaleImage(double factor);
  void adjustScrollBar(QScrollBar *scrollBar, double factor);

QMenu *fileMenu;
QMenu *toolsMenu;
QMenu *helpMenu;
  QMenu *viewMenu;

QToolBar *fileToolBar;
QToolBar *editToolBar;
  QToolBar *zoomToolBar;

  QAction *openAct;
  QAction *printAct;
  QAction *saveAct;
  QAction *deleteAct;
  QAction *zoomInAct;
  QAction *zoomOutAct;
  QAction *normalSizeAct;
  QAction *fitToWindowAct;
  QAction *setColorAct;
  QAction *multiColorAct;
QAction *exitAction;
QAction *aboutAction;
QAction *PijAction;
QAction *twoDMAction;
  QAction *CreateImageAction;
QAction *aboutQtAction;

  QWidget *centralWidget;

  QList<PijWizard *> *pijList;
  int countpij;
  QList<twoDLWizard *> *twoDLList;
  int counttwoDL;
  QList<PicWizard *> *PicList;
  int countPic;

  QGroupBox *previewGroupBox;
  IconPreviewArea *previewArea;
  QScrollArea *previewscrollArea;

  QGroupBox *imagesGroupBox;
  QList<double> process_line;
  QLabel *label;
  QScrollArea *scrollArea;
  double scaleFactor;

  QPrinter printer;

  QImage *image;
  QImage *newimage;
  QString imgpath;
  int pindex;

  QVector<QRgb> colors;
  };
```

Ένα σημαντικό σημείο του προγράμματος είναι η εφαρμογή οδηγών

(wizards), έτσι ώστε η εκτέλεση κάποιων λειτουργιών από τον χρήστη να είναι ευκολότερη. Σε αυτό το σημείο θα ήταν ενδιαφέρον να αναφέρουμε μερικά στοιχεία για την κατασκευή του οδηγού (wizard). Ένας οδηγός είναι ένας ειδικός τύπος παραθύρου διαλόγου για είσοδο δεδομένων (input dialog window), αποτελείται από μια σειρά από σελίδες και σκοπός του είναι να οδηγήσει το χρήστη μέσα από μια διαδικασία βήμα-βήμα. Η κλάση QtWizard κληρονομεί από τη γενικότερη κλάση του Qt QDialog και κάθε σελίδα ανήκει στην κλάση QtWizardPage. Κάθε στιγμή μόνο μια σελίδα προβάλλεται στον οδηγό και κάθε σελίδα έχει τα ακόλουθα χαρακτηριστικά, τα οποία αποδίδονται από τη κλάση QtWizard : ένα τίτλο, ένα υπότιτλο και ένα σύνολο από χώρους για εικόνες (pixmaps), οι οποίοι χρησιμοποιούνται ή όχι ανάλογα με το στυλ που έχει επιλεγεί. Το παρακάτω σχήμα δείχνει την απόδοση αυτών των χαρακτηριστικών σε μια σελίδα με Modern style.




Σχ.3.8:Τα χαρακτηριστικά (attributes) της κλάσης QtWizardPage όπως αποδίδονται στην κλάση QWizard σε Modern Style.

Μπορούμε να εισάγουμε μια σελίδα σε έναν οδηγό χρησιμοποιώντας τις ακόλουθες μεθόδους της κλάσης QtWizard, QtWizard::addPage() or QtWizard::setPage(). Κάθε σελίδα προσφέρει 5 εικονικές μεθόδους, τις οποίες ο προγραμματιστής μπορεί να εφαρμόσει, έτσι ώστε να επεξεργαστεί όπως θέλει τη σελίδα:

- Η μέθοδος initializePage() καλείται για να αρχικοποιηθούν τα περιεχόμενα της σελίδας όταν ο χρήστης πατά το κουμπί **Next**. Αν θέλουμε να διαφοροποιήσουμε τα περιεχόμενα μιας σελίδας ανάλογα με το περιεχόμενο των προηγούμενων της τότε πρέπει να χρησιμοποιήσουμε αυτή τη μέθοδο.
- Η μέθοδος cleanupPage() καλείται όταν θέλουμε να καθαρίσουμε τα περιεχόμενα μιας σελίδας όταν ο χρήστης πατά το κουμπί **Back**.
- Η μέθοδος validatePage() επιβεβαιώνει τη σελίδα όταν ο χρήστης πατά **Next** ή **Finish**.
- Η μέθοδος nextId() επιστρέφει τον αριθμό ID της επόμενης σελίδας. Αυτή είναι χρήσιμη όταν κατασκευάζουμε μη γραμμικούς οδηγούς (non-linear wizards) και μας επιτρέπει να δημιουργήσουμε διαφορετικά μονοπάτια της ροής του οδηγού ανάλογα με τις πληροφορίες που δίνει ο χρήστης.
- Η μέθοδος isComplete() καλείται για να διευκρινιστεί αν τα κουμπιά **Next** ή **Finish** πρέπει να μπορούν να πατηθούν ή όχι.

Σε πολλούς οδηγούς τα περιεχόμενα μιας σελίδας επηρεάζουν τις προκαθορισμένες τιμές πεδίων επόμενης σελίδας ή ακόμη χρησιμοποιούνται για τις ενέργειες που ακολουθούν την ολοκλήρωση ενός οδηγού. Για την εύκολη επικοινωνία μεταξύ σελίδων ένας ειδικός μηχανισμός πεδίων υποστηρίζεται από την κλάση QtWizard, που επιτρέπει σε έναν προγραμματιστή να καταχωρίσει μια τιμή σε ένα πεδίο (π.χ. σε μια περιοχή γραμμής QLineEdit) και να την προσπελάσει από οποιαδήποτε σελίδα. Τα πεδία αυτά είναι γενικά και έχουν ισχύ σε ολόκληρο τον οδηγό και έτσι μια σελίδα μπορεί να έχει πρόσβαση σε αυτά χωρίς να γνωρίζει τη λογική που είναι οργανωμένη μια άλλη σελίδα. Τα πεδία που θέλουμε να κάνουμε γενικά τα καταχωρούμε με τη μέθοδο registerField() και τα προσπελαύνουμε με τις field() και setField(). Ως παράδειγμα ακολουθεί η καταχώρηση του περιεχομένου του αντικειμένου fileline, που είναι μια γραμμή κειμένου(Text Line), ως γενικό πεδίο με το όνομα sourcefilename, έτσι ώστε αυτό να είναι προσβάσιμο από παντού μέσα στον οδηγό. Ο αστερίσκος σημαίνει ότι η συμπλήρωση αυτού του πεδίου από το χρήστη είναι υποχρεωτική ώστε να είναι ενεργοποιημένο το κουμπί Next.

```
fileLine = new QLineEdit(this);
  registerField("sourcefilename*",fileLine );
```

### 3.2.2 Οι λειτουργίες του προγράμματος

Σε αυτή την ενότητα θα αναλύσουμε τις διάφορες λειτουργίες της εφαρμογής QSPECT. Όπως αναφερθήκαμε και προηγουμένως σκοπός αυτής της εφαρμογής είναι η ανακατασκευή τομών από δεδομένα που προέρχονται από τομογραφία SPECT. Τα δεδομένα αυτά είναι στη μορφή του ημιτονογράμματος για κάθε τομή. Η ανακατασκευή των τομών γίνεται με την εφαρμογή ενός επαναληπτικού αλγόριθμου. Το πρόγραμμά μας δίνει τη δυνατότητα στο χρήστη να επιλέξει ανάμεσα σε δυο επαναληπτικούς αλγόριθμους, τον MLEM και τον OSEM.

- **Κατασκευή του πίνακα πιθανοτήτων συστήματος (system's probability matrix)**

Στην κεντρική ιδέα ενός επαναληπτικού αλγορίθμου δίνεται μια αρχική μη μηδενική εκτίμηση της εικόνας και σε κάθε επαναληπτικό βήμα η εικόνα ανανεώνεται χρησιμοποιώντας τα δεδομένα προβολής, την εκτίμηση της εικόνας του προηγούμενου βήματος και ένα κανόνα προσδοκώμενης ελάττωσης (expectation minimization rule). Το στοιχείο κλειδί σε αυτούς τους αλγορίθμους είναι ο πίνακας πιθανοτήτων του συστήματος, ο οποίος μας παρέχει τις πιθανότητες, που έχει ένα φωτόνιο που εκπέμπεται από ένα pixel της άγνωστης εικόνας, να ανιχνευτεί σε κάποια περιοχή του ανιχνευτή. Σε αυτόν τον πίνακα πιθανοτήτων λαμβάνεται υπ' όψιν η γεωμετρία του συστήματος. Επίσης, άλλες φυσικές παράμετροι μπορούν να συμπεριληφθουν όπως, η ανταπόκριση της διάταξης παραλληλισμού της ακτινοβόλησης (collimators response), η εξασθένηση κλπ..

Όπως καταλαβαίνει κανείς για την ανακατασκευή της τομής με τη βοήθεια επαναληπτικών αλγορίθμων απαραίτητη είναι η γνώση του γεωμετρικού πίνακα του συστήματος P (βλ. κεφάλαιο 2). Γι' αυτό το λόγο πρέπει ο χρήστης πρώτα απ' όλα να κατασκευάσει το κατάλληλο πίνακα πιθανοτήτων για το σύστημά του. Η κατασκευή αυτού του πίνακα πιθανοτήτων γίνεται μέσω ενός οδηγού και είναι δυνατή η επιλογή ανάμεσα σε τρεις διαφορετικούς αλγορίθμους : i) ιδανική διάταξη παραλληλισμού ii) μη ιδανική διάταξη παραλληλισμού iii) μη ιδανική διάταξη παραλληλισμού και

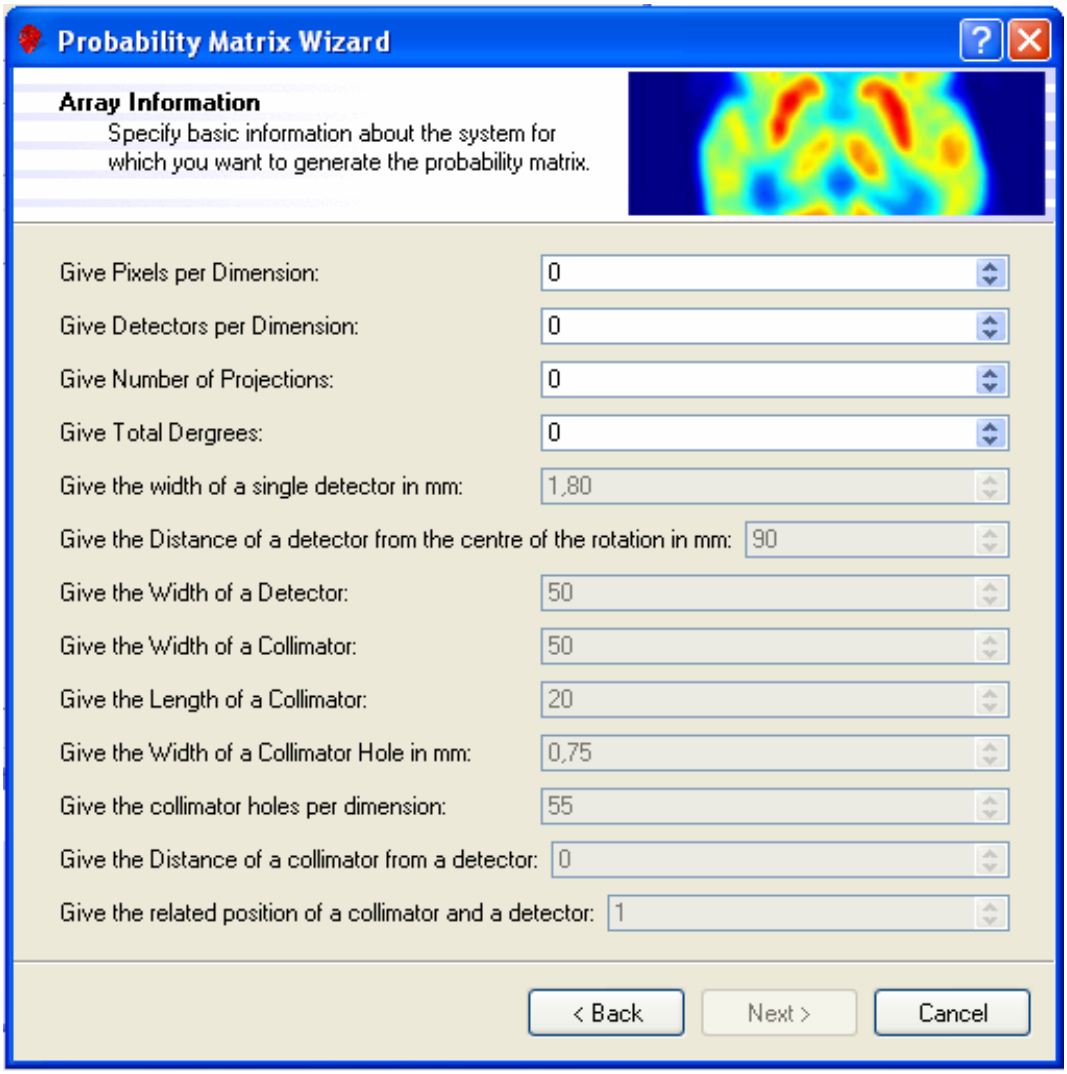
διακριτοποιημένος σπινθηριστής (κρύσταλλος).

i. Είναι η πιο απλή περίπτωση, όπου η περιοχή ανίχνευσης του κάθε ανιχνευτή καθορίζεται από δυο παράλληλες γραμμές. Επομένως η απόσταση κάθε pixel από τον ανιχνευτή δεν επηρεάζει τους υπολογισμούς. Από τον χρήστη ζητούνται το πλήθος των pixels ανά διάσταση, το πλήθος των ανιχνευτών ανά διάσταση, το πλήθος των επιθυμητών προβολών και ο συνολικός αριθμός των γωνιών προς ανίχνευση.

ii. Για την κατασκευή του πίνακα πιθανοτήτων για μη ιδανική διάταξη παραλληλισμού χρειαζόμαστε επιπλέον τις εξής παραμέτρους της διάταξης : το μήκος της διάταξης και την απόστασή της από το κέντρο περιστροφής καθώς και το πάχος της οπής της διάταξης παραλληλισμού. Εδώ πρέπει να σημειώσουμε ότι κάνουμε την παραδοχή ότι οι οπές της διάταξης παραλληλισμού συμπίπτουν τέλεια με τα pixels του ανιχνευτή. Αυτή είναι και η περίπτωση που απαντάται στα κλινικά συστήματα, όπου δεν υπάρχει διακριτοποιημένος κρύσταλλος. Οι παραπάνω πληροφορίες δίνονται από τον χρήστη ως Distance of a detector from the centre of the rotation in mm (εδώ περιλαμβάνονται και το μήκος της διάταξης παραλληλισμού) και width of a single detector in mm (είναι ίδια με το πάχος των οπών της διάταξης παραλληλισμού).

iii. Στην περίπτωση του διακριτοποιημένου σπινθηριστή (κρυστάλλου, pixilated scintillator) η σχετική θέση των οπών της διάταξης παραλληλισμού και των pixels του κρυστάλλου λαμβάνονται υπ' όψιν, παρόλο που στην πράξη αυτό δεν μπορεί να πραγματοποιηθεί εύκολα. Περιορίζουμε το πρόβλημα στις 2 διαστάσεις για να μπορεί αυτό το φαινόμενο να υπολογιστεί γεωμετρικά και να εισαχθεί στον πίνακα πιθανοτήτων, δεδομένου ότι η ανακατασκευή πραγματοποιείται σε διαδοχικές τομές. Σε τρισδιάστατη ανακατασκευή θα έπρεπε να μελετηθεί το φαινόμενο στο χώρο. Στους διακριτοποιημένους σπινθηριστές συνήθως ο αριθμός των Pixels του κρυστάλλου καθορίζουν το μέγεθος της ανακατασκευασμένης εικόνας και ένα Pixel αντιστοιχεί σε ένα ανιχνευτή. Όταν η οπή του συστήματος παραλληλισμού καλύπτει ένα μέρος ενός pixel ορίζεται ένας 'υπό-ανιχνευτής'. Ο χρήστης για την κατασκευή του

συγκεκριμένου πίνακα πιθανοτήτων πρέπει να δώσει επιπλέον τις παρακάτω πληροφορίες : το πάχος ενός ανιχνευτή, το μήκος, το πάχος του συστήματος παραλληλισμού και το πάχος των οπών του, τον αριθμό των οπών του συστήματος παραλληλισμού ανά διάσταση, την απόσταση και τη σχετική θέση του συστήματος παραλληλισμού και ενός ανιχνευτή. [11]




Σχ.3.7:Εισαγωγή δεδομένων από τον χρήστη για την κατασκευή του πίνακα συστήματος με την μορφή οδηγού(wizard)

Για την κατασκευή του επιθυμητού πίνακα πιθανοτήτων συστήματος από τον χρήστη χρησιμοποιείται ένας αντίστοιχος οδηγός (wizard), για τον οποίο δημιουργήσαμε μια κλάση, PijWizard, η οποία κληρονομεί από την κλάση του Qt QtWizard. Ο οδηγός αυτός αποτελείται από 4 σελίδες, τις IntroPage, ArrayInfoPage, OutputFilesPage, ConclusionPage. Στην πρώτη σελίδα, IntroPage, ο χρήστης επιλέγει ποιο αλγόριθμο επιθυμεί να χρησιμοποιήσει . Στην ArrayInfoPage ζητούνται τα

απαραίτητα δεδομένα συστήματος για τον εκάστοτε αλγόριθμο που έχει επιλεγεί. Στην OutputFilesPage ζητείται η διαδρομή στο δίσκο και το όνομα του αρχείου που θα αποθηκευθεί ο πίνακας πιθανοτήτων και εδώ υπάρχει ένα προκαθορισμένο (default) όνομα που σχηματίζεται από τα στοιχεία που έχουν δοθεί στην προηγούμενη σελίδα. Ο χρήστης πρέπει να επιβεβαιώσει τις προηγούμενες ενέργειες στην ConclusionPage για να ξεκινήσει η κατασκευή του πίνακα.

Οι αλγόριθμοι για την κατασκευή του πίνακα πιθανοτήτων και για τις 3 περιπτώσεις ήταν υλοποιημένοι σε γλώσσα C και ήταν αναγκαία η μετατροπή τους σε γλώσσα C++ για να ενσωματωθούν στον κώδικα του γραφικού περιβάλλοντος του προγράμματος. Σε αυτό το σημείο ίσως θα ήταν ενδιαφέρον να αναφέρουμε κάποια στοιχεία αυτής της μετατροπής.

Η C είναι μια γενικής χρήσης διαδικαστική γλώσσα προγραμματισμού σε αντίθεση με την C++ που είναι μια αντικειμενοστρεφής γλώσσα. Ένα πρόγραμμα σε C δεν περιέχει κλάσεις παρά μόνο συναρτήσεις και δομές. Τα προγράμματα σε C που υλοποιούσαν τους παραπάνω αλγόριθμους για την κατασκευή του πίνακα πιθανοτήτων ήταν δομημένα σε μια συνάρτηση main(). Μέσα στην main() πραγματοποιούνταν όλες οι λειτουργίες για την κατασκευή του πίνακα και την εγγραφή του σε ένα αρχείο εγγράφου .txt . Τα προγράμματα δεν αλληλεπιδρούσαν με τον χρήστη και οι παράμετροι του εκάστοτε πίνακα πιθανοτήτων, που ήθελε ο χρήστης να κατασκευάσει, δίνονταν μέσα στον κώδικα και πριν από τη μεταγλώττιση του προγράμματος. Με αυτόν τον τρόπο για την κατασκευή πινάκων με διαφορετικές παραμέτρους έπρεπε ο χρήστης να αλλάξει τις παραμέτρους μέσα στον κώδικα, να μεταγλωττίσει ξανά τον κώδικα και να τρέξει το πρόγραμμα. Επίσης τα προγράμματα σε C έτρεχαν μέσα από τη γραμμή εντολών.

Στα πλαίσια της μετατροπής από C σε C++ δημιουργήσαμε τρεις διαφορετικές κλάσεις για κάθε έναν από τους τρεις τρόπους κατασκευής του πίνακα πιθανοτήτων, τις οποίες θέσαμε να κληρονομούν από τη κλάση QWidget του Qt έτσι ώστε να μπορούμε να εκμεταλλευτούμε τις λειτουργίες που παρέχει αυτό το εργαλείο. Οι διάφορες παράμετροι(το πλήθος των pixels ανά διάσταση, το πλήθος των ανιχνευτών ανά διάσταση κλπ.) δηλώθηκαν ως ιδιωτικά μέλη της κλάσης στο αντίστοιχο header file. Στον υπάρχον κώδικα έγιναν κάποιες αλλαγές έτσι ώστε κάποιες συναρτήσεις, που χρησιμοποιούνταν, να μετατραπούν στις αντίστοιχες της C++ και στη συνέχεια χωρίσαμε τον κώδικα σε 4 μέρη. Το πρώτο μέρος υλοποιεί δυο κατασκευαστές για

την εκάστοτε κλάση, όπου γίνονται κάποιες αρχικοποιήσεις και ανοίγεται το αρχείο προς εγγραφή. Επίσης δίνεται στο χρήστη η δυνατότητα να θέσει και να συλλέξει τις παραμέτρους για το κάθε είδος πίνακα πιθανοτήτων μέσω αντίστοιχων μεθόδων set και get, οι οποίες υλοποιήθηκαν inline. Τα δυο κύρια μέρη του προγράμματος είναι η μέθοδος υπολογισμού του πίνακα πιθανοτήτων με βάση τις παραμέτρους που έχουν δοθεί και η μέθοδος εγγραφής του στο αρχείο. Τέλος, σε κάθε μια από τις τρεις κλάσεις έχουν υλοποιηθεί μέσω του Qt μπάρες προόδου (progress bar) έτσι ώστε να δίνεται στο χρήστη η πληροφορία σε ποιο στάδιο βρίσκεται η κατασκευή του πίνακα πιθανοτήτων.

Ακολουθεί ενδεικτικά η δήλωση στο αντίστοιχο header file της κλάση Pij_Parallel_NoCollimator_NoDistance, που αντιστοιχεί στην πρώτη περίπτωση.

```
class Pij_Parallel_NoCollimator_NoDistance: public QWidget
{
	Q_OBJECT
public:
	Pij_Parallel_NoCollimator_NoDistance();
	Pij_Parallel_NoCollimator_NoDistance(string filename);
	int File_Write();
	void setPixels_Per_Dimension(int
val){Pixels_Per_Dimension=val;};
	void setDetectors_Per_Dimension(int
val){Detectors_Per_Dimension=val;};
	void setNumber_Of_Captures(int val){Number_Of_Captures=val;};
	void setDegrees_Of_Captures(int val){Degrees_Of_Captures=val;};
	int getPixels_Per_Dimension(){return Pixels_Per_Dimension;};
	int getDetectors_Per_Dimension(){return
Detectors_Per_Dimension;};
	int getNumber_Of_Captures(){return Number_Of_Captures;};
	int getDegrees_Of_Captures(){return Degrees_Of_Captures;};

private slots:
	void cancel();

private:
	float Pixel_Area(float Th,long int Detector_Number,long int
Pixel_Number);
	int Pixels_Per_Dimension;
	int Detectors_Per_Dimension;
	int Number_Of_Captures;
	int Degrees_Of_Captures;
	long int All_Detectors;
	long int Total_Pixels;
	FILE *arxeio;
	QProgressDialog *progress;
	string Filename;

	};
```

Η μακροεντολή Q_OBJECT στην αρχή της κλάσης είναι απαραίτητη για όλες τις κλάσεις που χρησιμοποιούν τη λειτουργία του Qt signals and slots. Η κλάση διαθέτει δυο κατασκευαστές, έναν που δίνεται το όνομα του αρχείου που θα κατασκευαστεί και έναν που χρησιμοποιείται ένα προκαθορισμένο όνομα. Στο πρόγραμμα μας ο χρήστης δίνει το όνομα που επιθυμεί να έχει το αρχείο που θα κατασκευαστεί. Στη μέθοδο File_Write() συμπληρώνεται το αρχείο με τα δεδομένα που προκύπτουν από τον υπολογισμό με βάση τις πληροφορίες που έχει δώσει ο χρήστης. Ο υπολογισμός αυτός γίνεται στην συνάρτηση Pixel_Area(float Th,long int Detector_Number,long int Pixel_Number) η οποία είναι ιδιωτική και καλείται μέσα από την File_Write(). Οι υπόλοιπες συναρτήσεις που ακολουθούν στις δηλώσεις των δημοσίων μεθόδων είναι ειδικές συναρτήσεις για την εισαγωγή(set) ή την εξαγωγή(get) πληροφοριών σχετικά με τις παραμέτρους του συστήματος , όπως το πλήθος των ανιχνευτών ανά διάσταση, και υλοποιούνται inline στο header file όπως βλέπουμε. Αυτές οι μέθοδοι καλούνται από τον οδηγό (wizard) όταν ο χρήστης έχει εισάγει τις πληροφορίες του συστήματος που του ζητούνται και επιβεβαιώσει την κατασκευή του πίνακα. Η συνάρτηση slot cancel() καλείται όταν ο χρήστης πατήσεις το κουμπί cancel στην μπάρα προόδου (progress bar) και η κατασκευή του πίνακα ακυρώνεται. Στο πεδίο των ιδιωτικών δηλώσεων δηλώνονται επίσης μεταβλητές που αντιστοιχούν στις παραμέτρους του συστήματος καθώς και διάφορες βοηθητικές μεταβλητές.

- **<u>Κατασκευή του αρχείου ανακατασκευής μιας τομής</u>**

Για την ανακατασκευή μιας τομής από δεδομένα SPECT δίνεται η δυνατότητα στο χρήστη να επιλέξει ανάμεσα σε δυο αλγορίθμους ανακατασκευής, τον MLEM και τον OSEM, οι οποίοι και οι δυο είναι επαναληπτικοί αλγόριθμοι. Για την εφαρμογή τους είναι αναγκαίος ο πίνακας πιθανοτήτων συστήματος, τον οποίο αναλύσαμε προηγουμένως πως κατασκευάζουμε, και το ημιτονόγραμμα της επιθυμητής τομής.

Όπως έχουμε αναφέρει και στο κεφάλαιο 2 τα κυριότερα μειονεκτήματα των επαναληπτικών αλγόριθμων είναι οι εκτεταμένοι υπολογισμοί και η μεγάλη χρονική διάρκεια της επεξεργασίας. Όμως, ο αλγόριθμος διατεταγμένων υποσυνόλων

(Ordered Subsets Expectation Maximization –OSEM), που βασίζεται στην ίδια ιδέα με τον MLEM, συγκλίνει πολύ πιο γρήγορα. Η βασική του καινοτομία είναι ότι χωρίζει το σύνολο των προβολών σε υποσύνολα (subsets) και εφαρμόζει τον MLA, σε καθένα από αυτά. Για παράδειγμα, χωρίζοντας τα δεδομένα των προβολών σε 6 διαδοχικά, μη επικαλυπτόμενα υποσύνολα ο χρόνος ανακατασκευής μειώνεται στο 1/6, στις ίδιες προγραμματιστικές συνθήκες.

Οι αλγόριθμοι αυτοί ήταν αρχικά αναπτυγμένοι σε γλώσσα C και τους μετατρέψαμε σε C++ , όπως κάναμε και για τους αλγορίθμους κατασκευής του πίνακα πιθανοτήτων του συστήματος. Η τεχνική που ακολουθήθηκε είναι η ίδια. Δημιουργήσαμε δυο κλάσεις για τους δυο αλγορίθμους MLEM και OSEM, την twoDMlem120 και την twoDOsem120 αντίστοιχα, οι οποίες κληρονομούν από την κλάση QWidget του Qt και ακολουθήσαμε τα αντίστοιχα βήματα όπως και για τους αλγορίθμους κατασκευής του πίνακα πιθανοτήτων.

Ας παρατηρήσουμε λίγο το header file της κλάσης twoDMlem120 ως παράδειγμα.

```
class twoDMlem120 : public QWidget
{
    Q_OBJECT
public:
      twoDMlem120();
      twoDMlem120(QString pijfilename,QStringList
sinogramfilename,QStringList savefilename);
      void init(int Pixels_Per_Dim,int Detectors_Per_Dim,int
Number_Of_Capt,int Repet);
      int Pijset();
      int ImageRec();
private slots:
      void progresscancel();
private:
      int Pixels_Per_Dimension;
      int Detectors_Per_Dimension;
      int Number_Of_Captures;
      int Repetitions;
      long int All_Detectors;
      long int Total_Pixels;
      float **Sinogram;
      float *Estimated_Slice;
      float *Sum1;
      float *Sum2;
      float *Pij;
      float *PijDj;
      long int *Lines_Of_Pij;
      long int *Rows_Of_Pij;
      long int *Number_Of_Nonzero_Pij;
      long int Nonzero_Pij;
      long int Counter1;
      long int Counter2;
```

```
        long int Counter3;
        long int k;
        ifstream  arxeio1;
        ifstream  arxeio2;
        ofstream  arxeio3;
        QProgressDialog *progress;
        string savefile;
        QStringList outputfiles;
        char * pijname;
        QStringList sinograms;
};
```

Από τους δυο παραπάνω κατασκευαστές χρησιμοποιούμε αυτόν, στον οποίο τα ονόματα των αρχείων με τον πίνακα συστήματος, το ημιτονόγραμμα και του αρχείου όπου θα αποθηκευθεί η ανακατασκευή της τομής περνιούνται ως παράμετροι. Οι επόμενες τρεις μέθοδοι init(int Pixels_Per_Dim,int Detectors_Per_Dim,int Number_Of_Capt,int Repet), Pijset(), ImageRec() είναι το κύριο σώμα της κλάσης. Η πρώτη αρχικοποιεί ένα αντικείμενο twoDMlem120 με τις απαραίτητες παραμέτρους που έχει δώσει ο χρήστης. Η δεύτερη διαβάζει τον πίνακα συστήματος και η τρίτη εφαρμόζει τον αλγόριθμο ανακατασκευής (εδώ τον MLEM) και γράφει το αποτέλεσμα στο αρχείο εξόδου που έχει δοθεί. Η συνάρτηση progresscancel() ενεργοποιείται όταν ο χρήστης πατήσει το κουμπί cancel της μπάρας προόδου. Οι επόμενες μεταβλητές αφορούν τα στοιχεία που έχει δώσει ο χρήστης καθώς και διάφορα βοηθητικά στοιχεία.

Οι παράμετροι που είναι ανάγκη να δώσει ο χρήστης για την ανακατασκευή της τομής μέσω των αλγορίθμων OSEM και MLEM είναι το πλήθος των pixels ανά διάσταση, το πλήθος των ανιχνευτών ανά διάσταση, το πλήθος των προβολών και το πλήθος των επαναλήψεων που θα γίνουν. Για τον OSEM χρειάζεται να δοθεί ακόμα ο αριθμός των υποσυνόλων που θα χωριστούν τα δεδομένα των προβολών κατά τον υπολογισμό της ανακατασκευής της τομής. Οι παραπάνω πληροφορίες καθώς επίσης και οι τοποθεσίες όπου βρίσκονται το αρχείο για τον πίνακα συστήματος και το αρχείο με τις πληροφορίες των προβολών σε μορφή ημιτονογράμματος για την τομή που πρόκειται να ανακατασκευαστεί ζητούνται από τον χρήστη μέσω ενός οδηγού (wizard). Όπως αναφέραμε και για την περίπτωση του πίνακα συστήματος προηγουμένως έτσι και εδώ δημιουργήσαμε μια κλάση twoDLWizard με 3 σελίδες, τις StartPage, SinoInfoPage, FinishPage.

Στην περίπτωση του οδηγού που χρησιμοποιούμε στο πρόγραμμά μας για την ανακατασκευή μιας τομής κάνουμε τις παρακάτω δηλώσεις στο header file.

```
class twoDLWizard : public QtWizard
{
    Q_OBJECT

public:
    twoDLWizard(QWidget *parent = 0);
    twoDLWizard(const twoDLWizard &pj , QWidget *parent = 0);
    enum { Page_Intro, Page_Info, Page_Conclusion };
    void accept();
   };
```

Δηλώνουμε δυο κατασκευαστές εκ των οποίων ο ένας είναι κατασκευαστής αντιγραφής (copy constructor) και ο οποίος είναι απαραίτητος να κατασκευαστεί επειδή η κλάση κληρονομεί από την QtWizard. Στο πρόγραμμά μας, ο κατασκευαστής αντιγραφής δεν χρησιμοποιείται και στο αρχείο υλοποίησης της κλάσης έχει τις ίδιες λειτουργίες με τον άλλο κατασκευαστή. Η μέθοδος accept() καλείται όταν ο χρήστης πατήσει το κουμπί Finish και αποδεχτεί τα περιεχόμενα που έχει εισάγει στις σελίδες του οδηγού.

Στις δηλώσεις για μια σελίδα του οδηγού δηλώνουμε τα διάφορα στοιχεία της σελίδας όπως περιοχές κειμένου, κουμπιά, ετικέτες κλπ., καθώς και τις διάφορες μεθόδους λειτουργικότητας της σελίδας. Ακολουθεί η δήλωση για την σελίδα StartPage ως παράδειγμα.

```
class StartPage : public QtWizardPage
{
    Q_OBJECT

public:
    StartPage(QWidget *parent = 0);

protected:
    void initializePage();

private slots:
    void browse();
    void browsesino();

private:
    QLabel *label;
    QLabel *labelsino;
    QLabel *sourcefileLabel;
    QLineEdit *fileLine;
    QPushButton *BrowseButton;
    QLabel *sourcefileLabelsino;
    QLineEdit *fileLinesino;
    QPushButton *BrowseButtonsino;
    QLabel *labelalg;
    QRadioButton *radio1;
    QRadioButton *radio2;
    QVBoxLayout *layout;
};
```

Οι συναρτήσεις slot browse() και browsesino() καλούνται όταν ο χρήστης πατήσει το κουμπί για την αναζήτηση στο σύστημα του αρχείου για τον πίνακα συστήματος και του αρχείου για το ημιτονόγραμμα της τομής που πρόκειται να ανακατασκευαστεί αντίστοιχα.

Στο αρχείο υλοποίησης (implementation file) της κλάσης twoDLWizard εισάγουμε τις σελίδες StartPage, SinoInfoPage και FinishPage στον κατασκευαστή της κλάσης

```
twoDLWizard::twoDLWizard(QWidget *parent)
    : QtWizard(parent)
{

    setPage(Page_Intro, new StartPage);
    setPage(Page_Info, new SinoInfoPage);
    setPage(Page_Conclusion, new FinishPage);

    setStartId(Page_Intro);


setWindowIcon(QIcon(QString::fromUtf8("C:/thesis/diplo/images/hot/car
diology_128_hot.png")));
    setPixmap(QtWizard::BannerPixmap,
QPixmap("C:/thesis/diplo/images/banner.png"));
    setPixmap(QtWizard::BackgroundPixmap,
QPixmap("C:/thesis/diplo/images/background.png"));
    setWindowTitle(tr("Image Reconstruction file Wizard"));
      }
```

και κατασκευάζουμε την μέθοδο accept(), στην οποία παίρνουμε όλες τις πληροφορίες που έχει δώσει ο χρήστης στις σελίδες του οδηγού και καλούμε τις κατάλληλες συναρτήσεις για την κατασκευή του αρχείου ανακατασκευής της τομής.

```
void twoDLWizard::accept()
{
      QString pijfile = field("sourcefilename").toString();
      QString sinofiles = field("sourcefilenamesino").toString();
```

Εδώ βλέπουμε τη λειτουργία των γενικών πεδίων field, μέσω των οποίων οι επιθυμητές μεταβλητές έχουν ισχύ σε ολόκληρο των οδηγό. Αποθηκεύουμε τις γενικές μεταβλητές sourcefilename, sourcefilenamesino που στη συγκεκριμένη περίπτωση πρόκειται για τις τοποθεσίες μέσα στο δίσκο των αρχείων με τον πίνακα συστήματος και το ημιτονόγραμμα.

```
      int siz = sinofiles.size();
      sinofiles.remove(siz-1,1);
      sinofiles.remove(0,1);
      QStringList sinofile = sinofiles.split(",");
      QString temp = sinofile.takeFirst();
      sinofile.append(temp);
```

Στην τελευταία σελίδα του οδηγού ο χρήστης επιλέγει αν θέλει να δει τις εικόνες των τομών που θα ανακατασκευαστούν ή αν θέλει να απλώς να αποθηκεύσει το αρχείο ανακατασκευής της τομής ή και τα δυο. Γι αυτό λόγο αυτό υπάρχουν στην τελευταία σελίδα δυο check boxes, όπου ο χρήστης μπορεί να κάνει αυτές τις επιλογές. Είναι αναγκαίο το περιεχόμενο αυτών των checkboxes να αποθηκευθούν σε γενικά πεδία fields, έτσι ώστε να είναι προσβάσιμα από παντού στον οδηγό. Το πεδίο check1 αφορά το checkbox που ο χρήστης δίνει την πληροφορία αν θέλει να αποθηκεύσει το αρχείο ανακατασκευής της τομής σε ένα αρχείο κειμένου. Αν θέλει να το αποθηκεύσει καλείται να δώσει ένα όνομα, αλλιώς τα περιεχόμενα της ανακατασκευής αποθηκεύονται σε ένα προσωρινό αρχείο που μετά το τέλος της ανακατασκευής διαγράφεται.

```
QStringList outfiles;
if(field("check1").toBool())
{
      for(int i=0; i<sinofile.size() ; i++)
      {
            outfiles.append(QFileDialog::getSaveFileName(
              this,
              "Choose a filename to save the image
reconstruction file for the \n"
                        +sinofile.at(i),
              QDir::currentPath(),
              "Images (*.txt)"));
      }
}
else
{
      for(int i=0 ; i<sinofile.size() ; i++)
      {
            QString t = sinofile.at(i);
            QStringList h = t.split(".");
            t.clear();
            t = h[0];
            t.append("temp.");
            t.append(h[1]);
            outfiles << t;
      }
}
```

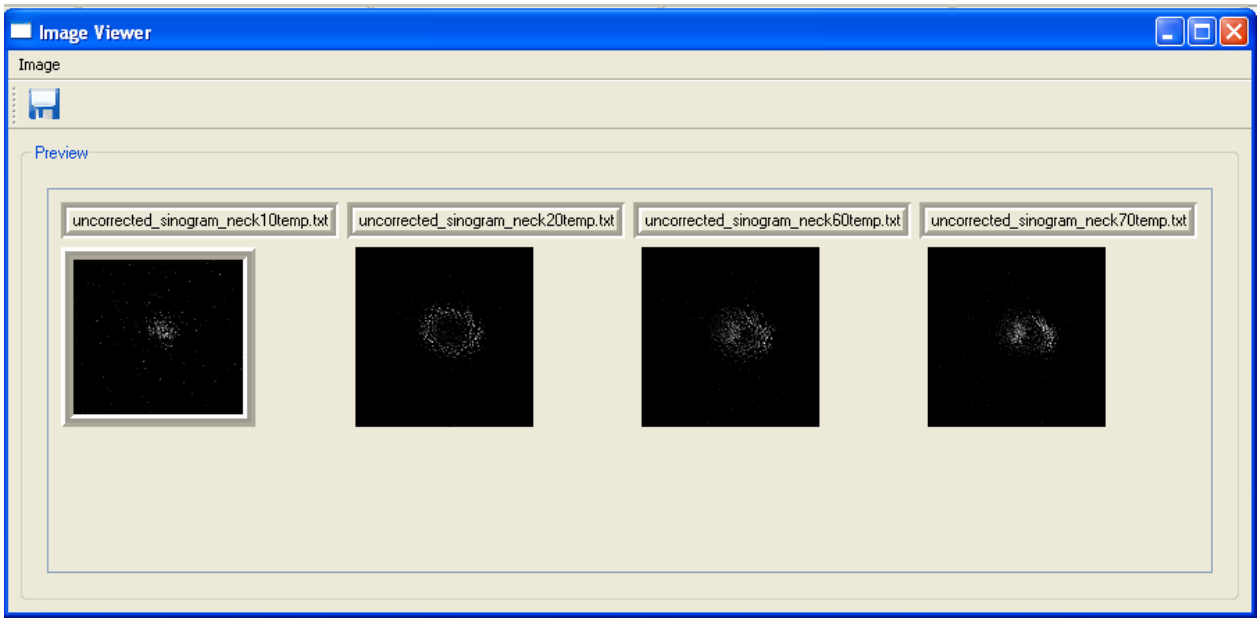
Στο παρακάτω τμήμα του κώδικα ελέγχεται ποιος αλγόριθμος έχει επιλεγεί και αντίστοιχα δημιουργείται ένα αντικείμενο της κλάσης twoDMlem120 ή της twoDOsem120, τα οποία στη συνέχεια αρχικοποιούνται με τις πληροφορίες που έχει δώσει ο χρήστης (αυτό γίνεται με τη βοήθεια των αντίστοιχων γενικών πεδίων) και στη συνέχεια διαβάζεται ο πίνακας του συστήματος με την Pijset και ανακατασκευάζεται η τομή με την ImageRec.

```
	if(field("Mlem").toBool())
	{
		twoDMlem120 *t = new
twoDMlem120(pijfile,sinofile,outfiles);
		t->init(field("pixelsperdimension").toInt(),
			field("detectorsperdimension").toInt(),
			field("numberofprojections").toInt(),
			field("repetitions").toInt());
		if(t->Pijset()==1)
		{
			if(t->ImageRec()==1)
				QMessageBox::information(this, "Action ok",
				"The image reconstruction file was
successfully created!");
			else
				QMessageBox::information(this, "Action
canceled",
				"The image reconstruction file creation
attempt was canceled!");
		}
		else
			QMessageBox::information(this, "Action canceled",
			"The image reconstruction file creation attempt was
canceled!");
	}
	else if(field("Osem").toBool())
	{
		twoDOsem120 *t = new
twoDOsem120(pijfile,sinofile,outfiles);
		t->init(field("pixelsperdimension").toInt(),
			field("detectorsperdimension").toInt(),
			field("numberofprojections").toInt(),

	field("repetitions").toInt(),field("subsets").toInt());
		if(t->Pijset()==1)
		{
			if(t->ImageRec()==1)
				QMessageBox::information(this, "Action ok",
				"The image reconstruction files were
successfully created!");
			else
				QMessageBox::information(this, "Action
canceled",
				"The image reconstruction files creation
attempt was canceled!");
		}
		else
			QMessageBox::information(this, "Action canceled",
			"The image reconstruction files creation attempt
was canceled!");
	}
```

Όταν ο χρήστης θέλει να δει τις εικόνες των τομών που έχουν ανακατασκευαστεί δημιουργείται ένα αντικείμενο της κλάσης DisplayImages. Η κλάση αυτή δέχεται ως ορίσματα τις τοποθεσίες των αρχείων με τις ανακατασκευασμένες τομές, τις διαστάσεις τους και μια παράμετρο τύπου bool που δείχνει αν τα αρχεία μορφής .txt των ανακατασκευασμένων τομών θα κρατηθούν ή θα διαγραφούν. Οι εικόνες των

ανακατασκευασμένων τομών προβάλλονται σε ένα ξεχωριστό παράθυρο όπου ο χρήστης μπορεί να επιλέξει ποιες θέλει να αποθηκεύσει σε μορφή .png .

```
        if(field("check").toBool())
        {
              DisplayImages *imgv = new DisplayImages(outfiles,
                    field("pixelsperdimension").toInt(),
                    field("detectorsperdimension").toInt(),
                    !field("check1").toBool());
              imgv->show();
        }
        QDialog::close();
        this->close();
}
```




Σχ.3.9:Ένα αντικείμενο της κλάσης DisplayImages, μέσω του οποίου προβάλλονται οι εικόνες από τις ανακατασκευασμένες τομές και ο χρήστης μπορεί να επιλέξει ποιες θα αποθηκεύσει.

- **<u>Δημιουργία της εικόνας της τομής</u>**

Είναι δυνατή η δημιουργία της εικόνας μιας τομής από ένα αποθηκευμένο αρχείο της ανακατασκευής της. Όταν ανακατασκευάζεται μια εικόνα με την λειτουργία της προηγούμενης ενότητας δημιουργείται ένα αρχείο κειμένου, όπου αποθηκεύονται οι τιμές του κάθε Pixel της ανακατασκευής. Οι τιμές αυτές αντιστοιχούν στους υπολογισμούς που γίνονται με τον αλγόριθμο ανακατασκευής που χρησιμοποιείται κάθε φορά και εκφράζουν σε γενικές γραμμές την ποσότητα των εκπεμπόμενων φωτονίων. Οι τιμές αυτές όμως δεν μπορούν να αντιστοιχηθούν σε αποχρώσεις του γκρι έτσι ώστε να μπορεί η εικόνα να προβληθεί, γιατί περιέχουν και τιμές που

ενδέχεται να είναι πολύ μεγάλες ή ακόμα και δεκαδικές τιμές. Επίσης, είναι αναγκαίο να γίνει ένα είδος βαθμονόμησης για το κάθε Pixel για να μειώσουμε την εμβέλεια των τιμών αυτών και να τις κατατάξουμε στο πεδίο 0-255.

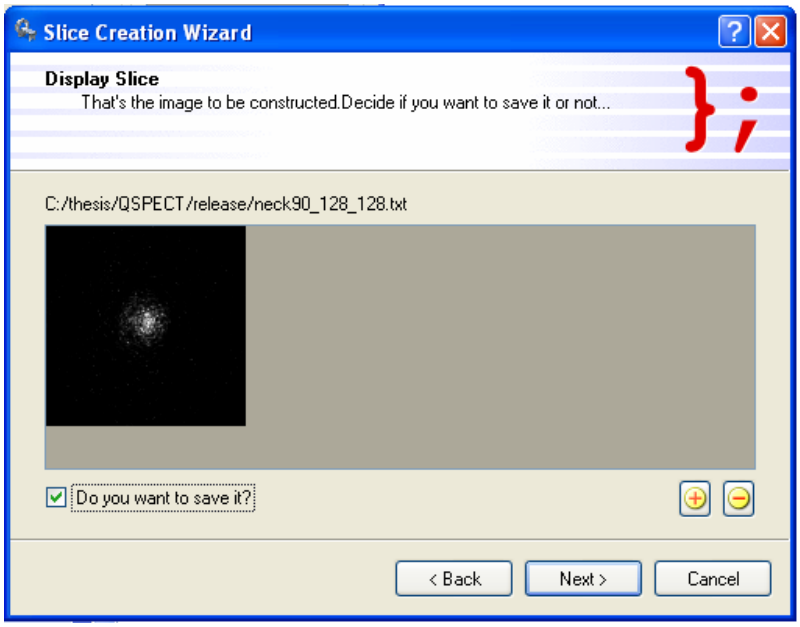
Οι παραπάνω αναγκαίες ενέργειες γίνονται μέσα από έναν κατάλληλο οδηγό που βοηθάει το χρήστη να μετατρέψει ένα αρχείο κειμένου με τις τιμές των pixel της ανακατασκευασμένης τομής σε αρχείο εικόνας .bmp. Ο οδηγός αυτός, PicWizard, λειτουργεί ανάλογα με τους 2 προηγούμενους οδηγούς που αναλύσαμε. Ζητά από το χρήστη τη διεύθυνση στο δίσκο του αποθηκευμένου αρχείου με την ανακατασκευασμένη τομή, καθώς και τις διαστάσεις της εικόνας που πρόκειται να δημιουργηθεί. Στη συνέχεια δημιουργεί την εικόνα και την προβάλλει και ρωτά τον χρήστη αν θέλει να την αποθηκεύσει ή όχι. Πρακτικά, οι ίδιες ενέργειες γίνονται και από την κλάση DisplayImages που αναφέρθηκε στην προηγούμενη ενότητα.

Ακολουθεί το κομμάτι κώδικα που κάνει τη βαθμονόμηση στις τιμές που διαβάζονται από το αρχείο της ανακατασκευασμένης τομής.

```
QRgb value;
QFile file(rootpath);
if (!file.open(QIODevice::ReadOnly | QIODevice::Text))
      return;
double max = 0;
double min = 0;
QTextStream in(&file);
while (!in.atEnd()) {
      QString line = in.readLine();
      QStringList list1 = line.split(" ");
      for (int w = 0; w < list1.size(); w++)
      {
            if(!list1.at(w).isEmpty())
            {
                  if(list1.at(w).toDouble()<min)
                        min = list1.at(w).toDouble();
                  if(list1.at(w).toDouble()>max)
                        max = list1.at(w).toDouble();
                  process_line<<list1.at(w).toDouble();
            }
      }
}
```

Στον παραπάνω βρόγχο while διαβάζονται και εισάγονται σε μια λίστα οι τιμές από το αρχείο κειμένου και επίσης βρίσκονται η μέγιστη και η ελάχιστη τιμή. Στη συνέχεια στο παρακάτω κομμάτι κώδικα, γίνεται μια βαθμονόμηση όλων των τιμών και αντιστοιχίζονται σε τιμές στο διάστημα [0,255]. Κάθε τιμή πολλαπλασιάζεται με 255 και διαιρείται με την μέγιστη τιμή που έχει βρεθεί προηγουμένως.

```
      int k=0;
    QString status = QString("process_line.size = %1 ")
                        .arg(process_line.size());
      progress = new QProgressDialog("Creating bmp file ...",0 ,0
,process_line.size());
      progress->setWindowTitle("Progress...");
      progress->show();
      while( k<process_line.size()){
            progress->setValue(k);
            qApp->processEvents();
            for(int i=0;i<image.width();i++)
                  for(int j=0;j<image.height();j++)
                  {
                        double v = (process_line.at(k))*255/max;
                        value = qRgb(v, v, v );
                        image.setPixel(i,j,value);
                        k++;
                  }
      }
      progress->close();
```



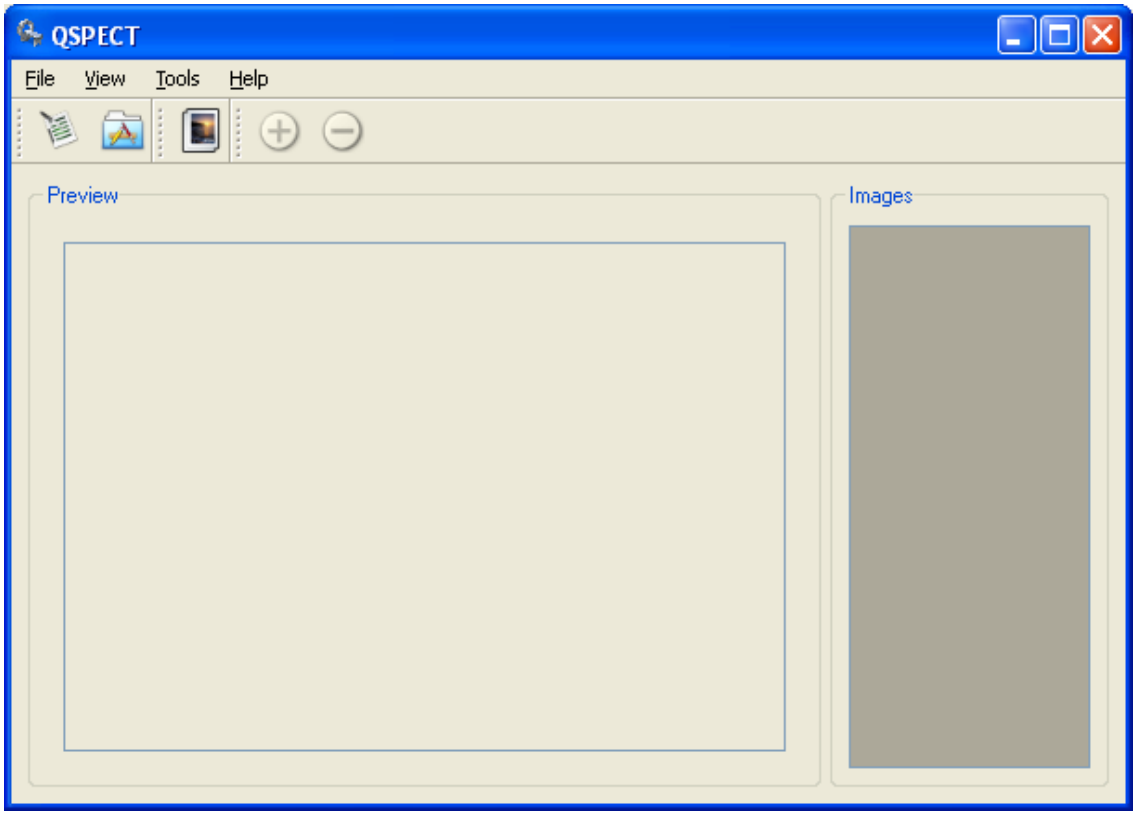

Σχ.3.10:Ο οδηγός PicWizard δημιουργεί την εικόνας μιας τομής από ένα αποθηκευμένο αρχείο της ανακατασκευής της

# ΚΕΦΑΛΑΙΟ 4

# ΠΑΡΑΔΕΙΓΜΑ ΕΚΤΕΛΕΣΗΣ

## 4.1 Εισαγωγή

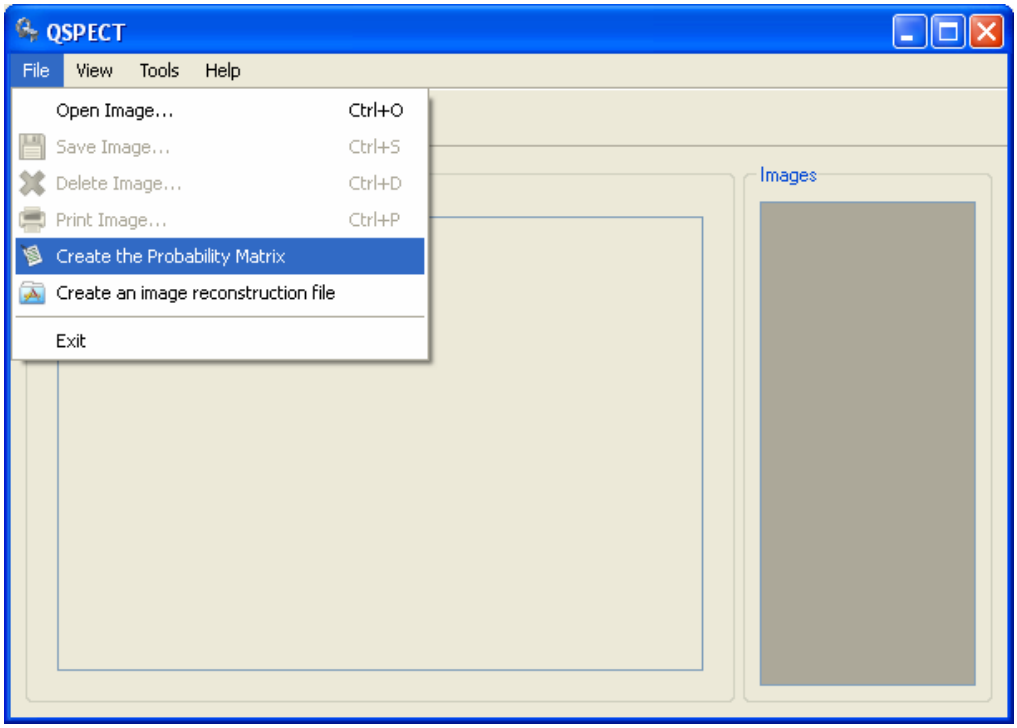
Στο κεφάλαιο αυτό θα παρουσιάσουμε ένα παράδειγμα εκτέλεσης των λειτουργιών της εφαρμογής QSPECT. Οι λειτουργίες που θα παρουσιαστούν είναι οι εξής : 1) Δημιουργία του πίνακα πιθανοτήτων του συστήματος, 2) Δημιουργία της ανακατασκευής μιας τομής και εξαγωγή της είτε σε μορφή αρχείου κειμένου .txt είτε σε μορφή αρχείου εικόνας .bmp, 3) Δημιουργία αρχείου εικόνας .bmp από αντίστοιχο αρχείο κειμένου .txt ανακατασκευασμένης τομής, 4) Φόρτωμα και προβολή εικόνων.



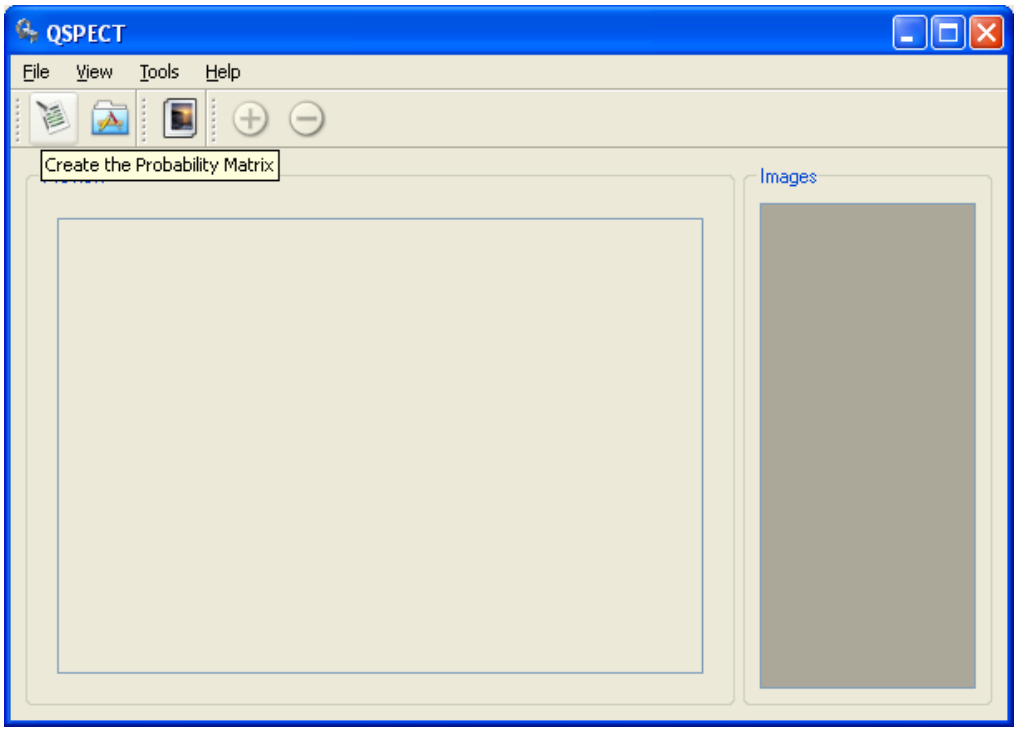

Εικόνα 4.1 : Κεντρικό μενού της εφαρμογής QSPECT σε Windows XP.

### 4.1.1 Δημιουργία του πίνακα πιθανοτήτων του συστήματος

Μπορούμε να ξεκινήσουμε αυτή τη διαδικασία με 2 τρόπους, από το μενού File επιλογή Create the Probability Matrix ή πατώντας το πρώτο κουμπί από την μπάρα κουμπιών (βλ. τις 2 επόμενες εικόνες).

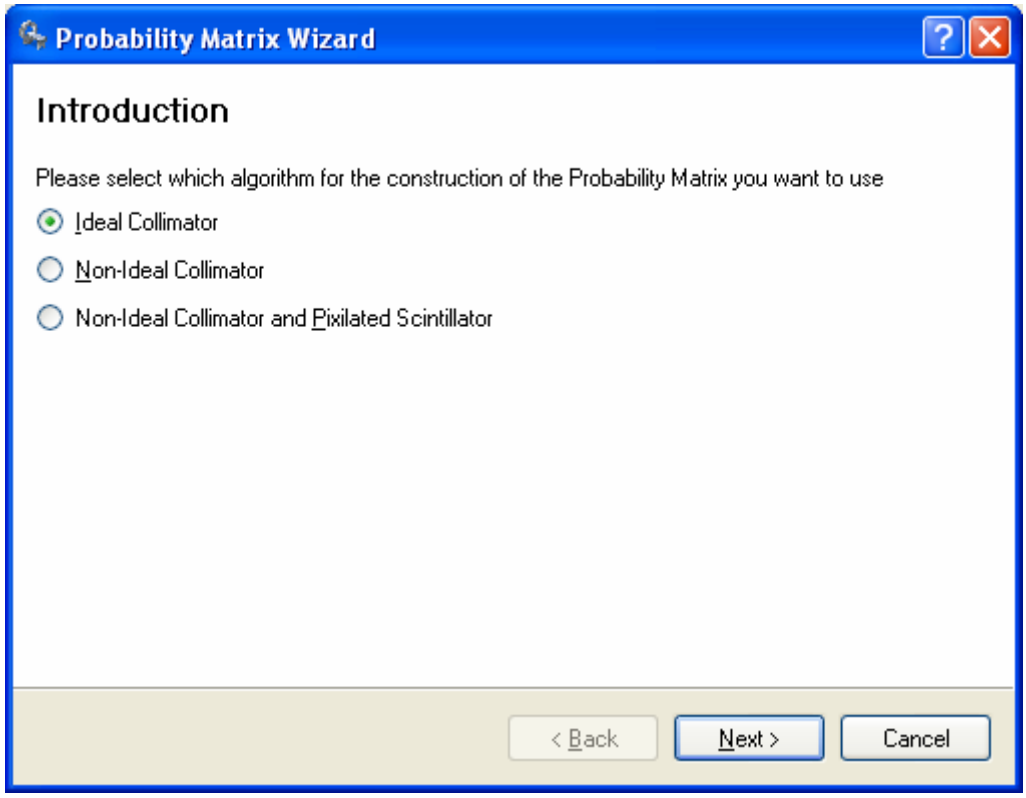



Εικόνα 4.2



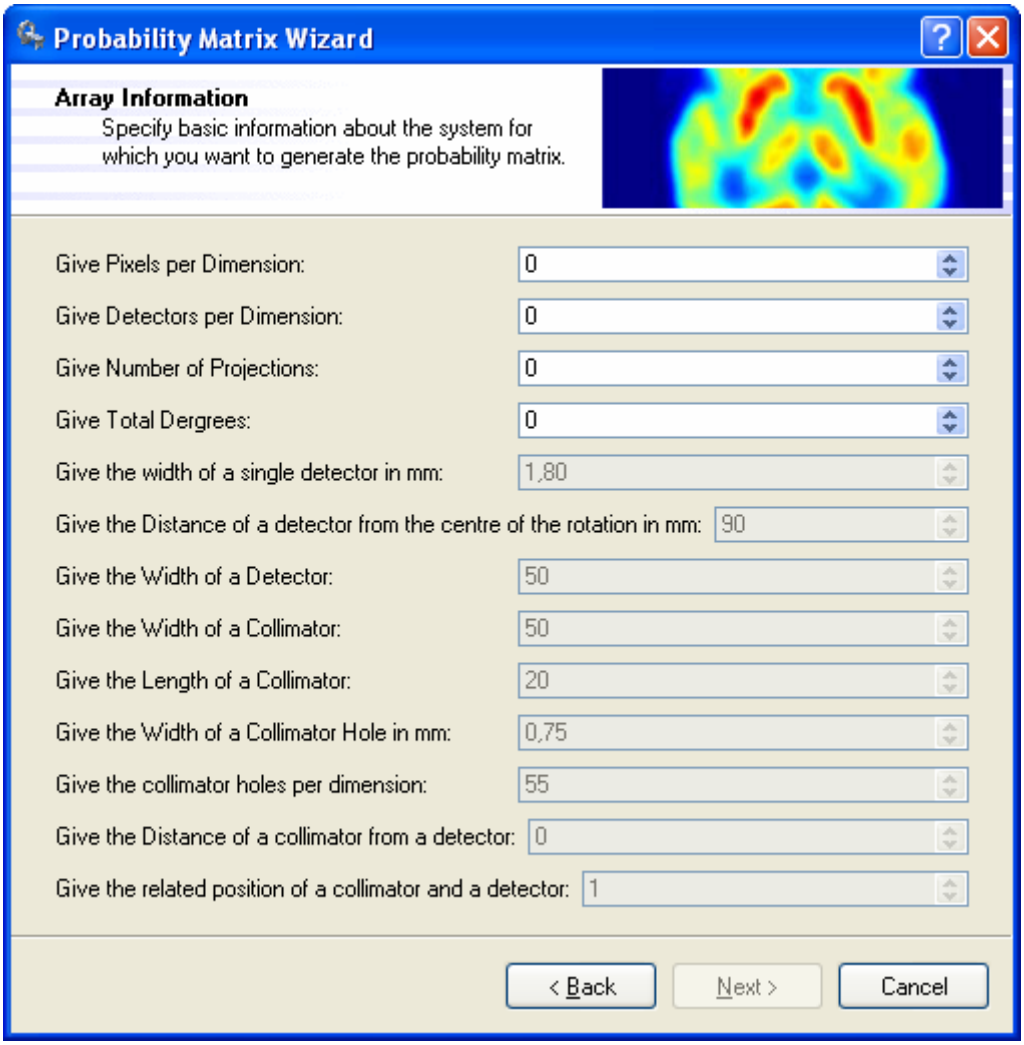

Εικόνα 4.3

Αφού κάνουμε μια από αυτές τις δυο επιλογές, ξεκινά ένας οδηγός (wizard), που βοηθά το χρήστη να κατασκευάσει σε ένα αρχείο κειμένου μορφής .txt το πίνακα πιθανοτήτων του συστήματός του. Στο πρώτο βήμα του wizard επιλέγεται ένας από τους 3 δυνατούς αλγορίθμους κατασκευής του πίνακα συστήματος, όπως αυτοί περιγράφονται στο κεφάλαιο 3.



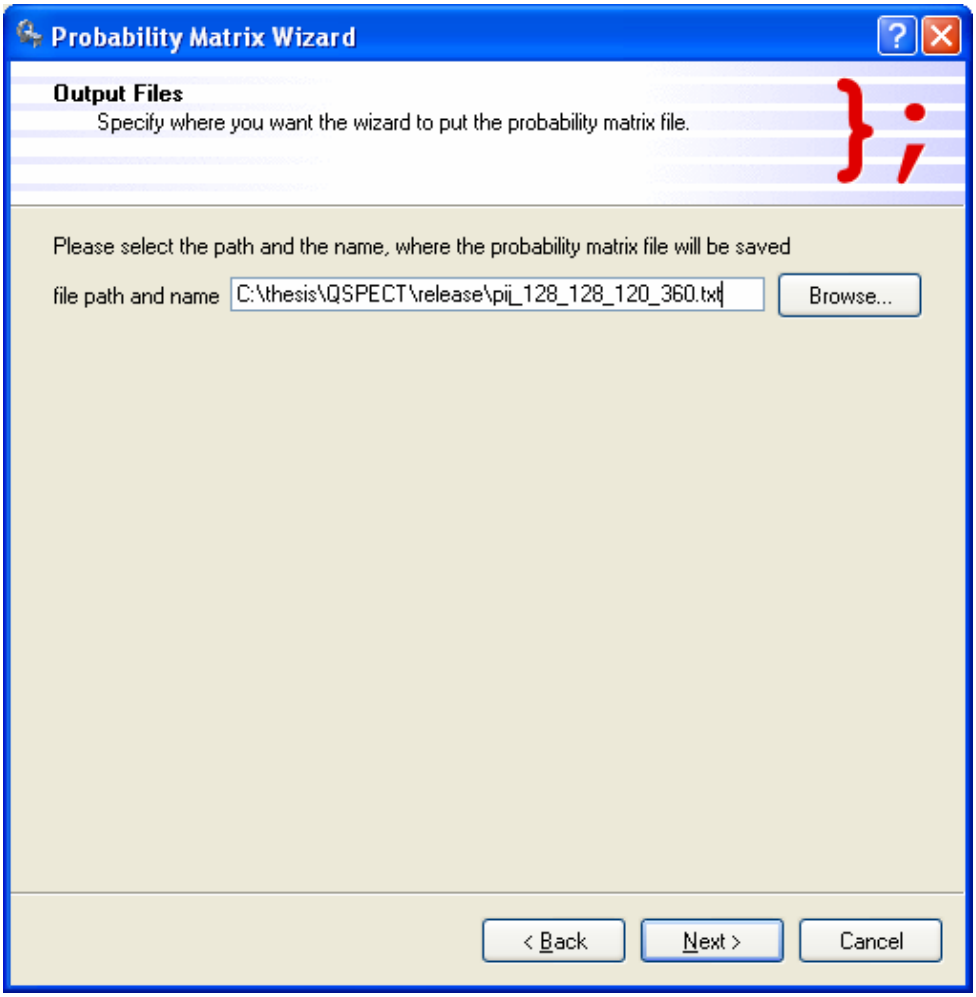

Εικόνα 4.4 : Επιλογή του αλγορίθμου κατασκευής του πίνακα πιθανοτήτων του συστήματος.

Στο δεύτερο βήμα ο χρήστης δίνει τις απαιτούμενες πληροφορίες (π.χ. πλήθος Pixels ανά διάσταση, αριθμός προβολών κ.α.) σχετικά με το σύστημά του και για το οποίο θα κατασκευαστεί ο πίνακας πιθανοτήτων. Διαφορετικές πληροφορίες είναι απαραίτητες για διαφορετικούς αλγόριθμους και σε μερικές από αυτές υπάρχουν κάποιες συνήθεις τιμές προεπιλεγμένες. Το κουμπί Next ενεργοποιείται μόνο αφού ο χρήστης έχει εισάγει όλα τα απαραίτητα δεδομένα.



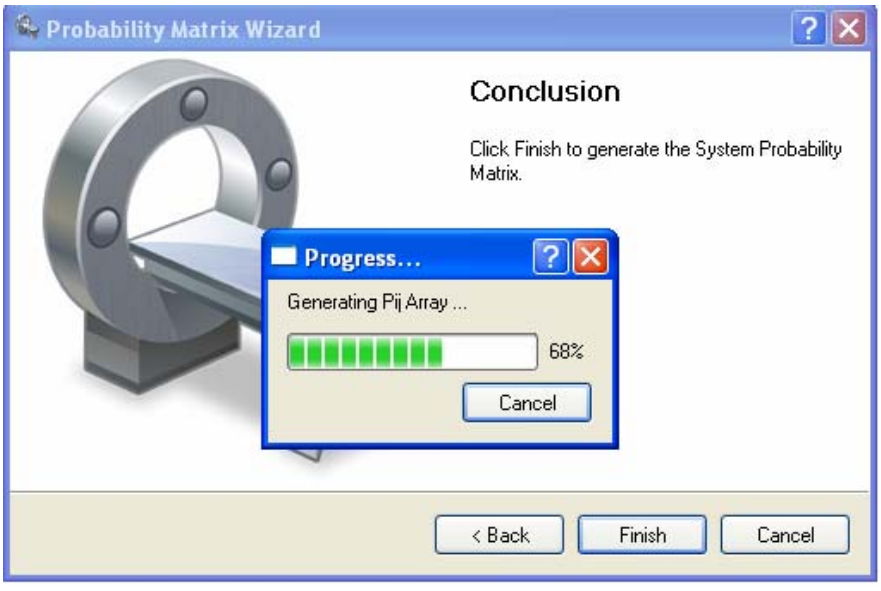
Εικόνα 4.5 : Εισαγωγή των απαραίτητων πληροφοριών για την κατασκευή του πίνακα συστήματος από το χρήστη.

Μετά την εισαγωγή των δεδομένων του συστήματος, ο χρήστης καλείται να επιλέξει ένα όνομα με το οποίο θα αποθηκευτεί σε ένα αρχείο κειμένου μορφής (txt) ο πίνακας συστήματος. Το προεπιλεγμένο όνομα που δίνεται από το πρόγραμμα είναι της μορφής <pij_αριθμός Pixels ανά διάσταση_αριθμός ανιχνευτών ανά διάσταση_αριθμός προβολών_σύνολο γωνιών> και αποθηκεύεται στο σημείο στο σκληρό δίσκο όπου υπάρχει και το πρόγραμμα QSPECT. Ο χρήστης μπορεί να επιλέξει ένα διαφορετικό όνομα και τοποθεσία αποθήκευσης πατώντας το κουμπί Browse.




Εικόνα 4.6 : Επιλογή της τοποθεσίας και του ονόματος αποθήκευσης του πίνακα πιθανοτήτων.

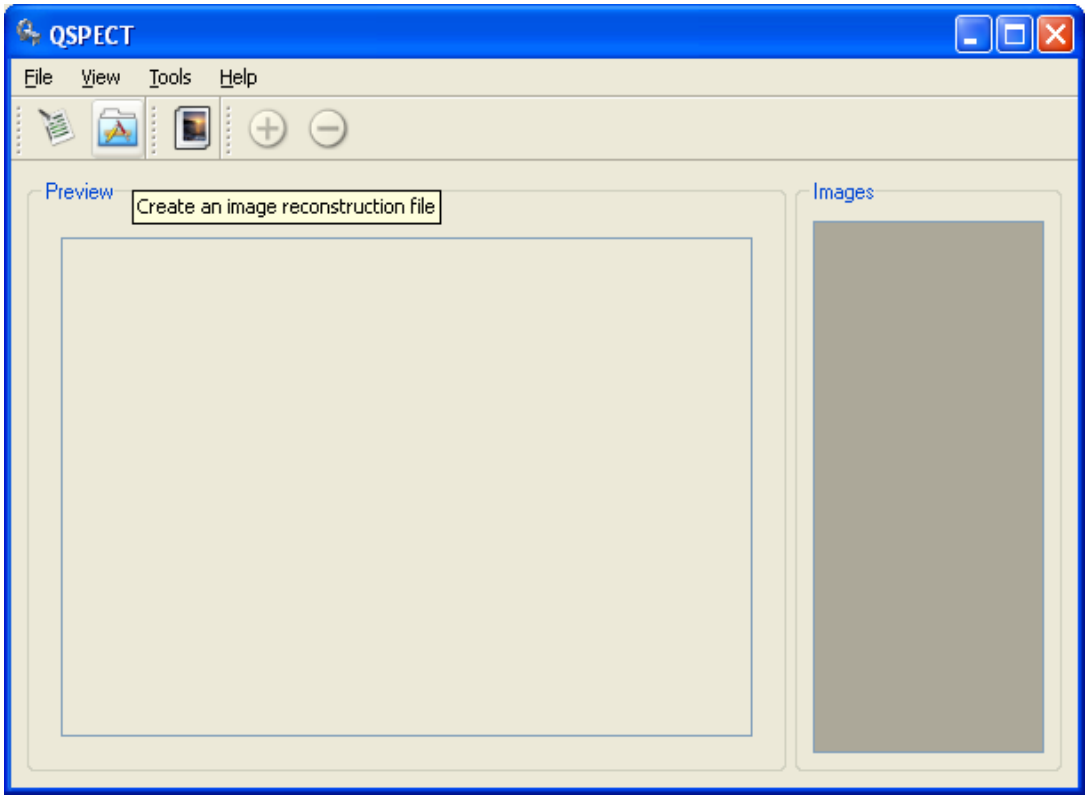
Τέλος, ο χρήστης ενεργοποιεί την κατασκευή του πίνακα πατώντας το κουμπί Finish στο τελευταίο στάδιο του οδηγού. Η πρόοδος της κατασκευής του πίνακα φαίνεται με μια μπάρα προόδου (Progress bar).




Εικόνα 4.7 : Η πρόοδος στην κατασκευή του πίνακα πιθανοτήτων δείχνεται με μια αντίστοιχη μπάρα.

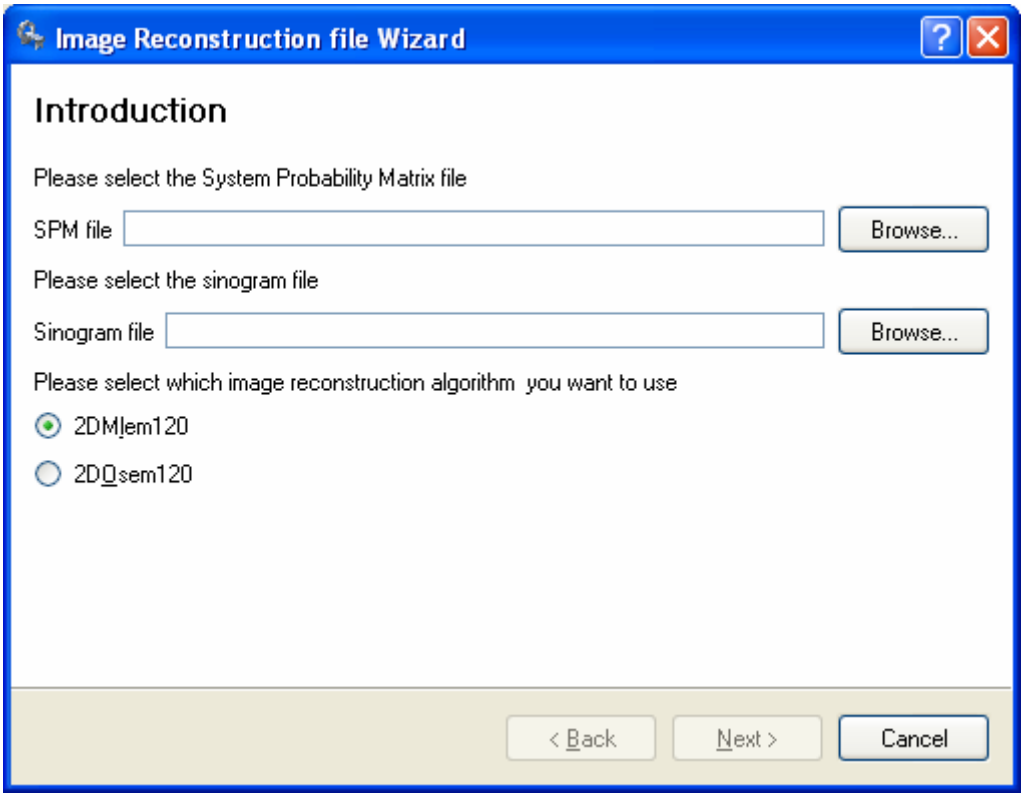
### 4.1.2 Δημιουργία της ανακατασκευής μιας τομής και εξαγωγή της είτε σε μορφή αρχείου κειμένου .txt είτε σε μορφή αρχείου εικόνας .bmp

Όπως και στον οδηγό για την κατασκευή του πίνακα συστήματος έτσι και στον οδηγό για την ανακατασκευή μιας τομής μπορούμε να ξεκινήσουμε την διαδικασία είτε από το μενού File->Create an Image Reconstruction File είτε από το δεύτερο κουμπί στην μπάρα κουμπιών.



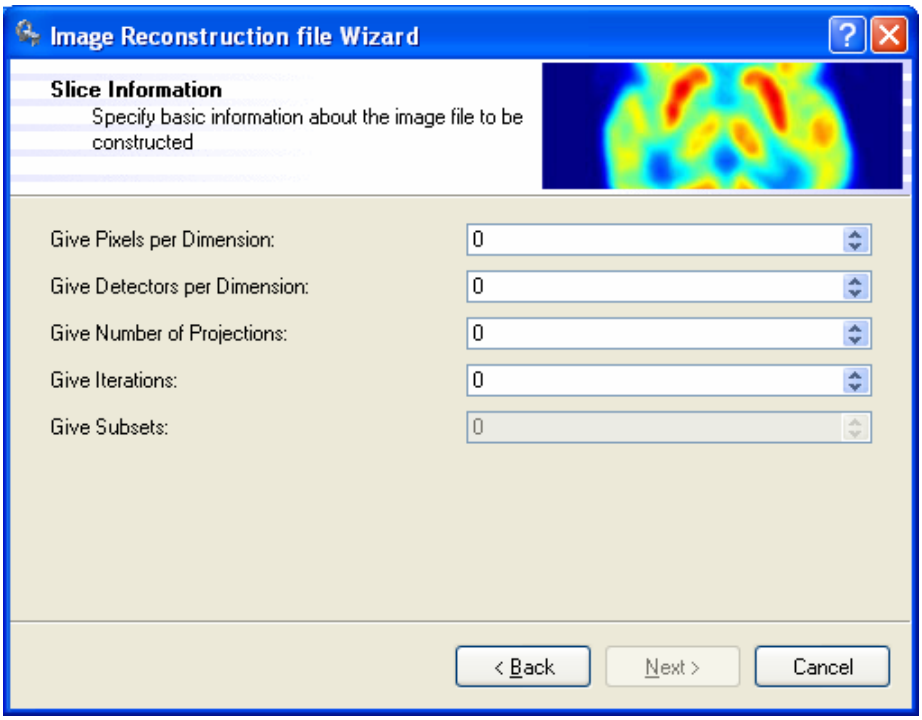

Εικόνα 4.8 : Επιλογή «Δημιουργία του αρχείου ανακατασκευής μιας τομής».

Ξεκινώντας τον οδηγό ο χρήστης καλείται να δώσει το αρχείο όπου περιέχεται ο πίνακας πιθανοτήτων συστήματος και ο οποίος θα χρησιμοποιηθεί για την ανακατασκευή της τομής καθώς επίσης και το αρχείο με τα δεδομένα από την τομογραφία SPECT σε μορφή ημιτονογράμματος. Τα αρχεία αυτά πρέπει να είναι σε μορφή αρχείου κειμένου .txt. Στο πρώτο στάδιο του οδηγού επιλέγεται επίσης και ο επαναληπτικός αλγόριθμος ανακατασκευής που θα χρησιμοποιηθεί και θα είναι ένας από τους MLEM και OSEM.



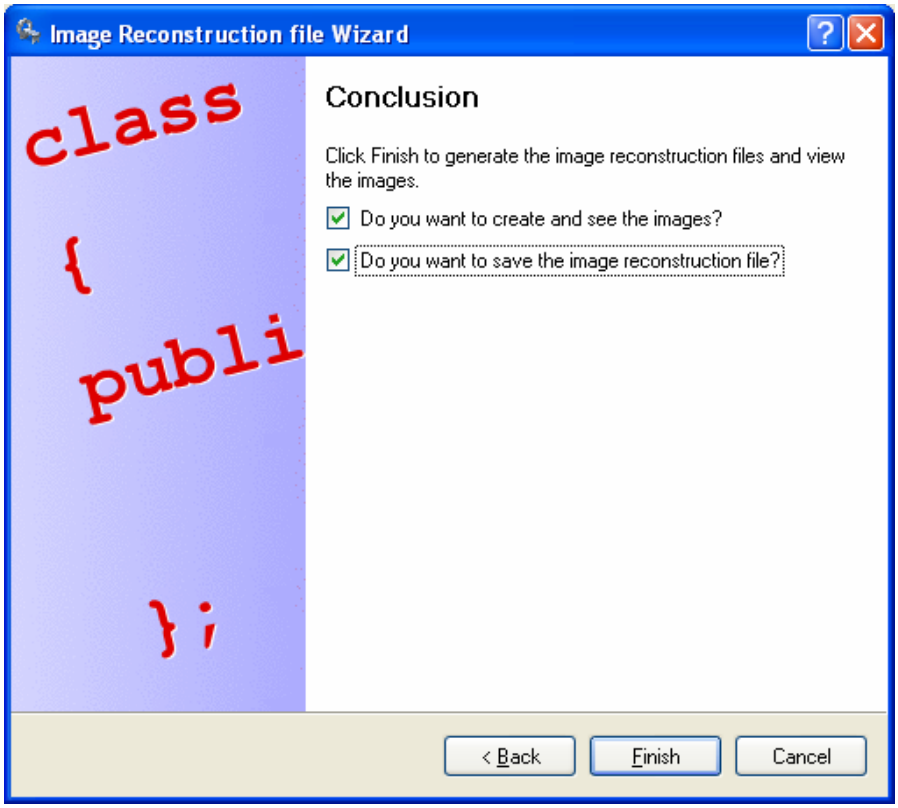

Εικόνα 4.9 : Ο χρήστης εισάγει τα αρχεία για τον πίνακα πιθανοτήτων και το ημιτονόγραμμα της τομής που πρόκειται να ανακατασκευαστεί και επιλέγει με βάση ποιον αλγόριθμο θα γίνει η ανακατασκευή.

Στη συνέχεια ζητούνται το πλήθος των Pixels ανά διάσταση, των ανιχνευτών ανά διάσταση, των προβολών, ο αριθμός των επαναλήψεων που θα πραγματοποιήσει ο επαναληπτικός αλγόριθμος και στην περίπτωση του αλγορίθμου OSEM ο αριθμός των υποσυνόλων. Τα δυο πρώτα στοιχεία πρακτικά αντιστοιχούν και στις διαστάσεις τις εικόνας που θα παραχθεί (π.χ. 128x128). Το κουμπί Next ενεργοποιείται μόνο αν ο χρήστης έχει δώσει όλες τις παραπάνω πληροφορίες.



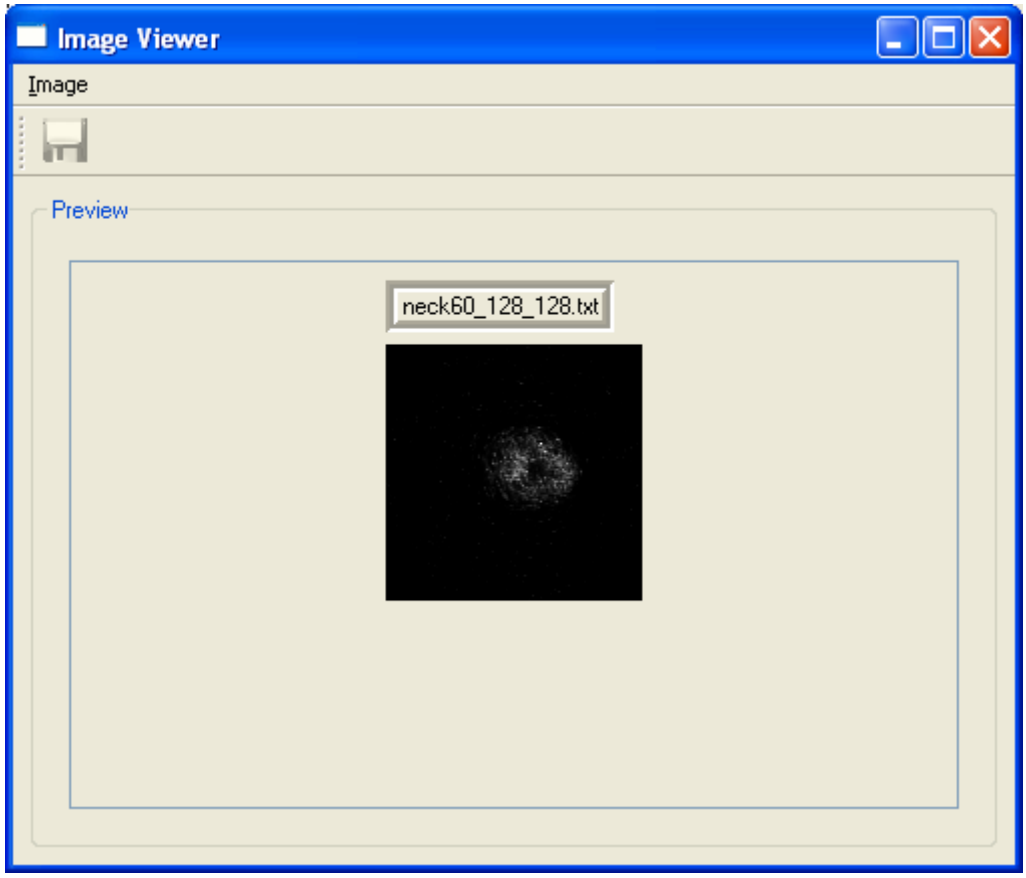

Εικόνα 4.10 : Ζητούνται οι απαραίτητες πληροφορίες ανάλογα με τον αλγόριθμο που έχει επιλεχθεί.

Στο τελευταίο βήμα του οδηγού ο χρήστης ερωτάται αν θέλει να παραχθεί η εικόνα της ανακατασκευασμένης τομής σε μορφή αρχείου εικόνας .bmp και αν θέλει να αποθηκευθεί το αρχείο κειμένου .txt της ανακατασκευής (βλ. εικόνα 4.11).




Εικόνα 4.11

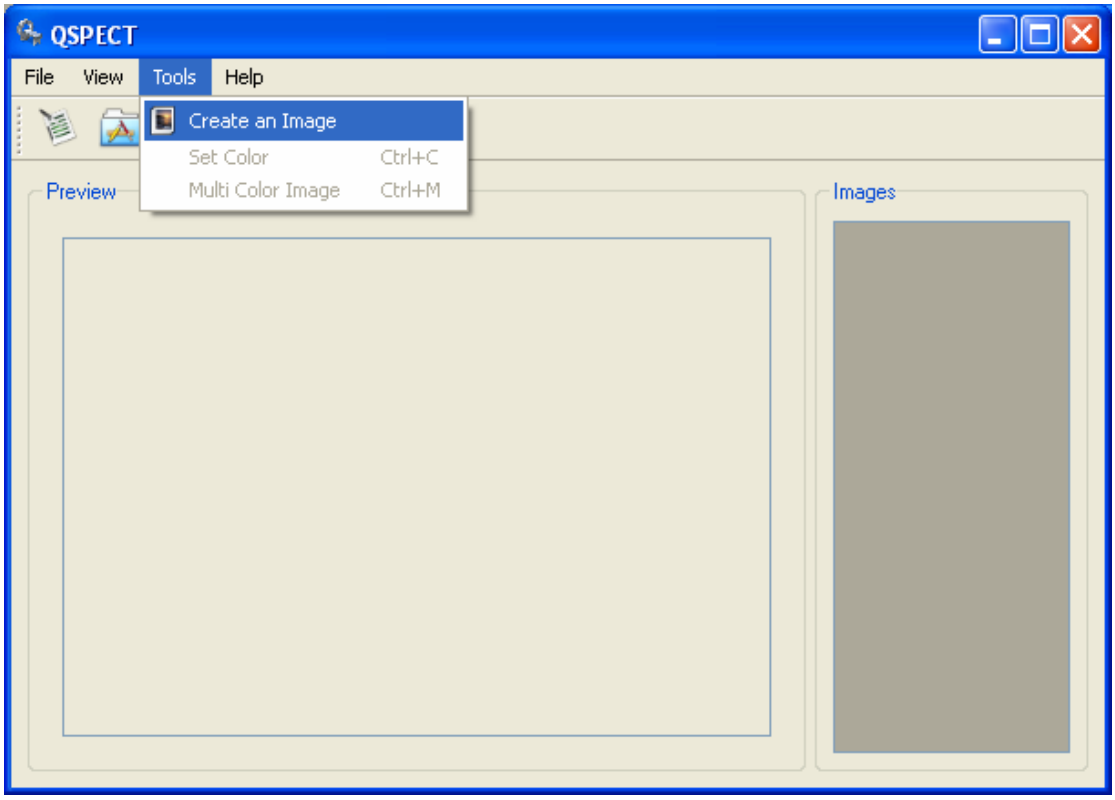
Αν από το τελευταίο βήμα του οδηγού έχει επιλεχθεί η εξαγωγή των αποτελεσμάτων σε αρχείο εικόνας σε μορφή .bmp εμφανίζεται ένα νέο παράθυρο όπου προβάλλεται η τομή και ο χρήστης μπορεί να την επιλέξει με διπλό click και να την αποθηκεύσει ως εικόνα. Επίσης, δίνεται η δυνατότητα για πολλαπλή ανακατασκευή περισσότερων από μια τομές κάθε φορά. Σε αυτήν την περίπτωση οι τομές που θα ανακατασκευαστούν πρέπει να έχουν κοινά χαρακτηριστικά, όπως αυτά ζητούνται στο βήμα 2 του οδηγού. Πρακτικά οι εικόνες που θα δημιουργηθούν θα πρέπει να έχουν όλες το ίδιο μέγεθος.




Εικόνα 4.12: Προβολή μιας τομής μετά την ανακατασκευή της.

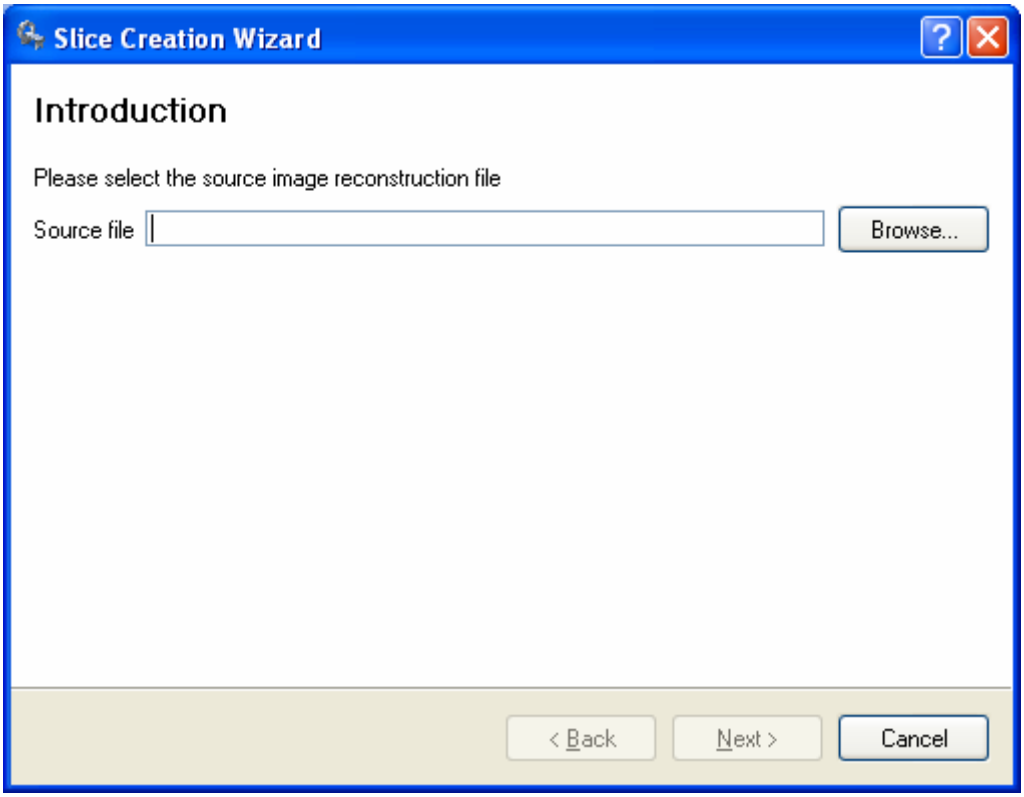
### 4.1.3 Δημιουργία αρχείου εικόνας .bmp από αντίστοιχο αρχείο κειμένου .txt ανακατασκευασμένης τομής

Όταν γίνεται η ανακατασκευή μιας τομής από τα δεδομένα SPECT, το αρχείο της ανακατασκευής μπορεί να αποθηκευθεί σε ένα αρχείο κειμένου μορφής ,txt όπως είδαμε και προηγουμένως. Αυτό το αρχείο κειμένου μπορεί να διαβαστεί κάποια άλλη στιγμή από το πρόγραμμα και να παραχθεί η εικόνα της τομής σε μορφή .bmp. Η διαδικασία αυτή ξεκινά από την επιλογή Tools->Create an Image ή πατώντας το τρίτο κουμπί στην μπάρα κουμπιών.

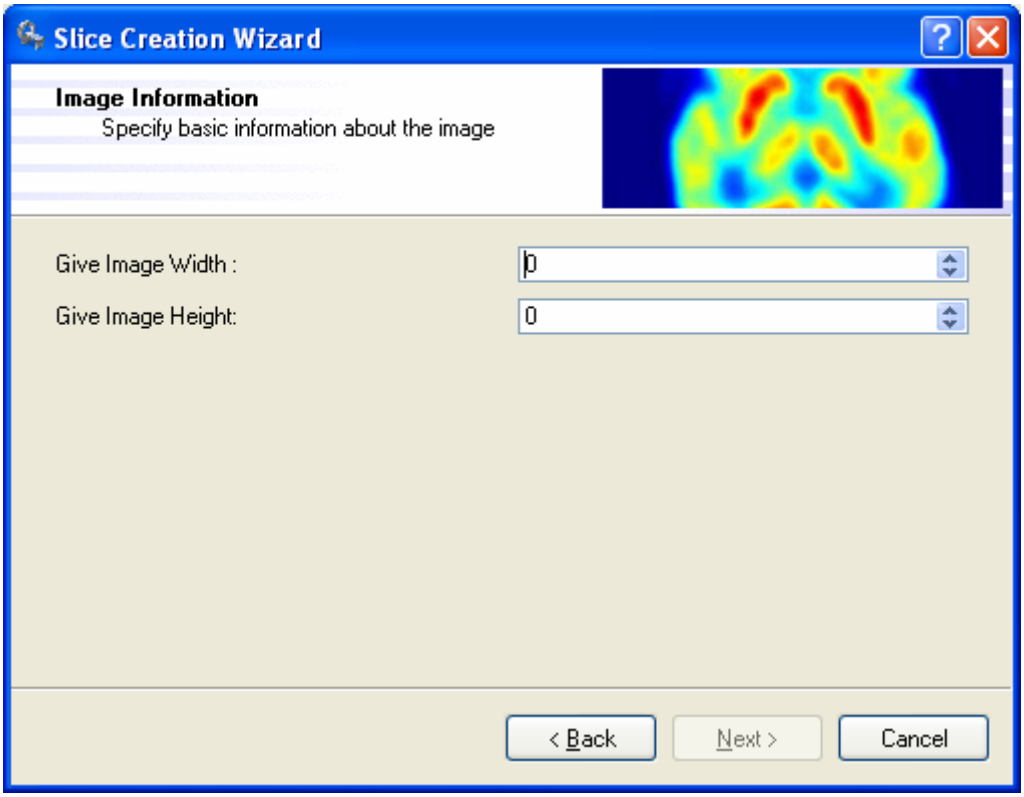



Εικόνα 4.13: Επιλογή «Δημιουργία εικόνας»

Χρησιμοποιείται και σε αυτήν την περίπτωση ένας απλός οδηγός, όπου αρχικά ο χρήστης εισάγει το αρχείο κειμένου από όπου θα διαβαστεί η ανακατασκευασμένη τομή. Στη συνέχεια ζητούνται οι διαστάσεις της προς δημιουργία εικόνας, οι οποίες πρακτικά συμπίπτουν με τα χαρακτηριστικά Pixels per Dimension, Detectors per Dimension , που δώσαμε κατά την ανακατασκευή της τομής από τα δεδομένα SPECT.

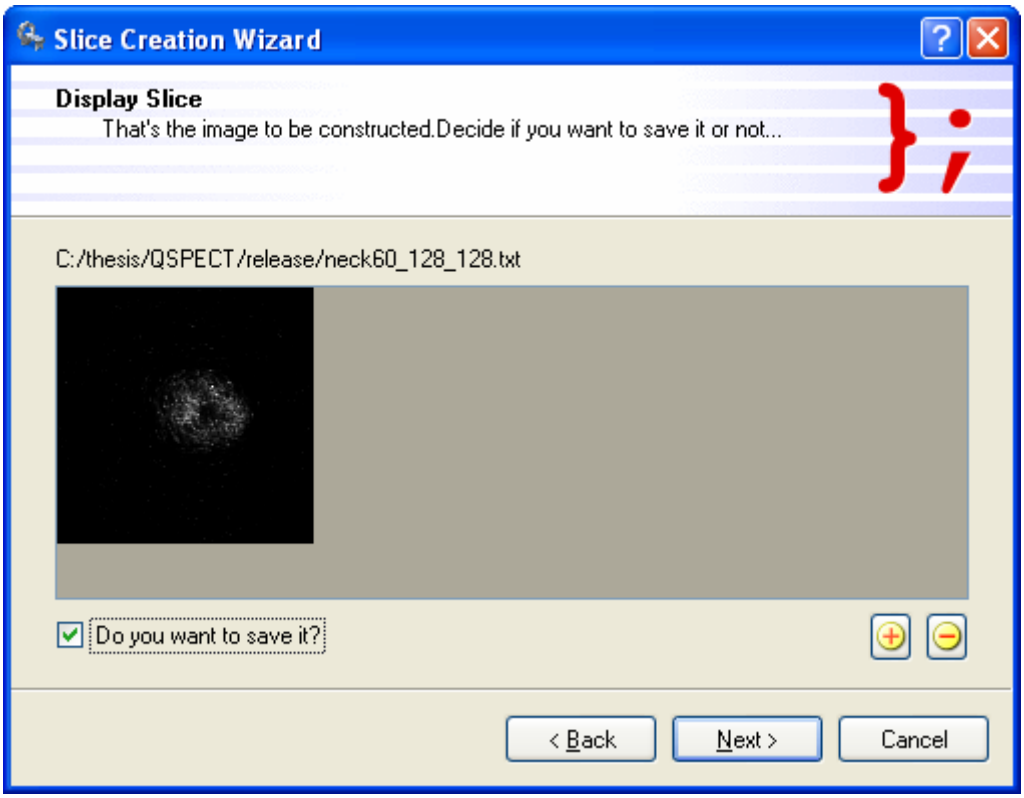



Εικόνα 4.14: Εισαγωγή του αρχείου κειμένου της ανακατασκευασμένης τομής.



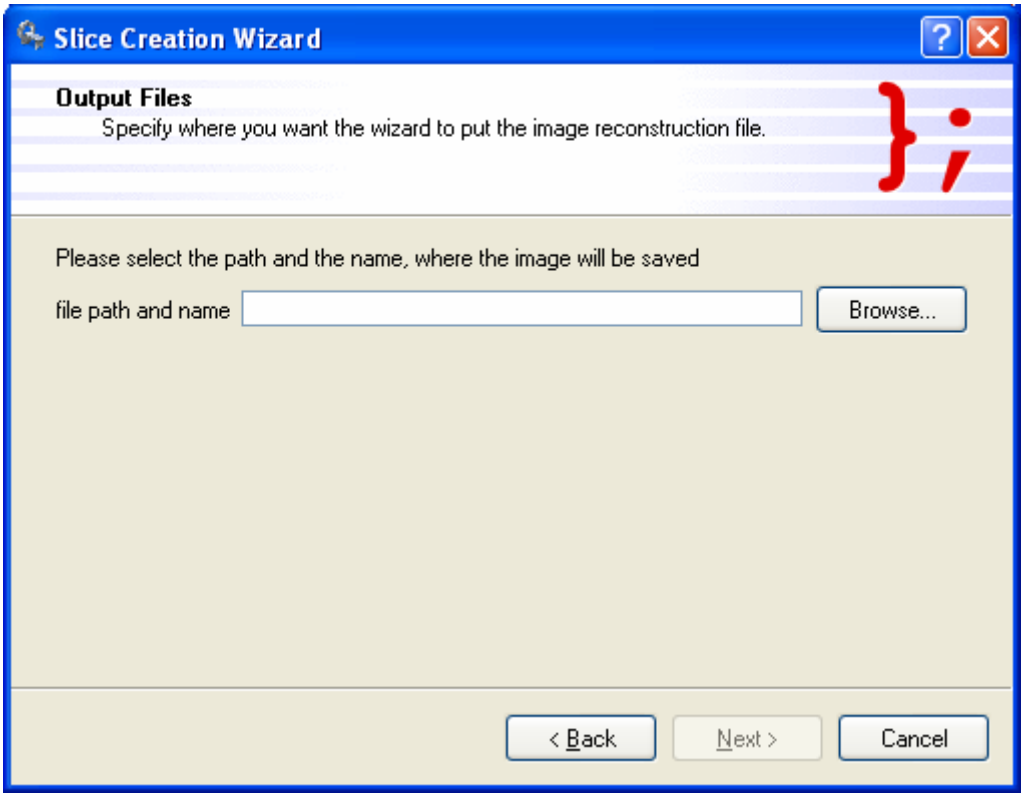

Εικόνα 4.15: Ζητούνται οι διαστάσεις τις προς δημιουργίας εικόνας.

Μετά τη δημιουργία της εικόνας, αυτή προβάλλεται και ο χρήστης μπορεί να κάνει zoom in ή zoom out και να αποφασίσει αν θέλει να την αποθηκεύσει ή όχι.

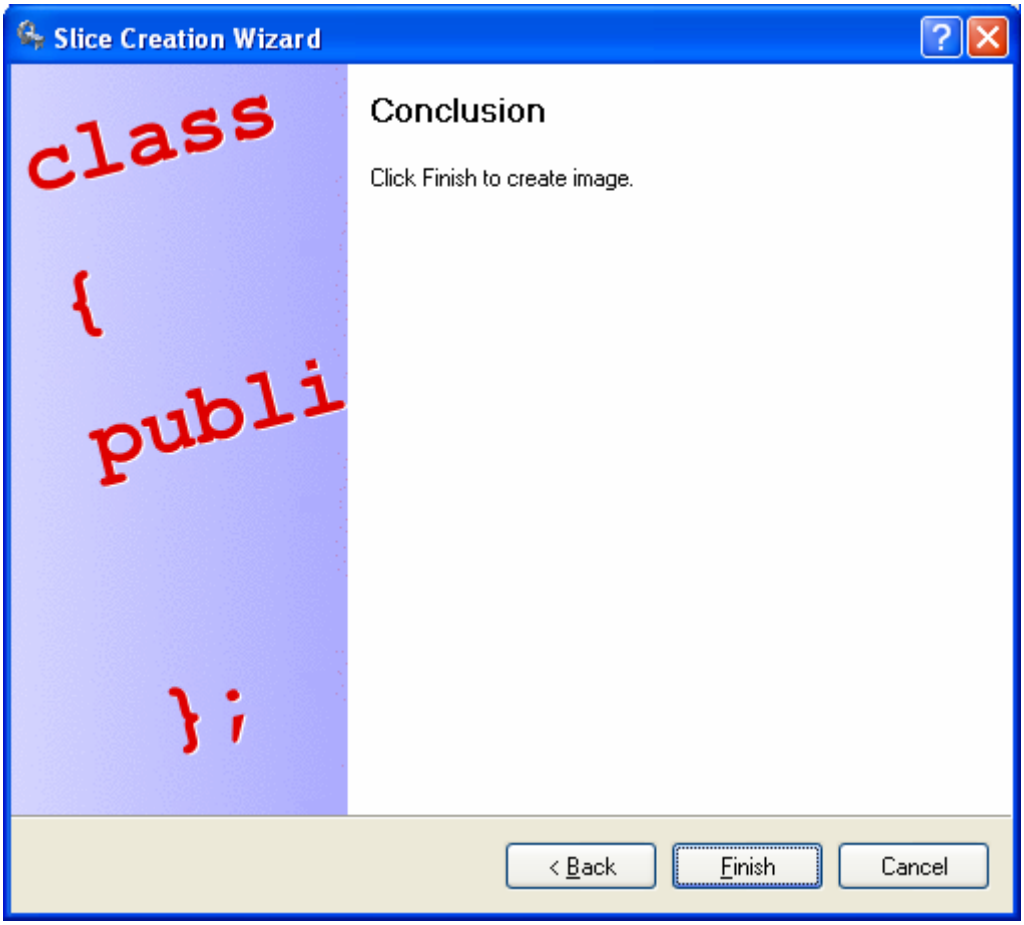



Εικόνα 4.16: Προβολή της εικόνας, δυνατότητα για zoom in, zoom out. Ο χρήστης αποφασίζει αν θέλει να την αποθηκεύσει ή όχι.

Αν επιλεγεί η αποθήκευση της εικόνας τότε η επόμενη σελίδα του οδηγού φαίνεται στην εικόνα 4.16 και ο χρήστης καλείται να δώσει ένα όνομα και μια τοποθεσία αποθήκευσης. Αν ο χρήστης δεν επιθυμεί την αποθήκευση της εικόνας, τότε ο οδηγός προβάλλει την σελίδα τέλους όπως φαίνεται στην εικόνα 4.17.



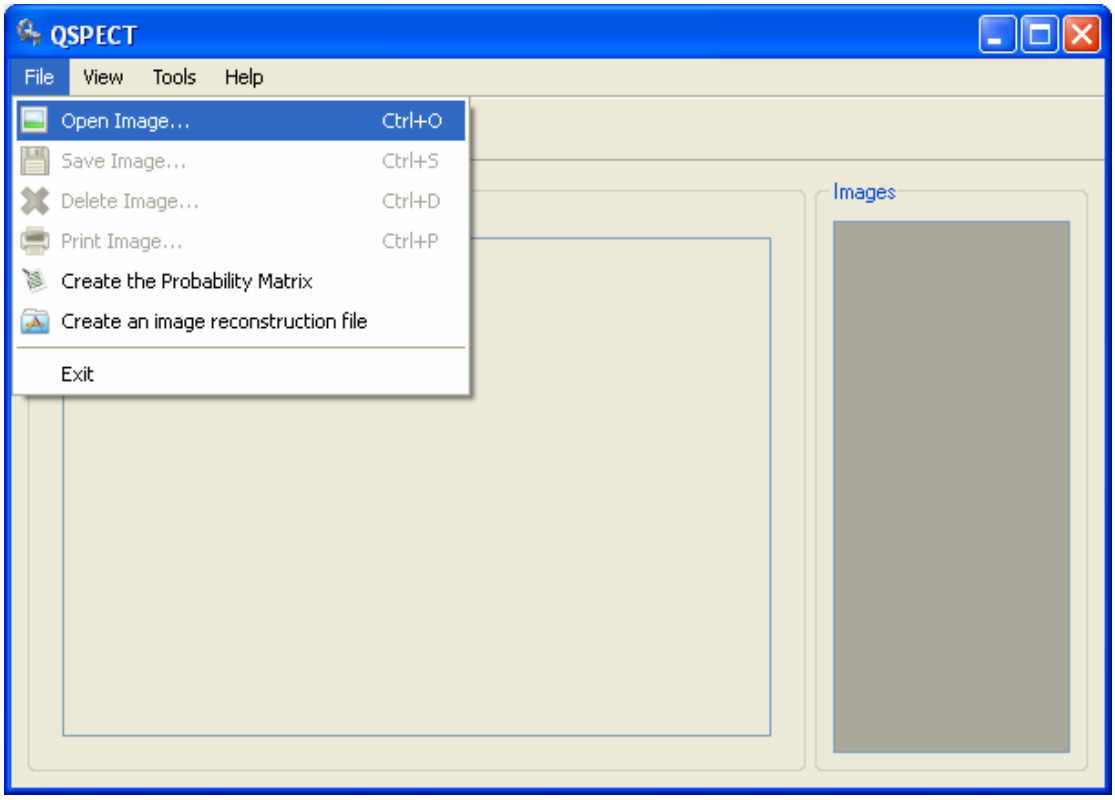

Εικόνα 4.17: Εισαγωγή ονόματος και τοποθεσίας αποθήκευσης.



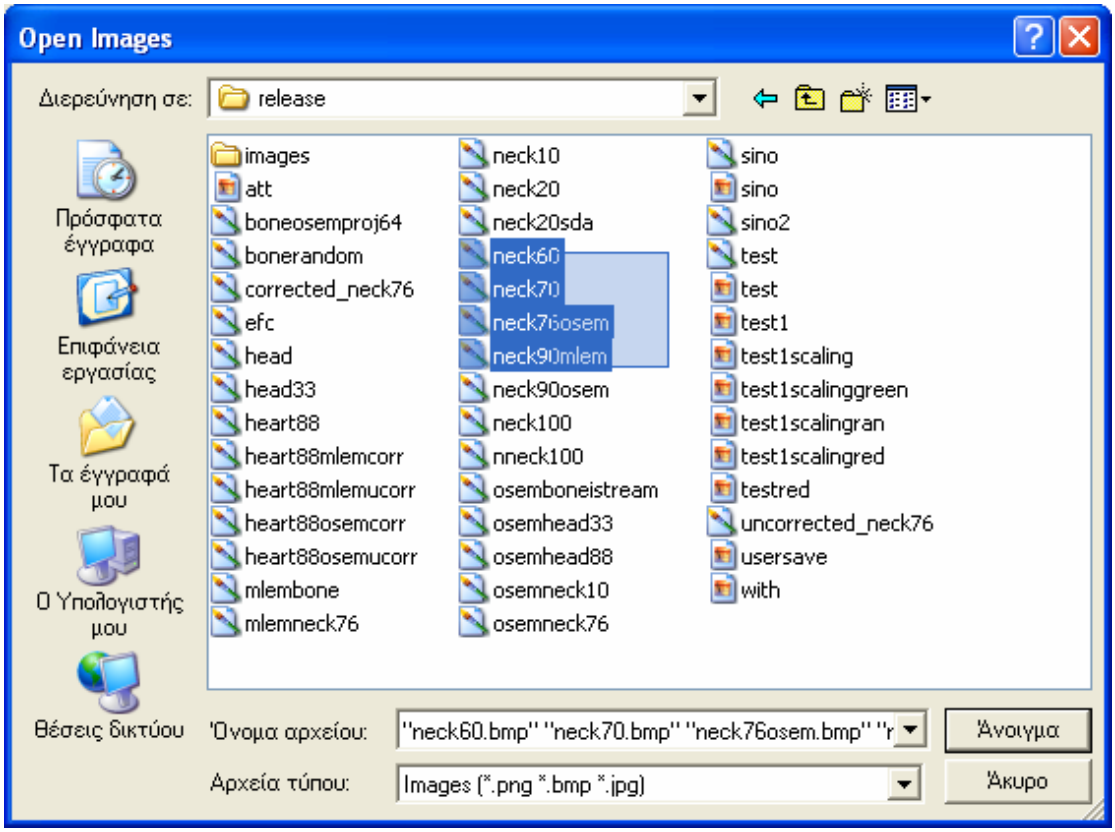

Εικόνα 4.18: Σελίδα τέλους του οδηγού.

### 4.1.4 Φόρτωση και προβολή εικόνων

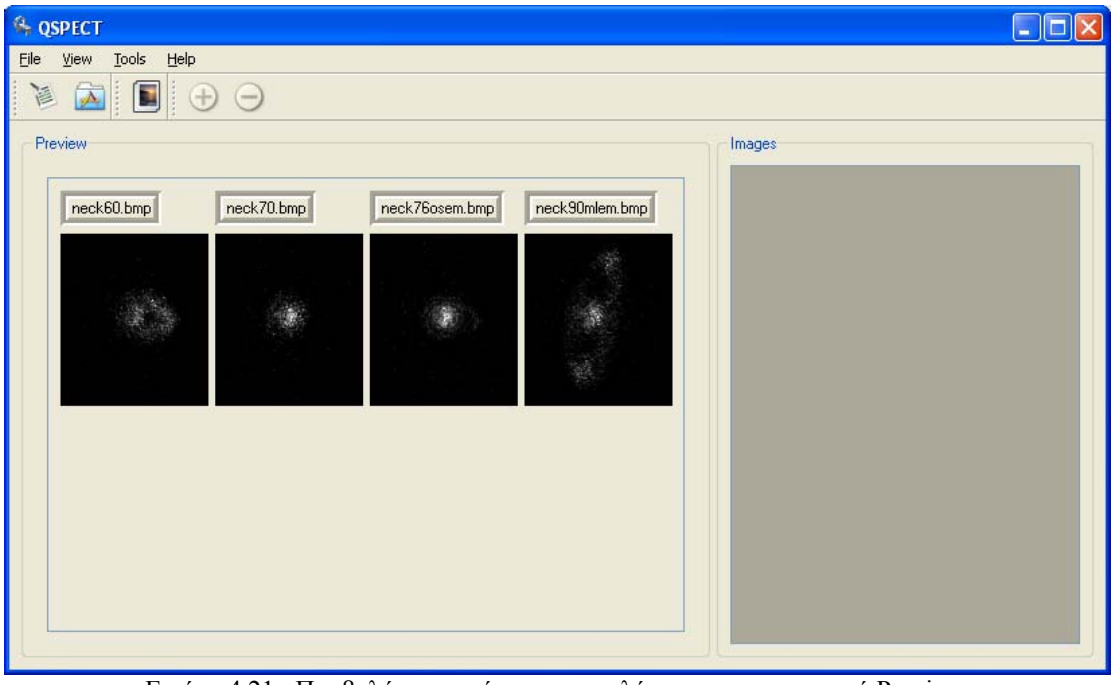
Στην κύρια περιοχή της εφαρμογής QSPECT μπορούν να φορτωθούν και να προβληθούν αρχεία εικόνων τύπου .png, .bmp, .jpg . Ο χρήστης μπορεί να ανοίξει μια εικόνα ή πολλές μαζί από την επιλογή File->Open Image… . Στη συνέχεια, οι εικόνες προβάλλονται στην περιοχή με τον τίτλο Preview και με το όνομα τους πάνω από αυτές. Οι εικόνες αυτές έχουν μέγιστο μέγεθος 140x140. Οι ενέργειες αυτές φαίνονται στις εικόνες 4.18, 4.19, 4.20.




Εικόνα 4.19: Άνοιγμα και προβολή εικόνας

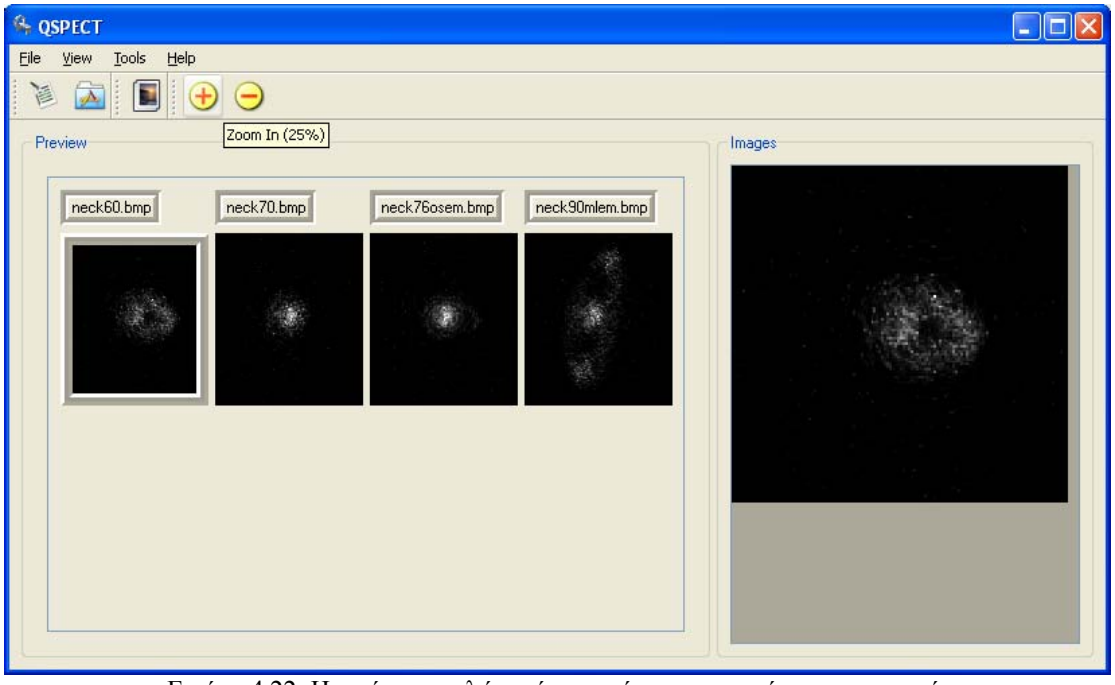



Εικόνα 4.20 : Μπορούν να ανοιχτούν περισσότερες από μια εικόνες.

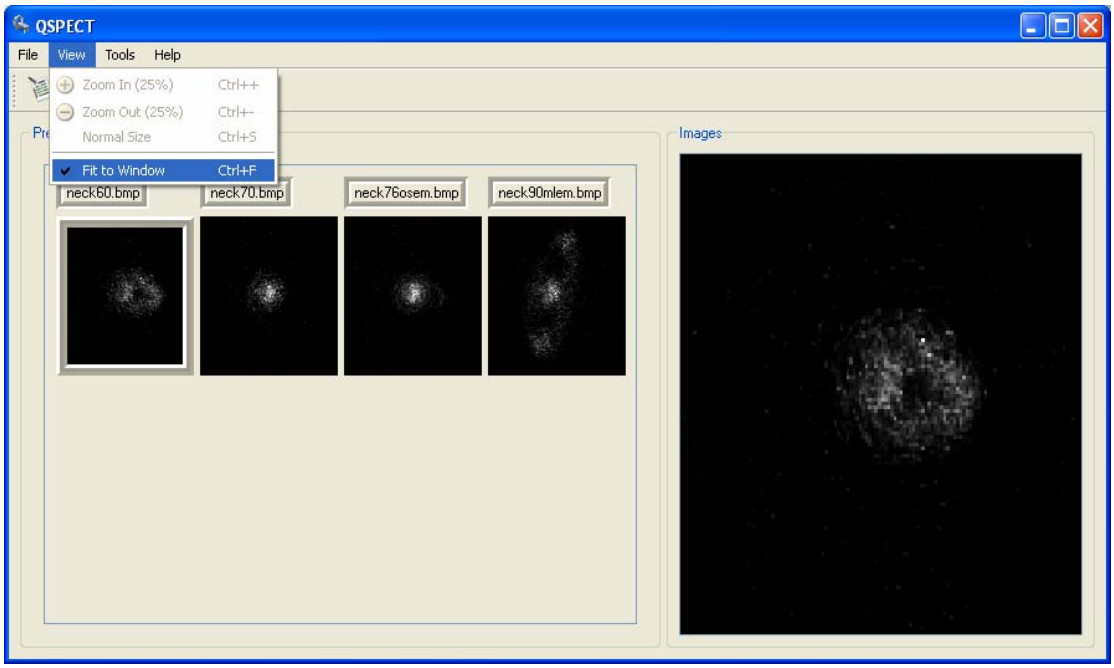



Εικόνα 4.21 : Προβολή των εικόνων που επιλέχτηκαν στην περιοχή Preview.

Κάνοντας διπλό αριστερό click σε κάποια από τις εικόνες που έχουμε ανοίξει, το πλαίσιο της εικόνας αυτής γίνεται πιο έντονο και η εικόνα φορτώνεται στην περιοχή Images. Μπορούμε μεταβάλλουμε τον χώρο που πιάνουν οι περιοχές Preview και Images μετακινώντας το κάθετο διαχωριστικό τους δεξιά ή αριστερά. Από την στιγμή που μια εικόνα θα φορτωθεί στην περιοχή Images, είναι δυνατές κάποιες επιπλέον λειτουργίες. Ο χρήστης μπορεί να μεγαλώσει ή να μικρύνει την εικόνα (zoom in, zoom out) κάνοντας click στα κουμπιά με τα + ή το – αντίστοιχα ή από τις επιλογές View->Zoom in , View->Zoom out. Στο μενού View υπάρχουν ακόμα οι επιλογές Normal Size και Fit to Window. Η πρώτη επαναφέρει την εικόνα στο αρχικό της μέγεθος και η δεύτερη μεγαλώνει ή μικραίνει τόσο την εικόνα ώστε να καλύπτει ακριβώς όλη την περιοχή Images.

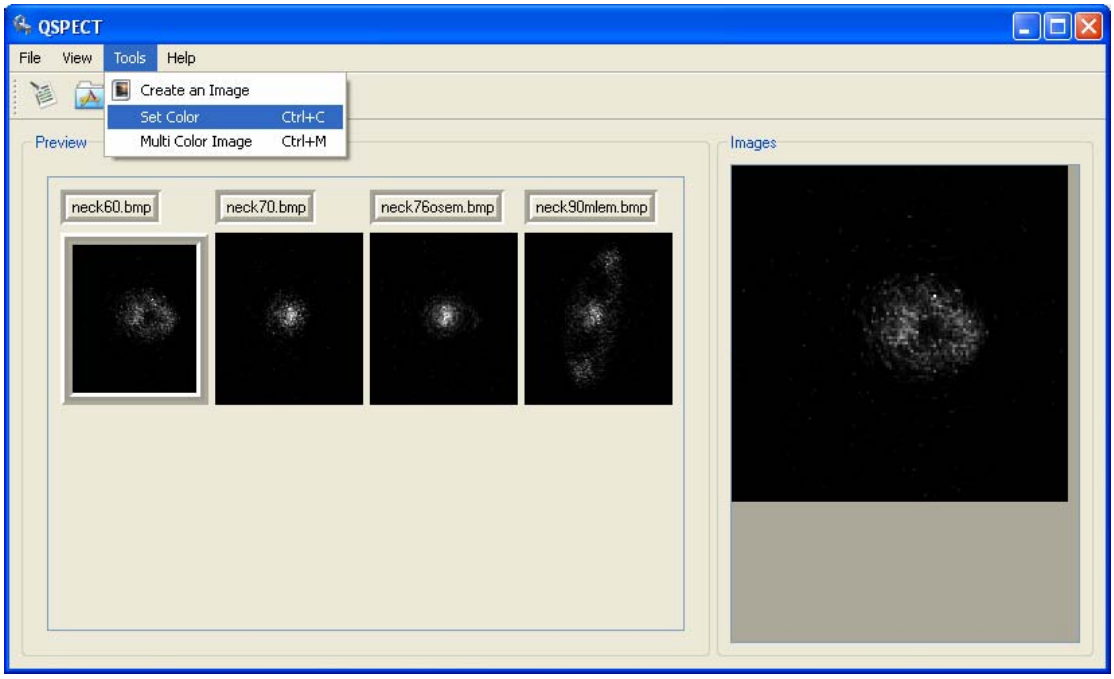



Εικόνα 4.22: Η εικόνα μεγαλώνει ή μικραίνει με τα αντίστοιχα κουμπιά.

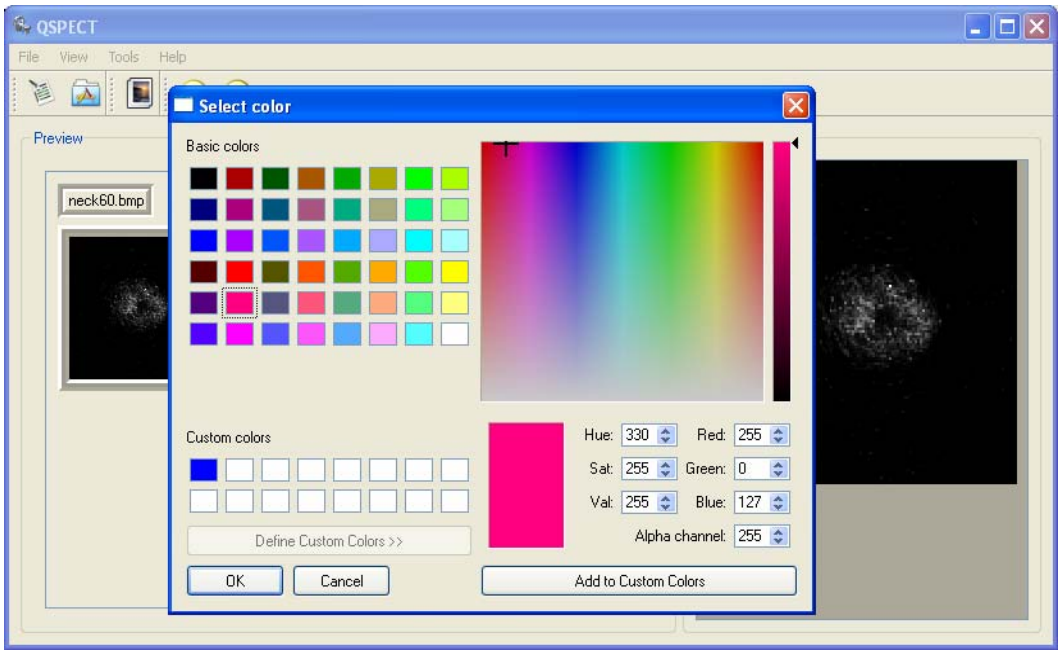



Εικόνα 4.23 : Η εικόνα καλύπτει ακριβώς όλη την περιοχή Images τσεκάροντας την επιλογή Fit to Window.

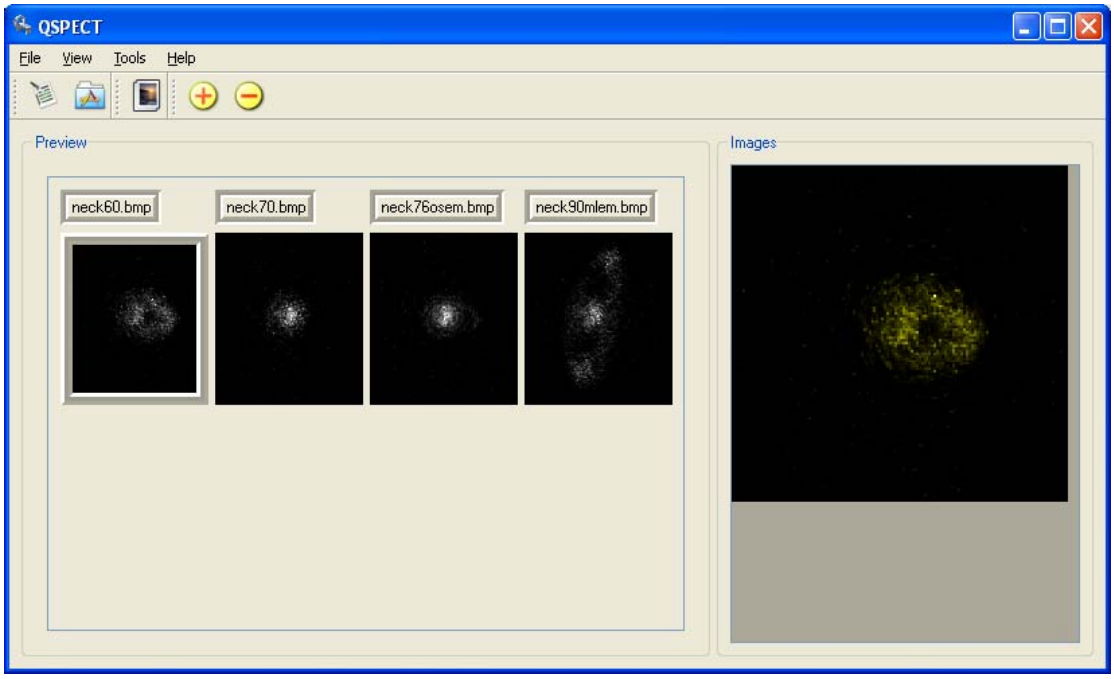
Στο μενού Tools μας δίνονται δυο ακόμη δυνατότητες επεξεργασίας της εικόνας με τις επιλογές Set Color και Multi Color Image. Με την πρώτη επιλογή μπορούμε να δώσουμε μια χρωματική απόχρωση στην εικόνα μας. Πατώντας αυτήν την επιλογή εμφανίζεται ένα παράθυρο όπου μπορούμε να επιλέξουμε το χρώμα της επιθυμητής απόχρωσης (βλ. εικόνες 4.23, 4.24, 4.25) και στη συνέχεια τα Pixels που δεν έχουν τιμή μηδέν (η τιμή μηδέν ισοδυναμεί με μαύρο χρώμα) θα πάρουν τιμές με βάση το χρώμα που επιλέξαμε. Η δεύτερη επιλογή, Multi Color Image, είναι μια προσπάθεια να κάνουμε την εικόνα μιας τομής έγχρωμη. Τα χρώματα που χρησιμοποιούνται είναι δέκα και ξεκινώντας από το μοβ έως το κόκκινο αντιστοιχίζονται σε Pixels με μικρές τιμές έως Pixels με μεγάλες τιμές (βλ. εικόνες 4.26, 4.27) . Όταν γίνει μια από τις δυο παραπάνω αλλαγές σε μια εικόνα, τότε ενεργοποιούνται οι επιλογές File->Save Image… και File->Delete Image… . Τέλος, όταν κάνουμε δυο συνεχόμενες αλλαγές σε μια εικόνα, π.χ. πρώτα της δίνουμε μια χρωματική απόχρωση και μετά την κάνουμε έγχρωμη, η δεύτερη αλλαγή θα γίνει με βάση την αρχική εικόνα. Αν θέλουμε η δεύτερη αλλαγή να ενεργήσει πάνω στην εικόνα που έχει παραχθεί από την πρώτη αλλαγή, τότε θα πρέπει να αποθηκεύσουμε αυτήν την εικόνα και να την ανοίξουμε από το File->Open Image… .



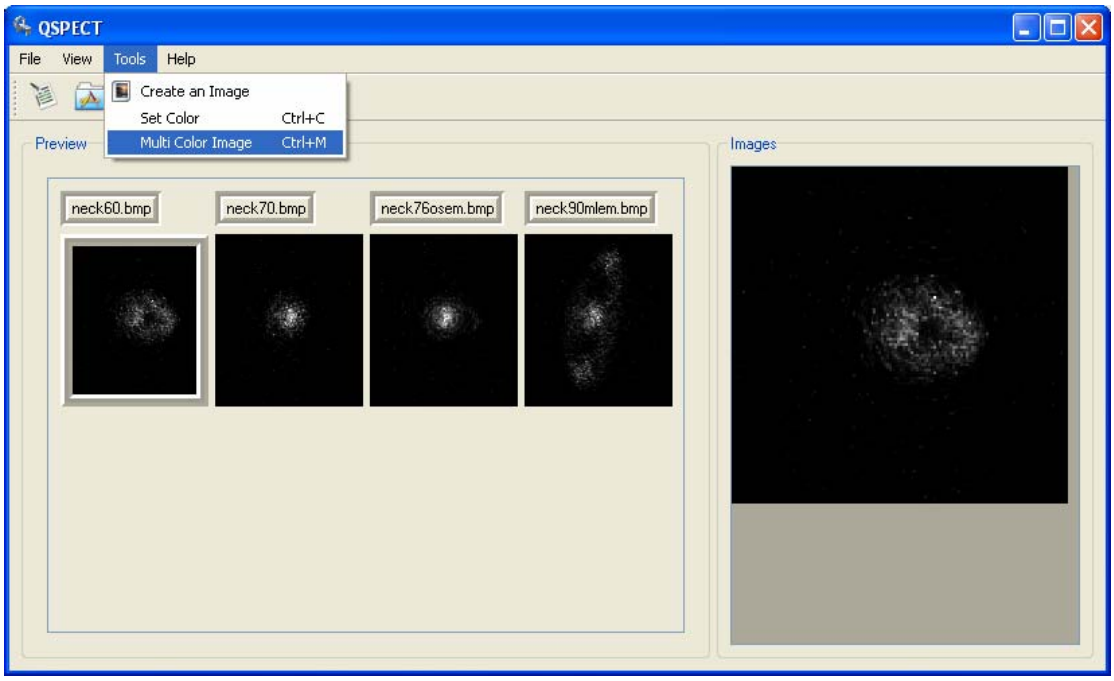

Εικόνα 4.24 : Μπορούμε να δώσουμε μια χρωματική απόχρωση στην επιλεγμένη εικόνα με την επιλογή Tools->Set Color



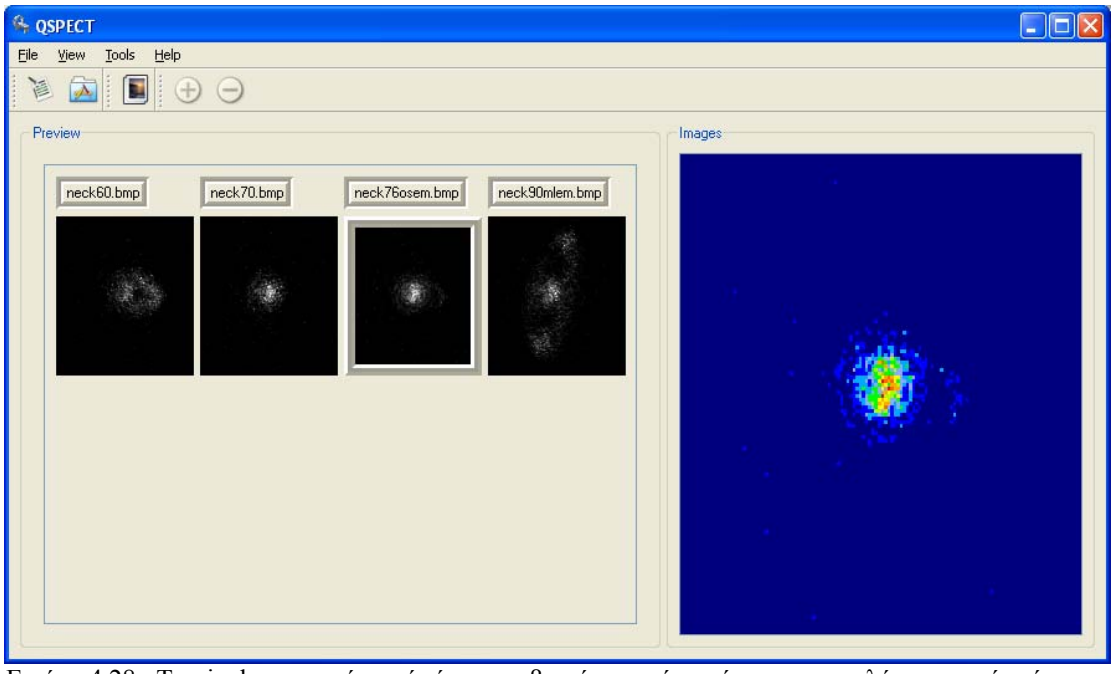

Εικόνα 4.25 : Εμφανίζεται μια παλέτα χρωμάτων για να επιλέξουμε το επιθυμητό χρώμα.



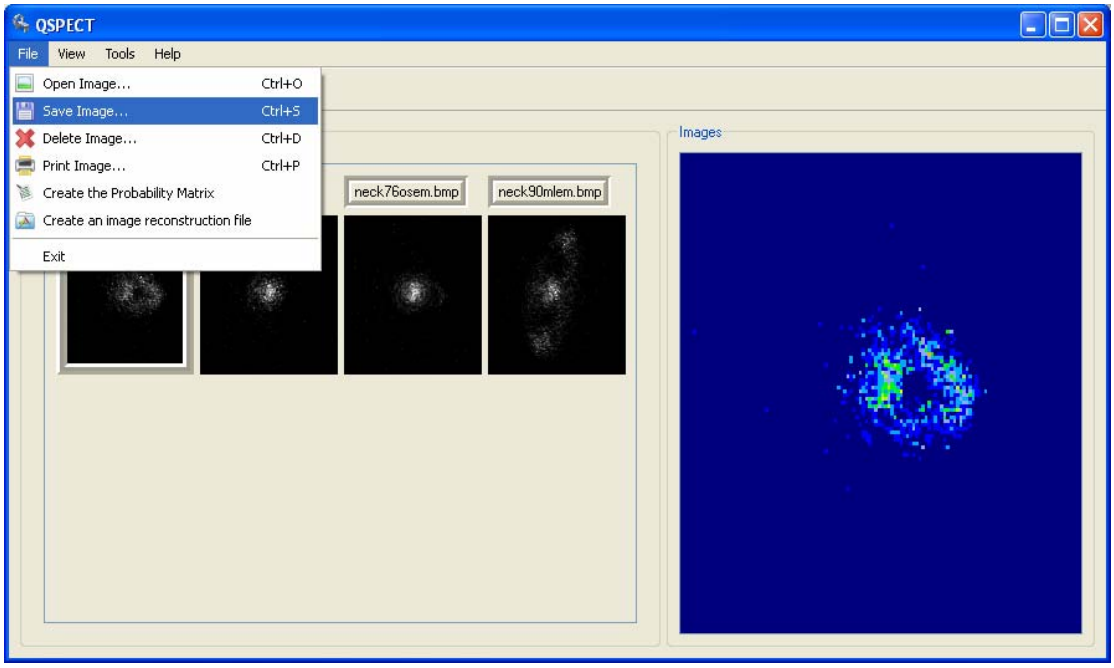

Εικόνα 4.26 :Η εικόνα μας σε απόχρωση κίτρινου. Τα Pixels με τιμές μηδέν εξακολουθούν να έχουν μαύρο χρώμα.




Εικόνα 4.27 :Μπορούμε να κάνουμε μια εικόνα έγχρωμη με την επιλογή Tools->Multi Color Image.



Εικόνα 4.28 : Τα pixels με μικρές τιμές έχουν μοβ χρώμα, ενώ αυτές με τις μεγαλύτερες τιμές κόκκινο.



Εικόνα 4.29 :Μετά από μια αλλαγή σε μια εικόνα μπορούμε να αποθηκεύσουμε την αλλαγμένη εικόνα.

# ΚΕΦΑΛΑΙΟ 5

# Συμπεράσματα - Μελλοντικές προοπτικές

## 5.1 Συμπεράσματα

Σε αυτό το κεφάλαιο θα συνοψίσουμε τα συμπεράσματα αυτής της εργασίας και θα αναφέρουμε μερικές μελλοντικές προοπτικές της.

- Όπως αναφέρθηκε και σε προηγούμενα κεφάλαια κατά τη διάρκεια της διπλωματικής ήταν αναγκαία η εξοικείωση με ένα εργαλείο δημιουργίας εφαρμογών με γραφικό περιβάλλον, όπως είναι το AWT ή το Swing για τη γλώσσα Java ή το Qt για τη γλώσσα C++. Η γλώσσα C++ επιλέχθηκε ως η γλώσσα, στην οποία αναπτύχθηκε η εφαρμογή μας καθώς ήταν διαθέσιμα τα προγράμματα για τον πίνακα πιθανοτήτων συστήματος και τους αλγόριθμους ανακατασκευής σε γλώσσα C και έτσι η μετατροπή τους σε C++ ήταν ευκολότερη. Για την κατασκευή του γραφικού περιβάλλοντος χρησιμοποιήθηκε το εργαλείο Qt, το οποίο είναι γνωστό λόγω του πλήθους ευρέως χρησιμοποιούμενων εφαρμογών που έχουν αναπτυχθεί με αυτό όπως το γραφικό περιβάλλον εργασίας για σταθμούς UNIX, KDE, ο web browser Opera, Google Earth, Skype κ.ά.. Το Qt χρησιμοποιείται κυρίως για συγγραφή εφαρμογών ανοικτού κώδικα (open source). Η εξοικείωση με το Qt έγινε με τη μελέτη συγγραμμάτων και εγχειριδίων και με την συγγραφή απλούστερων εφαρμογών.
- Πραγματοποιήθηκε βιβλιογραφική επισκόπηση της πυρηνικής ιατρικής απεικόνισης και ειδικότερα της τομογραφίας SPECT καθώς και των μεθόδων τομογραφικής ανακατασκευής. Οι αλγόριθμοι ανακατασκευής που χρησιμοποιήθηκαν ήταν οι επαναληπτικοί αλγόριθμοι MLEM και OSEM. Για την κατασκευή του πίνακα πιθανοτήτων συστήματος , που είναι αναγκαίος για την εφαρμογή των επαναληπτικών αλγορίθμων ανακατασκευής,

δημιουργήθηκε η δυνατότητα επιλογής ανάμεσα σε 3 διαφορετικούς αλγορίθμους ανάλογα με τα κατασκευαστικά χαρακτηριστικά του συστήματος.

- Οι αλγόριθμοι ανακατασκευής καθώς και οι αλγόριθμοι κατασκευής του πίνακα πιθανοτήτων ήταν διαθέσιμοι σε κώδικα C και για το λόγο αυτό εργαστήκαμε πάνω στην μετατροπή τους σε κώδικα C++ και την σύνδεσή τους με το γραφικό περιβάλλον, που αναπτύχθηκε με το Qt.
- Για να είναι ευκολότερη η διαδικασία ανακατασκευής μια τομής SPECT για τον χρήστη, χρησιμοποιήσαμε wizards για τύπου βήμα-βήμα διαδικασίες. Το ίδιο μοντέλο χρησιμοποιήσαμε για την κατασκευή του πίνακα πιθανοτήτων του συστήματος και την κατασκευή ενός αρχείου εικόνας από ένα αρχείο ανακατασκευασμένης τομής.
- Η ορθή λειτουργία της εφαρμογής και των αλγορίθμων ανακατασκευής δοκιμάστηκε χρησιμοποιώντας κλινικά δεδομένα, τα οποία παρήχθησαν από SPECT/CT camera Varicam, του οίκου GE Medical από την κλινική Charite Verichow του Βερολίνου.

## 5.2 Μελλοντικές προοπτικές

Στην εργασία αυτή έγινε μια πρώτη προσπάθεια για την ανάπτυξη λογισμικού με γραφικό περιβάλλον για την ανακατασκευή τομών από δεδομένα SPECT , έτσι ώστε να είναι φιλικό προς το χρήστη. Ακολουθούν κάποιες πιθανές μελλοντικές προοπτικές.

- Βελτιστοποίηση του κώδικα, τόσο του γραφικού περιβάλλοντος όσο και των αλγορίθμων ανακατασκευής ώστε η κάθε λειτουργία να γίνεται γρηγορότερα και με αποτελεσματικότερο τρόπο. Για παράδειγμα, όσον αφορά τη λειτουργία του γραφικού περιβάλλοντος και την επικοινωνία του λογισμικού με το χρήστη θα μπορούσαν να χρησιμοποιηθούν πολυνηματικές τεχνικές (multi threads techniques).
- Δημοσιοποίηση της εφαρμογής στο διαδίκτυο ως δωρεάν εφαρμογής ανοικτού κώδικα. Ο επισκέπτης της σελίδας θα μπορεί να «κατεβάσει»

(download) την εφαρμογή στον προσωπικό του υπολογιστή μαζί ίσως με κάποια παραδείγματα εκτέλεσης και οδηγίες.

- Ενσωμάτωση περισσότερων λειτουργιών. Θα μπορούσαν να ενσωματωθούν άλλα είδη αλγορίθμων ανακατασκευής καθώς επίσης και δυνατότητα διόρθωσης της απόσβεσης των φωτονίων. Ακόμα θα ήταν χρήσιμο να ενσωματωθούν κάποιες περισσότερες λειτουργίες επεξεργασίας εικόνας, έτσι ώστε η εφαρμογή να γίνει πιο αυτόνομη όσον αφορά την επεξεργασία των εικόνων των τομών.

# Βιβλιογραφία


1. Δ. Κουτσούρης, Κ.Ν., Σ. Παυλόπουλος, ed. *Ιατρικά Απεικονιστικά Συστήματα*. 2004, Εκδόσεις Τζιόλα: Θεσσαλονίκη.
2. A. C. Kak, M.S., ed. *Principles of Computerized Tomographic Imaging*. 1988, IEEE Press.
3. *http://en.wikipedia.org/wiki/Computed_tomography*.
4. Groch, M.W. and W.D. Erwin, *SPECT in the year 2000: basic principles.* J Nucl Med Technol, 2000. **28**(4): p. 233-44.
5. Γαζής, Ε.Ν., *ΙΟΝΤΙΖΟΥΣΕΣ ΑΚΤΙΝΟΒΟΛΙΕΣ, ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΛΟΓΙΑ ΚΑΙ ΙΑΤΡΙΚΗ.* 2002.
6. *http://en.wikipedia.org/wiki/Single_photon_emission_computed_tomography*. [cited.
7. *http://en.wikipedia.org/wiki/Gamma_camera*.
8. Zeng, G.L., *Image reconstruction--a tutorial.* Comput Med Imaging Graph, 2001. **25**(2): p. 97-103.
9. *http://bigwww.epfl.ch/demo/jtomography/*.
10. Loudos, G., *ΑΝΑΠΤΥΞΗ ΤΟΜΟΣΠΙΝΘΗΡΟΓΡΑΦΙΚΗΣ γ-ΚΑΜΕΡΑ ΥΨΗΛΗΣ ΕΥΑΙΣΘΗΣΙΑΣ ΚΑΙ ΔΙΑΚΡΙΤΙΚΗΣ ΙΚΑΝΟΤΗΤΑΣ.* 2002.
11. Loudos, G.K., *An efficient analytical calculation of probability matrix in 2D SPECT.* Comput Med Imaging Graph, 2008. **32**(2): p. 83-94.
12. Loudos, G., *NOTES ON THE IMPLEMENTATION OF ITERATIVE RECONSTRUCTION ALGORITHMS.* Created on Sunday21 January 2001, Translated and revised on Sunday 31 October 2004.
13. Shepp, L.A. and Y. Vardi, *Maximum likelihood reconstruction for emission tomography.* IEEE Trans Med Imaging, 1982. **1**(2): p. 113-22.
14. Lange, K., M. Bahn, and R. Little, *A theoretical study of some maximum likelihood algorithms for emission and transmission tomography.* IEEE Trans Med Imaging, 1987. **6**(2): p. 106-14.
15. Hudson, H.M. and R.S. Larkin, *Accelerated image reconstruction using ordered subsets of projection data.* IEEE Trans Med Imaging, 1994. **13**(4): p. 601-9.

16. *http://osem.s-pla.net/*.
17. *http://trolltech.com/products/qt*.
18. *http://en.wikipedia.org/wiki/Qt_(toolkit)*.
19. ASA, T., *Qt Assistant by Trolltech*. 2000-2008.