ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΟΜΕΑΣ ΦΥΣΙΚΗΣ

Εφαρμογές του μετασχηματισμού Radon στην αναγνώριση εικόνας και υλοποίηση σε ψηφιακό επεξεργαστή σήματος (DSP)

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημήτρης Κουζής – Λουκάς

Επιβλέπων: Θεόδωρος Αλεξόπουλος
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα Οκτώβριος 2004

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΟΜΕΑΣ ΦΥΣΙΚΗΣ

Εφαρμογές του μετασχηματισμού Radon στην αναγνώριση εικόνας και υλοποίηση σε ψηφιακό επεξεργαστή σήματος (DSP)

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημήτρης Κουζής – Λουκάς

Επιβλέπων: Θεόδωρος Αλεξόπουλος
Αν. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 29η Οκτωβρίου 2004.

Θ. Αλεξόπουλος Αν. Καθηγητής Ε.Μ.Π.	Στ. Μαλτέζος Επ. Καθηγητής Ε.Μ.Π.	Γ. Τσιπολίτης Επ. Καθηγητής Ε.Μ.Π.

Αθήνα Οκτώβριος 2004

.....
Δημήτρης Κουζής – Λουκάς
Διπλωματούχος της σχολής Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών Ε.Μ.Π.

e-mail επικοινωνίας: lookfwd@doit4me.gr
web page: http://www.doit4me.gr/~lookfwd

Περίληψη

Ο σκοπός της διπλωματικής εργασίας είναι η υλοποίηση μίας αυτόνομης ολοκληρωμένης διάταξης αναγνώρισης εικόνας, βασισμένης σε ένα σύγχρονο πυρήνα (επεξεργαστή) ψηφιακής επεξεργασίας σήματος (DSP).

Αυτό που ζητάμε να ανιχνεύσουμε είναι η θέση και η γωνία των οπών στήριξης ειδικών κυλινδρικών ανιχνευτών μιονίων που θα χρησιμοποιηθούν στον ανιχνευτή ATLAS του πειράματος LHC στο CERN. Για την λύση αυτού του προβλήματος αναγνώρισης εικόνας χρησιμοποιήθηκε ο κυκλικός μετασχηματισμός Radon λόγω της υψηλής απόρριψης θορύβου. Για την υποστήριξη του συστήματος χρησιμοποιήθηκαν τεχνικές υπολογιστικής γραφικής (αλγόριθμος του Bresenham), κωδικοποίητες και αποκωδικοποίητες video βασισμένοι στο πρωτόκολλο ψηφιακού video ITU 656, γενικές τεχνικές επεξεργασίας σήματος και εικόνας (σχεδιασμός φίλτρων, cross correlation, ανίχνευση αιχμών) και στατιστικής αναγνώρισης προτύπων (κατηγοριοποιητές Bayes).

Το σύστημα αυτό αποτελείται από κάμερα, την αναπτυξιακή πλακέτα EZ-KIT Lite για τον επεξεργαστή σήματος ADSP – BF533® της Analog Devices και μία οθόνη τηλεόρασης στην οποία παρουσιάζονται τα αποτελέσματα.

Η αναλυτική παρουσίαση της μεθοδολογίας που ακολουθήθηκε και άλλων στοιχείων, μπορούν να δώσουν στον αναγνώστη ένα χαρακτηριστικό παράδειγμα των βημάτων της διαδικασίας ανάπτυξης συστημάτων αναγνώρισης εικόνας.

Λέξεις Κλειδιά:

Μετασχηματισμός Radon και Hough, BF533, αναγνώριση εικόνας, πείραμα ATLAS, ITU 656

Abstract

The purpose of this thesis is to implement a complete standalone image recognition system based on a state of the art DSP core.

We want to measure the position and the angle of mounting points of some special cylindrical Meuon detectors that are being used in high energy physics experiments and more specifically in the ATLAS detector of LHC experiment at CERN. In order to accomplish this complex image recognition task we use the circular version of Radon transform because of its robustness against noise. In order to complete the system we use techniques from several areas like computer graphics (Bresenham algorithm), video encoders/decoders using the ITU 656 modern digital video protocol, general signal and image processing techniques (filter design, cross correlation, edge detection) and statistical pattern recognition (Bayesian classifiers).

The setup includes a video camera with standard video out, EZ-KIT Lite development board for Analog Devices' ADSP – BF533® digital signal processor and a video monitor where the results are being presented.

The in depth analysis and presentation of the methodology we used while developing this application and complete theory coverage can advance the reader with a complete overview of the steps that are required for developing an embedded image recognition system.

Keywords:

Radon and Hough transform, BF 533, image recognition, ATLAS experiment, ITU 656

Ευχαριστίες

Θα ήθελα να ευχαριστήσω όλους όσους με βοήθησαν στην εκπόνηση αυτής της διπλωματικής εργασίας και όσους μου συμπαραστάθηκαν κατά τη διάρκεια των σπουδών μου στο Εθνικό Μετσόβιο Πολυτεχνείο.

Ειδικότερα, ευχαριστώ τα μέλη της τριμελούς επιτροπής Μαλτέζο Σταύρο και Τσιπολίτη Γιώργο για την ουσιαστική εποικοδομητική κριτική. Ιδιαίτερα θα ήθελα να ευχαριστήσω τον επιβλέποντα, Αλεξόπουλο Θεόδωρο. Η υποστήριξη και η εμπιστοσύνη του συνέβαλλαν ανεκτίμητα στην ολοκλήρωση αυτής της διπλωματικής εργασίας.

Τέλος θα ήθελα να αφιερώσω αυτή την εργασία σε όλους όσους στήριξαν με την αγάπη τους την προσπάθειά μου για το καλύτερο.

Πίνακας περιεχομένων

Πίνακας εικόνων και σχημάτων

Εισαγωγή

Α. Το πρόβλημα και η τεχνικές που χρησιμοποιούνται

Όπως φαίνεται και από τον τίτλο αυτής της εργασίας, πραγματεύεται τα ακόλουθα τρία αντικείμενα:

1. Τη θεωρία του μετασχηματισμού Radon
2. Τη θεωρία επεξεργασίας και αναγνώρισης εικόνας
3. Την υλοποίηση συστημάτων βασισμένα σε ψηφιακούς επεξεργαστές σήματος (DSPs)

Η θεωρία για τα τρία αυτά αντικείμενα παρουσιάζεται ξεχωριστά και εφαρμόζεται στην πράξη για την αντιμετώπιση ενός σύνθετου προβλήματος αναγνώρισης εικόνας.

Το πρόβλημά προς επίλυση είναι η αναγνώριση ενός συγκεκριμένου προτύπου στην εικόνα κυλινδρικού σωλήνα ολίσθησης για ανίχνευση μιονίων (Monitored Drift Tube - MDT). Πάνω από 300.000 τέτοιοι αισθητήρες θα χρησιμοποιηθούν στον ανιχνευτή ATLAS για μετρήσεις γεγονότων με μιόνια κατά την διεξαγωγή του πειράματος υψηλών ενεργειών με τον επιταχυντή LHC στο CERN το οποίο αναμένεται να αρχίσει το 2007.

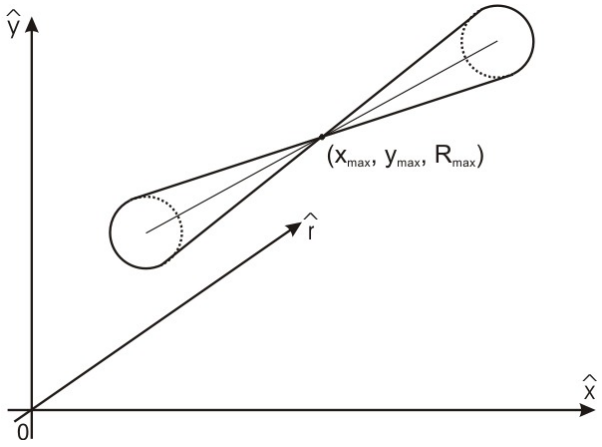
Εικόνα 1. Το πρότυπο το οποίο θέλουμε να αναγνωρίσουμε

Δύο στοιχεία πρέπει να αναγνωρίσουμε σε εικόνες των MDTs όπως φαίνεται στην Εικόνα 1. Το πρώτο είναι η θέση (κέντρο) του κυκλικού δίσκου της επιφάνειας του MDT και το δεύτερο είναι η γωνιακή θέση των δύο μικρών οπών στήριξης που βρίσκονται στην περιφέρεια του. Δεν γίνεται καμία υπόθεση για το περιεχόμενο της λαμβανόμενης εικόνας εκτός από ότι η γωνία παρατήρησης θεωρείται αξονική, ώστε το περίγραμμά του να προσεγγιστεί αρκετά καλά ως κυκλικό και όχι ελλειπτικό. Οι πληροφορίες αυτές είναι πάρα πολύ χρήσιμες στην αυτοματοποίηση διαδικασιών κατασκευής και ελέγχου τους π.χ. με τη βοήθεια κάποιου ρομποτικού βραχίονα.

Η διάταξη που χρησιμοποιούμε για την υλοποίηση του παραπάνω συστήματος αποτελείται από μία κάμερα η οποία συλλαμβάνει (capture) την εικόνα, το αναπτυξιακό σύστημα του επεξεργαστή ADSP-BF533 της Analog Devices, EZ-KIT BF533^{TM} το οποίο κάνει όλη την επεξεργασία και μία οθόνη τηλεοράσεως στην οποία προβάλλονται τα αποτελέσματα. Είναι δυνατή η παρουσίαση των μετρήσεων και σε υπολογιστή μέσω της σειριακής θύρας του αναπτυξιακού.

Για την ανίχνευση αυτού του προτύπου χρησιμοποιείται ο μετασχηματισμός Radon. Ο μετασχηματισμός Radon είναι ένας μαθηματικός μετασχηματισμός στατιστικής φύσεως που τυχαίνει ευρείας αποδοχής στην επεξεργασία εικόνας (πολλές φορές με το όνομα Hough transform που αποτελεί ειδική περίπτωση του – διακριτή εκδοχή). Η πιο διαδεδομένη του μορφή είναι αυτή του γραμμικού μετασχηματισμού Radon που μετατρέπει έναν x-y χώρο σε ένα χώρο γραμμών r-φ. Για τους σκοπούς της εφαρμογής μας ήταν αναγκαία η κυκλική έκδοση του μετασχηματισμού για την οποία η βιβλιογραφία είναι εξαιρετικά περιορισμένη. Αυτός μετατρέπει ένα δισδιάστατο x,y

χώρο σε ένα τρισδιάστατο χώρο κύκλων x,y,R πετυχαίνοντας με αυτόν τον τρόπο αύξηση των διαστάσεων κατά μία. Στον χώρο x,y,R είναι πολύ πιο εύκολο να ξεχωρίσουμε σχήματα κύκλων σε σύγκριση με τον χώρο x, y.



Εικόνα 2. Κυκλικός μετασχηματισμός Radon ενός κύκλου

Συγκεκριμένα κάθε κύκλος στον χώρο x,y μετασχηματίζεται σε κάτι σαν διπλό κώνο στον χώρο x,y,R. Στην κορυφή του κώνου αυτού παρατηρούμε ένα μέγιστο της τιμής του μετασχηματισμού Radon το οποίο μπορεί να έχει μέγεθος πάνω από πέντε φορές μεγαλύτερο από την μέση τιμή του μετασχηματισμού. Το σημείο αυτού του μεγίστου μας δίνει τις πληροφορίες του κέντρου του κύκλου και της ακτίνας του.

Ο μετασχηματισμός αυτός μας δίνει δυνατότητα πολύ αξιόπιστης αναγνώρισης κύκλων ενώ η στατιστική υφή του τον κάνει να απορρίπτει διάφορες μορφές θορύβου. Το μειονέκτημά του είναι το υπολογιστικό του κόστος και οι αυξημένες απαιτήσεις σε μνήμη λόγω. Οι σύγχρονες όμως δυνατότητες επεξεργασίας των DSPs σε συνδυασμό με κατάλληλο προγραμματισμό και βελτιστοποίηση του κώδικα δίνουν τη δυνατότητα να κατασκευάσουμε σύστημα που να μπορεί να εκτελεί σε πραγματικό χρόνο τόσο το μετασχηματισμό όσο και ανάλυση πάνω στο χώρο Radon.

Για την εφαρμογή του μετασχηματισμού Radon και την ολοκλήρωση ενός συστήματος αναγνώρισης εικόνας βασισμένου σε αυτόν, χρειάζεται ένα πλήθος ειδικών τεχνικών.

1. Για τη σύλληψη της εικόνας και για τη γραφική παρουσίαση των αποτελεσμάτων σε κάποια οθόνη είναι απαραίτητη η εμβάθυνση στα ψηφιακά πρότυπα video και στο ειδικό hardware σύλληψης και παραγωγής των σχετικών σημάτων (video decoders – encoders).
2. Για να γίνει σωστά ο μετασχηματισμός Radon χρειάζεται πάνω στην εικόνα να έχει πραγματοποιηθεί προ-επεξεργασία για την εξάλειψη του οπτικού θορύβου που δημιουργείται κατά την σύλληψη και ανίχνευση αιχμών (edge tracing).
3. Για την αναγνώριση των κύκλων στον χώρο Radon και την αποτελεσματική απόρριψη του θορύβου απαιτούνται διεργασίες ψηφιακής επεξεργασίας σήματος (συναρτήσεις cross correlation, ψηφιακά φίλτρα) και της αναγνώρισης προτύπων (Bayesian classifiers).
4. Για την απεικόνιση των αποτελεσμάτων χρειάζονται διεργασίες γραφικής πάνω σε καμβά (raster) που προέρχονται από μεθόδους υπολογιστικής γραφικής (computer graphics).

Για την υλοποίηση του συστήματος σε DSP είναι απαραίτητη η γενική γνώση του τρόπου λειτουργίας και των ιδιαιτεροτήτων των DSPs. Για τον συγκεκριμένο επεξεργαστή που χρησιμοποιούμε στο σύστημά μας (ADSP-BF533®) είναι απαραίτητη η γνώση σε βάθος τόσο των δυνατοτήτων του πυρήνα όσο και της λειτουργίας των ενσωματωμένων περιφερειακών του και ιδιαίτερα αυτών που εμείς χρησιμοποιούμε. Επιπλέον είναι απαραίτητη η γνώση των δυνατοτήτων του αναπτυξιακού συστήματος και η λεπτομερής γνώση της λειτουργίας των περιφερικών που χρησιμοποιούμε δηλαδή του κωδικοποιητή και αποκωδικοποιητή video. Επιπλέον την διαδικασία ανάπτυξης βοηθούν το περιβάλλον λογισμικού VisualDSP++ του οποίου τις δυνατότητες πρέπει να γνωρίζουμε και η έντυπη και ηλεκτρονική τεκμηρίωση στις οποίες μπορούμε να ανατρέχουμε κάθε φορά που εμφανίζεται κάποιο πρόβλημα. Πρέπει να σημειωθεί ότι ο επεξεργαστής αυτός

κυκλοφόρησε μόλις το 2003 και συνεπώς η βιβλιογραφία που τον αφορά είναι σχετικά περιορισμένη. Ευελπιστούμε ότι η παρούσα εργασία θα αποτελέσει μία σημαντική συνεισφορά στην βιβλιογραφία του επεξεργαστή αυτού και μάλιστα στην Ελληνική γλώσσα.

Β. Οργάνωση και δομή του συγγράμματος

Ο αναγνώστης θα βρει σε αυτή την διπλωματική πληροφορίες και χρήσιμες αναφορές για όλα όσα περιγράφηκαν στην προηγούμενη ενότητα. Προαπαιτούμενα για την κατανόηση του κειμένου είναι η μία γενική γνώση όρων της τεχνολογίας υπολογιστών και των μαθηματικών. Για την κατανόηση του κώδικα, όπου αυτός υπάρχει, συνίσταται η γνώση της γλώσσας C και του Matlab. Επίσης χρειάζεται υπομονή σε ορισμένα σημεία που ο αναγνώστης πιθανώς συναντά έννοιες με τις οποίες δεν έχει ξαναασχοληθεί. Σε αυτές τις περιπτώσεις ίσως θα βοηθούσε η μελέτη και των σχετικών αναφορών.

Το σύγγραμμα αυτό είναι χωρισμένο σε δύο μέρη. Το μέρος Α παρουσιάζει πληροφορίες για όλους τους τομείς που αναφέραμε στην προηγούμενη ενότητα. Το μέρος Β παρουσιάζει τις λεπτομέρειες υλοποίησης και την αξιολόγηση της εφαρμογής που αναπτύξαμε για την λύση του προβλήματος που περιγράψαμε. Παρακάτω παρουσιάζουμε συνοπτικά τα κεφάλαια καθώς και τον σκοπό τον οποίο εξυπηρετούν.

Μέρος Α. Θεωρητικά

Κεφάλαιο 1^{ο}: Στο κεφάλαιο αυτό παρουσιάζεται η θεωρία του κλασικού (γραμμικού) και κυκλικού μετασχηματισμού Radon. Στο τέλος παρουσιάζονται εφαρμογές οι οποίες αναδεικνύουν την χρησιμότητα του μετασχηματισμού αυτού. Το κεφάλαιο αυτό αποσκοπεί στο να χρησιμοποιηθεί ως αναφορά για τον αναγνώστη που ενδιαφέρεται για τον μετασχηματισμό Radon και τις εφαρμογές του. Εκτός από τον ορισμό παρουσιάζονται και οι πολύ χρήσιμες ιδιότητές του τόσο για την γραμμική όσο και για την κυκλική και γενικευμένη μορφή του. Επίσης στην ενότητα 1.3 παρουσιάζεται σύνοψη της θεωρίας που χρησιμοποιείται για το πειραματικό μέρος.

Κεφάλαιο 2^{ο}: Στο κεφάλαιο αυτό παρουσιάζονται τεχνικές της επεξεργασίας και αναγνώρισης εικόνας. Το αντικείμενο αυτό είναι πολύ εκτενές και φυσικά δεν μπορεί να καλυφθεί μέσα σε ένα μόνο κεφάλαιο. Ο στόχος μας είναι να παρουσιάσουμε στον μη ειδικό αναγνώστη την γενική εικόνα για το τι είναι ένα σύστημα επεξεργασίας και αναγνώρισης εικόνας και επιγραμματικά τις σημαντικότερες τεχνικές που χρησιμοποιούνται για κάθε τμήμα της υλοποίησής του. Οι αναφορές μπορούν να αποτελέσουν πολύτιμο υλικό για όποιον θέλει να ασχοληθεί για να ενημερωθεί περισσότερο για κάποιες τεχνικές που πιθανώς τον ενδιαφέρουν. Αναλύονται εκτενώς οι τεχνικές που χρησιμοποιούνται στην δικιά μας εφαρμογή. Ειδικά στην ενότητα 2.5.1.1 γίνεται ανάλυση του συστήματος αναγνώρισης προτύπου που χρησιμοποιούμε.

Κεφάλαιο 3^{ο}: Το κεφάλαιο αυτό παρουσιάζει την τεχνολογία των DSPs και αναλύει σε βάθος τον επεξεργαστή και το αναπτυξιακό που χρησιμοποιούμε στην εφαρμογή μας. Το κεφάλαιο αυτό αποσκοπεί στο να δώσει στον αναγνώστη που ασχολείται για πρώτη φορά με τα DSPs μία γερή βάση σχετικά με την δομή και τις αρχές λειτουργίας τους. Αμέσως μετά παρουσιάζεται αναλυτικά ο επεξεργαστής BF533® και το αναπτυξιακό του σύστημα όπως και το λογισμικό που το συνοδεύει. Ο στόχος αυτού του δεύτερου μέρους είναι να δώσει σε όποιον θελήσει να ασχοληθεί με αυτόν τον σχετικά «νέο» επεξεργαστή ειδικές γνώσεις που θα τον βοηθήσουν να μπορέσει να αναπτύξει γρήγορα νέες εφαρμογές χωρίς να μπει στον κόπο της πλήρους ανάγνωσης των εκτενών εγχειριδίων του επεξεργαστή.

Μέρος Β. Υλοποίηση

Κεφάλαιο 4^{ο}: Στο σύντομο αυτό κεφάλαιο παρουσιάζεται εκτενώς το πρόβλημα που πρέπει να λυθεί και γίνεται αναφορά σε προηγούμενες εκδοχές που έχουν χρησιμοποιηθεί για την μερική του λύση. Το κεφάλαιο αυτό σκοπεύει να παρουσιάσει στον αναγνώστη τις απαραίτητες λεπτομέρειες που αφορούν το συγκεκριμένο πρόβλημα ώστε να μπορεί να κατανοήσει καλύτερα τα επόμενα κεφάλαια.

Κεφάλαιο 5^{ο}: Στο κεφάλαιο αυτό παρουσιάζεται η πρότυπη υλοποίηση κάποιον κομματιών της εφαρμογής σε PC με τη βοήθεια του πακέτου λογισμικού Matlab^{TM}. Η προτυποποίηση αυτή ήταν απαραίτητη για να ελέγξουμε γρήγορα αν ο μετασχηματισμός Radon και οι διάφορες άλλες τεχνικές μπορούν να δώσουν λύση στο πρόβλημά μας και να αποκτήσουμε αίσθηση των σημείων που πρέπει να προσέξουμε κατά την υλοποίηση στο DSP. Το κεφάλαιο αυτό έχει πολύ ενδιαφέρον για τον αναγνώστη που θέλει να παρακολουθήσει βήμα – βήμα την διαδικασία ανάπτυξης μίας εφαρμογής αναγνώρισης εικόνας ώστε να κατανοήσει τη μεθοδολογία που ακολουθείται και να αποκτήσει αίσθηση των προβλημάτων που καλείται κανείς να αντιμετωπίσει στην πορεία και του σκεπτικού λύσεώς τους.

Κεφάλαιο 6^{ο}: Στο κεφάλαιο αυτό παρουσιάζεται η τελική υλοποίηση στο DSP. Το κεφάλαιο αυτό έρχεται ως συνέχεια των κεφαλαίων 3 και 5 για να παρουσιάσει ένα πρακτικό παράδειγμα υλοποίησης ενός συστήματος DSP. Το κεφάλαιο αυτό θα φανεί πολύ χρήσιμο σε όποιον θέλει να αναπτύξει ένα σύστημα DSP αφού θα γνωρίσει από κοντά όλα τα πράγματα που θα πρέπει να προσέξει κατά την υλοποίησή του.

Κεφάλαιο 7^{ο}: Στο κεφάλαιο αυτό γίνεται αξιολόγηση της απόδοσης της όλης διάταξης και ωφέλιμη κριτική σχετικά με τις τεχνικές που χρησιμοποιήθηκαν. Το κεφάλαιο αυτό συμπληρώνεται από προτάσεις για το πως διάφορες τεχνικές που χρησιμοποιήθηκαν μπορούν να αξιοποιηθούν και σε άλλες εφαρμογές όπως και πιθανές εφαρμογές ολόκληρης της διάταξης. Το κεφάλαιο αυτό θα φανεί χρήσιμο σε όποιον θελήσει να αξιοποιήσει περαιτέρω το περιεχόμενο αυτής της διπλωματικής.

Σημείωση: Η διπλωματική αυτή συνοδεύεται από τον πηγαίο κώδικα που χρησιμοποιήθηκε σε διάφορα μέρη της υλοποίησης. Δεν κρίθηκε σκόπιμο να τυπωθεί αυτός ο κώδικας μιας και έχει αρκετά μεγάλο όγκο και η τυπωμένη έκδοσή του ελάχιστη πρακτική αξία θα είχε. Τα σημεία του κώδικα με ιδιαίτερο ενδιαφέρον παρουσιάζονται αποσπασματικά στα αντίστοιχα κεφάλαια. Όποιος θέλει τον πλήρη πηγαίο κώδικα μπορεί να τον κατεβάσει από το διαδύκτιο.

Μέρος Α. Θεωρία

Κεφάλαιο 1. Θεωρία και εφαρμογές μετασχηματισμού Radon

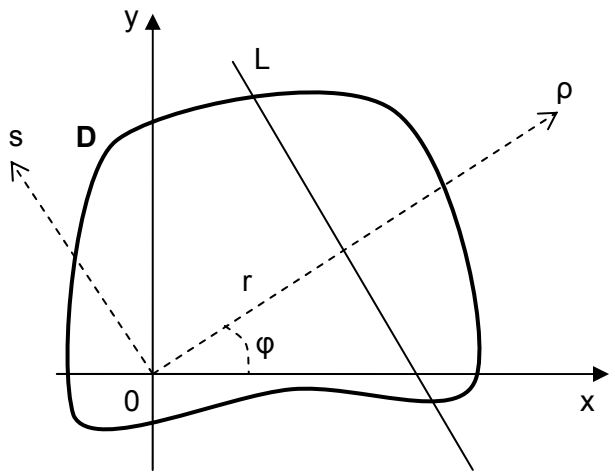
Στο κεφάλαιο αυτό παρουσιάζεται η θεωρία του κλασικού (γραμμικού) και κυκλικού μετασχηματισμού Radon. Στο τέλος παρουσιάζονται εφαρμογές οι οποίες αναδεικνύουν την χρησιμότητα του μετασχηματισμού αυτού. Το κεφάλαιο αυτό αποσκοπεί στο να χρησιμοποιηθεί ως αναφορά για τον αναγνώστη που ενδιαφέρεται για τον μετασχηματισμό Radon και τις εφαρμογές του. Εκτός από τον ορισμό παρουσιάζονται και οι πολύ χρήσιμες ιδιότητές του τόσο για την γραμμική όσο και για την κυκλική και γενικευμένη μορφή του. Επίσης στην ενότητα 1.3 παρουσιάζεται σύνοψη της θεωρίας που χρησιμοποιείται για το πειραματικό μέρος.

1.1 Κλασικός μετασχηματισμός Radon.

Ο γραμμικός μετασχηματισμός Radon αποτελεί την πλέον διαδεδομένη μορφή του. Το μεγαλύτερο μέρος των πληροφοριών αυτής της ενότητας έχει ως πηγή του το βιβλίο του Stanely R. Deans[1] το οποίο είναι αφιερωμένο σε αυτόν τον μετασχηματισμό. Θα παρουσιάσουμε τον ορισμό σε διάφορες μορφές χρήσιμες για διαφορετικές εφαρμογές και χρήσιμες ιδιότητές του.

1.1.1 Ορισμός

Έστω μία συνάρτηση $f : \Re^2 \to \Re$ δύο μεταβλητών x, y που ορίζεται σε ένα χώρο D του $\Re^2$ όπως φαίνεται στο παρακάτω σχήμα και L μία τυχαία ευθεία στο επίπεδο.



Εικόνα 3. Ορισμός του μετασχηματισμού Radon

Η απεικόνιση που ορίζεται από την προβολή (ολοκλήρωμα πάνω σε ευθεία) της f πάνω σε όλες τις πιθανές γραμμές L είναι ο δισδιάστατος μετασχηματισμός Radon υπό την προϋπόθεση ότι το ολοκλήρωμα υπάρχει. Πιο συγκεκριμένα,

$$\breve{f} = \Re f = \int_L f(x,y)ds \quad (1)$$

όπου ds είναι στοιχειώδες τμήμα πάνω στη ευθεία.

Θεωρώντας την παραμετρική μορφή της L η εξίσωσή της δίνεται από την σχέση:

$$r = x\cos\phi + y\sin\phi$$

[1] The Radon Transform and Some of Its Applications, Stanley R. Deans, John Wiley & Sons, 1983

Ο μετασχηματισμός Radon είναι λοιπόν το ολοκλήρωμα (1) για κάθε ευθεία που ορίζεται από τα r, φ. Συνεπώς αν το $\breve{f}(r,\phi)$ είναι γνωστό για κάθε r, φ τότε αυτός είναι ο δισδιάστατος μετασχηματισμός Radon της συνάρτησης $f(x,y)$ ενώ αν είναι γνωστός μόνο για συγκεκριμένες τιμές των r, φ λέμε ότι έχουμε ένα δείγμα του μετασχηματισμού Radon.

Αν θεωρήσουμε το περιστραμένο κατά φ σύστημα συντεταγμένων ρ, s τότε τα x και y για τυχαίο σημείο δίνονται από τις σχέσεις:

$$x = \rho\cos\phi - s\sin\phi$$
$$y = \rho\sin\phi + s\cos\phi$$

Έτσι μπορούμε να γράψουμε τον ορισμό (1) με την μορφή απλού ολοκληρώματος:

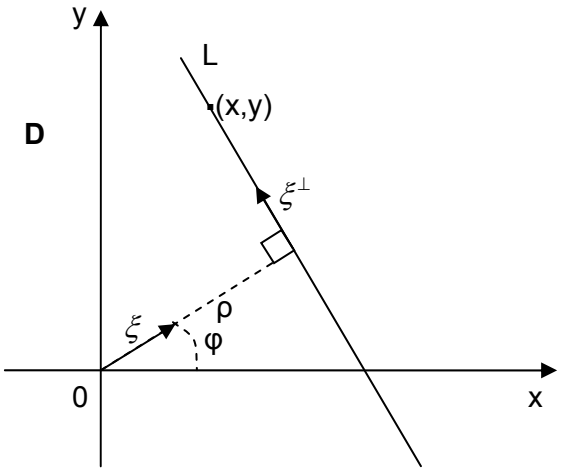
$$\breve{f}(\rho,\phi) = \int_{-\infty}^{\infty} f(\rho\cos\phi - s\sin\phi, \rho\sin\phi + s\cos\phi)ds \quad (2)$$

Στην προσπάθειά μας να γενικεύσουμε και σε περισσότερες διαστάσεις, θα εισάγουμε τον διανυσματικό συμβολισμό.

Αν $\mathbf{x} = (x,y)$ είναι ένα διάνυσμα τότε η συνάρτηση f που ορίσαμε προηγουμένως μπορεί να παρασταθεί συνοπτικά ως $f(\mathbf{x})$. Μπορούμε να ορίσουμε δύο κάθετα μοναδιαία διανύσματα ξ και $\xi^{\perp}$ ως εξής:

$$\begin{aligned} \xi &= (\cos\phi, \sin\phi) \\ \xi^{\perp} &= (-\sin\phi, \cos\phi) \end{aligned} \quad (3)$$

Τα διανύσματα αυτά φαίνονται στο παρακάτω σχήμα.



Εικόνα 4. Διανυσματικός ορισμός του μετ/μου Radon

Με την εισαγωγή αυτού ων των συμβολισμών, ο ορισμός του μετ/μου Radon απλοποιείται σημαντικά ως έκφραση στην παρακάτω μορφή:

$$\breve{f}(\rho,\xi) = \int_{-\infty}^{\infty} f(\rho\xi + t\xi^{\perp})dt \quad (4)$$

Όπως φαίνεται από την σχέση (3) τα ξ και $\xi^{\perp}$ εξαρτώνται άμεσα από το φ. Συνεπώς οι εκφράσεις $\breve{f}(\rho,\xi)$ και $\breve{f}(\rho,\phi)$ είναι ισοδύναμες.

Η συνάρτηση της ευθείας μπορεί να γραφτεί σε διανυσματική μορφή ως:

$$\rho = \xi \cdot \mathbf{x} = x\cos\phi + y\sin\phi$$

Εισάγοντας την συνάρτηση δέλτα του Dirac μπορούμε να απλοποιήσουμε ακόμα περισσότερο την έκφραση του μετ/μου Radon ολοκληρώνοντας σε όλο τον χώρο $\Re^2$ την συνάρτηση f και αναθέτοντας στην συνάρτηση δέλτα να «διαλέξει» τα σημεία που βρίσκονται πάνω στην ευθεία. Θυμίζουμε ότι η συνάρτηση δέλτα $\delta(x)$ έχει τιμή 0 για κάθε $x \neq 0$ ενώ για x = 0 έχει τιμή τέτοια ώστε,

$$\int_{-\infty}^{\infty} \delta(x)dx = 1$$

Όπως φαίνεται από τη παραπάνω σχέση, η έκφραση $\rho - \xi \cdot \mathbf{x}$ είναι 0 για όλα τα $\mathbf{x}$ που ανήκουν στην ευθεία L ενώ είναι μη μηδενικά για όλα τα υπόλοιπα σημεία. Συνεπώς ο μετασχηματισμός Radon μπορεί να γραφεί:

$$\breve{f}(\rho,\xi) = \int f(\mathbf{x})\delta(\rho - \xi \cdot \mathbf{x})d\mathbf{x} \quad (5)$$

όπου η ολοκλήρωση γίνεται σε όλο τον χώρο $\Re^2$.

Το πλεονέκτημα αυτής της μορφής είναι ότι γενικεύεται ως είναι και σε μεγαλύτερες διαστάσεις. Πραγματικά, αν θεωρήσουμε το $\mathbf{x}$ να είναι διάνυσμα στον χώρο $\Re^3$, $\mathbf{x} = (x, y, z)$, $d\mathbf{x} = dxdydz$ και αν το διάνυσμα ξ είναι ένα μοναδιαίο διάνυσμα στον $\Re^3$, τότε η έκφραση 5 μας δίνει τον μετασχηματισμό Radon για τον τρισδιάστατο χώρο μόνο που τώρα η ολοκλήρωση δεν γίνεται πάνω σε ευθεία αλλά σε επίπεδο.

Αντίστοιχα μπορούμε να γενικεύσουμε σε n διαστάσεις όπου το $\mathbf{x}$ θα είναι ένα διάνυσμα στον $\Re^n$, $\mathbf{x} = (x_1, x_2, ...x_n)$, το $d\mathbf{x} = dx_1dx_2...dx_3$ ο στοιχειώδης όγκος και το ξ ένα μοναδιαίο διάνυσμα το οποίο ορίζει την κατεύθυνση ενός υπερεπιπέδου σύμφωνα με την σχέση:

$$\rho = \xi \cdot \mathbf{x} = \xi_1 x_1 + \xi_2 x_2 + ... + \xi_n x_n$$

Η έκφραση (5) είναι και σ' αυτή την περίπτωση ο μετασχηματισμός Radon της f.

1.1.1.1 Παραδείγματα γραμμικού μετασχηματισμού

1. Θεωρούμε τη συνάρτηση $f(x,y) = e^{-x^2-y^2}$. Θα υπολογίσω τον μετασχηματισμό Radon. Από την εξίσωση (5), άμεσα έχουμε:

$$\breve{f}(\rho,\xi) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-x^2-y^2} \cdot \delta(\rho - \xi_1 x - \xi_2 y)dxdy$$

Θεωρώντας τον γραμμικό μετασχηματισμό $\begin{pmatrix} u \\ \upsilon \end{pmatrix} = \begin{pmatrix} \xi_1 & \xi_2 \\ -\xi_2 & \xi_1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}$ το διάνυσμα $\xi = (\xi_1, \xi_2)$ παραμένει μοναδιαίο και στον καινούργιο χώρο u-υ. Συνεπώς ο μετασχηματισμός γίνεται:

$$\breve{f}(\rho,\xi)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-u^2-\upsilon^2}\cdot\delta(\rho-u)d\upsilon du=e^{-\rho^2}\int_{-\infty}^{\infty}e^{-\upsilon^2}d\upsilon=\sqrt{\pi}e^{-\rho^2}$$

δηλαδή $R\left\{e^{-x^2-y^2}\right\}=\sqrt{\pi}e^{-\rho^2}$.

Παρατηρούμε ότι ο μετασχηματισμός όχι μόνο υπολογίζεται ακριβώς αλλά έχει και πολύ απλή μορφή. Επίσης παρατηρούμε ότι ο μετασχηματισμός δεν εξαρτάται από την γωνία φ (διάνυσμα ξ) αλλά μόνο από την απόσταση ρ από το κέντρο των αξόνων. Αυτό είναι απόλυτα αναμενόμενο λόγο της κυκλικής συμμετρίας της συνάρτησης f γύρω από την αρχή των αξόνων.

2. Ανάλογος είναι και ο χειρισμός για την τρισδιάστατη περίπτωση. Τότε ο κατάλληλος μετασχηματισμός είναι ο εξής:

$$\begin{pmatrix}u\\ \upsilon\\ w\end{pmatrix}=\begin{pmatrix}\xi_1 & \xi_2 & \xi_3\\ -\xi_1\xi_2/q & q & -\xi_2\xi_3/q\\ -\xi_3/q & 0 & \xi_1/q\end{pmatrix}\begin{pmatrix}x\\ y\\ z\end{pmatrix}$$

όπου $q=\sqrt{\xi_1^2+\xi_3^2}$ και λόγω του ότι το ξ είναι μοναδιαίο διάνυσμα, είναι $|\xi|=\sqrt{\xi_1^2+\xi_2^2+\xi_3^2}=1$. Αυτά οδηγούν στο αποτέλεσμα:

$$R\left\{e^{-x^2-y^2-z^2}\right\}=\pi e^{-\rho^2}$$

και με ανάλογους συλλογισμούς στη n-διάστατη περίπτωση είναι:

$$R\left\{e^{-x_1^2-x_2^2\ldots-x_n^2}\right\}=\left(\sqrt{\pi}\right)^{n-1}e^{-\rho^2}$$

1.1.2 Ιδιότητες του μετασχηματισμού

Ο γραμμικός μετασχηματισμός Radon έχει πολλές χρήσιμες ιδιότητες. Τις βασικότερες από αυτές θα αναπτύξουμε παρακάτω. Για πιο προχωρημένες ιδιότητες μπορεί κανείς να ανατρέξει στην βιβλιογραφία[1].

1.1.2.1. Γραμμικότητα

Σύμφωνα με βασικές ιδιότητες ολοκληρωμάτων έχουμε:

$$R\{c_1f+c_2g\}=\int\left(c_1f(\mathbf{x})+c_2g(\mathbf{x})\right)\delta(\rho-\xi\cdot\mathbf{x})d\mathbf{x}=$$
$$=\int c_1f(\mathbf{x})\delta(\rho-\xi\cdot\mathbf{x})d\mathbf{x}+\int c_2g(\mathbf{x})\delta(\rho-\xi\cdot\mathbf{x})d\mathbf{x}=c_1R\{f\}+c_2R\{g\}$$

δηλαδή $R\{c_1f+c_2g\}=c_1R\{f\}+c_2R\{g\}$.

1.1.2.2. Μεγέθυνση

Αν θεωρήσουμε $\mathbf{y}=\lambda\mathbf{x}$ έχουμε $d\mathbf{y}=\lambda^n d\mathbf{x}$ όπου n η διάσταση που μελετάμε. Έχουμε λοιπόν:

$$R\{f(\lambda\cdot\mathbf{x})\}=R\{f(\mathbf{y})\}=\int f(\mathbf{y})\delta\left(\rho-\frac{1}{\lambda}\xi\cdot\mathbf{y}\right)\frac{1}{\lambda^n}d\mathbf{y}=\frac{1}{\lambda^n}\breve{f}\left(\rho,\frac{\xi}{\lambda}\right)=$$

$$=\int f(\mathbf{y})\delta\left(\frac{1}{\lambda}(\lambda\rho-\xi\cdot\mathbf{y})\right)\frac{1}{\lambda^n}d\mathbf{y}=\frac{1}{\lambda^{n-1}}\breve{f}(\lambda\rho,\xi)$$

1.1.2.3. Μετατόπιση

Αν θεωρήσουμε την συνάρτηση $f(\mathbf{x}-\mathbf{a})$ όπου $\mathbf{a}$ ένα τυχαίο διάνυσμα μετατόπισης, $R\{f(\mathbf{x}-\mathbf{a})\}=\int f(\mathbf{x}-\mathbf{a})\delta(\rho-\xi\cdot\mathbf{x})d\mathbf{x}=\int f(\mathbf{y})\delta(\rho-\xi\cdot\mathbf{a}-\xi\cdot\mathbf{y})d\mathbf{y}$ όπου προφανώς $\mathbf{y}=\mathbf{x}-\mathbf{a}$. Δηλαδή $R\{f(\mathbf{x}-\mathbf{a})\}=\breve{f}(\rho-\xi\cdot\mathbf{a},\xi)$.

1.1.2.4. Γενικευμένοι μετασχηματισμοί σε δύο διαστάσεις

Με την βοήθεια μετασχηματισμών δύο διαστάσεων είναι δυνατόν να μετασχηματίσουμε συναρτήσεις που είναι δύσκολο να υπολογιστούν σε συναρτήσεις που είναι πιο εύκολο να υπολογιστούν π.χ. ελλείψεις σε κύκλους.

Αυτό που θέλουμε να υπολογίσουμε είναι ο μετασχηματισμός $R\{f(\mathbf{y})\}$ όπου $\mathbf{y}=\mathbf{A}\cdot\mathbf{x}$ όπου $\mathbf{A}$ ένας αντιστρέψιμος πίνακας γραμμικού μετασχηματισμού. Αν $\mathbf{B}=\mathbf{A}^{-1}$, τότε $\mathbf{x}=\mathbf{A}^{-1}\mathbf{y}=\mathbf{B}\mathbf{y}$ και $d\mathbf{x}=|\det\mathbf{B}|\,d\mathbf{y}$ λόγω του ότι η Jacobian του μετασχηματισμού είναι το μέτρο της ορίζουσας του $\mathbf{B}$.

Συνεπώς έχουμε:

$$R\{f(\mathbf{A}\mathbf{x})\}=\int f(\mathbf{A}\mathbf{x})\delta(\rho-\xi\cdot\mathbf{x})d\mathbf{x}=|\det\mathbf{B}|\int f(\mathbf{y})\delta(\rho-\xi\cdot\mathbf{B}y)d\mathbf{y}=|\det\mathbf{B}|\int f(\mathbf{y})\delta(\rho-\mathbf{B}^T\xi\cdot y)d\mathbf{y}$$

δηλαδή:

$$R\{f(\mathbf{A}\mathbf{x})\}=|\det\mathbf{B}|\,\breve{f}\left(\rho,\mathbf{B}^T\xi\right) \quad (6)$$

Παράδειγμα:

Έστω ότι έχουμε συνάρτηση $f(x,y)$ στο $\Re^2$ και πίνακα μετασχηματισμού $\mathbf{A}=\lambda\mathbf{I}=\lambda\begin{bmatrix}1&0\\0&1\end{bmatrix}$.

Τότε $\mathbf{B}=\mathbf{A}^{-1}=\frac{1}{\lambda}\mathbf{I}$ και $\det\mathbf{B}=\frac{1}{\lambda^2}=\det\mathbf{A}^{-1}$.

Συνεπώς από την (6) έχουμε:

$$R\{f(\mathbf{A}\mathbf{x})\}=R\{f(\lambda\mathbf{x})\}=\frac{1}{\lambda^2}\breve{f}\left(\rho,\frac{1}{\lambda}\xi\right)$$

ακριβώς όπως περιμέναμε την ιδιότητα 1.1.2.2.

1.1.2.5. Μετασχηματισμός παραγώγων

Δεδομένης μίας συνάρτησης $f(\mathbf{x})$ αυτό που ζητάμε είναι να υπολογιστεί ο μετασχηματισμός Radon της παραγώγου $\partial f/\partial x_k$. Αν ξαναθυμηθούμε τον ορισμό της παραγώγου, μπορούμε να δούμε ότι:

$$\frac{\partial f(\mathbf{x})}{\partial x_k} = \lim_{\varepsilon \to 0} \frac{f\left(\mathbf{x} + \frac{\varepsilon}{\xi_k}\right) - f(\mathbf{x})}{\frac{\varepsilon}{\xi_k}}$$

όπου $f\left(\mathbf{x} + \left(\varepsilon/\xi_k\right)\right)$ σημαίνει $f\left(x_1, x_2, ..., x_k + \varepsilon/\xi_k, ..., x_n\right)$ και ξ_k είναι ο k-οστός όρος του ξ. Αν πάρουμε τώρα τον μετ/μο Radon της παραπάνω έκφρασης και εφαρμόζοντας την ιδιότητα της μετατόπισης με $\mathbf{a} = \left(0, 0, ..., \varepsilon / \xi_k, ..., 0\right)$ έχουμε:

$$R\left\{\frac{\partial f(\mathbf{x})}{\partial x_k}\right\} = \lim_{\varepsilon \to 0} R\left\{\frac{f\left(\mathbf{x} + \left(\varepsilon/\xi_k\right)\right) - f(\mathbf{x})}{\left(\varepsilon/\xi_k\right)}\right\} = \lim_{\varepsilon \to 0}\left\{\xi_k \frac{R\left\{f\left(\mathbf{x} + \left(\varepsilon/\xi_k\right)\right)\right\} - R\left\{f(\mathbf{x})\right\}}{\varepsilon}\right\} =$$

$$= \xi_k \lim_{\varepsilon \to 0}\left\{\frac{\breve{f}(\rho + \varepsilon, \xi) - \breve{f}(\rho, \xi)}{\varepsilon}\right\} = \xi_k \frac{\partial \breve{f}(\rho, \xi)}{\partial \rho}$$

εκμεταλλευόμενοι γραμμικότητα και παραγωγίζοντας προς κάθε όρο έχουμε:

$$R\left\{\sum_{k=1}^{n} \alpha_k \frac{\partial f(\mathbf{x})}{\partial x_k}\right\} = \mathbf{a} \cdot \xi \frac{\partial \breve{f}(\rho, \xi)}{\partial \rho}$$

Γενικεύοντας μπορούμε να δούμε ότι αν παραγωγίσουμε παραγώγους δευτέρας τάξεως έχουμε:

$$R\left\{\sum_{l=1}^{n}\sum_{k=1}^{n} \alpha_k b_l \frac{\partial^2 f(\mathbf{x})}{\partial x_l \partial x_k}\right\} = (\mathbf{a} \cdot \xi)(\mathbf{b} \cdot \xi) \frac{\partial^2 \breve{f}(\rho, \xi)}{\partial \rho^2}$$

ενώ στη γενική περίπτωση αν έχουμε ένα γραμμικό τελεστή $\mathbf{L}\left(\partial/\partial x_1, ..., \partial/\partial x_n\right)$ με σταθερούς συντελεστές είναι:

$$R\{\mathbf{L}f\} = \mathbf{L}\left(\xi_1 \frac{\partial}{\partial \rho}, ..., \xi_n \frac{\partial}{\partial \rho}\right) \breve{f}(\rho, \xi)$$

Για παράδειγμα αν έχουμε έναν τελεστή $\mathbf{L} = a\frac{\partial}{\partial x_1} + b\frac{\partial^3}{\partial x_2 \partial x_3^2}$, τότε ο μετασχηματισμός Radon θα είναι:

$$R\{\mathbf{L}f\} = \left(a\xi_1 \frac{\partial}{\partial \rho} + b\xi_2 \xi_3^2 \frac{\partial^3}{\partial \rho^3}\right) \breve{f}(\rho, \xi)$$

1.1.2.6 Υπολογισμός παραγώγων μετασχηματισμού

Θα υπολογίσουμε τις παραγώγους του μετασχηματισμού Radon ως προς κάποια από τις μεταβλητές του ξ_k. Είναι:

$$\frac{\partial \breve{f}}{\partial \xi_k} = \int f(\mathbf{x}) \frac{\partial}{\partial \xi_k} \delta(\rho - \xi \cdot \mathbf{x}) d\mathbf{x}\,.$$

Ισχύει ότι από ιδιότητες της συνάρτησης δέλτα:

$$\frac{\partial \breve{f}}{\partial \xi_k}\delta(\rho-\xi\cdot\mathbf{x})=-x_k\frac{\partial}{\partial\rho}\delta(\rho-\xi\cdot\mathbf{x})$$

Συνεπώς:

$$\frac{\partial \breve{f}}{\partial \xi_k}=-\frac{\partial}{\partial\rho}\int x_k f(\mathbf{x})\delta(\rho-\xi\cdot\mathbf{x})d\mathbf{x}\Rightarrow\frac{\partial}{\partial\xi_k}R\{f(\mathbf{x})\}=-\frac{\partial}{\partial\rho}R\{x_k f(\mathbf{x})\}$$

1.1.2.7. Περιστροφή

Για την μελέτη της περιστροφής εξετάζουμε την δισδιάστατη περίπτωση[6]. Για την διευκόλυνσή μας θεωρούμε την συνάρτηση f σε πολικές συντεταγμένες, $f(x,y)=f(\rho,\phi)$. Ο μετασχηματισμός της f περιστραμένης κατά ϕ_0 έχει ως εξής:

$$R\{f(r,\phi-\phi_0)\}=\breve{f}(\rho,\theta)=\int_{-\infty}^{\infty}\int_{0}^{\pi}f(r,\phi-\phi_0)\delta(\rho-r\cos\phi\cos\theta-r\sin\phi\sin\theta)|r|d\phi dr\,.$$

Θέτοντας την νέα μεταβλητή $s=\phi-\phi_0$, έχουμε:

$$\int_{-\infty}^{\infty}\int_{0}^{\pi}f(r,\phi-\phi_0)\delta(\rho-r\cos\phi\cos\theta-r\sin\phi\sin\theta)|r|d\phi dr=$$
$$\int_{-\infty}^{\infty}\int_{0}^{\pi}f\phi(r,s)\delta(\rho-r\cos(\theta-s-\phi_0))|r|dsdr=\breve{f}(\rho,\theta-\phi_0)$$

1.1.2.8. Μετασχηματισμός συνέλιξης

Αν $\breve{g}=R\{g\}$ και $\breve{h}=R\{h\}$, θεωρούμε την συνάρτηση f της συνέλιξης των g και h:

$$f(\mathbf{x})=g*h=\int g(\mathbf{y})h(\mathbf{x}-\mathbf{y})d\mathbf{y}$$

Θέλουμε να υπολογίσουμε τον μετασχηματισμό Radon της παραπάνω συνέλιξης:

$$\breve{f}(\rho,\xi)=R\{f(\mathbf{x})\}=R\{g*h\}=\int d\mathbf{x}\int d\mathbf{y}\,g(\mathbf{y})h(\mathbf{x}-\mathbf{y})\delta(\rho-\xi\cdot\mathbf{x})=$$
$$=\int d\mathbf{y}\,g(\mathbf{y})\int d\mathbf{x}\,h(\mathbf{x}-\mathbf{y})\delta(\rho-\xi\cdot\mathbf{x})$$

Θέτοντας $\mathbf{z}=\mathbf{x}-\mathbf{y}$, έχουμε:

$$\breve{f}(\rho,\xi)=\int d\mathbf{y}\,g(\mathbf{y})\int d\mathbf{z}\,h(\mathbf{z})\delta(\rho-\xi\cdot\mathbf{y}-\xi\cdot\mathbf{z})=\int d\mathbf{y}\,g(\mathbf{y})\breve{h}(\rho-\xi\cdot\mathbf{y},\xi)$$

Όμως θέτοντας $s=\xi\cdot\mathbf{y}$ και από τις ιδιότητες της συνάρτησης δέλτα έχουμε:

$$\breve{h}(\rho-\xi\cdot\mathbf{y},\xi)=\int_{-\infty}^{\infty}ds\,\breve{h}(\rho-s,\xi)\delta(s-\xi\cdot\mathbf{y})$$

Συνεπώς:

$$\breve{f}(\rho,\xi)=\int d\mathbf{y}\, g(\mathbf{y})\int_{-\infty}^{\infty} ds\, \breve{h}(\rho-s,\xi)\,\delta(s-\xi\cdot\mathbf{y})=\int_{-\infty}^{\infty} ds\, \breve{h}(\rho-s,\xi)\int d\mathbf{y}\, g(\mathbf{y})\,\delta(s-\xi\cdot\mathbf{y})=$$

$$=\int_{-\infty}^{\infty}\breve{g}(s,\xi)\;\breve{h}(\rho-s,\xi)\,ds=\breve{g}*\breve{h}$$

Δηλαδή $R\{g*h\}=R\{g\}*R\{h\}$. Βλέπουμε δηλαδή ότι ο μετ/μος Radon της συνέλιξη δύο συναρτήσεων είναι ίσος με την συνέλιξη των μετ/ων τους σε αντίθεση με τον μετ/μο Fourier όπου ο μετ/μος της συνέλιξης είναι ίσος με το γινόμενο των μετ/ων συναρτήσεων.

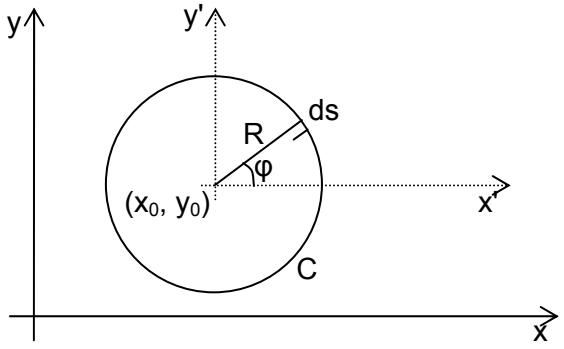
1.1.2.9. Περιοδικότητα

$$R[f(r,\phi)]=\breve{f}(r_0,\phi_0)=\breve{f}(r_0,\phi_0+2\pi k),k\in\Im$$

1.2 Κυκλικός μετασχηματισμός Radon.

Ο κυκλικός μετασχηματισμός Radon μπορεί να οριστεί με δύο διαφορετικούς τρόπους. Ο ένας περιορίζεται στις δύο διαστάσεις ενώ ο άλλος γενικεύεται σε περισσότερες. Ο μετασχηματισμός Radon έχει πολλές ενδιαφέρουσες ιδιότητες που μπορούν να απλοποιήσουν τον υπολογισμό του σε αρκετές περιπτώσεις. Το μεγαλύτερο μέρος των πληροφοριών αυτής της ενότητας έχει ως πηγή την εργασία του επιβλέποντος καθηγητή Θ. Αλεξόπουλου για την επίλυση προβλημάτων φυσικής υψηλών ενεργειών.

1.2.1 Ορισμός



Ο ορισμός του μετασχηματισμού Radon είναι ο εξής:

$$R[f]=\breve{f}(R,x_0,y_0)=\oint_C f(x,y)ds$$

Όπου C ο κύκλος με κέντρο χ_0, y_0 και ακτίνα R. Από τις γνωστές μας σχέσεις, ο κύκλος μπορεί να εκφραστεί σε παραμετρική μορφή:

$$\begin{aligned}x&=x_0+x'=x_0+R\cos\phi\\ y&=y_0+y'=y_0+R\sin\phi\end{aligned},\phi\in[0,2\pi)$$

Οπότε ο μετασχηματισμός γίνεται:

$$R[f]=\int_0^{2\pi} f(x_0+R\cos\phi,y_0+R\sin\phi)Rd\phi=R\int_0^{2\pi} f(x_0+R\cos\phi,y_0+R\sin\phi)d\phi \quad (7)$$

Η σχέση αυτή μας δίνει μία πολύ ενδιαφέρουσα πληροφορία. Ότι ο μετασχηματισμός Radon είναι ανάλογος της ακτίνας του κύκλου στον οποίο ολοκληρώνουμε.

1.2.1.1 Παραδείγματα κυκλικού μετασχηματισμού

1. $f(x,y) = e^{-x^2-y^2}$

$$R[f] = \breve{f} = R\int_0^{2\pi} e^{-(x_0+R\cos\phi)^2-(y_0+R\sin\phi)^2}\,d\phi = Re^{-x_0^2-y_0^2-R^2}\int_0^{2\pi} e^{-2R(x_0\cos\phi+y_o\sin\phi)}d\phi =^*$$

$$2\pi Re^{-x_0^2-y_0^2-R^2} I_0(2R\sqrt{x_0^2+y_0^2})$$

$$^* \int_0^{2\pi} e^{p\cos x+q\sin x}\cos(a\cos x + b\sin x - mx)dx =$$

$$\left\{\pi[(b-p)^2+(a-q)^2]^{-m/2}\right\}\left\{(A+iB)^{m/2} I_m\left(\sqrt{C-iD}\right)+(A-iB)^{m/2} I_m\left(\sqrt{C+iD}\right)\right\}$$

για $(b-p)^2+(a+q)^2 > 0, m = 0,1,2,3...$

όπου

$$A = p^2 - q^2 + a^2 - b^2$$
$$B = 2pq + 2ab$$
$$C = p^2 + q^2 - a^2 - b^2$$
$$D = -2(ap+bq)$$

Σύμφωνα με τον παραπάνω τύπο θέλουμε να υπολογίσουμε το

$$\int_0^{2\pi} e^{-2Rx_0\cos\phi-2Ry_o\sin\phi}d\phi$$

Δηλαδή έχουμε $p = -2Rx_0$, $q = -2Ry_0$, $a = b = 0$, συνεπώς $(b-p)^2+(a-q)^2 = = b^2+a^2 = 4R^2(x_0^2+y_0^2) > 0$.

Επιπλέον είναι:

$$A = p^2 - q^2 = (2R)^2(x_0^2 - y_0^2)$$
$$B = 2pq = 8R^2 x_0 y_0$$
$$C = p^2 + q^2 = (2R)^2(x_0^2 + y_0^2)$$
$$D = -2(ap+bq) = 0$$

Συνεπώς $\int_0^{2\pi} e^{-2Rx_0\cos\phi-2Ry_o\sin\phi}d\phi = 2\pi I_o\left(2R\sqrt{x_0^2+y_0^2}\right) = 2\pi I_o\left(i2R\sqrt{x_0^2+y_0^2}\right)$, επειδή $I_o(x) = I_o(ix)$

2. $f(x,y) = e^{-(x-x_0)^2-(y-y0)^2}$

$$R[e^{-(x-x_0)^2-(y-y0)^2}] = R\int_0^{2\pi} e^{-(\cancel{x_0}+R\cos\phi-\cancel{x_0})^2-(\cancel{y_0}+R\sin\phi-\cancel{y_0})^2}d\phi = 2\pi Re^{-R^2}$$

1.2.1.2 Γενικευμένος κυκλικός μετασχηματισμός Radon

Ορίζουμε τον γενικευμένο κυκλικό μετασχηματισμό Radon:

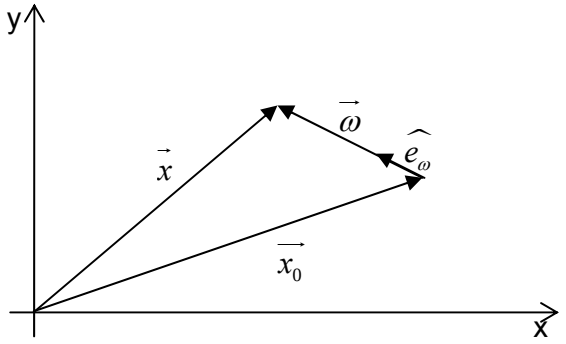
$$R[f] = \breve{f} = \int_S f\left(\vec{x}\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right)d\vec{x}$$

Όπου S το πεδίο ορισμού της συνάρτησης f. Όπως μπορούμε να παρατηρήσουμε η παραπάνω συνάρτηση «επιλέγει» την περιοχή πάνω στην οποία γίνεται η ολοκλήρωση μέσω μίας συνάρτησης δ αντί για τις παραμέτρους της f όπως στον προηγούμενο ορισμό. Έτσι στην 2-διάστατη περίπτωση η ολοκλήρωση γίνεται και πάλι πάνω στον κύκλο $\left|\vec{x} - \overrightarrow{x_0}\right| = R$ ενώ σε τρεις διαστάσεις η ολοκλήρωση γίνεται πάνω σε σφαίρα ακτίνας R και κέντρου $\overrightarrow{x_0}$. Αντίστοιχα γενικεύεται και σε περισσότερες των τριών διαστάσεων.

Αν θεωρήσουμε σύστημα συντεταγμένων με αρχή το $\overrightarrow{x_0}$, τότε μπορούμε να ορίσουμε την απόσταση $\vec{\omega}$ τυχαίου σημείου $\vec{x}$ από το κέντρο των νέων αξόνων $\vec{\omega} = \vec{x} - \overrightarrow{x_0} \Rightarrow \vec{x} = \overrightarrow{x_0} + \vec{\omega} \Rightarrow d\vec{x} = d\vec{\omega}$. Με αυτό τον τρόπο ο μετασχηματισμός γίνεται:

$$R[f] = \breve{f} = \int f\left(\overrightarrow{x_0} + \vec{\omega}\right)\delta\left(R - \omega\right)\omega d\omega d\phi = \int f\left(\overrightarrow{x_0} + \mathrm{R}\,\hat{\mathrm{e}}_\omega\right)Rd\phi = R\int f\left(\overrightarrow{x_0} + \mathrm{R}\,\hat{\mathrm{e}}_\omega\right)d\phi$$

όπου $\widehat{e_\omega}$ είναι το μοναδιαίο διάνυσμα κατά τη διεύθυνση του $\vec{\omega}$ όπως φαίνεται στην Εικόνα 5.



Εικόνα 5. Ορισμός του διανύσματος $\widehat{e_\omega}$

Ας δούμε πως χειριζόμαστε το πρώτο παράδειγμα με τον νέο ορισμό:

$$f(x,y) = e^{-x^2-y^2} = e^{-\left|\vec{x}\right|^2}$$

$$f = R\int e^{-\left|\overrightarrow{x_0}+\mathrm{R}\,\hat{\mathrm{e}}\omega\right|^2}d\phi = R\int e^{-\left(\overrightarrow{x_0}^2+\mathrm{R}^2+2R\overrightarrow{x_0}\hat{\mathrm{e}}\omega\right)}d\phi = \mathrm{Re}^{-x_0^2-y_0^2-R^2}\int_0^{2\pi} e^{2R(x_0\cos\phi+y_0\sin\phi)}d\phi\,,$$

όπου $\overrightarrow{x_0} = x_0\hat{i} + y_0\hat{j}$, $\widehat{e_\omega} = \cos\phi\cdot\hat{i} + \sin\phi\cdot\hat{j}$. Φτάσαμε δηλαδή στο ίδιο αποτέλεσμα όπως ήταν αναμενόμενο.

1.2.2 Ιδιότητες του μετασχηματισμού

Όπως μπορεί κανείς να παρατηρήσει πολλές από τις ιδιότητες του γραμμικού μετ/μου Radon ισχύουν και για τον κυκλικό. Θα δούμε όμως ότι σε αρκετές περιπτώσεις ο τρόπος απόδειξης

διαφέρει αρκετά. Ο κυκλικός μετασχηματισμός έχει μερικές επιπλέον ενδιαφέρουσες ιδιότητες π.χ. περιοδικότητα.

1.2.2.1. Γραμμικότητα

$$R[C_1 f + C_2 g] = \int_S \left(C_1 f\left(\vec{x}\right) + C_2 g\left(\vec{x}\right)\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x}$$

$$= \int_S C_1 f\left(\vec{x}\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x} + \int_S C_2 g\left(\vec{x}\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x} =$$

$$C_1 R[f] + C_2 R[g]$$

δηλαδή $R[C_1 f + C_2 g] = C_1 R[f] + C_2 R[g]$.

1.2.2.2. Μεγέθυνση

Για συντελεστή μεγέθυνσης λ > 0 ο μετασχηματισμός Radon θα είναι:

$$\breve{f}\left(R', \overrightarrow{x_0}'\right) = R\left[f\left(\lambda\vec{x}\right)\right] = \int_S f\left(\lambda\vec{x}\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x}.$$

Θεωρώ $\lambda\vec{x} = \vec{y} \Rightarrow \vec{x} = \vec{y}/\lambda$. Τότε έχουμε:

$$R\left[f\left(\lambda\vec{x}\right)\right] = R\left[f\left(\vec{y}\right)\right] = \int_S f\left(\vec{y}\right)\delta\left(R - \left|\vec{y}/\lambda - \overrightarrow{x_0}\right|\right)\frac{d\vec{y}}{\lambda} =$$

$$\int_S f\left(\vec{y}\right)\delta\left(\frac{1}{\lambda}\left(\lambda R - \left|\vec{y} - \lambda\overrightarrow{x_0}\right|\right)\right)\frac{d\vec{y}}{\lambda} = \int_S f\left(\vec{y}\right)\not{\lambda}\,\delta\left(\left(\lambda R - \left|\vec{y} - \lambda\overrightarrow{x_0}\right|\right)\right)\frac{d\vec{y}}{\not{\lambda}} =$$

$$\breve{f}\left(\lambda R, \lambda\overrightarrow{x_0}\right)$$

Παράδειγμα

Να υπολογιστεί το $R\left[e^{-\frac{(x-x_0)^2}{2\sigma^2} - \frac{(y-y_0)^2}{2\sigma^2}}\right]$.

Από το δεύτερο παράδειγμα ξέρουμε ότι $R[e^{-(x-x_0)^2-(y-y0)^2}] = 2\pi R e^{-R^2}$. Αν θεωρήσουμε $\lambda = \frac{1}{\sqrt{2}\sigma}$, τότε το ζητούμενο γίνεται:

$R\left[e^{-\frac{(x-x_0)^2}{2\sigma^2} - \frac{(y-y_0)^2}{2\sigma^2}}\right] = R\left[e^{-\left(\lambda(x-x_0)\right)^2 - \left(\lambda(y-y_0)\right)^2}\right]$ το οποίο σύμφωνα με την τελευταία ιδιότητα είναι ίσο με $\breve{f} = \sqrt{2}\frac{\pi}{\sigma} R e^{-\frac{R^2}{2\sigma^2}}$.

1.2.2.3. Μετατόπιση

$$R\left[f\left(\vec{x} - \vec{a}\right)\right] = \int_S f\left(\vec{x} - \vec{a}\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x}.$$

Αν θεωρήσουμε $\vec{x} - \vec{a} = \vec{y} \Rightarrow \vec{x} = \vec{y} + \vec{a} \Rightarrow d\vec{x} = d\vec{y}$, τότε έχουμε

$$R\left[f\left(\vec{x} - \vec{a}\right)\right] = \int_S f\left(\vec{y}\right)\delta\left(R - \left|\vec{y} - \left(\overrightarrow{x_0} - \vec{a}\right)\right|\right) d\vec{x} = \breve{f}\left(R, \overrightarrow{x_0} - \vec{a}\right).$$

Με $\vec{a}' = -\vec{a}$ έχουμε επίσης $R\left[f\left(\vec{x} + \vec{a}\right)\right] = \breve{f}\left(R, \overrightarrow{x_0} + \vec{a}\right)$.

Με $\vec{a}' = \overrightarrow{x_0}$ έχουμε προφανώς $R\left[f\left(\vec{x} - \overrightarrow{x_0}\right)\right] = \breve{f}\left(R, \vec{0}\right)$

1.2.2.4. Γενικευμένοι μετασχηματισμοί σε δύο διαστάσεις

Θεωρούμε ότι $R\left[f\left(\vec{x}\right)\right] = \breve{f}\left(R, \overrightarrow{x_0}\right)$, και θέλουμε να υπολογίσουμε το $R\left[f\left(\underline{\underline{A}}\vec{x}\right)\right]$ όπου $\underline{\underline{A}}$ ένας πίνακας μετασχηματισμού $\vec{x} \to \underline{\underline{A}}\vec{x}$.

Θεωρούμε $\underline{\underline{A}}\vec{x} = \vec{y} \Rightarrow \vec{x} = \underline{\underline{A}}^{-1}\vec{y} \overset{B=A^{-1}}{\Rightarrow} \vec{x} = \underline{\underline{B}}\vec{y} \Rightarrow d\vec{x} = \left|\det \underline{\underline{B}}\right| d\vec{y}$.

$$R\left[f\left(\underline{\underline{A}}\vec{x}\right)\right] = \int_S f\left(\vec{y}\right)\delta\left(R - \left|\underline{\underline{B}}\vec{y} - \overrightarrow{x_0}\right|\right)\left|\det \underline{\underline{B}}\right| d\vec{y} =$$

$$\left|\det \underline{\underline{B}}\right| \int_S f\left(\vec{y}\right)\delta\left(R - \left|\underline{\underline{B}}\left(\underbrace{\vec{y} - \underline{\underline{B}}^{-1}\overrightarrow{x_0}}_{\vec{z}}\right)\right|\right) d\vec{y}.$$

Θέλουμε να υπολογίσουμε το $\underline{\underline{B}}z$. Έχουμε $\underline{\underline{B}}\vec{z} \cdot \underline{\underline{B}}\vec{z} = \left|\underline{\underline{B}}\vec{z}\right|^2 \Rightarrow \left(\underline{\underline{B}}^T \underline{\underline{B}}\vec{z}\right)\vec{z} = \lambda\left|\vec{z}\right|^2$.

Αν θεωρήσουμε πίνακα της μορφής $\underline{\underline{B}} = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}$, τότε είναι:

$$\left(\underline{\underline{B}}^T \underline{\underline{B}}\vec{z}\right)\vec{z} = \begin{bmatrix} \alpha & \gamma \\ \beta & \delta \end{bmatrix}\begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} =$$

$$\left(\alpha^2 + \gamma^2\right)z_1^2 + \left(\beta^2 + \delta^2\right)z_2^2 + 2z_1 z_2\left(\alpha\beta - \gamma\delta\right) = \lambda\left|\vec{z}\right|^2$$

$$\alpha^2 + \gamma^2 = \beta^2 + \delta^2 = \lambda,\ \alpha\beta + \gamma\delta = 0 \Rightarrow \gamma = -\frac{\alpha\beta}{\delta}, \alpha^2 + \frac{\alpha^2\beta^2}{\delta^2} = \beta^2 + \delta^2.$$

Συνεπώς

$$R\left[f\left(\underline{\underline{A}}\vec{x}\right)\right] = \left|\det \underline{\underline{B}}\right| \int_{\Re^2} f\left(\vec{y}\right)\delta\left(R - \sqrt{\lambda}\left|\vec{y} - \underline{\underline{B}}^{-1}\overrightarrow{x_0}\right|\right) d\vec{y} = \left|\det \underline{\underline{B}}\right| \int_{\Re^2} f\left(\vec{y}\right)\delta\left(\frac{R}{\sqrt{\lambda}} - \left|\vec{y} - \underline{\underline{B}}^{-1}\overrightarrow{x_0}\right|\right)\frac{d\vec{y}}{\sqrt{\lambda}} =$$

$$= \frac{\left|\det \underline{\underline{B}}\right|}{\sqrt{\lambda}}\breve{f}\left(\frac{R}{\sqrt{\lambda}}, \underline{\underline{A}}\overrightarrow{x_0}\right) \Rightarrow R\left[f\left(\underline{\underline{A}}\vec{x}\right)\right] = \frac{\left|\det \underline{\underline{A}}^{-1}\right|}{\sqrt{\lambda}}\breve{f}\left(\frac{R}{\sqrt{\lambda}}, \underline{\underline{A}}\overrightarrow{x_0}\right)$$

Παράδειγμα

$\underline{\underline{A}} = k\underline{\underline{I}} = k\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Τότε $\underline{\underline{B}} = \underline{\underline{A}}^{-1} = \frac{1}{k}\underline{\underline{I}}$ και $\det \underline{\underline{B}} = \frac{1}{k} = \det \underline{\underline{A}}^{-1}$.

Επίσης $\lambda = \alpha^2 + \gamma^2 = \frac{1}{k^2} \Rightarrow \sqrt{\lambda} = \frac{1}{k}$.

Συνεπώς $R\left[f\left(k\underline{\underline{I}}\vec{x}\right)\right] = R\left[f\left(k\vec{x}\right)\right] = \frac{|1/k|}{1/k}\breve{f}\left(kR, k\overrightarrow{x_0}\right) \Rightarrow R\left[f\left(k\vec{x}\right)\right] = \breve{f}\left(kR, k\overrightarrow{x_0}\right)$, ακριβώς όπως περιμέναμε από τη δεύτερη ιδιότητα.

1.2.2.5. Υπολογισμός παραγώγων

$\overrightarrow{x_0} = (x_0, y_0) = (\xi_1, \xi_2)$. $R\left[f\left(\vec{x}\right)\right] = \breve{f}\left(R, \overrightarrow{x_0}\right) = \int\limits_{\mathbb{R}^2} f\left(\vec{x}\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x}$.

$$\frac{\partial \breve{f}}{\partial \xi_1} = \int\limits_{\mathbb{R}^2} f\left(\vec{x}\right)\frac{\partial}{\partial \xi_1}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x}$$

$$\frac{\partial}{\partial R}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) = -\frac{\partial}{\partial \xi_1}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right)\frac{\partial \xi_1}{\partial\left|\vec{x} - \overrightarrow{x_0}\right|} \Rightarrow$$

$$\frac{\partial}{\partial R}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) = -\frac{\partial}{\partial \xi_1}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right)$$

$$\frac{\partial\left|\vec{x} - \overrightarrow{x_0}\right|}{\partial \xi_1} = \frac{\partial\sqrt{\left(x - \xi_1\right)^2 + \left(y - \xi_2\right)^2}}{\partial \xi_1} = \frac{1}{\cancel{2}\sqrt{\left(x - \xi_1\right)^2 + \left(y - \xi_2\right)^2}}\cancel{2}\left(x - \xi_1\right)(-1) = -\frac{x - \xi_1}{\left|\vec{x} - \overrightarrow{x_0}\right|}$$

Συνεπώς: $\frac{\partial}{\partial \xi_1}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) = \frac{x - \xi_1}{\left|\vec{x} - \overrightarrow{x_0}\right|}\frac{\partial}{\partial R}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right)$.

Έστω $\vec{x} = (x_1, x_2), \overrightarrow{x_0} = (\xi_1, \xi_2)$. Τότε είναι:

$$\frac{\partial \breve{f}}{\partial \xi_1} = \frac{\partial}{\partial R}\int\limits_{\mathbb{R}^2} f\left(\vec{x}\right)\frac{x_1 - \xi_1}{\left|\vec{x} - \overrightarrow{x_0}\right|}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x} = \frac{\partial}{\partial R}R\left[\frac{x_1 - \xi_1}{\left|\vec{x} - \overrightarrow{x_0}\right|} f\left(\vec{x}\right)\right] =$$

$$= \frac{\partial}{\partial R}R\left[x_1 \cdot \frac{f\left(\vec{x}\right)}{\left|\vec{x} - \overrightarrow{x_0}\right|}\right] - \xi_1\frac{\partial}{\partial R}R\left[\frac{f\left(\vec{x}\right)}{\left|\vec{x} - \overrightarrow{x_0}\right|}\right]$$

Αντίστοιχα, $\frac{\partial \breve{f}}{\partial \xi_2} = \frac{\partial}{\partial R}R\left[\frac{x_2 - \xi_2}{\left|\vec{x} - \overrightarrow{x_0}\right|} f\left(\vec{x}\right)\right]$.

$$\sum_{k=1}^{2} a_k \frac{\partial \breve{f}}{\partial \xi_k} = \frac{\partial}{\partial R}R\left[\frac{\vec{a}\cdot\left(\vec{x} - \overrightarrow{x_0}\right)}{\left|\vec{x} - \overrightarrow{x_0}\right|} f\left(\vec{x}\right)\right] \text{ ή } \left(\vec{a}\frac{\partial}{\partial \overrightarrow{x_0}}\right)R\left[f\left(\vec{x}\right)\right] = \frac{\partial}{\partial R}R\left[\frac{\vec{a}\cdot\left(\vec{x} - \overrightarrow{x_0}\right)}{\left|\vec{x} - \overrightarrow{x_0}\right|} f\left(\vec{x}\right)\right]$$

Παράγωγος δευτέρας τάξης

$$\frac{\partial^2 \breve{f}}{\partial \xi_2 \partial \xi_1} = \frac{\partial}{\partial R}\int f\left(\vec{x}\right)\frac{x_1 - \xi_1}{\left|\vec{x} - \overrightarrow{x_0}\right|^2}\left(-\frac{\partial\left|\vec{x} - \overrightarrow{x_0}\right|}{\partial \xi_2}\right)\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x} +$$

$$+\frac{\partial}{\partial R}\int f\left(\vec{x}\right)\frac{x_1 - \xi_1}{\left|\vec{x} - \overrightarrow{x_0}\right|}\frac{\partial}{\partial \xi_2}\delta\left(R - \left|\vec{x} - \overrightarrow{x_0}\right|\right) d\vec{x} =$$

$$= \frac{\partial}{\partial R}\int f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)\left(x_2-\xi_2\right)}{\left|\vec{x}-\vec{x_0}\right|^3}\delta\left(R-\left|\vec{x}-\vec{x_0}\right|\right)d\vec{x}+$$

$$+\frac{\partial^2}{\partial R^2}\int f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)\left(x_2-\xi_2\right)}{\left|\vec{x}-\vec{x_0}\right|^2}\delta\left(R-\left|\vec{x}-\vec{x_0}\right|\right)d\vec{x}\Rightarrow$$

$$\frac{\partial^2 \breve{f}}{\partial\xi_2\partial\xi_1}=\frac{\partial}{\partial R}R\left[f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)\left(x_2-\xi_2\right)}{\left|\vec{x}-\vec{x_0}\right|^3}\right]+\frac{\partial^2}{\partial R^2}R\left[f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)\left(x_2-\xi_2\right)}{\left|\vec{x}-\vec{x_0}\right|^2}\right]=\frac{\partial^2 \breve{f}}{\partial\xi_1\partial\xi_2}$$

Συνεπώς:

$$\frac{\partial^2 \breve{f}}{\partial\xi_2\partial\xi_1}=\frac{\partial^2 \breve{f}}{\partial\xi_1\partial\xi_2}=\frac{\partial}{\partial R}\left\{R\left[f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)\left(x_2-\xi_2\right)}{\left|\vec{x}-\vec{x_0}\right|^3}\right]+\frac{\partial}{\partial R}R\left[f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)\left(x_2-\xi_2\right)}{\left|\vec{x}-\vec{x_0}\right|^2}\right]\right\}$$

Παράγωγος δεύτερης τάξεως μέρος 2ο

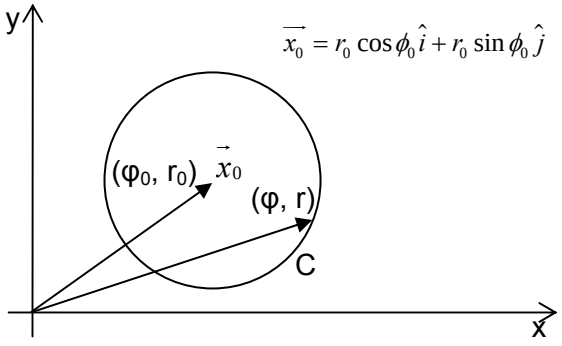
$$\frac{\partial^2 \breve{f}}{\partial\xi_1^{\,2}}=-\frac{\partial}{\partial R}\int f\left(\vec{x}\right)\frac{1}{\left|\vec{x}-\vec{x_0}\right|}\delta\left(R-\left|\vec{x}-\vec{x_0}\right|\right)d\vec{x}+\frac{\partial}{\partial R}\int f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)^2}{\left|\vec{x}-\vec{x_0}\right|^3}\delta\left(R-\left|\vec{x}-\vec{x_0}\right|\right)d\vec{x}+$$

$$\frac{\partial^2}{\partial R^2}\int f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)^2}{\left|\vec{x}-\vec{x_0}\right|^2}\delta\left(R-\left|\vec{x}-\vec{x_0}\right|\right)d\vec{x}\Rightarrow$$

$$\frac{\partial^2 \breve{f}}{\partial\xi_1^{\,2}}=-\frac{\partial}{\partial R}R\left[\frac{f\left(\vec{x}\right)}{\left|\vec{x}-\vec{x_0}\right|}\right]+\frac{\partial}{\partial R}R\left[f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)^2}{\left|\vec{x}-\vec{x_0}\right|^3}\right]+\frac{\partial^2}{\partial R^2}R\left[f\left(\vec{x}\right)\frac{\left(x_1-\xi_1\right)^2}{\left|\vec{x}-\vec{x_0}\right|^2}\right]$$

1.2.2.6. Περιστροφή

Θα εξετάσουμε τον κυκλικό μετασχηματισμό Radon σε πολικές συντεταγμένες.



$$R[f]=\breve{f}\left(R,r_0,\phi_0\right)=\int f(x,y)\delta\left(R-\left|\vec{x}-\vec{x_0}\right|\right)dx,dy$$

$$\vec{x}=r\cos\phi\hat{i}+r\sin\phi\,\hat{j},\quad \begin{matrix} x=r\cos\phi \\ y=r\sin\phi \end{matrix}\Big|\begin{matrix} x_0=r_0\cos\phi_0 \\ y_0=r_0\sin\phi_0 \end{matrix}$$

$$\left|\vec{x}-\overrightarrow{x_0}\right| = \sqrt{\left(x-x_0\right)^2+\left(y-y_0\right)^2} = \sqrt{r^2+r_0^2-2rr_0\cos\left(\phi-\phi_0\right)}$$

Συνεπώς:

$$\breve{f}\left(R,r_0,\phi_0\right) = \int f_p(r,\phi)\delta\left(R-\sqrt{r^2+r_0^2-2rr_0\cos\left(\phi-\phi_0\right)}\right)\cdot r\cdot dr,d\phi$$

Αν τώρα περιστραφεί κατά μία γωνία θ_0, έχουμε $\phi\to\phi+\theta_0$. Τότε,

$$R\left[f\left(\vec{x}\right)\right] = R\left[f\left(r,\phi+\theta_0\right)\right] = \int f\left(r,\phi+\theta_0\right)\delta\left(R-\sqrt{r^2+r_0^2+2rr_0\cos\left(\phi-\phi_0\right)}\cdot r\cdot drd\phi\right)$$

θέτω $\phi+\theta_0\to\omega\Rightarrow d\phi=d\omega,\phi=\omega-\theta_0$. Τότε,

$$R\left[f\left(\vec{x}\right)\right] = \int f\left(r,\omega\right)\delta\left(R-\sqrt{r^2+r_0^2+2rr_0\cos\left(\omega-\left(\theta_0+\phi_0\right)\right)}\cdot r\cdot drd\phi\right)$$

Τελικά:

$R\left[f\left(r,\theta_0+\phi\right)\right] = \breve{f}\left(R,r_0,\theta_0+\phi_0\right)$, όπου $R\left[f\left(r,\phi\right)\right] = \breve{f}\left(R,r_0,\phi_0\right)$

1.2.2.7. Περιοδικότητα

$$R[f\left(r,\phi\right)] = \breve{f}\left(R,r_0,\phi_0\right) = \breve{f}\left(R,r_0,\phi_0+2\pi k\right), k\in\Im$$

1.2.2.8. Διατήρηση μάζας

$$M = \int_{\Re^2} f\left(\vec{x}\right)d\vec{x} = \int_0^\infty \breve{f}\left(R,\overrightarrow{x_0}\right)dR = \int_0^\infty dR\int_{\Re^2} d\vec{x}f\left(\vec{x}\right)\delta\left(R-\left|\vec{x}-\overrightarrow{x_0}\right|\right) =$$

$$= \int_{\Re^2} f\left(\vec{x}\right)d\vec{x}\underbrace{\int_0^\infty dR\delta\left(R-\left|\vec{x}-\overrightarrow{x_0}\right|\right)}_{1} = \int_{\Re^2} f\left(\vec{x}\right)d\vec{x} = M$$

1.3 Γενίκευση του μετ/μού Radon και μετ/μος Hough

Στις προηγούμενες ενότητες μελετήσαμε τον γραμμικό και τον κυκλικό μετασχηματισμό Radon. Κατά αντίστοιχο τρόπο μπορεί να οριστεί ο ελλειπτικός[9] μετασχηματισμός Radon, μετασχηματισμός πάνω σε κωνικές τομές[2] ή και μετασχηματισμός με βάση οποιοδήποτε γεωμετρικό τόπο πάνω στον οποίο ολοκληρώνουμε. Συγκεκριμένα μία γενική μορφή του μετασχηματισμού Radon μία πραγματικής συνάρτησης $f\left(\mathbf{x}\right)$ με $\mathbf{x}\in\Re^m$, μπορεί να οριστεί ως[2]:

$$R\left\{f\right\} = \breve{f}\left(\mathbf{u},c\right) = \int f\left(\mathbf{x}\right)\delta\left(I\left(\mathbf{x},\mathbf{u}\right)-c\right)d\mathbf{x}$$

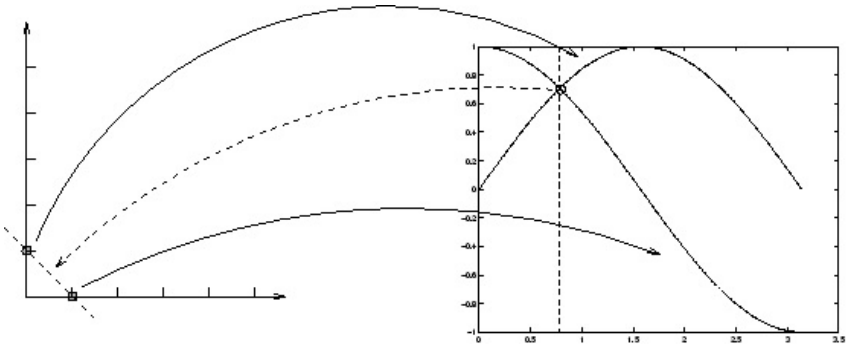
όπου $\mathbf{u}\in\Re^n$, $c\in\Re$ και $I\left(\mathbf{x},\mathbf{u}\right)$ πραγματική συνεχής συνάρτηση. Προφανώς οι ιδιότητες σε κάθε εκδοχή του μετασχηματισμού είναι δυνατόν να έχουν πάρα πολύ μεγάλες διαφορές και θα πρέπει να εξετάζονται από την αρχή.

[2] The general quadratic Radon transform, Koen Denecker, Jeroen Van Overloop and Frank Sommen, 1998

Ο Hough το 1962[3] δημιούργησε ένα μετασχηματισμό για την ανίχνευση ευθειών γραμμών σε ψηφιακές εικόνες. Ο μετασχηματισμός αυτός αποτελεί ειδική περίπτωση (διακριτή) του μετασχηματισμού Radon. Συγκεκριμένα ο Hough παρατήρησε ότι σε ψηφιακές εικόνες μετά το edge tracing τα περισσότερα σημεία (pixels) είναι συνήθως μηδενικά, με ελάχιστες εξαιρέσεις που είναι μονάδες. Συνεπώς ο μετ/μος Radon σπαταλά υπολογιστική ισχύ για να αθροίζει μηδενικά. Ακριβώς εκεί βασίζεται ο μετασχηματισμός Hough ο οποίος εξοικονομεί υπολογιστική ισχύ μετασχηματίζοντας με ειδικό τρόπο μόνο αν στον x-y χώρο η τιμή ενός σημείου είναι μη-μηδενική. Δηλαδή εκεί που ο μετασχηματισμός Radon υπολογίζει την τιμή ενός σημείου του χώρου Radon ως ολοκλήρωμα πάνω στον x-y χώρο, ο μετασχηματισμός Hough υπολογίζει για κάθε μη μηδενικό σημείο του x-y χώρου πως αυτός επηρεάζει τον χώρο Hough (χαράζοντας τις αντίστοιχες καμπύλες για κάθε σημείο).

Η αρχή με βάση την οποία γίνεται η ανίχνευση γραμμών με τον μετασχηματισμό Radon/Hough είναι η εξής. Αν αναζητήσουμε τον μετασχηματισμό Radon ενός μεμονωμένου σημείου (pixel) το οποίο ορίζεται ως $f(x,y)=\delta(x-x_0)\delta(y-y_0)$ θα δούμε ότι αυτός έχει μία ημιτονοειδή μορφή στο φ-ρ επίπεδο της μορφής $\rho = x_0\cos\phi + y_0\sin\phi$.

Επιπλέον όλα τα σημεία που ανήκουν σε μία ευθεία που ορίζεται από την απόσταση ρ_0 και γωνία φ_0 στον x-y χώρο αντιστοιχίζονται στον φ-ρ χώρο σε ημιτονοειδείς καμπύλες που όλες συμβάλουν στο ίδιο σημείο, το (φ_0, ρ_0). Συνεπώς ο μετασχηματισμός Radon μπορεί να μετασχηματίζει ευθείες σε σημεία όπως φαίνεται στην παρακάτω εικόνα.



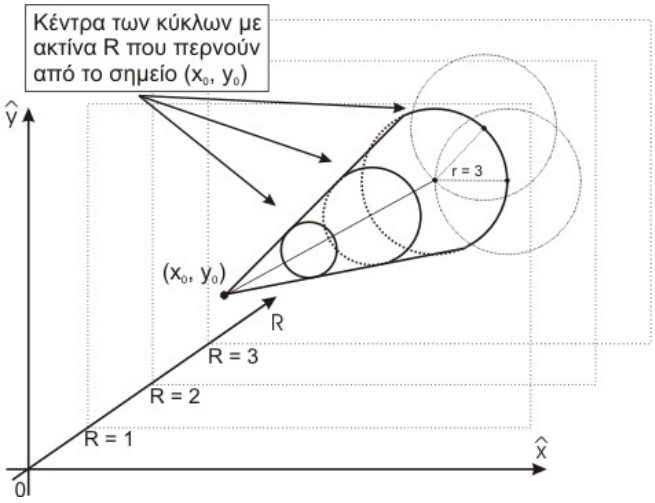
Εικόνα 6. Ανίχνευση γραμμών με τον μετασχηματισμό Radon (Hough)[4]

Κατά αντίστοιχο τρόπο στον κυκλικό μετ/μο Radon ένα σημείο μετασχηματίζεται σε ένα κώνο. Συγκεκριμένα αν ορίσουμε το διάνυσμα $\mathbf{x_0}=(x_0,y_0)$ τότε η συνάρτηση σημείου γίνεται: $f(x,y)=\delta(x-x_0)\delta(y-y_0)=\delta(\mathbf{x}-\mathbf{x_0})$ και ο κυκλικός της μετασχηματισμός Radon είναι:

$$R\{f\}=\breve{f}(R,\mathbf{x})=\int_S \delta(\mathbf{s}-\mathbf{x_0})\,\delta\left(R-|\mathbf{s}-\mathbf{x}|\right)d\mathbf{s}=\delta\left(R-|\mathbf{x_0}-\mathbf{x}|\right)$$

Δηλαδή μη μηδενικός μόνο στην επιφάνεια ενός κώνου που αποτελείται από κύκλους με κέντρο (x_0,y_0) και ακτίνα R.

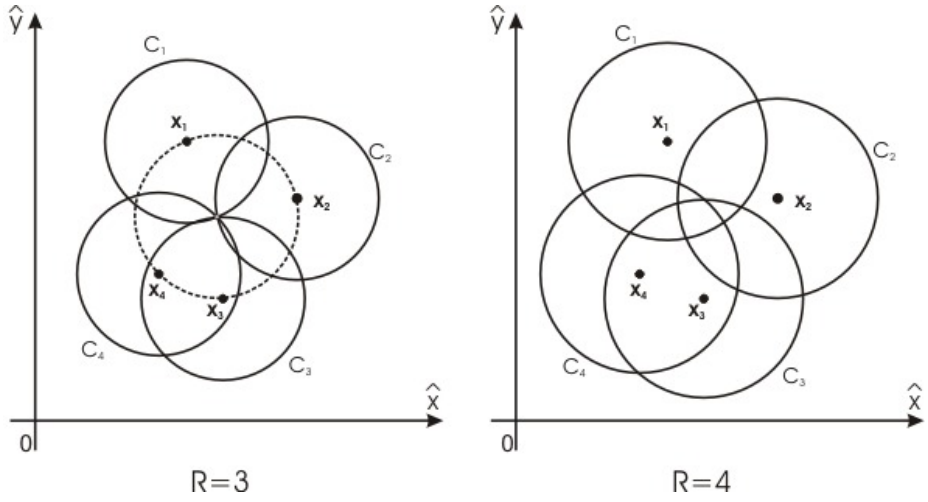
[3] P.V.C. Hough. A Method and Means for Recognizing Complex Patterns. US Patent: 3,069,654, Dec. 1962
[4] http://www2.lut.fi/~kyrki/ipcv02/hough/exercise/exercise.html



Εικόνα 7. Κυκλικός μετασχηματισμός σημείου

Στην παραπάνω εικόνα φαίνεται αυτός ο μετασχηματισμός. Παρατηρούμε ότι για R = 0 το σημείο (x_0, y_0) απεικονίζεται στην ίδια ακριβώς θέση του χώρου Radon πάνω στους νέους άξονες $\hat{x}\,\hat{y}$. Για μεγαλύτερα R το σημείο απεικονίζεται σε ένα κύκλο ακτίνας R. Ένας κύκλος με ακτίνα R αποτελεί τον γεωμετρικό τόπο των κέντρων των κύκλων με ακτίνα R που περνάνε από το κέντρο του όπως φαίνεται και στην παραπάνω εικόνα για R=3. Με αυτό τον τρόπο κάθε σημείο (x_0, y_0) στον χώρο x y «επιλέγει» στον χώρο Radon όλα τα σημεία που θα μπορούσαν να αποτελούν κέντρο κύκλου που περνά από το σημείο (x_0, y_0).

Λόγω της γραμμικότητας η υπέρθεση πολλών τέτοιων σημείων (pixels) από μία εικόνα δημιουργεί ένα άθροισμα τέτοιων κύκλων στον χώρο Radon.



Εικόνα 8. Αρχή λειτουργίας αναγνώρισης κύκλων

Για να απλουστεύσουμε λίγο την ανάλυσή μας θα εξετάσουμε δύο «διατομές» του χώρου Radon, για R = 3 και για R = 4. Έστω ότι έχουμε στον χώρο x-y τέσσερα σημεία $\mathbf{x_1}...\mathbf{x_4}$ που προέρχονται από ένα κύκλο ακτίνας R = 3. Όπως φαίνεται στην Εικόνα 8 αριστερά, το κάθε ένα τους θα δημιουργεί στον χώρο Radon για ακτίνα R = 3 ένα κύκλο ακτίνας 3. Βλέπουμε διάφορα σημεία στα οποία τέμνονται οι κύκλοι μεταξύ τους και στις περισσότερες περιπτώσεις τέμνονται ανά δύο. Σε μία μόνο περίπτωση τέμνονται και οι τέσσερις στο ίδιο σημείο. Λόγω της υπέρθεσης των τεσσάρων ομοίων κύκλων, στο σημείο αυτό η τιμή του μετασχηματισμού Radon θα είναι τέσσερις φορές μεγαλύτερη από την τιμή των αντίστοιχων κύκλων και δύο φορές μεγαλύτερη από κάθε σημείο τομής δύο κύκλων. Το σημείο αυτό αποτελεί το κέντρο κύκλου ο οποίος περνά και από τα τέσσερα σημεία. Για R = 4 το κάθε σημείο δημιουργεί ένα κύκλο ακτίνας 4 όπως φαίνεται στην

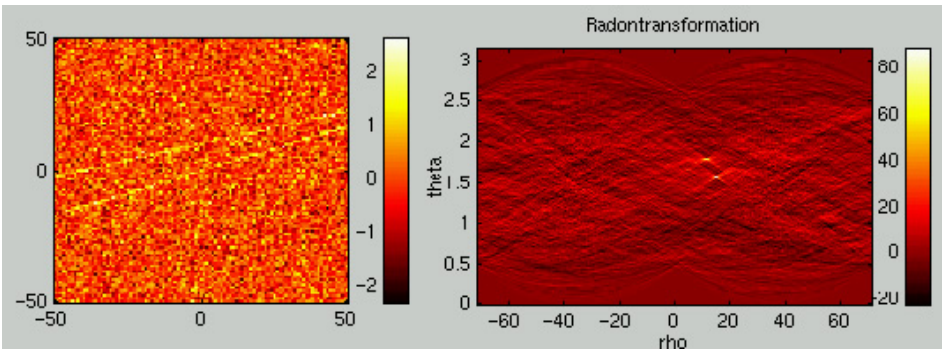
Εικόνα 8 δεξιά. Όπως μπορούμε να παρατηρήσουμε οι τέσσερις κύκλοι έχουν από-εστιαστεί και δεν τέμνονται πια όλοι σε κοινό σημείο. Βλέπουμε δηλαδή ότι η αναζήτηση μεγίστων στον χώρο Radon μπορεί να μας εντοπίσει με μεγάλη ακρίβεια τα στοιχεία του βέλτιστου κύκλου που περνά από σημεία.

Η διακριτή έκδοση του κυκλικού μετασχηματισμού Radon, ο κυκλικός μετασχηματισμός Hough κάνει ακριβώς αυτό το πράγμα. Σε μία εικόνα που έχει εφαρμοστεί φίλτρο ανίχνευσης αιχμών, ο μετασχηματισμός Hough για κάθε σημείο με αιχμή πάει και διαγράφει έναν κύκλο για κάθε R στον χώρο Hough αυξάνοντας κατά ένα τις τιμές της περιφέρειάς του. Αναζητώντας μετά μέγιστα στον χώρο Hough μπορούμε να ανιχνεύσουμε τους βέλτιστα διαγεγραμμένους κύκλους στην εικόνα.

Πρέπει να σημειώσουμε ότι όπως φαίνεται από την σχέση (7) ορισμού του κυκλικού μετ/μού Radon, η τιμή του (εκτός από τη συνάρτηση f) εξαρτάται ευθέως και από την ακτίνα R και συγκεκριμένα είναι ανάλογη. Αυτό είναι κάτι το αναμενόμενο αφού όπως αυξάνεται η ακτίνα R γίνεται ολοκλήρωση στον χώρο x-y πάνω σε αυξανόμενη περίμετρο κύκλου (2πR). Όταν λοιπόν θελήσουμε να βρούμε το μέγιστο στον χώρο Radon για να εντοπίσουμε τον βέλτιστο κύκλο, θα πρέπει να έχουμε υπόψη μας την παραπάνω παρατήρηση και να φροντίσουμε να κανονικοποιήσουμε σωστά τα αποτελέσματά μας πριν την σύγκριση.

1.4 Εφαρμογές του μετασχηματισμού Radon

Ο μετασχηματισμός Radon στην κλασική κυρίως εκδοχή βρίσκει πολλές εφαρμογές. Ο στατιστικός χαρακτήρας του τον κάνει αναντικατάστατο σε εφαρμογές στις οποίες υπάρχει πολύς θόρυβος. Για παράδειγμα στην παρακάτω εικόνα[5] αριστερά βλέπουμε δύο γραμμές ανάμεσα σε πάρα πολύ θόρυβο. Δεξιά φαίνεται ο μετασχηματισμός Radon της εικόνας και όπως μπορεί κανείς να δει οι δύο γραμμές ξεχωρίζουν ξεκάθαρα ως δύο σημεία με ένταση περίπου τέσσερις φορές μεγαλύτερη από αυτή του θορύβου.



Εικόνα 9. Ανθεκτικότητα μετ/μου Radon απέναντι στον θόρυβο

Οι κυριότεροι τομείς στους οποίους χρησιμοποιείται ο μετασχηματισμός Radon είναι οι εξής:

- Ιατρική
- Αστρονομία
- Οπτική
- Μοριακή βιολογία
- Γεωφυσική - Σεισμολογία
- Επιστήμη υλικών

Συγκεκριμένα αν θέλαμε να σκιαγραφήσουμε τις δύο δημοφιλέστερες κατηγορίες εφαρμογών, αυτές θα ήταν οι εξής:

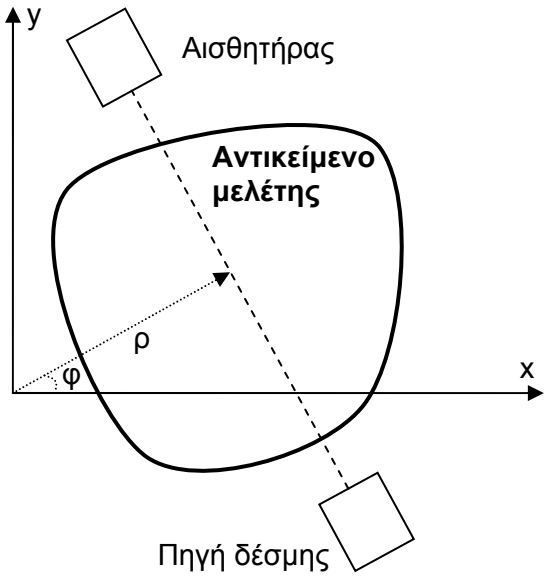
1. Εφαρμογές σε επεξεργασία και αναγνώριση εικόνας

[5] http://eivind.imm.dtu.dk/staff/ptoft/Radon/Radon.html

2. Εφαρμογές σε περιστροφικούς σαρωτές (scanners)

Η εφαρμογές που τον χρησιμοποιούν για αναγνώριση εικόνας ακολουθούν παρεμφερές σκεπτικό με αυτό που περιγράφηκε στην προηγούμενη ενότητα. Μία εικόνα φιλτράρεται, περνάει από κάποιο τελεστή ανίχνευσης αιχμών και στην συνέχεια μετασχηματίζεται στον χώρο Radon όπου συγκεκριμένα χαρακτηριστικά της μπορούν να διακριθούν πιο εύκολα.

Οι εφαρμογές σε περιστροφικούς σαρωτές βασίζονται στην εξής απλή ιδέα.



Εικόνα 10. Αρχή εφαρμογών μετ/μου Radon

Έστω ότι έχουμε ένα αντικείμενο του οποίου κάποια ιδιότητα θέλουμε να μελετήσουμε. Ο πιο απλός τρόπος να την μελετήσουμε είναι να το κόψουμε σε διατομές και να μετρήσουμε σε κάθε διατομή την ιδιότητα που μας ενδιαφέρει. Αν το αντικείμενο είναι πολύτιμο ή δεν έχουμε άμεση πρόσβαση σε αυτό τότε δεν μπορούμε να το καταστρέψουμε. Αυτό που μπορούμε να κάνουμε είναι να βρούμε μία δέσμη κάποιου τύπου π.χ. οπτική – ακτίνων Χ κ.α. η οποία να επηρεάζεται από την ιδιότητα που θέλουμε να μετρήσουμε. Αν σε μία διατομή η ιδιότητα έχει τιμή $f(x,y)$ τότε καθώς η δέσμη διαπερνά την διατομή παίρνουμε το ολοκλήρωμα της f πάνω στην ευθεία της δέσμης. Κάνοντας πολλές τέτοιες μετρήσεις σε διάφορες γωνίες φ και σε διάφορες αποστάσεις ρ παίρνουμε στην ουσία τον μετασχηματισμό Radon της συνάρτησης f. Στην συνέχεια εφαρμόζοντας κάποια από τις μεθόδους αναστροφής του μετασχηματισμού Radon[1,6], μπορούμε να έχουμε μία προσέγγιση της συνάρτησης f της ιδιότητας της οποίας θέλουμε να μετρήσουμε. Εκτελώντας πολλές διαδοχικές σειρές μετρήσεων της $\breve{f}(\rho,\phi)$ κατά τον άξονα z και εκτελώντας την αναστροφή για κάθε ύψος, μπορούμε να έχουμε και τρισδιάστατη άποψη για την ιδιότητα που μας ενδιαφέρει.

Στην διαδικασία της μέτρησης εμφανίζονται διάφορα προβλήματα όπως η διακριτική ικανότητα των αισθητήρων, η σκέδαση της ακτινοβολίας κ.α. τα οποία κάθε άλλο παρά τετριμμένη κάνουν την μεθοδολογία για την ανάπτυξη μίας τέτοιας εφαρμογής. Παρόλα αυτά το γεγονός ότι μπορούμε να μετρήσουμε μία ιδιότητα του εσωτερικού ενός αντικειμένου χωρίς να χρειαστεί να το καταστρέψουμε κάνει τον μετασχηματισμό Radon ανεκτίμητο.

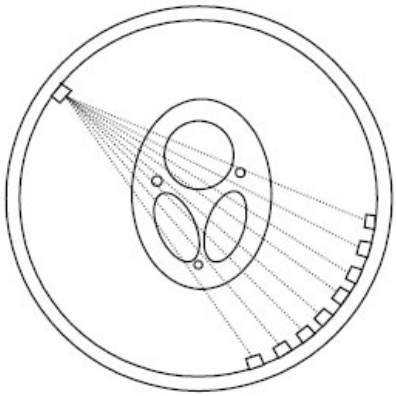
Παρακάτω θα εξετάσουμε κάποια χαρακτηριστικά παραδείγματα.

[6] The Radon Transform, Theory and Implementation, Peter Toft, Ph.D. Thesis, 1996

1.4.1 Τομογραφία

Η διαδεδομένη εφαρμογή του μετασχηματισμού Radon είναι στην τομογραφία. Χρησιμοποιώντας δέσμες ακτίνων Χ ή φωτονίων (SPECT) ή ποζιτρονίων (PET) μπορούμε να σαρώνουμε το υπό μελέτη αντικείμενο το οποίο μπορεί να είναι για παράδειγμα ένας ασθενής ή μία μηχανική κατασκευή και να ανοικοδομούμε μία εικόνα του εσωτερικού του χωρίς καμία μηχανική παρέμβαση.

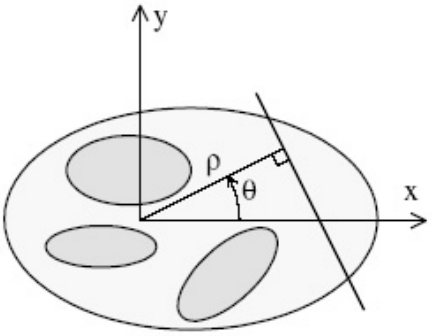
Η πιο άμεση χρήση του μετασχηματισμού Radon γίνεται στα CT-scanners (Computerized Tomography scanners) για την ανακάλυψη των οποίων επιβραβεύτηκαν το 1979 ο Cormack και ο Hounsfield με το βραβείο Nobel ιατρικής. Θα εξετάσουμε την λειτουργία του λίγο πιο αναλυτικά ως ένα παράδειγμα τομογράφου οι αρχές και για τα άλλα όργανα είναι παρόμοιες.



Εικόνα 11. Διάταξη CT scanner [6]

Το CT scanner μπορεί να αποτελείται από ένα δακτύλιο που από την μία πλευρά έχει μια πηγή ακτίνων Χ ενώ από την άλλη αρκετούς αισθητήρες για την ακτινοβολία αυτή όπως φαίνεται στο παραπάνω σχήμα. Θα υποθέσουμε ότι εξετάζουμε τον εγκέφαλο ενός ασθενή χωρίς αυτό να σημαίνει ότι οι εφαρμογές περιορίζονται μόνο εκεί. Στην πράξη κάθε αντικείμενο με μη-ομογενές εσωτερικό που αλληλεπιδρά με την δεδομένη ακτινοβολία θα μπορούσε να εξεταστεί με την ίδια διάταξη.

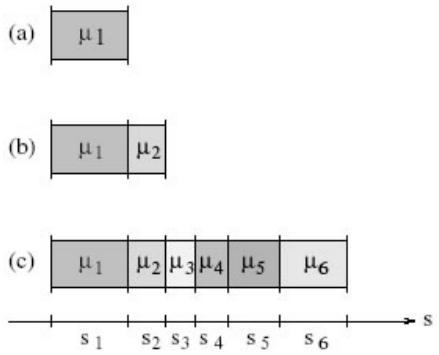
Καθώς το σύστημα περιστρέφεται γύρω από τον ασθενή γίνεται δυνατή η λήψη σήματος από όλα τα μέρη του εγκεφάλου. Υποθέτοντας ότι οι ακτίνες Χ ταξιδεύουν σε ευθείες γραμμές, η δέσμη εξασθενείτε κατά μήκος της ευθείας με ένα συντελεστή εξασθένισης μ. Μετά από την εξέταση, ο συντελεστής εξασθένισης μπορεί να προσδιοριστεί με ανοικοδόμηση για κάθε σημείο του επιπέδου. Η μέθοδος ανοικοδόμησης μπορεί να βασιστεί στην αναστροφή του μετασχηματισμού Radon.



Εικόνα 12. Ορισμός συστήματος συντεταγμένων σε τομογράφο

Υποθέτοντας ότι έχουμε ένα επίπεδο δισδιάστατο CT-scanner μπορούμε να εξετάσουμε μόνο μία διατομή του εγκεφάλου. Κάθε σημείο του εγκεφάλου μπορεί να προσδιοριστεί με δύο

συντεταγμένες. Την απόσταση ρ από την αρχή των αξόνων και την γωνία σε σχέση με τον κύριο άξονα θ, όπως φαίνεται στην Εικόνα 12. Υποθέτοντας ότι το εξεταζόμενο αντικείμενο δεν είναι ομογενές, ο συντελεστής εξασθένησης μπορεί να εκφραστεί ως μία συνάρτηση των x και y, π.χ. $\mu(x,y)$. Υποθέτουμε ότι η πηγή και ένας από τους αισθητήρες ορίζουν μία ευθεία (ρ,θ). Κατά τη διαδρομή μέσα από το υπό εξέταση αντικείμενο, διαφορετικές δέσμες θα συναντάν διαφορετικά «εμπόδια» και για διαφορετικά διαστήματα όπως μπορούμε να δούμε για παράδειγμα στην Εικόνα 13.



Εικόνα 13. Κάθε δέσμη συναντά διαφορετικά "εμπόδια"

Στην πρώτη περίπτωση (a) η λαμβανόμενη ένταση θα είναι σύμφωνα με τον νόμο της εκθετικής μείωσης:

$$I(\rho,\theta) = I_0 e^{-\mu_1 s_1}$$

Αν δύο διαφορετικά ομογενή μέσα διαπερνούνται όπως φαίνεται στην περίπτωση (b) η λαμβανόμενη ένταση είναι:

$$I(\rho,\theta) = I_0 e^{-\mu_1 s_1 - \mu_2 s_2}$$

Στην γενική περίπτωση (c) για κάθε νέο επίπεδο, ένας εκθετικός όρος προστίθεται:

$$I(\rho,\theta) = I_0 e^{-\sum_i \mu_i s_i}$$

Αν όπως ορίσαμε παραπάνω ο συντελεστής εξασθένησης κατά μήκος μίας ευθείας είναι μία συνάρτηση των x,y, τότε το παραπάνω άθροισμα μετατρέπεται σε ένα ολοκλήρωμα πάνω στα στοιχειώδη μήκη ds.

$$I(\rho,\theta) = I_0 e^{-\int \mu(x,y)ds}$$

Στην παραπάνω σχέση η παράμετρος I_0 συμβολίζει την ένταση της πηγής και το s συμβολίζει την παράμετρο της κανονικής μορφής μίας ευθείας, όπου το (x,y) βρίσκεται πάνω στην ευθεία που ορίζεται από τα (ρ,θ).

Αν ορίσουμε την προβολή πάνω στην ευθεία της παραπάνω σχέσης ως:

$$P(\rho,\theta) = \log\left(I_0 / I(\rho,\theta)\right)$$

οδηγούμαστε αβίαστα στον μετασχηματισμό Radon αφού η τιμή της είναι:

$$P(\rho,\theta) = \log\left(I_0 / I(\rho,\theta)\right) = \int \mu(x,y)\,ds = \int_{-\infty}^{\infty} \mu\left(\rho\cos\theta - s\sin\theta, \rho\sin\theta + s\cos\theta\right) ds =$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,y)\,\delta\left(\rho - x\cos\theta - y\sin\theta\right) dxdy$$

Συνεπώς η τιμή της εξασθένισης $\mu(x,y)$ μπορεί να υπολογιστεί από το δείγμα του μετασχηματισμού Radon που προκύπτει από τις προβολές με κάποια μέθοδο αναστροφής του μετασχηματισμού αυτού. Η απεικόνιση της εξασθένησης αυτής μπορεί να δώσει μία λεπτομερή εικόνα για την δομή του περιεχομένου της υπό εξέταση δομής. Τρισδιάστατες αναπαραστάσεις μπορούν να αναπαρασταθούν με την βοήθεια ενός CT-scanner που κινείται και κατά τον άξονα z.

1.4.2 Εφαρμογές επεξεργασίας εικόνας

Υπάρχουν πάρα πολλές εφαρμογές αναγνώρισης εικόνας που παρουσιάζουν ενδιαφέρον. Θα παρουσιάσουμε μία από αυτές η οποία παρουσιάζει ιδιαίτερο εκπαιδευτικό ενδιαφέρον ως προς την μεθοδολογία της.

Η πρώτη έρχεται από τον τομέα της ωκεανογραφίας και χρησιμοποιεί τον μετασχηματισμό Radon για την μέτρηση των ιδιοτήτων μίας ειδικής κατηγορίας θαλασσίων κυμάτων που ονομάζονται κύματα Rossby[7]. Τα κύματα αυτά δημιουργούνται σε ωκεανούς εξαιτίας της περιστροφής της γης και κινούνται με ταχύτητες μερικών εκατοστών ανά δευτερόλεπτο. Η μέτρηση της ταχύτητας και της κατεύθυνσής τους εξετάζεται στη συγκεκριμένη δημοσίευση με βάση δεδομένα από ειδικό radar που υπάρχει σε δορυφόρους και μετράει με ακρίβεια το ύψος (στην περίπτωσή αυτή μας ενδιαφέρει το ύψος από την επιφάνεια της θάλασσας).

Στο παρελθόν οι τεχνικές που είχαν εφαρμοστεί στις περισσότερες περιπτώσεις θεωρούσαν δεδομένη την κατεύθυνση της κίνησης αυτών των κυμάτων που είναι γνωστό ότι κατά κύριο λόγο γίνεται στην κατεύθυνση Ανατολής - Δύσης. Η υπόθεση αυτή αγνοεί όμως ένα ποσοστό του κύματος που διαδίδεται στην κατεύθυνση Βορά – Νότου. Με την βοήθεια του μετασχηματισμού Radon γίνεται δυνατή η μέτρηση και των δύο συνιστωσών. Η πρωτοτυπία του άρθρου έγκειται στο ότι χρησιμοποιεί τον μετασχηματισμό Radon σε τρεις διαστάσεις όπου οι διαστάσεις αυτές είναι οι x-y της εικόνας και ο χρόνος t με την βοήθεια του οποίου προσδιορίζεται η ταχύτητα των κυμάτων.

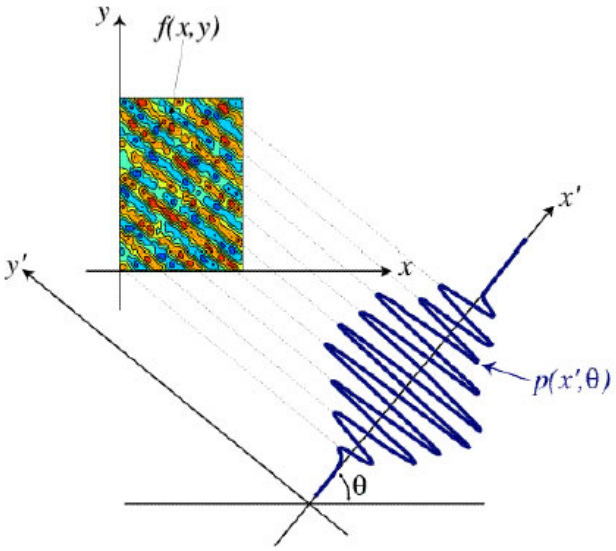
Η «παραδοσιακή» μέθοδος υπολογισμού γίνεται με την βοήθεια ειδικών διαγραμμάτων (Hovmöller) όπου στον ένα άξονα απεικονίζεται η θέση των μετώπων του κύματος κατά τη διεύθυνση Ανατολής – Δύσης και στον άλλο ο χρόνος. Η μέτρηση της ταχύτητας γίνεται μετρώντας την απόσταση των μετώπων σε συγκεκριμένο χρονικό διάστημα. Φυσικά ενώ αυτή η μέθοδος είναι εύκολο να εφαρμοστεί σε λίγα διαγράμματα, μία πιο αυτοματοποιημένη και ακριβής μέθοδος χρειάζεται για την μέτρηση σε όλη την περιοχή ενός ωκεανού.

Λόγω της περιοδικότητας των κυμάτων όπως εμφανίζονται στα διαγράμματα μία κοινή πρακτική για την αυτοματοποίηση της μέτρησης της ταχύτητας των κυμάτων είναι η χρήση του μετασχηματισμού Fourier[8]. Συγκεκριμένα σε μερικές συχνότητες εμφανίζονται μέγιστα τα οποία αντιστοιχούν στην ταχύτητα που ζητάμε.

Μία πιο άμεση μέθοδος πηγάζει κατευθείαν από τον τρόπο που εμπειρικά μετρά κανείς την ταχύτητα από το διάγραμμα. Η μέθοδος αυτή συμπίπτει με τον μετασχηματισμό Radon όπως μπορεί να δει κανείς στο παρακάτω σχήμα:

[7] Use of the 3D Radon transform to examine the properties of oceanic Rossby waves, Peter G Challenor, Paolo Cipollini and David Cromwell, Journal of Atmospheric and Oceanic Technology, November 2000

[8] Σελίδα 53, A Study of the Indian Ocean Circulation using Satellite Observations and Model Simulations, Dr. Bulusu Subrahmanyam, Ph.D Thesis, 2004



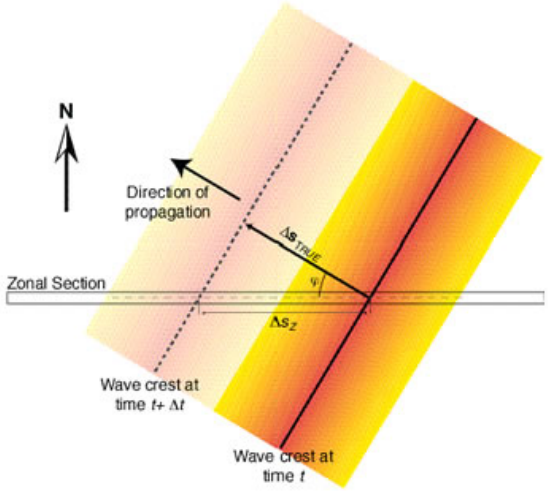
Εικόνα 14. Ο μετασχηματισμός Radon στην μέτρηση της ταχύτητας κυμάτων

Στο σχήμα αυτό φαίνεται ένα (εξιδανικευμένο) διάγραμμα Hovmöller και ο μετασχηματισμός του κατά Radon για μία επιλεγμένη γωνία. Για τον προσδιορισμό της γωνίας αυτής κάνουμε τον μετασχηματισμό για θ από 0^o έως 180^o και παρατηρούμε το σήμα κατά ρ. Όταν η ευθεία προβολής του μετασχηματισμού ευθυγραμμιστεί με τις ευθείες κορυφών – κοιλάδων του κύματος, σε αυτή τη γωνία η ενέργεια (μεταβολή) του σήματος κατά ρ γίνεται μέγιστη. Συγκεκριμένα ζητάμε την γωνία θ_{max} που μεγιστοποιεί την παρακάτω έκφραση της ενέργειας:

$$P(\theta) = \int R^2(\rho,\theta)\, d\rho$$

όπου R ο μετασχηματισμός Radon του διαγράμματος. Με αυτό τον τρόπο μπορούμε να προσδιορίσουμε την κατεύθυνση διάδοσης των κυμάτων στο διάγραμμα χώρου – χρόνου η οποία φυσικά είναι η ταχύτητά τους. Με αυτή τη μέθοδο απλοποιείται σημαντικά ο υπολογισμός της ταχύτητας των κυμάτων.

Η πραγματική όμως ισχύς του μετασχηματισμού Radon φαίνεται στην τρισδιάστατη περίπτωση.



Εικόνα 15. Τρισδιάστατη έκδοση μέτρησης της ταχύτητας

Στην παραπάνω εικόνα βλέπουμε την πραγματική δισδιάστατη εικόνα του κύματος. Αυτό διαδίδεται όπως συμβαίνει στην πραγματικότητα κατά μία κατεύθυνση η οποία παρουσιάζει γωνία φ σε σχέση με την διεύθυνση Ανατολής – Δύσης που μετράμε συνήθως με τα διαγράμματα Hovmöller. Μετά από χρόνο Δt το κύμα έχει στην πραγματικότητα διανύσει διάστημα Δs_{TRUE} και

συνεπώς η πραγματική του ταχύτητα είναι V_{TRUE} = Δs_{TRUE} / Δt. Αυτό που μετράγαμε με την προηγούμενη μεθοδολογία ήταν το Δs_z όπως εμφανίζεται στην Εικόνα 15 η οποία είναι $\Delta s_Z = |\Delta s_{TRUE}| / \cos\phi \geq |\Delta s_{TRUE}|$ και συνεπώς $V_Z = \Delta s_z / \Delta t = V_{TRUE} / \cos\phi \geq |V_{TRUE}|$.

Για τον χειρισμό της τρισδιάστατης περίπτωσης ορίζεται ο μετασχηματισμός Radon σύμφωνα με την σχέση:

$$R(x', y', \theta, \phi) = \int_{z'} f(x, y, z)\, dz'$$

Ο οποίος είναι όμοιος με τον μετασχηματισμό όπως τον έχουμε εξετάσει μέχρι τώρα με την διαφορά ότι η ολοκλήρωση γίνεται τώρα πάνω σε ένα επίπεδο x'y' το οποίο είναι περιστραμένο κατά θ και φ σε σχέση με το επίπεδο xy. Πιο συγκεκριμένα το νέο σύστημα αξόνων x'y'z' ορίζεται με βάση το xyz σύμφωνα με τις σχέσεις:

$$x' = x\cos\phi\cos\theta + y\sin\phi\cos\theta + z\sin\theta$$
$$y' = -x\sin\phi + y\cos\phi$$
$$z' = -x\cos\phi\sin\theta - y\sin\phi\sin\theta + z\cos\theta$$

Επαναλαμβάνοντας διαδικασία ίδια με την δισδιάστατη περίπτωση προσδιορίζουμε τα θ_{max} και φ_{max} που μεγιστοποιούν την ενέργεια. Το θ_{max} μας δίνει την ταχύτητα όπως στην δισδιάστατη περίπτωση ενώ το φ_{max} μας δίνει την διεύθυνση της διάδοσης του κύματος.

Αυτή η εφαρμογή αναδεικνύει την πραγματική δύναμη του μετασχηματισμού Radon και την γενικότητα των εφαρμογών του. Ο μετασχηματισμός Radon αποδεικνύεται πάρα πολύ χρήσιμος όχι μόνο σε εφαρμογές επεξεργασίας εικόνας με τον «κλασικό ορισμό» της έννοιας εικόνα αλλά με τον γενικότερο ορισμό που περιλαμβάνει οτιδήποτε στους δύο άξονες του «πλάνου» από χρόνο μέχρι απόκλιση κύματος από σημείο αναφοράς.

Επιπλέον ο μετασχηματισμός Radon στις τρεις διαστάσεις εμφανίζεται ως ένα πάρα πολύ ισχυρό εργαλείο λύσης αναγνώρισης εικόνας υπό την γενικευμένη έννοια, αναγνωρίζοντας τα επίπεδα που μας ενδιαφέρουν μέσα στον χώρο.

Μία ακόμα ενδιαφέρουσα παρατήρηση είναι ότι στην περίπτωση αυτή δεν μας ενδιέφερε άμεσα η μεταβλητή ρ ούτε και ψάχναμε μέγιστα σε αυτόν καθ' αυτόν τον μετασχηματισμό Radon. Αυτό που ψάχναμε ήταν τα μέγιστα στην ενέργεια του σήματος σε κάποια διατομή. Πραγματικά, διάφορα κριτήρια μπορούν να εφαρμοστούν στον χώρο Radon δίνοντάς μας την δυνατότητα να μετράμε και να «ανακαλύπτουμε» διαφορετικά χαρακτηριστικά του σήματος που μας ενδιαφέρει.

Εφαρμογές αναγνώρισης εικόνας μπορεί να βρει κανείς σε διάφορου τομείς όπως π.χ. στην αστρονομία όπου βάσεις δεδομένων με εικόνες του έναστρου ουρανού πρέπει να καθαρίζονται με αυτοματοποιημένο τρόπο από θόρυβο όπως πέρασμα αεροπλάνων ή θόρυβο που οφείλεται στην συσκευή λήψης[9]. Για την πραγματοποίηση αυτού του σύνθετου στόχου ο ελλειπτικός μετασχηματισμός Hough συνδυάζεται με πολλές άλλες τεχνικές όπως η τεχνική εξόρυξης δεδομένων «Renewal String» που αναπτύχθηκε ειδικά γι' αυτή την εφαρμογή.

[9] Cleaning Sky Survey Databases using Hough Transform and Renewal String Approaches, A. J. Storkey, N. C. Hambly, C. K. I. Williams, R. G. Mann, September 2003

Εικόνα 16. Ανίχνευση ενός CD-ROM κάτω από μία εφημερίδα με υβριδική τεχνική

Επιπλέον υβριδικές τεχνικές μπορούν να επιτύχουν εντυπωσιακά αποτελέσματα. Για παράδειγμα [10]μπορεί να χρησιμοποιηθεί ο μετασχηματισμός Hough για την γρήγορη εύρεση ενός βέλτιστου κύκλου σε μία εικόνα και στην συνέχεια με επέκταση αυτού του κύκλου να προσεγγιστεί μία έλλειψη με πολύ καλή ακρίβεια, με λιγότερο υπολογιστικό κόστος απ' ότι θα χρειαζόταν για να ανιχνευτεί η ίδια έλλειψη με ελλειπτικό Hough μετασχηματισμό.

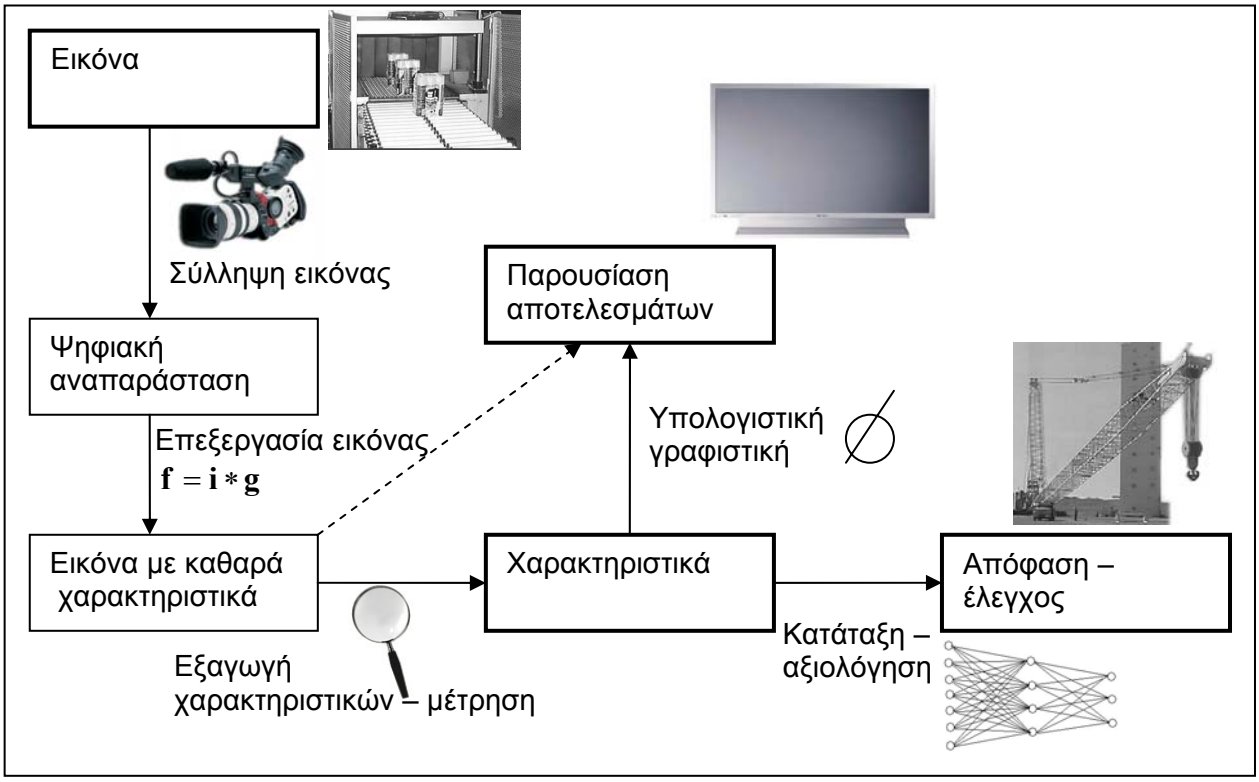
[10] Automatic Detection of Circular Objects by Ellipse Growing, Kenichi KANATANI, Naoya OHTA, November 2001

Κεφάλαιο 2. Επεξεργασία και αναγνώριση εικόνας

Στο κεφάλαιο αυτό παρουσιάζονται τεχνικές της επεξεργασίας και αναγνώρισης εικόνας. Το αντικείμενο αυτό είναι πολύ εκτενές και φυσικά δεν μπορεί να καλυφθεί μέσα σε ένα μόνο κεφάλαιο. Ο στόχος μας είναι να παρουσιάσουμε στον μη ειδικό αναγνώστη την γενική εικόνα για το τι είναι ένα σύστημα επεξεργασίας και αναγνώρισης εικόνας και επιγραμματικά τις σημαντικότερες τεχνικές που χρησιμοποιούνται για κάθε τμήμα της υλοποίησής του. Οι αναφορές μπορούν να αποτελέσουν πολύτιμο υλικό για όποιον θέλει να ασχοληθεί για να ενημερωθεί περισσότερο για κάποιες τεχνικές που πιθανώς τον ενδιαφέρουν. Αναλύονται εκτενώς οι τεχνικές που χρησιμοποιούνται στην δικιά μας εφαρμογή. Ειδικά στην ενότητα 2.5.1.1 γίνεται ανάλυση του συστήματος αναγνώρισης προτύπου που χρησιμοποιούμε.

2.1 Συστήματα επεξεργασίας και αναγνώρισης εικόνας

Η επεξεργασία και αναγνώριση εικόνας είναι μία σύνθετη διαδικασία αποτελούμενη από πολλά βήματα. Με την βοήθειά της γίνεται δυνατή η ανάκτηση και παρουσίαση χρήσιμης πληροφορίας από μία εικόνα. Απώτερος σκοπός είναι η εκμετάλλευση της εξαγόμενης πληροφορίας για αυτοματοποίηση διαδικασιών που με άλλο τρόπο θα ήταν από δύσκολο έως αδύνατο να αυτοματοποιηθούν και φυσικά η μείωση του συνολικού κόστους ενός συστήματος.



Εικόνα 17. Ο Χάρτης της επεξεργασίας και αναγνώρισης εικόνας

Όπως μπορούμε να δούμε στο παραπάνω σχήμα είναι αρκετά τα επί μέρους τμήματα τα οποία πρέπει να υλοποιηθούν σε ένα σύστημα αναγνώρισης εικόνας.

Το πρώτο βήμα που πρέπει να γίνει είναι η σύλληψη της εικόνας. Αυτό συνήθως θα γίνει με κάποια camera αλλά ανάλογα με την εφαρμογή οι εξειδικευμένες απαιτήσεις μπορεί να επιβάλουν από μία απλή φωτογραφική μηχανή ως ένα τηλεσκόπιο ή ηλεκτρονικό μικροσκόπιο.

Όταν πλέον έχουμε την εικόνα σε ψηφιακή μορφή αυτή συνήθως δεν είναι σε μορφή που να μπορεί να μας οδηγήσει σε ενδιαφέροντα συμπεράσματα. Με επεξεργασία της εικόνας μπορούμε να παράγουμε εικόνα καθαρότερη, με έντονα τα χαρακτηριστικά που μας ενδιαφέρουν. Επιπλέον

μπορούμε κατά την επεξεργασία να αναιρέσουμε διάφορες παραμορφώσεις που δέχτηκε η εικόνα κατά την διαδικασία της σύλληψης είτε λόγω ατέλειας της συσκευής σύλληψης π.χ. παραμόρφωση φακού, είτε για οποιοδήποτε άλλο λόγο π.χ. όταν το αντικείμενο κινείται με ταχύτητα εμφανίζεται θολό.

Μετά από τη φάση της επεξεργασίας η εικόνα έχει καθαρά τα χαρακτηριστικά που αναζητούμε. Ακόμα και μέχρι σε αυτό το σημείο έχουμε κάνει κάτι πάρα πολύ σημαντικό. Για παράδειγμα η επανάσταση στην τεχνολογία των ιατρικών οργάνων οφείλεται σε πολύ μεγάλο βαθμό στην «καθαρή εικόνα» που προσφέρουν τα σύγχρονα όργανα στους γιατρούς. Η ψηφιακή τεχνολογία τους δίνει την δυνατότητα να καθαρίζουν, να αποθηκεύουν και να συγκρίνουν εικόνες καθώς επίσης και να κάνουν συγκεκριμένες μετρήσεις πάνω τους.

Από την καθαρή εικόνα μπορούμε να κάνουμε εξαγωγή των χαρακτηριστικών που θέλουμε και μέτρηση αυτών. Για παράδειγμα σε μία εικόνα από ένα εργοστάσιο που παράγει σωλήνες, τα χαρακτηριστικά μπορούν να είναι η διάμετρος των σωλήνων, το χρώμα τους (για την ανίχνευση ρωγμών) ή οτιδήποτε άλλο. Οι μετρήσεις για τα χαρακτηριστικά που μας ενδιαφέρουν αποτελούν τη χρήσιμη πληροφορία που μας παρέχει το σύστημα αναγνώρισης εικόνας.

Αυτή την πληροφορία μπορούμε να την παρουσιάσουμε σε κάποιο χειριστή. Εδώ μας βοηθάει πάρα πολύ η τεχνολογία της υπολογιστικής γραφικής (computer graphics) που μας δίνει την δυνατότητα να παρουσιάσουμε τα δεδομένα στον χειριστή με ρεαλιστικό και χρήσιμο τρόπο. Για παράδειγμα στο σύστημα που μετράει διαμέτρους σωλήνων είναι δυνατόν οι ρωγμές να επισημαίνονται με κάποιο ειδικό διακριτικό π.χ. κόκκινο κύκλο, ώστε να διευκολύνεται ο χειριστής στον εντοπισμό τους. Γενικά με την βοήθεια των computer graphics είναι δυνατόν να υπερτίθενται πάνω στην εικόνα με γραφικό τρόπο πληροφορίες για την θερμοκρασία, την εντατική κατάσταση και άλλες ιδιότητες ενός αντικειμένου.

Σε αρκετές περιπτώσεις μπορεί το σύστημα να ολοκληρωθεί ακόμα περισσότερο κάνοντας περιττή την ύπαρξη χειριστή. Πραγματικά από τα χαρακτηριστικά είναι δυνατόν να γίνει κατάταξη της παρατηρούμενης εικόνας και με βάση αυτά να ληφθούν αποφάσεις. Για παράδειγμα μία βιομηχανία παραγωγής ψωμιού μπορεί να εκμεταλλευτεί την επεξεργασία εικόνας για την αυτόματη απόρριψη καρβελιών που δεν έχουν ψηθεί καλά (με βάση το χρώμα) ή δεν έχουν το σωστό σχήμα ή δεν έχουν ομοιόμορφη κατανομή στο σουσάμι[11].

Παρακάτω θα εξετάσουμε αναλυτικά τις επί μέρους διεργασίες της επεξεργασίας εικόνας παραθέτοντας τεχνικές που χρησιμοποιούνται γενικά και δίνοντας φυσικά ιδιαίτερη έμφαση σε αυτές που χρησιμοποιούμε εμείς στο δικό μας σύστημα.

2.2 Σύλληψη εικόνας

Η πρώτη διαδικασία σε ένα σύστημα αναγνώρισης εικόνας είναι η σύλληψη της εικόνας. Αυτή θα γίνεται με την βοήθεια κάποιας κάμερας ή κάποιου άλλου τύπου συστοιχίας φωτοαισθητήρων. Όσο καλύτερο είναι το ψηφιακό σήμα που λαμβάνουμε από την κάμερα, τόσο πιο εύκολη γίνεται η επεξεργασία αργότερα. Αντίθετα αν η ψηφιακή εικόνα είναι θολή (επειδή π.χ. η κάμερα δεν είναι σωστά εστιασμένη) ή έχει μικρό δυναμικό εύρος (υπερφωτισμένη ή υποφωτισμένη), καμία τεχνική δεν μπορεί να την επαναφέρει και συνεπώς το σύστημα ψηφιακής επεξεργασίας και αναγνώρισης εικόνας δεν πρόκειται να δουλέψει.

2.2.1 Η αντίληψη μίας εικόνας

Τις περισσότερες φορές ένα σύστημα αναγνώρισης εικόνας θέλει να αναπαραστήσει μία διαδικασία την οποία μπορεί να κάνει ο άνθρωπος. Είναι σημαντικό να γνωρίζουμε κάποια πράγματα για το πως αντιλαμβάνεται ο άνθρωπος μία εικόνα για να μπορέσουμε να μοντελοποιήσουμε ορθά την διαδικασία αναγνώρισης. Ο άνθρωπος αντιλαμβάνεται μία εικόνα με

[11] gtresearchnews.gatech.edu/newsrelease/bakery.htm (bakery.pdf)

βάση την σύνθετη διαδικασία της όρασης στην οποία τα μάτια και το νευρικό σύστημα παίζουν τον κύριο ρόλο. Είναι σημαντικό να γνωρίζουμε ότι:

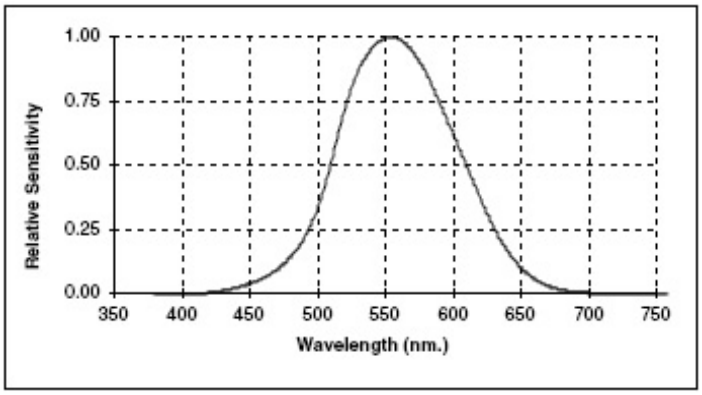
1. Δεν ξέρουμε τα πάντα για το ανθρώπινο οπτικό σύστημα.
2. Δεν υπάρχει «τυπικός» ανθρώπινος παρατηρητής. Ο κάθε ένας μας βλέπει ένα αρκετά διαφορετικό κόσμο και από οπτικής απόψεως.

Παρόλα αυτά οι έρευνες πάνω στην ψυχολογία της αντίληψης έχουν δώσει κάποιες απαντήσεις σε ορισμένα ερωτήματα για την ανθρώπινη αντίληψη και όραση. Η ενότητα που ακολουθεί έχει ως κύρια πηγή του το Digital Signal Processing Handbook[14]. Αναλυτικές πληροφορίες μπορεί κανείς να βρει και στο βιβλίο Digital Image Processing[12].

2.2.1.1 Ευαισθησία ως προς την φωτεινότητα

Ας υποθέσουμε ότι μία περιοχή μίας εικόνας έχει μία σταθερή ένταση φωτός σε ένα χρώμα με μήκος κύματος λ, δηλαδή ότι $I(\lambda) = I_0$.

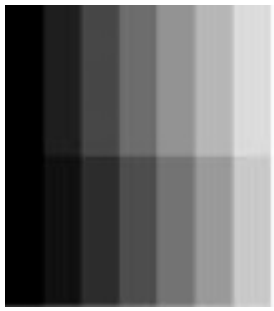
Η ευαισθησία με βάση το χρώμα (μήκος κύματος) φαίνεται στην Εικόνα 18.



Εικόνα 18. Η ευαισθησία της ανθρώπινης αντίληψης ως προς το μήκος κύματος

Όπως βλέπουμε κάθε άλλο παρά γραμμική είναι. Η μέγιστη ευαισθησία του είναι στην περιοχή που ονομάζουμε ορατό φάσμα φωτός.

Η ευαισθησία ως συνάρτηση της έντασης του φωτός I_0 είναι και αυτή κάτι το μη αντικειμενικό. Μία καλή προσέγγιση είναι ο νόμος του Weber-Fechner που λέει ότι το ερέθισμα που λαμβάνουμε είναι ανάλογο του λογαρίθμου της έντασης $R = \log(I_0)$. Αυτό φαίνεται στην παρακάτω εικόνα.



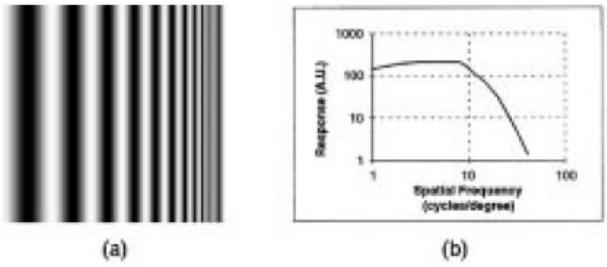
Εικόνα 19. Πάνω δI = k, κάτω δI = k x I

Παρατηρούμε επίσης το Mach band effect. Παρόλο που το χρώμα είναι σταθερό κατά μήκος κάθε λωρίδας, κοντά στις αιχμές νομίζουμε ότι βλέπουμε κάποιες διακυμάνσεις, (το σκοτεινότερο να

[12] Digital Image Processing, William K. Pratt, John Wiley & Sons, 2001

φαίνεται πιο σκοτεινό και το φωτεινότερο πιο φωτεινό), οι οποίες δίνουν και μία αίσθηση καμπυλότητας. Αυτό είναι μία μόνο από τις πολλές ψευδαισθήσεις[12] της οπτικής μας αντίληψης.

2.2.1.2 Ευαισθησία ως προς την χωρική συχνότητα

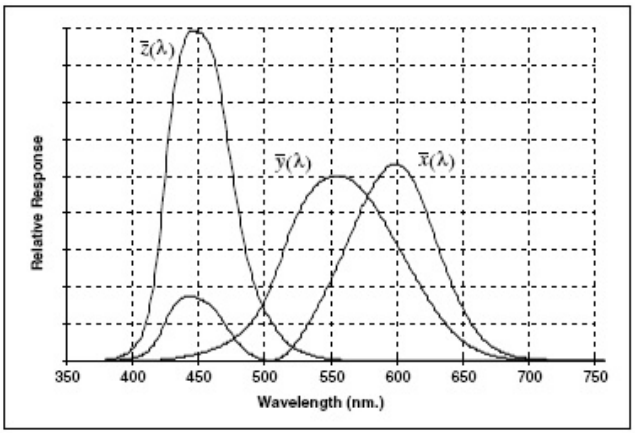


Εικόνα 20. Απόκριση χωρικής συχνότητας

Αν εφαρμόσουμε μία ένταση της μορφής $I(x) = I_0 \cos(a \cdot x)$, μπορούμε να μετρήσουμε την αντίληψη ως προς την χωρική συχνότητα. Όπως βλέπουμε στην παραπάνω εικόνα, η ένταση κάθε άλλο παρά γραμμική είναι. Συγκεκριμένα μία τυπική καμπύλη απόκρισης φαίνεται στο (b) μέρος της παραπάνω εικόνας.

2.2.1.3 Χρωματική ευαισθησία

Ο άνθρωπος βασίζει την αντίληψη του χρώματος με βάση το ερέθισμα που του δίνουν τρία διαφορετικά είδη αισθητηρίων κυττάρων τα οποία βρίσκονται πάνω στον αμφιβληστροειδή.



Εικόνα 21. Χρωματική απόκριση για τα τρία είδη κυττάρων

Η απόκριση των τριών αυτών ειδών κυττάρων φαίνεται στην Εικόνα 21. Το τελικό «σήμα» που λαμβάνει ο εγκέφαλος από κάθε είδος κυττάρων μπορεί να υπολογιστεί ως εξής:

$$X = \int_0^\infty I(\lambda)\bar{x}(\lambda)d\lambda,\ Y = \int_0^\infty I(\lambda)\bar{y}(\lambda)d\lambda,\ Z = \int_0^\infty I(\lambda)\bar{z}(\lambda)d\lambda$$

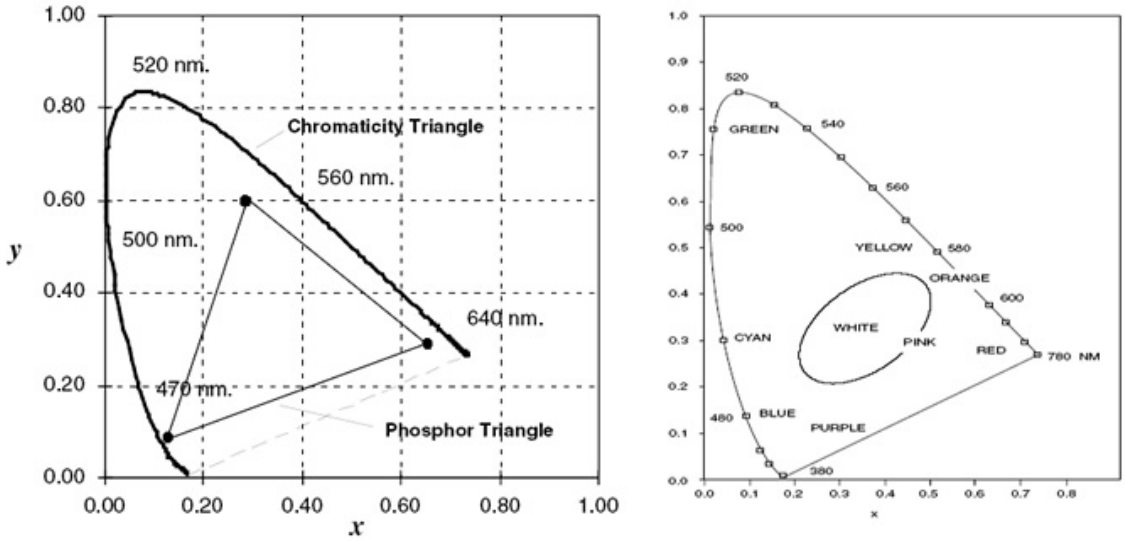
Αν θέλουμε να απαλλαγούμε από την πληροφορία της έντασης μπορούμε να ορίσουμε τις εξής χρωματικές συντεταγμένες:

$$x = \frac{X}{X+Y+Z},\ y = \frac{Y}{X+Y+Z},\ z = \frac{Z}{X+Y+Z}$$

Προφανώς κάθε χρωματική συντεταγμένη μπορεί να παίρνει τιμές από 0 έως 1 και φυσικά μόνο δύο από αυτές είναι ανεξάρτητες. Η τρίτη μπορεί να υπολογιστεί με βάση τις άλλες δύο από τον τύπο π.χ. $z = 1 - (x + y)$. Η x ονομάζεται κόκκινη χρωματική συντεταγμένη και η y πράσινη.

Όλες οι χρωματικές κατανομές που φαίνονται στον παρατηρητή ως το ίδιο χρώμα έχουν τις ίδιες χρωματικές συντεταγμένες. Παρατηρείστε ότι πολλά διαφορετικά χρώματα τα αντιλαμβανόμαστε ως ένα λόγω των μη γραμμικών αποκρίσεων που φαίνονται στην Εικόνα 21.

Αν είχαμε την δυνατότητα να παράγουμε τελείως μονοχρωματική δέσμη σε όλο το εύρος του τύπου $I(\lambda) = \delta(\lambda - \lambda_0)$ και παρατηρούσαμε τις τιμές των χρωματικών συντεταγμένων θα παρατηρούσαμε το λεγόμενο CIE χρωματικό τρίγωνο που φαίνεται στην παρακάτω εικόνα.



Εικόνα 22. Χρωματικό διάγραμμα με το CIE τρίγωνο και το τρίγωνο του φωσφόρου

Όλα τα χρώματα που μπορούμε να δούμε βρίσκονται μέσα και πάνω στο τρίγωνο CIE. Τα χρώματα που βρίσκονται πάνω στο τρίγωνο είναι τα καθαρά χρώματα ενώ όσα βρίσκονται μέσα γίνονται πιο παστέλ, όσο πιο πολύ προχωράμε προς το κέντρο.

2.2.1.4 Χρωματικοί χώροι (Color spaces)

Χρησιμοποιώντας φώσφορο με τρία διαφορετικά χρώματα, μπλε, κόκκινο και πράσινο οι οθόνες μπορούν να συνθέσουν όλα τα χρώματα που βρίσκονται μέσα στο χρωματικό τρίγωνο του φωσφόρου. Προφανώς από τις οθόνες δεν μπορούμε να δούμε όλα τα χρώματα. Συγκεκριμένα δεν μπορούμε να δούμε τα χρώματα που βρίσκονται εκτός του τριγώνου του φωσφόρου και εντός του CIE τριγώνου. Το ουσιαστικό ερέθισμα που θα αντιληφθεί το μάτι από τις εντάσεις του φωσφόρου δίνεται από τον παρακάτω πίνακα:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.61 & 0.17 & 0.20 \\ 0.30 & 0.59 & 0.11 \\ 0.00 & 0.07 & 1.11 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Ένα άλλο σύστημα που συναντάται πάρα πολύ συχνά στα πρωτόκολλα video PAL, NTSC και SECAM είναι το σύστημα YUV[13]. Το σήμα RGB μπορεί να ανασυντεθεί από το σήμα YUV με βάση τον παρακάτω πίνακα.

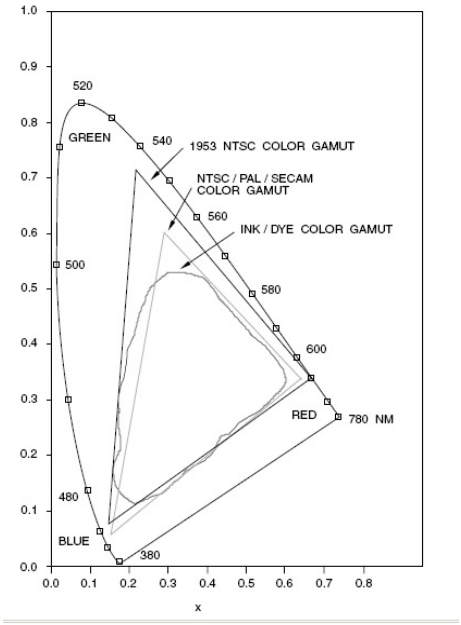
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.14 \\ 1 & -0.40 & -0.58 \\ 1 & 2.03 & 0 \end{bmatrix} \cdot \begin{bmatrix} Y \\ U \\ V \end{bmatrix}$$

[13] Video Demystified, Keith Jack, LLH Technology Publishing, 2001

Το, όχι και τόσο προφανές, προτέρημα αυτού του συστήματος είναι ότι ο συντελεστής Y περιέχει επαρκή πληροφορία φωτεινότητας του σήματος και τα U και V ουσιαστικά περιέχουν την πληροφορία του χρώματος. Με αυτόν τον τρόπο οι παλαιότεροι ασπρόμαυροι δέκτες μπορούσαν να δείχνουν κανονικά το τηλεοπτικό πρόγραμμα χρησιμοποιώντας την Y πληροφορία η οποία μεταδιδόταν με τον ίδιο ακριβώς τρόπο όπως στις ασπρόμαυρες εκπομπές. Η χρωματική πληροφορία κωδικοποιημένη με ειδικό τρόπο έδινε χρώμα στους έγχρωμους δέκτες.

Ο χρωματικός χώρος που χρησιμοποιείται για την μετάδοση ψηφιακού video είναι ο YcbCr και είναι μία τροποποίηση του YUV. Η αφαίρεση του -128 είναι ένα «κόλπο» για να δημιουργεί προσημασμένους αριθμούς από απρόσημους.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 1.37 & 0 \\ 1 & -0.70 & -0.34 \\ 1 & 0 & 1.73 \end{bmatrix} \cdot \begin{bmatrix} Y_{601} \\ Cr-128 \\ Cb-128 \end{bmatrix}$$



Εικόνα 23. Χρωματικά τρίγωνα για διάφορους τύπους κωδικοποίησης

Στην παραπάνω εικόνα βλέπουμε τα χρωματικά τρίγωνα τα οποία αντικατοπτρίζουν τις χρωματικές δυνατότητες διαφόρων τεχνολογιών.

2.2.2 Η κάμερα

Η πρώτη επεξεργασία την οποία παθαίνει η εικόνα σε ένα σύστημα επεξεργασίας και αναγνώρισης εικόνας είναι η σύλληψή της από κάποια κάμερα. Συγκεκριμένα αν θεωρήσουμε την πραγματική εικόνα την οποία βλέπουμε ως μία συνεχή συνάρτηση έντασης (ή και χρώματος όπως περιγράψαμε παραπάνω) $f(x,y) \to \Re$, μετά την σύλληψη από τα εικονοστοιχεία το σήμα που έχουμε είναι:

$$f_s(x,y) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} f(jx,ky)\,\delta(x - j\Delta x, y - k\Delta y)$$

όπου Δx και Δy η απόσταση των pixels κατά τον οριζόντιο και κατακόρυφο άξονα. Από τον μετασχηματισμό κιόλας αυτό προκύπτει το aliasing effect που εμφανίζεται στις περιοχές της εικόνας με υψηλές χωρικές συχνότητες. Για την αποφυγή του, ειδικά anti-aliasing φίλτρα πρέπει να εφαρμοστούν. Στην συνέχεια το A/D κάνει το παραπάνω σήμα ψηφιακό δηλαδή το ανάγει σε

πεπερασμένες διακριτές τιμές από το αρχικό του συνεχές εύρος. Σε αυτή την διαδικασία εισάγεται ο λεγόμενος θόρυβος κβαντισμού.

2.2.2.1 Θόρυβος

Πολλοί τύποι οπτικού θορύβου εισάγονται κατά την σύλληψη μίας εικόνας. Οι σημαντικότεροι φαίνονται παρακάτω και εκτός από τον πρώτο όλοι οι υπόλοιποι καθορίζονται από τις δυνατότητες της κάμερας[14]:

1.	Θόρυβος φωτονίων	Ο θόρυβος αυτός εισάγεται από την στατιστική φύση που κυβερνά την εκπομπή των φωτονίων. Δεν μπορούμε να υποθέσουμε ότι σε δύο συνεχόμενες παρατηρήσεις για ίδιο χρονικό διάστημα, στην ίδια ένταση, θα μετρήσουμε τον ίδιο αριθμό φωτονίων. Το μόνο που μπορούμε να εγγυηθούμε είναι μία μέση τιμή σε μεγάλο αριθμό παρατηρήσεων. Προφανώς αυτόν τον θόρυβο ο οποίος είναι μετρήσιμος δεν μπορούμε να τον αποφύγουμε, όσο καλή κάμερα και αν έχουμε και δημιουργεί προβλήματα σε ιδιαίτερα ασθενή σήματα (σκοτεινές εικόνες). Με τις σύγχρονες κάμερες, αυτός αποτελεί την σημαντικότερη πηγή θορύβου της διαδικασίας της σύλληψης.
2.	Θερμικός θόρυβος	Το υλικό σύλληψης της κάμερας υπόκεινται τον θερμικό θόρυβο που συνοδεύει κάθε ηλεκτρική ή μηχανική διαδικασία. Για αυτό τον λόγο είναι σίγουρο ότι η κάμερα θα «μετράει» ακόμα και όταν βρίσκεται στο απόλυτο σκοτάδι. Το ρεύμα που μετράται λέγεται dark current. Δύο τεχνικές που μπορούν να εφαρμοστούν για την μείωση του dark current είναι η ψύξη της διάταξης της κάμερας (οπότε ο θερμικός θόρυβος μειώνεται) και ο υπολογισμός της μέσης τιμής του και αφαίρεση αυτού από το σήμα. Η δεύτερη τεχνική μπορεί να μειώνει φαινομενικά το μετρούμενο dark current αλλά δεν μειώνει την τυπική του απόκλιση και πιθανώς περιορίζει το μετρούμενο δυναμικό εύρος.
3.	Ηλεκτρονικός θόρυβος ολοκληρωμένου	Αυτός ο θόρυβος οφείλεται στην διαδικασία ανάγνωσης του σήματος από τον αισθητήρα μέσα από τους ημιαγωγούς του ολοκληρωμένου του αισθητήρα της κάμερας, πριν από την μετατροπή του σήματος σε ψηφιακό. Αυτός ο θόρυβος μπορεί να μειωθεί με κατάλληλη ρύθμιση της συχνότητας σύλληψης και πιο ποιοτικό ηλεκτρονικό σύστημα. Θα έπρεπε να σημειώσουμε ότι σε μερικές περιπτώσεις το σήμα μπλε σήμα ενισχύεται περισσότερο από το πράσινο και το κόκκινο γεγονός που μπορεί να οδηγεί σε διαφορετική ποσότητα θορύβου σε αυτό το κανάλι.
4.	Θόρυβος κβαντισμού	Ο θόρυβος κβαντισμού προκαλείται κατά την διαδικασία ψηφιοποίησης από το A/D. Αυτός εξαρτάται από των αριθμό των bits στον οποίο γίνεται ψηφιοποίηση. Όσο περισσότερα bits, τόσο λιγότερος θόρυβος.

2.2.2.2 Προδιαγραφές

Είναι πολύ σημαντικό να ελέγχουμε τις προδιαγραφές της κάμερας ώστε να καλύπτουν τις ανάγκες της εφαρμογής μας. Αυτό είναι πολύ σημαντικό ιδιαίτερα αν έχουμε υψηλές απαιτήσεις από αυτήν π.χ. θέλουμε να λαμβάνει εικόνες με πολύ ασθενή φωτισμό ή θέλουμε να έχει καλή γραμμικότητα ως προς την ένταση ή ως προς το μήκος κύματος.

1.	Γραμμικότητα	Είναι επιθυμητό η σχέση μεταξύ της πραγματικής έντασης του φωτός και του μετρούμενου σήματος να είναι γραμμική. Στην πράξη ένα τυπικό μοντέλο του σήματος εξόδου out ως προς το σήμα εισόδου a είναι το εξής $out = gain \cdot a^{\gamma} + offset$ όπου ο όρος γ χαρακτηρίζει την κάμερα. Στην ιδανική περίπτωση θα έπρεπε γ=1 και offset = 0. Ένα θετικό είναι αν ο κατασκευαστής έχει μετρήσει και δίνει τις παραμέτρους της παραπάνω

[14] Digital Signal Processing Handbook, Vijay K. Madisetti, Douglas B. Williams, CRC Press, 1999

		σχέσης οπότε μπορεί κανείς να αντισταθμίσει σφάλματα υπολογιστικά με τον κίνδυνο όμως να μειώσει το δυναμικό εύρος.
2.	Απόλυτη ευαισθησία	Για να προσδιορίσουμε την ευαισθησία αυτή χρειαζόμαστε πληροφορίες για τον θόρυβο. Αν ο θόρυβος έχει για παράδειγμα σ = 100 φωτοηλεκτρόνια, τότε για να εγγυηθούμε την ανίχνευση ενός σήματος θα έπρεπε αυτό να έχει ένταση μεγαλύτερη από π.χ. 3σ δηλαδή 300 φωτοηλεκτρόνια. Αυτή η ελάχιστη μετρούμενη ποσότητα είναι η απόλυτη ευαισθησία. Αν θεωρήσουμε ότι μόνο ο φωτονικός θόρυβος υπάρχει, τότε η απόλυτη ευαισθησία θα είναι 10 φωτοηλεκτρόνια, τιμή που επιτυγχάνεται με τις σύγχρονες κάμερες.
3.	Σχετική ευαισθησία	Η σχετική ευαισθησία ορίζεται ως $S = gain^{-1}$, με το $gain$ όπως ορίστηκε στην σχέση της γραμμικότητας παραπάνω. Αυτή μας λέει ουσιαστικά πόσα φωτοηλεκτρόνια χρειάζονται για να ανιχνευτούν δύο διαφορετικοί τόνοι γκρίζου.
4.	Signal to noise ratio	Οι τιμές για τους τύπους σφάλματος που παρουσιάστηκαν παραπάνω και αφορούν την κάμερα αποτελούν τις πληροφορίες για το SNR της κάμερας.
5.	Σκίαση	Κατά την λήψη μίας εικόνας μετράται η σκίαση τόσο από εξωτερικούς λόγους όσο και από ανομοιογένεια της ευαισθησίας από pixel σε pixel της κάμερας. Αυτό το φαινόμενο μπορεί να αντιμετωπιστεί με διάφορους τρόπους όπως μοντελοποίηση της ευαισθησίας κάθε pixel.
6.	Σχήμα pixel	Τα pixels συνήθως δεν είναι τετράγωνα γιατί εμφανίζονται σε οθόνες με αναλογία πλάτους – ύψους 4:3. Αυτό έχει ως αποτέλεσμα το σχήμα να φαίνεται σωστά στην οθόνη αλλά οι αναλογίες των αντικειμένων αν τις μετρήσουμε σε pixels να είναι διαφορετικές από τις πραγματικές.
7.	Ποσοστό πλήρωσης	Αυτό ισχύει για τις CCD κάμερες και είναι ο λόγος του εμβαδού του πυριτίου το οποίο είναι φωτοευαίσθητο προς το συνολικό εμβαδόν. Αυτός συνήθως δεν είναι 100% γιατί χρειάζεται χώρος τόσο για τα ηλεκτρονικά σάρωσης όσο και για να υπάρχει χώρος ανάμεσα σε γειτονικά pixel για λόγους ασφαλείας. Αν είναι πολύ κοντά το ένα στο άλλο μπορεί ηλεκτρόνια να μεταπηδήσουν από το ένα pixel σε γειτονικό (blooming).
8.	Φασματική ευαισθησία	Η ευαισθησία αλλάζει σε διαφορετικά μήκη κύματος της προσπίπτουσας ακτινοβολίας. Υπάρχουν κάμερες με μεγαλύτερη ευαισθησία στις υπέρυθρες, στο ορατό ή σε άλλα μέρη του φάσματος. Αυτό μπορεί να οφείλεται εκτός από τον τύπο του αισθητήρα και σε οπτικά φίλτρα που τοποθετούνται στις κάμερες. Η φασματική ευαισθησία πρέπει να λαμβάνεται υπόψη και για την φασματική ανόρθωση της εικόνας μετά αν αυτό είναι απαραίτητο.
9.	Ταχύτητα κλείστρου	Αυτή η ταχύτητα ρυθμίζει τον χρόνο έκθεσης της φωτοευαίσθητης επιφάνειας στην ακτινοβολία. Αυτή μπορεί να είναι από μία εικόνα κάθε 500ns έως όσο μεγάλη θέλει ο χρήστης.
10.	Ταχύτητα ανάγνωσης δεδομένων	Αυτός είναι ο ρυθμός με τον οποίο μπορούν να μεταφερθούν δεδομένα από τον δίαυλο του φωτοευαίσθητου αισθητήρα. Αυτός για standard video δίνεται από την σχέση: $$R = \left(\frac{images}{s}\right)\left(\frac{lines}{image}\right)\left(\frac{pixels}{line}\right)$$

2.2.2.3 Τοποθέτηση

Η εγκατάσταση ενός συστήματος επεξεργασίας και αναγνώρισης εικόνας σε κάποιο χώρο πρέπει να γίνεται με αρκετή προσοχή ώστε τα αντικείμενα που θέλουμε να μελετήσουμε να φαίνονται καθαρά και χωρίς παραμορφώσεις. Σε μερικές εφαρμογές αυτό είναι το πιο κρίσιμο μέρος[15] ενώ σε άλλες απλά λίγη προσοχή αρκεί.

[15] Intelligent Image Processing, Steve Mann, University of Toronto, 2002, John Wiley & Sons

Εικόνα 24. Τοποθέτηση κάμερας: Η εικόνα στην οθόνη αριστερά συμπληρώνει το τοπίο[15]

Αυτά τα οποία πρέπει να προσέχουμε κατά την τοποθέτηση μίας κάμερας είναι τα εξής:

1. Φωτισμός	Η τοποθέτηση της κάμερας πρέπει να μην εμποδίζει τον φωτισμό του αντικειμένου και να λαμβάνει την εικόνα καθαρά. Ο επαρκής φωτισμός πρέπει να εξασφαλίζεται ακόμα και με τεχνητά μέσα. Η όψη δεν θα πρέπει να έχει σκιάσεις ούτε από την κάμερα ούτε από τα αντικείμενα που βρίσκονται κοντά.
2. Παραμορφώσεις	Η απόσταση από το παρατηρούμενο αντικείμενο πρέπει να είναι αρκετή ώστε να φαίνονται ξεκάθαρα οι λεπτομέρειες που μας ενδιαφέρουν. Επιπλέον θα πρέπει να αποτρέπεται η λήψη με παραμορφώσεις π.χ. από γωνία.
3. Σταθερή εικόνα	Σε περίπτωση που υπάρχουν κραδασμοί στον χώρο θα πρέπει η στήριξη της κάμερας να γίνεται με τέτοιο τρόπο ώστε να εξασφαλίζεται σταθερή εικόνα. Αυτό μπορεί να γίνει με ειδικές αντικραδασμικές διατάξεις και με την σωστή επιλογή του σημείου στήριξης.

2.2.3 Πρότυπα αναλογικού και ψηφιακού Video

Ανάμεσα στην κάμερα και τον ψηφιακό επεξεργαστή σήματος (DSP) παρεμβάλλονται πρωτόκολλα αναλογικού ή (και) ψηφιακού video. Στις περισσότερες, εκτός από λίγες πολύ εξειδικευμένες εφαρμογές, χρησιμοποιούνται τα τυποποιημένα πρωτόκολλα αναλογικού (PAL – NTSC) και ψηφιακού (ITU-R BT.656) video. Όποιος θέλει να υλοποιήσει σύστημα βασισμένο σε DSP είναι απαραίτητο να γνωρίζει αρκετά πράγματα για αυτά γιατί με βάση αυτά θα καταφέρει να πάρει εικόνα από μία κάμερα ή να την εμφανίσει σε μία οθόνη. Αρκετές φορές δεν θα υπάρχει κάποιος driver που να κάνει την δουλειά για εμάς και συνεπώς η επίπονη εργασία της δημιουργίας όλων των σημάτων θα πρέπει να γίνει εξ' αρχής. Πολύ καλή αναφορά σχετικά με το interface σήματος video με έναν επεξεργαστή BlackFin και της σχετικής θεωρίας μπορεί κανείς να βρει στο σχετικό Technical Note[16] της Analog Devices®.

2.2.3.1 Αναλογικό video, PAL, NTSC

Δύο είναι οι πιο διαδεδομένοι τύποι αναλογικού video, το PAL που χρησιμοποιείται στην Ευρώπη και το NTSC που χρησιμοποιείται στην Αμερική. Στον παρακάτω πίνακα φαίνονται τα χαρακτηριστικά των δύο τύπων.

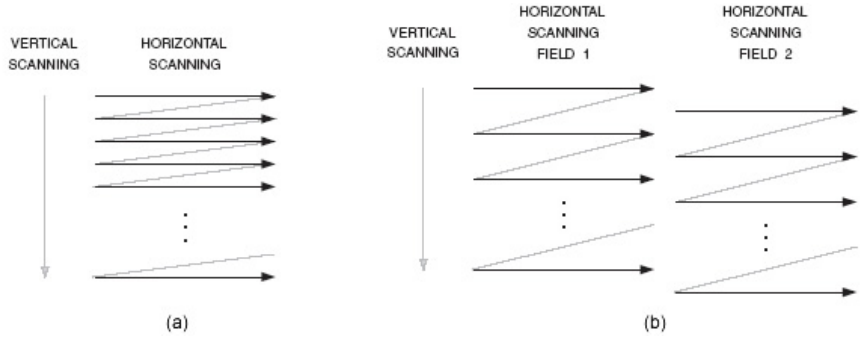
Ιδιότητα	NTSC	PAL
Πλαίσια / δευτερόλεπτο	30	25
ms / πλαίσιο	33.3	40.0
Γραμμές / πλαίσιο	525	625

[16] EE-203, Interfacing the ADSP-BF535/ADSP-BF533 Blackfin® Processor to NTSC/PAL video decoder over the asynchronous port, Thorsten Lorenzen, 2003

Interlace	2:1	2:1
μs / γραμμή	63.56	64.00

Όπως μπορούμε να παρατηρήσουμε οι διαφορές είναι μικρές και έχουν τις ρίζες τους στην διαφορά συχνότητας του ρεύματος τροφοδοσίας. Στο Αμερικάνικο σύστημα NTSC όπου η τάση τροφοδοσίας έχει συχνότητα 60Hz είναι πιο εύκολος ο συγχρονισμός αν μεταφέρονται 30 εικόνες ανά δευτερόλεπτο ενώ στο Ευρωπαϊκό (PAL) όπου η τάση τροφοδοσίας είναι 50Hz είναι πιο εύκολο να μεταφέρονται 25 εικόνες ανά δευτερόλεπτο.

Πολλές λεπτομέρειες για τα δύο αυτά πρωτόκολλα μπορεί κανείς να βρει στο κεφάλαιο 8 του Video Demystified[13]. Αυτό το οποίο πρέπει να θυμάται κανείς είναι ότι για να αποφευχθεί το τρεμόπαιγμα, τα σήματα της αναλογικής τηλεόρασης είναι πεπλεγμένα (interlaced) πράγμα που σημαίνει ότι ένα καρέ μεταφέρεται σε δύο πλαίσια όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 25. (a) Μη πεπλεγμένο video (b) πεπλεγμένο video

Επί πλέον πολλά σήματα συγχρονισμού χρειάζονται τα οποία μεταφέρονται μέσα στο σήμα της εικόνας. Ένα για κάθε γραμμή (οριζόντιος συγχρονισμός) και ένα για κάθε πλαίσιο (κατακόρυφος συγχρονισμός).

2.2.3.2 Ψηφιακό video, BT.656

Πολλοί τύποι ψηφιακού video υπάρχουν[17]. Ο κυριότερος είναι αυτός που χρησιμοποιούμε εμείς στην εφαρμογή μας, ο BT 656. Αυτός περιγράφει interlaced σήματα σε αναλύσεις 858 x 525 στα 30 πλαίσια ανά δευτερόλεπτο (αντίστοιχη του NTSC) και 864 x 625 στα 25 πλαίσια ανά δευτερόλεπτο (αντίστοιχη του PAL).

Ο συγχρονισμός επιτυγχάνεται με την βοήθεια μίας ειδικής ακολουθίας δεδομένων. Αυτή έχει δύο μορφές την SAV (Start of Active Video) και την EAV (End of Active Video). Μέσα σε αυτήν κωδικοποιούνται τρία πολύ σημαντικά σήματα, τα H (Horizontal Blanking), V (Vertical Blanking), F (Frame 1/Frame 2). Όταν έχουμε μετάβαση του H από μηδέν σε ένα, η ακολουθία είναι τύπου EAV ενώ όταν έχουμε μετάβαση από ένα σε μηδέν η ακολουθία είναι τύπου SAV.

	8-bit Data								10-bit Data	
	D9 (MSB)	D8	D7	D6	D5	D4	D3	D2	D1	D0
preamble	1	1	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
status word	1	F	V	H	P3	P2	P1	P0	0	0

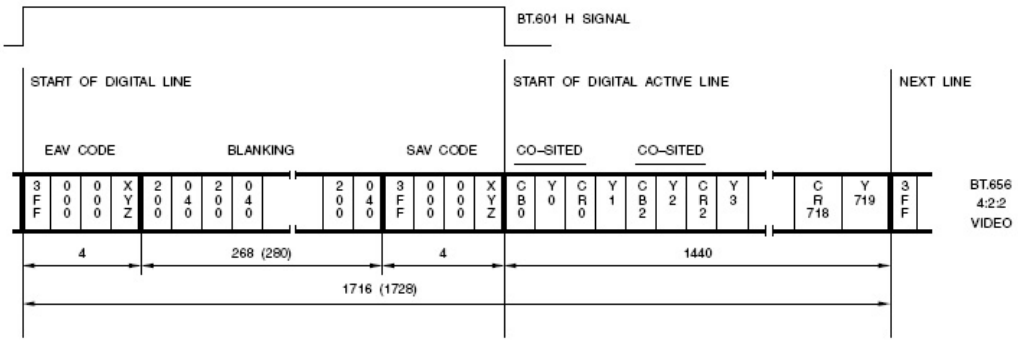
Εικόνα 26. Ακολουθία EAV/SAV

Στην Εικόνα 26 βλέπουμε την μορφή των ακολουθιών EAV/SAV. Όπως μπορούμε να συμπεράνουμε, η μορφή τους σε δεκαεξαδικό θα είναι 0xFF0000XX όπου το τελευταίο byte είναι

[17] Table 6.1, p. 93. Video Demystified, Keith Jack, LLH Technology Publishing, 2001

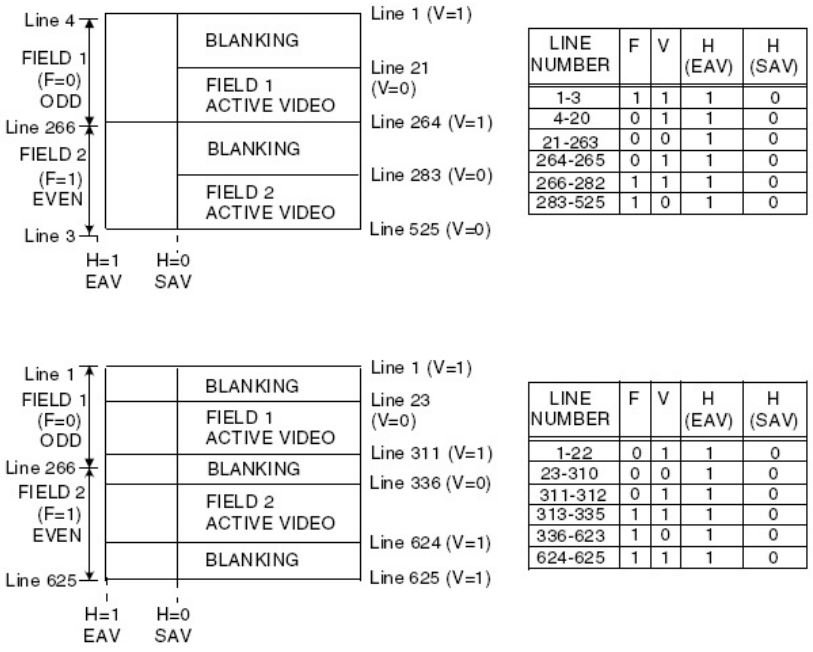
αυτό που ουσιαστικά διαφοροποιείται. Τα P0 έως P3 είναι τρία bit προστασίας που περιέχουν κωδικοποιημένα και πάλι τα F,V,H ώστε να μπορεί ο δέκτης να κάνει έλεγχο για το αν το byte αυτό μεταφέρθηκε σωστά ή όχι και πιθανώς να το διορθώσει αν η «βλάβη» είναι το πού ενός bit.

Η μορφή των δεδομένων video που μεταφέρεται είναι τύπου YcbCr (BT.601 video format) το οποίο περιγράφηκε στην ενότητα με τους χώρους χρώματος. Το σημαντικό είναι ότι γίνεται συμπίεση τύπου 4:2:2. Αυτό σημαίνει ότι για κάθε δύο σημεία στέλνεται μία μόνο φορά η πληροφορία χρώματος Cb, Cr και δύο φορές (μία για κάθε σημείο) η πληροφορία φωτεινότητας Y. Έχει παρατηρηθεί ότι η αντίληψη του ανθρώπου στις διαφορές γειτονικών χρωμάτων είναι πολύ ασθενέστερη από ότι η αντίληψη διαφορετικής φωτεινότητας. Έτσι, δύο γειτονικά σημεία μοιράζονται το ίδιο χρώμα με διαφορετική φωτεινότητα το κάθε ένα και με αυτό τον τρόπο έχουμε μείωση της μεταφερόμενης πληροφορίας κατά 33%.



Εικόνα 27. Πληροφορία για μία γραμμή κατά BT656

Στην Εικόνα 27 φαίνεται η κωδικοποίηση μίας γραμμής κατά BT656. Μέσα σε παρένθεση φαίνονται η τιμές για συστήματα 864 x 625 (αντίστοιχα του PAL) ενώ εκτός παρένθεσης τα 858 x 525 (αντίστοιχα του NTSC). Εμάς μας ενδιαφέρει κυρίως η έκδοση αντίστοιχη του PAL. Όπως μπορούμε να παρατηρήσουμε, μία γραμμή αρχίζει με ένα σήμα οριζόντιου συγχρονισμού. Αυτό σηματοδοτείται με μία εντολή EAV. Από εκεί και πέρα και μέχρι να ανιχνευτεί το επόμενο SAV όλα τα δεδομένα θεωρούνται πληροφορία συγχρονισμού. Σε αυτό τον «ελεύθερο χρόνο» συνήθως κωδικοποιούνται πληροφορίες teletext, ήχου ή υποτιτλισμού. Μετά το τέλος του σήματος SAV 720 δείγματα (1440 bytes) πληροφορίας μεταδίδονται που είναι η πληροφορία έντασης και χρώματος για κάθε σημείο της γραμμής. Στην συνέχεια ακολουθεί η μετάδοση της επόμενης γραμμής.



Εικόνα 28. Ενεργό σήμα και σήματα συγχρονισμού για video 525 και 625 γραμμών[18]

Στην Εικόνα 28 βλέπουμε τη μορφή ενός ολόκληρου πλαισίου για τις περιπτώσεις των 525 και των 625 γραμμών. Θα περιγράψουμε την περίπτωση των 625 γραμμών (κάτω) Οι γραμμές 1 έως 22 είναι γραμμές κατακόρυφου συγχρονισμού. Πληροφορία εικόνας δεν περιέχεται σε αυτές τις γραμμές και μπορούν να χρησιμοποιηθούν για την μεταφορά δεδομένων γενικής φύσεως. Σε αυτές τις γραμμές, το σήμα V κατά τα EAV και SAV πρέπει να είναι 1 που σηματοδοτεί την κατάσταση κάθετου συγχρονισμού. Οι επόμενες 288 γραμμές (23-310) μεταφέρουν την ωφέλιμη εικόνα video για το μονό πλαίσιο (Field 1). Μετά ακολουθούν δύο γραμμές κατακόρυφου συγχρονισμού του μονού πλαισίου και 23 γραμμές συγχρονισμού έναρξης του ζυγού πλαισίου (πλαίσιο 2). Στην συνέχεια έρχονται άλλες 288 γραμμές (336-623) ωφέλιμου σήματος video του ζυγού πλαισίου. Στο τέλος ακολουθούν δύο τελευταίες γραμμές κατακόρυφου συγχρονισμού του ζυγού πλαισίου. Αυτή η διαδικασία επαναλαμβάνεται 25 φορές το δευτερόλεπτο μεταφέροντας τα καρέ του ψηφιακού video από τον πομπό στον δέκτη.

Όπως βλέπουμε, 16% της πληροφορίας κάθε γραμμής και 7% κάθε πλαισίου είναι πληροφορία συγχρονισμού. Αυτή η «σπατάλη» επιβάλλεται για λόγους εύκολης μετατροπής σε μορφές αναλογικού video και το κυριότερο για την εύκολη οδήγηση και σωστή λειτουργία των κλασικών οθονών καθοδικού σωλήνα. Ο χώρος όπως είπαμε αυτός μπορεί να αξιοποιηθεί για τη μεταφορά πληροφοριών άλλου τύπου.

2.3 Ψηφιακή επεξεργασία εικόνας

Η επεξεργασία εικόνας είναι υποκλάδος της επεξεργασίας σήματος. Η εικόνα και το video είναι σήματα δύο και τριών διαστάσεων αντίστοιχα. Όλες λοιπόν οι τεχνικές που γνωρίζαμε για τα μονοδιάστατα σήματα (π.χ. ήχος) υπάρχουν και για την εικόνα/video. Μία πραγματική εικόνα είναι όπως είπαμε και προηγουμένως μία συνάρτηση $f(x,y)$ της έντασης του φωτός. Με την βοήθεια της κάμερας ή άλλου μηχανισμού μετατρέπουμε μία περιοχή αυτής της εικόνας σε μία διακριτή εκδοχή $F(j,k)$ και $-J \le j \le J$ και $-K \le k \le K$ κατάλληλη για την αποθήκευση και επεξεργασία σε κάποιο επεξεργαστή σήματος. Αυτή μπορεί να παρασταθεί και με την μορφή πίνακα $\mathbf{F}[m,n]$. Υπάρχουν πάρα πολλές τεχνικές για την επεξεργασία εικόνας και πάρα πολλοί μετασχηματισμοί που μπορεί κανείς να εφαρμόσει προκειμένου να πετύχει την βελτίωση της. Οι τρεις κατηγορίες λειτουργιών που θα βρει κανείς σχεδόν σε κάθε σύστημα επεξεργασίας και αναγνώρισης εικόνας είναι οι εξής:

[18] Σελίδα 11-18, ADSP-BF533 Blackfin™ Processor Hardware Reference

1. Τεχνικές ιστογράμματος
2. Τεχνικές βασισμένες στην συχνότητα
3. Τεχνικές ανίχνευσης αιχμών

Αυτές θα εξετάσουμε αναλυτικότερα στις επόμενες ενότητες.

2.3.1 Τεχνικές ιστογράμματος

Με την βοήθεια των τεχνικώς ιστογράμματος μπορούμε να ανορθώσουμε προβλήματα στην κατανομή φωτεινότητα της εικόνας. Μία εικόνα που είναι υπερφωτισμένη ή υποφωτισμένη μπορεί να βελτιωθεί ώστε να ξεχωρίζουν τα χαρακτηριστικά που μας ενδιαφέρουν.

Για κάθε εικόνα ορίζονται σημαντικά στατιστικά μεγέθη όπως:

1. Η κατανομή πιθανότητας της φωτεινότητας $P(a)$ που μας δίνει την πιθανότητα ένα σημείο τυχαία επιλεγμένο από την εικόνα να έχει τιμή φωτεινότητας μικρότερη ή ίση με a. Προφανώς η συνάρτηση αυτή είναι μη φθίνουσα και συνεπώς $dP/da \geq 0$.
2. Η κατανομή πυκνότητας πιθανότητας της φωτεινότητας. Αυτή είναι η πιθανότητα ένα τυχαίο σημείο της εικόνας να έχει τιμή φωτεινότητας μεταξύ a και $a + \Delta a$. Με δεδομένο την κατανομή πιθανότητας που ορίσαμε παραπάνω, η κατανομή πυκνότητας πιθανότητας μπορεί να οριστεί ως $p(a) = dP(a)/da$.

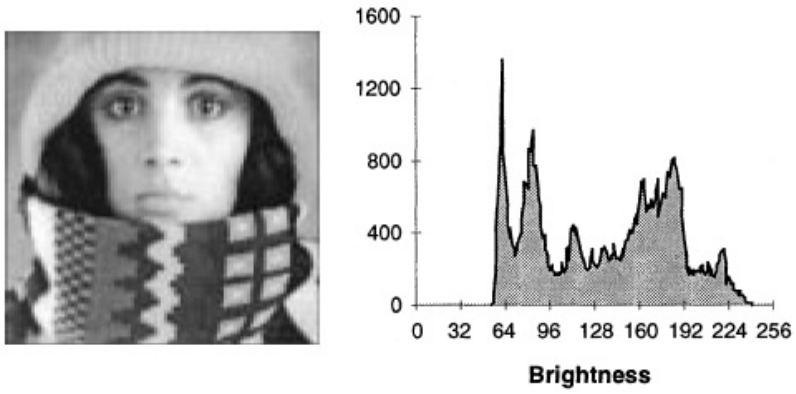
Σε μία τυχαία εικόνα μπορούμε να μετρήσουμε τα παραπάνω μετρώντας τον αριθμό των σημείων (pixels) με μία τιμή a. Προφανώς αυτό είναι μία συνάρτησης της φωτεινότητας $h[a]$. Αυτή λέγεται συνάρτηση ιστογράμματος. Αν ορίσουμε τη συνολική φωτεινότητα μίας εικόνας με διαστάσεις M x N ως το άθροισμα της φωτεινότητας όλων των σημείων:

$$\Lambda = \sum_{n=1}^{N} \sum_{m=1}^{M} F(m,n) = \sum_{\alpha} h[a]$$

τότε η συνάρτηση πυκνότητας πιθανότητας για την εικόνα αυτή δίνεται από τη σχέση

$$p[a] = \frac{h[a]}{\Lambda}$$

Προφανώς η συνάρτηση αυτή είναι κανονικοποιημένη.



Εικόνα 29. Ιστόγραμμα εικόνας

Στην Εικόνα 29 βλέπουμε μία εικόνα και τη συνάρτηση ιστογράμματός της. Στην περιοχή από 0 έως 50 παρατηρούμε ότι δεν υπάρχει κανένα σημείο. Επιπλέον μπορούμε να ξεχωρίσουμε εύκολα τρεις περιοχές, την περιοχή γύρω από το 64 (γκρίζο), την περιοχή γύρω από το 75 (γκρίζο) και την περιοχή γύρω από το 180 (άσπρο). Προφανώς αυτές οι τρεις περιοχές είναι το φόντο, το καπέλο και οι άσπρες περιοχές στο πρόσωπο και το κασκόλ.

Αν η εικόνα που μετράμε έχει συνολικά $C = M \cdot N$ σημεία τότε ορίζονται η μέση τιμή και η διασπορά της φωτεινότητας:

$$m_F = \frac{1}{C}\sum_{n=1}^{N}\sum_{m=1}^{M} F(m,n),\ \sigma^2 = \sqrt{\left(\frac{1}{C-1}\right)\left(\sum_{n=1}^{N}\sum_{m=1}^{M} F(m,n) - C \cdot m_F\right)}$$

Με βάση τις τιμές του ιστογράμματος αυτά μπορούν να εκφραστούν πιο απλά ως:

$$m_F = \frac{1}{C}\sum_{\alpha} \alpha \cdot h[a],\ \sigma^2 = \sqrt{\left(\frac{1}{C-1}\right)\left(\sum_{a} \alpha^2 \cdot h[a]\right) - C \cdot m_F^2}$$

Με βάση το ιστόγραμμα μπορούμε να σχεδιάσουμε μία συνάρτηση μετασχηματισμού $a' = t(a)$ που μετασχηματίζει την φωτεινότητα κάθε σημείου της εικόνας και θα την ανορθώνει έτσι ώστε να έχει το επιθυμητό ιστόγραμμα και να ξεχωρίζουν καθαρά τα χαρακτηριστικά που θέλουμε. Τέτοιες συναρτήσεις είναι τα γνωστά μας brightness/contrast που έχει που έχει σχεδόν κάθε δέκτης τηλεόρασης. Συγκεκριμένα το πρώτο είναι μία πρόσθεση σταθεράς και το δεύτερο ένας πολλαπλασιασμός με σταθερά. Επίσης μία τέτοια συνάρτηση μπορεί να ανορθώσει μη γραμμικότητα των μέσων λήψης ή αναπαραγωγής εικόνας που περιγράψαμε σε προηγούμενη ενότητα (gamma correction).

Στο ιστόγραμμα μίας εικόνας βλέπουμε ότι παρουσιάζει μία ελάχιστη και μία μέγιστη τιμή πέρα από τις οποίες η τιμή του ιστογράμματος είναι 0. Για παράδειγμα οι τιμές αυτές για την Εικόνα 29 είναι minimum = 50, maximum = 244. Η εικόνα θα φαίνεται καλύτερα αν εκμεταλλεύεται όλο το δυναμικό εύρος της δηλαδή έχει για ελάχιστο το 0 και για μέγιστο το 255. Για τον λόγο αυτό μπορούμε να χρησιμοποιήσουμε τη συνάρτηση μετασχηματισμού

$$t(a) = \frac{255}{maximum - minimum}(a - minimum)$$

η οποία θα ανορθώσει ικανοποιητικά την εικόνα. Είναι πολύ σημαντικό να το κάνουμε αυτό ειδικά όταν θα γίνει παρακάτω ανίχνευση αιχμών ώστε ακόμα και σε άτονες εικόνες να ξεχωρίζουν ικανοποιητικά οι αιχμές.

2.3.2 Τεχνικές βασισμένες στη συχνότητα

Πιο σύνθετη επεξεργασία εικόνας από αυτή των ιστογραμμάτων μπορεί να γίνει λαμβάνοντας υπόψη την χωρική συχνότητα μίας εικόνας. Στην εργασία αυτή απαραίτητα εργαλεία είναι η συνέλιξη και ο μετασχηματισμός Fourier.

Ορίζεται η συνέλιξη $c(x,y)$ σε δύο διαστάσεις δύο σημάτων $a(x,y)$ και $b(x,y)$ από την σχέση:

$$c(x,y) = a \otimes b = a(x,y) * b(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{\infty} a(\xi,\zeta)\, b(x-\xi, y-\zeta)\, d\xi\, d\zeta$$

ή για διακριτά σήματα:

$$c[m,n] = a[m,n] * b[m,n] = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} a[j,k] b[m-j, n-k]$$

Τρεις πολύ χρήσιμες ιδιότητες είναι:

- η αντιμεταθετική $a * b = b * a$,
- η μεταθετική $a * (b * c) = (a * b) * c = a * b * c$ και
- η επιμεριστική $a * (b + c) = (a * b) + (a * c)$

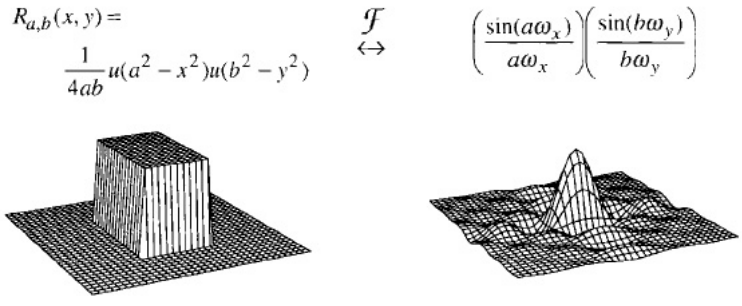
Με την βοήθεια αυτών των τριών σχέσεων μπορούμε να επιταχύνουμε πάρα πολύ τον υπολογισμό φίλτρων. Για παράδειγμα έστω ότι θέλουμε να εφαρμόσουμε σε μία εικόνα $i(x,y)$ ένα χαμηλοπερατό φίλτρο $l(x,y)$ και στην συνέχεια ένα φίλτρο εύρεσης αιχμών $e(x,y)$. Η προφανής διαδικασία θα ήταν να υπολογίσουμε το $(i * l) * e$ το οποίο σημαίνει ότι θα εφαρμόζαμε πρώτα το χαμηλοπερατό φίλτρο και στην συνέχεια το φίλτρο ανίχνευσης αιχμών. Σύμφωνα όμως με τις παραπάνω ιδιότητες μπορούμε να λάβουμε το ίδιο αποτέλεσμα από τη σχέση $i * (l * e)$. Που σημαίνει ότι αρκεί να κάνουμε την συνέλιξη των δύο φίλτρων την οποία μπορούμε να υπολογίσουμε μία φορά κατά την διάρκεια της σχεδίασης και στην συνέχεια κατά τη διάρκεια της εκτέλεσης να υπολογίζουμε μόνο μία συνέλιξη με την τρέχουσα εικόνα.

Προσοχή θα πρέπει να δοθεί στην χρήση των παραπάνω ιδιοτήτων για συνελίξεις διακριτών φίλτρων. Η καλύτερη τεχνική είναι να σχεδιάζουμε ένα φίλτρο στον συνεχή χώρο, να εφαρμόζουμε τις ιδιότητές που θέλουμε και στο τέλος να διακριτοποιούμε με κάποια συνάρτηση παραθύρου. Αν εφαρμόσουμε τις ιδιότητες της συνέλιξης κατευθείαν σε διακριτά φίλτρα μπορεί να μην πάρουμε ακριβώς τα αποτελέσματα που περιμένουμε εξαιτίας του πεπερασμένου αριθμού των στοιχείων τους.

Ορίζονται ο μετασχηματισμός Fourier και ο ανάστροφός του από τις σχέσεις:

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j(ux+vy)} dxdy\,,\ f(x,y) = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v) e^{j(ux+vy)} dudv$$

Με τη βοήθεια των δύο αυτών σχέσεων μπορούμε να περνάμε από τον χώρο της εικόνας στον χώρο της συχνότητας και αντίστροφα.



Εικόνα 30. Μετασχηματισμός Fourier ενός τετραγώνου

Για παράδειγμα στην Εικόνα 30 μπορούμε να δούμε τον μετασχηματισμό Fourier ενός μοναδιαίου τετραγώνου το οποίο θα φαινόταν σαν ένα άσπρο τετράγωνο σε μαύρο φόντο ως εικόνα.

2.3.2.1 Φίλτρα

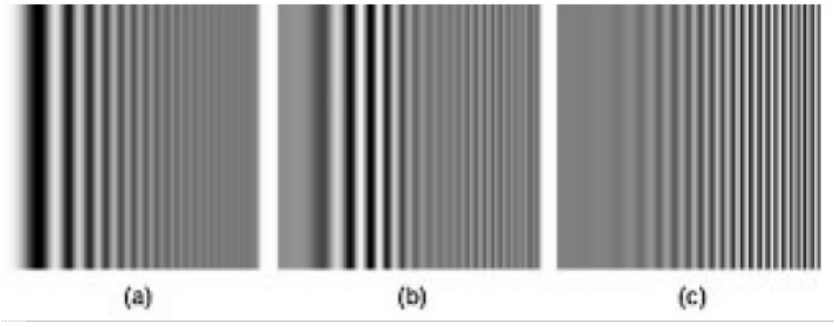
Είναι πολύ εύκολο να σχεδιάσουμε ψηφιακά φίλτρα για εικόνες. Στις περισσότερες περιπτώσεις αρκεί να σχεδιάσουμε τον πυρήνα $\mathbf{h}(n)$ αντίστοιχο ψηφιακό φίλτρο για μία διάσταση και να υπολογίσουμε υπολογίσουμε τους συντελεστές της δισδιάστατης εκδοχής $\mathbf{h}(m,n)$ με βάση την κυκλική συμμετρία. Αν θέλουμε κάτι πιο προχωρημένο όπως για παράδειγμα το φίλτρο μας να έχει διαφορετική συμπεριφορά για κάθε άξονα, μπορούμε να σχεδιάσουμε το φίλτρο στην συνεχή του μορφή και στο τέλος να διακριτοποιήσουμε με τις δισδιάστατες εκδόσεις των, γνωστών μας από την μονοδιάστατη περίπτωση, συναρτήσεων παραθύρου.

Για παράδειγμα δύο απλές εκδοχές για ένα χωρικό χαμηλοπερατό είναι φίλτρο είναι το εξής:

$$h_{rect}[j,k] = \frac{1}{16}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix},\ h_{circ}[j,k] = \frac{1}{12}\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Την συνάρτηση μεταφοράς του πρώτου φίλτρου την είδαμε είδη στην Εικόνα 30. Παρατηρούμε ότι ο κεντρικός λοβός βρίσκεται στις χαμηλές συχνότητες (κέντρο του χώρου Fourier) ενώ σε ψηλότερες συχνότητες η απολαβή μικραίνει σημαντικά. Κατά τα γνωστά η συνέλιξη δύο συναρτήσεων στον κανονικό χώρο οδηγούν σε πολλαπλασιασμό στο χώρο Fourier. Συνεπώς αν υπολογίσουμε την συνέλιξη του παραπάνω πυρήνα με την εικόνα, θα παρατηρήσουμε εξασθένηση των υψηλών συχνοτήτων.

Με την χρήση ενός χαμηλοπερατού φίλτρου μπορούμε να αφαιρέσουμε διάφορα είδη θορύβου τα οποία χαρακτηρίζονται από υψηλές συχνότητες π.χ. τυχαία φωτεινά σημεία μέσα στην εικόνα. Με αυτό τον τρόπο μπορούμε να εξασφαλίζουμε ότι ένας τελεστής αναγνώρισης αιχμών δεν πρόκειται να ξεγελαστεί και να αναγνωρίζει μεμονωμένα σημεία και θόρυβο ως αιχμές.

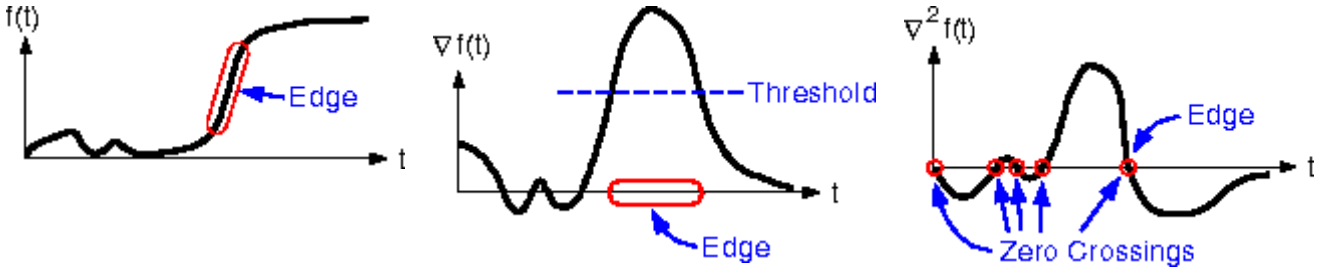


Εικόνα 31. Ψηφιακά φίλτρα σε πρότυπο με διάφορες οπτικές συχνότητες[14]

Στην Εικόνα 31 μπορούμε να δούμε τις επιπτώσεις ενός (α) χαμηλοπερατού, (β) φίλτρου ζώνης και (γ) υψιπερατού φίλτρου. Τέτοιες εικόνες με πρότυπα μπορούν να φανούν πολύ χρήσιμα δείγματα κατά την σχεδίαση φίλτρων.

2.3.3 Αναγνώριση ακμών

Η ανίχνευση των ακμών μας είναι απαραίτητη για τη μορφολογική ανάλυση μίας εικόνας γιατί μας δίνει αίσθηση των ορίων των αντικειμένων που απεικονίζονται. Με αυτόν τον τρόπο μπορούμε στην συνέχεια να εφαρμόσουμε κάποιο μετασχηματισμό όπως ο Hough και να ανιχνεύσουμε ευθείες, κύκλους και άλλα σχήματα.



Εικόνα 32. Μία ακμή, η πρώτη και η δεύτερη παράγωγός του[19]

Στην Εικόνα 32 βλέπουμε μία αιχμή και τις παραγώγους της σε μία διάσταση. Παρατηρούμε ότι για την ανίχνευση ακμών με την πρώτη παράγωγο χρειάζεται να ψάξουμε για μέγιστα ή τουλάχιστον για περιοχές πάνω από ένα κατώφλι ενώ στην δεύτερη παράγωγο αρκεί να ψάξουμε για σημεία μηδενισμού. Λόγω του ότι η παράγωγος πολλαπλασιάζει την ένταση των υψίσυχνων συνιστωσών, η ανίχνευση ακμών είναι συνήθως πολύ ευαίσθητη στον θόρυβο.

Στην δισδιάστατη διακριτή περίπτωση ορίζεται παράγωγος κατά κατεύθυνση και αναπαριστάται με πίνακα π.χ. $\mathbf{h}_x$ κατά τον οριζόντιο άξονα, $\mathbf{h}_y$ κατά τον κατακόρυφο και $\mathbf{h}_\theta = \cos\theta \cdot \mathbf{h}_x + \sin\theta \cdot \mathbf{h}_y$ κατά τυχαία διεύθυνση θ.

Μπορούμε να ορίσουμε τη διανυσματική παράγωγο $\nabla a = \frac{\partial a}{\partial x}\hat{i}_x + \frac{\partial a}{\partial y}\hat{i}_y = (\mathbf{h}_x \otimes a)\hat{i}_x + (\mathbf{h}_y \otimes a)\hat{i}_y$.

Το μέτρο της θα είναι $|\nabla a| = \sqrt{(\mathbf{h}_x \otimes a)^2 + (\mathbf{h}_y \otimes a)^2}$ και η διεύθυνσή της $\arctan\left(\mathbf{h}_y \otimes a \,/\, \mathbf{h}_x \otimes a\right)$.

Υπάρχουν πολλές εκδοχές για του τύπους των παραγώγων $\mathbf{h}_x$ και $\mathbf{h}_y$. Οι πιο ευρέως χρησιμοποιούμενες είναι του Sobel και του Prewitt όπου ορίζονται ως οι παρακάτω πίνακες:

Sobel: $\mathbf{h}_x = \frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$, $\mathbf{h}_y = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

Prewitt: $\mathbf{h}_x = \frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$, $\mathbf{h}_y = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

Αντίστοιχα μπορεί να οριστεί και η δεύτερη παράγωγος ή τελεστής Laplace:

$$\nabla^2 a = \frac{\partial^2 a}{\partial x^2} + \frac{\partial^2 a}{\partial y^2} = (\mathbf{h}_{2x} \otimes a) + (\mathbf{h}_{2y} \otimes a)$$

για τον αντίστοιχο πίνακας υπάρχουν διάφορες εκδοχές όπως:

$$\mathbf{h} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}, \mathbf{h} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}, \mathbf{h} = \begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

Ο θόρυβος μπορεί να προκαλέσει σοβαρά προβλήματα σε αυτόν τον τελεστή. Η λύση είναι να κοιτάξουμε είτε στην εικόνα πριν τον εφαρμόσουμε είτε στην εικόνα μετά την εφαρμογή και να

[19] http://www.owlnet.rice.edu/~elec539/Projects97/morphjrks/moredge.html

εξετάσουμε αν η μεταβολή που προκάλεσε την ανίχνευση ακμής ήταν μεγάλη ή όχι. Μπορούμε για παράδειγμα να πάρουμε το μέγιστο και το ελάχιστο σε ένα 3x3 παράθυρο ενός σημείου μετά την εφαρμογή του τελεστή Laplace και να ανιχνεύσουμε αν η διαφορά τους είναι μεγαλύτερη από το κατώφλι.

Συγκεντρωτικά η διαδικασία ανίχνευσης ακμών με την βοήθεια του τελεστή Laplace έχει ως εξής:

1. Εφαρμόζουμε ένα χαμηλοπερατό φίλτρο στην εικόνα
2. Εφαρμόζουμε τον τελεστή Laplace
3. Βρίσκουμε τα περάσματα από το μηδέν και ελέγχουμε αν το εύρος της μεταβολής που τα προκάλεσε είναι μεγαλύτερο από μία τιμή κατωφλιού

2.4 Εξαγωγή χαρακτηριστικών

Χαρακτηριστικό μίας εικόνας μία ποσότητα την οποία μπορείς να μετρήσεις και με βάση αυτή να ξεχωρίσεις την εικόνα ή ένα μέρος της. Μερικά χαρακτηριστικά είναι φυσικά από της απόψεως ότι ορίζονται με βάση ορατά χαρακτηριστικά της εικόνας ενώ άλλα τεχνητά χαρακτηριστικά εξάγονται ως αποτελέσματα επεξεργασίας της. Φυσικά χαρακτηριστικά είναι για παράδειγμα η φωτεινότητα μίας εικόνας ενώ τεχνητά χαρακτηριστικά είναι τα ιστογράμματα και το φάσμα συχνότητας.

Με την βοήθεια της εξαγωγής χαρακτηριστικών μπορούμε να μετρήσουμε χαρακτηριστικά της εικόνας με βάση τα οποία στη συνέχεια να την κατατάξουμε. Επίσης μπορούμε να εντοπίσουμε αντικείμενα που μας ενδιαφέρουν μέσα στην εικόνα. Αυτή η περίπτωση μας ενδιαφέρει εμάς. Οι κυριότερες μέθοδοι είναι οι εξής:

1. Κατάτμηση βάσει φωτεινότητας	Αυτή την περίπτωση την σκιαγραφήσαμε όταν μιλούσαμε για τα ιστογράμματα. Επιλέγοντας περιοχές με κάποιο συγκεκριμένο εύρος φωτεινότητας μπορούμε να ξεχωρίσουμε αντικείμενα μέσα στην εικόνα. Επίσης μπορούν να χρησιμοποιηθούν ιστογράμματα ανά άξονα ή ανά κατεύθυνση για την ανίχνευση πιο σύνθετων αντικειμένων.
2. Συνάρτηση συσχέτισης	Η μέθοδος αυτή προβλέπει την συνέλιξη μίας εικόνας αναφοράς με την εικόνα που δοκιμάζουμε. Αν η εικόνα αναφοράς εμφανίζεται μέσα στην εικόνα, τότε η συνάρτηση συσχέτισης παρουσιάζει μέγιστο στο συγκεκριμένο σημείο. Αυτή η τεχνική είναι πολύ χρήσιμη και για την αναγνώριση περιοχών με συγκεκριμένο επαναλαμβανόμενο γέμισμα. Τις περισσότερες φορές συμφέρει πάρα πολύ από πλευράς υπολογιστικού φόρτου να εκτελέσουμε τη συσχέτιση στον χώρο Fourier αντί κατευθείαν πάνω στην εικόνα.
3. Μετασχηματισμός Hough	Για τον μετασχηματισμό αυτό μιλήσαμε στο πρώτο κεφάλαιο. Το κυριότερο χαρακτηριστικό του είναι ότι μπορεί να αναγνωρίζει εύκολα γεωμετρικά σχήματα μέσα στην εικόνα. Αυτά εμφανίζονται ως μέγιστα στον χώρο Hough.
4. Ελαστικές χορδές (Snakes)[20]	Αυτές είναι οι πιο σύγχρονες εκδοχές για την αναγνώριση σχημάτων, υπολογίζονται σχετικά γρήγορα και οι εφαρμογές τους είναι πάρα πολλές. Συγκεκριμένα ένα περίγραμμα τοποθετείται αρχικά τυχαία στην εικόνα και μετά αφήνεται να κινηθεί από την δυναμική της εικόνας (παράγωγοι, φωτεινότητα κ.τ.λ.) και με κάποιους περιορισμούς που αποτρέπουν για παράδειγμα το δίπλωμά τους ή την δημιουργία πτυχώσεων ανάλογα με τις ανάγκες της εφαρμογής.

[20] Geodesic Active Regions and Level Set Methods: Contributions and applications in Artificial Vision, Nikos K. Paragios, Ph.D. Thesis, 2000

2.5 Κατηγοριοποίηση

Τα χαρακτηριστικά που εξάγαμε από την παραπάνω διαδικασία είναι στην γενική περίπτωση νούμερα. Μπορούμε να τα ενσωματώσουμε όλα μαζί συμβολίζοντάς τα ως ένα διάνυσμα $\mathbf{x}$. Σε ένα αυτοματοποιημένο σύστημα αναγνώρισης εικόνας δεν μας αρκεί να απεικονίσουμε τις τιμές τους στον χρήστη. Πρέπει να βρούμε ένα τρόπο να τα αξιολογήσουμε και να πάρουμε αποφάσεις με βάση αυτά. Υπάρχει μία ποικιλία μεθοδολογιών[21] για την αναγνώριση προτύπων και την κατηγοριοποίηση δεδομένων πάσης φύσεως.

1.	Παραμετρικές μέθοδοι	Αρκετές από αυτές έχουν βασιστεί στην θεωρία του Bayes η οποία καθορίζει την βέλτιστη απόφαση με βάση την ελαχιστοποίηση του κόστους λάθους και της πιθανότητάς του. Αυτές οι τεχνικές υποθέτουν ότι από πριν γνωρίζουμε την μορφή της πιθανότητας λάθους συναρτήσει του διανύσματος χαρακτηριστικών $\mathbf{x}$ η οποίο αρκετές φορές υποτίθεται ότι είναι κανονική.
2.	Μη παραμετρικές μέθοδοι με δάσκαλο	Άλλες τεχνικές δεν προϋποθέτουν την γνώση της κατανομής πιθανότητας αλλά μπορούν και «μαθαίνουν» τι πρέπει να κάνουν από ένα περιορισμένο αριθμό δεδομένων εκπαίδευσης με την βοήθεια κάποιου «δασκάλου» που μπορεί να τους «δείξει» την σωστή απόφαση.
3.	Μη παραμετρικές μέθοδοι χωρίς δάσκαλο	Επίσης υπάρχουν τεχνικές που μπορούν να μαθαίνουν να ξεχωρίζουν κλάσεις μόνο από δεδομένα εκπαίδευσης. Μπορούν δηλαδή να βρίσκουν στα δεδομένα εκπαίδευσης ομάδες με όμοια χαρακτηριστικά και να αξιολογούν καινούρια δεδομένα κατατάσσοντας τα σε κάποια από τις ομάδες χωρίς να ξέρουν τι είναι οι ομάδες αυτές.

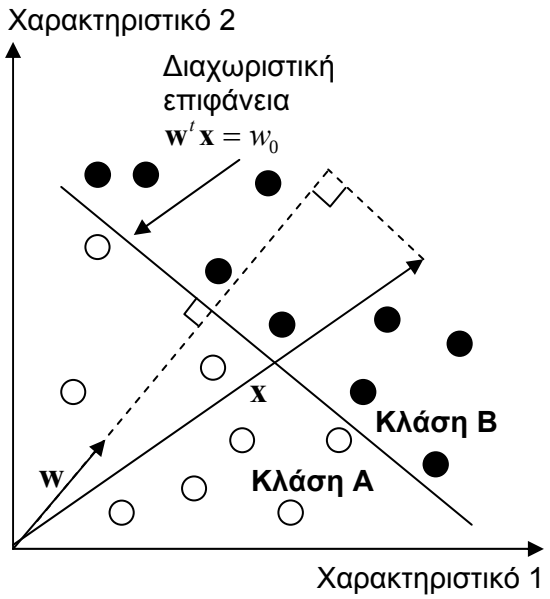
Το σύστημα κατηγοριοποίησης είναι το μέρος του οποίου η σχεδίαση εξαρτάται περισσότερο από οποιοδήποτε άλλο από την εφαρμογή (application specific). Πλήρης γνώση των διαθέσιμων τεχνικών δίνει στον σχεδιαστή την ελευθερία επιλογής της καλύτερης αρχιτεκτονικής για το συγκεκριμένο πρόβλημα. Πολλές φορές διαφορετικές τεχνικές χρειάζονται να συνδυαστούν για να φέρουν τα καλύτερα δυνατά αποτελέσματα.

Πρέπει να σημειωθεί ότι μία από τις πιο διαδεδομένες τεχνικές είναι τα νευρωνικά δίκτυα. Αυτά ανήκουν στις μη παραμετρικές μεθόδους με δάσκαλο. Την τόσο μεγάλη τους αποδοχή την οφείλουν στην σχετικά εύκολη εκπαίδευση και το μεγάλο πλήθος τεχνικών εκπαίδευσης, το μικρό υπολογιστικό κόστος λειτουργία τους μετά την εκπαίδευση και το σχετικά εύκολο μαθηματικό υπόβαθρο που έχει δημιουργηθεί για την μελέτη τους. Την στοιχειώδη δομή ενός νευρωνικού δικτύου, τον νευρώνα (perceptron), θα χρησιμοποιήσουμε και εμείς για το σύστημά μας.

2.5.1. Α αρχιτεκτονική του νευρώνα[22]

Αυτό που ζητάμε είναι να ανιχνεύσουμε πότε ένα διάνυσμα $\mathbf{x}$ ανήκει σε μία κλάση Α ή σε μία κλάση Β. Υποθέτουμε ότι οι κλάσεις αυτές είναι γραμμικά διαχωρίσιμες δηλαδή υπάρχει μία γραμμή στις δύο διστάσεις ή ένα υπερεπίπεδο σε περισσότερες που να μπορεί να διαχωρίσει τις δύο αυτές κλάσεις. Η γραμμή ή το υπερεπίπεδο χωρίζει τον χώρο σε δύο μέρη. Όταν τυχαίο διάνυσμα $\mathbf{x}$ βρίσκεται στο ένα μέρος αυτό ανήκει στην κλάση Α. Όταν βρίσκεται στο άλλο ανήκει στην κλάση Β όπως φαίνεται στην Εικόνα 33.

[21] Pattern Classification, Richard Duda, Peter Hart, David Stork, John Wiley and Sons, 2001
[22] Στατιστική αναγνώριση προτύπων, Θ. Αλεξόπουλος, Α. Τζαμαριουδάκη, 2003



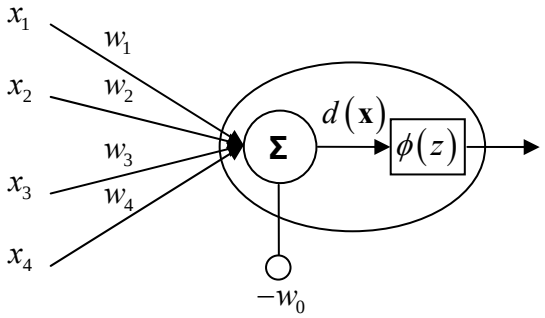
Εικόνα 33. Γραμμικά διαχωρίσιμες κλάσεις

Η γενική εξίσωση μίας γραμμής ή ενός υπερεπιπέδου είναι η $\mathbf{w}^t\mathbf{x} = w_0$. Αν πάρουμε δύο σημεία $\mathbf{x_1}$, $\mathbf{x_2}$ πάνω στην γραμμή τότε έχουμε από την παραπάνω σχέση:

$$\left.\begin{array}{l}\mathbf{w}^t\mathbf{x}_1 = w_0\\ \mathbf{w}^t\mathbf{x}_2 = w_0\end{array}\right\} \Rightarrow \mathbf{w}^t\left(\mathbf{x}_1 - \mathbf{x_2}\right) = 0 \Rightarrow \mathbf{w} \perp \left(\mathbf{x}_1 - \mathbf{x_2}\right)$$

Συνεπώς το διάνυσμα $\mathbf{w}$ είναι κάθετο στην ευθεία και στην ουσία ορίζει την κλίση της ευθείας απόφασης, το $\mathbf{w}^t\mathbf{x}$ είναι το μήκος της προβολής του τυχαίου διανύσματος $\mathbf{x}$ στην κατεύθυνση του $\mathbf{w}$ σε μονάδες $|\mathbf{w}|$ και το w_0 ορίζει την απόσταση της διαχωριστικής ευθείας κατά την κατεύθυνση του $\mathbf{w}$ σε μονάδες $|\mathbf{w}|$. Προφανώς για να ταξινομήσουμε ένα τυχαίο διάνυσμα αρκεί να υπολογίσουμε το $\mathbf{w}^t\mathbf{x}$ και αν αυτό είναι μικρότερο του w_0, τότε ανήκει στην κλάση Α αλλιώς στην κλάση Β ή ισοδύναμα έστω $d(\mathbf{x}) = \mathbf{w}^t\mathbf{x} - w_0$ τότε αν $d(\mathbf{x}) < 0$ ανήκει στην κλάση Α αλλιώς στην κλάση Β. Αν $d(\mathbf{x}) = 0$ δηλαδή το διάνυσμα είναι πάνω στην διαχωριστική ευθεία, τότε μπορούμε να το κατηγοριοποιήσουμε αυθαίρετα.

Η όλη παραπάνω διαδικασία μπορεί να παρασταθεί συμβολικά με την αρχιτεκτονική του νεύρωνα (perceptron).



Εικόνα 34. Η αρχιτεκτονική του νεύρωνα

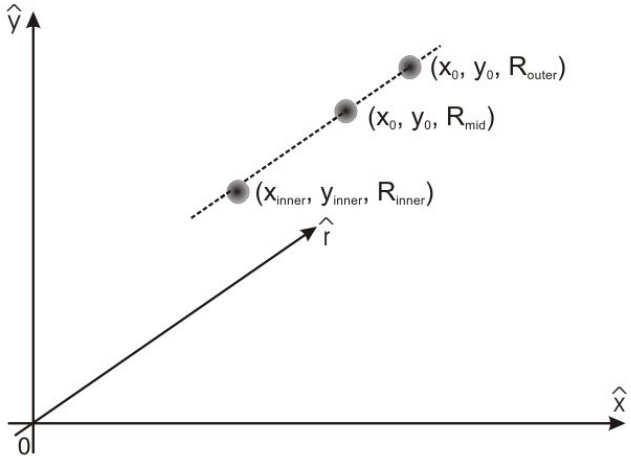
Παρατηρούμε δηλαδή ότι ένας νεύρωνας δεν είναι τίποτα άλλο παρά η υλοποίηση ενός υπερεπιπέδου. Η συνάρτηση $\phi(z)$ ονομάζεται συνάρτηση ενεργοποίησης και μας καθορίζει τι τιμή θα εξάγει ο νεύρωνας για κάθε τιμή της $d(\mathbf{x})$. Συνήθως αυτή είναι μία μοναδιαία βηματική συνάρτηση οπότε και ο νεύρωνας εξάγει 0 για την κλάση Α και 1 για την κλάση Β ή πάνω στην διαχωριστική επιφάνεια.

2.5.1.1 Η αναγνώριση του προτύπου της εφαρμογής μας

Θα γίνουμε πιο σαφείς δίνοντας ένα παράδειγμα για το πως μπορεί να χρησιμοποιηθεί ένας νεύρωνας για την αναγνώριση του προτύπου της εφαρμογής μας.

Εικόνα 35. Οι τρεις κύκλοι του προτύπου που θέλουμε να αναγνωρίσουμε

Αν παρατηρήσει κανείς το πρότυπο το οποίο θέλουμε να αναγνωρίσουμε στην Εικόνα 35 θα δει ότι χαρακτηρίζεται από τρεις κύκλους. Ο ένας είναι ο εξωτερικός κύκλος (έστω C_{outer}), ο άλλος είναι ο κύκλος που διαχωρίζει το μεταλλικό πλαίσιο από το κεραμικό υλικό (έστω C_{mid}) και ο τρίτος ο εσωτερικός κεραμικός κύκλος στήριξης (έστω C_{inner}). Γνωρίζουμε από την γεωμετρία του ανιχνευτή ότι $R_{mid} = 0.7 \cdot R_{outer}$ και $R_{inner} = 0.5 \cdot R_{outer}$. Παρατηρούμε επίσης ότι οι δύο εξωτερικοί κύκλοι έχουν το ίδιο κέντρο ενώ ο εσωτερικός έχει στην γενική περίπτωση διαφορετικό λόγω του ότι βρίσκεται σε διαφορετικό επίπεδο (προεξέχει) και συνεπώς επηρεάζεται πάρα πολύ από την γωνία από την οποία παρατηρούμε τον αισθητήρα.



Εικόνα 36. Το πρότυπο στον χώρο Radon

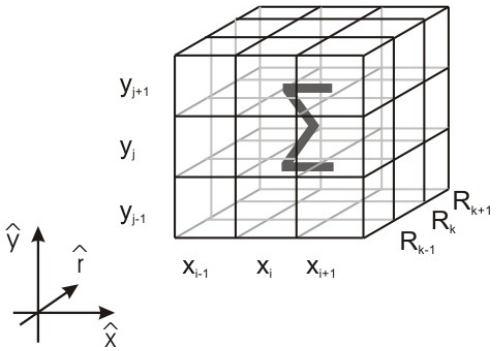
Αν μετασχηματίσουμε την εικόνα αυτή στον χώρο Radon θα παρατηρήσουμε τρία πολύ ισχυρά μέγιστα στα σημεία που αντιστοιχούν στους τρεις κύκλους όπως φαίνεται στην Εικόνα 36. Λόγω της προοπτικής (μπορεί οι κύκλοι να έχουν ελαφρά παραμορφωθεί και να φαίνονται ως ελλείψεις) και λόγω του θορύβου κβαντισμού θα παρατηρούμε υψηλές τιμές τριγύρω από το πραγματικό κέντρο (x_0, y_0) όπως φαίνεται και στο παραπάνω σχήμα. Επιπλέον ο εσωτερικός κύκλος θα έχει κέντρο (x_{inner}, y_{inner}) το οποίο θα ταυτίζεται με το (x_0, y_0) μόνο στην περίπτωση που η κάμερα είναι

απόλυτα ευθυγραμμισμένη με την επιφάνεια του ανιχνευτή. Στη γενική περίπτωση θα είναι διαφορετικό αλλά σχετικά κοντά στο (x_0, y_0).

Αυτό που θα θέλαμε είναι έναν κατηγοριοποιητή (classifier) που να μπορεί να απαντήσει στο ερώτημα αν το τυχαίο σημείο (x, y, R) του χώρου Radon είναι κέντρο του εξωτερικού δακτυλίου του προτύπου που θέλουμε να αναγνωρίσουμε ή όχι. Αυτό που έχουμε ως δεδομένο είναι η συνάρτηση $f(x, y, R)$ του μετασχηματισμού Radon της εικόνας. Συνεπώς θα ήταν απλό να βάλουμε σε ένα νεύρωνα ως διάνυσμα εισόδου το

$$\mathbf{x} = \begin{pmatrix} f(x, y, R_{outer}) \\ f(x, y, R_{mid}) \\ f(x, y, R_{inner}) \end{pmatrix} = \begin{pmatrix} f(x, y, R) \\ f(x, y, 0.7 \cdot R) \\ f(x, y, 0.5 \cdot R) \end{pmatrix}$$

και να του ζητήσουμε να μας εκτιμήσει αν το σημείο (x, y, R) αποτελεί κέντρο του κύκλου. Με αυτό τον τρόπο όμως θα αγνοούσαμε ένα πολύ σημαντικό πρόβλημα, τον θόρυβο.



Εικόνα 37. Χωρικά φίλτρα στον τρισδιάστατο χώρο Radon

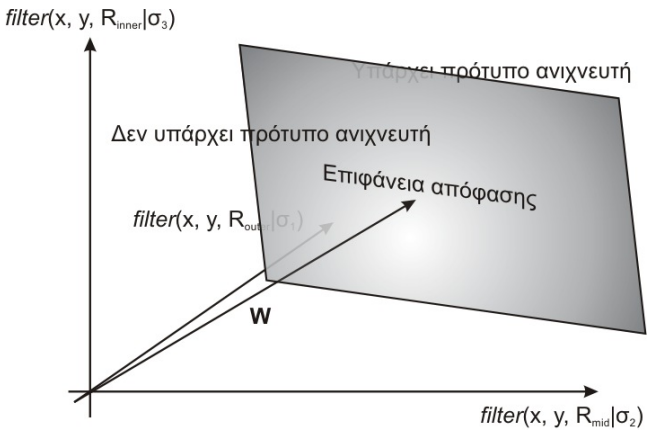
Ο θόρυβος μπορεί να αντιμετωπιστεί με χωρικά φίλτρα στον χώρο Radon. Ορίζουμε ένα χωρικό φίλτρο όπως φαίνεται στην Εικόνα 37 ως:

$$filter(x, y, R \mid \sigma) = \sum_{i=x-\sigma}^{x+\sigma} \sum_{j=y-\sigma}^{y+\sigma} \sum_{k=R-\sigma}^{R+\sigma} w(i-x, j-y, k-R) \cdot f(i, j, k)$$

Το $w(x, y, R)$ για $x, y, R = -\sigma, -\sigma+1, ..., \sigma-1, \sigma$ είναι ο πυρήνας του φίλτρου. Προφανώς θα μπορούσαμε να χρησιμοποιήσουμε και διαφορετικό σ για κάθε άξονα αλλά κάτι τέτοιο δεν χρειάζεται στην συγκεκριμένη εφαρμογή. Τις φιλτραρισμένες αυτές τιμές θα εισάγουμε στον νεύρωνα από τον οποίο θα ζητάμε να αναγνωρίσει αν στο σημείο (x, y, R) έχουμε κύκλο.

$$\mathbf{x} = \begin{pmatrix} filter(x, y, R \mid \sigma_1) \\ filter(x, y, 0.7 \cdot R \mid \sigma_2) \\ filter(x, y, 0.5 \cdot R \mid \sigma_3) \end{pmatrix}$$

Το μόνο που μένει είναι να υπολογίσουμε τα $\mathbf{w}$ και w_0 ώστε να τοποθετήσουμε κατάλληλα την επιφάνεια απόφασης όπως φαίνεται στην Εικόνα 38.



Εικόνα 38. Επιφάνεια απόφασης στον χώρο των χαρακτηριστικών (feature space)

Η εύρεση αυτών των παραμέτρων μπορεί να γίνει πειραματικά δοκιμάζοντας τον classifier με διάφορες εικόνες και παρατηρώντας την απόφασή του.

Θα πρέπει να αντιμετωπιστεί με σκεπτικισμό ο συν-υπολογισμός του εσωτερικού κύκλου C_{inner} λόγω του ότι το κέντρο του μπορεί να μεταβάλλεται. Για να αντιμετωπιστεί αυτό και να μπορεί να συμβάλλει θετικά και ο C_{inner} στην αναγνώριση του προτύπου, πρέπει το σ_3 να είναι αρκετά μεγάλο. Η ανίχνευση του κέντρου του εσωτερικού κύκλου μπορεί να βοηθήσει στην εκτίμηση της γωνίας του tube ως προς το επίπεδο παρατήρησης.

Επιπλέον θα έπρεπε να παρατηρήσουμε ότι η αντίθεση φωτεινότητας ανάμεσα στον εξωτερικό και τον μεσαίο δακτύλιο, δεν είναι τόσο ισχυρή όσο θα θέλαμε. Πιθανώς λοιπόν να μην έχουμε τόσο ισχυρή ένδειξη ακμών για τον μεσαίο δακτύλιο, κάτι το οποίο οδηγεί σε χαμηλή τιμή για τον συντελεστή βάρους του, w_2. Στην οριακή συνθήκη, αυτός μηδενίζεται δηλαδή ο δακτύλιος αυτός δε λαμβάνεται υπόψιν.

Η διαδικασίας αναγνώρισης προτύπου όπως περιγράφηκε παραπάνω δίνει ένα σύνολο k σημείων $\left(x_n, y_n, R_n\right),\ n = 0...k$ στο χώρο Radon στα οποία ανιχνεύεται επιτυχώς η ύπαρξη του προτύπου του ανιχνευτή.

Ένα φαινόμενο το οποίο συμβαίνει πολύ συχνά είναι γύρω από το κέντρο του προτύπου να υπάρχουν και μερικά ακόμα ανιχνευόμενα κέντρα. Αυτό είναι αναμενόμενο αφού όπως είπαμε παραπάνω στον χώρο Radon υπάρχει μία ευρεία περιοχή γύρω από κάθε κέντρο κύκλου με υψηλές τιμές. Προφανώς και αυτά τα σημεία θα πρέπει να συμβάλουν στον υπολογισμό του πραγματικού κέντρου του ανιχνευτή. Για την ανίχνευση τέτοιων συστοιχιών σημείων (clusters) στο χώρο Radon θα μπορούσαμε να εφαρμόσουμε κάποια από τις μη παραμετρικές τεχνικές αναγνώρισης προτύπων χωρίς δάσκαλο ώστε να προσδιορίσουμε τα κέντρα των κλάσεων τα οποία αποτελούν και τα κέντρα των προτύπων. Όπως είπαμε, πολλές φορές πρέπει να συνδυάζουμε τεχνικές προκειμένου να έχουμε τα καλύτερα αποτελέσματα.

Στην περίπτωσή μας έχουμε το πλεονέκτημα ότι γνωρίζουμε ότι όλα τα ανιχνευόμενα κέντρα ενός προτύπου θα πρέπει να βρίσκονται μέσα σε μία σφαίρα στον χώρο Radon με ακτίνα το πολύ r_0 το οποίο μπορούμε να υπολογίσουμε πειραματικά. Με αυτό τον τρόπο υπολογίζοντας την ευκλείδεια απόσταση $\sqrt{\Delta x^2 + \Delta y^2 + \Delta R^2}$ μεταξύ ενός σημείου και όλων των υπολοίπων μπορούμε εύκολα να ομαδοποιήσουμε τα σημεία που ανήκουν στο ίδιο πρότυπο και να πάρουμε ως κέντρο κάτι που να τα συμπεριλαμβάνει όλα π.χ. τον μέσο όρο τους.

2.6 Υπολογιστική γραφική

Στον τομέα της υπολογιστικής γραφικής (computer graphics) οι εξελίξεις τα τελευταία χρόνια είναι ραγδαίες κυρίως λόγω των αυξημένων απαιτήσεων των σύγχρονων λειτουργικών συστημάτων,

της βιομηχανίας των ηλεκτρονικών παιχνιδιών και του κινηματογράφου. Η ανάγκη για δημιουργία φωτορεαλιστικών γραφικών οδήγησε στην υιοθέτηση σύνθετων μοντέλων που προκύπτουν από την μελέτη των φαινομένων της οπτικής και σύνθετων αλλά κομψών αλγορίθμων για την ταχύτερη δημιουργία γραφικών. Τα θεμέλια των computer graphics βασίζονται στη γεωμετρία.

Από ένα σύστημα παραγωγής υπολογιστικών γραφικών (χωρίς κίνηση) έχουμε τις παρακάτω απαιτήσεις[23]:

1. Γεωμετρία δύο και τριών διαστάσεων
    a. Παραμετρική αναπαράσταση καμπυλών, γραμμών, κύκλων, πολυγώνων και κειμένου
    b. Γεωμετρικοί μετασχηματισμοί – χειρισμός πολλών συστημάτων αναφοράς
    c. Clipping: Απομόνωση ενός μέρους του σχήματος σε παραμετρική μορφή
    d. Δείκτες βάθους - απόκρυψη επιφανειών που καλύπτονται από άλλες
    e. Παραμετρική αναπαράσταση σχημάτων τριών διαστάσεων βασισμένη σε ευθύγραμμα τμήματα ή καμπύλες
    f. Αναγνώριση κρυμμένων πλευρών – δημιουργία σχήματος που φαίνεται σε πλάνο (clipping με προοπτική)
2. Rendering (Απόδοση) δύο και τριών διαστάσεων
    a. Σχεδιασμός παραμετρικών καμπυλών, γραμμών, κύκλων και κειμένου
    b. Γέμισμα κλειστών επιφανειών – textures
    c. Μετασχηματισμός γεμισμάτων επιφανειών για συνεπή απεικόνιση
    d. Antialiasing
    e. Απόδοση βάθους – επιπέδων
    f. Σχεδιασμός σχημάτων τριών διαστάσεων με προοπτική
    g. Προχωρημένες δυνατότητες γεμίσματος: σκίαση, αντανακλάσεις, fractals
3. Ειδικά εφέ
    a. Απόδοση νερού – σκόνης – ομίχλης – φωτιάς και άλλων ειδικών μορφών υλικών
    b. Απόδοση ταχύτητας κίνησης – motion blur και άλλων οπτικών φαινομένων

Παρόλο που τα παραπάνω θεωρούνται δεδομένα για όποιον προγραμματίζει σε ενός υψηλού επιπέδου περιβάλλον, π.χ. Windows με υποστήριξη Direct X, στις περισσότερες περιπτώσεις των embedded συστημάτων π.χ. κινητά τηλέφωνα, όλα αυτά πρέπει να κατασκευαστούν από την αρχή, στην καλύτερη περίπτωση αξιοποιώντας έτοιμες βιβλιοθήκες. Και στην περίπτωση ενός συστήματος επεξεργασίας και αναγνώρισης εικόνας το οποίο πρέπει να ΜΗΝ υλοποιηθεί σε κλασικό υπολογιστή (π.χ. για μείωση κόστους – βάρους – αύξηση αυτονομίας σε φορητό σύστημα) το πιθανότερο είναι ότι θα πρέπει να υλοποιηθεί ένα μεγάλο μέρος της υποδομής υποστήριξης γραφικών από την αρχή.

Σε ένα ιατρικό όργανο τρισδιάστατης τομογραφίας θα ήταν απαραίτητες οι δυνατότητες απεικόνισης εικόνων τριών διαστάσεων και τομών. Στην δική μας περίπτωση οι ανάγκες για γραφικά είναι αρκετά πιο περιορισμένες. Αυτό που χρειαζόμαστε είναι η αναπαράσταση στοιχειωδών δισδιάστατων γραφικών και πιο συγκεκριμένα κύκλων και ευθειών, το γέμισμα κάποιων συγκεκριμένων ορθογωνίων, η απεικόνιση στατικού κειμένου και δυναμικών αριθμών. Στην ουσία από αυτά τα μόνα που παρουσιάζουν κάποιο ενδιαφέρον από αλγοριθμικής πλευράς είναι η σχεδίαση γραμμών και κύκλων. Όλα τα υπόλοιπα μπορούν να σχεδιαστούν εύκολα με προκατασκευασμένες εικόνες που απλά αντιγράφονται πάνω στην τελική εικόνα.

2.6.1 Σχεδιασμός ευθειών

Από πλευράς γεωμετρίας οι ευθείες δεν παρουσιάζουν κανένα ιδιαίτερο ενδιαφέρον. Όπως ξέρουμε κάθε γραμμή μπορεί να αναπαρασταθεί με βάση δύο παραμέτρους m και k με την έκφραση:

$$y = mx + k$$

[23] Computer Graphics A programming approach Second Edition by Steven Harrington McGRAW-HILL 1988

Ο όρος m μας δίνει την κλίση της ευθείας ενώ ο k την απόκλιση κατά τον κατακόρυφο άξονα. Είναι προφανές ότι δεν μπορεί κανείς να ζωγραφίσει μία ευθεία σε ένα πεπερασμένο χώρο όπως είναι μία οθόνη. Αυτό το οποίο μπορεί να σχεδιαστεί είναι ένα ευθύγραμμο τμήμα το οποίο ορίζεται από δύο σημεία, της αρχής (έστω (x_1, y_1)) και του τέλους του (έστω (x_2, y_2)). Με βάση αυτά τα σημεία μπορούν να υπολογιστούν οι παράμετροι της ευθείας από τις εξής σχέσεις:

$$m = \frac{y_2 - y_1}{x_2 - x_1},\ k = y_1 - \alpha x_1$$

Αν είναι $x_{min} = \min(x_1, x_2)$ και $x_{max} = \max(x_1, x_2)$ και αντίστοιχα $y_{min} = \min(y_1, y_2)$ και $y_{max} = \max(y_1, y_2)$, είναι προφανές ότι το ευθύγραμμο αυτό τμήμα θα εκτείνεται στον άξονα x από x_{min} έως x_{max} και στον άξονα y από y_{min} έως y_{max}. Αν θέλαμε να συμπληρώσουμε το «γεωμετρικό μας υπόβαθρο» θα έπρεπε να συμπληρώσουμε ότι η απόσταση ενός τυχαίου σημείου (έστω (x_0, y_0)) από μία ευθεία είναι[24]:

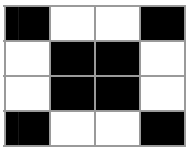
$$d = \frac{\left|(x_2 - x_1)(y_1 - y_0) - (x_1 - x_0)(y_2 - y_1)\right|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$

Αυτό που θέλουμε τώρα είναι να έχουμε την καλύτερη δυνατή αναπαράσταση – απεικόνιση ενός τέτοιου ευθύγραμμου τμήματος πάνω στην ψηφιακή εικόνα μας. Αυτή μπορεί να αναπαρασταθεί με ένα δισδιάστατο πίνακα. Στην απλή περίπτωση που θα εξετάσουμε εμείς κάθε στοιχείο μπορεί να έχει την τιμή 0 ή την τιμή 1. Όταν έχει την τιμή 0, το αντίστοιχο στοιχείο είναι λευκό στην οθόνη ενώ όταν έχει την τιμή ένα το αντίστοιχο στοιχείο εμφανίζεται μαυρισμένο στην οθόνη.

Για παράδειγμα ο πίνακας

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

σχηματίζει στην οθόνη μία αναπαράσταση ανάλογη ενός “x” (ενώ αν όπου 1 είχαμε 0 και αντίστροφα, θα είχαμε αναπαράσταση ενός “o”).



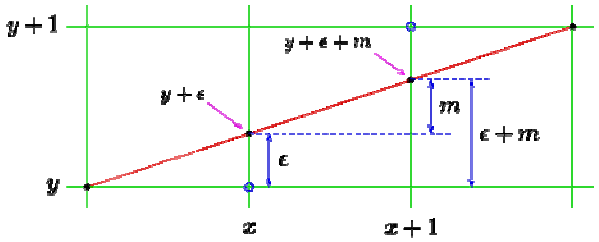
Ο πιο απλός τρόπος να ζωγραφίσουμε σε ένα τέτοιο καμβά ένα ευθύγραμμο τμήμα είναι για κάθε ακέραιο x στην περιοχή από x_{min} έως x_{max} να υπολογίζουμε από την εξίσωση της ευθείας την τιμή του y. Κάνουμε στρογγυλοποίηση του y στον πλησιέστερο ακέραιο και θέτουμε το αντίστοιχο στοιχείο του πίνακα [x, y] ίσο με μονάδα. Στην πράξη αυτός ο απλός αλγόριθμος είναι απαράδεκτα αργός. Για να υπολογιστεί για ένα σημείο το αντίστοιχο y χρειάζεται να γίνει ένας πολλαπλασιασμός και μία πρόσθεση και όλα αυτά για ένα μόνο στοιχείο. Πιο βελτιωμένες εκδόσεις προ-υπολογίζουν το βήμα dy στον y άξονα κατά την αύξηση κατά ένα του x και ελαχιστοποιούν το υπολογιστικό κόστος σε μία πρόσθεση ανά στοιχείο. Ακόμα όμως κι έτσι χρειάζονται να γίνονται επιπλέον πράξεις για τον υπολογισμό της στρογγυλοποίησης.

Την πιο φημισμένη λύση έδωσε ο καθηγητής του πανεπιστημίου Winthrop[25], Jack Bresenham[26] με τον ομώνυμο αλγόριθμο όταν ακόμα εργαζόταν στα εργαστήρια της IBM το 1962[27]. Ο αλγόριθμος

[24] http://mathworld.wolfram.com/Point-LineDistance2-Dimensional.html
[25] http://www.winthrop.edu/news/releases/archivereleases/spring2000/bresenham.htm

αυτός έχει εξαιρετικές επιδόσεις αφού μπορεί να υλοποιηθεί εξολοκλήρου με ακέραια αριθμητική, χωρίς να χρησιμοποιεί πολλαπλασιασμούς ή διαιρέσεις (παρά μόνο κάποια shifts). Ένας αντίστοιχα γρήγορος αλγόριθμος με antialising είναι ο αλγόριθμος του Xiaolin Wu[28,29] που δημοσιεύτηκε το 1991.

Η λειτουργία του έχει ως εξής:

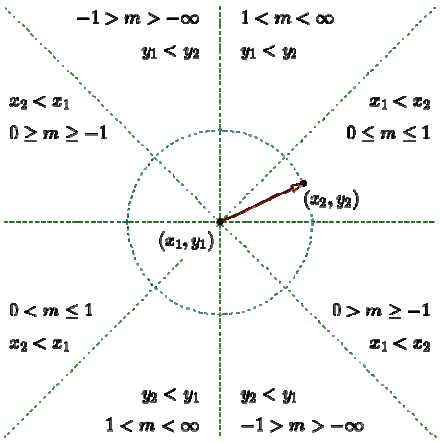


Εικόνα 39. Σχεδίαση γραμμών με τον αλγόριθμο του Bresenham[30]

Έστω ότι έχουμε μία ευθεία με κλίση m μεταξύ 0 και 1. Αν βρισκόμαστε σε ένα τυχαίο σημείο (x,y) της ευθείας τότε για το επόμενο σημείο έχουμε μόνο δύο επιλογές. Είτε θα είναι το (x+1, y) είτε το (x+1, y+1). Για κάθε σημείο ορίζουμε την ποσότητα σφάλματος $\varepsilon \in [-0.5, 0.5)$ που μας δίνει το σφάλμα ανάμεσα στον μαθηματικά υπολογισμένο σημείο και το αποτέλεσμα της στρογγυλοποίησης που αναγκαζόμαστε να κάνουμε, αφού ο πίνακας δέχεται μόνο ακέραιους δείκτες. Για το επόμενο βήμα επιλέγουμε το (x+1, y) αν ε + m < 0.5 αλλιώς το (x+1, y+1). Στην πρώτη περίπτωση το νέο σφάλμα γίνεται ε = ε + m, στην δεύτερη γίνεται ε = ε + m – 1. Σε κάθε περίπτωση για το νέο σφάλμα ισχύει $|\varepsilon| < 0.5$ πράγμα που μας δείχνει ότι η απόφασή μας ήταν η βέλτιστη (αφού αν ελέγξει κανείς την περίπτωση που παίρναμε την λάθος απόφαση θα δει ότι το σφάλμα είναι σαφώς μεγαλύτερο). Ακόμα δεν έχει φανεί να αποφεύγουμε του «ακριβούς» υπολογισμούς κινητής υποδιαστολής.

Αν πολλαπλασιάσουμε την συνθήκη μας “ε + m < 0.5” με 2Δx και επειδή m = Δy/Δx παίρνουμε: 2εΔx + 2Δy < Δx. Θέτοντας ε’ = εΔx, έχουμε 2(ε’+Δy) < Δx ως νέα συνθήκη. Αν εξετάσουμε και τις δύο περιπτώσεις ανανέωσης του σφάλματος ε’ στο επόμενο βήμα έχουμε ε = ε + m → ε’ = ε’ + Δy και ε = ε + m – 1→ ε’ = ε’ + Δy + Δx. Παρατηρούμε ότι με την εισαγωγή του ε’ όλες οι εκφράσεις είναι ακέραιες (θεωρούμε τα Δx, Δy ακέραια) και ο μόνος πολλαπλασιασμός είναι με το δύο που μπορεί να υλοποιηθεί με ένα αριστερό shift πολύ γρήγορα. Ο αλγόριθμος αυτός προφανώς είναι πάρα πολύ γρήγορος. Το μόνο του μειονέκτημα είναι ότι ζωγραφίζει γραμμές μόνο για γωνίες από 0 έως π/4 (m μεταξύ 0 και 1).

[26] http://www.udayton.edu/~cps/cps560/notes/Lines/bresnham.htm
[27] http://www.cs.wpi.edu/~matt/courses/cs563/talks/history.html
[28] http://encyclopedia.thefreedictionary.com/Bresenham%27s%20line%20algorithm
[29] http://www.ece.mcmaster.ca/~xwu/
[30] http://www.cs.helsinki.fi/group/goa/mallinnus/lines/bresenh.html



Εικόνα 40. Όλες οι περιπτώσεις που πρέπει να καλυφθούν

Όπως φαίνεται στην παραπάνω εικόνα υπάρχουν αρκετές ακόμα περιπτώσεις που πρέπει να καλυφθούν. Παρόλα αυτά αν παρατηρήσει κανείς ότι τα υπόλοιπα προβλήματα είναι στην ουσία συμμετρικά ως προς το πρώτο συμπεραίνει ότι το πρόβλημα έχει λυθεί πλήρως. Στην περίπτωση για παράδειγμα γωνιών από π/4 έως π/2 όπου το m (πιθανώς) απειρίζεται, αρκεί να εξετάσουμε το πρόβλημα μετασχηματίζοντας όπου x'→y και όπου y'→x. Αλλάζοντας δηλαδή την σειρά των αξόνων. Άμεσα μεταβαίνουμε στην περίπτωση που εξετάσαμε. Ο τελικός αλγόριθμος γίνεται λίγο μπερδεμένος[31] με τον έλεγχο αυτών των περιπτώσεων αλλά η ταχύτητα εκτέλεσής του δεν παύει να είναι ταχύτατη εξαιτίας των απλούστατων αριθμητικών πράξεων. Η (μέση) ταχύτητα μπορεί να αυξηθεί ακόμη περισσότερο αν στην αρχή γίνεται έλεγχος για τις περιπτώσεις που η γραμμή είναι οριζόντια ή κάθετη οπότε και μπορεί να χρησιμοποιηθεί τελείως τετριμμένος και ταχύτατος αλγόριθμος.

Η γενική ιδέα για έναν αλγόριθμο με antialiasing είναι ότι δεν συμπληρώνει μόνο το ένα από τα δύο pixels που έχει το μικρότερο σφάλμα αλλά και τα δύο με μία ακέραια τιμή (αντιστρόφως) ανάλογη του αντίστοιχου σφάλματος. Με αυτό τον τρόπο παράγονται γραμμές με καλύτερο αισθητικό αποτέλεσμα. Μία εξαιρετικά γρήγορη υλοποίηση μπορεί κανείς να δει στην δημοσίευση του Xiaolin Wu[32]. Εκεί λαμβάνεται υπόψη το μήκος λέξης της μηχανής και αξιοποιώντας την υπερχείλιση γίνεται δυνατή η χάραξη γραμμής με μόλις τις μισές αριθμητικές πράξεις από αυτές του αλγορίθμου του Bresenham. Λόγω όμως της λειτουργίας antialiasing ο αλγόριθμος υποχρεώνεται σε διπλάσιο αριθμό εγγραφών στη μνήμη.

2.6.2 Σχεδιασμός κύκλων

Η πιο βολική μορφή για τον κύκλο είναι η παραμετρική με παράμετρο το φ. Αν έχουμε ένα κύκλο με κέντρο (x_0, y_0) και ακτίνα R αυτός μπορεί να παρασταθεί από το εξής ζεύγος εξισώσεων:

$$\begin{aligned} x &= x_0 + R\cos\phi \\ y &= y_0 + R\sin\phi \end{aligned}, \text{ για } \phi \in [0, 2\pi)$$

Προφανώς λοιπόν ο απλούστερος αλγόριθμος για να ζωγραφίσει κανείς ένα κύκλο, είναι να υπολογίσει τα στρογγυλοποιημένα x και y για κάθε γωνία φ από 0 έως 2π, αυξανόμενη με κάποιο βήμα και να θέσει τα αντίστοιχα στοιχεία του πίνακα [x, y] ίσα με μονάδα. Για άλλη μία φορά όμως αυτός ο αλγόριθμος είναι απαράδεκτα αργός. Ο υπολογισμός των cos και sin είναι εξαιρετικά δαπανηρός. Παρόλα αυτά αν κάποιος προεπιλέξει το βήμα για το φ, μπορεί να προϋπολογίσει τις τιμές των cos και sin και να τις αποθηκεύσει σε κάποιο βοηθητικό lookup table, αυξάνοντας κατά πολύ την ταχύτητα. Επιπλέον μπορούμε να αξιοποιήσουμε την οκταπλή συμμετρία του κύκλου και

[31] http://encyclopedia.thefreedictionary.com/Bresenham%27s%20line%20algorithm%20C%20code
[32] An efficient antialiasing technique, Xiaolin Wu, July 1991, Computer Graphics, Volume 25, Number 4

να υπολογίσουμε τα Rcosφ και Rsinφ μόνο για $\phi \in [0, \pi/4)$. Τα σημεία υπολογίζονται λόγω της συμμετρίας από τις σχέσεις:

$$\begin{matrix} x = x_0 \pm R\cos\phi \\ y = y_0 \pm R\sin\phi \end{matrix} \text{ και } \begin{matrix} x = x_0 \pm R\sin\phi \\ y = y_0 \pm R\cos\phi \end{matrix}, \text{ για } \phi \in [0, \pi/4)$$

Για μία ακόμη φορά την κλασική λύση έδωσε ο Bresenham με την εκδοχή του αλγορίθμου του για τον σχεδιασμό κύκλων το 1977[33] και ο Xiaolin Wu[32] ακολούθησε με τη γρήγορη εκδοχή με antialiasing. Ο τρόπος σκέψης για τον αλγόριθμο για κύκλους είναι περίπου ο ίδιος. Εδώ στο πρώτο τεταρτημόριο παρατηρούμε ότι το y είναι μια μη αύξουσα συνάρτηση του x. Συνεπώς αν βρισκόμαστε στο σημείο (x, y) έχουμε τρεις διαθέσιμες επιλογές, να κινηθούμε στο (x+1, y), είτε στο (x+1, y-1), είτε στο (x, y-1). Και πάλι επιστρατεύεται μία (ακέραια) μεταβλητή σφάλματος που καθορίζει την επιλογή κίνησης. Για περισσότερες πληροφορίες μπορεί κανείς να ανατρέξει στις αντίστοιχες δημοσιεύσεις.

2.7 Εφαρμογές της αναγνώρισης εικόνας

Η επεξεργασία και αναγνώριση εικόνας είναι ένας κλάδος αιχμής στις ημέρες μας γιατί δίνει απάντηση σε πολλά σύγχρονα προβλήματα με μειωμένο κόστος σε σύγκριση με εναλλακτικές τεχνικές. Οι εφαρμογές είναι πολλές, εμείς όμως θα παρουσιάσουμε μόνο τις σημαντικότερες από αυτές.

Αναγνώριση χαρακτήρων - Optical Character Recognition (OCR)

Η αξιόπιστη αναγνώριση γραμμάτων είναι κάτι το οποίο πλέον το θεωρούμε δεδομένο. Η ουσιαστική λύση στο πρόβλημα της αναγνώρισης χαρακτήρων δόθηκε από την στιγμή που οι εφαρμογές έπαψαν να εστιάζουν στην αναγνώριση γραμμάτων και γενίκευσαν σε επίπεδο λέξης ή φράσης. Λεξικά με λέξεις καταταγμένες κατά σειρά συχνότητας εμφάνισης στον κοινό λόγο και γραμματικοί και συντακτικοί κανόνες δημιουργούν σύνθετες εξαρτήσεις που επιδρούν στην απόφαση για την αναγνώριση κάθε χαρακτήρα. Ακόμα πάντως και τώρα η αναγνώριση χειρογράφου κειμένου με ποσοστά επιτυχίας άνω του 90% δεν είναι εύκολη υπόθεση. Χαρακτηριστικό παράδειγμα πεδίου όπου η αναγνώριση χαρακτήρων σε πραγματικό χρόνο έχει βρει ευρεία αποδοχή είναι στην αυτόματη αναγνώριση πινακίδων αυτοκινήτου σε χώρους στάθμευσης. Θα έπρεπε να σημειώσουμε ότι υπάρχουν εφαρμογές αναγνώρισης μουσικής παρτιτούρας που χρησιμοποιούν τον μετασχηματισμό Hough για την ευθυγράμμιση με το πεντάγραμμο και την σωστή αναγνώριση των συμβόλων[34].

Έλεγχος πρόσβασης

Ο έλεγχος πρόσβασης με βάση βιομετρικά χαρακτηριστικά είναι μία από τις πιο σημαντικές εφαρμογές της αναγνώρισης εικόνας[35]. Η ερευνητική και εμπορική προσπάθεια έχει εστιαστεί κυρίως στην αναγνώριση δακτυλικών αποτυπωμάτων, αναγνώριση προσώπου και ματιών. Μεγάλα συστήματα αναγνώρισης προσώπου και δακτυλικών αποτυπωμάτων εγκαθιστούνται συνεχώς σε αστυνομικά τμήματα, μεγάλες εταιρίες και κέντρα δημοσίων μεταφορών όπως αεροδρόμια[36]. Είναι σημαντικό να σημειώσουμε ότι για την υποστήριξη των απαιτήσεων αυτών των συστημάτων χρειάζονται αντίστοιχα μεγάλες βάσεις δεδομένων βιομετρικών χαρακτηριστικών. Μεγάλη προσπάθεια γίνεται πλέον ώστε τα συστήματα αναγνώρισης βάσει βιομετρικών χαρακτηριστικών να γίνουν πιο μικρά φτηνά και ευέλικτα, ώστε να επεκτείνουν το εύρος των εφαρμογών τους σε καταναλωτικές συσκευές όπως προσωπικές ατζέντες, ασφάλεια συναλλαγών μέσω Internet και πιο ασφαλείς συναλλαγές μέσω ATMs. Ένα τυπικό σύστημα αναγνώρισης βάσει

[33] A Linear Algorithm for Incremental Digital Display of Circular Arcs, Jack Bresenham, 1977, Graphics and Image Processing
[34] http://www.cse.iitk.ac.in/~amit/courses/768/98/dash/#hough
[35] http://biometrics.cse.msu.edu/
[36] http://www.identix.com/

βιομετρικών χαρακτηριστικών αποτελείται από τρία μέρη, τον αισθητήρα αναγνώρισης, την βαθμίδα εξαγωγής χαρακτηριστικών και την βαθμίδα αναγνώρισης προτύπων όπου γίνεται και η τελική επαλήθευση για το αν τα χαρακτηριστικά που μετρώνται ταιριάζουν με κάποιου εξουσιοδοτημένου ατόμου.

Ιατρική

Οι ιατρικές εφαρμογές των συστημάτων επεξεργασίας και αναγνώρισης εικόνας έχουν λύσει πάρα πολλά και σημαντικά προβλήματα κυρίως για την διάγνωση ασθενειών. Όπως φάνηκε και από τις εφαρμογές του μετασχηματισμού Radon, η λειτουργία των αξονικών τομογράφων, οργάνων τα οποία χρησιμοποιούνται πλέον ευρέως για ένα πλήθος διαγνωστικών εξετάσεων, βασίζεται πλήρως στην ψηφιακή επεξεργασία και ανακατασκευή εικόνας. Οι μεγαλύτερες προσπάθειες αυτή τη στιγμή[37] εστιάζονται στην αυτοματοποίηση διαγνωστικών τεχνικών ώστε να μπορεί να γίνεται έλεγχος πολλών δειγμάτων σε όσο το δυνατόν λιγότερο χρόνο και με τη μεγαλύτερη δυνατή αξιοπιστία.

Αναγνώριση τοπολογίας σε αεροφωτογραφίες

Η αύξηση των εφαρμογών ηλεκτρονικής χαρτογράφησης που παρατηρείται τα τελευταία χρόνια αποτελεί ένα καινούριο απαιτητικό κλάδο εφαρμογών για αναγνώριση εικόνας. Τα συστήματα GIS (Geographical Information Systems) χρειάζονται εργαλεία που να βοηθούν τους ανθρώπους που κάνουν χαρτογράφηση με βάση την αυτόματη αναγνώριση εικόνας από αεροφωτογραφίες. Σε μεγάλες πόλεις που τα οικοδομικά τετράγωνα ξεχωρίζουν εύκολα, ακόμα και απλά συστήματα αναγνώρισης εικόνας μπορούν να δώσουν αποδοτικές λύσεις.

Βιομηχανικός αυτοματισμός

Η αναγνώριση εικόνας χρησιμοποιείται ευρέως στον βιομηχανικό αυτοματισμό. Όπως ένας άνθρωπος μπορεί οπτικά να επιβλέπει μία διαδικασία σε μία γραμμή παραγωγής, με τον ίδιο τρόπο και ένα αυτοματοποιημένο σύστημα αναγνώρισης εικόνας μπορεί να κάνει το ίδιο[38]. Οι δυνατότητες ενός συστήματος αναγνώρισης εικόνας είναι πιο περιορισμένες από αυτές ενός ανθρώπου. Αρκούν όμως για την μέτρηση αντικειμένων, την ανάγνωση barcodes, και την αναγνώριση ελαττωμάτων που φαίνονται. Οι εφαρμογές αναγνώρισης εικόνας εμφανίζουν σαφή πλεονεκτήματα όταν χρειάζεται αναγνώριση εικόνας σε υψηλές ταχύτητες, υψηλές μεγεθύνσεις, 24-ωρη λειτουργία, πιθανώς σε αντίξοες συνθήκες και σε επαναλαμβανόμενες λειτουργίες. Επιπλέον η μέτρηση από απόσταση μέσω της κάμερας εξασφαλίζει αυξημένη διάρκεια ζωής απέναντι σε εναλλακτικές τεχνικές.

[37] http://www.informatics.bangor.ac.uk/~kuncheva/activities/aprmi2004.htm
[38] http://www.wordiq.com/definition/Machine_vision

Κεφάλαιο 3. Δομή και λειτουργία των DSPs

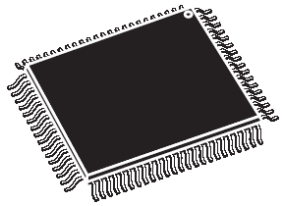
Σε αυτό το κεφάλαιο θα εξετάσουμε τις βασικές αρχές της δομής και της λειτουργίας των Ψηφιακών Επεξεργαστών Σήματος (DSPs), θα δούμε τις εφαρμογές τους στην σύγχρονη αγορά και στη συνέχεια θα εστιάσουμε στον επεξεργαστή και τα εργαλεία τα οποία θα χρησιμοποιήσουμε στην ανάπτυξη του δικού μας συστήματος δηλαδή τον επεξεργαστή της Analog Devices BF533®, το αναπτυξιακό σύστημα EZ-KIT Lite®, το λογισμικό περιβάλλον ανάπτυξης VisualDSP++® και τις δυνατότητες που μας παρέχει και την έντυπη και ηλεκτρονική τεκμηρίωση που τα συνοδεύει.

3.1 Γενικά χαρακτηριστικά των DSPs και εφαρμογές

Στο μέρος αυτό θα εξετάσουμε τι είναι τα DSPs, σε ποια χαρακτηριστικά τους οφείλουν τις αυξημένες επιδόσεις τους και τις εφαρμογές των DSPs στην σύγχρονη αγορά. Έχει γίνει προσπάθεια στο μέρος αυτό να μην γίνεται χρήση εξαιρετικά εξειδικευμένης ορολογίας ώστε να μπορεί να χρησιμεύσει ως μία εισαγωγή για κάποιον μη ειδήμονα στην τεχνολογία των DSPs.

3.1.1 Τι είναι τα DSPs;

Ένα DSP (Digital Signal Processor) είναι ένα ολοκληρωμένο κύκλωμα το οποίο μπορεί να εκτελεί μαθηματικές πράξεις γρήγορα. Φυσικά αυτός ο ορισμός αφήνει περιθώριο για σύγχυση αφού και ο επεξεργαστής ενός κοινού computer είναι σε θέση να εκτελεί μαθηματικές πράξεις γρήγορα. Στην πραγματικότητα, ενώ μερικά χρόνια πριν μπορούσες άνετα να χαράξεις τις διαχωριστικές γραμμές μεταξύ ενός κοινού επεξεργαστή και ενός DSP, με το πέρασμα των χρόνων, οι δύο αυτοί κόσμοι φαίνονται να συγκλίνουν.



Εικόνα 41. Ολοκληρωμένο κύκλωμα

Η βασική διαφορά είναι στην οπτική με την οποία αντιμετωπίζει κανείς ένα υπολογιστικό σύστημα. Τα DSPs χρησιμοποιούνται για να εκτελούν με πάρα πολύ υψηλή ταχύτητα, επαναλαμβανόμενα κομμάτια κώδικα με πολλές αριθμητικές πράξεις. Η έμφαση δίνεται σαφώς στην υψηλή ταχύτητα.

Αντίθετα στους γενικής χρήσης επεξεργαστές, προτεραιότητα δίνεται στο να μπορούν να εκτελούν πολλά γενικά κομμάτια κώδικα και έμφαση δίνεται στο να μπορούν να καλύψουν ένα μεγάλο εύρος εφαρμογών όπως και το να μπορούν να προγραμματιστούν εύκολα και με μικρό κόστος. Εκτός από ειδικές περιπτώσεις, η ταχύτητα είναι μεν ένας σημαντικός παράγοντας αλλά όχι ο σημαντικότερος.

Για να δώσουμε ένα παράδειγμα, όταν έχουμε να αναπτύξουμε μία εφαρμογή π.χ. αναγνώρισης φωνής. Η λογική πορεία κατά την ανάπτυξη είναι να χρησιμοποιήσουμε στην αρχή κάποιον γενικής χρήσης επεξεργαστή, πιθανότατα του computer μας. Με αυτό τον τρόπο μπορούμε να πειραματιστούμε και να κάνουμε γρήγορα ένα πρότυπο ώστε να αποδείξουμε ότι η ιδέα μας είναι υλοποιήσιμη και να οριστικοποιηθεί ο αλγόριθμος που θα χρησιμοποιήσουμε. Ο βασικός μας γνώμονας σε αυτή την φάση είναι η άνεση και ευκολία προγραμματισμού. Βασικά πλεονεκτήματα της εργασίας στον γενικής χρήσεως επεξεργαστή, είναι ότι μπορούμε να προγραμματίσουμε γρήγορα, χρησιμοποιώντας κάποια εργαλεία υψηλού επιπέδου π.χ. το MATLAB™ και να γλυτώσουμε πολύτιμο χρόνο χρησιμοποιώντας το έτοιμο γενικής χρήσης λογισμικό. Δεν χρειάζεται, για παράδειγμα, να ξαναγράψουμε drivers για την κάρτα ήχου.

Όταν πλέον έχουμε αναπτύξει το πρότυπό μας στον υπολογιστή, γίνονται φανερές κάποιες αδυναμίες του συστήματος.

1. Ο αλγόριθμός μας τρέχει αργά στο computer. Δεν αναγνωρίζει τη φωνή σε πραγματικό χρόνο (real-time) αλλά με καθυστέρηση.
2. Όταν τρέχει ο αλγόριθμός μας καταναλώνει μεγάλο μέρος της υπολογιστικής ισχύος του επεξεργαστή. Δεν μπορούμε π.χ. να βλέπουμε ταυτόχρονα σελίδες στο internet.

3. Αν το σύστημά μας πρόκειται να ενσωματωθεί σε κάποια κρίσιμη εφαρμογή, π.χ. πλοήγηση αεροπλάνων, δεν θα θέλαμε σε καμία περίπτωση να «κολλήσει» για οποιονδήποτε λόγο.
4. Είναι εξαιρετικά ασύμφορη και άβολη η χρήση του υπολογιστή. Αν θελήσουμε να εκμεταλλευτούμε εμπορικά 10.000 συσκευές που να κάνουν αναγνώριση φωνής δεν θα θέλουμε να έχουν το κόστος ή το μέγεθος ενός υπολογιστή.
5. Ένας υπολογιστής καταναλώνει πάρα πολύ ενέργεια και συνεπώς το σύστημα δεν θα μπορεί να είναι φορητό δηλαδή να τροφοδοτείται από μπαταρία.

Όλα τα παραπάνω μας οδηγούν άμεσα στην χρήση ενός DSP. Ο αλγόριθμος θα μπορεί να εκτελείται ταχύτατα και αξιόπιστα χωρίς να επιβαρύνει κάποιο γενικής χρήσης σύστημα. Το κόστος ανά μονάδα μπορεί να είναι της τάξης των 5€ για μεγάλες ποσότητες (τιμές 2004) και η κατανάλωση ισχύος ελάχιστη σε συνδυασμό με ελάχιστο βάρος και όγκο. Φυσικά ο προγραμματισμός μίας τέτοιας μονάδας είναι σαφώς πιο επίπονος και ακριβός απ' ότι προηγουμένως αλλά το κόστος ανάπτυξης είναι ένα κόστος που πληρώνεται μόνο στην αρχή της ζωής ενός προϊόντος (non-recurring cost) και συνεπώς συμφέρει, επειδή μειώνει δραματικά το κόστος ανά μονάδα όπως και την ανταγωνιστικότητα του προϊόντος.

Ένας ακόμα απλός τρόπος να διαχωρίσει τα DSPs από τους κοινούς επεξεργαστές είναι να εξετάσει την φιλοσοφία προγραμματισμού τους. Τα προγράμματα των συνήθων επεξεργαστών βασίζονται πάρα πολύ στις καταστάσεις και τις μεταβάσεις μεταξύ αυτών π.χ. ένα PC είναι σε κατάσταση αναμονής μέχρι κάποιος να πατήσει ένα πλήκτρο, όπου μπαίνει στην κατάσταση επεξεργασίας αυτού του γεγονότος και μετά επιστρέφει στην αρχική κατάσταση αναμονής. Τα προγράμματα των DSPs βασίζονται στην ροή δεδομένων. Τα σήματα εισόδου περνάνε από τις μαθηματικές συναρτήσεις και στην συνέχεια εξάγονται ως σήματα εξόδου ή αξιολογούνται με κάποιο τρόπο από το ίδιο το DSP. Αυτή η διαδικασία επαναλαμβάνεται συνέχεια. Για παράδειγμα, το DSP του γραφικού ισοσταθμιστή ενός στερεοφωνικού, λαμβάνει συνεχώς το σήμα του ήχου από κάποια πηγή, το επεξεργάζεται εφαρμόζοντας μαθηματικές συναρτήσεις στο τέλος το εξάγει στις βαθμίδες που το στέλνουν στα ηχεία.

3.1.2 Χαρακτηριστικά αρχιτεκτονικής υψηλής ταχύτητας

Τα κυριότερα χαρακτηριστικά που κάνουν την ταχύτητα εκτέλεσης αριθμητικών αλγορίθμων σε DSP τόσο υψηλή είναι τα εξής [39,40]:

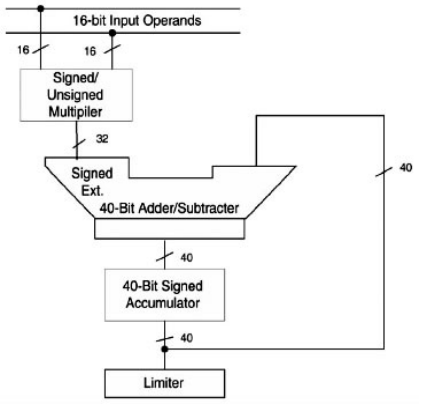
1. Εκτέλεση πολλαπλασιασμού και πρόσθεσης σε έναν κύκλο
2. Αυτόματη τροποποίηση δεικτών στην μνήμη
3. Ειδικοί τρόποι προσπέλασης της μνήμης
4. Αποδοτικοί επαναληπτικοί βρόχοι υλοποιημένοι στο hardware
5. Αποδοτική κωδικοποίηση εντολών
6. Λειτουργία pipelining
7. Ειδικές συναρτήσεις υλοποιημένες στο hardware
8. Κανάλια DMA
9. Πολλαπλασιασμός ταχύτητας πυρήνα με PLL

Παρακάτω θα εξετάσουμε κάθε ένα από αυτά αναλυτικότερα.

3.1.2.1 Εκτέλεση πολλαπλασιασμού και πρόσθεσης σε έναν κύκλο

[39] Special Purpose Digital Processors (DSP) (F. Mayer-Lindenberg, TUHH) dsp.pdf
[40] (http://www.bdti.com) (Ole Wolf, wolf@bdti.com)

Το βασικό δομικό στοιχείο κάθε υπολογιστικής μονάδας ενός DSP είναι η μονάδα πολλαπλασιασμού – πρόσθεσης (MAC). Η μονάδα αυτή είναι απαραίτητη γιατί οι δύο εργασίες που καλούνται να εκτελέσουν συχνότερα τα DSPs, ο υπολογισμός γρήγορων μετασχηματισμών Fourier (FFT – inverse FFT) και τα ψηφιακά φίλτρα (FIR, IIR), εκτελούνται ουσιαστικά ως μια σειρά πολλαπλασιασμών και προσθέσεων.



Εικόνα 42. Μονάδα MAC

Μία μονάδα MAC μπορεί να εκτελέσει ένα πλήρη πολλαπλασιασμό δύο αριθμών και την πρόσθεσή τους σε ένα μόνο κύκλο. Το block διάγραμμα μίας μονάδας MAC φαίνεται στα δεξιά. Μπορούμε να δούμε τον 16x16 bit πολλαπλασιαστή ο οποίος ως έξοδό του έχει ένα νούμερο 32bit όπως ήταν αναμενόμενο. Το νούμερο αυτό δρομολογείται μετά στην ALU όπου γίνεται η πρόσθεση ή αφαίρεση στην συνέχεια το αποτέλεσμα οδηγείται σε μία μεταβλητή άθροισης (accumulator). Στο επόμενο βήμα θα προστεθεί το επόμενο αποτέλεσμα του πολλαπλασιασμού στον accumulator. Ο accumulator έχει μέγεθος 40bit. Συνεπώς μετά από ($2^{40-32} = 2^8$) 256 προσθέσεις 32-bitων αριθμών είναι δυνατόν να ξεχειλίσει. Γι' αυτήν την περίπτωση προβλέπονται διάφορες τεχνικές χειρισμού της υπερχείλισης όπως η τεχνική του κορεσμού (saturation) όπου δίνεται η μέγιστη τιμή που μπορεί να πάρει ο accumulator.

Παρατηρούμε ότι με μία μονάδα MAC είναι πολύ εύκολος ο υπολογισμός εκφράσεων του τύπου $a = \sum_{n=0}^{N-1} x_n \cdot y_n$ οι οποίες όπως είπαμε είναι η βάση των περισσοτέρων χρήσιμων συναρτήσεων επεξεργασίας σήματος.

Θα έπρεπε να σημειώσουμε ότι τα περισσότερα DSPs σήμερα έχουν πάνω από 2 μονάδες MAC δηλαδή μπορούν να πραγματοποιήσουν περισσότερους από 2 πολλαπλασιασμούς και 2 προσθέσεις ανά κύκλο. Αν υπολογίσει κανείς ότι οι ταχύτητες του πυρήνα ενός DSP μπορεί να είναι και πάνω από 600M κύκλοι / δευτερόλεπτο καταλαβαίνει ότι ένας τέτοιος επεξεργαστής μπορεί να υπολογίσει 1.200.000.000 πολλαπλασιασμούς και προσθέσεις ανά δευτερόλεπτο. Αυτή την στιγμή υπάρχουν διαθέσιμα DSPs με πάνω από 4 φορές περισσότερη επεξεργαστική ισχύ από αυτή που περιγράφηκε.

IPs, MACs και FLOPs

Εδώ πρέπει να κάνουμε μία παρένθεση για να ορίσουμε κάποιες μονάδες που χρησιμοποιούνται για να εκτιμήσουν την επεξεργαστική ισχύ των επεξεργαστών γενικά αλλά και των DSPs ειδικότερα. Οι κυριότερες μονάδες που χρησιμοποιούνται είναι οι εξής:

Μονάδα	Περιγραφή	Συνήθως συναντάμε
IPs	Ισχύει για όλους τους επεξεργαστές. Είναι ο αριθμός των εντολών ανά δευτερόλεπτο (instructions per second) που μπορεί να εκτελέσει ο επεξεργαστής.	MIPs
MACs	Ισχύει για τα DSPs. Είναι ο αριθμός των πολλαπλασιασμών – προσθέσεων που μπορεί να εκτελεστεί ανά δευτερόλεπτο. (Multiplications – Accumulations per second).	MMACs, GMACs
FLOPs	Ισχύει για τα DSPs που μπορούν να κάνουν πράξεις κινητής υποδιαστολής (περισσότερα γι' αυτές παρακάτω). Είναι ο αριθμός των πράξεων κινητής υποδιαστολής ανά δευτερόλεπτο. (Floating-point Operations Per second).	MFLOPs, GFLOPs

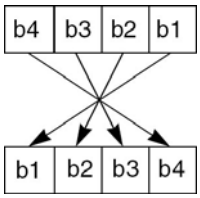
Θα έπρεπε να σημειώσουμε ότι τα MIPs ενός επεξεργαστή είναι συνήθως λιγότερα ή ίσα από τα MACs ή FLOPs του. Αυτό γιατί αν ο επεξεργαστής έχει περισσότερες από μία βαθμίδες MAC, τότε μπορεί σε ένα κύκλο να κάνει περισσότερες από μία προσθέσεις – πολλαπλασιασμούς. Όλες όμως οι πράξεις που εκτελούνται σε ένα κύκλο λαμβάνονται υπ' όψιν ως μία εντολή (instruction). Επίσης η ταχύτητα εκτέλεσης όλων των εντολών που δεν είναι αριθμητικές, είναι η ίδια ανεξάρτητα από το πόσες MAC βαθμίδες έχει το DSP. Συνεπώς ένα DSP με 4 MAC βαθμίδες το οποίο τρέχει

με ταχύτητα πυρήνα 500MHz, έχει επεξεργαστική ισχύ 500MIPs ή 2GMACs αφού σε κάθε κύκλο μπορεί να κάνει τέσσερις πολλαπλασιασμούς – προσθέσεις.

Το συμπέρασμα που βγαίνει από την παραπάνω ανάλυση είναι ότι το πλεονέκτημα της επεξεργαστικής ισχύος των DSPs εμφανίζεται κυρίως σε ορισμένες αριθμητικές πράξεις όπου ο πυρήνας μπορεί να δουλεύει πολλές βαθμίδες παράλληλα και όχι γενικότερα σε κάθε τύπου επεξεργασία.

3.1.2.2 Αυτόματη τροποποίηση δεικτών στην μνήμη

Αν παρατηρήσει κανείς την έκφραση $a = \sum_{n=0}^{N-1} x_n \cdot y_n$, μπορεί να δει τους δείκτες n κάθε στοιχείου που πολλαπλασιάζεται. Βλέπουμε ότι για την γρήγορη υλοποίηση του παραπάνω υπολογισμού χρειάζεται να παίρνουμε στοιχεία τα οποία είναι συνεχόμενα τοποθετημένα στην μνήμη (n = 0, 1, 2…). Η αρχιτεκτονική λοιπόν των DSPs προβλέπει εντολές που εκτός από την λειτουργία MAC που περιγράφηκε παραπάνω, κάνουν και αυτόματη τροποποίηση των δεικτών των στοιχείων x_n και y_n στον ίδιο κύκλο με τον υπολογισμό. Πιο συγκεκριμένα προβλέπονται διάφοροι τρόποι τροποποίησης των δεικτών όπως αυτόματη αύξηση ή αυτόματη μείωση πριν ή μετά την πράξη (pre- and post-modification), κυκλική διευθυνσιοδότηση, που είναι πολύ χρήσιμη για να γεμίζει αποθηκευτικούς χώρους (buffers – FIFOs) και διευθυνσιοδότηση ανάστροφου bit (bit-reversed addressing) που δημιουργεί δείκτες όπως ακριβώς όπως απαιτούνται για τους μετασχηματισμούς FFT και inverse FFT.



Εικόνα 43. Διευθυνσιοδότηση ανάστροφου bit

Προφανώς πρέπει να υπάρχει μία βαθμίδα δημιουργίας διευθύνσεων (Data Address Generators / Address Generator Units) για κάθε μονάδα MAC η οποία θα ενημερώνει αυτόματα τους δείκτες και κάθε βαθμίδα πρέπει να διαχειρίζεται δύο δείκτες (συχνά αναφέρονται και ως δείκτης X και Y.

3.1.2.3 Ειδικοί τρόποι προσπέλασης της μνήμης

Όπως έγινε φανερό από τα παραπάνω τα DSPs χειρίζονται πολλές μεταβλητές ταυτόχρονα. Για παράδειγμα σε μία απλή MAC εντολή, πρέπει σε ένα κύκλο να διαβάζονται δύο θέσεις μνήμης και (πιθανώς) να γράφεται το αποτέλεσμα σε μία τρίτη θέση. Προσθέτοντας περισσότερες MAC βαθμίδες, πολλαπλασιάζεται και η ανάγκη για περισσότερες αναγνώσεις και εγγραφές από την μνήμη. Οι παραδοσιακοί επεξεργαστές δεν μπορούν να κάνουν πάνω από μία εγγραφή – ανάγνωση της μνήμης ανά κύκλο δηλαδή δεν θα μπορούσαν με κανένα τρόπο να αξιοποιήσουν την παράλληλη επεξεργασία των MAC βαθμίδων. Τα DSPs προσπελαύνουν με ειδικό τρόπο την μνήμη (ιδιαίτερα την cache) και χωρίζοντάς την σε πολλές ανεξάρτητες ομάδες (banks) καταφέρνουν να κάνουν πολλές αναγνώσεις και εγγραφές σε ένα μόνο κύκλο μεγιστοποιώντας την απόδοση του DSP.

3.1.2.4 Αποδοτικοί επαναληπτικοί βρόχοι υλοποιημένοι στο hardware

Το τελευταίο χαρακτηριστικό που μας λείπει για τον υπολογισμό των εκφράσεων του τύπου $a = \sum_{n=0}^{N-1} x_n \cdot y_n$ είναι το στοιχείο της επανάληψης. Προφανώς πρέπει να γίνουν N επαναλήψεις της πράξης του πολλαπλασιασμού και της πρόσθεσης όπως και της αύξησης των δεικτών για να υπολογιστεί η παραπάνω έκφραση. Οι παραδοσιακοί επεξεργαστές θα χρειάζονταν τουλάχιστον 3 εντολές για την υλοποίηση μίας επανάληψης, μία που θα αυξάνει τον μετρητή, μία που θα τον

συγκρίνει με τον αριθμό των επαναλήψεων και μία που θα πηγαίνει στην αρχή του βρόχου επανάληψης αν ο μετρητής είναι μικρότερος του αριθμού επαναλήψεων. Κάθε μία από αυτές τις εντολές κοστίζει από έναν κύκλο. Γίνεται προφανές ότι μία τέτοια σπατάλη θα ήταν απαράδεκτη για αρχιτεκτονικές υψηλής ταχύτητας γιατί ουσιαστικά αναιρεί όλο τον «κόπο» που κάναμε με τις μονάδες MAC για να ολοκληρώνουμε πολλές πράξεις σε ένα κύκλο.

Για να λυθεί αυτό το πρόβλημα τα σύγχρονα DSP έχουν δομές επαναληπτικών βρόχων υλοποιημένες πάνω στο chip (hardware). Αυτό σημαίνει ότι βάζεις σε κάποια ειδική μεταβλητή την διεύθυνση στην οποία αρχίζει η επανάληψη, σε μία άλλη την διεύθυνση στην οποία τελειώνει και σε μία τρίτη τον αριθμό επαναλήψεων και το DSP αναλαμβάνει να εκτελέσει σωστά τις επαναλήψεις χωρίς ούτε μία επιπλέον γραμμή κώδικα. Το σημαντικότερο όμως είναι ότι οι επαναλήψεις σε αυτή την περίπτωση γίνονται χωρίς να καταναλώνει ο επεξεργαστής πολύτιμους κύκλους για αυτές (zero-overhead). Με αυτό τον τρόπο η παραπάνω μαθηματική έκφραση μπορεί να υπολογιστεί με μόνο κόστος, τους κύκλους που χρειάζονται για τους πολλαπλασιασμούς – προσθέσεις από τις βαθμίδες MAC.

3.1.2.5 Αποδοτική κωδικοποίηση εντολών

Μία εντολή αποθηκεύεται στην μνήμη (προγράμματος) του επεξεργαστή και έχει συνήθως κάποιο τυπικό μέγεθος π.χ. 16 ή 32 bit. Παραδοσιακά οι επεξεργαστές κωδικοποιούν μία εντολή στα 32 bit της εντολής π.χ. μία εντολή είναι η A = B + 4 και μπορεί να κωδικοποιείται σε 32-bit μορφή ως:

10111110	00010000 00100111	00000100
Κωδικος εντολης	16-bit διευθυνση της μεταβλητης b	Σταθερα προσθεσης (4)

Παρατηρείστε ότι η εντολή πρέπει να έχει μέσα στα 32 της bit και το νούμερο – σταθερά της πρόσθεσης το οποίο από μόνο του μπορεί να είναι αρκετά bit. Γίνεται φανερό ότι η κωδικοποίηση των εντολών είναι μία δύσκολη υπόθεση γιατί πρέπει πάρα πολύ πληροφορία να «χωρέσει» μέσα σε ένα μικρό αριθμό bits.

Για να μπορούν τα DSPs να εκμεταλλεύονται τις πολλές βαθμίδες MAC και τις δυνατότητες παράλληλης επεξεργασίας πρέπει να είναι δυνατή η σύνθετη και αποδοτική κωδικοποίηση πολλών εργασιών στα 32-bit μίας εντολής, γιατί το φόρτωμα μίας μόνο εντολής (32-bit) είναι δυνατόν να ολοκληρωθεί σε ένα κύκλο του επεξεργαστή. Για παράδειγμα κάποια DSPs μπορούν να κωδικοποιήσουν δύο προσθέσεις, δύο πολλαπλασιασμούς, τέσσερις αναγνώσεις 16-bit μεταβλητών και την αύξηση δύο δεικτών σε μία μόνο εντολή 32-bit.

3.1.2.6 Λειτουργία pipelining

Αυτή είναι μία λειτουργία η οποία δεν είναι ορατή άμεσα στον προγραμματιστή και δρα ως υποδομή για όλα τα παραπάνω. Είναι σημαντικό να γνωρίζει κανείς κάποια πράγματα γι' αυτήν γιατί σε ορισμένες περιπτώσεις κάποιοι λάθος χειρισμοί μπορεί να οδηγήσουν σε εξαιρετικά πιο αργή εκτέλεση του κώδικα [41,42].

Παρ' όλα όσα είπαμε απλοποιημένα μέχρι τώρα, ο επεξεργαστής είναι αδύνατον να εκτελέσει μία εντολή (ολόκληρη) σε ένα μόνο κύκλο. Για να το έκανε αυτό θα έπρεπε π.χ. να έγραφε το αποτέλεσμα μίας πράξης την ίδια στιγμή που θα προσπαθούσε να διαβάσει και να αναγνωρίσει ποια είναι αυτή και τι δεδομένα χρειάζεται! Στην πραγματικότητα οι περισσότερες σύνθετες εντολές δεν μπορούν να εκτελεστούν σε έναν κύκλο του επεξεργαστή. Για να ξεπεραστεί αυτό το πρόβλημα χρησιμοποιείται το pipelining.

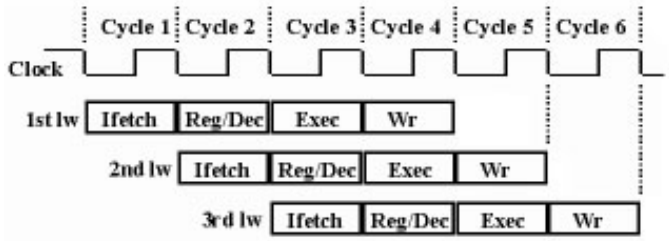
Το pipelining βασίζεται στην εξής απλή αρχή. Μία συνήθης εντολή μπορεί να χωριστεί σε μικρότερα βήματα. Για παράδειγμα μία πρόσθεση, π.χ. Γ = Α + Β, χωρίζεται στα εξής βήματα:

[41] http://cs-alb-pc3.massey.ac.nz/notes/59304/l12.html
[42] http://www.webopedia.com/TERM/P/pipelining.html

Α) Fetch	Διάβασμα εντολής	Διαβάζει την εντολή
Β) Decode	Ανάκτηση δεδομένων	Διάβασμα από την μνήμη των Α και Β
Γ) Execute	Εκτέλεση υπολογισμού	Εκτέλεση της πρόσθεσης
Δ) Write	Αποθήκευση	Αποθήκευση του αποτελέσματος στην μεταβλητή Γ

Με την λειτουργία pipelining κάθε ένα από τα βήματα εκτελείται από μία ξεχωριστή βαθμίδα σε ένα κύκλο ρολογιού (αυτό είναι δυνατόν να γίνει) αλλά όλες οι βαθμίδες είναι απασχολημένες με ένα κομμάτι εκτέλεσης μίας διαφορετικής εντολής.

Για παράδειγμα, αν θέλαμε να εκτελέσουμε τρεις πράξεις στην σειρά, στον πρώτο κύκλο η βαθμίδα που κάνει την αναγνώριση θα αναγνώριζε την πρώτη πράξη. Στον δεύτερο κύκλο η βαθμίδα που κάνει την αναγνώριση θα αναγνωρίζει την δεύτερη πράξη και η βαθμίδα που κάνει την ανάκτηση δεδομένων, θα έφερνε τα δεδομένα για την πρώτη πράξη. Στο τρίτο βήμα η βαθμίδα αναγνώρισης θα αναγνώριζε την τρίτη πράξη, η βαθμίδα ανάκτησης δεδομένων θα έφερνε τα δεδομένα για την δεύτερη πράξη και η βαθμίδα εκτέλεσης πράξεων θα εκτελούσε την πρώτη πράξη κ.ο.κ. Το παράδειγμα αυτό δίνεται σχηματικά με το παρακάτω σχήμα.



Εικόνα 44. Η λειτουργία pipelining

Όπως μπορούμε να δούμε μετά τον τρίτο κύκλο φαίνεται να ολοκληρώνεται μία εντολή σε κάθε κύκλο. Η πρώτη στον κύκλο 4, η δεύτερη στον κύκλο 5 και η τρίτη στον κύκλο 6. Αντίστοιχα αν είχαμε 100 εντολές, μετά τους τρεις πρώτους κύκλους θα φαινόταν σαν να εκτελείται μία εντολή ανά κύκλο δηλαδή οι 100 εντολές θα εκτελούνταν συνολικά σε 103 κύκλους. Αν αναλογιστεί κανείς ότι η κάθε εντολή χρειάζεται τέσσερις κύκλους για να εκτελεστεί, εύκολα μπορεί να δει ότι χωρίς την λειτουργία pipelining, οι τρεις εντολές του παραδείγματος θα ήθελαν για να εκτελεστούν 3x4=12 κύκλους ενώ οι 100 εντολές θα χρειάζονταν 100x4 = 400 κύκλους. Χρησιμοποιώντας λοιπόν την λειτουργία pipelining έχουμε βελτίωση της ταχύτητας στην πρώτη περίπτωση κατά 200% στην δεύτερη περίπτωση περίπου κατά 400% (δηλαδή όσο 100% για κάθε βαθμίδα)! Η λειτουργία αυτή μας δίνει την ψευδαίσθηση ότι μία εντολή εκτελείται σε ένα κύκλο ενώ στην πραγματικότητα εκτελείται σε πολύ περισσότερες.

Που βρίσκεται το πρόβλημα; Το πρόβλημα βρίσκεται στις περιπτώσεις που εντολές χρησιμοποιούν δεδομένα τα οποία εξαρτώνται από αμέσως προηγούμενες εντολές. Για παράδειγμα αν οι τρεις εντολές του παραπάνω παραδείγματος είναι οι Α = 1 + 1, Β = Α + 1 και Γ = Β + 1. Η «έξυπνη» λογική του επεξεργαστή δεν θα αφήσει φυσικά να γίνει αριθμητικό λάθος. Απλά σε αυτή την περίπτωση το pipelining ΔΕΝ θα λειτουργήσει. Η εντολή Β θα κολλήσει στην φάση της αποκωδικοποίησης και θα περιμένει να ολοκληρωθεί και η φάση αποθήκευσης της πρώτης εντολής. Τότε θα συνεχίσει να εκτελείται η δεύτερη εντολή στην φάση execute, με τα σωστά δεδομένα ενώ η τρίτη εντολή θα περιμένει κι αυτή να ολοκληρωθεί η δεύτερη ώστε να έχει τα σωστά δεδομένα. Η κατάσταση αυτή ονομάζεται stall και μπορεί να καθυστερήσει πολύ την ταχύτητα εκτέλεσης. Για παράδειγμα το παρακάτω πρόγραμμα θα τρέχει πολύ πιο αργά απ' ότι θα περιμέναμε:

```
for (i=0; i < 255; i++) {
      a[i] += b[i] * c[i];
      b[i] += a[i] * c[i];
}
```

Υπάρχει και άλλη μία περίπτωση που οδηγεί στην ακύρωση της λειτουργίας pipelining. Αυτή είναι όταν έχουμε κάποια συνθήκη με έλεγχο. Για παράδειγμα: «Αν το x είναι 3 τότε κάνε το Α, αλλιώς κάνε το Β». Πριν να ολοκληρωθεί η εκτέλεση της εντολής «αν» δεν είναι δυνατόν να εκτελεστεί

ούτε το Α ούτε το Β γιατί δεν ξέρουμε ποιο από τα δύο πρέπει να εκτελεστεί. Για να αντιμετωπιστεί αυτό το πρόβλημα πολλοί επεξεργαστές χρησιμοποιούν μία τεχνική πρόβλεψης διακλάδωσης (branch prediction)[43]. Αυτή τις περισσότερες φορές δεν είναι τίποτα περισσότερο από το να αρχίσουν να εκτελούν το Α χωρίς να νοιαστούν για το αν αυτό είναι το σωστό.

Πριν να φτάσει η εντολή στην φάση αποθήκευσης (την τελευταία φάση), τίποτα δεν αλλάζει στην μνήμη. Αν δεν εκτελεστεί δηλαδή η τελευταία φάση της εντολής, η εντολή είναι σαν να μην εκτελέστηκε.

Όταν κάνουμε λοιπόν branch prediction όπως περιγράφηκε παραπάνω, αρχίζει να εκτελείται το Α και όταν πλέον έχει υπολογιστεί η εντολή «αν», εάν μεν το Α είναι το σωστό, ολοκληρώνεται και η φάση της αποθήκευσης και έχουμε αδιάλειπτη λειτουργία του pipelining, αν πάλι κάναμε λάθος και έπρεπε να εκτελεστεί το Β, το αποτέλεσμα της Α δεν αποθηκεύεται και ξεκινάει εξ' αρχής η εκτέλεση της Β. Με αυτόν τον τρόπο έχουμε 50% πιθανότητα να κάνουμε σωστή επιλογή και η εντολή «αν» να μην επηρεάσει την ταχύτητα εκτέλεσης. Με διάφορες τεχνικές αυτή η πιθανότητα μπορεί να βελτιωθεί ακόμα περισσότερο, π.χ. ο επεξεργαστής να επιλέγει ως εκδοχή μίας διακλάδωσης, αυτή που εκτελέστηκε την προηγούμενη φορά.

Στην περίπτωση του «αν» η πιθανότητα των Α και του Β μπορεί να φαίνεται 50%. Στην περίπτωση όμως μίας διακλάδωσης του τύπου «όσο» π.χ. «όσο το x είναι μικρότερο του 3 κάνε το Α και μετά συνέχισε στο Β», το Α φαίνεται να έχει πολύ μεγαλύτερη πιθανότητα εκτέλεσης. Αν όμως εμείς κάνουμε (στο συγκεκριμένο πλαίσιο) εσφαλμένη χρήση της εντολής «όσο» και στην ουσία η συνθήκη του «όσο», σπάνια επαληθεύεται, θα έχουμε πάρα πολύ συχνά λάθος επιλογή. Χαρακτηριστικό παράδειγμα είναι η αναγωγή μίας τριγωνομετρικής μεταβλητής στο πρώτο τεταρτημόριο:

```
while (phi > (PI/2)) { phi -= (PI/2); }
while (phi < 0) { phi += (PI/2); }
```

Αν για το σήμα μας η μεταβλητή phi τις περισσότερες φορές θα είναι στο πρώτο τεταρτημόριο και πολύ σπάνια θα χρειάζεται να εκτελεστεί η αύξηση ή η μείωση του phi, τότε η παραπάνω επιλογή είναι λάθος γιατί το pipelining θα «υποθέτει» ότι συνήθως θα εκτελούνται αυτές οι εντολές και θα έχουμε σημαντική καθυστέρηση.

Μία άλλη περίπτωση που μπορεί να αναστείλει τη λειτουργία pipelining είναι η χρήση εντολών `goto` ειδικά όταν αναφέρονται σε απομακρυσμένα κομμάτια μνήμης π.χ. στην διεύθυνση 3300-0000 υπάρχει εντολή `goto` προς τη διεύθυνση F300-0000. Γενικά πρέπει κανείς να ανατρέξει στο τεχνικό εγχειρίδιο του συγκεκριμένου DSP και να μελετήσει την λειτουργία pipelining για να δει ποιες είναι καλές τακτικές προγραμματισμού και ποιες όχι.

Όπως μπορούμε λοιπόν να δούμε, η τεχνική pipelining κάνει δυνατή την εκτέλεση των εντολών με εξαιρετικά μεγάλες ταχύτητες αλλά εμπεριέχει και τον κίνδυνο του stall για όσους γράφουν προγράμματα απρόσεκτα.

3.1.2.7 Ειδικές συναρτήσεις υλοποιημένες στο hardware

Πολλές φορές χρειάζεται ο υπολογισμός κάποιων ειδικών συναρτήσεων. Ένα πολύ συχνά εμφανιζόμενο παράδειγμα, είναι να απαιτείται ο υπολογισμός της ευκλείδειας απόστασης (μέτρο) ενός διανύσματος ($\sqrt{x^2 + y^2}$). Πραγματικά, αν έχεις κάνει έναν μετασχηματισμό FFT του σήματός σου, τότε στην έξοδο θα έχεις ένα μιγαδικό σήμα στον χώρο των συχνοτήτων. Το μέτρο του μιγαδικού αυτού σήματος μπορεί να δώσει πάρα πολύ χρήσιμες πληροφορίες όπως την ισχύ του σήματος σε μία συγκεκριμένη συχνότητα.

Όταν ο αλγόριθμος χρησιμοποιεί μία τέτοια εξεζητημένη, αλλά συχνά εμφανιζόμενη συνάρτηση, υπάρχουν δύο επιλογές. Ή θα υλοποιηθεί προγραμματιστικά ή θα γίνει επιλογή ενός DSP που την

[43] http://www.x86.org/articles/branch/branchprediction.htm

έχει υλοποιημένη στο hardware. Η δεύτερη επιλογή έχει το σαφές πλεονέκτημα ότι μπορεί να ολοκληρωθεί όλος ο υπολογισμός σε ένα μόνο κύκλο, ενώ πιθανώς να έχει το μειονέκτημα να είναι το συγκεκριμένο DSP λίγο πιο ακριβό.

Αν μία «ειδική» συνάρτηση χρησιμοποιείται συχνά από έναν αλγόριθμο τότε μπορεί να συμφέρει να επιλεγεί ένα DSP που να την έχει υλοποιημένη σε hardware γιατί με αυτό τον τρόπο θα αυξηθεί κατά πολύ η ταχύτητα εκτέλεσης και κατά συνέπεια μπορεί να μειωθεί η ταχύτητα χρονισμού του DSP με ανάλογη μείωση στην κατανάλωση ισχύος και συνεπώς μεγαλύτερη αυτονομία. Επίσης η μείωση της απαιτούμενης ταχύτητας χρονισμού μπορεί να οδηγήσει σε επιλογή ενός φτηνότερου DSP.

3.1.2.8 Κανάλια DMA

Για να αντιγραφεί ένας πίνακας ένα άλλο σημείο πρέπει να εκτελεστούν κάποιες εντολές. Συγκεκριμένα, αν N το μέγεθος του πίνακα, πρέπει να γίνουν N πράξεις ανάθεσης που θα αντιγράφουν τα στοιχεία μίας θέσης μνήμης σε μία άλλη και οι αντίστοιχες αυξήσεις των δεικτών.

Για να ληφθούν δεδομένα από κάποιο περιφερειακό (π.χ. A/D ή D/A) τα πράγματα είναι ακόμη πιο δύσκολα. Συγκεκριμένα πρέπει το πρόγραμμα να περιμένει να έρθει κάποιο σημάδι – interrupt από το περιφερειακό που να πληροφορεί ότι υπάρχουν νέα δεδομένα και στην συνέχεια να αρχίσει η ανάγνωση από αυτό με κάποιον ειδικό τρόπο.

Καμία από τις δύο λειτουργίες δεν απαιτεί κάποια ιδιαίτερη εξυπνάδα ή όπως θα μπορούσαμε αλλιώς να πούμε, χρήση της ALU του επεξεργαστή. Γι' αυτές ακριβώς τις περιπτώσεις υπάρχουν τα κανάλια DMA (Direct Memory Access) και τα περιφερειακά με υποστήριξη DMA. Αυτά αναλαμβάνουν να κάνουν τη δουλειά της αντιγραφής από και προς την μνήμη στο παρασκήνιο ΧΩΡΙΣ να καταναλώσουν καθόλου επεξεργαστική ισχύ. Επειδή τα περισσότερα DSP επεξεργάζονται σήματα υπάρχει επιτακτική ανάγκη για την μεταφορά μεγάλων ποσοτήτων δεδομένων και τα κανάλια DMA προσφέρουν σημαντική ανακούφιση στον φόρτο του επεξεργαστή.

Ας δούμε όμως ένα παράδειγμα. Έστω ότι θέλουμε να πάρουμε ένα frame εικόνας, να του εφαρμόσουμε ένα φίλτρο η υλοποίηση του οποίου χρειάζεται και ένα αντίγραφο του frame και στην συνέχεια να το στείλουμε προς απεικόνιση. Έστω ότι το frame εικόνας αποτελείται από 1024 bytes δεδομένων.

Η διαδικασία χωρίς κανάλια DMA έχει ως εξής.

1. Ο επεξεργαστής περιμένει ειδοποίηση από το περιφερειακό σύλληψης εικόνας ότι υπάρχουν νέα δεδομένα
2. Ο επεξεργαστής εκτελεί 1024 αντιγραφές από το περιφερειακό στις αντίστοιχες θέσεις μνήμης
3. Ο επεξεργαστής εκτελεί 1024 αντιγραφές για να δημιουργήσει το αντίγραφο
4. Ο επεξεργαστής εφαρμόζει το φίλτρο
5. Ο επεξεργαστής εκτελεί 1024 αντιγραφές από την μνήμη στο περιφερειακό απεικόνισης

Δηλαδή ο επεξεργαστής, πέρα από την εφαρμογή του φίλτρου, εκτελεί τουλάχιστον 3072 αντιγραφές σε, το λιγότερο, 3072 κύκλους.

Η ίδια διαδικασία αν το σύστημα έχει κανάλια DMA έχει ως εξής.

1. Ο επεξεργαστής δίνει εντολή σε ένα DMA κανάλι για ανάγνωση 1024 bytes από το περιφερειακό σύλληψης. Μετά μπορεί να κάνει οτιδήποτε θέλει. π.χ. να εφαρμόζει το φίλτρο στο προηγούμενο frame!
2. Όταν πλέον η μεταφορά έχει ολοκληρωθεί ο επεξεργαστής δίνει εντολή σε ένα κανάλι DMA για την αντιγραφή 1024 bytes από μία θέση μνήμης σε μία άλλη για την δημιουργία του αντιγράφου. Μετά επίσης μπορεί να κάνει οτιδήποτε.
3. Ο επεξεργαστής εφαρμόζει το φίλτρο

4. Ο επεξεργαστής δίνει εντολή σε ένα κανάλι DMA για την μεταφορά 1024 bytes προς το περιφερειακό απεικόνισης. Φυσικά η εργασία αυτή μπορεί να γίνεται παράλληλα με την σύλληψη του frame του βήματος 1!

Συνεπώς ο επεξεργαστής σε αυτή την περίπτωση το μόνο που είχε να κάνει πέρα από το φίλτρο είναι η διαχείριση των τριών καναλιών DMA που μπορεί να ολοκληρωθεί σε λιγότερους από 20 κύκλους.

Γίνεται προφανές ότι τα κανάλια DMA μπορούν να σώσουν πολύτιμο χρόνο εκτέλεσης από τον επεξεργαστή αναλαμβάνοντας την μεταφορά μεγάλων ποσοτήτων δεδομένων από και προς την μνήμη. Ειδικά στις περιπτώσεις που το DSP έχει να χειριστεί περιφερειακά μπορεί το κέρδος να είναι ακόμα μεγαλύτερο γιατί ο επεξεργαστής δεν θα χρειαστεί να περιμένει μέχρι να υπάρχουν διαθέσιμα δεδομένα κατά την ανάγνωση δεδομένων από το περιφερειακό ή να έχει επεξεργαστεί το περιφερειακό τα δεδομένα κατά την εγγραφή δεδομένων στο περιφερειακό.

3.1.2.9 Πολλαπλασιασμός ταχύτητας πυρήνα με PLL

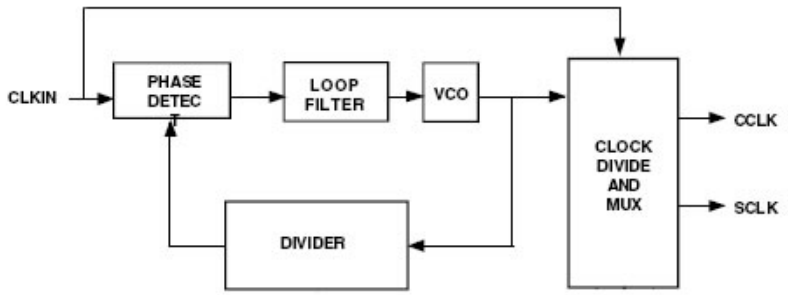
Όσο κι αν φαίνεται περίεργο είναι εξαιρετικά ανεπιθύμητο να έχει κανείς ένα σύστημα που να «τρέχει» στο 1GHz. Ναι μεν το σύστημα θα μπορεί να εκτελεί εξαιρετικά γρήγορα τις πράξεις αλλά το κόστος ενός τέτοιου συστήματος είναι εξαιρετικά υψηλό.

1. Η μνήμη που τρέχει σε υψηλές ταχύτητες είναι συνήθως του τύπου SRAM και είναι πάρα πολύ ακριβή.
2. Τα περιφερειακά (π.χ. A/Ds και D/As) που τρέχουν σε υψηλές ταχύτητες είναι πάρα πολύ ακριβά.
3. Η σχεδίαση πλακετών που μπορούν να λειτουργούν σε τέτοιες ταχύτητες είναι πάρα πολύ δύσκολη και ακριβή. Επιπλέον συστήματα υψηλής ταχύτητας εμφανίζουν μεγαλύτερες πιθανότητες εμφάνισης σφάλματος και συνεπώς έχουν πολύ μεγαλύτερο κόστος συντήρησης.
4. Όσο υψηλότερη είναι η ταχύτητα, τόσο πιο μεγάλη είναι η κατανάλωση ρεύματος εξ' αιτίας των απότομων παλμών ρεύματος που απαιτούνται (μεγάλο di/dt) και αντίστοιχα μεγαλύτερος είναι και ο ηλεκτρομαγνητικός θόρυβος με τον οποίο μολύνουν το περιβάλλον.

Η μόνη λύση στο πρόβλημά μας είναι η «πολιτική δύο ταχυτήτων»! Το σύστημά μας θα τρέχει σε κάποια χαμηλή ταχύτητα με ένα ρολόι συστήματος (system/peripheral clock), ενώ ο πυρήνας του επεξεργαστή θα τρέχει σε μία πολύ υψηλότερη ταχύτητα με το ρολόι πυρήνα (core clock). Τα δύο ρολόγια πρέπει να είναι συγχρονισμένα για να μπορούν να ανταλλάσσουν δεδομένα μεταξύ τους, ο γρήγορος πυρήνας και τα αργά περιφερειακά. Για παράδειγμα ακριβώς κάθε 10 κύκλους του ρολογιού πυρήνα πρέπει να έχουμε ένα κύκλο του ρολογιού συστήματος.

Οι σύγχρονοι υπολογιστές παρόλο που διαφημίζουν ταχύτητες της τάξεως μερικών GHz στην ουσία δεν έχουν ταχύτητα συστήματος που να ξεπερνάει τα 133MHz. Τα πολλά GHz είναι η ταχύτητα με την οποία δουλεύει ο πυρήνας.

Την δημιουργία των ρολογιών του πυρήνα και του συστήματος την αναλαμβάνει ένα κύκλωμα το οποίο λέγεται PLL (Phase Locked Loop) και βρίσκεται μέσα στον επεξεργαστή. Το κύκλωμα αυτό είναι ένας συνδυασμός αναλογικού και ψηφιακού κυκλώματος που παίρνει ως είσοδο ένα σήμα χρονισμού (το οποίο παράγεται έξω από το DSP) και δημιουργεί τα ρολόγια του πυρήνα (CCLK) και του συστήματος (SCLK).



Εικόνα 45. Block διάγραμμα PLL

Μία πολύ σημαντική δυνατότητα που δίνει το PLL είναι να αλλάζει η συχνότητα χρονισμού κατά την διάρκεια εκτέλεσης του προγράμματος. Αυτό σημαίνει ότι εκεί που το DSP θέλει να εφαρμόσει ένα φίλτρο και θέλει μέγιστη απόδοση θα ανεβάσει την συχνότητα χρονισμού στο μέγιστο και θα καταναλώνει βέβαια και την μέγιστη ισχύ. Όταν τελειώσει με το φίλτρο και μετά έχει να κάνει κάτι το οποίο δεν απαιτεί τόση πολλή ταχύτητα, π.χ. να διαβάζει δεδομένα από κάποιο περιφερειακό, μπορεί να κατεβάσει την ταχύτητα χρονισμού στην ελάχιστη δυνατή τιμή και να καταναλώνει ελάχιστη ποσότητα ενέργειας.

Η δυνατότητα δυναμικής αλλαγής της συχνότητας χρονισμού του πυρήνα μας δίνει την δυνατότητα να συμβιβάσουμε τις υψηλές αποδώσεις με την χαμηλή κατανάλωση ρεύματος. Τεχνικά η αλλαγή συχνότητας πυρήνα γίνεται αλλάζοντας τον συντελεστή υποβίβασης του διαιρέτη (divider) που περιέχεται σε κάθε μονάδα PLL.

3.1.3 Κατηγορίες DSPs

Υπάρχουν πολλοί τρόποι να κατηγοριοποιήσει κανείς τα DSPs. Ένας πρόχειρος διαχωρισμός μπορεί να γίνει ανάλογα με το μέγεθος των δεδομένων που μπορούν να χειριστούν. Έτσι χωρίζονται σε 16-bit και 32-bit επεξεργαστές. Ο κυριότερος όμως διαχωρισμός τους είναι με βάση την αναπαράσταση που χρησιμοποιείται για τους αριθμούς και χωρίζονται σε δύο οικογένειες:

1. Επεξεργαστές με αριθμητική σταθερής υποδιαστολής (fixed-point arithmetic) και
2. επεξεργαστές με αριθμητική κινητής υποδιαστολής (floating-point arithmetic)

Θα θεωρήσουμε για να έχουμε ένα κοινό σημείο αναφοράς ότι μιλάμε για αριθμούς οι οποίοι αναπαριστούνται με 32-bit πληροφορίας.

Οι επεξεργαστές σταθερής υποδιαστολής εκτελούν στην πραγματικότητα πράξεις ακεραίων. Εμείς όμως για δική μας ευκολία θεωρούμε την υποδιαστολή όπου μας βολεύει και έχουμε με αυτόν τον τρόπο πράξεις πραγματικών αριθμών. Για παράδειγμα αν έχουμε ακεραίους, μπορούμε να θεωρήσουμε την υποδιαστολή αμέσως δεξιότερα από το bit με την ελάχιστη σημασία.

Πρόσημο	Ακέραιο μέρος	Κλασματικό μέρος
1 bit (+/-)	31 bits (2147483648 αριθμοί)	0 bits

Έτσι έχουμε 31-bit διαθέσιμα για την αναπαράσταση των ακεραίων (το αριστερότερο ψηφίο συνήθως θεωρείται ως το ψηφίο πρόσημου), γεγονός που μας δίνει ένα εύρος απεικόνισης από το -2147483648 έως το +2147483647.

Αν όμως ζητάμε να κάνουμε πράξεις σε αριθμούς με ακρίβεια τριών δεκαδικών ψηφίων τότε πρέπει να θεωρήσουμε ότι ο αριθμός έχει την υποδιαστολή 10 bit αριστερότερα από το τελευταίο σημαντικό ψηφίο. Με αυτόν τον τρόπο έχουμε 1 bit πρόσημου, 21 bits για το ακέραιο μέρος και 10 bits για το κλασματικό μέρος.

Πρόσημο	Ακέραιο μέρος	Κλασματικό μέρος
1 bit (+/-)	21 bits (2097152 αριθμοί)	10 bits (1024 αριθμοί)

Με αυτή την αναπαράσταση μπορούμε να δείξουμε αριθμούς από το -2097152,9990234375 έως τον 2097151,9990234375 με βήμα 0.0009765625. Έχουμε δηλαδή την επιθυμητή ακρίβεια των τριών δεκαδικών ψηφίων. Παρατηρούμε φυσικά ότι το δυναμικό εύρος μας μειώθηκε δραματικά από τα δύο τρισεκατομμύρια του προηγούμενου παραδείγματος στα δύο εκατομμύρια. Βλέπουμε λοιπόν ότι σε αυτή την αναπαράσταση, το εύρος του ακεραίου μέρους και η ακρίβεια του δεκαδικού μέρους έχουν ανταγωνιστικό ρόλο.

Ένα μεγάλο πρόβλημα επίσης εμφανίζεται στις περιπτώσεις σταθερής υποδιαστολής στην πράξη της διαίρεσης όπου είναι πολύ εύκολο να πάρουμε αποτελέσματα με πολύ περιορισμένη ακρίβεια. Για παράδειγμα αν με διαιρέσουμε το 200000/140000 θα πάρουμε ως αποτέλεσμα 1, δηλαδή ένα αποτέλεσμα με εξαιρετικά περιορισμένη ακρίβεια. Παρόμοιες δυστροπίες παρουσιάζονται σε άλλες πράξεις[44].

Στην αναπαράσταση κινητής υποδιαστολής αναπαριστούμε τους αριθμούς με τρόπο παρόμοιο με την επιστημονική απεικόνιση των αριθμών. Για παράδειγμα το 123456 θα απεικονίζεται ως 1.23456×10^2 και επειδή μιλάμε φυσικά στο δυαδικό σύστημα αυτό θα γινόταν 0.941895×2^{17}! Όπως μπορούμε να παρατηρήσουμε μπορούμε να έχουμε τρομερό δυναμικό εύρος αποθηκεύοντας δύο σχετικά μικρούς αριθμούς, το κλασματικό μέρος και τον εκθέτη. Αυτά στην ορολογία της αριθμητικής κινητής υποδιαστολής ονομάζονται mantissa και exponent αντίστοιχα. Ο πιο διαδεδομένος τρόπος απεικόνισης κινητής υποδιαστολής είναι ο IEEE Standard 754[45]. Σύμφωνα με αυτόν έχουμε 1 bit πρόσημου, 8 bit εκθέτη και 23 για το κλασματικό μέρος.

Πρόσημο	Εκθέτης (προσημασμένος)	Κλασματικό μέρος
1 bit (+/-)	8 bits (-126 έως 127)	23 bits (8388608 αριθμοί)

Με αυτό τον τρόπο μας δίνεται η δυνατότητα να αναπαραστήσουμε $\pm 2^{-126}$ έως $(2-2^{-23})x2^{127}$ δηλαδή περίπου από $\pm 10^{-45}$ έως 10^{39} με ακρίβεια σχεδόν δεκάτου του εκατομμυριοστού. Όπως βλέπουμε, η αναπαράσταση κινητής υποδιαστολής δίνει ένα τεράστιο δυναμικό εύρος και μεγάλη ακρίβεια ταυτόχρονα έτσι ώστε να καλύπτει τις ανάγκες κάθε εφαρμογής. Επιπλέον δεν αντιμετωπίζει κανένα πρόβλημα με την πράξη της διαίρεσης αφού μπορεί να επιστρέψει δεκαδικούς αριθμούς με ακρίβεια 23ων bits.

Προφανώς η πρώτη επιλογή θα ήταν η επιλογή ενός DSP με αριθμητική κινητής υποδιαστολής. Αυτή όμως δεν είναι και η πιο συμφέρουσα επιλογή. Ο λόγος γι' αυτό βρίσκεται σε μία τεχνική λεπτομέρεια. Ας δούμε την πρόσθεση δύο αριθμών κινητής υποδιαστολής, για ευκολία στο δεκαδικό σύστημα.

Έστω ότι έχουμε να προσθέσουμε το $0.34 \times 10^8 + 0.21 \times 10^6$. Προφανώς η πρόσθεση αυτή δεν μπορεί να γίνει άμεσα γιατί οι εκθέτες είναι διαφορετικοί. Θα πρέπει λοιπόν να γράψουμε:

$0.34 \times 10^8 + 0.21 \times 10^6 = 34 \times 10^6 + 0.21 \times 10^6 = 34.21 \times 10^6 = 0.3421 \times 10^8$

Παρατηρούμε ότι εκτός από την λογική της πρόσθεσης χρειάζεται να μπορούμε να κάνουμε γρήγορα και μετακινήσεις της υποδιαστολής ώστε να ανάγουμε τα νούμερα στην ίδια τάξη εκθέτη. Αυτό γίνεται με ένα αρκετά σύνθετο κύκλωμα που λέγεται barrel shifter.

Η δυνατότητα χειρισμού αριθμών κινητής υποδιαστολής απαιτεί αρκετά περισσότερα κυκλώματα μέσα στο ολοκληρωμένο κύκλωμα του DSP που συνεπάγεται μεγαλύτερο κόστος, υψηλότερη κατανάλωση ρεύματος, πιθανώς πιο αργούς χρόνους εκτέλεσης και περισσότερο χώρο. Στην ουσία δηλαδή έχουμε αρκετά μειονεκτήματα με μόνο πλεονέκτημα την αρκετά μεγαλύτερη ευκολία στον προγραμματισμό και τον συνδυασμό πραγματικά μεγάλου δυναμικού εύρους και ακρίβειας.

Αναγκαιότητα για την χρήση DSPs με δυνατότητες κινητής υποδιαστολής έχουμε μόνο όταν θέλουμε να κάνουμε γρήγορη προτυποποίηση και σε πραγματικά απαιτητικές εφαρμογές που

[44] http://home.earthlink.net/~yatescr/papers.htm
[45] http://stevehollasch.com/cgindex/coding/ieeefloat.html

απαιτούν μεγάλο δυναμικό εύρος. Σε όλες τις άλλες περιπτώσεις που μας ενδιαφέρει πολύ το χαμηλό κόστος και η χαμηλή κατανάλωση ρεύματος, οι επεξεργαστές σταθερής υποδιαστολής θα κάνουν μία χαρά την δουλειά τους.

Οι περισσότεροι άλλωστε αλγόριθμοι που έχουν σχεδιαστεί για DSPs έχουν γραφτεί για επεξεργαστές σταθερής υποδιαστολής και με σχετικά απλές τεχνικές μπορεί κανείς να ξεπεράσει τις περισσότερες αδυναμίες της σταθερής υποδιαστολής. Το μόνο το οποίο ουσιαστικά χρειάζεται είναι αρκετές ώρες εργασίας μηχανικών για να μετατρέψουν έναν floating point αλγόριθμο σε fixed point. Αν μάλιστα η υλοποίηση του αλγορίθμου προορίζεται για μία συγκεκριμένη εφαρμογή, είναι σχεδόν σίγουρο ότι η fixed point υλοποίηση θα είναι αρκετά γρηγορότερη και θα εκτελείται με λιγότερη απαίτηση σε ενέργεια.

Τα καλά βέβαια νέα είναι ότι πλέον οι fixed point επεξεργαστές έχουν γίνει αρκετά γρήγοροι ώστε να μπορούν να «εξομοιώνουν» πράξεις floating point με συμφέρουσα ταχύτητα και επιπλέον οι επεξεργαστές floating point, ακολουθώντας την συνήθη πορεία κάθε νέας τεχνολογίας, γίνονται συνεχώς όλο και πιο φτηνοί και προσιτοί στην αγορά.

Συνεπώς αν θέλαμε να συνοψίσουμε χοντρικά την αγορά των DSPs θα λέγαμε ότι υπάρχουν οι εξής κατηγορίες:

1. 16-bit επεξεργαστές σταθερής υποδιαστολής: Χαμηλού κόστους, γενικής χρήσης για εφαρμογές π.χ. ήχου, μετρήσεων, συνήθους αυτομάτου ελέγχου κ.τ.λ.
2. 24-bit επεξεργαστές σταθερής υποδιαστολής: Για εφαρμογές ήχου υψηλής πιστότητας.
3. 32-bit επεξεργαστές κινητής υποδιαστολής: Για απαιτητικές εφαρμογές υψηλής ακρίβειας και γρήγορη προτυποποίηση.

3.1.4 Εναλλακτικές τεχνολογίες

Υπάρχουν κάποιες εναλλακτικές τεχνολογίες υλοποίησης αλγορίθμων ψηφιακής επεξεργασίας σήματος. Οι σημαντικότερες είναι τα FPGAs (προγραμματιζόμενη λογική γενικής χρήσεως), τα ASICs (ειδικές εκδόσεις ολοκληρωμένων για πελάτες μεγάλων ποσοτήτων), και οι GPPs (γενικής χρήσης μικροεπεξεργαστές)[46]. Οι τεχνολογίες αυτές, αυτή την στιγμή, παίζουν στο σύνολό τους ένα μη ανταγωνιστικό ρόλο απέναντι στην τεχνολογία των προγραμματιζόμενων DSPs λόγω συγκεκριμένων αδυναμιών τους που εντέλει οδηγούν σε υψηλότερο κόστος. Σε πολλές όμως περιπτώσεις δρουν υποστηρίζοντας την τεχνολογία των DSPs, γεμίζοντας ορισμένα κενά όπου υπάρχουν εξαιρετικά εξειδικευμένες απαιτήσεις.

3.1.4.1 FPGAs

Τα DSPs, οι μικροεπεξεργαστές και οι πιο σύνθετοι επεξεργαστές όπως οι Pentium IV, όλα τους είναι φτιαγμένα από πολύ απλά στοιχειώδη κομμάτια. Συγκεκριμένα όλα τα επιτεύγματα του σύγχρονου ψηφιακού πολιτισμού μπορούν να κατασκευαστούν με πύλες NAND και πολύ «καλώδιο». Ακριβώς αυτό το πράγμα είναι τα FPGAs. Κάποιες στοιχειώδεις δομικές βαθμίδες όπως πύλες και flip-flops και ατελείωτα μέτρα ηλεκτρονικού καλωδίου.

Η όλη δουλειά κάποιου που θα ήθελε να προγραμματίσει ένα FPGA είναι να σχεδιάσει τις συνδέσεις μεταξύ των δομικών στοιχείων του. Δυστυχώς μία τέτοια περιγραφή θα ήταν περιορισμένης χρησιμότητας μιας και δεν θα μπορούσε να μεταφερθεί από ένα FPGA σε ένα άλλο με ελαφρώς διαφορετικές βαθμίδες. Θα έπρεπε να γινόταν η δουλειά του σχεδιασμού απ' την αρχή για κάθε περίπτωση.

Για να ξεπεραστεί αυτό το πρόβλημα δημιουργήθηκαν κάποιες γλώσσες περιγραφής hardware (Hardware Description Languages – HDL) με τις οποίες μπορεί κανείς να ΠΕΡΙΓΡΑΨΕΙ το τι θέλει να κάνει ένα FPGA. Μετά τρέχει ένα πρόγραμμα «σύνθεσης» αντίστοιχο με τους compilers για τα προγράμματα και αυτό σχεδιάζει τις ακριβείς συνδέσεις για το συγκεκριμένο FPGA που μας

[46]http://dspvillage.ti.com/docs/catalog/dspplatform/details.jhtml?templateId=5121&path=templatedata/cm/dspdetail/data/vil_getstd_whatis

ενδιαφέρει. Αν θελήσουμε να χρησιμοποιήσουμε την ίδια περιγραφή για κάποιο άλλο FPGA αρκεί να χρησιμοποιήσουμε ένα διαφορετικό πρόγραμμα «σύνθεσης» κατάλληλο για το νέο FPGA. Οι πιο διαδεδομένες γλώσσες HDL είναι η VHDL και η Verilog ενώ η εταιρεία Altera έχει κάνει μία ελαφρώς τροποποιημένη γλώσσα όμοια με την VHDL που την ονομάζει AHDL[47].

Η χρήση περιγραφικών γλωσσών κάνει δυνατή την επαναχρησιμοποίηση βαθμίδων. Αρκετές λοιπόν εταιρίες δημιουργούν τέτοιες περιγραφές σύνθετων βαθμίδων και τις ελέγχουν εξαντλητικά. Αυτή μπορεί να είναι μία πολύ σύνθετη και ακριβή διαδικασία. Οι ελεγμένες και ως εκ' τούτου εξαιρετικά αξιόπιστες βαθμίδες ονομάζονται Intellectual Properties (IPs). Αυτά είναι αντίστοιχα των συναρτήσεων (functions) που γνωρίζουμε από τις γλώσσες υψηλού επιπέδου και συχνά πωλούνται σε «πακέτα» αντίστοιχα των βιβλιοθηκών συναρτήσεων.

Με αυτά τα ολοκληρωμένα μπορεί να υλοποιηθεί οποιοδήποτε ψηφιακό «κατασκεύασμα» και φυσικά DSPs. Πιο συγκεκριμένα υπάρχουνε έτοιμες βιβλιοθήκες με IPs για σχεδόν κάθε λειτουργία που θα μπορούσε να κάνει ένα DSP. Βαθμίδες MAC, διαιρέτες, μαθηματικές συναρτήσεις όπως τετραγωνικές ρίζες, ημίτονα και συνημίτονα, ψηφιακά φίλτρα, όλα μπορούν να υλοποιηθούν πάνω σε ένα FPGA. Το καλύτερο όμως είναι ότι με τα FPGAs δεν τίθεται κανένας περιορισμός αρχιτεκτονικής όπως στα DSPs. Μπορείς για παράδειγμα να έχεις 256 βαθμίδες MAC την μία δίπλα στην άλλη και ως εκ τούτου να μπορείς να κάνεις μετασχηματισμό FFT ή ψηφιακό φίλτρο 256^{ης} τάξεως που θα εκτελείται σε ένα μόνο κύκλο. Επιπλέον είναι δυνατόν οι μισές βαθμίδες του «χειροποίητου» DSP υλοποιημένου σε FPGA να είναι με αριθμητική 16-bit σταθερής υποδιαστολής ενώ οι άλλες μισές να είναι με αριθμητική 64-bit κινητής υποδιαστολής! Προφανώς με τα FPGAs μπορεί να υλοποιηθεί οτιδήποτε αρκεί να επιλεγεί ένα αρκετά «μεγάλο» FPGA ώστε να χωράει ότι χρειάζεται.

Η μεγαλύτερη εταιρία παραγωγής FPGAs είναι αυτή τη στιγμή η Xilinx με αδιαμφισβήτητα και διαχρονικά την πρώτη θέση στην αγορά. Αμέσως μετά έρχονται οι Altera και η Lucent. Η Xilinx κάνει μία μεγάλη προσπάθεια να αναδείξει το πλήθος των εφαρμογών που μπορούν να έχουν τα FPGA της διαθέτοντας μία πολύ μεγάλη βιβλιοθήκη με IPs σε τιμές, βέβαια, ανάλογες της αξιοπιστίας τους (...). Επιπλέον ειδικά για εφαρμογές επεξεργασίας σήματος διαθέτει FPGAs με ειδικές βαθμίδες που διευκολύνουν την σχεδίαση και παρουσιάζουν κορυφαίες επιδόσεις. Μπορούμε να δούμε παρακάτω κάποια από τα σύγχρονά της προϊόντα[48]. (* Ο λόγος που ασχολούμαστε με τα προϊόντα της συγκεκριμένης εταιρίας δεν είναι σε καμία περίπτωση διαφημιστικός. Απλά αυτή την στιγμή διαθέτει τα προϊόντα με τις καλύτερες επιδόσεις και επαρκή στοιχεία για την παρουσίαση τους).

Υψηλότερη επίδοση DSP ανά οικογένεια			
Ισχυρότερο μέλος της οικογένειας	Μέγιστος αριθμός βαθμίδων MAC	Μέγιστη συχνότητα χρονισμού (MHz)	Συνολική επίδοση DSP (GMACs)
Virtex-4SX55	512*	500	256
Virtex-4FX140	192*	500	96
Virtex-4LX200	96*	500	48
Virtex-II Pro100	444**	300	133
Spartan-35000	104**	185	19
* Χρησιμοποιώντας τις βαθμίδες XtremeDSP μόνο (18x18 bit πολλαπλασιασμός, 48 bit αθροιστής)			
** Βαθμίδες MAC που χρησιμοποιούν τους έτοιμους 18x18 πολλαπλασιαστές μόνο			

Ο παραπάνω πίνακας τα λέει όλα! Ταχύτητες μέχρι και 256GMACs είναι δυνατόν να επιτευχθούν την στιγμή που τα σημερινά προγραμματιζόμενα DSPs δίνουν ταχύτητες το πολύ μέχρι 10GMACs! Η απίστευτη αυτή ταχύτητα οφείλεται στο ότι έχουμε πάρα πολλές βαθμίδες MAC να λειτουργούν παράλληλα. Πρέπει να σημειωθεί επίσης ότι αν χρειάζεται μία ειδική συνάρτηση π.χ. arcos(x/y), αυτή μπορεί να υλοποιηθεί κατευθείαν στο hardware και να εκτελείται σε ένα μόνο κύκλο ή και περισσότερες φορές ανά κύκλο, τοποθετώντας ίδιες βαθμίδες παράλληλα.

[47] Special Purpose Digital Processors (DSP) F. Mayer-Lindenberg, TUHH (dsp.pdf)
[48] http://www.xilinx.com/products/design_resources/dsp_central/info/fpga_overview.htm#comparison

Επιπλέον τα ολοκληρωμένα της οικογένειας Virtex-4 FX μπορούν να έχουν πάνω τους μέχρι και δύο κανονικούς 32-bit επεξεργαστές Power PC της IBM που «τρέχουν» στα 450MHz, βαθμίδες για άμεση σύνδεση σε δίκτυα Ethernet με ταχύτητες μέχρι και 1Gbps και πολλά άλλα που μπορούμε να τα δούμε στον παρακάτω συγκεντρωτικό πίνακα:

Χαρακτηριστικά	Virtex-II Pro Platform	Virtex-II	Spartan-3
Κεντρική μνήμη*	10 MB	3 MB	1.8 MB
Διασκορπισμένη μνήμη*	1.7 MB	1.45 MB	520 KB
18x18 bit πολλαπλασιαστές*	556	168	104
Λογικές κυψέλες*	125K	104K	74K
Διαθέσιμες γραμμές I/O*	1200	1108	784
Ταχύτητα ρολογιού	300+ MHz	220 MHz	185 MHz
Πομποδέκτες Multi-gigabit	Yes	No	No
Ενσωματωμένος PowerPC	Yes	No	No
MicroBlaze Soft Processor	Yes	Yes	Yes

* Μέγιστες τιμές

Προφανώς η τεχνολογία των FPGAs έχει σαφή πλεονεκτήματα. Δεν θα έπρεπε να παραλείψουμε την εύκολη προτυποποίηση με την βοήθεια του MATLAB™ και του Simulink™.

Γιατί λοιπόν μιλάμε ακόμα για DSPs και δεν ασχολούμαστε αποκλειστικά με FPGAs; Στην συγκεκριμένη περίπτωση η αιτία που δεν χρησιμοποιούνται τόσο πολύ είναι ίδια με την αιτία που τα κάνει να ξεχωρίζουν: Είναι πολύ ισχυρά! Πραγματικά, στο 90% των εφαρμογών δεν χρειάζεται ισχύς περισσότερη από αυτή που προσφέρουν τα προγραμματιζόμενα DSPs. Οι αδυναμίες των FPGAs σε σχέση με τα DSPs είναι οι εξής:

1. Έχουν μεγαλύτερο κόστος σε σύγκριση με DSP με τις ίδιες δυνατότητες. Αυτό είναι αναμενόμενο λόγω του ότι (για να το πούμε απλά), στην περίπτωση των DSPs, πληρώνεις μόνο το «καλώδιο» που χρειάζεσαι για την σύνδεση των λογικών βαθμίδων (που υποθέτουμε ότι είναι οι ίδιες και στις δύο περιπτώσεις), ενώ στα FPGAs πληρώνεις πολύ περισσότερο «καλώδιο» που μένει ανεκμετάλλευτο και τα «βύσματα».
2. Τυπικά έχουν μεγαλύτερη κατανάλωση ρεύματος από αυτή του αντίστοιχου DSP. Αυτό συμβαίνει κυρίως λόγω των αυξημένων ταχυτήτων λειτουργίας (έλλειψη PLL) και του λόγου που αναφέρθηκε παραπάνω.
3. Έχουν αυξημένη πολυπλοκότητα. Εκεί που ο προγραμματιστής έχει να ελέγξει μόνο τον αλγόριθμό του στην περίπτωση των DSPs, στα FPGAs πρέπει να ελέγξει ότι και η τελική σχεδίαση των συνδέσεων πάνω στο FPGA πράγματι δουλεύει, μέσα στα πλαίσια χρονισμού του συστήματος και γενικά ένα τεχνικό κομμάτι που στα DSPs είναι σίγουρο ότι δουλεύει σωστά μέσα στις δεδομένες προδιαγραφές. Αυτό μπορεί να κάνει αρκετά μεγαλύτερη τη διάρκεια ανάπτυξης μίας εφαρμογής και σίγουρα απαιτεί περισσότερο και πιο εξειδικευμένο προσωπικό από αυτό των DSPs.

Όταν λοιπόν συναντάμε FPGAs σε κάποια εφαρμογή, αυτή θα έχει εξαιρετικά μεγάλες απαιτήσεις για ταχύτητα και θα τα συναντάμε συνήθως σε συνδυασμό με κάποιο DSP. Το DSP θα εκτελεί τις κλασικές λειτουργίες επεξεργασίας σήματος, ενώ το FPGA θα υλοποιεί κάποιες ειδικές συναρτήσεις τις οποίες θα εκτελεί ταχύτατα ή κάποια διασύνδεση με εξωτερικά περιφερειακά για οικονομία των πολύτιμων ακροδεκτών (I/O) του DSP π.χ. διαχείριση μνήμης, PCI interface και άλλα. Τα πράγματα που θα υλοποιηθούν στο FPGA θα είναι συνήθως έτοιμα ελεγμένα IPs τα οποία «υποθέτουμε ότι δουλεύουν» και αφού δεν τα συνδέουμε εσωτερικά με κάποιο δικό μας «περίεργο» τρόπο μπορούμε να αποφύγουμε τη χρονοβόρα και ακριβή διαδικασία του εξαντλητικού ελέγχου. Φυσικά κάτω από αυτές τις συνθήκες μπορούν να χρησιμοποιηθούν FPGAs που δεν θα είναι ούτε πολύ ισχυρά ούτε και θα «τρέχουν» τρομερά γρήγορα. Με αυτό τον τρόπο μπορεί να επιτευχθούν πολύ οικονομικές λύσεις και με αρκετά χαμηλή κατανάλωση ρεύματος.

Μία προσωπική εκτίμηση είναι ότι στο μέλλον και αυτές οι δύο τεχνολογίες θα συγκλίνουν, όχι όμως με την αντικατάσταση των DSPs από FPGAs όπως θα περίμενε κανείς, αλλά από το αντίθετο. Συγκεκριμένα θεωρώ πιθανό να επικρατήσουν DSPs που θα έχουν ενσωματωμένη και

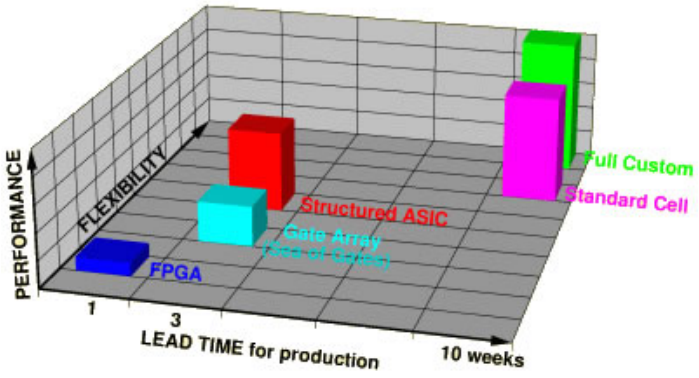
μία περιοχή FPGA για την υλοποίηση λογικής γενικής χρήσεως και των διασυνδέσεων με τα pins εισόδου εξόδου. (Προς αυτή την κατεύθυνση φαίνεται άλλωστε να κινείται και η οικογένεια Virtex-4 με τα ενσωματωμένα PowerPCs). Με αυτό τον τρόπο θα έχουμε με ένα μόνο ολοκληρωμένο, μέγιστη ευελιξία, ταχύτητα και φυσικά ευκολία στον σχεδιασμό και προγραμματισμό. Μεγάλο επίσης ρόλο θα παίξει και η ενσωμάτωση μνήμης τεχνολογίας Flash στα DSPs αυτά, έτσι ώστε να μπορούν να προγραμματίζονται πολλές φορές, εύκολα και γρήγορα και να μην υπάρχει η ανάγκη για κάποια εξωτερική μνήμη όπως απαιτείται μέχρι στιγμής για τα περισσότερα DSPs και FPGAs.

3.1.4.2 ASICs

Τα ASICs (Application Specific Integrated Circuits) είναι το αμέσως επόμενο βήμα μετά τα FPGAs. Είναι ολοκληρωμένα κυκλώματα κατά παραγγελία τα οποία κατασκευάζονται εξ' αρχής για μία συγκεκριμένη εφαρμογή. Υπάρχουν δύο κύριες εκδοχές για την σχεδίαση και κατασκευή τους:

1. Μερικώς ελεύθερη σχεδίαση που γίνεται πάνω σε υπόστρωμα με ήδη τυπωμένες λειτουργικές βαθμίδες. Οι βαθμίδες αυτές συνδέονται με την βοήθεια μίας μόνο αγώγιμης επίστρωσης η οποία σχεδιάζεται και καθορίζει τη λειτουργικότητα του ολοκληρωμένου. Η εκδοχή αυτή θυμίζει FPGAs μόνο που εδώ το «ηλεκτρονικό καλώδιο» των FPGAs έχει αντικατασταθεί από το πραγματικό «καλώδιο» της ειδικής επίστρωσης.
2. Πλήρης σχεδίαση όλων των επιστρώσεων. Αυτή προσφέρει την μέγιστη ευελιξία.

Με τα ASICs μπορούμε να έχουμε όλα τα πλεονεκτήματα των DSPs και των FPGAs μαζί, αφού μπορούν να σχεδιαστούν ολοκληρωμένα με τις βαθμίδες που χρειάζονται μόνο, όλες τις απαραίτητες συναρτήσεις για εξαιρετική ταχύτητα με την ελάχιστη δυνατή κατανάλωση ρεύματος και το κυριότερο το ελάχιστο δυνατό κόστος. Επιπλέον είναι δυνατόν εκτός από τα ψηφιακά μέρη του κυκλώματος να ενσωματωθούν και αναλογικά μέρη π.χ. τελεστικοί ενισχυτές, πηγές τάσης και ρεύματος, A/Ds – D/As και να έχουμε με αυτόν τον τρόπο ένα πλήρες σύστημα μέσα σε ένα μόνο ολοκληρωμένο[49].



Εικόνα 46. Συγκριτικός χάρτης τεχνολογιών ASICs και FPGA[50]

Όπως βλέπουμε στον παραπάνω χάρτη, οι διάφορες τεχνολογίες ASICs παρουσιάζουν εξαιρετικά πλεονεκτήματα ως προς τις επιδόσεις σε σύγκριση με τα FPGAs, αλλά σημαντικά μειονεκτήματα ως προς τον χρόνο που χρειάζεται για την ανάπτυξη μίας εφαρμογής.

Ελάχιστοι σχεδιάζουν ASICs για τους εξής τρεις βασικούς λόγους:

1. Πρέπει να είναι πάρα πολύ μεγάλη παραγγελία. Η δημιουργία ASICs αρχίζει να συμφέρει για ποσότητες από 5000 ολοκληρωμένα και πάνω για ολοκληρωμένα για τα οποία σχεδιάζεται μόνο η μάσκα (gate arrays) με κόστος ανά κομμάτι της τάξης των $10 και από 500.000 κομμάτια και πάνω για ολοκληρωμένα που σχεδιάζονται πλήρως με κόστος ανά

[49] www.opencores.org/forums.cgi/cores/2003/09/00043
[50] Dr. Tobias Ellermeyer - Reducing NRE and Turnaround Time with Structured ASICs (http://www.micram-me.com/services/asic.htm)

κομμάτι της τάξης των $3. Προφανώς είναι σχετικά λίγοι αυτοί που μπορούν να κάνουν τέτοιες παραγγελίες δεδομένου και του αυξημένου ρίσκου όπως αναλύουμε παρακάτω.
2. Τα ASICs δεν μπορούν να τροποποιηθούν μετά την αρχική τους σχεδίαση. Αυτό συμβαίνει επειδή στην πλειονότητά τους δεν είναι προγραμματιζόμενα. Συνεπώς πρέπει όσα κομμάτια παραγγείλεις, τόσα να πουλήσεις, χωρίς την παραμικρή δυνατότητα για τροποποίηση σύμφωνα με τις επιθυμίες κάποιου πελάτη.
3. Αν μία παρτίδα βρεθεί να έχει ένα ελάττωμα τότε όλη πάει για πέταμα. Αυτό συνεπάγεται αυξημένες απαιτήσεις εξαντλητικού ελέγχου, πολύ πιο αυστηρού από αυτόν των FPGAs και μάλιστα εξολοκλήρου με την χρήση εξομοίωσης σε υπολογιστές, αφού πρέπει να είσαι πολύ «καλός πελάτης» για να δεχτεί το εργοστάσιο παραγωγής ASIC να κάνει δείγματα πριν την μαζική παραγωγή. Θα έπρεπε να σημειωθεί ότι η προτυποποίηση των ASICs γίνεται πολλές φορές πάνω σε FPGAs. Μία από τις πιο οικονομικές εκδοχές για σχεδίαση ASIC με μάσκα περιλαμβάνει περίπου 7 εβδομάδες σχεδίασης με κόστος 3000$ ανά εβδομάδα και $5000 για τα πρότυπα[51]. Για πλήρως σχεδιαζόμενα ASICs το κόστος σχεδίασης είναι περίπου $500.000 και από $100.000 έως και πάνω από $2.000.000 για τα έξοδα παραγωγής μασκών.
4. Ο αυξημένος χρόνος ανάπτυξης κάνει το προϊόν να είναι ξεπερασμένο όταν έχει πλέον βγει στην αγορά.

Όπως είναι αναμενόμενο λοιπόν, σε ASIC βλέπουμε συνήθως υλοποιήσεις προϊόντων που ικανοποιούν «κλασικές» γενικού τύπου απαιτήσεις, που έχουν σίγουρο αγοραστικό κοινό και φυσικά μόνο από λίγες μεγάλες εταιρίες. Αυτό θα αλλάξει ελαφρώς στα επόμενα χρόνια καθώς η τεχνολογία κατασκευής ολοκληρωμένων εξελίσσεται και εμφανίζονται μέθοδοι παραγωγής χαμηλότερου κόστους όπως επίσης και γιατί θα αρχίσει σιγά σιγά να εμφανίζεται σε μεγαλύτερες ποσότητες εξοπλισμός κατασκευής που δεν θα τον χρειάζονται πλέον τα μεγάλα εργοστάσια και θα τα διαθέτουν μεταχειρισμένο σε αρκετά χαμηλές τιμές. Ακόμα και έτσι, τα ASICs θα συνεχίσουν να παραμένουν τεχνολογία για λίγους.

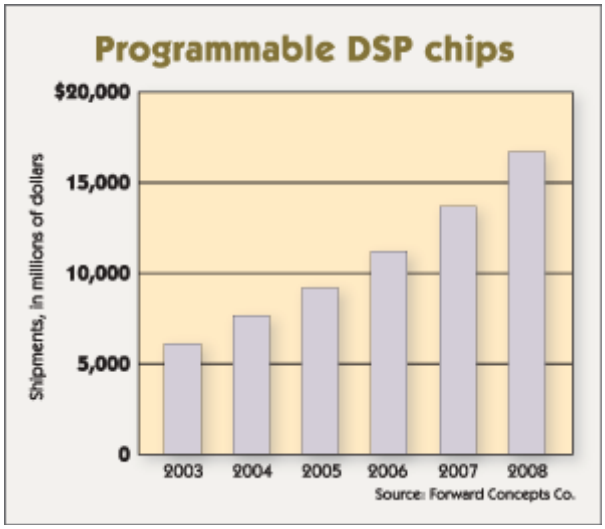
3.1.4.2 GPPs

Οι κλασικοί γενικής χρήσεως μικροεπεξεργαστές (GPPs) μπορούν να κάνουν κάποια επεξεργασία σήματος σε εφαρμογές με εξαιρετικά χαμηλές απαιτήσεις προσφέροντας χαμηλό κόστος. Παρόλα αυτά η περιορισμένη τους ταχύτητα και η υψηλή κατανάλωση ρεύματος είναι σοβαρά μειονεκτήματα. Η αγορά των γενικής χρήσης μικροεπεξεργαστών αναμένεται να περιοριστεί δραματικά τα επόμενα χρόνια μιας και το κόστος των DSPs τείνει να προσεγγίσει αυτό των GPPs ενώ τα πρώτα προσφέρουν ασύγκριτα μεγαλύτερες δυνατότητες.

3.1.5 Η εφαρμογές των DSPs στην σύγχρονη αγορά

Τα τελευταία χρόνια παρατηρείται μεγάλη αύξηση των εφαρμογών που βασίζονται σε DSPs και φυσικά μια πολύ αισιόδοξη ανοδική πορεία των πωλήσεων σε αυτόν τον κλάδο. Αυτή την στιγμή φαίνεται ότι η τεχνολογία των DSPs πραγματικά οδηγεί την αγορά των ημιαγωγών[52].

[51] http://www.arraydesign.com/design/
[52] http://www.eet.com/article/showArticle.jhtml?articleId=21400223



Εικόνα 47. Οι πωλήσεις των DSPs αναμένεται αυξάνονται συνεχώς τα επόμενα χρόνια.

Κατάταξη 2003	Εταιρία	Έσοδα για το 2003 (εκατομμύρια $ διεθνώς)	Έσοδα για το 2002 (εκατομμύρια $ διεθνώς)	Ποσοστό αγοράς %
1.	Texas Instruments	$2900	$2150	47%
2.	Agere Systems	$670	$685	11%
3.	Motorola	$631	$684	10%
4.	Analog Devices	$250	$242	4%

Πηγή: IC INSIGHTS

Η κορυφαία εταιρία του κλάδου αυτή την στιγμή είναι η Texas Instruments η οποία το 2003 κατείχε το 47% της αγοράς με μεγάλη απόσταση από τους επόμενους. Είναι μία εταιρία που δραστηριοποιήθηκε από νωρίς και δυναμικά στον χώρο των DSPs και η απότομη αύξηση της ζήτησης την βρήκε έτοιμη ωθώντας την στην κορυφή.

Η κύρια δραστηριοποίηση της TI είναι στην αγορά της κινητής τηλεφωνίας η οποία γνωρίζουμε ότι έχει πάρα πολύ μεγάλη αύξηση τα τελευταία χρόνια. Πιο συγκεκριμένα το 60% των κινητών τηλεφώνων έχει μέσα του DSP της Texas Instruments[53].

Στην δεύτερη θέση βρίσκεται η Agere Systems (πρώην Lucent Microelectronics) η οποία και αυτή δραστηριοποιείται έντονα στον τομέα των τηλεπικοινωνιών.

Σχεδόν με τις ίδιες πωλήσεις αλλά στην τρίτη θέση βρίσκεται και η Motorola με ελαφρώς καθοδική πορεία (πτώση 8% στις πωλήσεις) για το 2003. Όπως είναι αναμενόμενο μεγάλο ποσοστό (περίπου 50%) των DSPs της Motorola παραμένει στην ίδια εταιρία για της ανάγκες των κινητών της τηλεφώνων ενώ αρκετά διαδεδομένη είναι η χρήση τους και στις εφαρμογές επαγγελματικού ήχου υψηλής πιστότητας[54]. Το τμήμα DSP της Motorola πρόσφατα έγινε αυτόνομο ως νέα εταιρία με το όνομα Freescale Semiconductor σε μία προσπάθεια να επεκτείνει την αγορά της πέρα από τη μητρική εταιρεία.

Την τέταρτη θέση έχει η Analog Devices με σημαντικά όμως ανοδική πορεία (34% αύξηση πωλήσεων). Η Analog Devices είναι πολύ καλή στις ενσύρματες και ασύρματες επικοινωνίες. Συγκεκριμένα καταλαμβάνει την πρώτη θέση στις πωλήσεις στην αγορά ADSL modems και σε συστοιχίες modems που χρησιμοποιούνται για dialup συνδέσεις. Η συνεργασία της με την Microsoft για τα καινούργια της set-top boxes και η διάθεση πολλών νέων προϊόντων όπως του ADSP-BF561 και του ADSP-BF533 σε έκδοση των 750MHz αναμένεται να επισφραγίσει την ανοδική πορεία των πωλήσεων και για το 2004.

[53] http://www.manufacturing.net/pur/article/CA416351.html?stt=001&pubdate=05%2F20%2F04
[54] http://www.eetimes.com/story/OEG20010604S0057

Μία αναφορά πρέπει να γίνει επίσης και στην Atmel η οποία παρουσιάζει κάποια πολύ ισχυρά DSPs με εγγενή δυνατότητα επεξεργασίας μιγαδικών αριθμών και εκτέλεση δέκα με δεκαπέντε λειτουργιών ανά κύκλο κάνοντας δυνατό τον υπολογισμό ενός στοιχειώδους τμήματος FFT σε ένα μόνο κύκλο. Παρόλο που όλα αυτά είναι αρκετά νέα στην αγορά, έχουν γίνει αποδεκτά με ενθουσιασμό.

Στον τομέα των FPGAs και γενικότερα της προγραμματιζόμενης λογικής η Xilinx και η Altera είναι οι δύο μεγάλοι ανταγωνιστές αυτή τη στιγμή με την Xilinx τον αδιαμφισβήτητο πρωτοπόρο κατέχοντας το 47% της αγοράς και με μία ετήσια αύξηση των πωλήσεων κατά 50% μέσα στο 2003! Σίγουρα θα προσπαθήσει να επεκτείνει την κυριαρχία της και στις εφαρμογές DSPs πιθανώς διαθέτοντας με πιο ευνοϊκούς όρους τα IPs για DSP βαθμίδες, ρίχνοντας κι άλλο τις τιμές και δημιουργώντας FPGAs με περισσότερες διευκολύνσεις για τον σχεδιαστή ενός συστήματος DSP (έτοιμα interfaces για περιφερειακά π.χ. A/Ds – D/As – video encoders/decoders, περισσότερη on-chip RAM, έτοιμα interfaces για διάφορους τύπους εξωτερικής μνήμης κ.α.).

1. Από όλα τα παραπάνω γίνεται φανερό ότι αδιαμφισβήτητα ο μεγαλύτερος κλάδος εφαρμογών για DSPs είναι αυτός των ασύρματων επικοινωνιών ο οποίος το 2003 αύξησε τις πωλήσεις του 32% στα 4.2 δισεκατομμύρια δολάρια. Αυτός περιλαμβάνει κινητά τηλέφωνα, ασύρματες κάρτες δικτύου, συστήματα βασισμένα στις τεχνολογίες τηλεπικοινωνιών 2.5 και 3G (τρίτης γενιάς), συσκευές Bluetooth και άλλα.
2. Ο δεύτερος μεγάλος κλάδος είναι οι καταναλωτικές συσκευές (CD-players, τηλεοράσεις, κάμερες κ.ο.κ.) ο οποίος το 2003 είχε αύξηση των πωλήσεων κατά 108% στα 650 εκατ. δολάρια.
3. Μεγάλη αύξηση (70%) είχαν οι πωλήσεις για εφαρμογές που αφορούν computer και περιφερειακά και κυρίως ελεγκτές για CD-ROM drives και scanners.
4. Επίσης αυξήθηκαν οι πωλήσεις (24%) στην βιομηχανία αυτοκινήτων σε εφαρμογές όπως GPS, συσκευές hands free, αναγνώρισης φωνής και αναγγελίας, συσκευές αναπαραγωγής mp3 και video.
5. Συνεχώς αυξανόμενη ζήτηση φαίνεται να υπάρχει και σε όλες τις εφαρμογές που σχετίζονται με την επεξεργασία εικόνας και video. Γίνεται φανερό ότι οι απαιτήσεις του αγοραστικού κοινού θα προσανατολίζονται όλο και εντονότερα στην ενσωμάτωση εικόνας σε όλες τις καταναλωτικές συσκευές. Αυτές τις αυξημένες ανάγκες έρχονται να καλύψουν οι υπηρεσίες τηλεπικοινωνιών τρίτης γενιάς και είναι αναμενόμενο το ενδιαφέρον για εφαρμογές εικόνας και video να συνεχίσει να αυξάνεται.

Εκτός όμως από τις παραπάνω καλά καθορισμένες και εύκολα μετρήσιμες εφαρμογές των DSPs η τεχνολογία τους έχει ενσωματωθεί σε μία πληθώρα άλλων ολοκληρωμένων κυκλωμάτων τα οποία έχουν μέσα τους βαθμίδες DSP αλλά δεν το δηλώνουν καθώς θέλουν να προβάλουν την ειδική λειτουργία την οποία επιτελούν. Για παράδειγμα αρκετά ολοκληρωμένα κυκλώματα για modems είναι κοινά DSPs προγραμματισμένα για να εκτελούν μία συγκεκριμένη λειτουργία και «λανσάρονται» ως ειδικά κυκλώματα υποστήριξης τηλεπικοινωνιών. Ένα ακόμα τρανταχτό παράδειγμα είναι η εταιρία Infineon Technologies AG η οποία αυτή την στιγμή είναι ο τρίτος μεγαλύτερος προμηθευτής DSPs για κινητά τηλέφωνα η οποία αναφέρει μηδενικές πωλήσεις DSPs και όλα αυτά τα προϊόντα τα κατηγοριοποιεί ως ASICs.

Επίσης οι περισσότεροι μικροεπεξεργαστές (π.χ. dsPIC της Microchip, Mega-AVR της Atmel, η τεχνολογία MMX των επεξεργαστών της Intel) έχουν ενσωματώσει βαθμίδες MAC και άλλες τεχνικές εμπνευσμένες από τα DSPs, για να αυξήσουν τις επιδόσεις τους και να είναι ανοιχτοί σε μεγαλύτερο εύρος εφαρμογών κυρίως αυτοματισμών, ρομποτικής και ελέγχου.

Φυσικά αναρίθμητες εφαρμογές βασίζονται σε chips που έχουν υλοποιημένες μέσα τους DSP βαθμίδες όπως για παράδειγμα κάθε συσκευή που αναπαράγει ή κωδικοποιεί δεδομένα σε μορφές MPEG (video και mp3) ή JPEG (φωτογραφίες). Επιπλέον πολλά προϊόντα βασίζονται σε ένα και μόνο ολοκληρωμένο για όλη τους την λειτουργία (system – on – chip) και φυσικά παρουσιάζονται ως ολοκληρωμένες πλατφόρμες και όχι ως DSPs με παραδείγματα τους ελεγκτές για συσκευές DVD, δέκτες ραδιοφώνου ή video, set-top boxes (συσκευές που συνδέονται στην τηλεόραση και δίνουν την δυνατότητα για σύνδεση στο internet και λήψη ψηφιακών προγραμμάτων Digital TV από δορυφορικούς ή επίγειους σταθμούς), επεξεργαστές εφαρμογών για κινητά τηλέφωνα (το

δεύτερο chip των κινητών τηλεφώνων που δεν ασχολείται με τις τηλεπικοινωνίες αλλά με την επικοινωνία με τον χρήστη), ψηφιακές κάμερες και φωτογραφικές μηχανές, υπολογιστές χειρός (palmtops) και τηλεοράσεις υψηλής πιστότητας (HDTV).

Στην βιομηχανία, πίσω σχεδόν από κάθε αισθητήρα, όπου χρειάζεται αξιόπιστη μέτρηση και έλεγχος, κρύβεται μία βαθμίδα DSP. Ακόμα και για την μέτρηση της ηλεκτρικής ενέργειας που καταναλώνει ένα εργοστάσιο χρησιμοποιούνται πλέον DSPs αντικαθιστώντας τα κλασικά όργανα περιορισμένης ακρίβειας με τους μαγνητικούς δίσκους, προσφέροντας χαμηλότερες τιμές αλλά και αυξημένες δυνατότητες όπως χρέωση από απόσταση αξιολόγηση της ποιότητας του ρεύματος και άλλα.

Επίσης εφαρμογές υψηλής ακρίβειας και εξαιρετικά απαιτητικές είναι οι ιατρικές. Μεγάλη έρευνα έχει γίνει ειδικά στις συσκευές ηλεκτροκαρδιογράφων όπου πολύ ασθενή σήματα χαμηλής συχνότητας από την καρδιά πρέπει να διαχωριστούν μέσα από πολύ θόρυβο. Ο κλάδος αυτός πραγματικά αναζωογονήθηκε με την εξάπλωση των DSPs αφού είναι πλέον σε θέση να προσφέρει οικονομικές λύσεις με ακρίβεια που δεν ήταν δυνατό να επιτευχθεί με τις παλιές αναλογικές τεχνικές.

Εντυπωσιακό είναι ότι παρόλο που η λειτουργικότητα των DSPs αυξάνει, οι μέση τιμή πώλησής τους συνεχίζει να πέφτει. Για παράδειγμα ένα τυπικό DSP το 1995 κόστιζε $12 ενώ το 2000 κόστιζε η τιμή ενός τυπικού DSP είχε πέσει στα μόλις $6 και σήμερα περίπου στα $4-5. Αυτό δεν συμβαίνει π.χ. στην αγορά των επεξεργαστών ηλεκτρονικών υπολογιστών όπου ένας τυπικός επεξεργαστής σταθερά κοστίζει περίπου $150 τα τελευταία χρόνια.

Είναι φανερό ότι, όπως είπαμε και στην αρχή αυτού του μέρους, οι διαχωριστικές γραμμές για το τι είναι DSP και τι δεν είναι, είναι πλέον δύσκολο να χαραχθούν με σαφήνεια. Η τεχνολογία των ψηφιακών επεξεργαστών σήματος αυτή την στιγμή έχει φέρει επανάσταση καταλαμβάνοντας την αγορά σε όλους τους κλάδους της. Καινούρια προϊόντα βασισμένα σε DSP αναπτύσσονται καθημερινά προσφέροντας εξαιρετικές δυνατότητες σε εξαιρετικά χαμηλές τιμές. Εν κατακλείδι, το συμπέρασμα είναι ότι τα DSPs βρίσκονται παντού, καθημερινά τα χρησιμοποιούμε και αρκετές φορές ούτε καν το γνωρίζουμε.

3.2 Το ADSP-BF533® της Analog Devices™

Το ADSP-BF533 είναι ένας πλήρης πυρήνας DSP και έχει επιπλέον ενσωματωμένα πολλά χρήσιμα περιφερειακά. Για την λεπτομερή μελέτη του συνόλου της λειτουργικότητας του πυρήνα και των περιφερειακών παραπέμπεταί κανείς στο εγχειρίδιο του επεξεργαστή[55]. Οι 936 σελίδες του προφανώς δεν αποσκοπούν στην ανάγνωση από την αρχή ως το τέλος αλλά στο να αποτελέσουν μία βάση αναφοράς για κάθε λεπτομέρεια της λειτουργίας του επεξεργαστή στην οποία μπορεί κανείς να ανατρέξει όποτε αντιμετωπίζει κάποιο πρόβλημα. Παρακάτω θα ακολουθήσει μία συνοπτική παρουσίαση όλων των δυνατοτήτων επεξεργαστή και των περιφερειακών του με περισσότερη έμφαση να δίνεται στα στοιχεία τα οποία χρησιμοποιούμε στην δική μας εφαρμογή ώστε να δίνεται ένα σαφές υπόβαθρο για την κατανόηση της λειτουργίας της.

3.2.1 Ο πυρήνας του επεξεργαστή

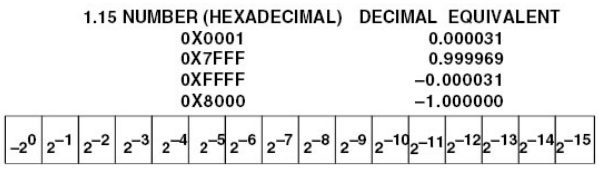
3.2.1.1 Γενικά χαρακτηριστικά

Το BF533® είναι ένας επεξεργαστής της σειράς BlackFin® της Analog Devices™. Ο επεξεργαστής αυτός συνδυάζει δύο βαθμίδες MAC και είναι έτσι ικανός να εκτελεί δύο πολλαπλασιασμούς και προσθέσεις ανά κύκλο. Επιπλέον με τις τέσσερις βοηθητικές video ALU μπορεί να εκτελεί ακόμα περισσότερες από τις παραπάνω πράξεις σε ένα κύκλο. (Οι video ALU μπορούν να εκτελούν κάποιες συγκεκριμένες πράξεις πάνω σε 8-bit δεδομένα. Τέτοιες πράξεις χρειάζονται σε αλγορίθμους κωδικοποίησης video π.χ. MPEG[56]). Η μέγιστη συχνότητα του πυρήνα είναι 600MHz και παράγεται από το ρόλοι περιφερειακών με τη βοήθεια του ενσωματωμένου PLL. Συνολικά λοιπόν ο επεξεργαστής παρουσιάζει μία επίδοση της τάξεως των 1.2 GMACs που είναι αρκετή για τις περισσότερες real-time εφαρμογές ήχου, εικόνας, data acquisition και ελέγχου.

Ο επεξεργαστής είναι βασικά 16-bitος αλλά αρκετοί καταχωρητές αντιμετωπίζονται ως 32-bites οντότητες. Επίσης αρκετές εντολές χειρίζονται δεδομένα σε ομάδες των 32-bit, των 16-bit ή και των 8-bit. Πάντως η αρχιτεκτονική του τον κατατάσσει στους 16-bitους επεξεργαστές.

Η αριθμητικές του ικανότητες περιορίζονται σε πράξεις σταθερής υποδιαστολής. Η ενσωματωμένη του όμως βαθμίδα barrel shifter κάνει (σύμφωνα με την κατασκευάστρια εταιρία) δυνατή την εξομοίωση – λειτουργία αριθμητικής κινητής υποδιαστολής με αρκετά ανταγωνιστική απόδοση. Παρόλα αυτά σε τεχνικά εγχειρίδια αναφέρεται ότι αυτό δε συνιστάται. Το συμπέρασμα είναι ότι αποτελεί απλά έναν γρήγορο fixed point επεξεργαστή.

Μπορεί να εκτελέσει πράξεις σε δεδομένα και να τα χειριστεί είτε ως απλά bits (Binary String – λογικές συναρτήσεις – OR, XOR, AND κ.τ.λ.) είτε ως προσημασμένους ή απρόσημους ακεραίους, είτε ως κλασματικούς αριθμούς στην μορφή 1.15, δηλαδή ένα ψηφίο προσήμου και 15 δεκαδικά ψηφία όπως φαίνεται στο παρακάτω σχήμα.



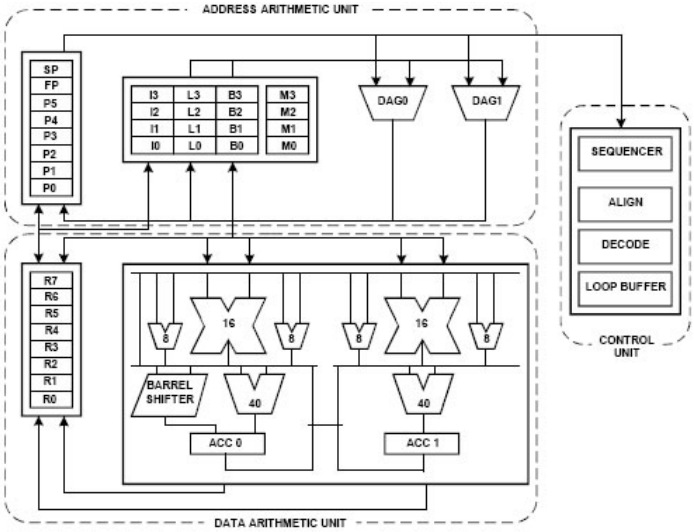
Εικόνα 48. Μορφή κλασματικών αριθμών 1.15

Στην ουσία όπως είπαμε σε προηγούμενη ενότητα οι κλασματικοί αριθμοί είναι απλά μία αναπαράσταση για τους προσημασμένους ακεραίους. Το σύνολο των αριθμητικών δυνατοτήτων του επεξεργαστή μπορεί να δει κανείς στο ανάλογο κεφάλαιο του manual[57]. Μία αναφορά θα

[55] ADSP-BF533 Blackfin™ Processor Hardware Reference
[56] ΕΛΕΚΤΟΡ, Απρίλιος 2004
[57] Σελίδα 2-16, ADSP-BF533 Blackfin Processor Hardware Reference

έπρεπε να γίνει και στον καταχωρητή ASTAT ο οποίος περιέχει πληροφορίες για τα αποτελέσματα των αριθμητικών πράξεων (υπόλοιπα, ένδειξη μηδέν κ.α.). Τέλος να σημειωθεί ότι αν και ο επεξεργαστής δεν υποστηρίζει την πράξη της διαίρεσης (η οποία είναι πολύ «ακριβή» σε επιφάνεια πυριτίου και επιπλέον χρησιμοποιείται σπάνια), έχει δύο εντολές (DIVS, DIVQ) που διευκολύνουν πολύ την υλοποίηση διαίρεσης προγραμματιστικά[58].

Ο επεξεργαστής χαρακτηρίζεται ως RISC–like[59] (Reduced Instruction Set Computer) που σημαίνει ότι έχουν γίνει προσπάθειες για να είναι σχετικά περιορισμένο το ρεπερτόριο εντολών σε ουσιώδεις απλές εντολές που εκτελούνται γρήγορα. Επιπλέον αρκετές εντολές του είναι SIMD (single instruction multiple data) που όπως είπαμε σε προηγούμενη ενότητα καταφέρνουν να κωδικοποιούν εντολές για την επεξεργασία πολλών δεδομένων σε μόνο μία γραμμή κώδικα. Οι απλές εντολές κωδικοποιούνται σε 16bit για να μπορούν να διαβάζονται γρήγορα ενώ οι σύνθετες DSP εντολές με πολλά ορίσματα κωδικοποιούνται σε 32bit.



Εικόνα 49. Διάγραμμα αρχιτεκτονικής πυρήνα

3.2.1.2 Βαθμίδα αριθμητικής

Στο παραπάνω διάγραμμα αρχιτεκτονικής του πυρήνα μπορούμε να παρατηρήσουμε το κάτω block που αποτελεί την βαθμίδα αριθμητικής δεδομένων. Αυτή περιλαμβάνει τις δύο βαθμίδες πολλαπλασιασμού με το σήμα X που έχουν εύρος 16-bit, τους δύο 40-bitους αθροιστές με το σήμα V, τις τέσσερις 8-bites video ALUs με τα μικρά σήματα V, τον Barrel Shifter, τους δύο καταχωρητές άθροισης (accumulators) με σύμβολα ACC 0 και ACC1 και το αρχείο καταχωρητών άμεσης προσπέλασης R0-R7. Οι πράξεις άθροισης μπορούν να γίνουν με στρογγυλοποίηση και με αποκοπή. Επιπλέον υποστηρίζεται η λειτουργία κορεσμού, για την ελαχιστοποίηση των συνεπειών μίας υπερχείλισης των αθροιστών. Εκτός από τις συνήθεις λογικές και αριθμητικές πράξεις, ο επεξεργαστής υποστηρίζει επιπλέον πράξεις για την επιτάχυνση ορισμένων λειτουργιών επεξεργασίας σήματος[60].

3.2.1.3 Βαθμίδα ελέγχου ροής

Στα δεξιά παρατηρούμε την βαθμίδα ελέγχου ροής, που περιλαμβάνει την βαθμίδα οργάνωσης (sequencer), τις βαθμίδες ευθυγράμμισης και αποκωδικοποίησης και την βαθμίδα επαναλήψεων. Με την βοήθεια της βαθμίδας ελέγχου ροής υλοποιούνται αλλαγές ροής (branches) προς μία συγκεκριμένη διεύθυνση (absolute), μία διεύθυνση σε σχέση με αυτή που βρίσκεται η εντολή (relative), έμμεση με την βοήθεια της τιμής κάποιου καταχωρητή (indirect) και κλήσεις

[58] Σελίδα 10-24, ADSP-BF53x Blackfin Processor Instruction Set Reference
[59] Webopedia: http://www.webopedia.com/TERM/R/RISC.html
[60] Σελίδα 1-3, ADSP-BF533 Blackfin Processor Hardware Reference

συναρτήσεων. Επιπλέον για την επιτάχυνση της εκτέλεσης βρόχων, πραγματοποιείται static branch prediction[61] που είναι η πιο εύκολη και οικονομική σε επιφάνεια πυριτίου λύση, δηλαδή η πραγματική πρόβλεψη για την επαλήθευση μίας συνθήκης γίνεται εξ' αρχής από τον προγραμματιστή ή τον compiler. Επιπλέον υποστηρίζεται το zero overhead looping δηλαδή η εκτέλεση βρόχων χωρίς καθυστερήσεις κατά την μετάβαση του προγράμματος από την μία εντολή στην άλλη. Ένα επίσης πολύ σημαντικό χαρακτηριστικό είναι ότι η αρχιτεκτονική αύτη έχει πλήρη έλεγχο εξαρτήσεων δεδομένων, δηλαδή η λειτουργία pipelining λειτουργεί πλήρως βασισμένη στο hardware χωρίς να χρειάζεται μέριμνα κατά τον προγραμματισμό (εκτός φυσικά από τις καλές τακτικές για αποφυγή καθυστερήσεων που αναφέραμε σε προηγούμενη ενότητα).

Η pipeline του Blackfin χωρίζει την κάθε εντολή σε 10 βήματα. Με αυτόν τον τρόπο μπορεί να εκτελεί ακόμα και τις πιο σύνθετες εντολές σε (φαινομενικά) μόνο ένα κύκλο.

Βήμα	Περιγραφή
Instruction Fetch 1 (IF1)	Ξεκινά η πρόσβαση στην μνήμη προγράμματος
Instruction Fetch 2 (IF2)	Συνεχίζει η ανάγνωση της εντολής από την μνήμη προγράμματος
Instruction Fetch 3 (IF3)	Ολοκλήρωση της ανάγνωσης της εντολής από την μνήμη
Instruction Decode (DEC)	Ευθυγράμμιση εντολής έναρξη αποκωδικοποίησης εντολής και πρόσβαση στο αρχείο καταχωρητών δεικτών
Address Calculation (AC)	Υπολογισμός διευθύνσεων δεδομένων και αλλαγής ροής
Execute 1 (EX1)	Έναρξη πρόσβασης στην μνήμη δεδομένων
Execute 2 (EX2)	Ανάγνωση από το αρχείο καταχωρητών
Execute 3 (EX3)	Ολοκλήρωση πρόσβασης στην μνήμη δεδομένων και έναρξη εκτέλεσης των εντολών δύο κύκλων
Execute 4 (EX4)	Εκτέλεση των εντολών ενός κύκλου
Write Back (WB)	Εγγραφή των αποτελεσμάτων στους καταχωρητές δεδομένων και δεικτών και επεξεργασία συμβάντων

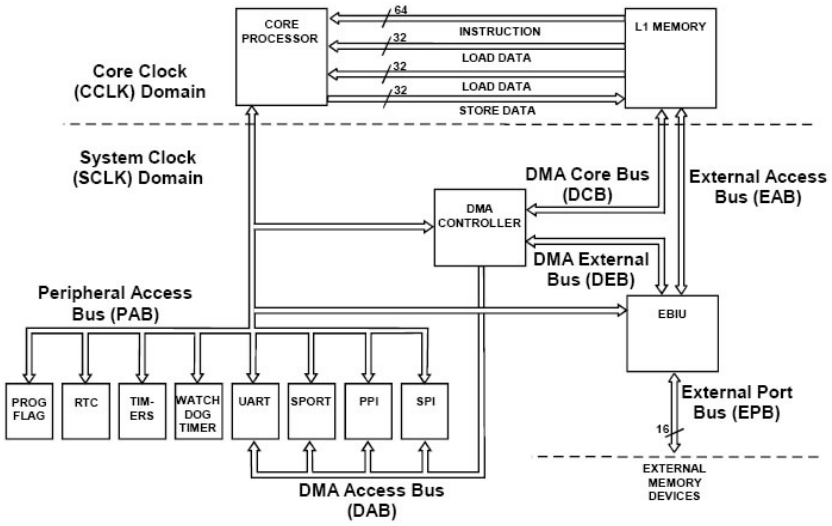
3.2.1.4 Βαθμίδα δημιουργίας διευθύνσεων

Στο πάνω μέρος του διαγράμματος βλέπουμε την βαθμίδα δημιουργίας διευθύνσεων δεδομένων. Αυτή δημιουργεί διευθύνσεις για τις δύο μονάδες MAC ώστε να μπορούν με μία κίνηση να επεξεργαστούν ζεύγη δεδομένων. Οι διάφοροι καταχωρητές δίνουν δυνατότητες δημιουργίας διευθύνσεων με πολλούς τρόπους κάνοντας εύκολη την γρήγορη υλοποίηση δομών όπως στοίβες ή κυκλικούς buffers. Υποστηρίζει επίσης την διευθυνσιοδότηση ανάστροφου bit για τη γρήγορη υλοποίηση FFT μετασχηματισμών.

3.2.1.5 Αρχιτεκτονική και διαχείριση μνήμης

Ο επεξεργαστής έχει αρχιτεκτονική διαχείρισης μνήμης τύπου Modified Harvard Architecture[62]. Αυτό σημαίνει το σύστημα έχει μία μόνο ενιαία εξωτερική μνήμη που αποθηκεύει το πρόγραμμα και τα δεδομένα αλλά εσωτερικά στον επεξεργαστή, το πρόγραμμα και τα δεδομένα αποθηκεύονται σε ξεχωριστές διαμερίσεις της εσωτερικής γρήγορης μνήμης (cache). Αυτή είναι μία από τις πλέον δεδομένες τεχνικές στα DSPs αφού προσφέρει μεγάλη ταχύτητα εκτέλεσης – διαμεταγωγής δεδομένων σε συνδυασμό με την οικονομία ενός μόνο ελεγκτή μνήμης.

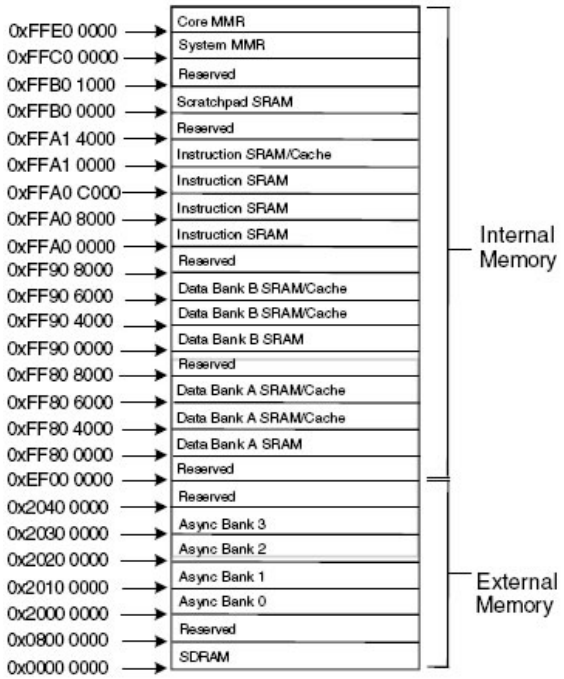
[61] http://www.pattosoft.com.au/jason/Papers/ValueRangeProp/
[62] http://www.cs.tcd.ie/Michael.Brady/ISD/MSc%20System%20Design%201.ppt



Εικόνα 50. Οι μνήμη του Blackfin® και οι δίαυλοι ροής δεδομένων

Πιο συγκεκριμένα η πρώτου επιπέδου μνήμη (L1 cache) τρέχει στην συχνότητα ρολογιού του πυρήνα και είναι χωρισμένη σε τρεις περιοχές, την περιοχή αποθήκευσης προγράμματος και δύο περιοχές αποθήκευσης δεδομένων. Υποστηρίζεται με τέσσερις διαύλους επικοινωνίας με τον πυρήνα όπως φαίνεται στο παραπάνω σχήμα, μία εύρους 64ων bit για την ανάγνωση εντολών, δύο εύρους 32 bit για την ανάγνωση δεδομένων και μία 32 bit για την αποθήκευση δεδομένων πίσω στην μνήμη.

Εκτός από την L1 cache, το BF533® μπορεί να διασυνδεθεί με εξωτερικές μνήμες διαφόρων τύπων μεταξύ των οποίων Flash (για πρόγραμμα) και SRAM (για δεδομένα και πρόγραμμα) μέσω του External Bus Interface Unit (EBIU)[63], μίας μονάδας οδηγού μνήμης που υποστηρίζει διάφορους τύπους εξωτερικής μνήμης. Γενικά ο δίαυλός με τον οποίο επικοινωνεί με τις μνήμες είναι εύρους 32-bit πράγμα το οποίο σημαίνει ότι έχει πρόσβαση σε 4GBytes διευθύνσεων. Στην πράξη μέσω αυτών μπορεί να διασυνδεθεί με το πολύ 128Mb εξωτερικής μνήμης SRAM η οποία όμως είναι αρκετή για τις περισσότερες εφαρμογές. Ευτυχώς για εμάς οι περισσότερες λειτουργίες της μνήμης εξυπηρετούνται πλήρως από το hardware χωρίς να χρειάζεται να μεριμνήσουμε παρά μόνο με κάποιες εντολές αρχικοποίησης.



Εικόνα 51. Χάρτης μνήμης BF533®

Μέσω διευθύνσεων μνήμης γίνεται και η επικοινωνία του επεξεργαστή με τα περιφερειακά που βρίσκονται μέσα στο ίδιο το chip όπως και περιφερειακά του συστήματος αν έχουν αυτή τη δυνατότητα. Για παράδειγμα, αν θέλει κανείς να ρυθμίσει την περιοδικότητα ενός από τους χρονιστές που περιέχεται στο chip, αρκεί να γράψει την επιθυμητή περίοδο στην κατάλληλη θέση μνήμης. Οι καταχωρητές αυτοί ονομάζονται καταχωρητές χαρτογραφημένοι στην μνήμη (Memory Mapped Registers – MMRs) και βρίσκονται στις υψηλότερες θέσεις μνήμης του επεξεργαστή όπως φαίνεται στην παραπάνω εικόνα.

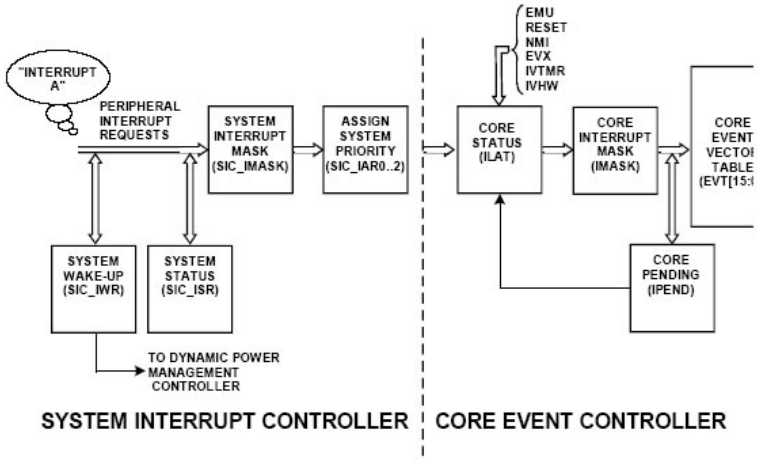
Μία επιπλέον δυνατότητα που δίνει ο επεξεργαστής αυτός είναι το κλείδωμα διαφόρων περιοχών μνήμης. Με αυτόν τον τρόπο διασφαλίζεται ότι κάποιο πρόγραμμα με λάθη δεν θα γράψει σε

[63] Σελίδα 17-1, ADSP-BF533 Blackfin Processor Hardware Reference

περιοχές της μνήμης που μπορεί να βλάψουν τον επεξεργαστή, το πρόγραμμά του ή δεδομένα ανώτερου επιπέδου ασφαλείας π.χ. δεδομένα λειτουργικού συστήματος. Συγκεκριμένα ο επεξεργαστής διαθέτει τρεις καταστάσεις λειτουργίας. Την κατάσταση χρήστη, την κατάσταση επιτηρητή και την κατάσταση εξομοίωσης. Στην κατάσταση χρήστη που είναι η πιο συνήθης για εκτέλεση προγράμματος, δεν είναι δυνατή η πρόσβαση σε προστατευμένες περιοχές της μνήμης και σε καταχωρητές που μπορούν να βλάψουν την ομαλή λειτουργία του συστήματος.

3.2.1.6 Διαχείριση συμβάντων

Τα συμβάντα (events) χωρίζονται σε δύο μεγάλες κατηγορίες, τα σύγχρονα και τα ασύγχρονα. Τα σύγχρονα προκαλούνται εξ' αιτίας κάποιας εντολής είτε σκόπιμα από τον χρήστη με την ειδική εντολή RAISE (software interrupts), είτε εξ' αιτίας σφάλματος κατά την εκτέλεση κάποιας εντολής (exceptions) π.χ. προσπάθεια πρόσβασης σε μνήμη που δεν επιτρέπεται[64]. Τα ασύγχρονα (interrupts) προκαλούνται από κάποιο περιφερειακό είτε από κάποια ακίδα του επεξεργαστή (Reset – NMI). Η διαχείριση των συμβάντων είναι αρκετά σύνθετη μιας και διάφορα events έχουν διαφορετική προτεραιότητα και επιτρέπεται η ταυτόχρονη εξυπηρέτηση περισσοτέρων του ενός συμβάντων ανάλογα με την προτεραιότητά τους.



Εικόνα 52. Ελεγκτές διαχείρισης συμβάντων

Γι' αυτό τον λόγο χρησιμοποιούνται δύο ανεξάρτητοι ελεγκτές (Εικόνα 52), ο ελεγκτής συμβάντων πυρήνα (CEC) που διαχειρίζεται συμβάντα του πυρήνα και έμμεσα των περιφερειακών και ο ελεγκτής διακοπών συστήματος (SIC) που διαχειρίζεται διακοπές των περιφερειακών και τις μεταφέρει σύμφωνα με τον προγραμματισμό του στα 7 λιγότερο σημαντικά συμβάντα του πυρήνα.

Ελεγκτής Πυρήνα		Συμβάν
EMU		Emulation (highest priority)
RST		Reset
NMI		NMI
EVX		Exception
IVHW		Hardware Error
IVTMR	Ελεγκτής Συστήματος	Core Timer
IVG7		PLL Wakeup Interrupt, DMA Error (generic), PPI Error Interrupt, SPORT0 Error Interrupt, SPORT1 Error Interrupt, SPI Error Interrupt, UART Error Interrupt
IVG8		Real-Time Clock interrupts, DMA0 Interrupt (PPI)
IVG9		DMA1 Interrupt (SPORT0 RX), DMA2 Interrupt (SPORT0 TX), DMA3 Interrupt (SPORT1 RX), DMA4 Interrupt (SPORT1 TX)
IVG10		DMA5 Interrupt (SPI), DMA6 Interrupt (UART RX), DMA7 Interrupt (UART TX)
IVG11		DMA5 Interrupt (SPI), DMA6 Interrupt (UART RX), DMA7 Interrupt (UART TX)
IVG12		Programmable Flags Interrupt A/B
IVG13		DMA8/9 Interrupt (Memory DMA Stream 0), DMA10/11 Interrupt (Memory DMA

[64] Table 4-11. Events That Cause Exceptions. BF533 Hardware manual.

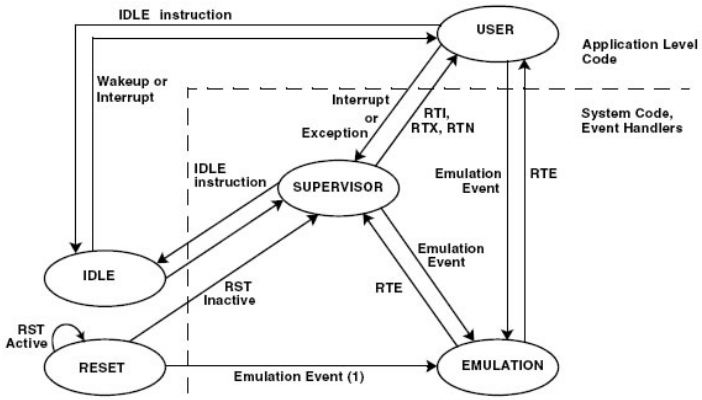
Στον παραπάνω πίνακα φαίνεται η δρομολόγηση των συμβάντων και η προτεραιότητα τους. Είναι σημαντικό ότι στα interrupts των περιφερειακών μπορεί να δοθεί διαφορετική προτεραιότητα και να «συνδεθούν» με τα interrupts πυρήνα (IVG7-12) με διαφορετικό τρόπο αλλάζοντας τις τιμές των καταχωρητών SIC_IAR0..2 όπως φαίνεται και από την Εικόνα 52. Πρέπει να σημειωθεί ότι τα interrupt flags (SIC_ISR) των περιφερειακών πρέπει να καθαρίζονται από το πρόγραμμα στην ρουτίνα εξυπηρέτησης interrupt αν δεν θέλουμε η τελευταία να επαναλαμβάνεται για πάντα. Στις θέσεις μνήμης με ονόματα EVT0-15 υπάρχουν οι διευθύνσεις των ρουτινών εξυπηρέτησης γεγονότων. Όταν ένα γεγονός πυρήνα ενεργοποιηθεί το πρόγραμμα θα «καλέσει» την συνάρτηση στην διεύθυνση μνήμης που έχει καταχωρηθεί στο EVTx που αντιστοιχεί στο γεγονός.

Για να λάβει κανείς ένα interrupt από ένα περιφερειακό (που είναι η πιο σύνθετη περίπτωση) πρέπει:

- Να ενεργοποιήσει το interrupt αυτό από το περιφερειακό. Το πως γίνεται αυτό εξετάζεται σε κάθε περιφερειακό ξεχωριστά.
- Να επιτρέψει το interrupt αυτό από τον ελεγκτή συστήματος μέσω του καταχωρητή SIC_IMASK.
- Να δρομολογήσει το interrupt αυτό στον ελεγκτή πυρήνα με την ανάλογη ρύθμιση ενός καταχωρητή SIC_IARx
- Να επιτρέψει το interrupt αυτό στον ελεγκτή πυρήνα μέσω του καταχωρητή IMASK
- Να γράψει τη διεύθυνση για την ρουτίνα εξυπηρέτησης στον αντίστοιχο καταχωρητή EVTx

3.2.1.7 Καταστάσεις λειτουργίας

Όπως είπαμε προηγουμένως υπάρχουν τρεις καταστάσεις λειτουργίας. Τώρα που είδαμε πως εργάζονται τα συμβάντα, μπορούμε να καταλάβουμε καλύτερα τις μεταβάσεις μεταξύ τους.



Εικόνα 53. Οι καταστάσεις λειτουργίας του επεξεργαστή και οι μεταβάσεις μεταξύ τους

Στο παραπάνω σχήμα, εκτός από τις καταστάσεις χρήστη (user), επιτηρητή (supervisor) και εξομοίωσης (emulation), βλέπουμε και τις καταστάσεις στασιμότητας (idle) και επανεκκίνησης (reset). Συνοπτικά το παραπάνω διάγραμμα μας λέει τα εξής:

1. Όταν ξεκινάει ο επεξεργαστής ή ενεργοποιηθεί η είσοδός reset, ο επεξεργαστής μεταβαίνει στο reset mode.
2. Μετά την εκτέλεση του κώδικα αρχικοποίησης, μεταβαίνει στο supervisor mode όπου (λογικά) γίνονται αρχικοποιήσεις των μεταβλητών του συστήματος και πιθανώς του λειτουργικού συστήματος.
3. Όταν τελειώσουν οι λειτουργίες του supervisor mode, ο επεξεργαστής μεταβαίνει στο user mode όπου πλέον μπορεί να εκτελεί προστατευμένο κώδικα χρήστη. Η κατάσταση αυτή είναι η πιο ασφαλής γιατί προστατεύει περιοχές μνήμης και πρόσβαση σε συγκεκριμένα περιφερειακά.

4. Σε περίπτωση που κάποια exception συμβεί π.χ. προσπάθεια πρόσβασης σε απαγορευμένη μνήμη, ο επεξεργαστής μεταβαίνει από το user mode στο supervisor mode για να «μαζέψει τα σπασμένα» και να επαναφέρει το σύστημα σε κάποια ομαλή κατάσταση λειτουργίας.
5. Επίσης σε περίπτωση που συμβεί κάποιο interrupt και πάλι μεταβαίνει στην κατάσταση supervisor για να εξυπηρετηθεί η ρουτίνα του interrupt.
6. ΣΕ ΚΑΘΕ ΠΕΡΙΠΤΩΣΗ εάν υπάρξει το συμβάν εξομοίωσης (το οποίο έχει μέγιστη προτεραιότητα) ο επεξεργαστής μεταβαίνει στην κατάσταση εξομοίωσης. Αυτή η κατάσταση είναι πάρα πολύ χρήσιμη για τον εντοπισμό σφαλμάτων τόσο στον κώδικα που εκτελείται σε supervisor mode (λειτουργικό) όσο και στον κώδικα χρήστη που εκτελείται σε user mode. Από την κατάσταση εξομοίωσης ο επεξεργαστής επιστρέφει στην κατάσταση που βρισκόταν προηγουμένως με την εντολή RTE.
7. Τέλος, στις καταστάσεις user και supervisor μπορεί με την βοήθεια της εντολής IDLE να μεταβεί ο επεξεργαστής στην κατάσταση στασιμότητας (idle mode) όπου καταναλώνει λιγότερο ρεύμα. Από αυτή την κατάσταση ο επεξεργαστής επιστρέφει στην κανονική λειτουργία (ξυπνάει) αν συμβεί κάποιο interrupt.

Σχολιάζοντας τα παραπάνω θα έπρεπε να σημειώσουμε ότι η ύπαρξη των supervisor και user modes είναι απαραίτητη για τη δημιουργία λειτουργικών συστημάτων. Αν παρόλα αυτά το μία εφαρμογή δεν χρειάζεται λειτουργικό σύστημα, μπορεί όλο το πρόγραμμα να εκτελείται στο supervisor mode, ώστε να έχει πλήρη πρόσβαση σε όλα τα resources του συστήματος.

3.2.1.8 Λειτουργίες εξοικονόμησης ενέργειας

Για την εξοικονόμηση ενέργειας ο επεξεργαστής έχει την δυνατότητα ρύθμισης τόσο της συχνότητας όσο και της τάσης λειτουργίας (του πυρήνα) καταφέρνοντας με αυτόν τον τρόπο να προσφέρει ευέλικτες λύσεις στις απαιτήσεις μας για μεγάλη υπολογιστική ικανότητα και μικρή κατανάλωση ρεύματος. Το κυριότερο είναι ότι όλα αυτά μπορούν να αλλάζουν δυναμικά μέσα από το πρόγραμμα με την βοήθεια του ελεγκτή δυναμικής διαχείρισης ενέργειας (Dynamic Power Management Controller). Δεν θα αναφερθούμε λεπτομερειακά στον DPMC. Όποιος θέλει περισσότερες λεπτομέρειες μπορεί να βρει στο manual[65]. Θα αρκεστούμε να πούμε ότι δίνει την δυνατότητα λειτουργίας σε τέσσερις καταστάσεις, Full On, Active, Sleep, Deep Sleep με ανάλογες δυνατότητες εξοικονόμησης ενέργειας.

Στις καταστάσεις ύπνου δεν γίνεται κανένας υπολογισμός. Ίσως μία τέτοια λειτουργία να φαίνεται χαζή αλλά για παράδειγμα σε ένα κινητό τηλέφωνο, το DSP δεν χρειάζεται να κάνει τίποτα για την περισσότερη ώρα, όταν ο χρήστης δεν μιλάει. Πολύ προσοχή στην κατάσταση Deep Sleep. Τα περιεχόμενα της εξωτερικής μνήμης είναι πιθανόν να σβήσουν (αν η λειτουργία της εξαρτάται από χρονισμό που παράγει ο επεξεργαστής).

Πάρα πολύ σημαντικό ρόλο σε αυτή τη λειτουργία παίζει το PLL (Phase Locked Loop) με την βοήθεια του οποίου ο επεξεργαστής παράγει τα σήματα χρονισμού για τον πυρήνα και τα περιφερειακά του, ως πολλαπλάσια του εξωτερικού σήματος ρολογιού. Το PLL δίνει πολλές δυνατότητες δημιουργίας σήματος χρονισμού αλλά θέλει και αρκετή προσοχή στην ρύθμισή του και κατά τη δυναμική αλλαγή παραμέτρων του.

Συγκεκριμένα θα πρέπει οι συχνότητες χρονισμού που δημιουργούνται να ΜΗΝ ξεπερνούν τις μέγιστες επιτρεπτές για ΟΛΑ τα περιφερειακά που οδηγούν. Για παράδειγμα το system clock παρουσιάζεται στην ακίδα clock out του επεξεργαστή με σκοπό να οδηγεί τις εξωτερικές μνήμες του συστήματος. Αν αυτές έχουν μέγιστη συχνότητα χρονισμού τα 100MHz, το system clock δεν θα πρέπει να ξεπερνάει την τιμή αυτή και σε καμία περίπτωση τα 133MHz που είναι το ανώτατο όριο για τα περιφερειακά του ίδιου του επεξεργαστή. Επιπλέον κατά την αλλαγή των ρυθμίσεων πρέπει να δίνεται πάρα πολύ προσοχή μιας και όταν αλλάζει ο λόγος διαίρεσης του PLL παίρνει κάποιο χρόνο μέχρι το PLL να «κλειδώσει» στην νέα συχνότητα και πρέπει να ακολουθηθεί ειδική διαδικασία ώστε να έχουμε ομαλή μετάβαση. Επιπλέον κάποια περιφερειακά μπορεί να έχουν

[65] Σελίδα 8-13, ADSP-BF533 Blackfin Processor Hardware Reference

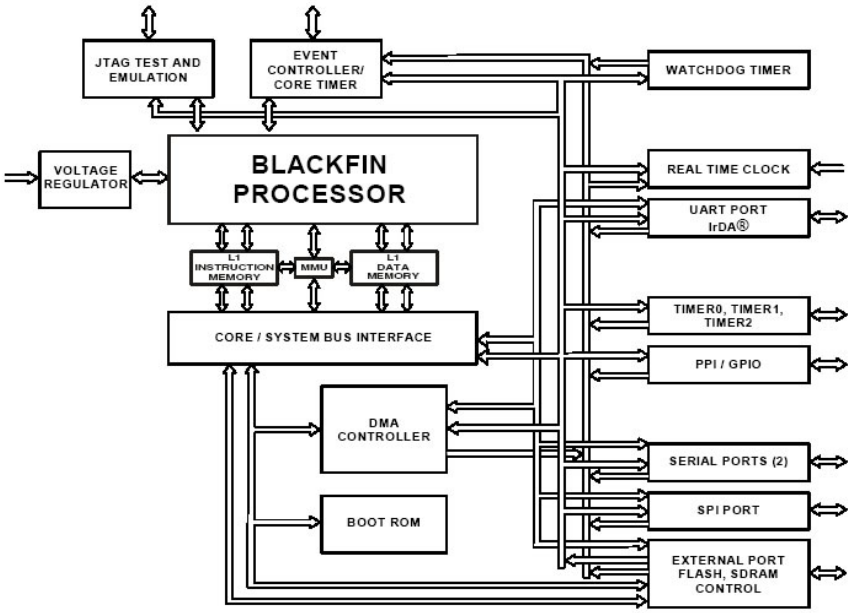
απώλειες δεδομένων κατά τη δυναμική αλλαγή συχνότητας. Συμπεραίνουμε λοιπόν ότι θέλει αρκετή προσοχή στις ρυθμίσεις των συχνοτήτων.

Για παράδειγμα, αν έχουμε ένα εξωτερικό σήμα ρολογιού στα 27MHz και μέγιστη συχνότητα μνήμης 133MHz (όπως στο EZ-KIT Lite™) και θέλουμε μέγιστη απόδοση (λειτουργία πυρήνα στα 600MHz, συστήματος στα 133MHz, απόδοση 1200MMACs), με την ελάχιστη δυνατή κατανάλωση, πρέπει να βάλουμε το PLL να πολλαπλασιάζει τη συχνότητα επί 22 (22 x 27 = 594MHz) με ρυθμίσεις[66] DF = 1, MSEL = 44, τον διαιρέτη συχνότητας πυρήνα στην μονάδα CSEL = 0 και τον διαιρέτη συστήματος σε διαίρεση δια 5 (594 / 5 = 112MHz) SSEL = 5. Φυσικά όσα περιφερειακά δεν χρησιμοποιούνται θα πρέπει να παραμείνουν απενεργοποιημένα για να μην καταναλώνουν ενέργεια. Ο τρόπος απενεργοποίησης περιγράφεται ξεχωριστά για κάθε περιφερειακό στις ανάλογες σελίδες του manual. Ο Dynamic Power Management Controller απενεργοποιεί αυτόματα και το ρολόι για τα περιφερειακά που είναι απενεργοποιημένα για μέγιστη εξοικονόμηση ενέργειας.

[66] Σελίδα 8-4: Table 8-1, 8-2, 8-3, ADSP-BF533 Blackfin Processor Hardware Reference

3.2.2 Τα περιφερειακά του επεξεργαστή

Κάπου εδώ θα σταμάταγε η εξέταση του επεξεργαστή μία με δύο δεκαετίες πριν. Μέχρι τότε τα DSPs ήταν ένας γρήγορος επεξεργαστικός πυρήνας με αυξημένες δυνατότητες εκτέλεσης αριθμητικών πράξεων και αρκετή cache. Όπως αναλύσαμε όμως σε προηγούμενη ενότητα, η σύγχρονη τακτική είναι να ενσωματώνονται επιπλέον ένα πλήθος περιφερειακών ώστε να γίνεται δυνατή η ολοκλήρωση πλήθους εφαρμογών με μόνο το chip του DSP, κάποια εξωτερική μνήμη και πιθανώς κάποια A/Ds - D/As. Είναι ακριβώς αυτά τα χαρακτηριστικά που κάνουν τους κόσμους των DSP και των μικροελεγκτών γενικής χρήσης να συγκλίνουν χρόνο με τον χρόνο.



Εικόνα 54. Ο επεξεργαστής και τα περιφερειακά του

Στο παραπάνω σχήμα μπορούμε να παρατηρήσουμε ότι το BF533® εκτός από τον δυνατό πυρήνα που εξετάσαμε έχει μία πληθώρα περιφερειακών που επικοινωνούν με τον πυρήνα και μεταξύ τους με αντίστοιχους διαύλους. Τα περιφερειακά αυτά είναι:

1. Ελεγκτής DMA	Μονάδα μεταφοράς δεδομένων ανεξάρτητα από τον πυρήνα
2. PPI	Παράλληλη θύρα με πολύ γρήγορη ροή δεδομένων (video)
3. 2 σειριακές θύρες	Σειριακές θύρες με γρήγορη ροή δεδομένων (ήχος)
4. Θύρα SPI	Σειριακή θύρα μικρής ταχύτητας για έλεγχο περιφερειακών
5. UART	Ασύγχρονη σειριακή θύρα για επικοινωνία π.χ. με PC – IrDA®
6. GPIO	Γενικής χρήσης I/O με αυξημένες δυνατότητες ελέγχου
7. 3 Χρονιστές	Χρονιστές γενικής χρήσης με αυξημένες δυνατότητες
8. Χρονιστής επιτήρησης	Κάνει αυτόματα reset αν το πρόγραμμα «κολλήσει»
9. Χρονιστής πυρήνα	Χρονιστής αφιερωμένος στον πυρήνα
10. Real time clock	Ρολόι πραγματικού χρόνου με δυνατότητα alarms
11. EBIU	Ελεγκτής εξωτερικών μνημών
12. Διεπαφή JTAG	Διασύνδεση για τον έλεγχο επεξεργαστή και προγράμματος
13. Σταθεροποιητής τάσης	Δημιουργεί τάσεις για τον πυρήνα
14. Boot rom	Μνήμη με πρόγραμμα που εκτελείται κατά την εκκίνηση

Θα εξετάσουμε σύντομα τα περισσότερα περιφερειακά και θα δώσουμε ιδιαίτερη έμφαση στο Parallel Peripheral Interface (PPI) και στον DMA controller που χρησιμοποιούνται ιδιαίτερα στην εφαρμογή μας. Ειδικά ο δεύτερος έχει εξέχουσα θέση στο σύνολο των εφαρμογών των DSPs και του BlackFin® ειδικότερα.

3.2.2.1 Ο ελεγκτής DMA

Όπως αναφέραμε σε προηγούμενη ενότητα ο ελεγκτής DMA αναλαμβάνει να κάνει μαζικές μεταφορές δεδομένων μεταξύ των περιφερειακών και της μνήμης χωρίς να σπαταλάει επεξεργαστική ισχύ του επεξεργαστή. Υπάρχουν 12 ανεξάρτητα DMA κανάλια, 8 εκ των οποίων εξυπηρετούν τα περιφερειακά και 4 την μνήμη. Ο ελεγκτής DMA του BF533® μπορεί να κάνει διάφορους τύπους μεταφοράς δεδομένων:

1. Από μνήμη σε μνήμη (Memory DMA - MDMA)
2. Μεταξύ μνήμης και του σειριακού interface (SPI)
3. Μεταξύ μνήμης και των δύο σειριακών θυρών (SPORT)
4. Μεταξύ μνήμης και ασύγχρονης σειριακής θύρας (UART)
5. Μεταξύ μνήμης και παράλληλου interface διασύνδεσης περιφερειακών (PPI)

Υπάρχουν δύο τύποι μεταφοράς DMA, η μεταφορά βασισμένη σε δομές περιγραφής και η μεταφορά βασισμένη σε καταχωρητές.

- Η μεταφορά βασισμένη σε καταχωρητές είναι η πιο απλή όπου ο προγραμματιστής απλά γράφει στους καταχωρητές του καναλιού DMA την περιγραφή της λειτουργίας που θέλει και ο ελεγκτής DMA αναλαμβάνει να εκτελέσει την εργασία.

- Η μεταφορά βασισμένη σε δομές περιγραφής είναι πιο σύνθετη αλλά δίνει πολύ περισσότερες επιλογές. Κατά τη λειτουργία αυτή ο προγραμματιστής δίνει στον ελεγκτή DMA μία λίστα από περιγραφές για τις μεταφορές που θέλει να γίνουν και ο ελεγκτής τις εκτελεί με την σειρά. Για παράδειγμα αν θέλουμε να μεταφέρουμε ένα πλαίσιο video χρησιμοποιώντας ελάχιστη μνήμη μπορούμε να χρησιμοποιήσουμε περιγραφές για το που βρίσκονται διάφορα κομμάτια του οι διάφοροι τύποι οριζόντιου και κατακόρυφου συγχρονισμού και η ωφέλιμη εικόνα.

Επίσης δίνεται η δυνατότητα λειτουργίας σε δισδιάστατο μοντέλο, πράγμα το οποίο θα μπορούσε να συμβολιστεί με την παρακάτω αναπαράσταση κώδικα.

```
pointer = START;
for (y=0; y < Y_COUNT; y++) {
   for (x=0; x < X_COUNT; x++) {
      …
      pointer += X_MODIFY
   }
   pointer += Y_MODIFY
}
```

Ένα ωραίο παράδειγμα χρήσης της λειτουργικότητας αυτής είναι η χρησιμοποίησή της για τον διαχωρισμό μίας εικόνας NxM σημείων σε μορφή RGB σε τρεις NxM εικόνες, μία για το κάθε χρώμα. Οι ρυθμίσεις είναι οι εξής:

```
X_MODIFY = (N*M)
X_COUNT = 3
Y_MODIFY = 1 - 2(N*M)
Y_COUNT = (N*M)
```

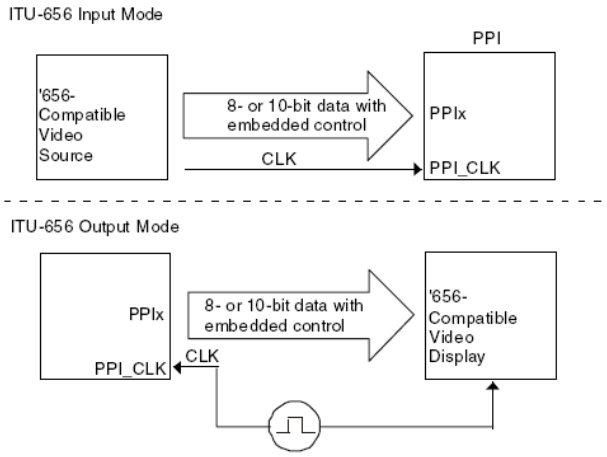
Παρατηρήστε ότι το `Y_MODIFY` είναι αρνητικό. Αυτό θα παράγει την ακολουθία δεικτών:

```
0,              (N*M),          2*( N*M),
1,              (N*M) +1,       2*( N*M) +1,
2,              (N*M) +2,       2*( N*M) + 2,
...
(N*M) - 1,      2*(N*M)  - 1,   3*( N*M) - 3,
0,              (N*M),          ...
```

Είναι σημαντικό να διαβάσει κανείς στο manual του επεξεργαστή τις ειδικές διαδικασίες που χρειάζονται για το ξεκίνημα και την λήξη μεταδόσεων DMA όπως και του χειρισμού των interrupts.

3.2.2.2 Parallel Peripheral Interface

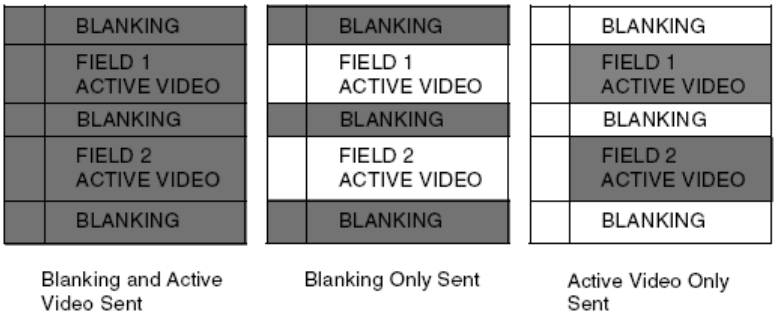
Το PPI είναι μία Half Duplex παράλληλη μονάδα επικοινωνίας. Αυτό σημαίνει ότι μπορεί είτε να λαμβάνει είτε να αποστέλλει δεδομένα από/προς την εξαρτώμενη συσκευή αλλά όχι και τα δύο ταυτόχρονα, γεγονός που αποτελεί ένα πρόβλημα για την δικιά μας εφαρμογή. Με την βοήθεια του PPI είναι δυνατή η μεταφορά πολύ μεγάλων ποσοτήτων δεδομένων όπως για παράδειγμα Video. Για την υποστήριξη μεταφοράς Video σύμφωνα με το πρωτόκολλο BT 656 που περιγράψαμε στην ενότητα 2.2.3.2, το PPI έχει αρκετές βοηθητικές λειτουργίες. Υποστηρίζει αποστολή και λήψη video με εύρος λέξης από 8 έως 10-bit όπως φαίνεται στην Εικόνα 55. Παρατηρούμε ότι και στις δύο περιπτώσεις πρέπει να παρέχεται στον PPI το εξωτερικό σήμα χρονισμού PPI_CLK με βάση το οποίο συγχρονίζεται επικοινωνία.



Εικόνα 55. Αποστολή και λήψη video με τη βοήθεια του PPI

Φυσικά δεν θα είχε νόημα να απασχολείται ο επεξεργαστής για κάθε byte που μεταφέρεται μέσα από αυτό το περιφερειακό. Το κανάλι DMA αναλαμβάνει την μεταφορά δεδομένων video από και προς την μνήμη. Το PPI διαθέτει διάφορους τρόπους με τους οποίους μπορεί να φιλτράρει την «ωφέλιμη» πληροφορία που εγγράφεται στην μνήμη.

- Μπορεί να αγνοεί τα μονά ή τα ζυγά δεδομένα. Με αυτόν τον τρόπο μπορεί να λαμβάνει μόνο την πληροφορία χρώματος (Cr-Cb) ή την πληροφορία φωτεινότητας (Y).
- Μπορεί να επιλέγει μόνο το Frame 1 αν δεν χρειαζόμαστε και τα δύο frames.



Εικόνα 56. Επιλεκτική λειτουργία λήψης video

- Μια επιπλέον πολύ χρήσιμη λειτουργία είναι αυτή που φαίνεται στην Εικόνα 56. Ο ελεγκτής μπορεί αυτόματα να επιλέγει την λήψη όλου του frame, μόνο της πληροφορίας κατακόρυφου συγχρονισμού και μόνο της ενεργούς περιοχής που έχει πληροφορία εικόνας των δύο Frames.

Με την βοήθεια των παραπάνω λειτουργιών είναι δυνατόν να περιοριστεί σημαντικά η ποσότητα της πληροφορίας που μεταφέρεται μέσα από το κανάλι DMA και να γίνει και η αντίστοιχη οικονομία μνήμης.

Πρέπει επίσης να αναφέρουμε την λειτουργία του καταχωρητή PPI_FRAME. Αυτός πρέπει να έχει την τιμή 625 για PAL και 525 για NTSC, δηλαδή τον αριθμό των γραμμών ανά Frame. Αν προκύψει ολοκλήρωση του Frame πριν ληφθεί ο αριθμός αυτός γραμμών τότε ένα σήμα σφάλματος ενεργοποιείται ενημερώνοντας για το πρόβλημα.

Εκτός φυσικά από τους παραπάνω τρόπους ειδικούς για Video το PPI μπορεί να χρησιμοποιηθεί για απλή παράλληλη μεταφορά δεδομένων εύρους 8 έως 16-bit. Για άνω των 8-bit λειτουργία δεσμεύονται γενικής χρήσης ακίδες (ενότητα 3.2.2.6).

3.2.2.3 Σειριακές θύρες

Το BF-533 έχει δύο πανομοιότυπες σύγχρονες σειριακές θύρες, την SPORT0 και την SPORT1. Με τη βοήθειά τους μπορεί να γίνεται γρήγορη μεταφορά δεδομένων σε διάφορα πρωτόκολλα. Αξίζει να διαβάσει κανείς το πλήθος των διαφορετικών τρόπων με τους οποίους μπορούν να λειτουργήσουν οι δύο αυτές θύρες. Λόγω της μεγάλης ταχύτητας μεταφοράς δεδομένων που επιτρέπουν, μπορούν να χρησιμοποιηθούν για την μεταφορά σήματος ψηφιακού ήχου υψηλής πιστότητας με την βοήθεια του πρωτοκόλλου I^2S. Η λειτουργία τους υποστηρίζεται από τα αντίστοιχα κανάλια DMA και interrupts. Επιπλέον είναι Full Duplex που σημαίνει ότι δεδομένα μπορούν να λαμβάνονται και να αποστέλλονται ταυτόχρονα.

3.2.2.4 Θύρα SPI

Ο επεξεργαστής διαθέτει μία θύρα SPI. Αυτό είναι ένα πολύ γνωστό πρωτόκολλο για την επικοινωνία μεταξύ ολοκληρωμένων κυκλωμάτων. Εκτός από τις ακίδες μεταφοράς δεδομένων χρειάζεται και άλλη μία τουλάχιστον ακίδα ανά ολοκληρωμένο η οποία να το ενεργοποιεί όταν δεδομένα προορίζονται προς αυτό. Η θύρα αυτή χρησιμοποιείται κυρίως για σχετικά χαμηλής ταχύτητας επικοινωνία και για την αποστολή ρυθμίσεων προς άλλα ολοκληρωμένα. Υποστηρίζει την λειτουργία DMA και επιπλέον μπορεί να δημιουργεί interrupts κάθε φορά που λαμβάνεται ένα byte ή όταν υπάρξει κάποιο σφάλμα.

3.2.2.5 UART

Με τη βοήθεια του UART μπορεί ο επεξεργαστής να συνδεθεί με άλλες συσκευές όπως έναν προσωπικό υπολογιστή μέσω της σειριακής θύρας ή με άλλες καταναλωτικές συσκευές μέσω μίας θύρας IrDA®. Η διαφορά του από τις δύο σειριακές θύρες SPORT0, SPORT1 είναι ότι το UART είναι ασύγχρονο δηλαδή δεν μεταφέρεται κάποιο σήμα ρολογιού μεταξύ των συσκευών σε επικοινωνία. Πρέπει να σημειωθεί ότι το UART αυτό μπορεί να παράγει διαφόρων τύπων interrupts π.χ. κάθε φορά που ολοκληρώνεται η λήψη ενός byte όπως επίσης ότι υποστηρίζει δύο ανεξάρτητα κανάλια DMA ένα για εγγραφή και ένα για ανάγνωση πράγμα που σημαίνει ότι ολόκληρες ακολουθίες από bytes μπορούν να αποστέλλονται ή να διαβάζονται χωρίς να ξοδεύεται επεξεργαστική από τον πυρήνα. Η λειτουργία και του UART είναι Full Duplex.

3.2.2.6 GPIO

Ο επεξεργαστής διαθέτει 16 ακίδες διαθέσιμες προς τον χρήστη (programmable flags). Κάθε μία τους μπορεί να χρησιμοποιηθεί ως ψηφιακή είσοδος ή έξοδος και στην πρώτη περίπτωση μπορεί να δημιουργεί interrupts κατά την μεταβολή της κατάστασής τους. Μερικά από αυτά χρησιμοποιούνται προαιρετικά από κάποια περιφερειακά του επεξεργαστή. Οι ακίδες αυτές είναι πάρα πολύ σημαντικές για τον σχεδιαστή ενός συστήματος αφού σε αυτές μπορεί να συνδέσει κατευθείαν φορτία για να τα ελέγξει ή σήματα εισόδου π.χ. διακόπτες ή άλλου αισθητήρες. Επιπλέον μέσω αυτών των ακίδων μπορούν να ενεργοποιούνται περιφερειακά για επικοινωνία SPI. Η ύπαρξη αυτών των ακίδων θεωρείται ένα από τα κορυφαία σημείο συνάντησης του κόσμου των μικροελεγκτών με τον κόσμο των DSPs.

3.2.2.7 Χρονιστές γενικής χρήσης

Ο επεξεργαστής διαθέτει τρεις όμοιους χρονιστές γενικής χρήσης που χρονίζονται από το ρολόι συστήματος (SCLK) ή από εξωτερική πηγή. Οι χρονιστές αυτοί έχουν εξαιρετικές δυνατότητες και δίνουν πραγματικά πάρα πολλές επιλογές στον σχεδιαστή συστήματος. Συγκεκριμένα μπορούν να λειτουργήσουν με τους παρακάτω τρόπους:

1. Διαμόρφωση πλάτους παλμού (PWM)	Σε αυτή τη λειτουργία ένα pin εξόδου ταλαντώνεται σε σταθερή συχνότητα με διαφορετική όμως αναλογία υψηλής – χαμηλής κατάστασης π.χ. 10% high – 90% low. Ρυθμίζοντας κανείς τη διάρκεια παλμού, μπορεί να καθορίσει την ενέργεια του παραγόμενου σήματος και με αυτό τον τρόπο να ελέγξει πλήθος ηλεκτρονικών διατάξεων.
2. Μέτρηση πλάτους παλμών και σύλληψη	Σε αυτή τη λειτουργία ένα pin εισόδου ελέγχει τον μετρητή και τον κάνει να μετρά την διάρκεια ενός παλμού και την περίοδό του. Με αυτόν τον τρόπο μπορεί να γίνει αυτόματη αναγνώριση και της ταχύτητας επικοινωνίας του UART, με την βοήθεια ειδικής λειτουργίας (τίθεται ως πηγή του σήματος το pin RX του UART).
3. Εξωτερικού γεγονότος	Σε αυτή τη λειτουργία μετράται ο αριθμός των γεγονότων (events) που ανιχνεύονται σε ένα pin εισόδου σε ένα συγκεκριμένο χρονικό διάστημα.

Το πολύ αναλυτικό κεφάλαιο “Using the Timer”[67] του manual παρουσιάζει τις ρύθμισης και λεπτομέρειες για τη λειτουργία για κάθε ένα από τους παραπάνω τρόπους λειτουργίας.

Λίγη προσοχή χρειάζεται στην συγγραφή μίας ρουτίνας εξυπηρέτησης interrupts για την αποφυγή λαθών π.χ. μπορεί να χρειάζεται η εκτέλεση μίας εντολής SSYNC για να εξασφαλίσουμε ότι θα προλάβει να αποθηκευτεί ο καθαρισμός της σημαίας διακοπής (interrupt flag) πριν από επιστροφή από interrupt (RTI).

Να σημειωθεί τέλος ότι η λειτουργία του PPI ως έξοδο μπορεί να δεσμεύει έως και δύο timers για την δημιουργία σημάτων συγχρονισμού.

3.2.2.8 Χρονιστής επιτήρησης

Σε βιομηχανικές εφαρμογές, σε πειράματα υψηλών ενεργειών, στο αυτοκίνητο και σε πολλές άλλες περιπτώσεις υπάρχει πολύ βεβαρημένο περιβάλλον θορύβου. Κατά τη σχεδίαση ενός συστήματος γίνονται πολλές προσπάθειες ώστε να μην εισέλθει όσο είναι δυνατόν αυτός ο θόρυβος στο σύστημα. Παρόλα αυτά αν θέλει κανείς να έχει σιγουριά σε ένα τέτοιο περιβάλλον πρέπει να έχει προβλέψει την περίπτωση να βρεθεί το σύστημα σε μη-προβλέψιμη κατάσταση π.χ. να «κολλήσει» χωρίς κανένα λόγο (bug) το software.

Για την λύση αυτού του προβλήματος υπάρχει ο χρονιστής επιτήρησης. Αυτός χρονίζεται από το ρολόι συστήματος και δημιουργεί ένα interrupt το οποίο μπορεί να είναι είτε reset, είτε NMI είτε οτιδήποτε άλλο αν δεν τον «ταΐσεις» (εξού και το όνομα watchdog) για κάποιο προκαθορισμένο χρονικό διάστημα. Πιο συγκεκριμένα ο προγραμματιστής επιλέγει τον αριθμό των κύκλων συστήματος μέχρι να ενεργοποιήσει το interrupt ο χρονιστής και αποθηκεύει αυτό τον αριθμό στον καταχωρητή WDOG_CNT. Για παράδειγμα αν WDOG_CNT = 2000, ο χρονιστής θα δημιουργεί interrupt κάθε 2000 κύκλους συστήματος. Για να αποτραπεί η δημιουργία του interrupt ο προγραμματιστής πρέπει ανά τακτά χρονικά διαστήματα μέσα στον κώδικά του να ενσωματώνει μία εντολή που θα επαναφέρει τον χρονιστή στην αρχική του κατάσταση. Αυτό γίνεται με την εγγραφή μίας οποιασδήποτε τιμής στον καταχωρητή WDOG_STAT.

Θέλει πολύ προσοχή για το που θα τοποθετήσει κανείς τις εντολές που επαναφέρουν τον χρονιστή επιτήρησης. Αυτές πρέπει να βρίσκονται διασκορπισμένες σε διάφορα σημεία του κώδικα αλλά όχι μέσα σε interrupts ή μέσα σε επαναληπτικούς βρόχους. Αυτό γιατί μπορεί το interrupt να εκτελείται κανονικά ή ο βρόχος να έχει «κολλήσει» (ιδιαίτερα βρόχοι τύπου while) και να γίνεται συνέχεια επανατοποθέτηση του χρονιστή ενώ ταυτόχρονα το σύστημα δεν δουλεύει. Επίσης πρέπει να σημειωθεί ότι στην κατάσταση εξοικονόμησης ενέργειας deep sleep, το ρολόι συστήματος

[67] Σελίδα 15-16, ADSP-BF533 Blackfin™ Processor Hardware Reference

σταματάει και ως εκ' τούτου ο χρονιστής επιτήρησης δεν θα δημιουργήσει ποτέ κανένα interrupt. Το μόνο πράγμα που μπορεί να επαναφέρει τον επεξεργαστή από αυτή την κατάσταση είναι το Real Time Clock το οποίο και αυτό μπορεί να χρησιμοποιηθεί ως χρονιστής επιτήρησης σε επίπεδο δευτερολέπτου. Στην συγκεκριμένη έκδοση του επεξεργαστή που έχουμε πάντως δεν ξυπνάει ούτε από το RTC[68]. Πάντως για ακόμη μεγαλύτερη ασφάλεια θα μπορούσε να χρησιμοποιηθεί εξωτερικό δικτύωμα που να κάνει reset μετά το πέρας κάποιου διαστήματος που δεν ανταποκρίνεται ο επεξεργαστής.

3.2.2.9 Χρονιστής πυρήνα

Ο χρονιστής πυρήνα έχει μία ξεχωριστή ιδιότητα. Τρέχει στην συχνότητα του πυρήνα (CCLK). Αυτό τον κάνει πολύ χρήσιμο για την δημιουργία ενός λειτουργικού συστήματος όπου μπορεί να χρησιμοποιηθεί για την διαχείριση των threads (διεργασίες που τρέχουν ψευδο-παράλληλα). Το ουσιαστικό πλεονέκτημα είναι ότι η λειτουργία του είναι ανεξάρτητη από τις ρυθμίσεις του PLL. Συνεπώς αν κάποιος χρειάζεται ένα interrupt κάθε 200 κύκλους πυρήνα ανεξάρτητα από το ρολόι συστήματος (SCLK) μπορεί να το πετύχει με πολύ καλή ακρίβεια με τη βοήθεια αυτού του χρονιστή. Επιπλέον μπορεί να φανεί χρήσιμος στην μέτρηση του χρόνου εκτέλεσης μίας συνάρτησης. Αν μηδενιστεί στην αρχή της εκτέλεσής της και μετρηθεί η τιμή του στο τέλος μπορεί κανείς να ξέρει πόσους κύκλους κατανάλωσε η συγκεκριμένη συνάρτηση. Για ακριβείς μετρήσεις βέβαια, πρέπει να έχουν απενεργοποιηθεί τα interrupts. Το 32-bito εύρος του δίνει πολύ καλή ακρίβεια σε οποιαδήποτε λειτουργία του. Και αυτός ο χρονιστής είναι εφοδιασμένος με prescaler για την μέτρηση μεγαλύτερων χρονικών διαστημάτων. Φυσικά λόγω της υψηλής ταχύτητας λειτουργίας του αυτός ο χρονιστής έχει και τις μεγαλύτερες απαιτήσεις σε ενέργεια.

3.2.2.10 Ρολόι πραγματικού χρόνου

Το ρολόι πραγματικού χρόνου λειτουργεί με ανεξάρτητο κρύσταλλο χρονισμού και τροφοδοσία από το υπόλοιπο σύστημα και μπορεί να εργάζεται ακόμα και όταν ο επεξεργαστής βρίσκεται σε κατάσταση deep sleep ή δεν τροφοδοτείται. Η δουλειά του είναι να μετράει αξιόπιστα τον χρόνο με την βοήθεια των ενσωματωμένων μετρητών, μέχρι 32768 ημέρες με ακρίβεια δευτερολέπτου. Έχει επίσης την δυνατότητα να δημιουργεί interrupts κάθε δευτερόλεπτο, λεπτό, ώρα και ημέρα όπως επίσης την δυνατότητα δημιουργίας δύο alarms, το ένα συγκεκριμένη ώρα κάθε μέρα και το άλλο συγκεκριμένη ώρα και μέρα τον χρόνο. Το ρολόι πραγματικού χρόνου είναι το μόνο περιφερειακό του επεξεργαστή που μπορεί με την βοήθεια κάποιου Interrupt να ξυπνήσει τον επεξεργαστή από την κατάσταση deep sleep.

3.2.2.11 EBIU

Η μονάδα αυτή αναλαμβάνει την επικοινωνία με εξωτερικές μνήμες πολλών τύπων όπως DRAM, SRAM, ROM, FIFOs, Flash και ASICs – FPGA modules. Πολλές από αυτές τις μνήμες χρειάζονται ειδικό χειρισμό και τη δημιουργία επιπλέον σημάτων (RAS-CAS κ.τ.λ.) για να διατηρήσουν σωστά τα περιεχόμενά τους. Πρέπει να γίνει πολύ προσεκτικά η σωστή ρύθμιση των παραμέτρων του EBIU ώστε να εξασφαλίζεται αξιόπιστη επικοινωνία με τις εξωτερικές μνήμες.

3.2.2.12 Διεπαφή JTAG

Ο επεξεργαστής διαθέτει μία standard διεπαφή JTAG με την βοήθεια της οποίας μπορεί κανείς να ελέγξει τη λειτουργία του επεξεργαστή. Με την χρήση του JTAG μπορούμε να κάνουμε τα εξής:

1. Έλεγχο των συνδέσεων των ολοκληρωμένων πάνω στην πλακέτα
2. Έλεγχο του ιδίου του ολοκληρωμένου
3. Έλεγχο της λειτουργίας του ολοκληρωμένου καθώς αυτό εργάζεται φυσιολογικά

Με την βοήθειά του γίνεται και ο προγραμματισμός του συστήματος όπως και το debugging με την βοήθεια του VisualDSP++.

[68] Anomaly [05000092], Σελίδα 4, Blackfin® ADSP-BF533 Anomaly list

3.2.2.13 Σταθεροποιητής τάσης

Ο ενσωματωμένος στον επεξεργαστή σταθεροποιητής τάσης μπορεί να δημιουργεί την εσωτερική τάση του επεξεργαστή από μία εξωτερική τάση από 2.25V έως 3.6V. Για την λειτουργία του χρειάζεται λίγα επιπλέον εξωτερικά εξαρτήματα[69]. Η εσωτερική τάση κυμαίνεται από 0.7V έως 1.2V και ρυθμίζεται δυναμικά σε βήματα των 50 mV με τη βοήθεια του καταχωρητή VR_CTL. Με την σωστή ρύθμιση της τάσης τροφοδοσίας μπορούμε να έχουμε τεράστια εξοικονόμηση ενέργειας περιορίζοντας βέβαια τις επιδόσεις του επεξεργαστή. Συγκεκριμένα στα 1.2V ο επεξεργαστής μπορεί να τρέχει στα 600MHz ενώ στα 0.8 η μέγιστη ταχύτητά του είναι 250MHz[70]. Μετά από κάθε μεγάλη αλλαγή της τάσης τροφοδοσίας πρέπει να περιμένουμε κάποιο χρόνο για να ξανακλειδώσει το PLL. Στην συγκεκριμένη έκδοση του Blackfin® που έχουμε εμείς ο σταθεροποιητής τάσης δεν φαίνεται να δουλεύει και πολύ καλά[71].

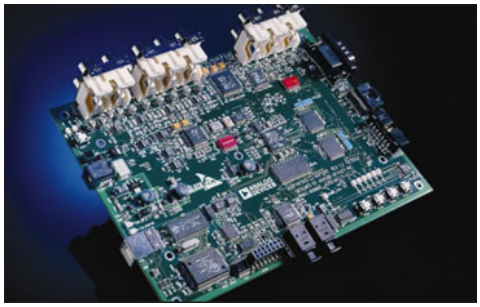
3.2.2.14 Boot rom

Ο επεξεργαστής έχει δύο pins (BMODE[1:0]) τα οποία καθορίζουν από πού θα αρχίσει να εκτελείται ο κώδικας μετά το reset[72]. Ο κώδικας της Boot rom εκτελείται αν επιλεχθεί από τα pins αυτά και δίνει την δυνατότητα εκτέλεσης κώδικα από εξωτερική 8-bitn flash μνήμη και από εξωτερική SPI μνήμη με 8-bitn ή 16-bitn διευθυνσιοδότηση. Αυτές οι επιλογές κάνουν τον επεξεργαστή ικανό να εκτελέσει κώδικα από μία μεγάλη ποικιλία εξωτερικών μνημών σε διάφορα κόστη χωρίς να χρειάζεται προσθήκη πρόσθετων. Αν δεν υπήρχε η ενσωματωμένη boot rom, αν ήθελε κανείς να εκτελέσει κώδικα από κάποιον άλλο τύπο μνήμης εκτός από την βασική (Async Bank 0 - διεύθυνση 0x2000 0000)[73], θα έπρεπε να επιβαρυνθεί με την προσθήκη επιπλέον εξωτερικών εξαρτημάτων.

[69] Σελίδα 8-28, ADSP-BF533 Blackfin™ Processor Hardware Reference
[70] Σελίδα 21, Blackfin® Embedded Processor Datasheet
[71] Anomaly [05000066], [05000092], Σελίδα 2 και 4, Blackfin® ADSP-BF533 Anomaly list
[72] Table 4-10, Σελίδα 4-41, ADSP-BF533 Blackfin™ Processor Hardware Reference
[73] Σελίδα 3-17, ADSP-BF533 Blackfin™ Processor Hardware Reference

3.3 Το αναπτυξιακό BF533 – EZKIT – LITE®

Το ADSP-BF533® από μόνο του δεν είναι ιδιαίτερα χρήσιμο αφού εν γένει δεν διαθέτει ούτε μνήμη RAM για τα δεδομένα (εκτός της εσωτερικής cache), ούτε μνήμη για την αποθήκευση του προγράμματός του, ούτε κάποια μη-ψηφιακή διασύνδεση με τον έξω κόσμο που να μας δίνει τη δυνατότητα να πειραματιστούμε με πραγματικά σήματα.



Εικόνα 57. Το αναπτυξιακό

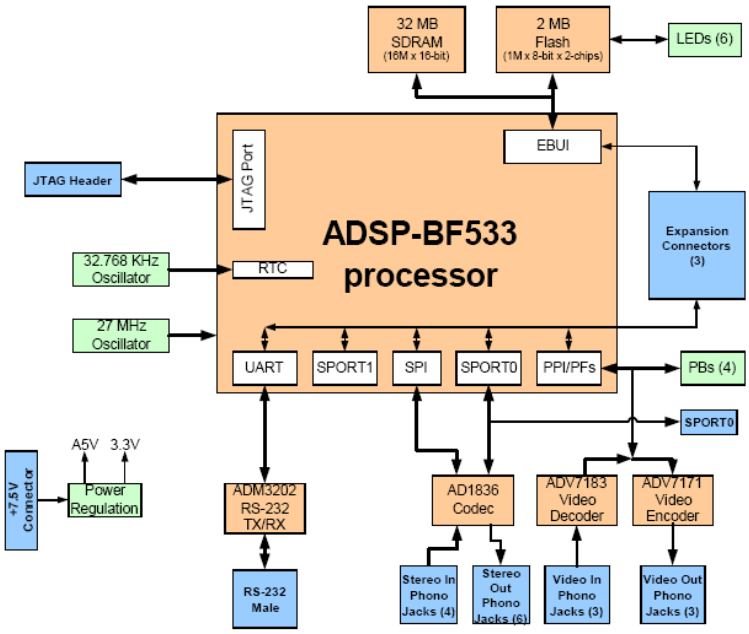
Για να διευκολύνει η κάθε κατασκευάστρια εταιρία την διαδικασία αξιολόγησης των επεξεργαστών και της γρήγορης προτυποποίησης, παρέχει έτοιμες πλακέτες με τον επεξεργαστή και τα απαραίτητα περιφερειακά ώστε να εργάζεται σωστά και να μπορούν εύκολα να αναπτυχθούν οι περισσότερες εφαρμογές που αναμένεται να καλύψει.

Η Analog Devices™ διαθέτει για το ADSP-BF533® την αναπτυξιακή πλατφόρμα με το όνομα ADSP-BF533 EZ-KIT Lite®. Ο λόγος για τον οποίο επιλέξαμε αυτόν τον επεξεργαστή ήταν η αναπτυξιακή του πλατφόρμα! Πραγματικά, αυτή τη στιγμή καμία άλλη από τις μεγάλες εταιρίες δεν διαθέτει αναπτυξιακό που να έχει τη δυνατότητα για προτυποποίηση video σε τιμή που να μπορεί να συγκριθεί με αυτή του ADSP-BF533 EZ-KIT Lite®. Οι περισσότερες πλατφόρμες έχουν μόνο ένα Audio Codec (A/D – D/A) που επιτρέπει μόνο την προτυποποίηση εφαρμογών ήχου. Για εφαρμογές Video οι περισσότεροι προτείνουν προαιρετικές plug-in πλακέτες. Επίσης πολλά άλλα αναπτυξιακά δεν διαθέτουν JTAG-USB interface και επαφίονται στο ότι ενδιαφερόμενος έχει ένα PCI-JTAG interface ή κάποιο άλλο τρόπο για τον προγραμματισμό τους, κάτι το οποίο δεν είναι καθόλου προφανές.

Επειδή η δικιά μας εφαρμογή είναι μία custom εφαρμογή video και οι επιδόσεις που χρειαζόμαστε καλύπτονται από τον επεξεργαστή ADSP-BF533 η επιλογή της συγκεκριμένης πλατφόρμας είναι η καλύτερη που θα μπορούσε να είχε γίνει τη δεδομένη χρονική στιγμή. Μερικούς μήνες αργότερα η Analog Devices™ ανακοίνωσε την διάθεση του ADSP-BF561 και του αντίστοιχου EZ-KIT Lite®. Ο επεξεργαστής αυτός θα ταίριαζε καλύτερα στις ανάγκες μας γιατί έχει δύο θύρες PPI επιτρέποντας την ταυτόχρονη DMA εγγραφή στον video encoder και DMA ανάγνωση από τον video decoder, πράγμα που θα απλοποιούσε αρκετά την υλοποίηση του λογισμικού και θα μείωνε σημαντικά τον χρόνο ανάπτυξης της εφαρμογής. Επιπλέον έχει διπλάσια υπολογιστική ισχύ (3 GMACs) χάρη στην ενσωμάτωση δύο ανεξάρτητων πυρήνων BlackFin™.

Το ADSP-BF533 EZ-KIT Lite® περιλαμβάνει μία πλακέτα με τον επεξεργαστή ADSP-BF533® και ένα πλήθος περιφερειακών που καλύπτουν εφαρμογές ήχου και video, το εγχειρίδιο χρήσης της, καλώδια USB, τροφοδοσίας κ.τ.λ. και ένα CD-ROM με περιορισμένη έκδοση του αναπτυξιακού περιβάλλοντος VisualDSP++™. Μέσα στο CD υπάρχουν και κάποια παραδείγματα τα οποία είναι έτοιμα να τρέξουν. Το επίπεδό τους είναι βασικό και έχει γίνει προσπάθεια ώστε να αναδεικνύουν τις δυνατότητες του επεξεργαστή και των περιφερειακών. Δεν θα αναφερθούμε καθόλου στην περιορισμένη έκδοση του VisualDSP++™ αφού θα περιγραφεί αναλυτικά η πλήρης έκδοσή του σε επόμενο κεφάλαιο. Το μόνο που μπορούμε να επιβεβαιώσουμε είναι ότι οι περιορισμοί δεν αφήνουν περιθώρια για σοβαρή ανάπτυξη με την περιορισμένη έκδοση.

Τα κυκλώματα της πλακέτας μας δίνουν την δυνατότητα να φορτώνουμε και να δοκιμάζουμε προγράμματα σε ελάχιστο χρόνο μέσω της θύρας USB ενός υπολογιστή. Επιπλέον μία πολύ μεγάλη διευκόλυνση είναι το Background Telemetry Channel (BTC) που μας δίνει την δυνατότητα να φορτώνουμε και να διαβάζουμε δεδομένα από και προς τον επεξεργαστή χωρίς να διακόψουμε την λειτουργία του. Όλα αυτά γίνονται δυνατά με την βοήθεια του JTAG προς USB interface της Cypress. Προσοχή θα έπρεπε να δοθεί στην πληθώρα διακοπτών που βρίσκονται πάνω στην πλακέτα. Ο σκοπός τους είναι να μπορούν να αποσυνδέσουν κάποιες βαθμίδες ώστε να ελευθερώνονται ακροδέκτες για γενικές εφαρμογές ή λειτουργίες της ίδιας της πλακέτας.



Εικόνα 58. Block διάγραμμα του αναπτυξιακού

Στην παραπάνω εικόνα φαίνονται οι βαθμίδες του επεξεργαστή και τα περιφερειακά που βρίσκονται διαθέσιμα πάνω στην πλακέτα. Αναλυτικά τα χαρακτηριστικά του είναι[74]:

1. Ο επεξεργαστής ADSP-BF533 σε θήκη 160-pin Mini-BGA και ρολόι στα 27 MHz
2. Μνήμη
    a. 32Mb (16M x 16-bits) SDRAM (Synchronous Dynamic Read Access Memory)
    b. 2MB (512K x 16 x 2chips) Flash (διατηρεί τα περιεχόμενά της χωρίς τροφοδοσία)
3. AD1836: A/D – D/A για εφαρμογές ήχου υψηλής πιστότητας (96 kHz – 24 bit) και τις αντίστοιχες υποδοχές εισόδου (4) – εξόδου(6) τύπου RCA για τα σήματα ήχου
4. Ολοκληρωμένα για υποστήριξη αναλογικού video (κοινής τηλεόρασης)
    a. Αποκωδικοποιητής video (ADV7183) με τρεις συνδετήρες εισόδου (RCA)
    b. Κωδικοποιητής video (ADV7171) με τρεις συνδετήρες εξόδου (RCA)
5. UART για σύνδεση με υπολογιστή μέσω της σειριακής θύρας και κυκλώματα οδήγησης
6. Leds ενδείξεων (τροφοδοσίας USB κ.τ.λ.) και 6 γενικής χρήσεως
7. 4 κουμπιά γενικής χρήσεως και 1 για reset με λογική καταστολής αναπηδήσεων
8. Τρεις 90άπινους συνδετήρες επέκτασης που δίνουν πρόσβαση σε όλες τις θύρες του επεξεργαστή (PPI, SPI, EBIU, Timers, UART, programmable flags, SPORT0 και SPORT1.
9. Συνδετήρα διασύνδεσης με άλλο JTAG interface πέραν αυτού της πλακέτας
10. Τα ολοκληρωμένα διασύνδεσης JTAG και Background Telemetry Channel για εύκολο προγραμματισμό μέσω USB
11. Σταθεροποιημένο τροφοδοτικό και φίλτρα απαραίτητα για την ομαλή λειτουργία ενός τέτοιου κυκλώματος υψηλών ταχυτήτων.
12. Η πλακέτα είναι ελεγμένη και πιστοποιημένη για ηλεκτρομαγνητική συμβατότητα.

Θα εξετάσουμε τη λειτουργία τους παρακάτω.

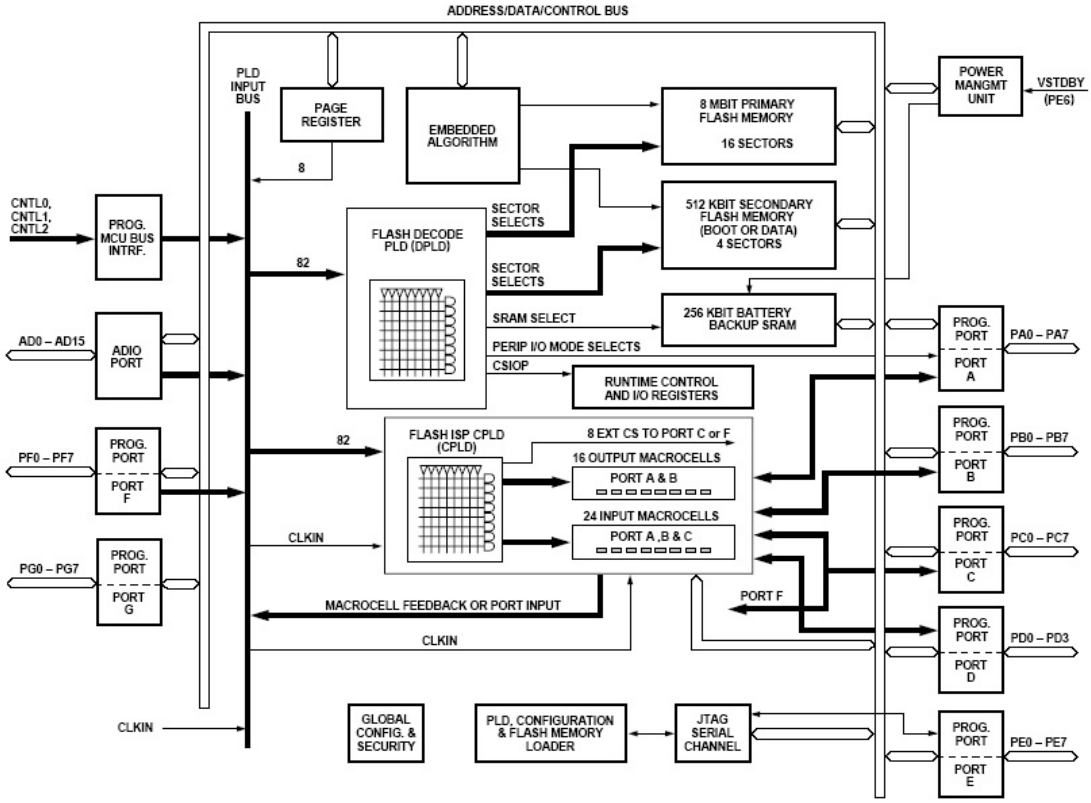
3.3.1 PSD4256G6V

Το πιο εντυπωσιακό περιφερειακό του συστήματος (μετά τον επεξεργαστή) είναι οι δύο μνήμες Flash που χρησιμοποιεί για την αποθήκευση του προγράμματος. Αν ερευνήσει κανείς το φυλλάδιό τους θα δει ότι κάθε άλλο παρά απλές Flash είναι. Οι «μνήμες» αυτές περιέχουν:

- 1Mbyte πρωτεύουσας Flash μνήμης
- 64Kbytes βοηθητικής Flash μνήμης

[74] ADSP-BF533 EZ-KIT Lite Evaluation System Manual (p. viii)

- 32Kbytes στατικής Ram με την δυνατότητα διατήρησης με μπαταρία
- PLD με πάνω από 3000 πύλες. Αυτό δίνει την δυνατότητα υλοποίησης εξωτερικής λογικής (μικρό FPGA).
- 7 πόρτες εισόδου-εξόδου με 52 pins
- Διεπαφή JTAG για τον προγραμματισμό και τον έλεγχο λειτουργίας
- Προγραμματιζόμενη μονάδα διαχείρισης κατανάλωσης ενέργειας



Εικόνα 59. Το διάγραμμα βαθμίδων των PSD4256G6V

Όπως μπορεί κανείς να δει στην Εικόνα 59 το περιεχόμενο των βαθμίδων αυτών είναι εντυπωσιακό. Το EZ-KIT® εκμεταλεύεται αρκετά τη λειτουργικότητα αυτών των ολοκληρωμένων βάζοντάς τα να ρυθμιζουν αρκετές λειτουργίες διεπαφής ανάμεσα στο DSP και τα περιφερειακά. Συγκεκριμένα[75] με την βοήθεια των ολοκληρωμένων αυτών γίνεται ο έλεγχος των εξωτερικών Leds, το Reset των audio codec και video encoder/decoder, και ο (πολύ σημαντικός για την εφαρμογή μας) έλεγχος για την πηγή του σήματος χρονισμού της πόρτας PPI (PPI_CLK).

3.3.2 SDRAM

Το EZ-KIT® διαθέτει 32Mb SDRAM με το ολοκληρωμένο M48LC4M16ATG-75. Το μόνο που χρειάζεται να πούμε για αυτή τη RAM είναι ότι οι ρυθμίσεις του EBIU για τον χειρισμό της δίνονται στην σελίδα 2-4 του manual του EZ-KIT®. Υπό κανονικές συνθήκες αυτές δεν θα χρειαστούν αφού το VisualDSP++ δημιουργεί αυτόματα τον κώδικα για την λειτουργία τους.

3.3.3 Συνολικά η μνήμη

Μπορούμε πλέον να συνοψίσουμε τη μνήμη του επεξεργαστή στο EZ-KIT Lite® στον παρακάτω πίνακα:

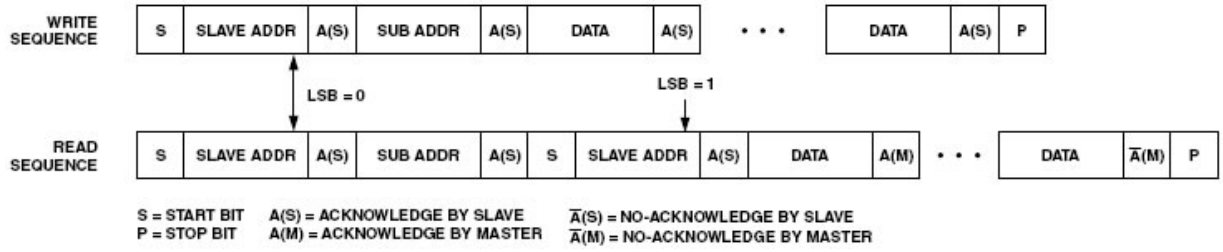
[75] ADSP-BF533 EZ-KIT Lite Evaluation System Manual (p. 2-8)

	Έναρξη	Λήξη	Περιγραφή
Εξωτερική μνήμη	0x0000 0000	0x01FF FFFF	SDRAM Bank 0 (SDRAM)
	0x2000 0000	0x2000 FFFF	ASYNC Memory Bank 0 (Primary Flash A)
	0x2010 0000	0x201F FFFF	ASYNC Memory Bank 1 (Primary Flash B)
	0x2020 0000	0x202F FFFF	ASYNC Memory Bank 2 (Flash A and B Secondary Memory, SRAM and Internal Registers)
	Υπόλοιπες διευθύνσεις		Δεν χρησιμοποιούνται
Εσωτερική μνήμη	0xFF80 0000	0xFF80 3FFF	Data Bank A SRAM 16 KB
	0xFF80 4000	0xFF80 7FFF	Data Bank A SRAM/CACHE 16 KB
	0xFF90 0000	0xFF90 3FFF	Data Bank B SRAM 16 KB
	0xFF90 4000	0xFF90 7FFF	Data Bank B SRAM/CACHE 16 KB
	0xFFA0 0000	0xFFA0 FFFF	Instruction SRAM 64 KB
	0xFFA1 0000	0xFFA1 3FFF	Instruction SRAM /CACHE 16 KB
	0xFFB0 0000	0xFFBO 0FFF	Scratch Pad SRAM 4 KB
	0xFFC0 0000	0xFFDF FFFF	System MMRs 2 MB
	0xFFE0 0000	0xFFFF FFFF	Core MMRs 2 MB
	Υπόλοιπες διευθύνσεις		Δεσμευμένες

3.3.4 Video Encoder/Decoder

Το EZ-KIT Lite® έχει ως video encoder το ADV7171 με τη βοήθεια του οποίου δημιουργείται σήμα video και ως video decoder το ADV7183 με το οποίο μπορούμε να αποκωδικοποιήσουμε video από μία camera ή οποιαδήποτε άλλη πηγή video. Αυτά υποστηρίζονται από ένα σύνολο τελεστικών ενισχυτών video και φίλτρων ώστε να υπάρχει σωστή διασύνδεση video. Και τα δύο τους χρησιμοποιούν PPI για τη γρήγορη μεταφορά δεδομένων και το πρωτόκολλο I^2C για την ρύθμισή τους.

Το πρωτόκολλο I^2C είναι ένα σειριακό πρωτόκολλο χαμηλότερης ταχύτητας από το SPI το οποίο όμως χρειάζεται μόνο δύο ακίδες για τη λειτουργία του. Θα αναφερθούμε συνοπτικά στο πρωτόκολλο αυτό γιατί ο BlackFin δεν έχει ελεγκτή I^2C και αν θέλουμε να προγραμματίσουμε τα ολοκληρωμένα αυτά θα πρέπει να τον υλοποιήσουμε εμείς προγραμματιστικά. Για λεπτομέρειες της υλοποίησης I^2C και την λίστα των παραμέτρων που μπορεί κανείς να προγραμματίσει μπορεί κάθε ενδιαφερόμενος να ανατρέξει στα manuals των ADV7171 και ADV7183. Η διασύνδεση I^2C των δύο ολοκληρωμένων συνδέεται στις γενικής χρήσης ακίδες PF0 που μεταφέρει το ρολόι (SCL) και PF1 που μεταφέρει δεδομένα (SDAT). Αυτό που κάνει κανείς μέσω του πρωτοκόλλου I^2C είναι να γράφει και να διαβάζει καταχωρητές από το προγραμματιζόμενο ολοκληρωμένο και με αυτό τον τρόπο να ελέγχει τη λειτουργία του.



Εικόνα 60. Διαδικασίες εγγραφής και ανάγνωσης με I^2C

Οι διαδικασίες εγγραφής και ανάγνωσης του επεξεργαστή φαίνονται στην Εικόνα 60.

- Κατά την διαδικασία εγγραφής αποστέλλεται ένα bit έναρξης, (S) στην συνέχεια 8bit με την διεύθυνση του ολοκληρωμένου στο οποίο απευθύνεται η εγγραφή (κάθε ολοκληρωμένο έχει μοναδική διεύθυνση κι έτσι δε χρειάζεται ξεχωριστή ακίδα ενεργοποίησης όπως στο SPI) και στην συνέχεια αναμένεται η απόκριση του προγραμματιζόμενου ολοκληρωμένου. Όταν αυτή έρθει αποστέλλεται η 8-bitη διεύθυνση του καταχωρητή που πρέπει να εγγραφεί και περιμένουμε και πάλι την απόκριση του ολοκληρωμένου. Μόλις γίνει αυτό στέλνουμε τα 8bits δεδομένων τα οποία θέλουμε να εγγραφούν σε αυτή τη θέση μνήμης και περιμένουμε

το ολοκληρωμένο να αποκριθεί. Αν θέλουμε να γράψουμε δεδομένα σε συνεχόμενους καταχωρητές, μπορούμε να συνεχίσουμε να στέλνουμε δεδομένα. Στο τέλος αποστέλλεται ένα bit λήξης και η επικοινωνία τερματίζει.

- Κατά τη διαδικασία ανάγνωσης επαναλαμβάνονται τα ίδια βήματα όσον αφορά τον καθορισμό της διεύθυνσης ολοκληρωμένου και της διεύθυνσης καταχωρητή. Με αυτόν τον τρόπο ο εσωτερικός δείκτης καταχωρητή του ολοκληρωμένου πηγαίνει στην κατάλληλη θέση. Στην συνέχεια επανεκινείται η διαδικασία με την αποστολή ενός νέου bit έναρξης. Τώρα αποστέλλεται η διεύθυνση ανάγνωσης του ολοκληρωμένου που κατά σύμβαση είναι η ίδια με την διεύθυνση εγγραφής μόνο που το λιγότερο σημαντικό ψηφίο της είναι 1 δηλαδή είναι κατά ένα μεγαλύτερη η αριθμητική τιμή της. Αμέσως μετά αρχίζει η ανάγνωση δεδομένων από το ολοκληρωμένο από τον ένα καταχωρητή στον επόμενο. Η διαδικασία τερματίζει αν ο ελεγκτής που διαβάζει το ολοκληρωμένο δεν αποκριθεί κατά την ολοκλήρωση της αποστολής κάποιου καταχωρητή αλλά στείλει ένα bit λήξης.

Οι διευθύνσεις εγγραφής και ανάγνωσης για τα δύο ολοκληρωμένα όπως χρησιμοποιούνται στο EZ-KIT LITE® φαίνονται στον παρακάτω πίνακα:

	Κωδικοποιητής video ADV7171[76]	Αποκωδικοποιητής video ADV7183[77]
Εγγραφή	0x54 (b01010100)	0x88 (b10001000)
Ανάγνωση	0x55 (b01010101)	0x89 (b10001001)

Πρέπει να σημειώσουμε ότι με την βοήθεια μίας ακίδας (ALSB) μπορεί να αλλαχθεί η διεύθυνση αυτών των ολοκληρωμένων έτσι ώστε να μπορούμε να έχουμε μέχρι δύο όμοια ολοκληρωμένα σε ένα δίαυλο.

Προγραμματισμός του κωδικοποιητή είναι απαραίτητο να γίνει, γιατί η εφαρμογή μας θέλουμε να χειρίζεται σήματα PAL ενώ το ολοκληρωμένο αυτό έχει ως προεπιλεγμένες τιμές αυτές που χρειάζονται για τη παραγωγή σημάτων NTSC. Οι τιμές που πρέπει να τεθούν για λειτουργία σε PAL δίνονται στις τελευταίες σελίδες του manual[78].

Για να δώσουμε ένα παράδειγμα προγραμματισμού με I^2C θα αναφέρουμε ότι αν στον δίαυλο γραφτούν τα εξής “Start bit, 0x54, 0x01, 0x80, Stop bit”, τότε θα γραφεί στο ολοκληρωμένο ADV7171 στην διεύθυνση μνήμης 1 η τιμή 0x80 η οποία θα κάνει το ολοκληρωμένο να παράγει στην οθόνη ένα πρότυπο με κατακόρυφες χρωματιστές μπάρες.

3.3.5 Audio Codec

Για την δημιουργία ήχου το EZ-KIT® χρησιμοποιεί το ολοκληρωμένο AD1836 που έχει τρία στερεοφωνικά κανάλια εξόδου και δύο κανάλια εισόδου με συχνότητα δειγματολειψίας εως 96kHz. Το AD1836 συνδέεται με τον επεξεργαστή μέσω της σειριακής θύρας SPORT0 και μπορεί να επικοινωνήσει μαζί του με δύο τρόπους, είτε με TDM (Time Division Multiplexing) είτε μέσω του πρωτοκόλλου I^2S. Το πρωτόκολλο I^2S επιτρέπει λειτουργία στα 96 kHz αλλά μόνο με δύο κανάλια εξόδου ενώ το TDM επιτρέπει μέχρι 48kHz αλλά για όλα τα κανάλια εισόδων – εξόδων. Σημαντικό είναι να αναφέρουμε ότι η ρύθμιση του AD1836 γίνεται με την βοήθεια της θύρας SPI και η ακίδα ενεργοποίησης είναι η PF4.

3.3.6 Leds και πλήκτρα

Τα Leds του EZ-KIT® μπορούμε να τα χειριστούμε μέσω της FLASH όπως είπαμε σε προηγούμενη ενότητα. Τα κουμπιά είναι διαθέσιμα στις ακίδες γενικής χρήσης του επεξεργαστή PF8-11. Από εκεί

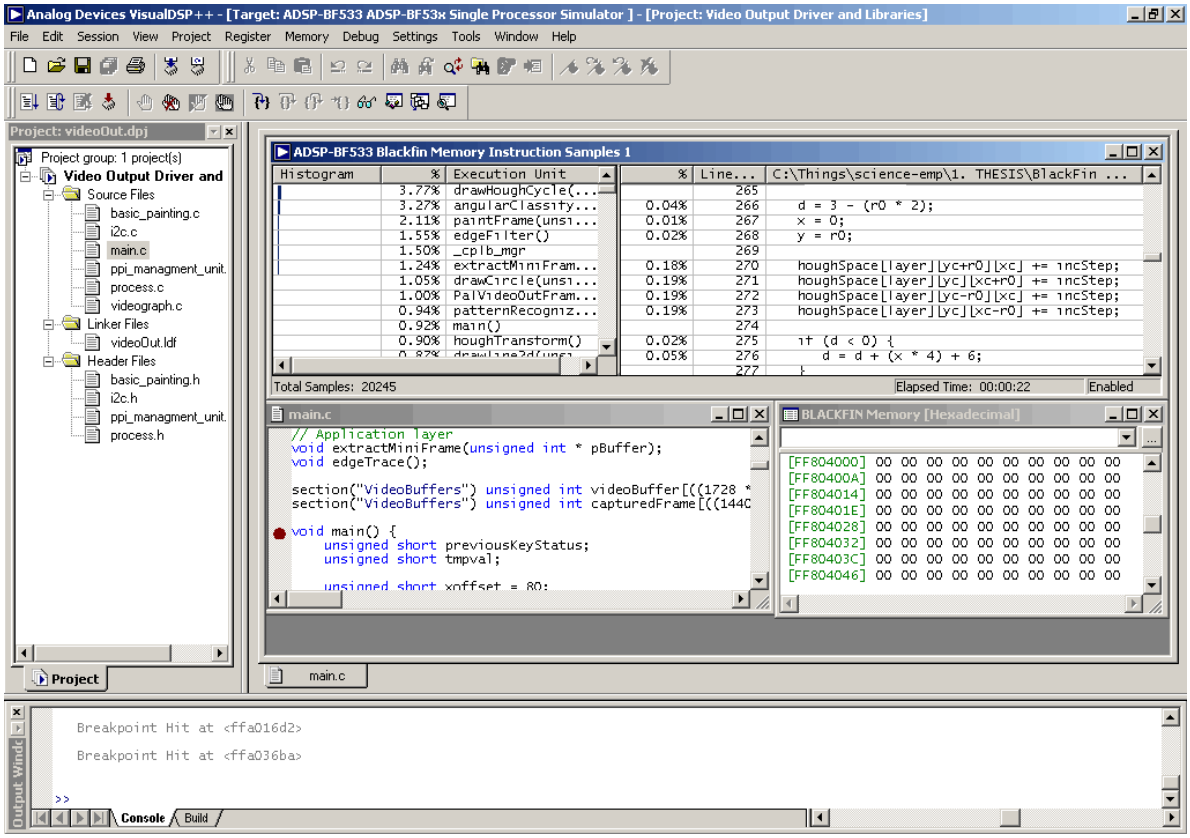
[76] Σελίδα 21, ADV7171 manual, Analog Devices, 2001
[77] Σελίδα 18, ADV7183 manual, Analog Devices, 2002
[78] Σελίδα 42, πίνακας “PAL B, D, G, H, I”, ADV7171 manual, Analog Devices, 2001

ο προγραμματιστής μπορεί να τα αξιοποιήσει είτε με τη δημιουργία interrupts είτε με την ανάγνωση της κατάστασής τους (polling).

3.4 Λογισμικό και τεκμηρίωση

3.4.1 VisualDSP++

Το VisualDSP++ είναι το επίσημο ολοκληρωμένο αναπτυξιακό περιβάλλον (IDE) που παρέχει η Analog Devices™ για τους BlackFin®. Με τη βοήθειά του μπορεί κανείς να γράψει προγράμματα σε συμβολική γλώσσα (assembly), C, C++ και συνδυασμούς τους.



Εικόνα 61. Το περιβάλλον ανάπτυξης VisualDSP++

Η συγγραφή του κώδικα γίνεται σε γραφικό περιβάλλον με αυτόματο τονισμό (με χρώμα) μερών του κώδικα όπως σχόλια, λέξεων κλειδιών των γλωσσών προγραμματισμού και ονόματα καταχωρητών του επεξεργαστή. Το περιβάλλον αυτό είναι το ίδια για αρκετά DSPs της Analog Devices πράγμα που διευκολύνει πολύ τους προγραμματιστές που προγραμματίζουν και άλλες οικογένειες.

Θα πρέπει να τονίσουμε το εύχρηστο σύστημα βοήθειας το οποίο περιλαμβάνει πλήρη βοήθεια για το ίδιο το VisualDSP++, τις γλώσσες προγραμματισμού που υποστηρίζει, τον BlackFin®, και τις βιβλιοθήκες. Θα πρέπει να τονίσουμε ότι αν επιλέξει κανείς μία λέξη στον κώδικα και πατήσει το πλήκτρο F1 αυτομάτων γίνεται αναζήτηση της λέξης αυτής στο σύστημα βοήθειας και παρουσιάζονται τα αποτελέσματα.

Σημαντική βοήθεια προσφέρουν τα παραδείγματα (σε assembly και C) τα οποία παρέχονται μαζί με το VisualDSP++. Μελετώντας τα κανείς, όπως κάναμε κι εμείς για την εφαρμογή μας, μπορεί να μάθει γρήγορα, πρακτικές προγραμματιστικές τακτικές για την σωστή λειτουργία του BlackFin® και να προχωρήσει τμηματικά στην δημιουργία σύνθετων εφερμογών.

Δεν θα έπρεπε να παραλείψουμε να αναφέρουμε τις πολύ σημαντικές βιβλιοθήκες για την γλώσσα C που παρέχονται μαζί με το λογισμικό. Σε αυτές έχει γίνει πάρα πολύ καλή δουλειά και παρέχονται σχεδόν όλες οι standard βιβλιοθήκες της C (stdio.h, stdlib.h κ.τ.λ.) αλλά και πάρα πολλές χρήσιμες βιβλιοθήκες για την επεξεργασία σήματος πραγματικών και μιγαδικών αριθμών. Το βιβλίο "C/C++ Compiler and Library Manual for Blackfin® Processors" που παρουσιάζουμε παρακάτων έχει αναλυτική παρουσίαση όλων των συναρτήσεων των βιβλιοθηκών. Θα θέλαμε να σημειώσουμε την πολύ χρήσιμη εντολή "printf()" η οποία βρίσκεται στην βιβλιοθήκη "stdio.h" με την βοήθεια της οποίας μπορεί κανείς να εμφανίσει μορφοποιημένα δεδομένα από τον BlackFin® στο αντίστοιχο πλαίσιο του VisualDSP++. Παρόλα αυτά και αυτή η εντολή λόγω του ότι χρησιμοποιεί την θύρα USB λειτουργεί με σχετικά μεγάλη καθυστέρηση.

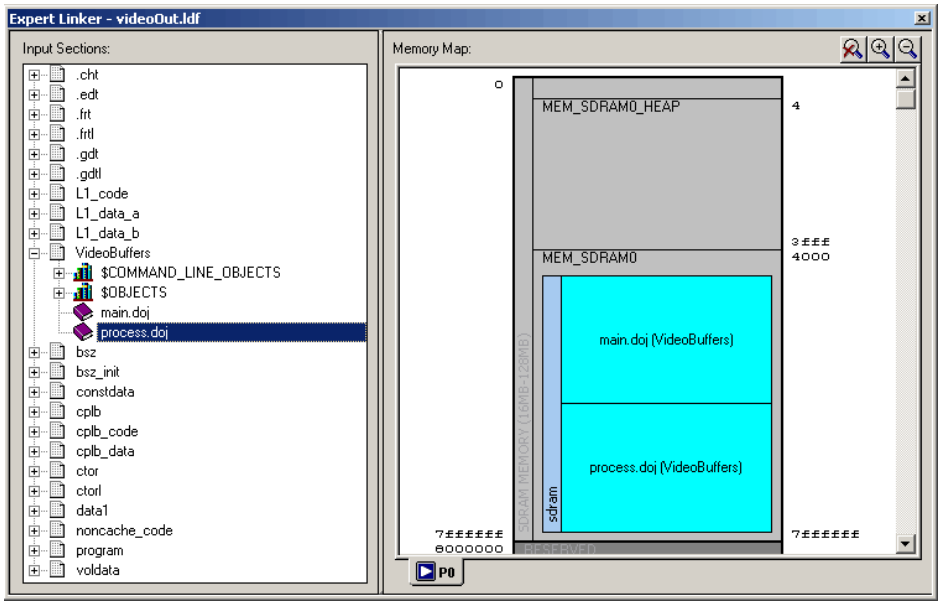
Πολύτιμο χαρακτηριστικό για την ανίχνευση και διόρθωση σφαλμάτων αποτελεί η δυνατότητα δημιουργίας γραφικών παραστάσεων, έγχρωμων ή ασπρόμαυρων εικόνων και video από την μνήμη του επεξεργαστή. Χρησιμοποιούμε κατά κόρον την δυνατότητα δημιουργίας εικόνων για την ανάπτυξη της εφαρμογής μας.

Στην πράξη επίσης φαίνεται πάρα πολύ χρήσιμο το σύστημα διαχείρισης εργασιών (projects) το οποίο επιτρέπει πολλά projects να είναι ανοιχτά ταυτόχρονα. Επιπλέον κατά την εκκίνηση του VisualDSP++ όλα (projects, debug windows κ.τ.λ.) είναι όπως κατά το κλείσιμό της εξοικονομώντας πολύτιμο χρόνο.

Αν θέλαμε να αναφέρουμε και ένα αρνητικό είναι ότι ο οδηγός (driver) διασύνδεσης με το EZ-KIT Lite™ έχει πιθανώς κάποια σφάλματα και κατά καιρούς κολλάει. Η λύση σε αυτές τις περιπτώσεις ώστε να μην χάσει κανείς την εργασία του είναι να αποσυνδέσει το USB καλώδιο, οπότε η εφαρμογή επανέρχεται στην κανονική της λειτουργία. Για την επανα-επικοινωνία με το EZ-KIT Lite™ μέσω του USB θα πρέπει να τερματίσει το VisualDSP++, να γίνει επανασύνδεση του καλωδίου και στην συνέχεια επανέναρξη του VisualDSP++. Τώρα γίνεται προφανές γιατί σώζει πολύ πολύτιμο χρόνο το ότι κατά την εκκίνηση του VisualDSP++ όλα είναι όπως τα άφησες.

Για την ανάπτυξη εφαρμογών παρέχονται πολλά χρήσιμα εργαλεία.

• Statistical / Linear profiler	Δύο βοηθητικά προγράμματα που μετράνε πόσος χρόνος ξοδεύεται σε κάθε σημείο του προγράμματος και ως σκοπό έχουν να αναδείξουν τα αργά σημεία του κώδικα. Προφανώς εκεί πρέπει να γίνει περαιτέρω βελτιστοποίηση αν θέλουμε να επιτύχουμε βελτίωση της ταχύτητας. Ο ένας τύπος profiler (linear) τρέχει ως εξομοίωση στο computer ενώ ο άλλος (statistical) λαμβάνει πραγματικές μετρήσεις από το DSP καθώς αυτό εκτελεί τον κώδικα.
• Expert Linker	Αυτό είναι ένα πάρα πολύ χρήσιμο εργαλείο με την βοήθεια του οποίου μπορούμε να τροποποιούμε με γραφικό τρόπο τα αρχεία τύπου .ldf. Εναλλακτικά αυτά μπορούν να τροποποιηθούν με το χέρι αλλά αυτό είναι αρκετά πιο επίπονο. Τα αρχεία .ldf χρησιμοποιούνται από τον linker και καθορίζουν σε ποιο σημείο της μνήμης του επεξεργαστή ή του συστήματος θα αποθηκευτούν τα αντικείμενα του προγράμματος όπως οι συναρτήσεις και τα δεδομένα. Για παράδειγμα στην Εικόνα 62 φαίνεται η τοποθέτηση μεγάλων πινάκων (arrays) τα οποία ορίζονται στα αρχεία main.c και process.c ενός προγράμματος και ανήκουν σε ένα τμήμα (section) που ονομάζεται "VideoBuffers", στην περιοχή της εξωτερικής μνήμης, από τη διεύθυνση 0x4000 έως το τέλος της. Με τον ίδιο τρόπο θα μπορούσαν να οριστούν μεταβλητές στην εσωτερική μνήμη (cache) ή ακόμα να οριστεί (γίνει map) στην εξωτερική μνήμη SDRAM η χρήση του Heap ώστε να μπορούμε να κάνουμε δέσμευση αρκετής μνήμης με εντολές όπως τη malloc (αρχικά ο Heap ορίζεται στην εσωτερική μνήμη, είναι μεν γρήγορος αλλά φυσικά έχει αρκετά περιορισμένο χώρο).



Εικόνα 62. Ο Expert Linker

• Background Telemetry Channel	Αυτή είναι μία λειτουργία η οποία επιτυγχάνεται με τον συνδυασμό του αναπτυξιακού (hardware) και του VisualDSP++ (software). Δημιουργείται ένα κανάλι, με την βοήθεια του οποίου μπορεί κανείς μέσα από κώδικα το DSP να ανοίγει αρχεία στον δίσκο, να στέλνει εικόνες ή video streaming στο PC και όλα αυτά χωρίς να παύει η λειτουργία του. Αυτό αποτελεί ένα πάρα πολύ χρήσιμο εργαλείο. Παρόλα αυτά δεν είναι εξαιρετικά γρήγορο λόγω των περιορισμών κυρίως του USB και γι'αυτό πρέπει να χρησιμοποιείται επιλεκτικά.
• VDK	Το VDK είναι ένα πλαίσιο ανάπτυξης το οποίο θα μπορούσε να θεωρηθεί και ως λειτουργικό σύστημα πραγματικού χρόνου. Δίνει κάποιες έτοιμες ρουτίνες για την δημιουργία και διαχείριση νημάτων, την επικοινωνία μεταξύ τους (semaphores κ.τ.λ.) και δημιουργεί ένα επίπεδο απομόνωσης ανάμεσα στο hardware και στο software που μπορεί να χρησιμεύσει ώστε να παράγονται προγράμματα εφαρμογών που θα τρέχουν και σε άλλες πλατφόρμες εκτός από τους BlackFin® με τροποποιήσεις μόνο σε επίπεδο λειτουργικού. Ένα επίσης θετικό χαρακτηριστικό είναι ότι ο προγραμματιστής μπορεί να ενσωματώσει στο τελικό πρόγραμμα μόνο τις δυνατότητες του VDK που χρησιμοποιεί και όχι όλο το λειτουργικό σύστημα. Επίσης με τη συνεργασία του VDK και του VisualDSP++ μπορεί κανείς να έχει προχωρημένες ενδείξεις για τη λειτουργία και την κατάσταση του συστήματος.
• VCSE	Το VCSE αποσκοπεί στο να δημιουργεί ένα πλαίσιο για κομμάτια κώδικα (modules) τα οποία θα μπορούν να επαναχρησιμοποιηθούν σε διάφορες εφαρμογές όπως είναι. Με αυτό τον τρόπο μπορεί να εξοικονομηθεί πολύτιμος χρόνος κατά την ανάπτυξη ενός συστήματος π.χ. αγοράζοντας modules από εταιρείες εξειδικευμένες σε κάποιο τομέα και χρησιμοποιώντας τα χωρίς να είναι απαραίτητη η τεχνογνωσία πάνω στις λεπτομέρειες λειτουργίας του module.

Το VisualDSP++ είναι το περιβάλλον στο οποίο θα βασιστούμε για να αναπτύξουμε και τη δικιά μας εφαρμογή. Αυτή είναι σε C ενώ κατά την προετοιμασία της μελετήθηκαν και έγιναν πειραματισμοί και σε προγράμματα σε assembly.

3.4.2 uCLinux

Θα πρέπει να αναφέρουμε ότι πολύ σημαντικό ρόλο στην ανάπτυξη λογισμικού για τον επεξεργαστή BlackFin® αναμένεται να παίξει το embedded λειτουργικό σύστημα για μικροπεξεργαστές, ucLinux[79]. Στο λειτουργικό αυτό σύστημα μπορούν να εκτελεστούν με μικρές μετατροπές τα περισσότερα προγράμματα που τρέχουν σε οποιοδήποτε σύστημα Linux.

Οι διαφορές ανάμεσα στο Linux και στο uCLinux για BlackFin® είναι λίγες[80]:

1. Δεν έχει πραγματική προστασία μνήμης, πράγμα που σημαίνει ότι μία προβληματική εφαρμογή μπορεί να δημιουργήσει πρόβλημα στο σύστημα.
2. Δεν υπάρχει η εντολή fork του συστήματος
3. Μπορεί να γίνει μόνο απλή δέσμευση μνήμης
4. Άλλες μικρές διαφορές

Αυτοί είναι φυσιολογικοί περιορισμοί που προκύπτουν από την απλοποίηση των απαιτήσεων του συστήματος και τη μείωση του κόστους. Φυσικά για κάθε ένα από τα παραπάνω υπάρχουν αντίστοιχες διεργασίες που υποκαθιστούν τις κανονικές, όχι όμως με τον συνήθη τρόπο που συναντάμε στο Linux.

Υπάρχουν πολλοί λόγοι για να γράψει κανείς τα προγράμματά του σε uCLinux όπως η ευκολία όταν υπάρχουν έτοιμοι drivers και η δυνατότητα χρησιμοποίησης βοηθητικών προγραμμάτων. Για να δώσουμε ένα παράδειγμα, για να γράψει κανείς ήχο με το EZ-KIT Lite® με την βοήθεια του uCLinux αρκεί να ανοίξει το /dev/dsp και να γράψει τα δεδομένα του όπως θα έκανε σε κάθε λειτουργικό Linux. Το πρόβλημα είναι φυσικά ότι για να γράψει εικόνα, για την οποία driver αυτή τη στιγμή δεν υπάρχει, ο προγραμματιστής θα πρέπει να προγραμματίσει ένα τυποποιημένο driver για Linux και όχι μίας απλή και γρήγορη λύση όπως θα έκανε σε μία γενική εφαρμογή. Αν όμως αυτός ο driver γραφεί μία φορά, τότε όλοι οι χρήστες του λειτουργικού θα μπορούν πάρα πολύ γρήγορα να γράφουν τις εφαρμογές τους και τα προγράμματα που είναι ήδη γραμμένα για Linux και χρησιμοποιούν γραφικά να τρέχουν και στο uCLinux.

Ένα επιπλέον μικρό μειονέκτημα είναι ότι το Linux δεν είναι hard real time operating system που σημαίνει ότι δεν μπορεί να εγγυηθεί ότι σε συγκεκριμένο (μικρό) χρονικό διάστημα θα έχει καλέσει μία ρουτίνα αν συμβεί κάποιο γεγονός. Προφανώς αυτό είναι ένα μειονέκτημα για εφαρμογές με πολύ υψηλές απαιτήσεις σε χρονική ακρίβεια αλλά οι περισσότερες εφαρμογές δεν είναι τέτοιες.

Βλέπουμε από τα παραπάνω ότι η λύση του uCLinux είναι μία πολύ καλή και ευέλικτη λύση την οποία όμως δεν θα χρησιμοποιήσουμε στην δικιά μας εφαρμογή γιατί δεν υπάρχει έτοιμος driver για Linux και γιατί θέλουμε να αξιοποιούμε το σύνολο της επεξεργαστικής ισχύος για τις ανάγκες της εφαρμογής μας η οποία επιπλέον δεν έχει υπερβολικές ανάγκες που να εξυπηρετούνται από ένα λειτουργικό σύστημα.

3.4.3 Βιβλία που συνοδεύουν το λογισμικό

Το VISUALDSP++ συνοδεύει πλήρης βιβλιογραφία που περιγράφει κάθε χαρακτηριστικό του περιβάλλοντος ανάπτυξης (IDE). Θα προσπαθήσω να συνοψίσω το περιεχόμενό τους ώστε να μπορεί κάποιος να βρει ακριβώς αυτό που θέλει γρήγορα. Η παρουσίαση θα γίνει με σειρά σημαντικότητας για τον χρήστη της κατηγορίας μου δηλαδή κάποιον που θέλει γνώσεις εισαγωγικές προς προχωρημένες και από την σκοπιά του Rapid Application Developer.

3.4.3.1 Getting Started Guide for 16-Bit Processors

Αυτό το βιβλίο το συνιστώ για διάβασμα cover to cover και το κυριότερο, άμεση πρακτική εφαρμογή των όσων λέει μέσα. Έχει την μορφή tutorial και δίνει μία γενική εικόνα όλων των

[79] www.uclinux.org
[80] uClinux as an Embedded OS on an Embedded Processor, Analog Devices, 2004

βασικών χαρακτηριστικών του περιβάλλοντος του VISUALDSP++ για Blackfin®. Ιδιαίτερα σημαντική είναι στο προχωρημένο tutorial η άσκηση δύο που μιλάει για το Background Telemetry Channel (BTC). Η ενσωμάτωση του BTC μέσα στο VISUALDSP++ παρουσιάζεται σε αυτή την άσκηση.

3.4.3.2 C/C++ Compiler and Library Manual for Blackfin® Processors

Το βιβλίο αυτό περιγράφει πλήρως όλες τις παραμέτρους και τα flags που αφορούν τον Compiler για τους Blackfin®. Είναι πολύ σημαντικό να διαβάσει κάποιος τα σημεία που αφορούν τις ιδιαιτερότητες των βασικών data types και των επεκτάσεών τους (fract16 και fract32) όπως υλοποιούνται για τους Blackfin® όπως και τα πολλά build-in functions τα οποία μπορούν να σώσουν πάρα πολύ χρόνο από τον προγραμματιστή και να εξασφαλίσουν ορθό και ταχύ κώδικα.
Είναι επίσης πολύ σημαντικό να δει κανείς τις προτροπές για ταχύτερο κώδικα στο μέρος 2. Επειδή σπάνια κάποιος που θα θελήσει να κάνει prototyping θα θελήσει να γράψει σε assembly είναι πολύ σημαντικό να ξέρει πως μπορεί να γράψει κώδικα τον οποίο θα κάνει σωστά optimize ο compiler.
Τέλος είναι απαραίτητο να ρίξει κανείς μία ματιά στα περιεχόμενα της C/C++ Run-time library (μέρος 3) και DSP Run-time library (μέρος 4) για να μπορεί να χρησιμοποιήσει αυτές τις functions και να σώσει πολύτιμο χρόνο. Οι σημαντικότερες functions για DSP έχουν υλοποιηθεί με πολύ βελτιστοποιημένο τρόπο στην DSP Run-time library συμπεριλαμβανομένων και fft – inverse fft σε 1 D ή 2D υλοποιήσεις, φίλτρων IIR και FIR και όλων των τύπων convolutions και correlations. Επιπλέον η C/C++ Run-time library περιλαμβάνει υλοποιήσεις των standard input και standard output functions που προσφέρουν πρόσβαση στο File system του υπολογιστή με τον οποίο είναι συνδεδεμένο ένα EZ-KIT LITE.

3.4.3.3 User's Guide for 16-Bit Processors

Το βιβλίο αυτό λέει όλα τα χαρακτηριστικά του VISUALDSP++ πολύ αναλυτικά και εξηγεί το σύνολο των λειτουργιών του. Αναλύονται όλα τα modules του προγράμματος από αυτά της συγγραφής κώδικα έως τα παράθυρα για debugging, simulation και profiling. Σημαντική και η περιγραφή των εργαλείων για την δημιουργία VCSE components. Το βιβλίο είναι αρκετά αναλυτικό, περιγράφει σχεδόν κάθε κουμπί σε κάθε toolbar και αναμένεται να ανατρέχει κανείς σε αυτό μόνο όταν έχει συγκεκριμένα προβλήματα και κυρίως στα μέρη 2 και 3.

3.4.3.4 Assembler and Preprocessor Manual for Blackfin® Processors

Το βιβλίο αυτό περιγράφει την λειτουργία του assembler και του preprocessor. Δεν νομίζω ότι χρειάζεται επιμελής ανάγνωση αυτού του βιβλίου αν κάποιος δεν έχει συγκεκριμένες απαιτήσεις. Από πλευράς Assembler αναλύονται τα keywords οι παράμετροι κλήσης και ο τρόπος με τον οποίο μπορεί κανείς να μπλέξει assembly με C και C++. Από πλευράς preprocessor αναλύεται η λειτουργία των macros και των τελεστών. Εδώ τα πράγματα είναι λίγο πολύ ίδια με την κλασική C.

3.4.3.5 Loader Manual for 16-Bit Processors

Το βιβλίο αυτό μιλάει για μία διαδικασία που βρίσκεται στο τέλος της διαδικασίας ανάπτυξης και είναι η δημιουργία αρχείων τα οποία μπορούν να φορτωθούν στον Blackfin® για να μπορεί αυτός να εκτελεί αυτόνομα το πρόγραμμα.

3.4.3.6 Product Release Bulletin for 16-Bit Processors

Το βιβλίο αυτό αφορά όσους είχαν χρησιμοποιήσει παλαιότερες εκδόσεις του VISUALDSP++ και θέλουν να ενημερωθούν γρήγορα για τις αλλαγές και τα καινούρια στοιχεία της έκδοσης 3.5.

3.4.4 Ηλεκτρονικά βιβλία

Το EZ KIT Lite® και ο επεξεργαστής BlackFin® υποστηρίζονται από ένα σύνολο βιβλίων σε ηλεκτρονική μορφή. Παρακάτω θα παρουσιάσουμε τα σημαντικότερα από αυτά.

3.4.4.1 ADSP-BF533 Blackfin™ Processor Hardware Reference

Το βιβλίο αυτό περιέχει όλες τις λεπτομέρειες της λειτουργίας του επεξεργαστή BlackFin®. Η γραφή του είναι αρκετά περιεκτική και οι σχεδόν χίλιες σελίδες του εξηγούν τη λειτουργία του πυρήνα και των περιφερειακών. Το βιβλίο αυτό απευθύνεται κυρίως σε προγραμματιστές και σχεδιαστές συστημάτων και μάλλον δεν ενδείκνυται ως εισαγωγικό βοήθημα για το θέμα. Καλύτερο είναι να ανατρέχει κανείς σε αυτό όταν έχει πολύ συγκεκριμένες απορίες για την λειτουργία του επεξεργαστή.

3.4.4.2 Blackfin® Embedded Processor, ADSP-BF531/ADSP-BF532/ADSP-BF533

Αυτό είναι ένα σαφώς μικρότερο βιβλίο (56 σελίδες) με μία συνοπτική παρουσίαση των δυνατοτήτων των επεξεργαστών που αναφέρονται στον τίτλο. Το σημαντικό είναι ότι στο βιβλίο αυτό αναφέρονται λεπτομέρειες και για τα αναλογικά χαρακτηριστικά του επεξεργαστή, όπως ο χρονισμός, η τάση τροφοδοσίας, η κατανάλωση και η μεταβολές με τη θερμοκρασία. Επίσης εδώ θα βρει κανείς πληροφορίες για τις ακίδες και το κέλυφος του BlackFin®. Το βιβλίο αυτό είναι πιο συνοπτικό και διαβάζεται πολύ πιο εύκολα.

3.4.4.3 Blackfin® DSP Instruction Set Reference

Στο βιβλίο αυτό περιέχονται πληροφορίες για την σύνταξη και την λειτουργία των εντολών του BlackFin®. Απευθύνεται κυρίως σε προγραμματιστές οι οποίοι θέλουν να αναπτύξουν εφαρμογές σε assembly ή θέλουν να χρησιμοποιήσουν inline assembly στα C προγράμματά τους. Ένα «ξεφύλλισμα» θα ωφελούσε και όποιον θέλει να προγραμματίσει για BlackFin® μιάς και θα τον «υποψίαζε» για το ποιές βελτιστοποιήσεις μπορεί να κάνει ο compiler χρησιμοποιώντας έτοιμες συναρτήσεις του hardware.

3.4.4.4 ADSP-BF533 EZ-KIT Lite™ Evaluation System Manual

Το βιβλίο αυτό περιγράφει τις λεπτομέρειες υλοποίησης του συστήματος EZ-KIT Lite™ του επεξεργαστή. Αυτό είναι απαραίτητο να το διαβάσει κανείς και μάλιστα με πολλή προσοχή αν θέλει να προγραμματίσει σωστά στο περιβάλλον του EZ-KIT Lite™. Ο λόγος και αυτού του βιβλίου είναι εξαιρετικά συνοπτικός. Πρέπει να σημειωθεί ότι στις τελευταίες σελίδες υπάρχει και το κύκλωμα του EZ-KIT Lite™ (εκτός από το copyrighted κομμάτι του JTAG interface της Cypress). Αυτό είναι και το μόνο σημείο στο οποίο μπορεί να βρει κανείς τι είναι κάθε βύσμα των συνδετήρων RCA του EZ-KIT Lite™ αν θέλει να αποφύγει τους «επικίνδυνους» πειραματισμούς.

3.4.4.5 Manuals των video encoder, decoder και audio codec

Στα βιβλία “ADV7183 Advanced Video Decoder with 10-Bit ADC and Component Input Support”, “ADV7170/ADV7171 Digital PAL/NTSC Video Encoder with 10-Bit SSAF™ and Advanced Power Management” και “AD1836A Multichannel 96 kHz Codec” μπορεί να βρει κανείς λεπτομέρειες για τα τρία πιο σημαντικά περιφερειακά του EZ-KIT Lite™. Όποιος θελήσει να αναπτύξει μία εφαρμογή στο παραπάνω περιβάλλον θα πρέπει να περάσει κάποιο χρόνο εξετάζοντας τη λειτουργία αυτών των ολοκληρωμένων με την βοήθεια των manuals τους. Σε αυτά περιγράφονται όλοι οι δυνατοί τρόποι λειτουργίας. Για παράδειγμα στην περίπτωση των επεξεργαστών video μπορούν να χρησιμοποιηθούν πολλοί διαφορετικοί τύποι φίλτρων που μπορεί σε ορισμένες περιπτώσεις να βελτιώνουν την ποιότητα της εικόνας. Όλες τις λεπτομέρειες μπορεί να τις βρει κανείς σε αυτά τα βιβλία.

3.4.4.6 Internet

Στο internet μπορεί να βρει κανείς πολλές πληροφορίες για τη λειτουργία των επεξεργαστών BlackFin®. Θα έπρεπε να επισημάνουμε τις πολλές ενδιαφέρουσες “Engineer To Engineer Notes” που μπορεί κανείς να βρει στο site της Analog Devices, όπως η “EE-149 – Tuning C Source Code for the Blackfin® Processor Compiler“ την οποία θα πρέπει να διαβάσει όποιος ενδιαφέρεται να γράψει αποδοτικό κώδικα σε C.

Οι κυριότερες διευθύνσεις στις οποίες μπορεί κανείς να βρει πληροφορίες για τον BlackFin® και τις εφαρμογές του είναι οι εξής:

URL	Περιγραφή
1. www.analog.com/technology/dsp/Blackfin/	Αυτό είναι το site της Analog Devices για τον BlackFin®. Εδώ μπορεί να βρει κανείς τεχνικά άρθρα, πληροφορίες για εφαρμογές, μία πολύ χρήσιμη knowledge base όπου υπάρχουν λύσεις για αρκετά προβλήματα και communities με προσανατολισμό σε συγκεκριμένες εφαρμογές
2. www.blackfin.org	Αυτό είναι ένα πολύ εκτενές forum με πληροφορίες και συζητήσεις σχετικές με τους BlackFin®. Είναι πολύ πιθανόν κανείς να βρει λύσεις για κάποιο πρόβλημά του σε αυτό το forum.
3. blackfin.uclinux.org	Αυτό είναι το site του uCLinux για BlackFin®. Εδώ μπορεί κανείς να βρει πληροφορίες για τις εξελίξεις, σχετικά με το δημοφιλές αυτό λειτουργικό σύστημα.

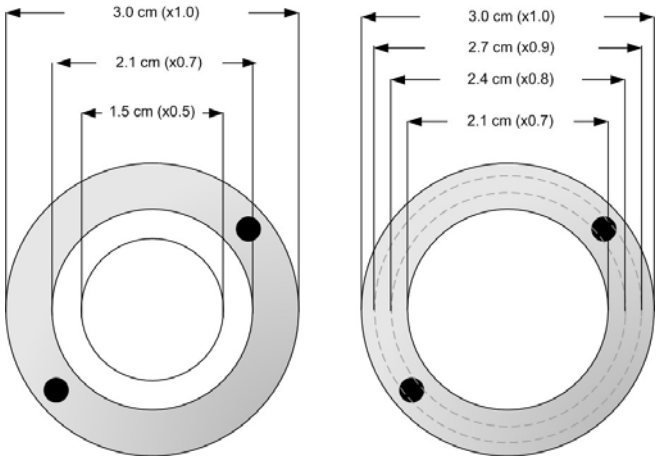
Μέρος Β. Υλοποίηση

Κεφάλαιο 4. Περιγραφή του προβλήματος

Στο σύντομο αυτό κεφάλαιο παρουσιάζεται εκτενώς το πρόβλημα που πρέπει να λυθεί και γίνεται αναφορά σε προηγούμενες εκδοχές που έχουν χρησιμοποιηθεί για την μερική του λύση. Το κεφάλαιο αυτό σκοπεύει να παρουσιάσει στον αναγνώστη τις απαραίτητες λεπτομέρειες που αφορούν το συγκεκριμένο πρόβλημα ώστε να μπορεί να κατανοήσει καλύτερα τα επόμενα κεφάλαια.

4.1 Περιγραφή προβλήματος

Το πρόβλημα που έχουμε να αντιμετωπίσουμε είναι η αναγνώριση ενός κυκλικού ανιχνευτή σε ένα θορυβώδες φόντο και στη συνέχεια να αναγνωρίσουμε πάνω σε αυτόν την γωνία των χαρακτηριστικών οπών στήριξης.

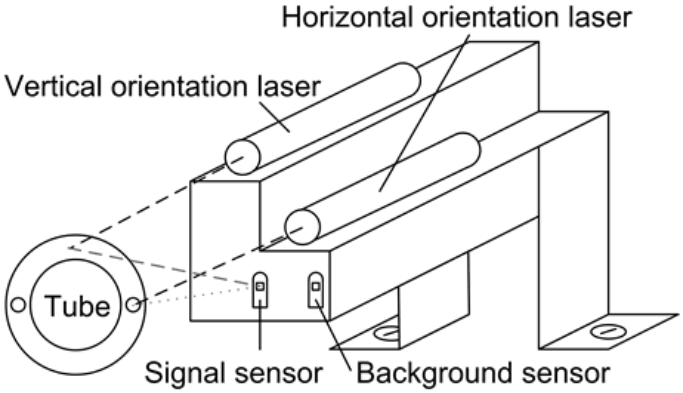


Εικόνα 63. Η γεωμετρία του ανιχνευτή

Τη γεωμετρία του προτύπου που θέλουμε να αναγνωρίσουμε μπορούμε να δούμε στην Εικόνα 63. Εκεί φαίνονται οι διαστάσεις όπως τις μετρήσαμε. Κατά την αναγνώριση εικόνας, θα μας φανούν πιο χρήσιμες οι σχετικές διαστάσεις κατά τις οποίες ορίζουμε ως κύκλο μοναδιαίας ακτίνας τον εξωτερικό και όλοι οι άλλοι εκφράζονται συναρτήσει αυτού. Στο δεξί μέρος της παραπάνω εικόνας μπορούμε να δούμε την περιοχή στην οποία βρίσκονται οι δύο οπές στήριξης.

4.2 Παλαιότερες λύσεις

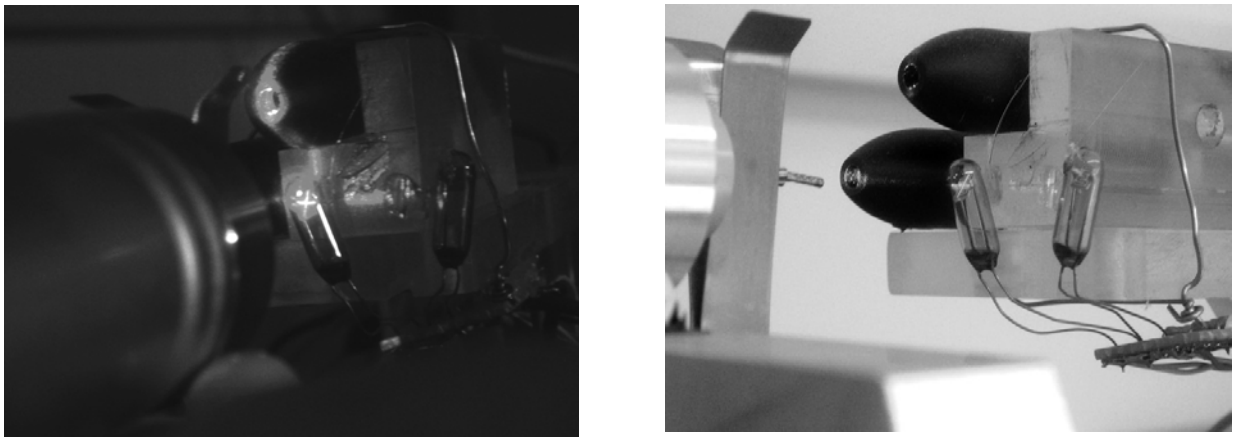
Η λύση που είχε χρησιμοποιηθεί παλαιότερα για το πρόβλημα της ευθυγράμμισης αυτών των αισθητήρων για την διαδικασία ελέγχου που λάμβανε χώρα στο Πολυτεχνείο από το Ελληνικό τμήμα ποιοτικού ελέγχου του πειράματος Atlas, είχε σχεδιαστεί, αναπτυχθεί και δοκιμαστεί με επιτυχία από τον γράφοντα υπό την επίβλεψη και με τη θερμή υποστήριξη του κυρίου Μαλτέζου, επίκουρου καθηγητή του τομέα φυσικής της σχολής μας. Η διάταξη η οποία χρησιμοποιήθηκε φαίνεται στην Εικόνα 64.



Εικόνα 64. Διάταξη ανίχνευσης θέσης οπών στήριξης

Σε αυτή υποτίθεται ότι ο κυλινδρικός ανιχνευτής είναι στερεωμένος στη βάση στήριξης (γνωστό κέντρο) και αυτό που ζητάμε είναι μία ένδειξη για το αν οι οπές στήριξης βρίσκονται σε οριζόντια, κάθετη ή άλλη θέση. Η πληροφορία αυτή θα μπορούσε να χρησιμοποιηθεί για την αυτοματοποιημένη περιστροφή και ευθυγράμμιση του αισθητήρα σε οριζόντια και κατακόρυφη θέση και την λήψη μετρήσεων όπως επιβάλλεται από τη διαδικασία ελέγχου.

Ο στόχος αυτός επιτεύχθηκε με τη βοήθεια δύο δεικτών laser, δύο φωτοαισθητήρων, μίας κάρτας συλλογής δεδομένων της National Instruments™ και μίας διάταξης στήριξης σχεδιασμένης με αρκετή ακρίβεια. Συγκεκριμένα κάθε ένα από τα δύο laser pointer εστίαζε στη θέση που βρίσκεται οπή όταν ο ανιχνευτής βρίσκεται στην οριζόντια ή κατακόρυφη θέση αντίστοιχα. Με προσεκτική ρύθμιση των κλίσεων των δεικτών laser οι αντανακλάσεις των δύο δεσμών πάνω στον αισθητήρα εστιάστηκαν πάνω στην φωτοαντίσταση μέτρησης.

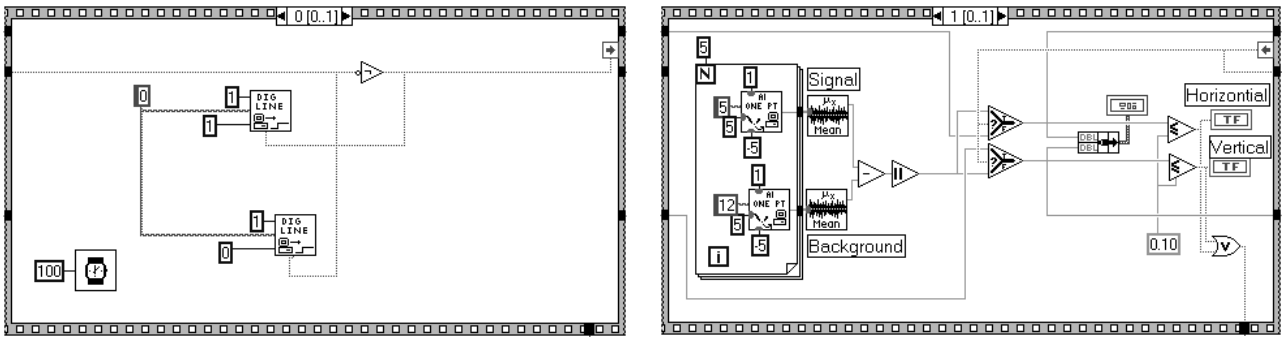


Εικόνα 65. Φωτογραφία της διάταξης

Όπως μπορεί κανείς να παρατηρήσει στην Εικόνα 65 αριστερά, πάνω στην φωτοαντίσταση εμφανίζεται ένας φωτεινός σταυρός. Το οριζόντιο μέρος του οφείλεται στο ένα laser pointer και το κατακόρυφο στο άλλο.

Όταν η οπή διακόπτει τη ροή μίας εκ των δύο δεσμών στην φωτοαντίσταση εμφανίζεται ουσιαστική μείωση στην μετρούμενη ένταση. Μία δεύτερη φωτοαντίσταση τοποθετημένη κοντά στην πρώτη μετράει τον φωτισμό υποβάθρου ο οποίος αφαιρείται μετά προγραμματιστικά ώστε να μην αλλοιώνονται οι μετρήσεις από αλλαγές στον περιβάλλοντα φωτισμό.

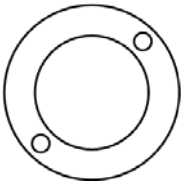
Χρειαζόταν να διαχωριστούν τα δύο σήματα, της οριζόντιας και της κατακόρυφης δέσμης, παρόλο που χρησιμοποιείται μόνο ένας αισθητήρας μέτρησης φωτός. Για τον λόγο αυτό ενεργοποιούνταν εναλλάξ το οριζόντιο laser και το κατακόρυφο laser και εν τω μεταξύ παίρνονταν οι αντίστοιχες μετρήσεις. Η διαδικασία αυτή επαναλαμβάνεται αρκετά γρήγορα και δίνει μία συνεχή ένδειξη της κατάστασης των δύο καναλιών, της οριζόντιας και της κατακόρυφης δέσμης.



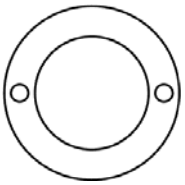
Εικόνα 66. Το πρόγραμμα στην γραφική γλώσσα του LabView™

Το πρόγραμμα που χρησιμοποιήθηκε γι' την υλοποίηση της παραπάνω λειτουργίας είναι το LabView™ της National Instruments™, κυρίως λόγω της ευκολίας με την οποία μπορούσε να επικοινωνήσει με την κάρτα συλλογής δεδομένων της ιδίας εταιρείας. Όπως μπορεί κανείς να δει στην Εικόνα 66, παίρνονται αρκετές μετρήσεις από κάθε κατάσταση (οριζόντια δέσμη – κάθετη δέσμη) και βγαίνει ο μέσος όρος με σκοπό να αναιρέσει τον, οπτικό κυρίως, θόρυβο όπως αυτόν που παράγουν διάφοροι τύποι λαμπτήρων. Στην συνέχεια αφαιρείται η ένταση της φωτοαντίστασης υποβάθρου και μετά η εξαγόμενη τιμή συγκρίνεται με μία τιμή κατωφλιού.

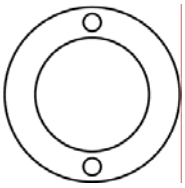
1. Τυχαία θέση

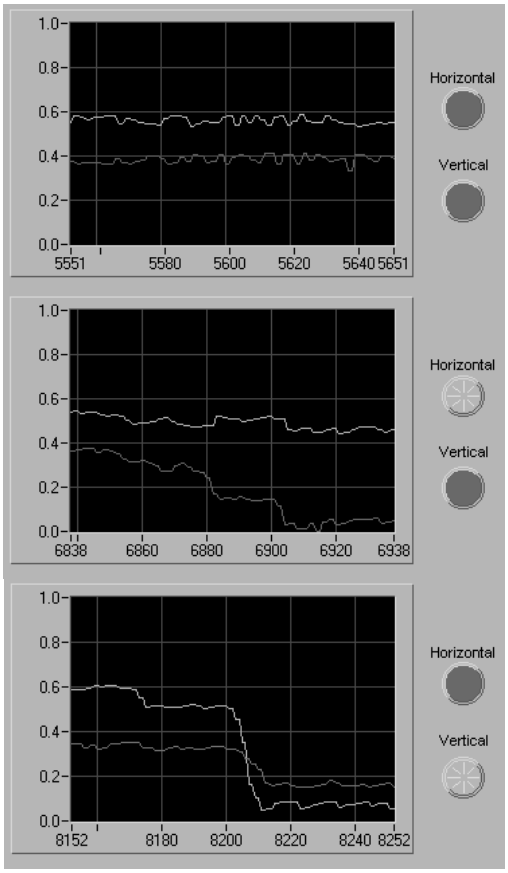


2. Οριζόντια θέση



3. Κατακόρυφη θέση





Εικόνα 67. Σήματα των δύο καναλιών για διαφορετικές θέσεις του αισθητήρα

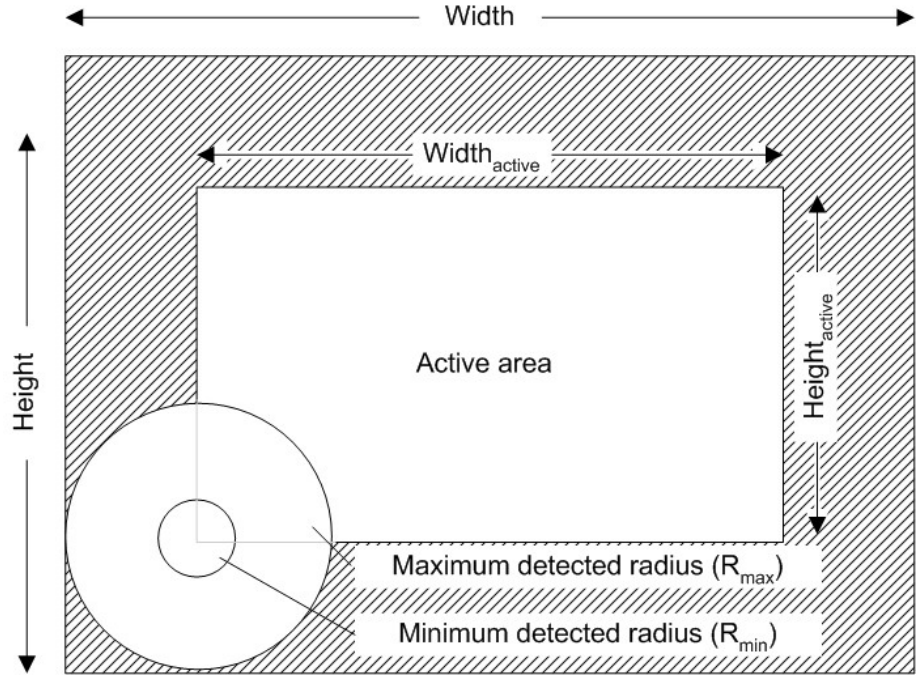
Μπορούμε να δούμε στην Εικόνα 67 το αποτέλεσμα της λειτουργίας του προγράμματος. Όταν είναι σε τυχαία θέση και τα δύο κανάλια έχουν υψηλή ένταση. Όταν βρίσκεται στην οριζόντια, η τιμή ενός καναλιού πέφτει δραματικά, κάτω από το κατώφλι και ενεργοποιείται η ένδειξη "Horizontal". Αντίστοιχα συμβαίνει και στην κατακόρυφη περίπτωση. Όπως μπορούμε να παρατηρήσουμε, ιδιαίτερα στην κατακόρυφη περίπτωση, τα δύο κανάλια δεν διαχωρίζονται πλήρως. Ο διαχωρισμός μπορεί να βελτιωθεί αρκετά με διάφορους τρόπους.

Η ουσιαστική αδυναμία της παραπάνω διάταξης είναι ότι δίνει ένδειξη μόνο για γωνίες 0^o και 90^o. Αυτό θα ανάγκαζε μία διάταξη περιστροφής και ευθυγράμμισης του αισθητήρα να κινείται αρκετά αργά ώστε να μπορεί να σταματάει αμέσως μόλις ανιχνευθεί μία από τις δύο καταστάσεις. Η χαμηλή ταχύτητα δεν θα έκανε πρακτική την κατασκευή μίας τέτοια διάταξης αφού ακόμα και γυρίζοντας τον ανιχνευτή με το χέρι μπορεί κανείς να πετύχει αρκετά καλύτερη ταχύτητα. Συνεπώς παρόλο που το εγχείρημα ολοκληρώθηκε με απόλυτη επιτυχία, το τελικό αποτέλεσμα δεν ήταν ικανό να καλύψει πλήρως τις ανάγκες του εργαστηρίου.

4.3 Προδιαγραφές συστήματος αναγνώρισης εικόνας

Ουσιαστική λύση στο παραπάνω πρόβλημα δίνει ένα σύστημα αναγνώρισης εικόνας. Αν έχουμε μέτρηση ακριβείας της γωνίας των οπών στήριξης, αυτή μπορεί να παρέχει την απαραίτητη ανάδραση που θα επιτρέψει σε ένα σύστημα αυτομάτου ελέγχου να εργάζεται σε ανταγωνιστικές ταχύτητες.

Αυτό ακριβώς το σύστημα θα πραγματευτούμε παρακάτω ως ένα παράδειγμα εφαρμογής των τεχνικών επεξεργασίας και αναγνώρισης εικόνας και του μετασχηματισμού Radon – Hough.



Εικόνα 68. Προδιαγραφές μίας συσκευής αναγνώρισης για το πρόβλημα αυτό

Στην Εικόνα 68 βλέπουμε τις βασικές προδιαγραφές που πρέπει να ορίσουμε για μία συσκευή που κάνει αναγνώριση του προτύπου της εικόνας του αισθητήρα. Αν έχουμε μία εικόνα με πλάτος Width και ύψος Height και θέλουμε να αναγνωρίσουμε πρότυπο με εξωτερική ακτίνα από R_{min} έως R_{max}, θα έχουμε μία ενεργή περιοχή στην οποία θα μπορούμε επιτυχώς να αναγνωρίζουμε το πρότυπο η οποία θα έχει πλάτος $Width_{active}$ = Width – $2xR_{max}$ και ύψος $Height_{active}$ = Height – $2xR_{max}$. Στην περιοχή η οποία φαίνεται γραμμοσκιασμένη στην Εικόνα 68 δεν μπορούμε να εγγυηθούμε την ανίχνευση του προτύπου, αφού στην περιοχή αυτή ο μέγιστος κύκλος θα εμφανίζεται κομμένος, δηλαδή θα υπάρχουν σημεία του εκτός της εικόνας μας.

Με δεδομένη την R_{min}, μπορούμε να υπολογίσουμε την ελάχιστη ακτίνα για την οποία θα πρέπει να υπολογιστεί ο μετασχηματισμός Hough. Συγκεκριμένα αν επιλέξουμε να κάνουμε αναγνώριση με τον εξωτερικό και τον μεσαίο δακτύλιο, που όπως είπαμε στο κεφάλαιο 2 είναι η καλύτερη επιλογή, τότε η $R_{HoughMin}$ είναι ίση με τον μικρότερο ακέραιο ο οποίος είναι μικρότερο ή ίσος με 0.7 x R_{min}. Μπορούμε λοιπόν να ορίσουμε το εύρος $\Delta R = R_{max} - R_{HoughMin}$, που μας δίνει τον αριθμό των ακτινών για τις οποίες πρέπει να υπολογιστεί ο μετασχηματισμός Hough.

Ο μετασχηματισμός Hough και οι διεργασίες που γίνονται πάνω σε αυτόν είναι οι πιο αργές για μια συσκευή σαν κι αυτή που περιγράφουμε επειδή ο χώρος Hough είναι χώρος τριών διαστάσεων. Είναι συνεπώς κρίσιμο να επιλέξουμε σωστά τις παραμέτρους του προβλήματος μέσω των οποίων μπορούμε να ρυθμίσουμε το υπολογιστικό κόστος της όλης διεργασίας.

Πιο συγκεκριμένα ο χώρος Hough θα έχει Width x Height x ΔR σημεία εκ των οποίων $Width_{active}$ x $Height_{active}$ x ΔR χρησιμοποιούνται για περαιτέρω διεργασίες πέρα από την δημιουργία του. Ο λόγος για τον οποίο στην δημιουργία του χώρου Hough δημιουργούμε παραπάνω σημεία από όσα θα επεξεργαστούμε είναι γιατί κάνουν τον υπολογισμό πιο γρήγορο (!). Πραγματικά αν εισάγαμε στον αλγόριθμο που σχεδιάζει κύκλους στον χώρο Hough ελέγχους για το αν ένα σημείο βγαίνει εκτός πλαισίου, ο αλγόριθμος θα εκτελείται πολύ πιο αργά. Βάζοντας αυτά τα επιπλέον σημεία είμαστε σίγουροι ότι ποτέ δεν θα ζωγραφιστεί κύκλος με σημεία εκτός πλαισίου στον χώρο Hough. Παρόλα αυτά αν η μνήμη μας ήταν περιορισμένη και θέλαμε να την μειώσουμε για λόγους κόστους θα μπορούσαμε να μειώσουμε την απαιτούμενη μνήμη εισάγοντας επιπλέον επεξεργαστικό κόστος σε αυτό το σημείο.

Με δεδομένο (όπως θα φανεί με μετρήσεις παρακάτω) ότι οι εικόνες από σήματα video μπορούν να αναπαρασταθούν βολικά σε εικόνες με μέγεθος 316 x 237 και υποδιαιρέσεις του σε δυνάμεις του δύο μπορούμε να δούμε διάφορες επιλογές που έχουμε για το μέγεθος της εικόνας και ΔR που θα επεξεργαστούμε. Στους υπολογισμούς λαμβάνονται 16 bit για κάθε σημείο του χώρου Hough.

Πλάτος	Ύψος	ΔR	Μνήμη	Αβεβαιότητα
316	237	30	4.29 Mbyte	1 pixel
158	118	30	1.07 Mbyte	4 pixel
79	59	30	0.27 Mbyte	16 pixel

Όπως μπορούμε να δούμε η πρώτη επιλογή δημιουργεί πολύ μεγάλο χώρο ο οποίος θα θέλει πολύ χώρο για να επεξεργαστεί, η τρίτη δεν θα έχει αρκετή ευκρίνεια οπότε η πιο αποδοτική λύση δίνεται από τη δεύτερη επιλογή.

Μία καλή επιλογή για το R_{max} είναι R_{max}=40 δηλαδή κύκλοι μέχρι 1/3 του ύψους της εικόνας. Αυτό δίνει $R_{HoughMin}$ = 10 και αντίστοιχα R_{min} = 15 δηλαδή 1/8 του ύψους της εικόνας. Αυτό μας δίνει μία ενεργή περιοχή ανίχνευσης κύκλων ίση με $Width_{active}$ = 78 σημεία και ύψος $Height_{active}$ = 38. Βλέπουμε δηλαδή ότι με τις επιλογές μας αυτές μπορούμε να ανιχνεύουμε κύκλους περίπου στο ½ του πλάτους και στο 1/3 του ύψους της εικόνας.

Ίσως θα θέλαμε να περιορίσουμε τις δυνατές τιμές του ΔR για να επιτύχουμε μεγαλύτερη ενεργή περιοχή. Αν θέσουμε ΔR = 23 έχουμε μείωση της μνήμης που απαιτείται σε 0.82 Mbyte. Με R_{max} = 33 δηλαδή περίπου 2/7 του ύψους της εικόνας έχουμε $R_{HoughMin}$ = 10 και R_{min} = 15 όπως και πριν. Με αυτό τον τρόπο η ενεργός περιοχή επεκτάθηκε σε $Width_{active}$ = 92 και $Height_{active}$ = 52 δηλαδή περίπου 4/7 του μήκους και ½ του ύψους της εικόνας, αύξηση δηλαδή του ενεργού εμβαδού κατά 61%. Αυτό φυσικά συνεπάγεται και αντίστοιχη αύξηση του επεξεργαστικού κόστους. Παρόλα αυτά η συσκευή με αυτές τις προδιαγραφές είναι σίγουρα καλύτερη από την προηγούμενη.

Αυτό που μπορούμε να παρατηρήσουμε είναι ότι όσο πιο «μακριά» θέλουμε να «βλέπουμε», (μεγάλα R_{max}) τόσο μειώνεται η περιοχή στην οποία μπορούμε να εστιάσουμε (ενεργός περιοχή) και αντίστροφα.

Οι παραπάνω υπολογισμοί αποσκοπούν στο να μας δώσουν «αίσθηση» του προβλήματος ώστε να μπορούμε να πάρουμε σωστές αποφάσεις αρχιτεκτονικής του συστήματος. Παρόλα αυτά η λεπτομερής υλοποίηση θα διαφέρει σε κάποια σημεία ώστε να επιτυγχάνεται βέλτιστη απόδοση.

Κεφάλαιο 5. Προτυποποίηση σε Matlab™

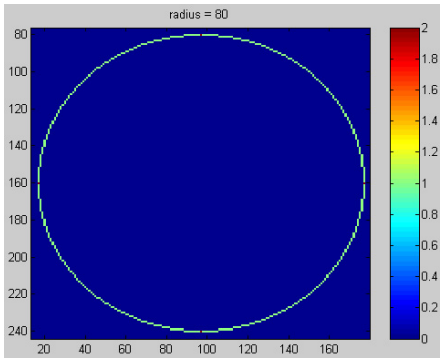
Στο κεφάλαιο αυτό παρουσιάζεται η πρότυπη υλοποίηση κάποιον κομματιών της εφαρμογής σε PC με τη βοήθεια του πακέτου λογισμικού Matlab™. Η προτυποποίηση αυτή ήταν απαραίτητη για να ελέγξουμε γρήγορα αν ο μετασχηματισμός Radon και οι διάφορες άλλες τεχνικές μπορούν να δώσουν λύση στο πρόβλημά μας και να αποκτήσουμε αίσθηση των σημείων που πρέπει να προσέξουμε κατά την υλοποίηση στο DSP. Το κεφάλαιο αυτό έχει πολύ ενδιαφέρον για τον αναγνώστη που θέλει να παρακολουθήσει βήμα – βήμα την διαδικασία ανάπτυξης μίας εφαρμογής αναγνώρισης εικόνας ώστε να κατανοήσει τη μεθοδολογία που ακολουθείται και να αποκτήσει αίσθηση των προβλημάτων που καλείται κανείς να αντιμετωπίσει στην πορεία και του σκεπτικού λύσεώς τους.

5.1 Προτυποποίηση αλγορίθμου του Bresenham

Ο αλγόριθμος αυτός είναι πολύ σημαντικός γιατί μπορούμε να τον χρησιμοποιήσουμε τόσο στην σχεδίαση των κύκλων όσο και στον σχεδιασμό των κώνων που αντιστοιχούν σε κάθε σημείο για τον μετασχηματισμό Hough. Ενώ θα περίμενε κανείς ότι ένας τόσο κλασικός και σχετικά απλός στην υλοποίησή του αλγόριθμος δεν έχει τίποτα το ιδιαίτερο, οι ανάγκες της εφαρμογής επέβαλαν την τροποποίηση ακόμα και αυτού. Ο αλγόριθμος σε Matlab™ φαίνεται στο παρακάτω πλαίσιο, όπου θεωρούμε ότι το κέντρο ορίζεται από τις συντεταγμένες X_C, Y_C και η ακτίνα από τη μεταβλητή radius.

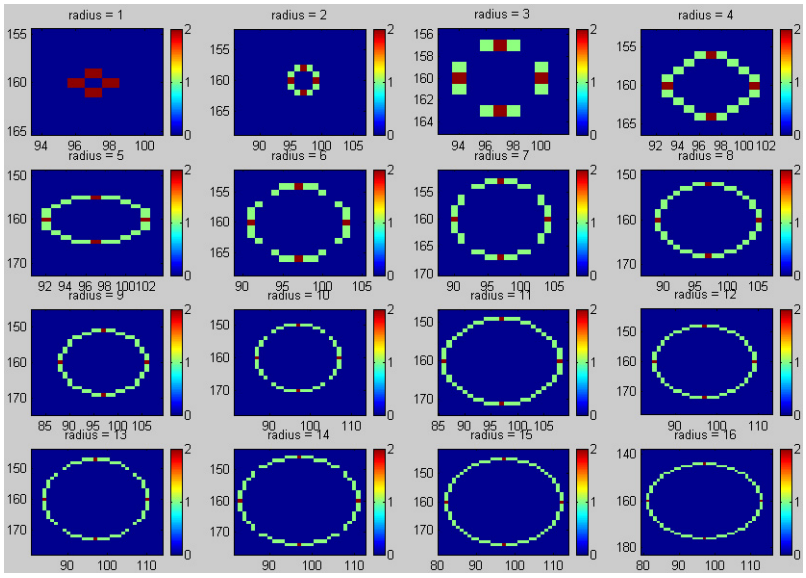
```
d = 3 - (radius * 2);
x = 0;
y = radius;

while x < y
      arr(Yc+x, Xc+y) = arr(Yc+x, Xc+y) + 1;
      arr(Yc+x, Xc-y) = arr(Yc+x, Xc-y) + 1;
      arr(Yc-x, Xc+y) = arr(Yc-x, Xc+y) + 1;
      arr(Yc-x, Xc-y) = arr(Yc-x, Xc-y) + 1;
      arr(Yc+y, Xc+x) = arr(Yc+y, Xc+x) + 1;
      arr(Yc+y, Xc-x) = arr(Yc+y, Xc-x) + 1;
      arr(Yc-y, Xc+x) = arr(Yc-y, Xc+x) + 1;
      arr(Yc-y, Xc-x) = arr(Yc-y, Xc-x) + 1;

      if (d < 0)
            d = d + (x * 4) + 6;
      else
            d = d + ((x-y) * 4) + 10;
            y = y -1;
      end
      x = x + 1;
end
```

Για να επαληθεύσουμε τη λειτουργία του κάναμε μία υλοποίηση σε Matlab™ (drawcircle.m) στην οποία η τιμή κάθε σημείου του κύκλου αύξανε κατά 1 όπως χρειάζεται για τον μετ/μό Hough.



Εικόνα 69. Κλασική υλοποίηση Bresenham

Με έκπληξη παρατηρήσαμε όπως φαίνεται και στην Εικόνα 69 ότι ενώ τα περισσότερα σημεία είχαν την τιμή 1 όπως είναι το σωστό, τα σημεία στις κορυφές είχαν την τιμή 2 δηλαδή ο αλγόριθμος τα είχε επιλέξει δύο φορές. Αυτό συμβαίνει πράγματι, γιατί στην αρχή (x = 0, y = R) η συμμετρία κάνει τα σημεία αυτά να επιλέγονται από τα δυο συναπτά τεταρτημόρια. Αυτό δε δημιουργεί πρόβλημα όταν χρειάζεται να ζωγραφίσουμε ένα κύκλο με ένα χρώμα. Απλά τα σημεία αυτά θα περαστούν δύο φορές. Στην περίπτωση όμως που θέλουμε να ζωγραφίσουμε μία περιοχή με διαφάνεια ή του αλγορίθμου Hough αυτό είναι πρόβλημα (στην πραγματικότητα όχι κρίσιμο, αφού η στατιστική φύση του αλγορίθμου θα αναιρούσε αυτή την μικρή ανωμαλία, αλλά για κύκλους μικρών ακτινών σίγουρα θα έδινε αναξιόπιστα αποτελέσματα).



Εικόνα 70. Κύκλοι από την κλασική υλοποίηση Hough με ακτίνα 1 έως 16

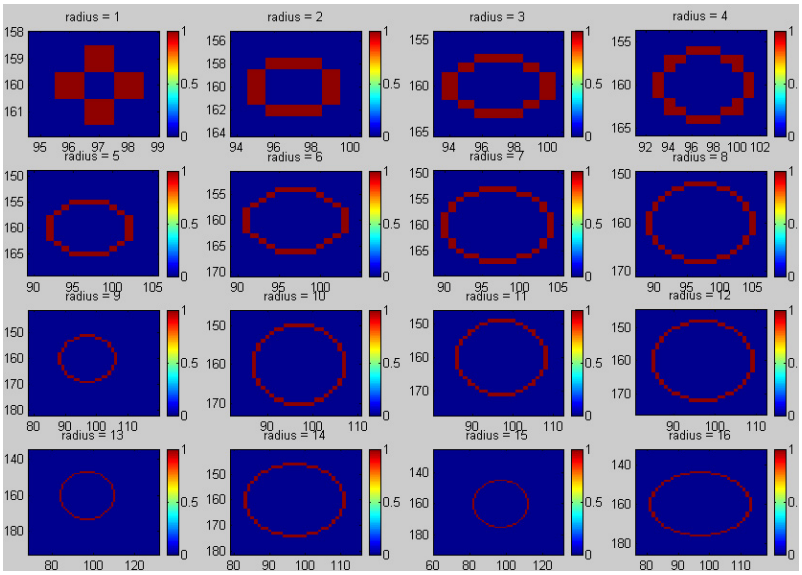
Για την καλύτερη μελέτη του φαινομένου αυτού δημιουργήσαμε μία πλατφόρμα ελέγχου αλγορίθμων κύκλων διαφόρων μεγεθών (test_circle_draw.m). Το αποτέλεσμα εκτέλεσης για κύκλους με ακτίνες από 1 έως 16 φαίνεται στην Εικόνα 70. Όπως μπορούμε να παρατηρήσουμε π.χ. για ακτίνα ίση με 3, το πρόβλημα είναι εμφανέστατο. Αν χρησιμοποιούσαμε αυτόν τον αλγόριθμο, τα σημεία κάθε κύκλου θα ζωγραφίζονταν από 0 έως 2 φορές για κάθε σημείο.

Παρατηρούμε δηλαδή ότι και ένα ακόμα σφάλμα παρουσιάζεται. Σε ορισμένες περιπτώσεις (r = 3, 6, 7, 10 κ.τ.λ.) τα σημεία που είναι σε γωνίες συμμετρικές των 45^{o} δε «ψηφίζονται» ούτε μία φορά. Αυτό συμβαίνει γιατί στις περιπτώσεις αυτές το x γίνεται ίσο με το y και ο αλγόριθμος τερματίζει χωρίς να επιλεχθούν τα συγκεκριμένα σημεία. Αν πάλι επιλέγαμε να εκτελείται ο αλγόριθμος και για x = y, τότε πάντα θα ζωγραφίζονταν τα προαναφερόμενα σημεία, αλλά στις υπόλοιπες (r = 1, 2, 4, 5, 8 κ.τ.λ.) περιπτώσεις θα ζωγραφίζονταν και αυτά δύο φορές.

Μία τροποποιημένη έκδοση δίνει λύση στα παραπάνω προβλήματα χωρίς την εισαγωγή επιπλέον υπολογιστικού κόστους, αλλά με την αύξηση των γραμμών κώδικα. Η έκδοση αυτή φαίνεται παρακάτω.

```
d = 3 - (radius * 2);
x = 0;
y = radius;

arr(Yc+radius, Xc) = arr(Yc+radius, Xc) + 1;
arr(Yc, Xc+radius) = arr(Yc, Xc+radius) + 1;
arr(Yc-radius, Xc) = arr(Yc-radius, Xc) + 1;
arr(Yc, Xc-radius) = arr(Yc, Xc-radius) + 1;

if (d < 0)
      d = d + (x * 4) + 6;
else
      d = d + ((x-y) * 4) + 10;
      y = y -1;
end
x = x + 1;

while x < y
      arr(Yc+x, Xc+y) = arr(Yc+x, Xc+y) + 1;
      arr(Yc+x, Xc-y) = arr(Yc+x, Xc-y) + 1;
      arr(Yc-x, Xc+y) = arr(Yc-x, Xc+y) + 1;
      arr(Yc-x, Xc-y) = arr(Yc-x, Xc-y) + 1;
      arr(Yc+y, Xc+x) = arr(Yc+y, Xc+x) + 1;
      arr(Yc+y, Xc-x) = arr(Yc+y, Xc-x) + 1;
      arr(Yc-y, Xc+x) = arr(Yc-y, Xc+x) + 1;
      arr(Yc-y, Xc-x) = arr(Yc-y, Xc-x) + 1;

      if (d < 0)
            d = d + (x * 4) + 6;
      else
            d = d + ((x-y) * 4) + 10;
            y = y -1;
      end
      x = x + 1;
end

if x==y
      arr(Yc+x, Xc+y) = arr(Yc+x, Xc+y) + 1;
      arr(Yc+x, Xc-y) = arr(Yc+x, Xc-y) + 1;
      arr(Yc-x, Xc+y) = arr(Yc-x, Xc+y) + 1;
      arr(Yc-x, Xc-y) = arr(Yc-x, Xc-y) + 1;
end
```

Όπως μπορούμε να παρατηρήσουμε αυτό που ουσιαστικά κάνει η παραπάνω εκδοχή είναι να παρακάμπτει την εκτέλεση του κυρίως μέρους του αλγορίθμου για την πρώτη φορά (x = 0, y = R) την οποία χειρίζεται ιδιαίτερα, αντιμετωπίζοντας έτσι το πρόβλημα των διπλοζωγραφισμένων σημείων στις συμμετρικές των 90^o. Επίσης στο τέλος παρατηρούμε ότι χειρίζεται ειδικά την περίπτωση που ο αλγόριθμος τερματίζει με x=y, οπότε και ζωγραφίζει ξεχωριστά τα σημεία που βρίσκονται σε συμμετρικά των 45^o.



Εικόνα 71. Κύκλοι από τη τροποποιημένη υλοποίηση Hough με ακτίνα 1 έως 16

Τα αποτελέσματα του τροποποιημένου αλγορίθμου φαίνονται στην Εικόνα 71. Όπως μπορούμε να δούμε ο αλγόριθμος υπολογίζει σωστά τα σημεία και σε κάθε περίπτωση τα ζωγραφίζει μόνο μία φορά. Η τεχνική αυτή μπορεί να χρησιμοποιηθεί χωρίς πρόβλημα για υλοποίηση του μετασχηματισμού Hough.

5.2 Προτυποποίηση του μετασχηματισμού Hough

Για να ελέγξουμε αν ο μετασχηματισμός Hough μπορεί να λύσει το πρόβλημά μας, πραγματοποιήσαμε πρόγραμμα σε Matlab™ και δοκιμαστικές εικόνες του MDT. Πρέπει να σημειωθεί ότι το Matlab™ περιέχει μία συνάρτηση που λέγεται radon και είναι ο γραμμικός μετασχηματισμός Radon. Εμείς φυσικά χρειαζόμαστε τον κυκλικό ο οποίος δεν υπάρχει.

Για να λειτουργήσει ο μετασχηματισμός Hough πρέπει να έχει γίνει ανίχνευση ακμών της εικόνας. Για τον σκοπό αυτό το Matlab™ διαθέτει την εντολή “edge” η οποία επιστρέφει μία δυαδική εικόνα (άσπρο μαύρο) με τα μόνα άσπρα σημεία, αυτά των ακμών. Η εντολή αυτή παίρνει ως παράμετρο την μέθοδο την οποία θέλουμε να χρησιμοποιήσουμε και οι μέθοδοι “canny” και “log” έδιναν τα καλύτερα αποτελέσματα.

Στην συνέχεια υλοποιήσαμε τον αλγόριθμο κυκλικού μετασχηματισμού Hough. Αυτός έχει ως εξής:

Για κάθε σημείο (x_0, y_0) ακμής της εικόνας
    Για κάθε R από $R_{ελάχιστο}$ έως $R_{μέγιστο}$
        Αύξησε κατά ένα τα σημεία του κύκλου με κέντρο (x_0, y_0) και ακτίνα R στο επίπεδο με r = R του χώρου Hough.

Για την υλοποίησή του στην αρχή χρησιμοποιήθηκε ένα .m αρχείου με εντολές της γλώσσας του Matlab™. Επειδή η γλώσσα αυτή είναι διερμηνευμένη (interpreted), εκτελείται εξαιρετικά αργά. Αυτό το μειονέκτημα μας περιόριζε την δυνατότητα πειραματισμού σε σχετικά μικρές εικόνες και περιορισμένο αριθμό ακτίνων R. Προφανώς χρειαζόταν κάποια καλύτερη τεχνική υλοποίησης.

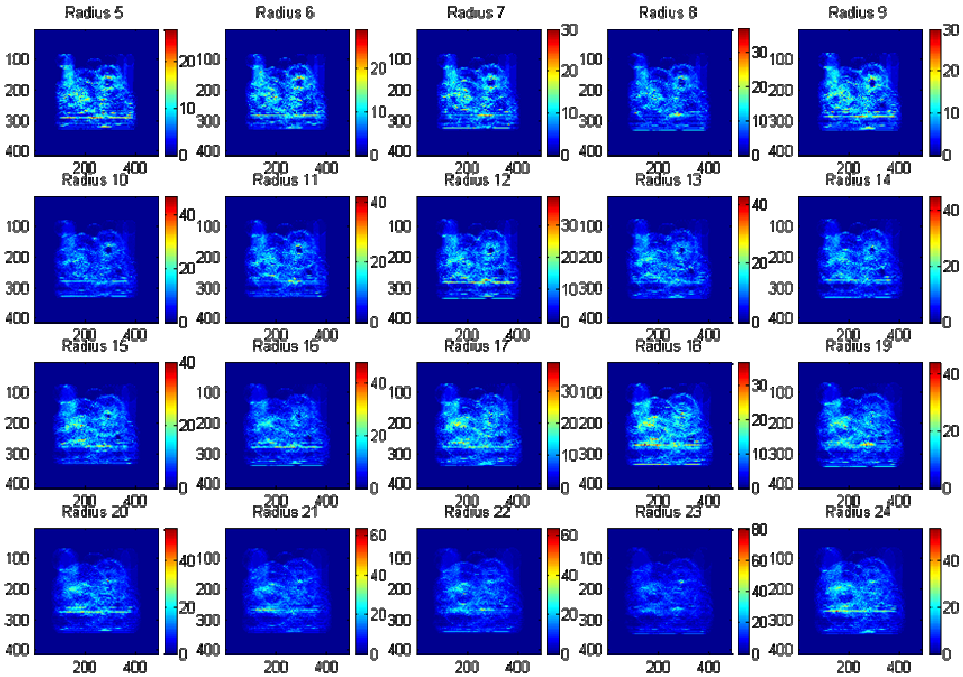
Αυτό που θα θέλαμε είναι να μπορέσουμε να γράψουμε σε κάποια γλώσσα τον αλγόριθμο και αυτός να μεταγλωττιστεί (compile) σε κώδικα μηχανής ο οποίος φυσικά θα εκτελείται πολύ ταχύτερα. Για αυτό τον σκοπό, το Matlab™ διαθέτει το πλαίσιο εργασίας MEX με την βοήθεια του οποίου μπορεί κανείς να γράφει συναρτήσεις σε C ή Fortran και να τις εκτελεί μέσα από το Matlab™ σαν κανονικές εντολές. Η εντολή μεταγλώττισης είναι η εντολή mex που καλείται μέσα από το Matlab™. Αυτή παράγει ένα αρχείο τύπου DLL (MEX file) το οποίο μπορεί να εκτελέσει το Matlab™.

Ο προγραμματισμός στο περιβάλλον εργασίας MEX δεν είναι ακριβώς όπως στη συνηθισμένη C. Για να μπορεί η ρουτίνα που γράφουμε να ανταλλάσσει δεδομένα με το Matlab™, χρειάζεται να χειρίζεται τα δεδομένα αυτά με την βοήθεια ειδικών συναρτήσεων και macros όπως και να δεσμεύει μνήμη με τη βοήθεια ειδικών συναρτήσεων (και όχι την κλασική malloc) οι οποίες δηλώνονται μέσα στο αρχείο ορισμών, "mex.h". Λεπτομερείς αναφορές για τη λειτουργία αυτού του προτύπου μπορεί κανείς να βρει στη σχετική βιβλιογραφία [81,82].

Το πρόγραμμά του μετασχηματισμού Hough, “qad.c” γίνεται compile με την εντολή “mex qad.c” και παράγεται το αρχείο MEX, qad.dll. Η βελτίωση των επιδόσεων είναι εντυπωσιακή. Το πρόγραμμα εκτελείται πάνω από 10 φορές πιο γρήγορα από την έκδοση σε γλώσσα Matlab™. Αξίζει κανείς να ασχοληθεί με το πλαίσιο εργασίας MEX αφού είναι αρκετά πλήρες και μπορεί να δώσει αποτελεσματική λύση όπου υπάρχει πρόβλημα ταχύτητας επεξεργασίας στο Matlab™.

Εικόνα 72. Εικόνα, ανίχνευση ακμών και μετασχηματισμός Hough

Με τη χρήση των αρχείων MEX έγινε δυνατή η επεξεργασία μεγαλύτερων εικόνων και με καλύτερη ακρίβεια. Στην Εικόνα 72 μπορούμε να δούμε την εικόνα, την εικόνα μετά την ανίχνευση ακμών και τον μετασχηματισμό Hough για ακτίνα ίση με την ακτίνα του MDT. Μπορούμε να παρατηρήσουμε ότι οι τιμές του χώρου Hough στην περιφέρεια βρίσκονται στην περιοχή μέχρι 600. Στο κέντρο όμως οι τιμές εκτοξεύονται πάνω από τα 1200, ακριβώς στο σημείο όπου έχουμε το κέντρο του κύκλου. Είναι προφανές ότι με την μέθοδο αυτή μπορούμε να ξεχωρίσουμε εύκολα κύκλους.



Εικόνα 73. Ο μετασχηματισμός Hough σύνθετης εικόνας

[81] MATLAB Application Program Interface Reference, V.6, ©COPYRIGHT 1984 - 2001 The MathWorks, Inc.
[82] MATLAB External Interfaces, V. 6, © COPYRIGHT 1984 - 2001 by The MathWorks, Inc.

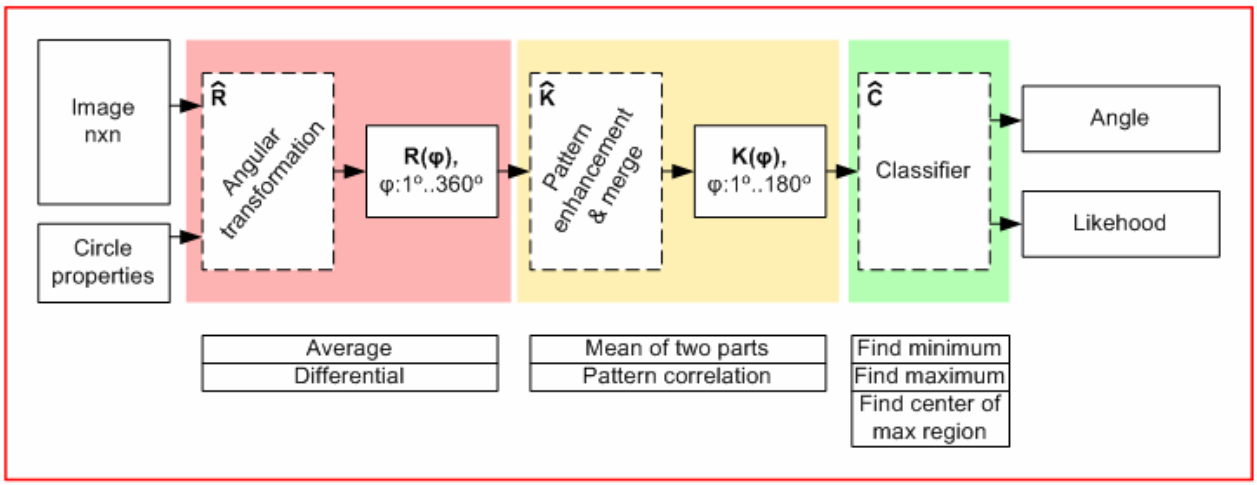
Για τον περαιτέρω έλεγχο του μετασχηματισμού και για την δημιουργία ρουτινών ανίχνευσης των μεγίστων που αντιστοιχούν στις χαρακτηριστικότερες ακτίνες της εικόνας, δημιουργήθηκε ένα πλαίσιο εργασίας με το οποίο μπορούσαν να ελεγχθούν διάφορες δοκιμαστικές εικόνες, να επιδειχθεί ο χώρος Hough και να παρουσιάζονται οι βέλτιστοι κύκλοι. Από το πλαίσιο αυτό στην Εικόνα 73 βλέπουμε τον μετασχηματισμό Hough για ακτίνες από 5 έως 24 εικονοστοιχεία. Χαρακτηριστικά μέγιστα φαίνονται για ακτίνα ίση με 23 όπου το μέγιστο είναι κατά πολύ υψηλότερο της μέσης τιμής του μετασχηματισμού.

Εικόνα 74. Βέλτιστοι κύκλοι όπως προκύπτουν από τον μετασχηματισμό

Στην Εικόνα 74 εμφανίζονται οι βέλτιστοι κύκλοι που ανιχνεύονται στο χώρο Hough. Όπως βλέπουμε αντικατοπτρίζουν κύκλους που πραγματικά υπάρχουν στην εικόνα. Επιπλέον η ανίχνευσή τους γίνεται ουσιαστικά χωρίς σφάλμα. Τα αποτελέσματα είναι εξαιρετικά. Ο κυκλικός μετασχηματισμός Hough πραγματικά μπορεί να δώσει αποτελέσματα στην εφαρμογή μας.

5.3 Υπολογισμός γωνίας οπών στήριξης του tube.

Όταν πλέον έχουμε καθορίσει πλήρως την θέση του προτύπου του MDT δηλαδή το κέντρο του και την ακτίνα του, πρέπει να υπολογίσουμε την γωνία των δύο οπών στήριξης. Η διαδικασία αυτή απαιτεί ένα δεύτερο σύστημα φίλτρων και classifier που να μπορεί να ξεχωρίζει τις οπές από τον θόρυβο.

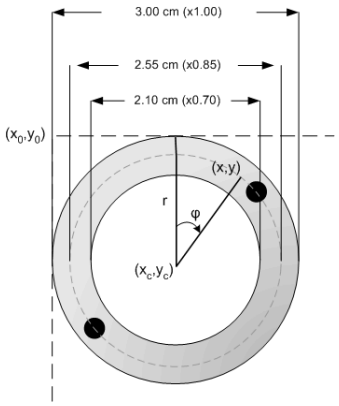


Εικόνα 75. Το block διάγραμμα του υπολογιστή γωνιών

Όπως φαίνεται στο παραπάνω διάγραμμα υπάρχουν αρκετές βαθμίδες που πρέπει να συνεργαστούν για τον υπολογισμό της γωνίας. Οι διακεκομμένες γραμμές δείχνουν συναρτήσεις ενώ οι συνεχείς, δομή δεδομένων.

5.4.1 Εργασία σε χώρο γωνιών.

Γίνεται εξαρχής προφανές ότι δεν μας συμφέρει να δουλέψουμε σε χώρο δύο διαστάσεων λόγω του μεγάλου υπολογιστικού κόστους αλλά προτιμότερο είναι να δουλέψουμε σε χώρο γωνιών. Όπως φαίνεται στο διπλανό σχήμα, αν έχουμε εντοπίσει στην εικόνα μας έναν tube στο σημείο (x_0, y_0) με διάμετρο $2 \cdot r$, τότε αυτός θα έχει κέντρο $(x_c, y_c) = (x_0 + r, y_0 + r)$. Όπως βλέπουμε από την γεωμετρία του προβλήματος, οι οπές στήριξης βρίσκονται σε απόσταση $0,85 \cdot r$ από το κέντρο του κύκλου. Συνεπώς τα σημεία που μας ενδιαφέρουν είναι τα:



Εικόνα 76. Ορισμός συντεταγμένων

$$(x, y) = \mathbf{A}(\phi) = (x_0 + r + r \cdot 0,85 \cdot \sin(\phi), y_0 + r + r \cdot 0,85 \cdot \cos(\phi), \phi \in [1^o, 360^o]$$

Με την βοήθεια της συνάρτησης $\mathbf{A}$ μπορεί να οριστεί η συνάρτηση $\hat{R}$ του πρώτου μέρους του παραπάνω διαγράμματος βαθμίδων.

Αν καθορίσουμε ένα οπτικό φίλτρο *nxn* σημείων **filter** τότε μπορούμε να το εφαρμόσουμε στην εικόνα, στο χημείο (x, y) σύμφωνα με τη σχέση:

$$(\mathbf{Image} * \mathbf{filter})(x, y) = Fi(\mathbf{Image}, \mathbf{filter}, (x, y)) = \sum_{i=-n/2}^{n/2} \sum_{j=-n/2}^{n/2} \mathbf{Image}(x - i, y - j) * \mathbf{filter}(i, j)$$

Μπορούμε λοιπόν να ορίσουμε την συνάρτηση

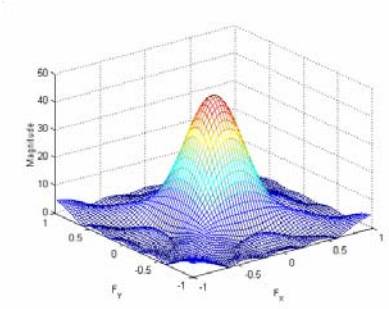
$$\hat{R}(\phi) = Fi(\mathbf{Image}, \mathbf{filter}, \mathbf{A}(\phi))$$

Με την βοήθειά της το πρόβλημα ανάγεται σε ένα μονοδιάστατο πρόβλημα μόνο της μεταβλητής φ. Ο σκοπός μας είναι να χρησιμοποιήσουμε ένα φίλτρο το οποίο θα εξάγει τα χαρακτηριστικά της εικόνας που μας ενδιαφέρουν και θα είναι όσο το δυνατόν λιγότερο ευαίσθητο στον θόρυβο.

5.4.2 Πρώτη προσέγγιση, φίλτρο φωτεινότητας.

Στην αρχή προσπαθήσαμε να ανιχνεύσουμε την φωτεινότητα του ενός σημείου (x, y). Για να μην είναι τόσο ευαίσθητο στον θόρυβο ορίζουμε ένα χαμηλοπερατό φίλτρο (blurring) **filter** της παρακάτω μορφής:

1	1	1	1	1
1	3	3	3	1
1	3	5	3	1
1	3	3	3	1
1	1	1	1	1

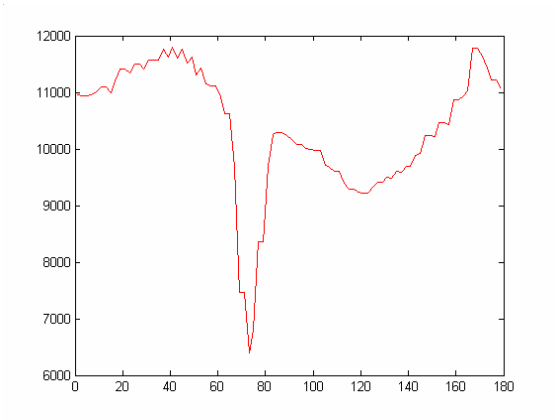


Εικόνα 77. Χαμηλοπερατό φίλτρο και η απόκριση συχνότητάς του.

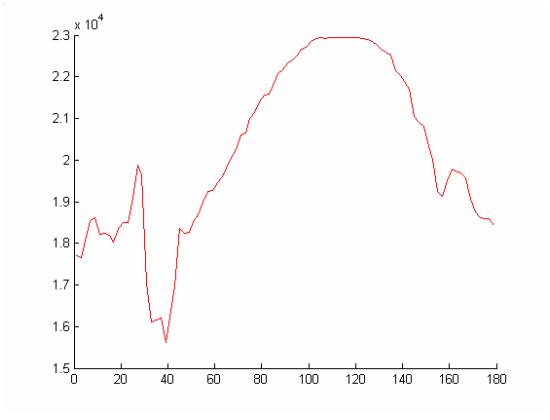
Εκμεταλλευόμενοι την συμμετρία των οπών ως προς το κέντρο, μπορούμε να μειώσουμε την ευαισθησία στον θόρυβο αθροίζοντας τις τιμές των δύο συμμετρικών σημείων για φ και φ+180^o. Η τελική συνάρτηση δίνεται από τον εξής τύπο:

$$\hat{K}_1(\phi) = \hat{R}(\phi) + \hat{R}(\phi + \pi), \phi \in [1^o, 180^o]$$

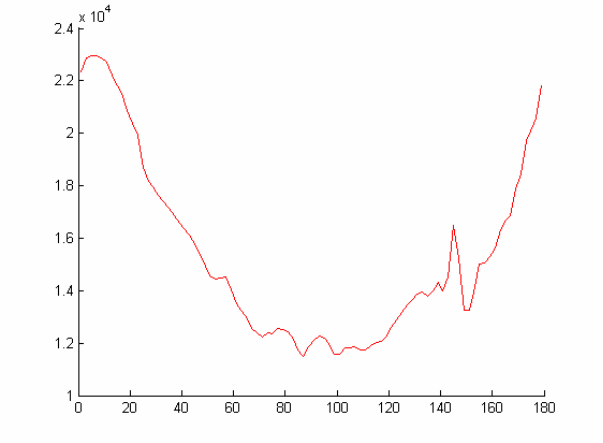
Η τιμή της συνάρτησης για τις δοκιμαστικές εικόνες φαίνεται στον παρακάτω πίνακα. Οι συναρτήσεις υλοποιήθηκαν στο m-file extractAngularFunction.m:



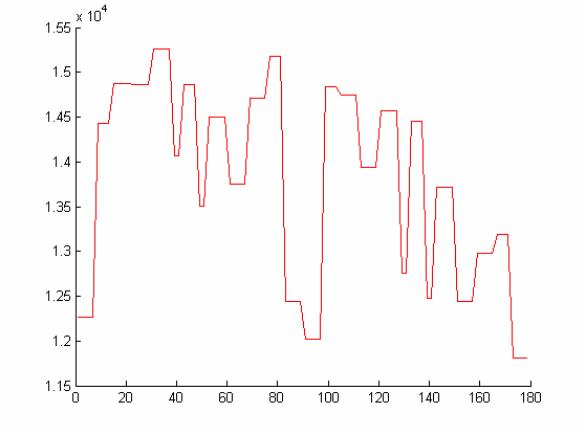
test8.bmp



test6.bmp

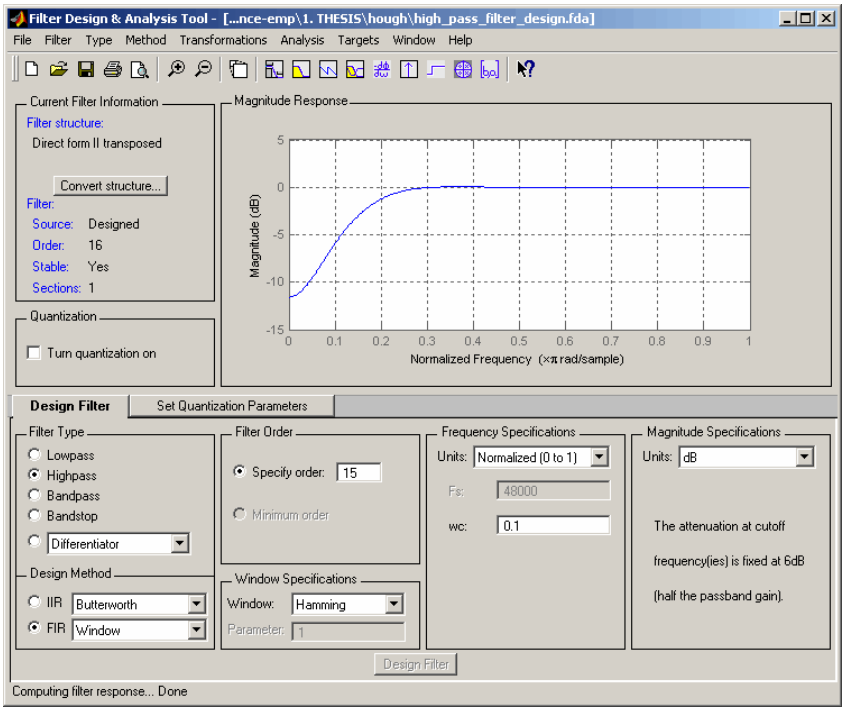


test2.bmp



test7.bmp

Στις δύο πρώτες περιπτώσεις η συνάρτηση αυτή δίνει σωστά αποτελέσματα ενώ στις δύο τελευταίες όχι. Στην περίπτωση του test2.bmp φαίνεται ξεκάθαρα η κύρια πηγή σφαλμάτων αυτής της συνάρτησης. Η έντονη ανακλαστικότητα της επιφάνειας του μετάλλου του δακτυλίου προκαλεί την εμφάνιση ενός πολύ ισχυρού σχεδόν ημιτονικού υποβάθρου σε όλα τα σήματα που βλέπουμε παραπάνω. Επειδή θα θέλαμε να έχουμε καλό αποτέλεσμα ανεξάρτητα από την ποιότητα φωτισμού θα προσπαθήσουμε να αντιμετωπίσουμε το πρόβλημα αυτό με την μέθοδο που παρουσιάζεται παρακάτω. Επιπλέον η παρατήρηση αυτή μας δίνει μία υπόνοια ότι κάποιος διαφορικός τελεστής μπορεί να είχε καλύτερα αποτελέσματα μιας και θα κατάφερνε εξαρχής να αναιρέσει το background.



Εικόνα 78. Παράμετροι σχεδίασης του υψιπερατού φίλτρου

Για την αντιμετώπιση του υποβάθρου θα χρησιμοποιήσουμε ένα υψιπερατό φίλτρο 15ης τάξης με ανόρθωση φάσης ώστε αυτή να είναι μηδέν. Το φίλτρο είναι σε θέση να ξεχωρίσει το σήμα από το υπόβαθρο και να αναδείξει το ελάχιστο της φωτεινότητας το οποίο αναζητούμε. Η εντολή για ένα τέτοιο φίλτρο είναι η filtfilt του Matlab™. Η σχεδίαση του φίλτρου έγινε με το fdatool του Matlab™ με τις παραμέτρους όπως φαίνεται στο διπλανό σχήμα.

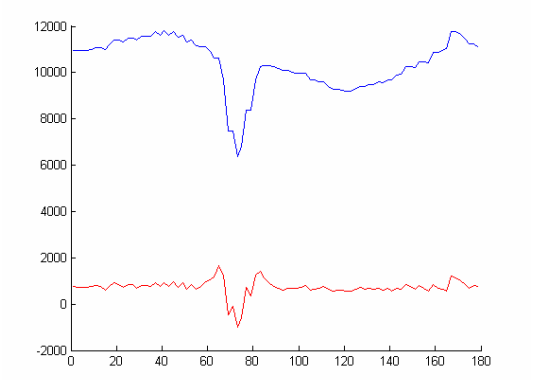
Συνεπώς η νέα συνάρτηση $\hat{K}$ είναι:

$$\hat{K}(\phi) = \hat{K}_1(\phi) * highpassFilter, \phi \in [1^o, 180^o]$$

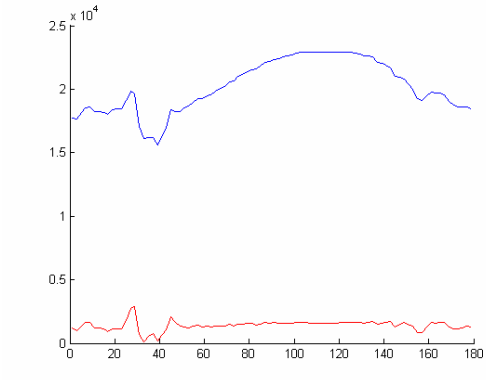
Ο classifier $\hat{C}$ ορίζεται με τον παρακάτω τρόπο:

$\hat{C}(\phi) = \phi_m \in [0, \pi)$ τέτοια ώστε $\hat{K}(\phi_m) \le \hat{K}(\phi), \forall \phi \in [1^o, 180^o]$

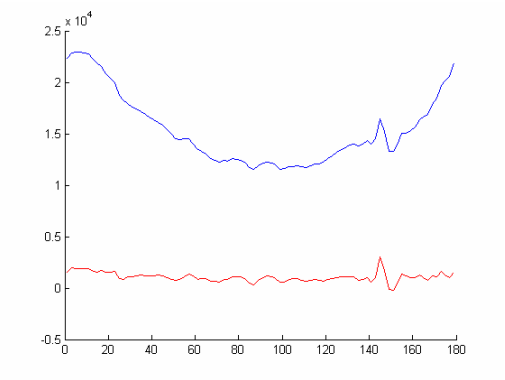
Τα αποτελέσματα της εφαρμογής της τεχνικής αυτής μαζί με την ευρεθείσα γωνία φαίνονται στους παρακάτω πίνακες. Το μπλε σήμα γραμμή είναι το αρχικό ενώ το κόκκινο είναι το φιλτραρισμένο.



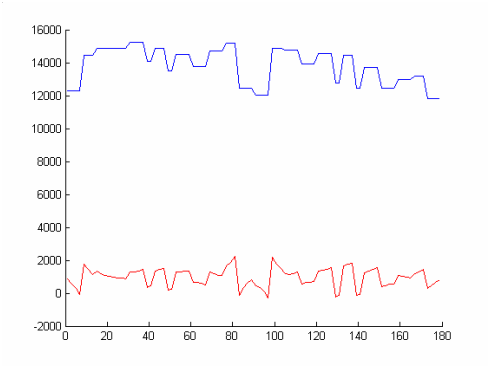
test8.bmp



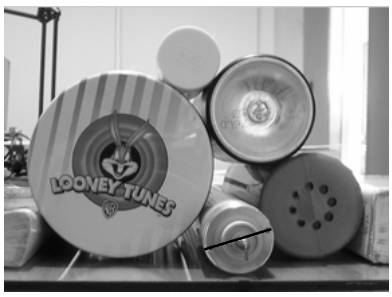
test6.bmp



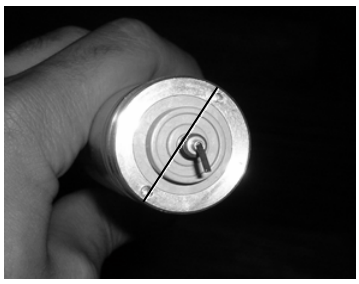
test2.bmp



test7.bmp



test8.bmp



test6.bmp

test2.bmp

test7.bmp

Εικόνα 79. Αποτελέσματα εφαρμογής μεθόδου σε δοκιμαστικές εικόνες

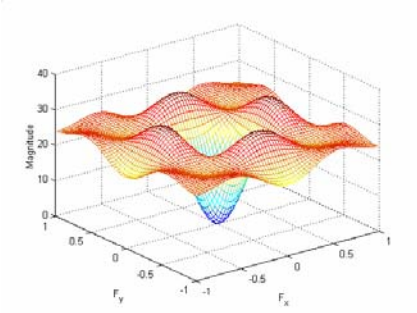
Όπως μπορούμε να δούμε τα αποτελέσματα είναι πάρα πολύ καλά. Ταυτόχρονα αναδεικνύεται όμως ένα πολύ σοβαρό πρόβλημα το οποίο εμφανίζεται ξεκάθαρα στις εικόνες test6.bmp και test2.bmp. Όπως μπορούμε να δούμε σε αυτές η ανιχνευόμενη γωνία εμφανίζεται όχι ακριβώς στο κέντρο των οπών αλλά στο σκοτεινότερο σημείο γύρω από αυτές, δηλαδή σε σημείο μίας από τις σκιές που δημιουργεί η οπή μέσα της. Αυτό είναι ένα σοβαρότατο πρόβλημα το οποίο εμφανίζεται στις περισσότερες περιπτώσεις που υπάρχει έντονος φωτισμός και το φως πέφτει κατακόρυφα στις οπές. Διαπιστώνουμε λοιπόν ότι το κέντρο των τρυπών δεν είναι το σκοτεινότερο σημείο. Το κριτήριο αυτό μπορεί να μας φτάσει μέχρις ενός σημείου ακρίβειας της τάξεως των $\pm 5^{o}$ αλλά σε καμία περίπτωση δεν μπορεί να δώσει πιο σαφή αποτελέσματα.

Στην ουσία γίνεται πλέον φανερό ότι είναι απαραίτητη η χρήση κάποιου διαφορικού τελεστή για την ανίχνευση των δύο αιχμών της κάθε οπής και η λήψη ως κέντρου της τρύπας του μέσου όρου των γωνιών των δύο αιχμών.

5.4.3 Δεύτερη προσέγγιση, γωνιακό edge detection.

Αυτή την φορά θα χρησιμοποιήσουμε ένα διαφορετικό φίλτρο για να πετύχουμε edge detection αντί για blurring. Το νέο μας φίλτρο **filter** είναι το εξής:

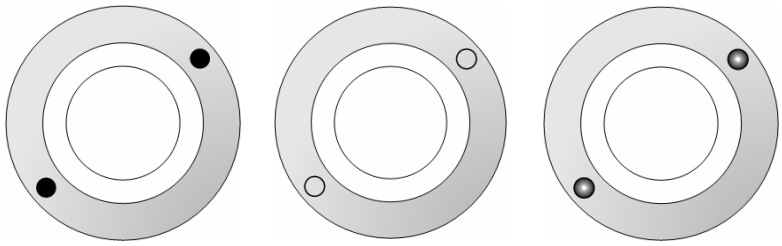
-1	-1	-1	-1	-1
-1	-1	-1	-1	-1
-1	-1	24	-1	-1
-1	-1	-1	-1	-1
-1	-1	-1	-1	-1



Εικόνα 80. Υψιπερατό φίλτρο (τελεστής edge detection) και η απόκριση συχνότητάς του

Θα προχωρήσουμε στην επεξεργασία τριών βασικών μοντέλων των οπών για να δούμε πώς αυτά απεικονίζονται από τις συναρτήσεις $\hat{R}$ και $\hat{K}_1$ με τον νέο πυρήνα και στη συνέχεια θα την εφαρμόσουμε πάνω σε πραγματικές εικόνες.

Τα τρία μοντέλα των οπών φαίνονται στον παρακάτω πίνακα. Οι συναρτήσεις της δεύτερης προσέγγισης υλοποιήθηκαν με την βοήθεια του m-file extractAngularFunction2.m:

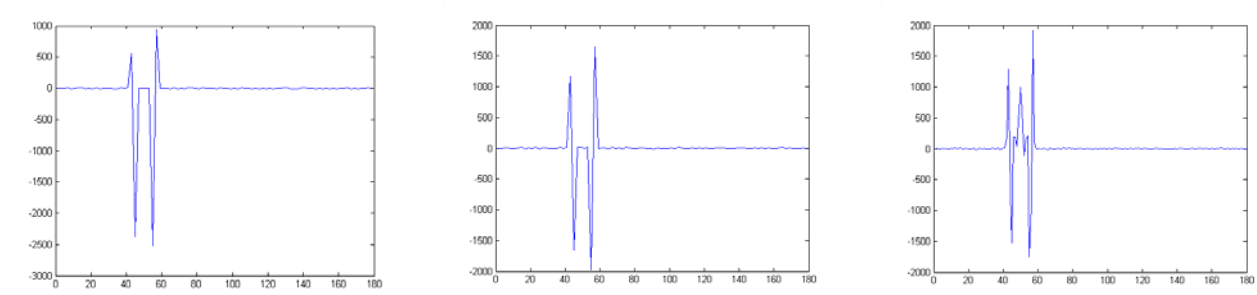


Μοντέλο 1 Μοντέλο 2 Μοντέλο 3

Εικόνα 81. Ιδανικά μοντέλα οπών

Έχουμε εφαρμόσει μία τεχνητή σκίαση για να προσομοιώσουμε τους ιριδισμούς του μετάλλου παρόλο που θα να εξασθενήσουν σημαντικά λόγω του διαφορικού τελεστή (υψιπερατό φίλτρο).

Τα αποτελέσματα της ανάλυσης φαίνονται στον παρακάτω πίνακα:



Μοντέλο 1 Μοντέλο 2 Μοντέλο 3

Εικόνα 82. Τα μοντέλα μετά την εφαρμογή του τελεστή

Όπως μπορούμε να παρατηρήσουμε τα μοντέλα 1 και 2 δεν παρουσιάζουν καμία ουσιαστική διαφορά. Το μοντέλο 3 διαφοροποιείται λίγο λόγω της καμπύλωσης και του λευκού στίγματος ανάμεσα στις δύο ακμές.

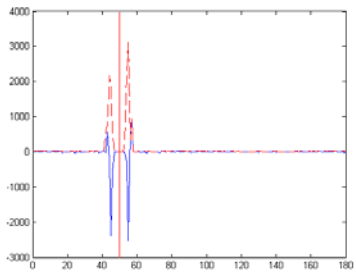
Το πιο προφανές που μπορούμε να κάνουμε για την ανάλυση των παραπάνω δεδομένων είναι να χρησιμοποιήσουμε την απόλυτη τιμή της διαφοράς του δείγματος n και του n+1. Η $\hat{K}$ θα μεγιστοποιείται στα σημεία ακμών.

Έτσι ορίζουμε την $\hat{K}(\phi) = \left|\hat{K}_1(\phi) - \hat{K}_1(\phi+1)\right|$

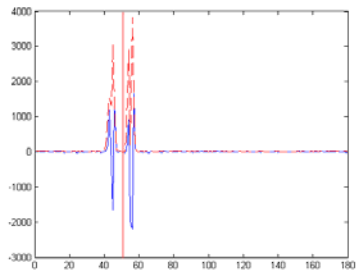
Στην συνέχεια εντοπίζουμε τα δύο μέγιστα και παίρνουμε την μέση τιμή των γωνιών στις οποίες αντιστοιχούν. Ορίζεται έτσι ο classifier $\hat{C}$ με την ακόλουθη διαδικασία:

Βρίσκουμε τις:
$\phi_{\max 1} \in [1^o, 180^o]$ τέτοια ώστε $\hat{K}(\phi_{\max 1}) \geq \hat{K}(\phi), \forall \phi \in [1^o, 180^o]$ και
$\phi_{\max 2} \in [1^o, 180^o]$ τέτοια ώστε $\hat{K}(\phi_{\max 2}) \geq \hat{K}(\phi), \forall \phi \in [1^o, 180^o] - \{\phi_{\max 1}\}$
και υπολογίζουμε την $\hat{C} = \phi_{\max} = (\phi_{\max 1} + \phi_{\max 2})/2$

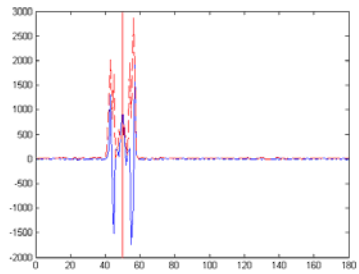
Το αποτέλεσμα της εφαρμογής της παραπάνω διαδικασίας στα μοντέλα 1-3 φαίνεται στους παρακάτω πίνακες. Η κατακόρυφη γραμμή στα διαγράμματα σημειώνει την υπολογισμένη γωνία.



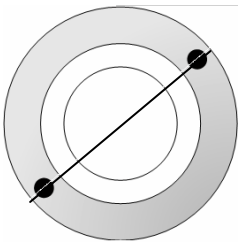
Μοντέλο 1



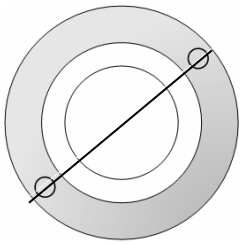
Μοντέλο 2



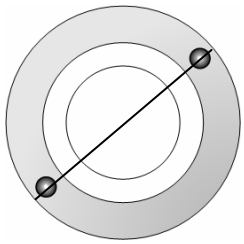
Μοντέλο 3



Μοντέλο 1

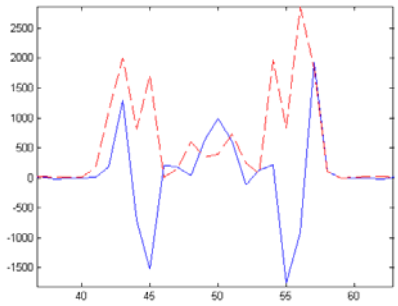


Μοντέλο 2



Μοντέλο 3

Όπως μπορούμε να παρατηρήσουμε τα αποτελέσματα είναι πάρα πολύ καλά. Αναδεικνύεται όμως από τις γραφικές παραστάσεις ένα ακόμη σημαντικό πρόβλημα. Στο μοντέλο 2 και το μοντέλο 3 παρουσιάζονται διπλές κορυφές που δεν παραπλανούν τον αλγόριθμο στα εξιδανικευμένα μοντέλα αλλά θα μπορούσαν να το κάνουν στην γενική περίπτωση. Πιο συγκεκριμένα βλέπουμε το πρόβλημα στο παρακάτω σχήμα το οποίο εμφανίζει την γραφική παράσταση της $\hat{K}$ (κόκκινη) για το μοντέλο 3.



Εικόνα 83. Κυμάτισμα στο μοντέλο 3

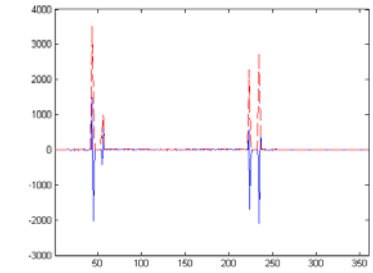
Μπορούμε να παρατηρήσουμε ότι εμφανίζεται ένα κυμάτισμα μήκους περίπου δύο μοιρών. Αυτό συμβαίνει λόγω της υπόθεσής μας ότι οι οπές βρίσκονται συμμετρικά ως προς το κέντρο, που

υλοποιείται με την συνάρτηση $\hat{K}_1$. Αυτή η υπόθεση βοηθάει στην πρώτη περίπτωση που εκτιμούσαμε την φωτεινότητα γιατί ακύρωνε τυχαίες διαφορές στην φωτεινότητα πάνω στον δακτύλιο που εμφανίζονταν από την μία πλευρά αλλά όχι στην άλλη. Στην περίπτωση όμως που χρησιμοποιούμε διαφορικό τελεστή όπως στην περίπτωσή μας όχι μόνο δεν βοηθάει αλλά δημιουργεί πρόβλημα. Μικρές αποκλίσεις στον υπολογισμό του κέντρου δημιουργούν ένα πλήθος από edges για κάθε οπή και μάλιστα πολύ κοντά η μία στην άλλη. Επιπλέον το ίδιο αποτέλεσμα μπορεί να παρουσιαστεί όταν η εικόνα του tube είναι υπό γωνία με τέτοιο τρόπο ώστε να άρει ελαφρώς τη συμμετρία ή ακόμα και λόγω της κβάντισης της τιμής του δείκτη (x,y) κατά την μετατροπή του από πραγματικό αριθμό σε ακέραιο.

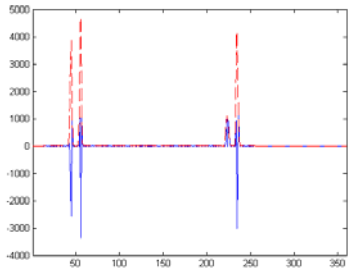
Στην νέα μας λοιπόν εκδοχή ορίζουμε την συνάρτηση $\hat{K}$ ως εξής:

$$\hat{K}(\phi) = \left|\hat{R}(\phi) - \hat{R}(\phi + 1)\right|, \phi \in [1^o, 360^o]$$

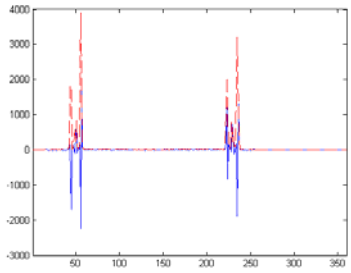
Παρατηρούμε ότι η συνάρτηση $\hat{K}$ δεν επιδρά πάνω στην συνάρτηση που χρησιμοποιούσε τη συμμετρία (K_1) αλλά κατευθείαν στην συνάρτηση φίλτρου $\hat{R}$. Επιπλέον ορίζεται για $\phi \in [1^o, 360^o]$. Τα αποτελέσματα των $\hat{R}$ (μπλε) και $\hat{K}$ (κόκκινο) για τα μοντέλα 1-3 φαίνονται παρακάτω.



Μοντέλο 1



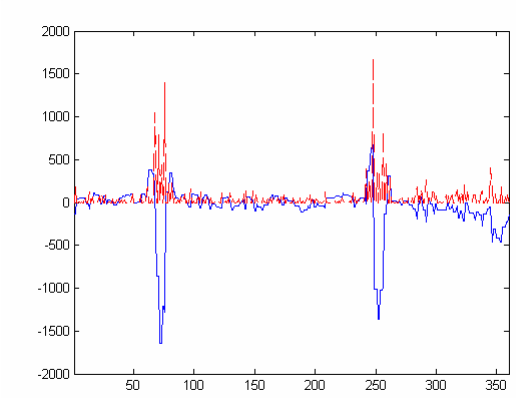
Μοντέλο 2



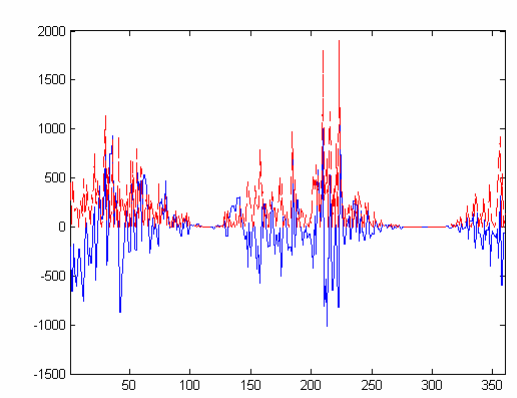
Μοντέλο 3

Μπορούμε να παρατηρήσουμε ότι τα μέγιστα είναι αυτή τη φορά απόλυτα καθαρά χωρίς κυματισμούς. Παρατηρούμε όμως ότι κάθε άλλο παρά σταθερού ύψους είναι. Βλέπουμε για παράδειγμα ότι στο μοντέλο 2 το τρίτο peak εμφανίζεται πολύ ασθενικό και υποψιαζόμαστε ότι πιθανώς θα χανόταν μέσα στο θόρυβο.

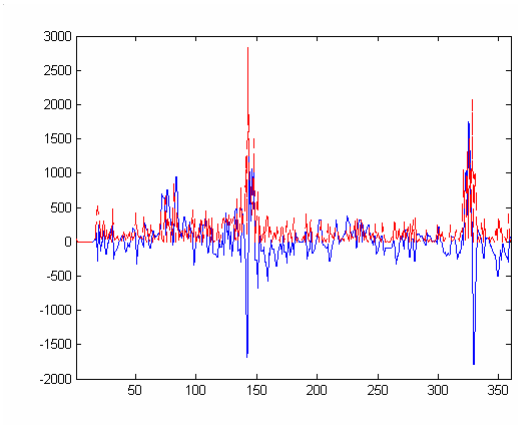
Αυτή την στιγμή χρησιμοποιώντας τα μοντέλα, έχουμε ένα πολύ σημαντικό πλεονέκτημα. Ξέρουμε ακριβώς τι ψάχνουμε να βρούμε στην $\hat{K}$ και το μόνο που μένει είναι να το ξεχωρίσουμε από το θόρυβο. Ας δούμε όμως την μορφή των $\hat{R}$ (μπλε) και $\hat{K}$ (κόκκινο) για τις πρότυπες εικόνες:



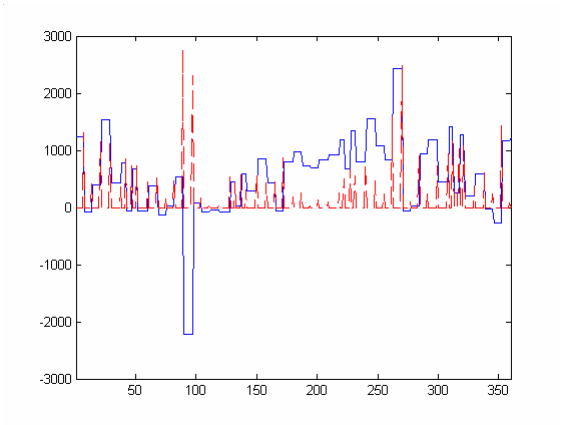
test8.bmp



test6.bmp



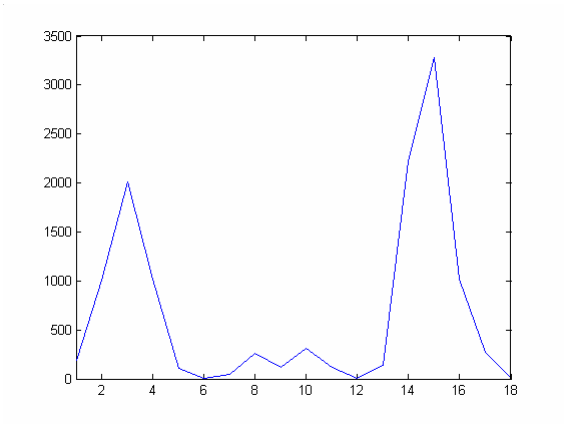
test2.bmp



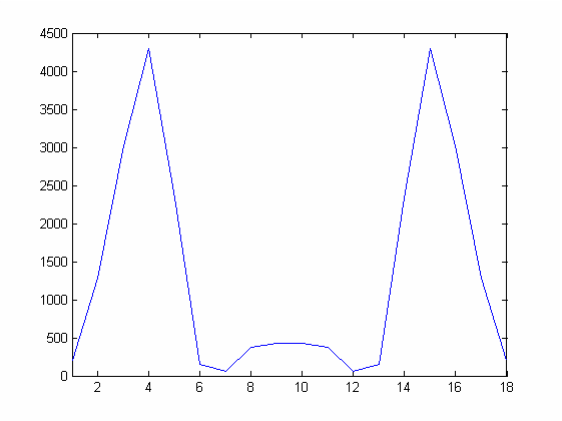
test7.bmp

Όπως μπορούμε να δούμε ο θόρυβος είναι πάρα πολύς και ενώ μπορεί κανένας να ξεχωρίσει αχνά τα πρότυπα των μοντέλων δεν μπορεί να δώσει μία σαφή εκτίμηση για τη γωνία των οπών.

Θα προσπαθήσουμε να καθαρίσουμε τις παραπάνω κατανομές αξιοποιώντας την γνώση μας από τα μοντέλα. Το πιο ρεαλιστικό μοντέλο που έχουμε είναι το μοντέλο 3. (Η παραγωγή του προτύπου γίνεται με την βοήθεια του createPattern.m).



Πρότυπο οπής



Συμμετρική έκδοσή του

Προφανώς εμείς θεωρούμε ιδανικά ότι η οπή εμφανίζει συμμετρικά edges και συνεπώς κάνουμε την συνάρτηση συμμετρική με την βοήθεια της σχέσης:
$patt_{sym}(x) = patt(x) + patt(n - x)$ όπου n το πλάτος του προτύπου. Η συμμετρική εκδοχή φαίνεται στην παραπάνω εικόνα.

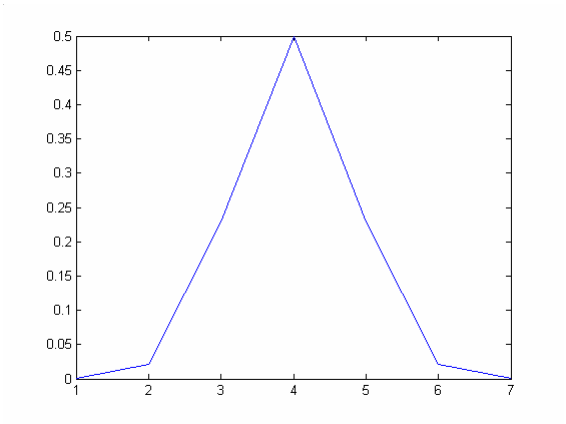
Προφανώς επίσης θα πρέπει να έχουμε μία ανοχή και να θεωρήσουμε ότι το παραπάνω πρότυπο θα συναντάται με κάποιο σφάλμα στις κατανομές. Θεωρούμε κανονική κατανομή των σφαλμάτων και εξομαλύνουμε κατάλληλα το πρότυπό μας. Συγκεκριμένα θεωρούμε μία κατανομή της μορφής:

$$p(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-(t-\mu)^2}{2\sigma^2}} dt$$

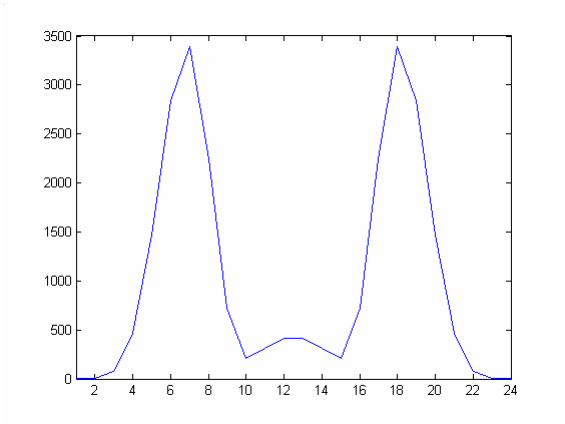
και δημιουργούμε 7 δείγματα της παραπάνω κατανομής με μ = 4 και σ = 0.8. Η κατανομή αυτή φαίνεται στο παρακάτω σχήμα. Συνυπολογίζουμε τα σφάλματα αυτά στο πρότυπο μας πραγματοποιώντας την συνέλιξη του προτύπου με την συνάρτηση σφάλματος:

$$pattern(x) = patt_{sym} * p = \sum_{k=1}^{N} p(k) * patt_{sym}(x - k)$$

Το πρότυπο με τα σφάλματα φαίνεται και αυτό στην παρακάτω εικόνα:

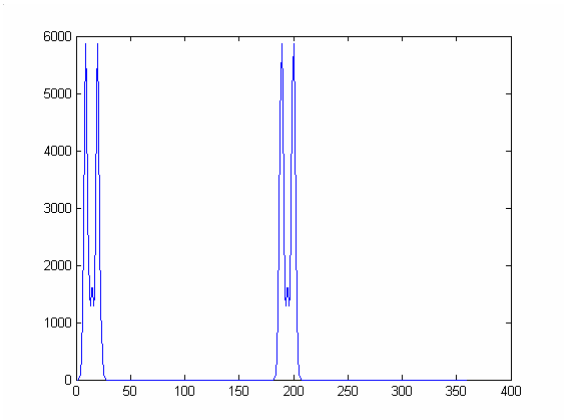


Ακολουθία 7 δειγμάτων της κανονικής κατανομής σφάλματος



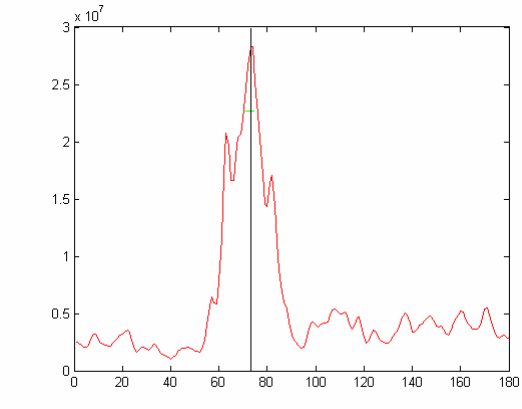
Πρότυπο με σφάλματα.

Αναμένουμε το πρότυπο αυτό να εμφανίζεται σε συμμετρικές γωνίες. Συνεπώς η τελική μορφή του προτύπου είναι:

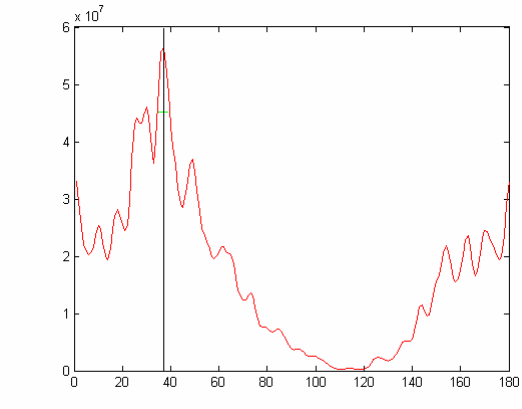


Εικόνα 84. Η τελική μορφή του προτύπου

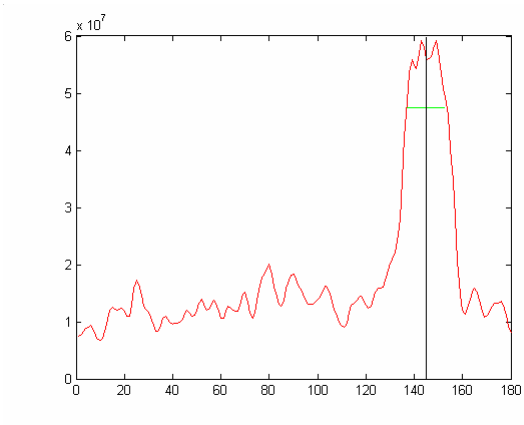
Τώρα το μόνο που μένει είναι να υπολογίσουμε την συσχέτιση (cross correlation) των συναρτήσεων $\hat{K}$ και του μοντέλου.



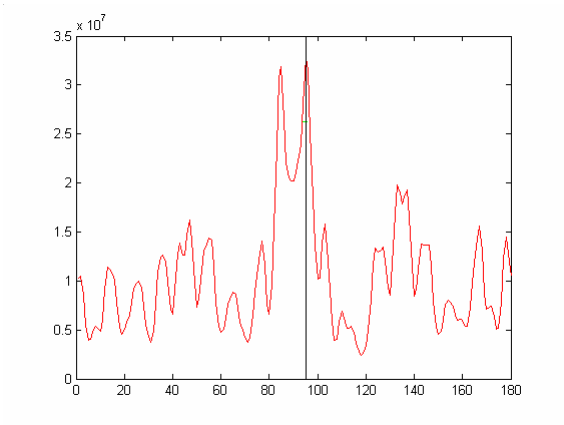
test8.bmp



test6.bmp



test2.bmp

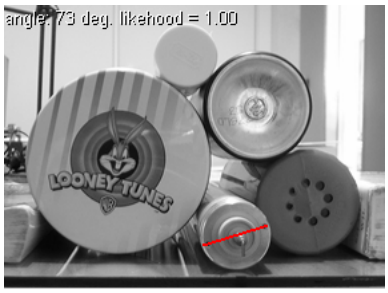


test7.bmp

Όπως μπορούμε να παρατηρήσουμε το σήμα έχει καθαρίσει σε πολύ μεγάλο βαθμό και το κυριότερο είναι ότι οι κορυφές που μας ενδιαφέρουν εμφανίζονται εξαιρετικά καθαρά.

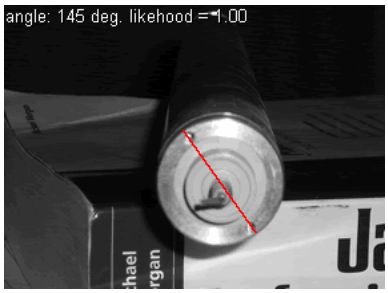
Για να προσδιορίσουμε το μέγιστο βρίσκουμε τη μέγιστη τιμή η οποία έστω ότι εμφανίζεται στην γωνία ϕ_0. Έστω K_0 η τιμή της $\hat{K}$ στο ϕ_0. Θέτουμε έναν συντελεστή χαλάρωσης $R = 0.8$. Βρίσκουμε τις γωνίες ϕ_1, ϕ_2 που ορίζουν ένα σύνολο $\Phi = [\phi_1\ \phi_2]$ τέτοιο ώστε $\phi_0 \in \Phi$ και $\hat{K}(\phi) > R \cdot K_0, \forall \phi \in \Phi$. Τέλος παίρνουμε τον μέσο όρο των γωνιών ϕ_1, ϕ_2. $\hat{C}_{relaxed} = (\phi_1 + \phi_2)/2$.

Οι γωνίες που υπολογίζονται με αυτόν τον τρόπο φαίνονται στον παρακάτω πίνακα.

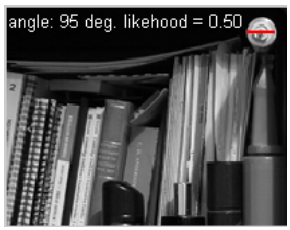


test8.bmp

test6.bmp



test2.bmp



test7.bmp

Προφανώς αυτή είναι η καλύτερη τεχνική από όσες εξετάστηκαν μέχρι στιγμής.

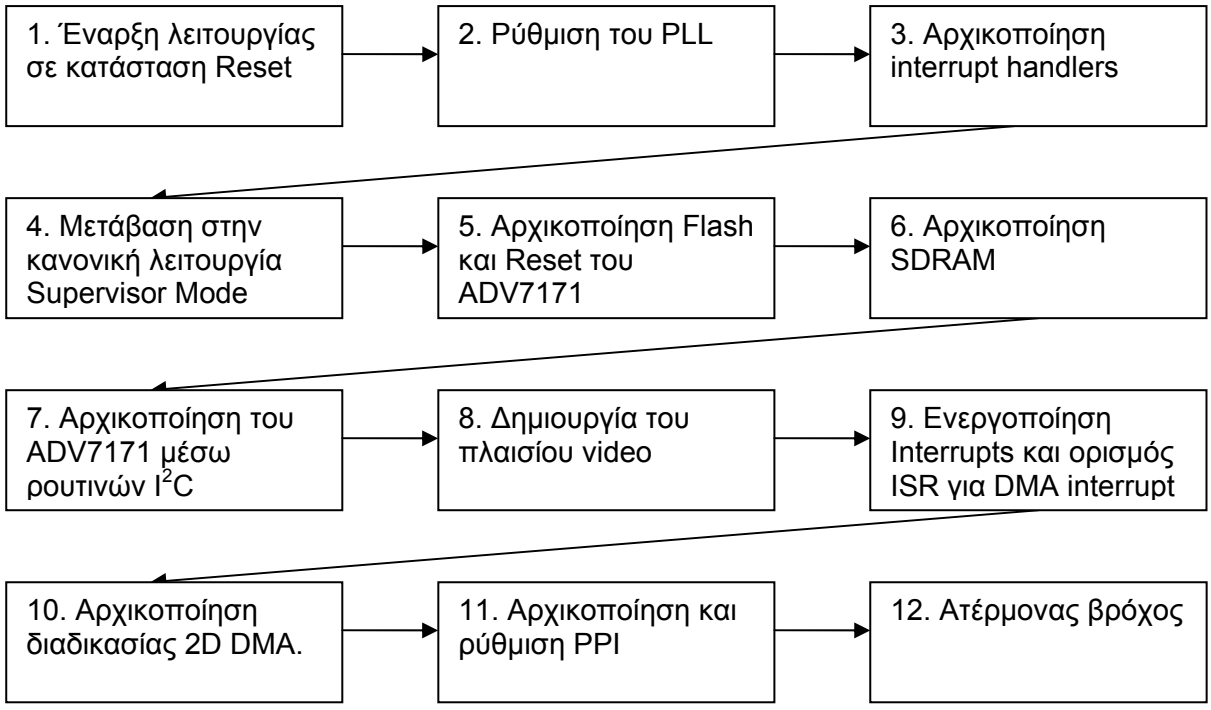
Κεφάλαιο 6. Ανάπτυξη στο DSP

Στο κεφάλαιο αυτό παρουσιάζεται η τελική υλοποίηση στο DSP. Το κεφάλαιο αυτό έρχεται ως συνέχεια των κεφαλαίων 3 και 5 για να παρουσιάσει ένα πρακτικό παράδειγμα υλοποίησης ενός συστήματος DSP. Το κεφάλαιο αυτό θα φανεί πολύ χρήσιμο σε όποιον θέλει να αναπτύξει ένα σύστημα DSP αφού θα γνωρίσει από κοντά όλα τα πράγματα που θα πρέπει να προσέξει κατά την υλοποίησή του.

6.1 Παράδειγμα δημιουργίας πλαισίου Video

Το περιοδικό Ελέκτορ (κατάλογος 06_ELEKTOR_EXAMPLE_030439-11) είχε δημοσιεύσει το παράδειγμα ενός προγράμματος για BlackFin® που παρήγαγε ένα πρότυπο με έγχρωμες στήλες στο σύστημα PAL. Το πρόγραμμα αυτό ήταν βασισμένο στο αντίστοιχο παράδειγμα του επεξεργαστή BlackFin® που έρχεται με το VisualDSP++. Θα το μελετήσουμε για να δούμε τι κάνει μιας και πάνω του θα βασίσουμε την παραγωγή σήματος video. Πρέπει να σημειωθεί ότι το αρχικό πρόγραμμα είχε ορισμένα λάθη τα οποία έδιναν ένα μερικό ασυγχρόνιστο σε ορισμένους δέκτες τηλεόρασης και τα οποία διορθώθηκαν στη πορεία (κατάλογος 01_PAL_REFERENCE).

Η διαδικασία είναι λεπτομερής και το πρόγραμμα γραμμένο σε assembly. Αυτό θα μας βοηθήσει να καταλάβουμε τις ρυθμίσεις που χρειάζονται για το σύστημα PAL αλλά και για τη λειτουργία και αρχικοποίηση του BlackFin. Η διαδικασία σχηματικά έχει ως εξής:



1. Στην αρχή ο επεξεργαστής βρίσκεται σε κατάσταση Reset και εκτελεί τον κώδικα αρχικοποίησης.

2. Το πρώτο βήμα του κώδικα αρχικοποίησης είναι να ρυθμίσει την λειτουργία του PLL. Συγκεκριμένα στον καταχωρητή PLL_CTL δίνεται η τιμή 0x2C00 που σημαίνει πολλαπλασιασμός επί 22, δηλαδή λειτουργία πυρήνα στα 27 * 22 = 594MHz. Κάθε αλλαγή του λόγου πολλαπλασιασμού του PLL χρειάζεται κάποιο χρόνο για να κλειδώσει κατά την οποία ο επεξεργαστής είναι καλό να μην εργάζεται. Ο χρόνος αυτός καθορίζεται με τη μεταβλητή PLL_LOCKCNT η οποία έχει τιμή 0x0200 που σημαίνει ότι αναμένεται ότι το PLL θα έχει κλειδώσει μετά από 512 κύκλους του ρολογιού συστήματος. Για όση διάρκεια περιμένουμε το PLL να κλειδώσει, ο επεξεργαστής τίθεται στη κατάσταση ύπνου με την εντολή IDLE. Το bit 0 (PLL_WAKEUP) του καταχωρητή SIC_IWR ενεργοποιείται για να επιτρέψει στον επεξεργαστή να

ξυπνήσει μετά το πέρας PLL_LOCKCNT κύκλων συστήματος. Αν η διαδικασία αυτή δεν γινόταν αμέσως μετά το Reset, θα έπρεπε να απενεργοποιούσαμε τα interrupts για να μην ξυπνήσει κατά λάθος ο επεξεργαστής. Όταν περάσει ο χρόνος σταθεροποίησης η τιμή του καταχωρητή κατάστασης PLL (PLL_STAT) είναι 0x0022 που σημαίνει ότι ο επεξεργαστής είναι σε κατάσταση πλήρους λειτουργίας (FULL_ON) και το PLL έχει κλειδώσει (PLL_LOCKED).

Στη συνέχεια η συχνότητα διαίρεσης του ρολογιού συστήματος τίθεται μέσω του καταχωρητή PLL_DIV στην τιμή 6 δηλαδή συχνότητα του PLL δια 6 που αντιστοιχεί σε 99 MHz. Η συνήθης τιμή για αυτή τη μεταβλητή είναι 5 δηλαδή λειτουργία στα 118.8 MHz. Δοκιμάσαμε και αυτή τη ρύθμιση και το σύστημα λειτουργούσε εξίσου καλά. Ίσως η τιμή διαίρεσης / 6 δόθηκε για λόγους εξοικονόμησης ενέργειας. Για τις αλλαγές των διαιρετών μετά το PLL δε χρειάζεται κάποια επίπονη διαδικασία όπως αυτή του PLL.

3. Στη συνέχεια γίνεται αρχικοποίηση των Interrupt Handlers (ρουτινών εξυπηρέτησης διακοπών). Αυτό γίνεται για λόγους ασφαλείας και στις περισσότερες περιπτώσεις το αποτέλεσμα είναι η παύση της λειτουργίας στο συγκεκριμένο σημείο. Σε περίπτωση exception ανάβουν όλα τα λαμπάκια και παύει η λειτουργία. Φυσικά κάτι τέτοιο δεν πρέπει να συμβαίνει κατά την ορθή λειτουργία του επεξεργαστή αλλά είναι πολύ χρήσιμο για την εύρεση σφαλμάτων. Όταν συμβαίνει αυτό μπορούμε να διαβάσουμε τον κώδικα σφάλματος στα πρώτα 6 bits του καταχωρητή SEQSTAT και να τα ερμηνεύσουμε με βάση τον σχετικό πίνακα του manual[83]. Η διεύθυνση της εντολής από την οποία προκλήθηκε το σφάλμα βρίσκεται στον καταχωρητή RETX. Με αυτόν τον τρόπο μπορούν σχετικά εύκολα να βρεθούν και να διορθωθούν απλά σφάλματα.

4. Μετά καλείται η ρουτίνα main σε κατάσταση Supervisor. Αυτό γίνεται με το εξής «κόλπο». Γίνεται αρχικοποίηση του supervisor stack pointer στην τιμή sp = 0xFF90 5400 που βρίσκεται στην Data Bank B της εσωτερικής μνήμης. Καταχωρείται ο Interrupt Handler για το event15 (software interrupt). Ενεργοποιείται στον καταχωρητή IMASK η αντίστοιχη διακοπή με τη βοήθεια της εντολής STI (sti 0x8000). Γίνεται ενεργοποίηση του Interrupt με τη βοήθεια της εντολής raise (raise 15). Ο κώδικας που εκτελείται πλέον είναι σε supervisor mode και είναι η ρουτίνα διακοπής Interrupt handler που αμέσως πριν ορίσαμε. Αυτή πηγαίνει και εκτελεί την ρουτίνα _main η οποία εκτελείται σε supervisor mode και έχει πλήρη δικαίωμα να τροποποιήσει όλους τους MMRs και την μνήμη (goto _main).

5. Η ρουτίνα _main το πρώτο που κάνει είναι να ενεργοποιεί τη μνήμη Flash με τις κατάλληλες ρυθμίσεις του EBIU και στη συνέχεια μέσω μίας ακίδας της κάνει reset στο ADV7171. Για τη λειτουργία της Flash τίθενται οι τυπικές τιμές των καταχωρητών EBIU_AMBCTL0 = 0x7bb07bb0 και EBIU_AMBCTL1 = 0x7bb07bb0 όπως και γίνονται 1 τα 4 λιγότερο σημαντικά ψηφία του καταχωρητή EBIU_AMGCTL (EBIU_AMGCTL|=0x0f). Τώρα πλέον που η επικοινωνία με την Flash έχει αποκατασταθεί, γίνεται reset στον κωδικοποιητή video. Αυτό γίνεται στην αρχή καθαρίζοντας τον καταχωρητή δεδομένων της πόρτας της FLASH (portA_data_out = 0), στην συνέχει θέτει την κατεύθυνση των ακίδων της θύρας αυτής ως εξόδους (portA_data_dir = 0xffff) και τέλος κάνει μία low to high μετάβαση της ακίδας RESET του κωδικοποιητή video (portA_data_out = RST_7171 (= 0x2)).

6. Στη συνέχεια εξασφαλίζεται ότι έχει αρχικοποιηθεί η SDRAM με τις κατάλληλες ρυθμίσεις του EBIU. Αυτό γίνεται ελέγχοντας το bit 3 (SDRS) του καταχωρητή κατάστασης EBIU_SDSTAT. Αν αυτό είναι 0 σημαίνει ότι έχει γίνει ορθή αρχικοποίηση. Στην αντίθετη περίπτωση γίνεται αρχικοποίηση των παραμέτρων της SDRAM (EBIU_SDRRC = 0x0817, EBIU_SDBCTL = 0x00000013, EBIU_SDGCTL = 0x0091998d την τιμή του οποίου μετά διαβάζουμε 0x8011998d). Την SDRAM την χρειαζόμαστε γιατί εκεί δημιουργούμε το πλαίσιο video.

7. Μία ακολουθία τιμών κατάλληλων για λειτουργία σε κατάσταση PAL αποστέλλεται στον κωδικοποιητή video μέσω μίας υλοποίησης του πρωτοκόλλου I^2C με λογισμικό. Για την ανάγκη της επικοινωνίας αυτής γίνεται αρχικοποίηση της κατεύθυνσης των PFx ακίδων. Τα PF0-3 είναι έξοδοι ενώ τα υπόλοιπα είσοδοι (FIO_DIR = 0x0f).

[83] Πίνακας 4-11, σελίδα 4-43, ADSP-BF533 Blackfin™ Processor Hardware Reference

8. Στη συνέχει δημιουργείται ένα πλήρες πλαίσιο ψηφιακού video στην μνήμη. Αυτό γίνεται χρησιμοποιώντας ένα πίνακα βασικών χρωμάτων, ένα πίνακα κωδικών EAV/SAV και ενός πίνακα που «μαρκάρει» ποιες γραμμές είναι γραμμές κατακόρυφου συγχρονισμού. Η υλοποίηση του τελευταίου πίνακα είναι πολύ αποδοτική αφού κάθε bit των 20 dwords του πίνακα PalLineMap αντιστοιχεί σε μία γραμμή video (20 dwords x 32bit = 640 γραμμές εκ των οποίων χρησιμοποιούνται μόνο οι πρώτες 625).

9. Στη συνέχεια τίθεται ο interrupt handler (EVT8 = &DMA_ISR) για τη λειτουργία DMA του DMA controller της PPI και ενεργοποιούνται τα αντίστοιχα Flags στους καταχωρητές ενεργοποίησης διακοπών του ελεγκτή συστήματος (SIC_IMASK= 0x00000100) και του ελεγκτή πυρήνα (IMASK |= IVG8). Η τιμή που μετράμε μετά από αυτό για την μεταβλητή IMASK είναι 0b10000001 00011111 (0x811f) που σημαίνει ότι είναι ενεργοποιημένα εκτός από τα βασικά interrupts (NMI, RESET κ.τ.λ.) τα interrupts IVG8 και IVG15. Το πρώτο είναι για το κανάλι DMA που μόλις ενεργοποιήσαμε και το δεύτερο είναι το software interrupt που χρησιμοποίησε η ρουτίνα reset (βήμα 4).

10. Στη συνέχει ενεργοποιείται η λειτουργία DMA. Αυτή λειτουργεί στην δισδιάστατη μορφή της αφού μόνό έτσι μπορεί να καλύψει τα 1072500 bytes του πλαισίου video που πρέπει να μεταφερθούν. Η διεύθυνση έναρξης της μετάδοσης DMA είναι εκεί που αρχίζει το πλαίσιο δηλαδή στη θέση στατικής μνήμης 0 (DMA0_START_ADDR = 0). Η ρύθμιση της λειτουργίας του γίνεται με τη βοήθεια του καταχωρητή DMA0_CONFIG. Συγκεκριμένα του ανατίθεται η τιμή 0x1091 που σημαίνει ενεργοποίηση, μεταφορά 2D, ενεργοποίηση interrupt στη λήξη αποστολής πλαισίου (outer loop) και autobuffer mode που σημαίνει ότι η λειτουργία DMA θα επαναλαμβάνεται αυτόματα. Οι τιμές του Inner loop τίθενται σε DMA0_X_COUNT = 0x06b4 = 1716 (σημεία ανά γραμμή) και DMA0_X_MODIFY = 1 (αύξηση κατά ένα) ενώ στην outer loop τίθενται DMA0_Y_COUNT = 0x0271 = 625 (γραμμές) και DMA0_Y_MODIFY = 1 (αύξηση κατά 1).

11. Το PPI ενεργοποιείται και τίθεται στην λειτουργία αποστολής (PPI_CONTROL = PORT_DIR| PORT_EN = 3) σε κατάσταση ITU656. Από το σημείο αυτό και μετά αρχίζει η αποστολή δεδομένων video και εμφανίζεται η εικόνα στην οθόνη.

12. Στο τέλος υπάρχει ο κλασικός ατέρμονος βρόχος που εμφανίζεται σε όλους σχεδόν τους μικροελεγκτές.

Πρέπει να σημειωθεί ότι ο DMA interrupt handler καθαρίζει το σημάδι του interrupt στον καταχωρητή DMA0_IRQ_STATUS (DMA0_IRQ_STATUS|= 1). Αν δεν το έκανε αυτό με το που θα τέλειωνε η εκτέλεσή του θα ξανακαλείτω αμέσως.

Πρέπει να αναφέρουμε ακόμη μία λεπτομέρεια. Κάθε αρχείο πριν από τον κώδικα έχει μία δήλωση τύπου ".section". Συγκεκριμένα υπάρχει η δήλωση ".section STARTUP_SECTION" της οποίας ακολουθεί η συνάρτηση Start που καλείται μετά το reset και κάνει αρχικοποιήσεις. Επιπλέον υπάρχει η δήλωση ".section program" όπου υπάρχει η συνάρτηση _PalVideoOutFrameBuffInit. Όλα τα άλλα κομμάτια κώδικα έχουν για section το "L1_code". Επιπλέον οι πίνακες δεδομένων έχουν ως section το "L1_data_a". Αυτά τα τμήματα (sections) έχουν ως σκοπό να ομαδοποιήσουν οντότητες (δεδομένα και ρουτίνες) τα οποία θα πρέπει να αποθηκευτούν σε μνήμη με συγκεκριμένες ιδιότητες και θέση. Η αντιστοίχηση των τμημάτων αυτών (input sections) με τα αντίστοιχα πραγματικά τμήματα μνήμης (output sections) γίνεται με τη βοήθεια ενός αρχείου .ldf το οποίο απευθύνεται στον linker και που στην περίπτωση του προγράμματος που εξετάζουμε είναι το "ADSP-BF533_ASM.ldf". Αν το εξετάσει κανείς θα δει ότι είναι αρκετά μικρό και απλό. Για παράδειγμα για τα input section "L1_code" και "program" έχουμε την αντιστοίχηση:

```
MEMORY
{
      PROGRAM { TYPE(RAM) START(0xFFA00000) END(0xFFA04FFF) WIDTH(8) }
}


PROCESSOR p0
{
```

```
      program
        {
            INPUT_SECTION_ALIGN(4)
            INPUT_SECTIONS( $OBJECTS(program) $LIBRARIES(program))
            INPUT_SECTIONS( $OBJECTS(L1_code) $LIBRARIES(L1_code))
        } >PROGRAM
}
```

Βλέπουμε λοιπόν ότι τα δύο αυτά τμήματα αντιστοιχίζονται στο section PROGRAM το οποίο έχει ευθυγράμμιση $4^{ων}$ byte (INPUT_SECTION_ALIGN(4)) που σημαίνει ότι οι διευθύνσεις έναρξης οντοτήτων αυτής της μνήμης θα έχουν διεύθυνση ακέραιο πολλαπλάσιο των $4^{ων}$ bytes δηλαδή τα δύο λιγότερο σημαντικά ψηφία τους θα είναι μηδέν. Τέλος η μνήμη αυτή θα είναι στις διευθύνσεις μνήμης από 0xFFA00000 έως 0xFFA04FFF δηλαδή σε ένα μικρό μέρος (20 Kbytes από τα 64 διαθέσιμα) της Instruction SRAM του επεξεργαστή. Αυτό φυσικά θα μπορούσε να προκαλέσει και προβλήματα για κάποιον ο οποίος θα ήθελε να εξελίξει το παράδειγμα. Μετά από μερικές σελίδες κώδικα θα εμφανιζόταν σφάλμα στον προγραμματιστή που θα τον πληροφορούσε ότι ο κώδικάς του δεν χωράει στην μνήμη. Αυτός τότε θα έπρεπε να τροποποιήσει το αρχείο .ldf και να διορθώσει το πρόβλημα.

6.2 Μέτρηση χρόνου και συμπεράσματα

Στο βήμα 8 παράγεται προγραμματιστικά ένα πλαίσιο video δηλαδή κάτι παραπάνω από 1Mbyte (1072500 bytes). Για την επίτευξη αυτού του στόχου χρησιμοποιούνται πάρα πολύ αποδοτικά οι βρόχοι υλοποιημένοι στο hardware και οι δείκτες της βαθμίδας DAG. Θέλουμε να ελέγξουμε πόσο γρήγορα μπορεί να γίνει η δημιουργία αυτών των δεδομένων, δεδομένου ότι δεν χρησιμοποιείται data cache. Το αποτέλεσμα ήταν ότι χρειάστηκαν 5473668 (0x0053 8584) κύκλοι πυρήνα που αντιστοιχούν σε 9.2 ms στα 594MHz δηλαδή 5.1 κύκλος ή 8.6ns ανά byte.

```
R2 = 0;
CYCLES = R2;
CYCLES2 = R2;
R2 = SYSCFG;
BITSET(R2,1);
SYSCFG = R2;

... (ότι πρέπει να μετρηθεί) ...

r5 = CYCLES;
r6 = CYCLES2;
nop; (εδώ μπαίνει breakpoint)
```

Εικόνα 85. Ακριβής τρόπος μέτρησης κύκλων

Η μέτρηση των κύκλων στον BlackFin® θέλει πολύ προσοχή. Ο μόνος αξιόπιστος τρόπος για μέτρηση των κύκλων μίας ρουτίνας είναι με τη βοήθεια του καταχωρητή cycles (64bit). Ο τρόπος χρήσης φαίνεται στην Εικόνα 85, η οποία πρέπει να εκτελεστεί με Run και όχι βήμα - βήμα. Κάθε φορά που πατάει κανείς επόμενο βήμα η pipeline αδειάζει δίνοντας ανακριβή αποτελέσματα. Η τιμή που υπολογίσαμε για τον χρόνο επαληθεύτηκε και με τη βοήθεια του Timer0 και πράγματι είναι 9.2ms (TIMER0_COUNTER = 0x0010B44B = 1094731 SCLK στα 118.8MHz = 9.2ms).

```
P0.L = lo(TIMER0_CONFIG);
P0.H = hi(TIMER0_CONFIG);
r5 = 0x41;
W[P0] = r5;
P0.L = lo(TIMER0_PERIOD);
P0.H = hi(TIMER0_PERIOD);
r5.l = 0xffff;
r5.h = 0xffff;
[P0] = r5;
P0.L = lo(TIMER0_WIDTH);
```

```
P0.L = lo(TIMER0_COUNTER);
P0.H = hi(TIMER0_COUNTER);
r3 = [P0];
```

```
P0.H = hi(TIMER0_WIDTH);
[P0] = r5;
P0.L = lo(TIMER_ENABLE);
P0.H = hi(TIMER_ENABLE);
r5 = 1;
[P0] = r5;
```

Εικόνα 86. Αρχικοποίηση Timer0 και μέτρηση χρόνου (κύκλων SCLK)

Το χρονοβόρο κομμάτι είναι αυτό που εκτελείται πάρα πολλές φορές και αυτό στην περίπτωσή μας είναι αυτό καθ' αυτό το «ζωγράφισμα» του πλαισίου. Αυτό αποτελείται από επαναληπτικούς βρόχους όπως ο παρακάτω που γεμίζει 45 (x2) γειτονικά pixels με το χρώμα της μπάρας το οποίο βρίσκεται αποθηκευμένο στον καταχωρητή R1.

```
P2 = 45;

LSETUP(PalOutputColorBarData_s, PalOutputColorBarData_e) LC1 = P2;

PalOutputColorBarData_s:
      r2 = r1 >> 24;
      b[p0++] = r2;
      r2 = r1 >> 16;
      b[p0++] = r2;
      r2 = r1 >> 8;
      b[p0++] = r2;
PalOutputColorBarData_e:b[p0++] = r1;
```

Όπως βλέπουμε χρησιμοποιούνται hardware loops και η αυτόματη αύξηση δείκτη (p0++). Αν μετρήσουμε τη ταχύτητα εκτέλεσης ενός στοιχείου από αυτά που βρίσκονται μέσα στο βρόχο θα δούμε ότι συνολικά ξοδεύονται 7 κύκλοι ρολογιού πυρήνα για την εκτέλεση κάθε επανάληψης. Αν όμως μετρήσουμε τον χρόνο εκτέλεσης όλου του βρόχου θα δούμε ότι παίρνει 878 κύκλους δηλαδή 878 / 45 = 19.5 κύκλους ανά επανάληψη. Προφανώς αυτό έρχεται σε μεγάλη ασυμφωνία με αυτό που περιμέναμε.

Ο λόγος είναι η πολύ μεγάλη καθυστέρηση κατά την εγγραφή των δεδομένων στην εξωτερική SDRAM η οποία τρέχει με το ρολόι συστήματος. Αν υπολογίσουμε πόσος χρόνος δίνεται σε κάθε επανάληψη θα βρούμε 19.5 / 594MHz = 32.8ns. Αν τον διαιρέσουμε δια τις τέσσερις εγγραφές στη μνήμη που γίνονται σε κάθε κύκλο θα βρούμε 8.2ns ανά εγγραφή byte δηλαδή περίπου ίσο με τα 8.6ns που υπολογίσαμε μετρώντας ολόκληρη την ρουτίνα ζωγραφικής (_PalVideoOutFrameBuffInit). Γνωρίζοντας ότι το ρολόι συστήματος τρέχει στα 118.8MHz υπολογίζουμε ότι ένας κύκλος είναι 8.4ns, ο χρόνος δηλαδή που υπολογίσαμε ανά byte.

Συνεπώς αυτό που καθορίζει ουσιαστικά τη ταχύτητα του συστήματος στην συγκεκριμένη εφαρμογή είναι ο χρόνος εγγραφής και ανάγνωσης από την εξωτερική μνήμη και όχι η ταχύτητα επεξεργασίας ή οι υπολογιστικές δυνατότητες του πυρήνα. Τα πράγματα θα μπορούσαν να βελτιωθούν δραματικά αν χρησιμοποιείτο η data cache η οποία τρέχει στο ρολόι πυρήνα. Για μέγιστη αξιοποίησή της θα έπρεπε ο πυρήνας να εκτελεί περισσότερες από τόσες εντολές ανά byte από τον λόγο του ρολογιού πυρήνα προς το ρολόι συστήματος (5 στην περίπτωσή μας), δηλαδή σε πιο απαιτητικές υπολογιστικά εφαρμογές.

6.3 Τροποποίηση προγράμματος για αλληλεπίδραση με χρήστη

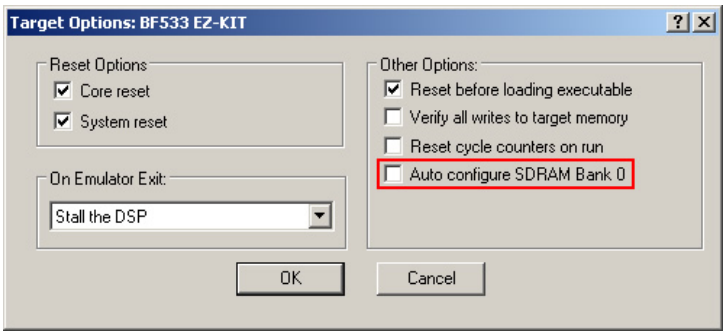
Για εξοικείωση με το πρόγραμμα αυτό το τροποποιήσαμε ώστε να λειτουργεί χωρίς το autobuffer στο DMA κανάλι (κατάλογος 02_ASM_INTERACT). Η διαδικασία DMA πρέπει να επανεκινείται από την αρχή για κάθε πλαίσιο. Με αυτόν τον τρόπο και χρησιμοποιώντας ένα από τα διαθέσιμα πλήκτρα του EZ-KIT Lite® (έστω πλήκτρο A) επιτρέψαμε να σταματάει η προβολή του προτύπου με το πάτημα ενός πλήκτρου. Με το ταυτόχρονο πάτημα ενός δεύτερου πλήκτρου (έστω B), ο κωδικοποιητής video με τη βοήθεια μίας ακολουθίας I^2C μπαίνει στην κατάσταση αυτόματης δημιουργίας του ιδίου προτύπου που έχει όμως τα χρώματα σε αντίθετη σειρά. Αυτή η κατάσταση

δεν επηρεάζεται από τη DMA λειτουργία και συνεπώς δεν διακόπτεται όταν πατάμε το πλήκτρο A. Αν ξαναπατηθεί ο ίδιος συνδυασμός πλήκτρων (A και B μαζί) το σύστημα ξαναγυρίζει στην κανονική λειτουργία όπου το A ρυθμίζει το αν θα μεταδίδεται η εικόνα. Η κατάσταση εμφανίζεται με τη βοήθεια τριών Leds.

Από τη λειτουργία της παραπάνω εφαρμογής αποδείχθηκε ότι το σύστημα δε μπορεί να χρησιμοποιηθεί για τη ταυτόχρονη σύλληψη και αναπαραγωγή εικόνας σε μεγάλες ταχύτητες. Αυτό γιατί κάθε φορά που διακόπτεται και επανεκινείται η DMA λειτουργία μετάδοσης εικόνας με το πάτημα του πλήκτρου, ο συνδυασμός τηλεόρασης και κωδικοποιητή video προκαλεί μία καθυστέρηση για περισσότερο από ¼ του δευτερολέπτου (μέχρι να κλειδώσει η εικόνα στο νέο σήμα με χρώμα). Συνεπώς ακόμα και αν δεσμευθεί ελάχιστος χρόνος για την σύλληψη ενός frame video θα υπάρχει εμφανής συνέπεια στην εικόνα που είναι και πολύ κουραστική για τα μάτια. Η λύση για το πρόβλημα αυτό είναι χωρίς αμφιβολία ο νέος BlackFin BF561® αφού διαθέτει δύο ανεξάρτητες θύρες PPI και διπλάσια επεξεργαστική ισχύ.

6.4 Μετάφραση σε C και βασικές λειτουργίες γραφικών

Επειδή το πρόγραμμά μας έχουμε αποφασίσει ότι θέλουμε να γραφεί σε C, είναι σκόπιμο να γίνει μετάφραση της παραπάνω διαδικασίας σε αυτή την ανώτερου επιπέδου γλώσσα. Πραγματικά, πραγματοποιήθηκε μετάφραση από assembly σε C (κατάλογος 03_C_IMAGE_SHOW) η οποία απέδειξε ότι οι δύο αυτοί κόσμοι δεν είναι απολύτως ίδιοι. Η αιτία που δημιούργησε τα περισσότερα προβλήματα είναι αρχικοποιήσεις οι οποίες γίνονται πριν την εκτέλεση της ρουτίνας main και σκοπό έχουν να φέρουν τον επεξεργαστή σε μία προκαθορισμένη κατάσταση, ώστε κάποιος που προγραμματίζει να μπορεί να ξεκινήσει αμέσως να γράφει κώδικα. Συγκεκριμένα γίνονται αυτόματες αρχικοποιήσεις για Interrupt Handlers, PLL και SDRAM setup ειδικά για το EZ-KIT Lite™.



Εικόνα 87. Ακύρωση της λειτουργίας αρχικοποίησης της SDRAM

Προβλήματα εμφανίστηκαν κατά την αρχικοποίηση της SDRAM μίας και αυτή αρχικοποιείτω μόνο μερικώς και επιπλέον λόγω ενός bug έκδοσης του VisualDSP++ όταν πήγαινες να εκτελέσεις δεύτερη φορά πρόγραμμα στον επεξεργαστή, τα πάντα κολλούσαν λόγω διπλής αρχικοποίησης της SDRAM. Η λύση σε όλα τα παραπάνω προβλήματα δόθηκε με την απενεργοποίηση της αρχικοποίησης της SDRAM όπως φαίνεται στην Εικόνα 87. Στην συνέχεια ακολούθησε αρχικοποίηση της SDRAM με τον σωστό τρόπο μέσω του προγράμματός μας.

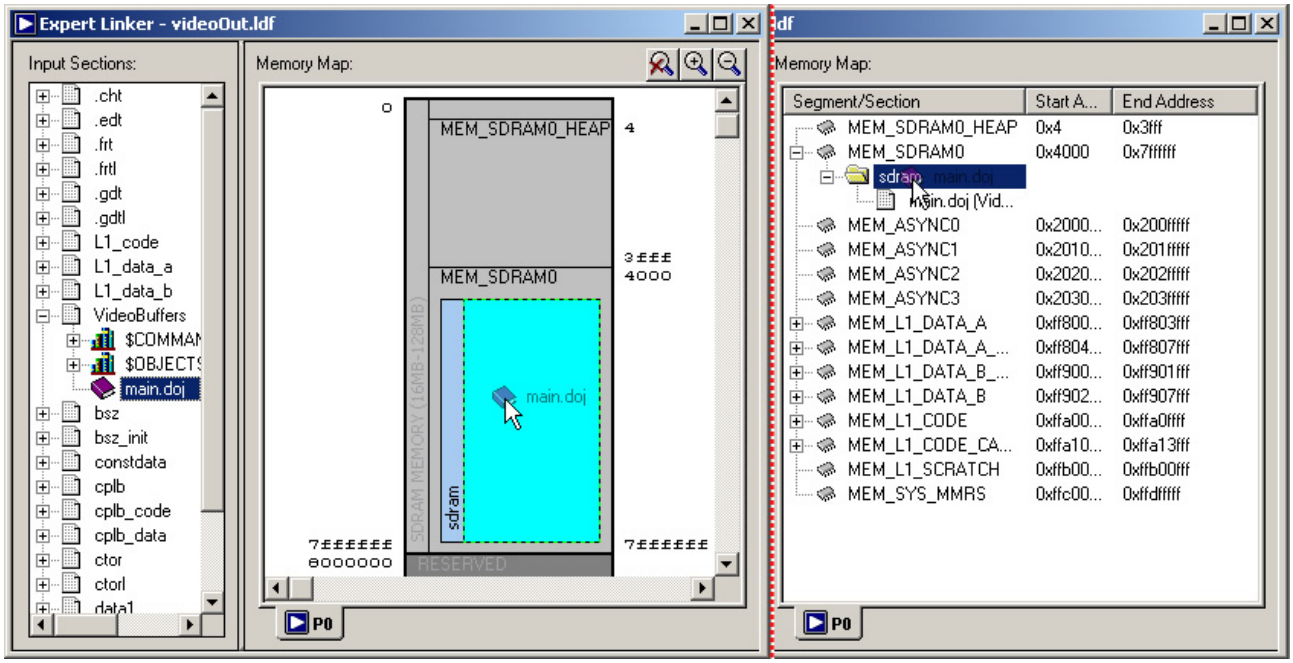
Επιπλέον κατά την αναβάθμιση του προγράμματος VisualDSP++ από την μία έκδοση στην επόμενη εμφανίστηκαν προβλήματα που σχετίζονταν με την αρχικοποίηση του διαιρέτη του PLL. Αυτό μας οδήγησε στο να κάνουμε εξ'αρχής επανέναρξη της ρύθμισης του PLL εξασφαλίζοντας έτσι σωστές ρυθμίσεις.

Για την δέσμευση χώρου στην μνήμη για τους δύο video buffers, ενώ στην assembly παίρναμε αυθαίρετα δύο pointers και γράφαμε στη μνήμη SDRAM, στην C πρέπει να γίνει ειδική διαδικασία που να καθορίζει τη μνήμη που θα δεσμευτεί για τις μεταβλητές αυτές. Αν προσπαθήσει κανείς να δεσμεύσει με απλό τρόπο τόσο μεγάλη μνήμη θα πάρει μήνυμα λάθους, αφού οι περιοχές μνήμης που είναι προεπιλεγμένες για τη δέσμευση μεταβλητών βρίσκονται στην εσωτερική μνήμη και

συνεπώς είναι πάρα πολύ μικρές. Για την δέσμευση των μεγάλων buffers, δηλώνεται ότι θα ανήκουν στον τομέα εισόδου (input sections), “VideoBuffers”.

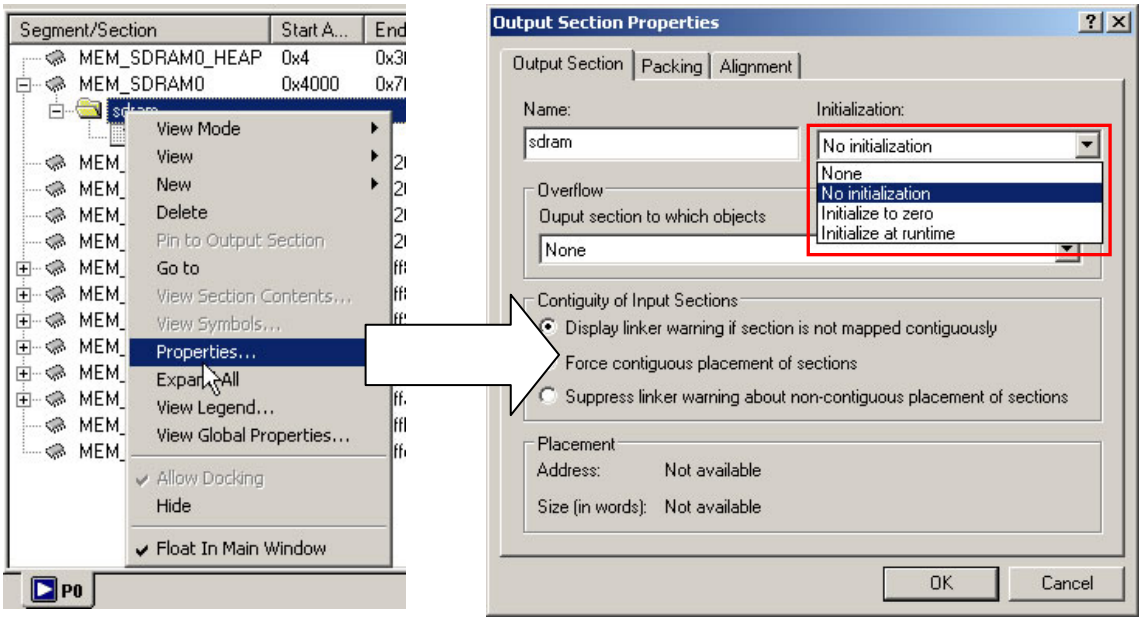
```
section("VideoBuffers") unsigned int videoBuffer[((1728 * 625) / 4)];
section("VideoBuffers") unsigned int videoBuffer2[((1728 * 625) / 4)];
```

Στη συνέχεια πρέπει να γίνει τροποποίηση του αρχείου .ldf που επεξεργάζεται ο linker για να ανατεθούν αυτοί οι τομείς στους αντίστοιχους εξωτερικούς τομείς της μνήμης.



Εικόνα 88. Ανάθεση input σε output sections με τον expert linker σε visual και tree view

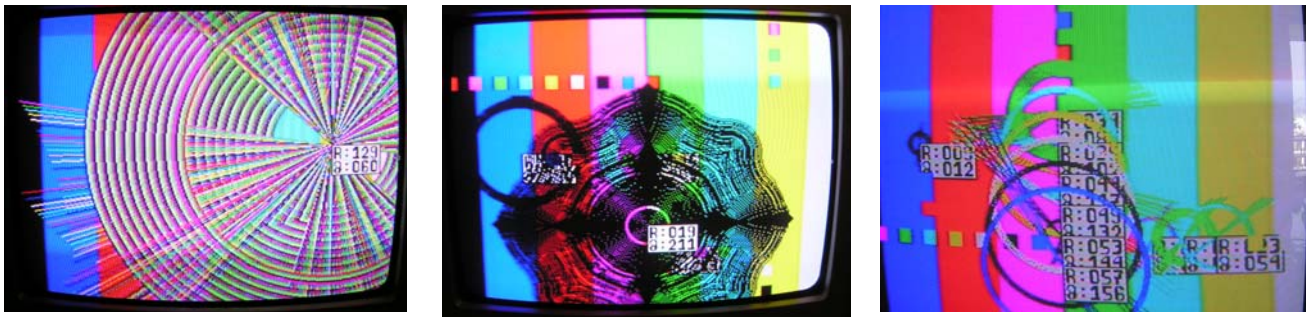
Αυτό γίνεται με τη βοήθεια του expert linker που καλείται αν κάνουμε διπλό κλικ στο αρχείο .ldf στο δέντρο του project, κάτω από την ενότητα Linker Files. Με τη βοήθειά του και με απλό drag n drop, μπορούμε να αντιστοιχίσουμε input sections σε output sections όπως φαίνεται στην Εικόνα 88. Κάνοντας αυτά, ολοκληρώνεται επιτυχώς η σύνθεση του προγράμματος αλλά το φόρτωμα στη μνήμη του επεξεργαστή γίνεται εξαιρετικά αργό, γιατί αρχικοποιείται όλη αυτή η μνήμη.



Εικόνα 89. Απενεργοποίηση αρχικοποίησης μνήμης

Με την κατάλληλη επιλογή (No initialization) όπως φαίνεται στην Εικόνα 89, απενεργοποιείται η αρχικοποίηση μνήμης. Η επιλογή None που είναι προεπιλεγμένη δεν μας κάνει. Από την επιλογή αυτή αρχίζει να εμφανίζεται μία προειδοποίηση (warning) από τον linker, αλλά αυτό δεν μας πειράζει.

Η έκδοση σε C για την ζωγραφική των μπάρων, χωρίς optimization, εκτελείται σε (cycles difference = 0x06C8264D) 113.780.301 κύκλους = 190ms δηλαδή εικοσαπλάσιο χρόνο από αυτόν της ρουτίνας σε assembly, μία τιμή η οποία δεν είναι κακή αν σκεφτεί κανείς ότι δεν χρησιμοποιούνται hardware loops.



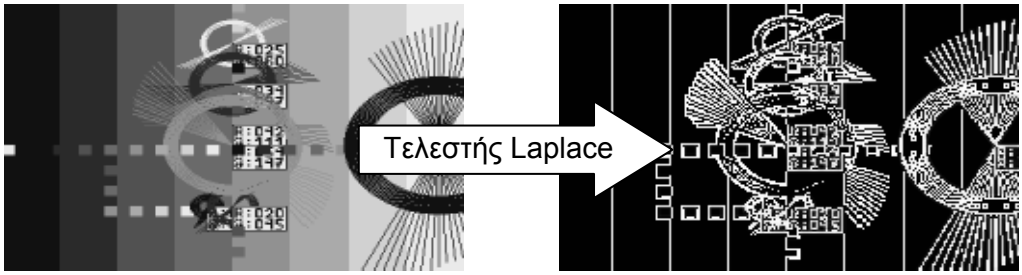
Εικόνα 90. Γραφικά στην οθόνη τηλεόρασης από τον BlackFin™

Όταν πλέον καταφέραμε να απεικονίσουμε σωστά τον Buffer στην οθόνη, επιδοθήκαμε στην συγγραφή ρουτινών σχεδιασμού γραφικών για τα βασικά στοιχεία που χρειαζόμαστε. Οι ρουτίνες αυτές περιλάμβαναν ένα γρήγορο αλγόριθμο γεμίσματος παραλληλογράμμων (fillRectangle), έναν αλγόριθμο αντιγραφής δυαδικών bitmaps 11 x 8bits (paint8x11tile) που χρησιμοποιήθηκε για την ζωγραφική των χαρακτήρων και δύο γρήγορες ρουτίνες ζωγραφικής οριζοντίων (drawHorizontalLine) και καθέτων (drawVerticalLine) γραμμών. Τα γράμματα ορίζονται μέσα στο header file, basic_painting.h. Με τη βοήθεια των παραπάνω συντέθηκε μία συνάρτηση paintFrame που ζωγράφιζε ένα παραλληλόγραμμο πλαίσιο, με μαύρο περίγραμμα όπου γράφονταν μέσα του οι ενδείξεις της γωνίας και της ακτίνας ενός κύκλου, συμβολιζόμενες με τους χαρακτήρες R και θ, όπως φαίνεται και στην Εικόνα 90. Πρέπει να επισημάνουμε τη σημαντικότητα της συνάρτησης div() της βιβλιοθήκης stdlib.h για τη δημιουργία της συνάρτησης paintFrame. Με τη βοήθειά της, με μία εκτέλεση της διαίρεσης παίρουμε και το πηλίκο, και το υπόλοιπο σε μία δομή div_t. Αν δεν υπήρχε αυτή η συνάρτηση θα έπρεπε να εκτελούσαμε μία εντολή διαίρεσης (/) και μία εντολή υπολοίπου (%) με διπλάσιο επεξεργαστικό κόστος.

Στη συνέχεια υλοποιήσαμε τον αλγόριθμο του Bresenham για την ζωγραφική κύκλων (drawCircle) και για την ζωγραφική ευθειών (drawline2d). Και οι δύο εκδοχές είναι καθαρά ακέραιες υλοποιήσεις με συνέπεια να έχουν μεγάλη ταχύτητα εκτέλεσης. Για να ζωγραφίσουμε την ευθεία που περνά από ένα κέντρο και έχει γωνία θ με το επίπεδο πρέπει να είμαστε σε θέση να υπολογίσουμε τα ημίτονα και συνημίτονα της γωνίας θ. Αυτό γίνεται με τη βοήθεια lookup table αν και ελαφρώς πιο αργή αλλά με λιγότερες απαιτήσεις σε μνήμη προγράμματος θα ήταν η τεχνική ανάπτυξης σε σειρά. Θα πρέπει να σημειωθεί ότι χρειάστηκαν διάφορα τεχνάσματα τα οποία γενικά χρησιμοποιούνται στις ακέραιες υλοποιήσεις ώστε να καταφέρουμε να αναπαραστήσουμε τα ημίτονα και συνημίτονα με ακεραίους. Η συνάρτηση που κάνει την ζωγραφική της γραμμής είναι η συνάρτηση drawMountLine ενώ η συνάρτηση paintInfo συνδυάζει όλες τις παραπάνω για να απεικονίσει έναν κύκλο με κέντρο x_c, y_c και ακτίνα radius, μία ευθεία που περνάει από το κέντρο του κύκλου και έχει γωνία theta με τον οριζόντιο άξονα και ένα ταμπλό στο κέντρο του κύκλου με τις πληροφορίες της ακτίνας και του theta όπως φαίνεται στην Εικόνα 90.

Θα πρέπει να σημειωθεί ότι θέλει πολύ προσοχή στους αλγορίθμους αυτούς στην κατάλληλη επιλογή του τύπου των μεταβλητών (char, short, int κ.τ.λ.). Στις δύο πρώτες εικόνες που φαίνονται στην Εικόνα 90, μπορεί κανείς να δει bugs που προκύπτουν αν χρησιμοποιήσει κανείς char αντί για short. Στην πρώτη από μία ακτίνα και για μεγαλύτερες, άρχιζαν να μικραίνουν οι κύκλοι που ζωγραφίζονταν ενώ στην δεύτερη, από μία ακτίνα και παραπάνω, οι κύκλοι γίνονταν κυματιστοί από μία αναδίπλωση του d στον αλγόριθμο του Bresenham. Θα πρέπει να σημειωθεί επίσης ότι οι

κύκλοι δεν είναι ακριβώς κυκλικοί λόγω του ότι τα pixels δεν είναι τετράγωνα, αλλά αυτό γίνεται εμφανές μόνο για μεγάλες ακτίνες και συνεπώς δεν μας απασχολεί περαιτέρω.



Εικόνα 91. Εφαρμογή του τελεστή Laplace σε σμίκρυνση της εικόνας εξόδου

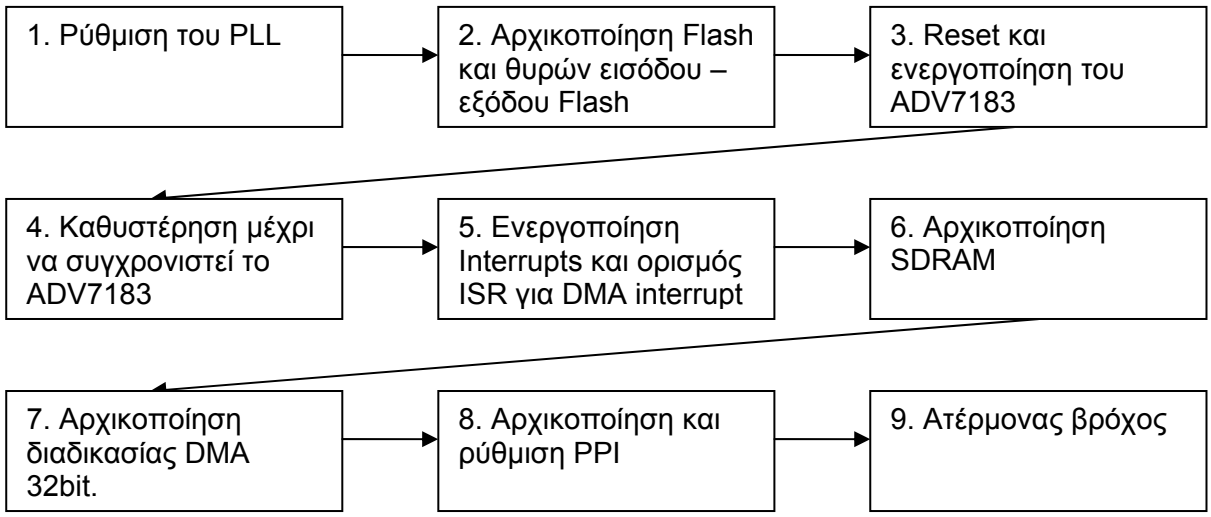
Τέλος προτυποποιήσαμε έναν απλό αλγόριθμο σμίκρυνσης εικόνας με κατακόρυφο λόγο 1:4 και οριζόντιο λόγο 1:3 ο οποίος εξάγει εικόνες 144 x 240 από τα πλαίσια video, όπως μπορούμε να δούμε στην Εικόνα 91. Στη συνέχεια εφαρμόσαμε ένα 3x3 τελεστή Laplace της μορφής:

-1	-1	-1
-1	8	-1
-1	-1	-1

και επαληθεύσαμε ότι όντως μπορούμε να διακρίνουμε τις ακμές της εικόνας. Αυτά αποτελούν μία καλή υποδομή για τη συνέχεια της εργασίας μας.

6.5 Παράδειγμα λήψης σήματος video

Υπάρχει ένα παράδειγμα σε C που έρχεται μαζί με τον BlackFin™ που κάνει δυνατή την λήψη ενός πλαισίου video. Αυτό είναι πιο απλό και η παρακολούθησή του θα είναι εύκολη για όσους εξέτασαν τα όσα παρουσιάστηκαν παραπάνω. Πρέπει να σημειωθεί ότι την απλότητά του αυτό το παράδειγμα την οφείλει σε μεγάλο βαθμό στο ότι το πρόγραμμα είναι γραμμένο σε C και έτσι πολύς κώδικας δημιουργείται αυτόματα από τον compiler.



1. Το πρώτο βήμα του κώδικα αρχικοποίησης είναι να ρυθμίσει την λειτουργία του PLL. Και αυτό το πρόγραμμα δίνει στον καταχωρητή PLL_CTL την τιμή 0x2C00 που σημαίνει πολλαπλασιασμός επί 22, δηλαδή λειτουργία πυρήνα στα 27 * 22 = 594MHz. Και πάλι περιμένουμε για PLL_LOCKCNT (= 0x0200) κύκλους συστήματος έως ότου να κλειδώσει το PLL. Αυτό γίνεται με τη βοήθεια της εντολής IDLE που κρατάει τον επεξεργαστή αδρανή έως ότου σταθεροποιηθεί το PLL. Ο επεξεργαστής ξυπνάει αν έχουμε ενεργοποιήσει το bit 0 (PLL_WAKEUP) του καταχωρητή SIC_IWR. Σημαντικό είναι ότι δεν αρχικοποιείται ο καταχωρητής PLL_DIV αφού η αρχική του τιμή (/5) είναι η πιο κατάλληλη.

2. Αμέσως μετά ρυθμίζεται η επικοινωνία με την εξωτερική μνήμη Flash (Init_EBIU). Οι ρυθμίσεις είναι ακριβώς ίδιες με αυτές που αναφέραμε στο προηγούμενο παράδειγμα. Στη συνέχεια αρχικοποιούνται οι θύρες εισόδου – εξόδου της μνήμης Flash (Init_Flash) με ίδιο τρόπο όπως στην προηγούμενη φορά. Συγκεκριμένα αρχικοποιούνται ως έξοδοι με αρχική τιμή 0.

3. Η κίνηση αυτή (Init_ADV) παρουσιάζει ιδιαίτερο ενδιαφέρον. Συγκεκριμένα γίνεται επανατοποθέτηση (reset) του αισθητήρα εικόνας με μία μετάβαση από χαμηλή στάθμη σε υψηλή για την αντίστοιχη ακίδα του video decoder. Επίσης ενεργοποιείται το bit 4 του καταχωρητή αυτού που επιλέγει ως πηγή χρονισμού της θύρας PPI (PPI_CLK) τον video decoder αντί για της συνήθους ρύθμισης που χρησιμοποιείται και στην περίπτωση παραγωγής εικόνας video μέσω του video encoder που είναι το σήμα χρονισμού να προέρχεται κατευθείαν από το ρολόι του συστήματος (27 MHz). Στη συνέχεια για κάποιο περίεργο λόγο ενεργοποιεί το bit που αντιστοιχεί στην ακίδα OE (ενεργοποίηση εξόδου – output enable) του καταχωρητή ελέγχου εισόδου, FIO_INEN, πράγμα που ενεργοποιεί τον καταχωρητή εισόδου για αυτό το κανάλι. Κάτι τέτοιο δεν φαίνεται απαραίτητο δεδομένου ότι στην αμέσως επόμενη εντολή ενεργοποιεί την ίδια ακίδα ως έξοδο με τη βοήθεια του καταχωρητή FIO_DIR. Στη συνέχεια καθαρίζει το συγκεκριμένο bit γράφοντας μονάδα στον καταχωρητή FIO_FLAG_C. Η ακίδα OE ενεργοποιείται και επιτρέπει στο σήμα που παράγεται από τον video decoder να οδηγείται στην θύρα PPI του επεξεργαστή. Και αυτό είναι μία ουσιαστική διαφορά από την περίπτωση του video encoder. Τότε η ακίδα OE πρέπει να είναι απενεργοποιημένη ώστε να αφήνει το PPI να στέλνει δεδομένα προς τον video encoder.

4. Στη συνέχεια αφήνεται κάποιος χρόνος μέχρι το ADV7183 να «κλειδώσει» στο εισερχόμενο σήμα εικόνας μετά από το reset. Αυτός ο χρόνος δίνεται τυπώνοντας ένα μικρό κείμενο στην οθόνη του υπολογιστή μέσω του USB καλωδίου και της εντολής printf, διαδικασία που παίρνει αρκετό χρόνο.

5. Στο επόμενο βήμα ενεργοποιούνται τα interrupts με τον εκτενή τρόπο του ορισμού κάθε δρομολόγησης με τη βοήθεια των καταχωρητών SIC_IAR0-2. Αυτό φυσικά είναι περιττό αφού στο τέλος επιλέγει την default τιμή για το περιφερειακό που μας ενδιαφέρει. Μετά καταχωρείται η συνάρτηση εξυπηρέτησης διακοπής (DMA0_PPI_ISR) και ενεργοποιείται το Interrupt για το περιφερειακό PPI στον ελεγκτή περιφερειακών με τη βοήθεια του καταχωρητή SIC_IMASK. Πρέπει να σημειωθεί ότι ο αναγνώστης δεν θα βρει την αντίστοιχη ενεργοποίηση του καταχωρητή IMASK στον κώδικα. Αυτή γίνεται έμμεσα από τη συνάρτηση δήλωσης της ISR: register_handler.

6. Στη συνέχεια αρχικοποιείται η επικοινωνία με την εξωτερική μνήμη SDRAM με τη βοήθεια της βαθμίδας EBIU. Οι τιμές που δίνονται είναι οι ίδιες με την περίπτωση του προηγούμενου παραδείγματος (video encoder). Την SDRAM την χρειαζόμαστε γιατί εκεί γίνεται η αποθήκευση του πλαισίου video.

7. Στην συνέχεια αρχικοποιείται η λειτουργία του καναλιού DMA με την βοήθεια του οποίου γράφονται τα δεδομένα από την θύρα PPI στην μνήμη. Συγκεκριμένα τίθεται DMA0_START_ADDR = 0 δηλαδή το πλαίσιο θα αποθηκευτεί στις πρώτες διευθύνσεις της SDRAM, DMA0_X_COUNT = 50000 που καλύπτει ένα μέρος μόνο του πλαισίου (περίπου 200kb) και πρέπει να επεκταθεί σε 2D για να καλύπτει και το υπόλοιπο μέρος, DMA0_X_MODIFY = 4 που χρησιμοποιείται επειδή έχει επιλεχθεί τα δεδομένα να γράφονται σε «πακέτα» των 32bits και η ρύθμιση του DMA0_CONFIG ώστε το κανάλι DMA να είναι ενεργοποιημένο, να δημιουργεί interrupt στη ολοκλήρωση του πλαισίου, να είναι ένα κανάλι εγγραφής στη μνήμη, να γράφει σε «πακέτα» των 32bits και να καθαρίζει την εσωτερική FIFO του DMA πριν από κάθε έναρξη λειτουργίας. Επίσης τίθεται ως προεπιλεγμένο περιφερειακό για το DMA, το PPI, κάτι που επίσης δε χρειάζεται αφού είναι η default τιμή.

8. Στη συνέχεια ενεργοποιείται το PPI. Σε αυτό τίθεται εύρος πλαισίου (PPI_FRAME) ίσο με 525 γραμμές (NTSC περίπτωση) και PPI_CONTROL που ενεργοποιεί το περιφερειακό, δηλώνει ότι θέλει ανάγνωση και του μονού και του ζυγού πλαισίου (FLD_SEL), ενεργοποιεί το πακετάρισμα δεδομένων (δύο 8-bit δεδομένα σε μία 16-bit μεταβλητή) και την μεταφορά τους από την λειτουργία DMA σε 32-bites ενότητες. Μετά από αυτή την εντολή αρχίζει η μεταφορά του σήματος video.

9. Στη συνέχεια υπάρχει ένας κλασικός ατέρμονος βρόχος.

Θα έπρεπε και πάλι να σημειώσουμε ότι ο εξυπηρετητής διακοπής που καλείται όταν ολοκληρώνεται η ανάγνωση ενός πλήρους πλαισίου video, καθαρίζει τη σχετική σημαία DMA0_IRQ_STATUS = 1 και τυπώνει ένα μήνυμα στην οθόνη του υπολογιστή με τη βοήθεια της εντολής printf.

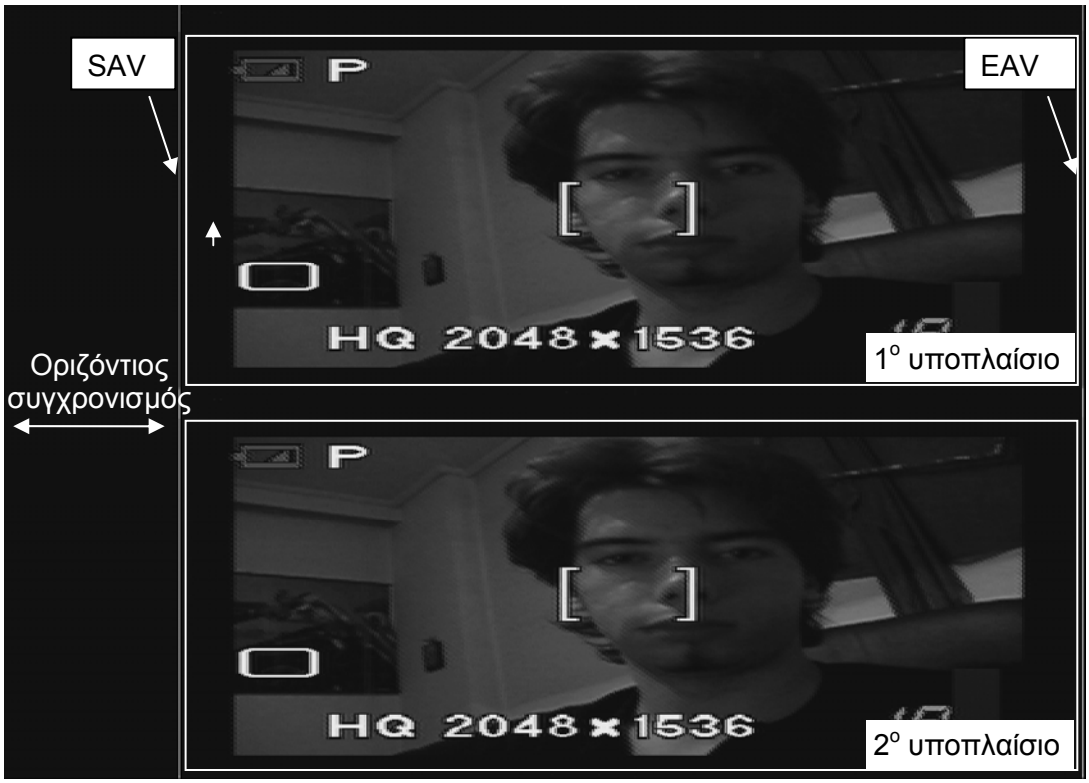
Θα πρέπει επίσης να σημειώσουμε ότι το κανάλι DMA δεν τίθεται σε κατάσταση autobuffer και έτσι, μετά την ολοκλήρωση της λήψης ενός πλαισίου video, αυτό θα σταματήσει την λειτουργία του.

6.6 Τροποποίηση του παραδείγματος λήψης video

Φυσικά και πάλι προβήκαμε στην τροποποίηση του παραδείγματος λήψης video (κατάλογος 04_C_IMAGE_CAPTURE). Συγκεκριμένα ενοποιήσαμε όλες τις μικρές συναρτήσεις αρχικοποίησης που ήταν διασκορπισμένες σε διάφορα αρχεία σε μία μόνο συνάρτηση που έκανε όλη την αρχικοποίηση. Στην συνέχεια περικόψαμε τα περιττά πράγματα και επαληθεύσαμε τη σωστή λειτουργία του.

Μετά κάναμε τη διαδικασία λήψης πλαισίου video να επανεκινείται με το πάτημα ενός πλήκτρου. Αυτό ήθελε προσοχή. Ενώ όλη η διαδικασία θα μπορούσε να γίνει απλά ενεργοποιώντας ξανά το κανάλι DMA (pDMA0_CONFIG |= DMA_EN) αυτό δεν έδινε καλά αποτελέσματα. Πιο συγκεκριμένα παρόλο που το κανάλι DMA έχει σταματήσει να γράφει στη μνήμη, η θύρα PPI συνεχίζει να λαμβάνει σήμα από τον video decoder. Αυτό έχει ως συνέπεια όταν ξαναενεργοποιήσουμε το DMA κανάλι να συνεχίζει η εγγραφή από το τυχαίο σημείο που διαβάζει εκείνη την στιγμή το PPI δίνοντας ένα ξεσυγχρονισμένο πλαίσιο.

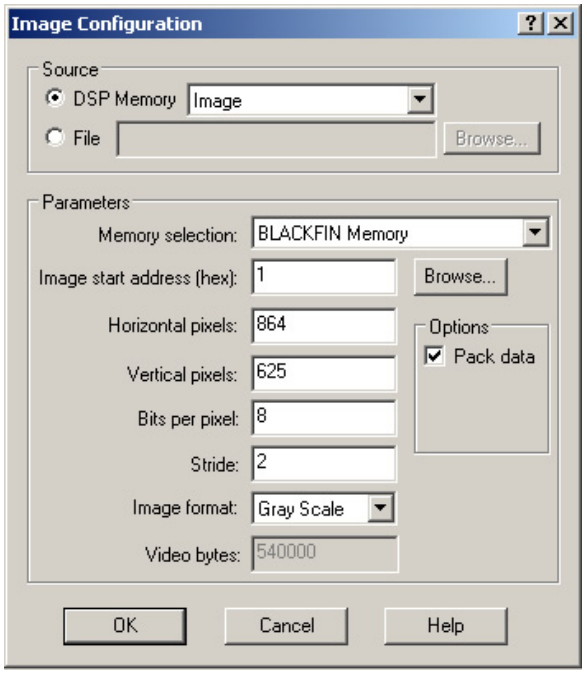
Για να ξεπεράσουμε το πρόβλημα αυτό αρκεί να επανεκινούμε και την λειτουργία του PPI όταν θέλουμε να λάβουμε καινούριο πλαίσιο (PPI_CONTROL = 0, ssync(), PPI_CONTROL = PPI_INIT_PARAMETERS). Το πολύ σημαντικό που πετυχαίνουμε με αυτή την τεχνική είναι η λήψη ενός πλαισίου video χωρίς να χρειαστεί επανεκκίνηση (reset) του ADV7183 (βήμα 3, 4 που περιγράφηκε παραπάνω) η οποία θέλει αρκετό χρόνο. Αν δεν γινόταν αυτό δεν θα μπορούσαμε να έχουμε λήψη video χωρίς αρκετά μεγάλη διακοπή που θα έδινε άσχημο αισθητικό αποτέλεσμα.



Εικόνα 92. Πλαίσιο video όπως λαμβάνεται από την κάμερα

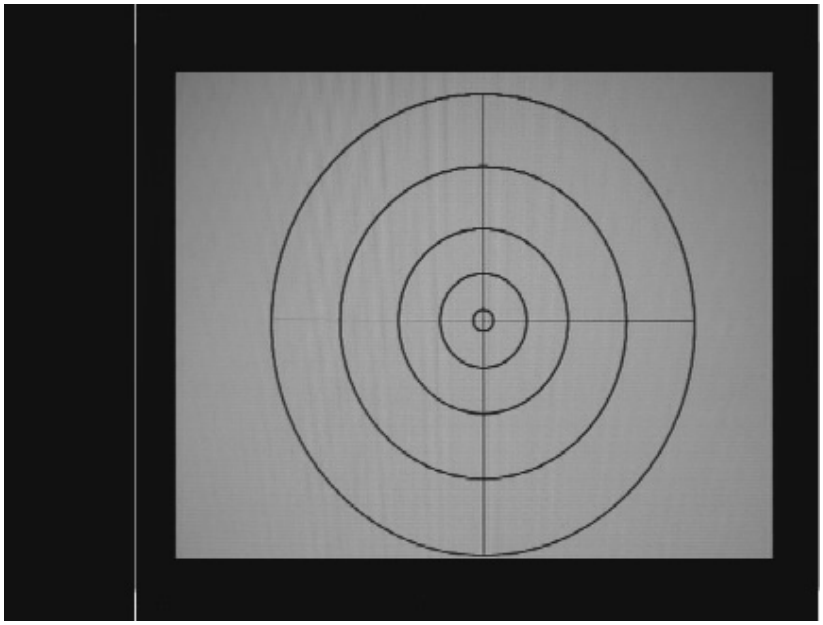
Στην Εικόνα 92 φαίνεται ένα πλήρες πλαίσιο video όπως λαμβάνεται από την κάμερα. Αυτό αποθηκεύεται στην εξωτερική SDRAM και έχει μέγεθος 625 lines x 1728 bytes = 1080000bytes. Γίνεται φανερό ότι ένα πλήρες πλαίσιο video περιέχει πολλή περισσότερη πληροφορία πέρα από την εικόνα. Καταρχάς παρατηρούμε ότι περιλαμβάνει 2 υποπλαίσια τα οποία όπως είπαμε συντίθενται (interlace) και δημιουργούν ένα μοναδικό πλαίσιο εικόνας το οποίο βλέπουμε στην οθόνη. Όλα όσα βρίσκονται αριστερά της πρώτης κατακόρυφης γραμμής είναι τα σήματα οριζόντιου συγχρονισμού. Η κατακόρυφη γραμμή είναι τα σήματα SAV τα οποία όπως βλέπουμε έχουν διαφορετική τιμή (φωτεινότητα) για το πρώτο και το δεύτερο υποπλαίσιο. Ότι βρίσκεται ανάμεσα στην πρώτη κατακόρυφη γραμμή και τη δεύτερη κατακόρυφη γραμμή που φαίνεται πολύ δεξιά είναι το ωφέλιμο σήμα video των δύο υποπλαισίων. Η δεύτερη κατακόρυφη γραμμή (στα δεξιά) είναι το σήμα EAV το οποίο και αυτό παρατηρούμε ότι έχει διαφορετική τιμή για το πρώτο και δεύτερο υποπλαίσιο. Επίσης, όσες γραμμές δεν ανήκουν σε κάποιο από τα δύο υποπλαίσια είναι γραμμές του σήματος κατακόρυφου συγχρονισμού.

Επιπλέον παρατηρούμε ότι δεν γίνεται πλήρης εκμετάλλευση ούτε του «χώρου» των δύο υποπλαισίων. Ένα 11% κατά τον κατακόρυφο άξονα και ένα 12% κατά τον οριζόντιο σε κάθε υποπλαίσιο κρατούνται μαύρα χωρίς να είναι σήματα συγχρονισμού. Την σύμβαση αυτή την «σέβονται» και οι τηλεοράσεις που δείχνουν μόνο αυτό το 90% κάθε υποπλαισίου σε όλο το εύρος της οθόνης τους.



Εικόνα 93. Ρυθμίσεις για την εμφάνιση της παραπάνω εικόνας

Οι ρυθμίσεις που πρέπει να τεθούν στον Image Viewer (View > Debug Windows > Image Viewer…) φαίνονται στην Εικόνα 93. Αν θελήσει κανείς να αποφύγει την παραμόρφωση 2:1 κατά τον κατακόρυφο άξονα που εισάγεται λόγω της ύπαρξης δύο υποπλαισίων μπορεί να τροποποιήσει τις τιμές Horizontal pixels: 432, Stride 4. Τότε βέβαια η εικόνα και πάλι θα είναι παραμορφωμένη λόγω των μη τετραγωνικών pixels. Επειδή το σήμα απευθύνεται σε οθόνες με λόγο μήκους – πλάτους 4:3, το σήμα παρουσιάζει την εικόνα συμπτυγμένη κατά τον οριζόντιο άξονα. Αυτές οι παραμορφώσεις μπορούν να αλλοιώσουν σημαντικά τη λειτουργία του κυκλικού μετασχηματισμού Hough αφού μετατρέπουν τους κύκλους σε ελλείψεις.



Εικόνα 94. Παραμόρφωση κυκλικού προτύπου

Για να μελετήσουμε περισσότερο τις παραμορφώσεις αυτές χρησιμοποιήσαμε ένα κυκλικό πρότυπο και μπορούμε να παρατηρήσουμε στην Εικόνα 94 την ισχυρή παραμόρφωση που υφίσταται (έχει υποστεί την 2:1 ανόρθωση όπως περιγράψαμε παραπάνω).

Πλάτος (pixels)	Ύψος (pixels)	Ύψος / Πλάτος
225	245	1.089
153	167	1.092
92	99	1.076
47	51	1.085
12	13	1.083

Όπως συμπεραίνουμε από το πείραμα υπάρχει μία σμίκρυνση του οριζοντίου άξονα κατά 8,5% (μέσος όρος). Οι διακυμάνσεις είναι αναμενόμενες λόγω του χωρικού θορύβου κβαντισμού κατά τη διακριτοποίηση από την κάμερα. Αυτή η παραμόρφωση θα πρέπει να ανορθωθεί προγραμματιστικά, προκειμένου να έχουμε σωστή λειτουργία του μετασχηματισμού Hough.

Τέλος από την παραπάνω εικόνα μπορούμε να μετρήσουμε τη θέση της ωφέλιμης εικόνας μέσα στο υποπλαίσιο. Συγκεκριμένα μετράμε ότι η εικόνα έχει μέγεθος 632 (316x2) x 257 σημείων και βρίσκεται 42 σημεία δεξιά του σήματος SAV και στην 36^{η} γραμμή από την αρχή του πλαισίου.

6.7 Συγχώνευση λειτουργιών λήψης και εκπομπής video

Συνοψίζοντας τα παραπάνω μπορούμε να ανακεφαλαιώσουμε τις λειτουργίες που χρειάζονται για την εναλλαγή από εκπομπή σε λήψη video. Από εδώ και πέρα όλα τα προγράμματα που περιγράφονται βρίσκονται στον κατάλογο 05_C_FINAL_VERSION σε διαφορετικά .c αρχεία. Για να ενσωματώσουμε τη λειτουργία λήψης video στο τροποποιημένο πρόγραμμα παραγωγής video που παρουσιάσαμε παραπάνω (παράγραφος 6.4) χρειάζεται:

A. Κατά την αρχικοποίηση να γίνεται Reset και στο ADV7183 (FlashA_PortA_Out |= RST_7183)
B. Πριν την έναρξη λήψης σήματος video να γίνεται
   1. επιλογή ADV7183 ως η πηγή σήματος χρονισμού για το PPI (FlashA_PortA_Out |= PPICLK_ADV7183_SELECT)
   2. ενεργοποίηση των εξόδων του ADV7183 (FIO_FLAG_C |= ADV7183_OE)
   3. ανάθεση του σωστού ISR για την εξυπηρέτηση διακοπής του ADV7183 (register_handler(ik_ivg8, DMA0_PPI_DECODER_ISR))

4. αρχικοποίηση των παραμέτρων DMA (DMA0_START_ADDR = target_buffer, DMA0_X_COUNT = 432, DMA0_X_MODIFY = 4, DMA0_Y_COUNT = 625, DMA0_Y_MODIFY = 4, DMA0_CONFIG = DMA_INIT_PARAMETERS)
5. αρχικοποίηση του PPI (PPI_FRAME = 625, PPI_CONTROL = PPI_INIT_PARAMETERS)

Γ. Μετά τη σύλληψη εικόνας πρέπει να γίνεται επανααρχικοποίηση της εκπομπής σήματος video μέσω του video encoder.

Θα έπρεπε να σχολιάσουμε το βήμα 3 λίγο περισσότερο. Επειδή η ρουτίνα ISR για το interrupt του PPI πρέπει να είναι όσο το δυνατόν πιο γρήγορη προκειμένου να μη διακοπεί η ροή των δεδομένων video, είναι καλό να μην εισάγονται σε αυτήν εντολές if και άλλες χρονοβόρες δομές. Για το λόγο αυτό μπορούμε να χρησιμοποιήσουμε δύο διαφορετικά ISR για κάθε τύπο εξυπηρέτησης (του video encoder και του video decoder) και να τα εναλλάσσουμε κατά την έναρξη της επικοινωνίας.

Αυτά όμως είναι μόνο τα προφανή βήματα. Μετά από αρκετές ώρες ανεπιτυχούς προσπάθειας λόγω ασυγχρόνιστου video, καταλήξαμε στην εξής τεχνική. Χρησιμοποιήσαμε ένα υπολογιστικά κατασκευασμένο πλήρες interlaced πλαίσιο video αποθηκευμένο σε μία μεταβλητή buffer1 με μπάρες που το είχαμε από προηγούμενη ενότητα. Ρυθμίσαμε τον ελεγκτή PPI να λαμβάνει μόνο τα ενεργά πλαίσια video δηλαδή (288*2)x720 σημεία. Στην συνέχεια αντιγράφουμε τα δύο ενεργά πλαίσια που μόλις συλλάβαμε στις σωστές θέσεις μέσα στον buffer1 με μία ρουτίνα (copyToBuffer). Τότε επιτέλους καταφέραμε να δούμε σήμα video στην οθόνη, με ένα μικρό σφάλμα... τα χρώματα ήταν μετατοπισμένα. Τότε έγινε προφανής ο λόγος της όλης δυσλειτουργίας του συστήματος. Κατά την 32-bit λήψη και μεταφορά δεδομένων στην σύλληψη της εικόνας, τα δεδομένα αναστρεφόντουσαν κατά λέξη (word reversed) σε σχέση με αυτά που περίμενε να «δει» ο video encoder. Συνεπώς τα σήματα συγχρονισμού που λαμβάνονταν από την κάμερα, μετά το word reverse δεν σήμαιναν τίποτα για τον video decoder, και τα πλαίσια δεν εμφανίζονταν στην οθόνη. Patch-άραμε στα γρήγορα την τεχνική μας τροποποιώντας την συνάρτηση copyToBuffer ως εξής,

```
unsigned int tmp;

tmp = *(src++);
*(dst++) = (tmp << 16) + (tmp >> 16);
```

και όλα δούλεψαν άψογα.

Θα έπρεπε να σημειώσουμε ότι δεν επιστρέψαμε στη τεχνική της σύλλήψης ολόκληρου του πλαίσιο video μαζί με τον συγχρονισμό αφότου βρήκαμε το σφάλμα αυτό. Το λαμβανόμενο πλαίσιο video είχε μία μικρή απόκλιση από το πλαίσιο video με τις μπάρες που ζωγραφίζαμε μέχρι πριν. Συγκεκριμένα το EAV ήταν στο τέλος του πλαισίου, σε αντίθεση με το τεχνητά ζωγραφισμένο πλαίσιο όπου το EAV ήταν στην αρχή του πλαισίου. Αυτό θα σήμαινε ότι θα έπρεπε να τροποποιηθούν οι ρουτίνες ζωγραφικής με την εισαγωγή μίας μικρής απόκλισης και η ρουτίνα scaling που είχε φτιαχτεί προηγουμένως. Αφού φτάσαμε στην κατάσταση να ζωγραφίζουμε πάνω στον κλασικό καμβά (buffer) με τις μπάρες, δεν κρίνεται σκόπιμο να αφιερωθεί περαιτέρω χρόνος για τροποποιήσεις ρουτινών που θα δίνουν στο τέλος το ίδιο αποτέλεσμα. Συνεπώς αφήσαμε τη τεχνική όπως έχει.

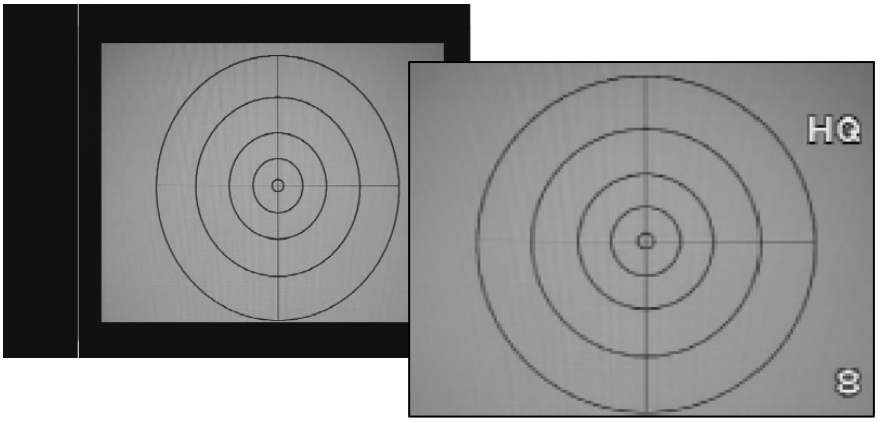
Συνεπώς αυτή την στιγμή, με την συνεργασία των αρχείων ppi_managment_unit.c, main.c, basic_painting.c και της συνάρτησης copyToBuffer (process.c) έχουμε ένα ολοκληρωμένο πλαίσιο με το οποίο μπορούμε να λαμβάνουμε εικόνες από σήμα video με το πάτημα ενός κουμπιού, να εκτελούμε βασικές λειτουργίες ζωγραφικής και να αποστέλλουμε στην οθόνη το σήμα της επεξεργασμένης εικόνας. Έχουμε αποκτήσει δηλαδή την υποδομή που θα μας προσέφερε ένα λειτουργικό σύστημα.

6.8 Λειτουργίες ανόρθωσης αναλογιών και ανίχνευσης ακμών

Μπορούμε πλέον να προχωρήσουμε στην δημιουργία υποδομής για αναγνώριση εικόνας ανάλογη με αυτή που μας προσέφερε το Matlab. Από εδώ και πέρα θα μιλάμε εξολοκλήρου για συναρτήσεις του αρχείου process.c.

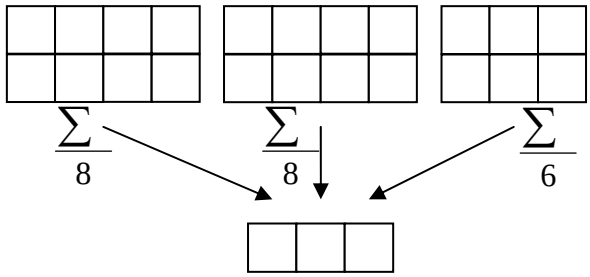
6.8.1 Ανόρθωση αναλογιών

Πρώτα από όλα έχουμε να αντιμετωπίσουμε ένα σημαντικό πρόβλημα, το ότι η εικόνα είναι παραμορφωμένη λόγω του ότι τα pixels δεν είναι τετράγωνα. Μετρώντας προσεκτικά τις αναλογίες παραπάνω, βρήκαμε ότι χρειαζόμαστε μία μεγέθυνση της εικόνας περίπου κατά 8.5%. Αν αυτή η παραμόρφωση αυτή δεν ανορθωθεί, όπως φαίνεται ξεκάθαρα στην Εικόνα 95 (πίσω πλαίσιο), δεν θα καταφέρουμε να αναγνωρίσουμε με ακρίβεια τους κύκλους γιατί αυτοί θα εμφανίζονται ως ελλείψεις.



Εικόνα 95. Η δοκιμαστική εικόνα πριν και μετά την ανόρθωση αναλογιών

Για την εργασία αυτή δημιουργήθηκε η συνάρτηση extractMiniFrame. Ή αρχή λειτουργίας της φαίνεται στην Εικόνα 96. Ομάδες των 8, 8 και 6 σημείων από το ένα υποπλαίσιο (σμίκρυνση κατά 50% κατά το ύψος) της αρχικής εικόνας αθροίζονται και παράγουν τρία σημεία της νέας εικόνας. Με αυτό τον τρόπο κατά τον οριζόντιο άξονα έχουμε μία αναλογία 3/11 = 27% ενώ κατά τον κατακόρυφο ¼ = 25%. Συνεπώς η τελική εικόνα θα έχει λόγο μήκους προς πλάτος σε σχέση με την αρχική ίσο με 27% / 25% = 1.09 δηλαδή μία μεγέθυνση κατά 9% κατά τον οριζόντιο άξονα, ακριβώς όπως θέλαμε.



Εικόνα 96. Λειτουργία ανόρθωσης αναλογιών (scaling)

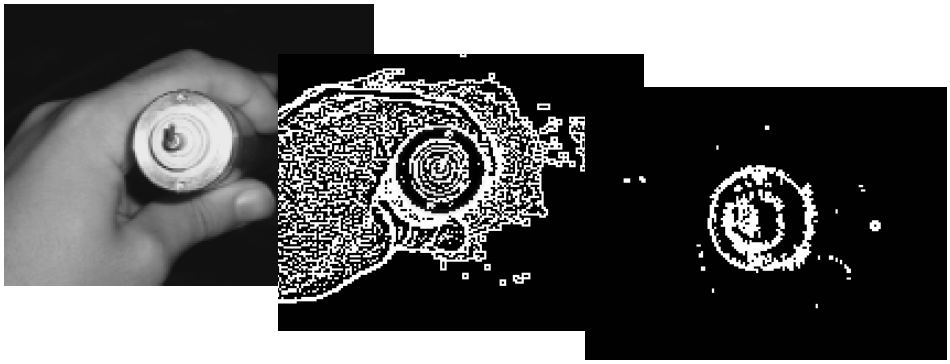
Η επιτυχία της μεθόδου αυτής φαίνεται στην Εικόνα 95 μπροστά πλαίσιο. Παρατηρούμε ότι η αναλογία μήκους πλάτους έχει πλέον ανορθωθεί και οι κύκλοι εμφανίζονται σωστά όπως θέλουμε και επαληθεύτηκε και με μετρήσεις.

Θα έπρεπε να επισημάνουμε ότι η σμίκρυνση συνολική σμίκρυνση της εικόνας κατά περίπου 25% δεν είναι μία αδυναμία της τεχνικής αλλά έγινε συνειδητά σε συμφωνία με τις προδιαγραφές της

ενότητας 4.3. Θα μπορούσαμε να επιτύχουμε με ανάλογα τεχνάσματα ανόρθωση των αναλογιών με οποιαδήποτε σμίκρυνση ή μεγέθυνση.

6.8.2 Ανίχνευση ακμών

Πάνω στην ανορθωμένη εικόνα που παράγεται με την προηγούμενη εικόνα θέλουμε να εκτελέσουμε ανίχνευση ακμών ώστε στη συνέχεια να μπορέσουμε να εφαρμόσουμε τον μετασχηματισμό Hough.



Εικόνα 97. Ανίχνευση ακμών με τη βοήθεια του τελεστή Laplace

Για τον σκοπό αυτό δημιουργήθηκε η συνάρτηση edgeFilter. Αυτή χρησιμοποιεί τον πυρήνα που παρουσιάσαμε στην ενότητα 6.4. Στην Εικόνα 97 φαίνεται αριστερά η αρχική εικόνα και στο κέντρο η εικόνα μετά την εφαρμογή του τελεστή Laplace.

Πρέπει να σημειωθεί ότι στην εικόνα αυτή έντονα άσπρα σημεία σημαίνουν μικρές εντάσεις εναλλαγής (μικρή δεύτερη παράγωγος) μιας και αντιστοιχούν σε αρνητικές τιμές κοντά το 0 (π.χ. 0xff = -1) οι οποίες φαίνονται ως έντονα άσπρες γιατί η αναπαράσταση της εικόνας τις λαμβάνει υπόψιν της ως απρόσημους ακεραίους. Συνεπώς οι σημαντικές ακμές βρίσκονται εκεί που βρίσκουμε πολλά γκρίζα σημεία δηλαδή έντονα αρνητικές τιμές.

Όπως μπορεί κανείς εύκολα να δει, από τη μεσαία εικόνα δεν βγαίνουν κατευθείαν συμπεράσματα για το που βρίσκονται οι ακμές. Γι' αυτό τον λόγο επιστρατεύουμε ένα επιπλέον κριτήριο. Παίρνουμε το κεντρικό σημείο και βρίσκουμε στα (όχι διαγώνια) γειτονικά του σημεία το σημείο με αντίθετο πρόσημο και τη μεγαλύτερη διαφορά σε σχέση με το κεντρικό. Αν η διαφορά αυτή είναι μεγαλύτερη από μία τιμή κατωφλιού την οποία ρυθμίζουμε με τα πλήκτρα του αναπτυξιακού, τότε ανιχνεύουμε επιτυχώς ένα σημείο ακμή. Αφήνουμε δηλαδή να περάσουν μόνο οι ακμές με μεγάλη ένταση – διακύμανση. Ο λόγος που ζητάμε το σημείο να έχει διαφορετικό πρόσημο από αυτό του κεντρικού είναι γιατί αυτό που ψάχνουμε μετά την εφαρμογή του τελεστή Laplace είναι τα περάσματα από το 0 (zero crossings) και αυτά συνοδεύονται από αλλαγή προσήμου.

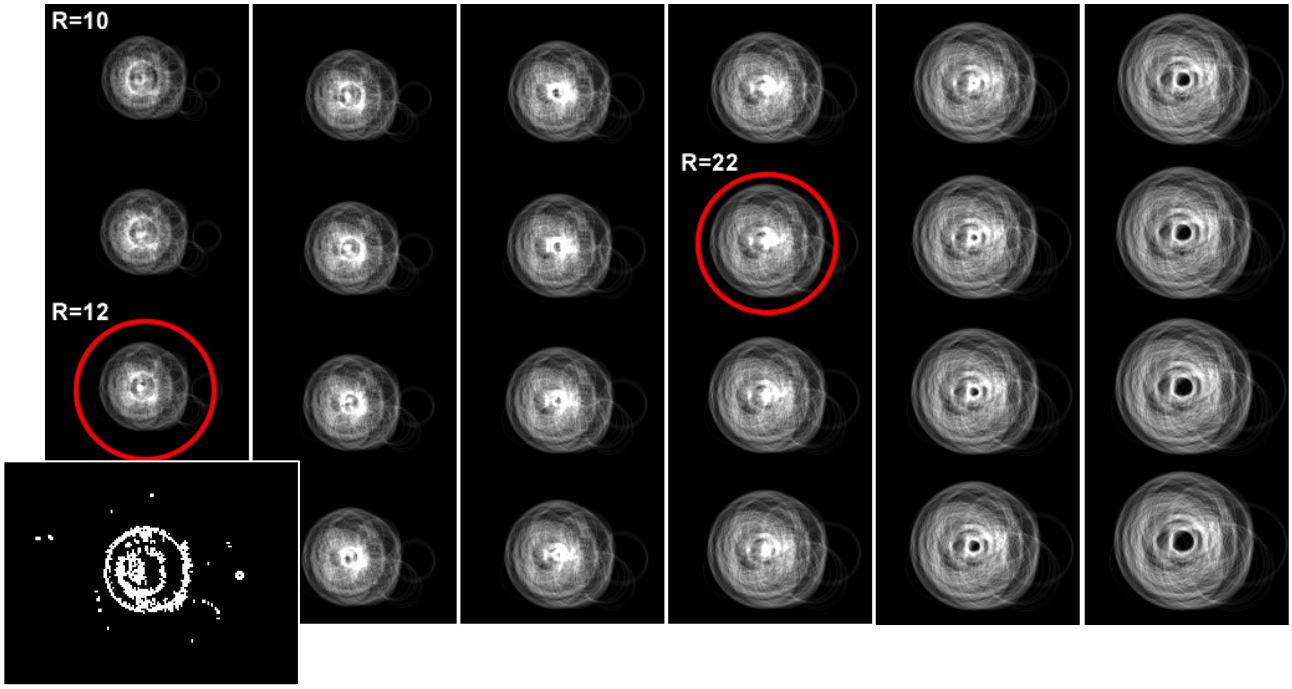
Την αποτελεσματικότητα της μεθόδου την βλέπετε στην Εικόνα 97 όπου στα δεξιά φαίνεται η εικόνα μετά την επιβολή κατωφλιού στην κεντρική εικόνα. Όπως μπορούμε να δούμε, ο θόρυβος εξασθένησε σημαντικά και έμειναν μόνο οι σημαντικές ακμές της εικόνας μας. Πρέπει να σημειώσουμε ότι τα αποτελέσματα μπορούν να βελτιωθούν σημαντικά με διάφορες τεχνικές "χαλάρωσης" και σύνδεσης ακμών. Μία εξ' αυτών δοκιμάστηκε και έχει παραμείνει ως κώδικας σε σχόλια μιας και δεν έδινε εντυπωσιακά καλύτερα αποτελέσματα. Το σημαντικό είναι όμως ότι υπάρχει αρκετό περιθώριο για βελτίωση της μεθόδου αναγνώρισης ακμών.

6.9 Υλοποίηση αναγνώρισης κύκλων

Η αναγνώριση κύκλων γίνεται σε δύο βήματα. Πρώτα γίνεται μετασχηματισμός Hough και στη συνέχεια γίνεται αναγνώριση του προτύπου στον χώρο Hough.

6.9.1 Μετασχηματισμός Hough

Στη συνέχεια έχουμε την υλοποίηση του κυκλικού μετασχηματισμού Hough. Θα εντυπωσιαστεί κανείς βλέποντας ότι ουσιαστικά είναι πέντε γραμμές όπως φαίνεται στη συνάρτηση houghTransform. Οι πρώτες 4 γραμμές αρχικοποιούν τον χώρο Hough (unsigned short houghSpace[DR][124][164]) στην τιμή 0 οπότε ουσιαστικά δεν αποτελούν μέρος του μετασχηματισμού. Ο κυκλικός αλγόριθμος Hough, αυτό που λέει είναι, γύρω από κάθε σημείο στο οποίο ανιχνεύτηκε ακμή, χάραξε στον χώρο Hough έναν κύκλο για κάθε ακτίνα. Για την χάραξη των κύκλων στον χώρο Hough χρησιμοποιήθηκε ο τροποποιημένος αλγόριθμος του Bresenham που προτυποποιήσαμε στην ενότητα 5.1 ενώ η «ένταση» του κάθε κύκλου γίνεται αντιστρόφως ανάλογη της περιφέρειάς του, πR (συνάρτηση initializeIncreaseStep), ώστε ένα ισχυρό μέγιστο σε μικρή ακτίνα, να έχει περίπου ίδια τιμή με ένα ισχυρό μέγιστο σε μία μεγαλύτερη. Με τον τρόπο αυτό μπορούμε να χρησιμοποιήσουμε ένα κοινό threshold για όλες τις ακτίνες.



Εικόνα 98. Μετασχηματισμός Hough από το DSP και τα ίχνη του προτύπου

Το αποτέλεσμα του μετασχηματισμού Hough για την εικόνα στην οποία πριν ανιχνεύσαμε ακμές, φαίνεται στην Εικόνα 98 και όπως βλέπουμε το αποτέλεσμα είναι εντυπωσιακό. Μπορούμε να δούμε στιγμιότυπα του χώρου Hough για ακτίνες από 10 έως 32 pixels. Αν τα φανταστεί κανείς το ένα πάνω στο άλλο αυτά συνθέτουν τον τρισδιάστατο χώρο Hough. Παρατηρούμε δύο πολύ έντονα μέγιστα για ακτίνες R=12 και R = 22 που αντιστοιχούν στον μικρό και στον μεγάλο δακτύλιο του προτύπου. Όπως περιμέναμε από τη γεωμετρία του προτύπου, η μία ακτίνα είναι σχεδόν η μισή της άλλης.

6.9.2 Μετασχηματισμός Hough

Όπως γίνεται φανερό από την παραπάνω εικόνα η γεωμετρία του προς αναγνώριση προτύπου, ξεχωρίζει εύκολα με βάση τα μέγιστα. Παρόλα αυτά στην πράξη, το κέντρο του μικρού κύκλου μπορεί να βρίσκεται σε μία αρκετά ευρεία περιοχή λόγω της πιθανής κλίσης του MDT. Δυστυχώς έπειτα από πειραματικές προσπάθειες δεν φάνηκε να είναι αποδοτική η χρήση ενός νεύρωνα ακόμα και με χωρικά φίλτρα με μεγάλο σ. Διαφάνηκε η ανάγκη να αναγνωρίσουμε του έντονους κύκλου, να τους αποθηκεύσουμε σε διανυσματική μορφή και στη συνέχεια αν θελήσουμε να τους επεξεργαστούμε περαιτέρω για την αξιολόγησή τους. Με αυτό το σκεπτικό θέσαμε προσεκτικά μία τιμή κατωφλιού, πάνω από την οποία ανιχνεύαμε την ύπαρξη ισχυρού κύκλου.

Δεν θα πρέπει να προκαλέσει έκπληξη ότι πολλά γειτονικά σημεία εμφανίζονταν ως κέντρα για κάθε κύκλο της εικόνας. Πραγματικά, αν παρατηρήσει κανείς την εικόνα μετά την ανίχνευση ακμών, θα δει ότι οι κύκλοι είναι παχύς και συνεπώς και στον χώρο Hough, τα μέγιστα εμφανίζονται σε μία ευρεία περιοχή γύρω από το πραγματικό κέντρο. Προφανώς χρειαζόταν κάποιος αλγόριθμος για ένωση (clustering) των γειτονικών σημείων για το ίδιο κέντρο.

Κάτι τέτοιο δεν είναι δύσκολο και υλοποιείται με τη συνάρτηση clusterPoint.

```
void clusterPoint(unsigned short x, unsigned short y, unsigned char r, unsigned
short value) {
	short i;

	i = findClosePoint(x, y, r);
	if (i < 0) {
		addPoint(x, y, r, value);
	}
	else {
		mergePoint(i, x, y, r, value);
	}
}
```

Κάθε κέντρο κύκλου αποθηκεύεται σε μία λίστα με μέγιστο 1024 κέντρα. Για κάθε νέο τυχαίο σημείο που ανιχνεύεται ως κέντρο κύκλου γίνεται ένας έλεγχος για το αν είναι «κοντά» με κάποιο κέντρο από τους είδη καταχωρημένους κύκλους. Αν αυτό δεν συμβαίνει, τότε καταχωρείται ως κέντρο νέου κύκλου. Στη περίπτωση που υπάρχει καταχωρημένος κύκλος κοντά, τότε γίνεται συγχώνευση του σημείου με τον καταχωρημένο κύκλο και ανάλογα με την ισχύ (τιμή) του μετασχηματισμού Hough για το νέο σημείο, αυτό συνεισφέρει (συμψηφίζεται) στην τιμή του κύκλου.

Μετά το τέλος αυτής της διαδικασίας έχουμε παρουσιάζονται πολλά κέντρα κοντά το ένα στο άλλο τα οποία προέρχονται από σημεία του χώρου από διαφορετικές περιοχές τα οποία όμως σιγά σιγά με τους συμψηφισμούς, φάνηκε ότι συγκλίνουν στο ίδιο κέντρο. Εφαρμόζεται λοιπόν μία συνάρτηση mergeNeighbours που ενώνει και τα τελικά σημεία τα οποία βρίσκονται κοντά.

Μετά από αυτή τη τεχνική έχουμε πλέον τα κέντρα των σημαντικότερων κύκλων της εικόνας μας και μάλιστα με μερικά δυαδικά (αντίστοιχο του δεκαδικά) ψηφία ακρίβεια τα οποία προκύπτουν από τους συμψηφισμούς.

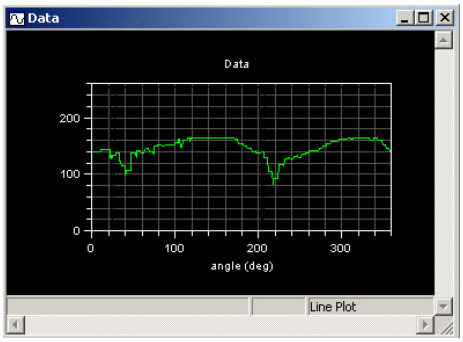
Σημαντική προσπάθεια δόθηκε ώστε η διαδικασία συμψηφισμού να έχει ακρίβεια στον συμψηφισμό των σημείων και ας είναι και αυτή υλοποιημένη με ακέραια αριθμητική. Η όλη αυτή τεχνική του clustering, παρουσιάζει πλεονεκτήματα έναντι εναλλακτικών τεχνικών εξ' αιτίας της μεγάλης ταχύτητας εκτέλεσης, αφού δρα πάνω σε έναν απλό διανυσματικό χώρο τριών σημείων. Θα έπρεπε τέλος να σημειώσουμε ότι στην αναγνώριση του προτύπου δεν χρησιμοποιήθηκαν οι μικροί (εσωτερικοί) κύκλοι του προτύπου, αφού αναγνώριση μόνο με τους μεγάλους (εξωτερικούς) κύκλους έδινε ικανοποιητικά αποτελέσματα.

6.10 Υλοποίηση αναγνώρισης οπών

Μετά από την ανίχνευση των κύκλων είμαστε πλέον έτοιμοι να προχωρήσουμε σε ανίχνευση της γωνίας των οπών πάνω στην περιφέρεια του δακτυλίου (συνάρτηση angularClassify). Για να το πετύχουμε αυτό χρειαζόμαστε σύμφωνα με τα συμπεράσματα της ενότητας 5.3 το «γωνιακό προφίλ» της έντασης του φωτισμού πάνω στον δακτύλιο. Αυτή τη φορά δε θα χρησιμοποιήσουμε τελεστή ανίχνευσης ακμών όπως στην ενότητα 5.3 λόγω του ότι η ανάλυση της εικόνας είναι σχετικά μικρή και κάτι τέτοιο δεν θα είχε ουσιαστικό νόημα.

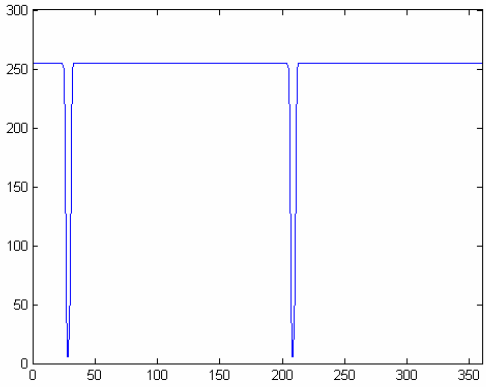
Θα χρησιμοποιήσουμε για την εξαγωγή του κύκλου, έναν πυρήνα blurring 5x5. Η μεγάλη αυτή τιμή είναι απαραίτητη για να αντιμετωπίσουμε τυχόν ανακρίβειες στον υπολογισμό του κέντρου, όσο και της ακτίνας. Ο πυρήνας αυτός είναι ο ίδιος που παρουσιάστηκε στο κεφάλαιο 5.3, Εικόνα 77.Οι γωνίες οι οποίες χρησιμοποιούνται ως κέντρα για τον πυρήνα αυτό παράγονται από ημίτονα και

συνημίτονα γωνιών με διάφορα τεχνάσματα και πάλι σε ακέραια αριθμητική. Η ακτίνα που χρησιμοποιείται είναι το 89% της ακτίνας του εξωτερικού δακτυλίου που ανιχνεύσαμε.



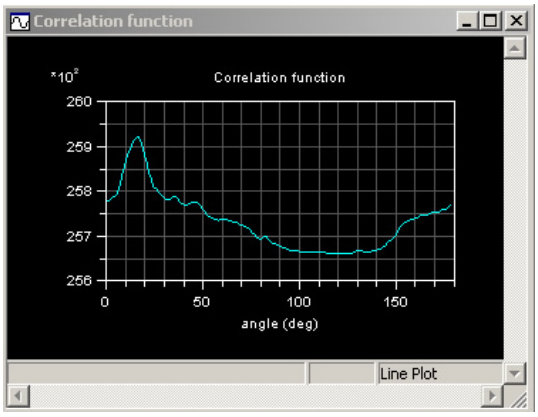
Εικόνα 99. Γωνιακό προφίλ φωτεινότητας

Το γωνιακό προφίλ όπως και στην ενότητα 5.4.3 είναι γεμάτο θόρυβο. Όπως βλέπουμε στην Εικόνα 99, τα ελάχιστα ξεχωρίζουν με το μάτι αλλά δεν μπορεί να χρησιμοποιηθεί ένα κατώφλι για να μας δώσει αξιόπιστη αναγνώριση της γωνίας. Προχωράμε και πάλι στην δημιουργία ενός προτύπου με τη βοήθεια του Matlab™ (createFinalPattern.m).



Εικόνα 100. Πρότυπο οπών κατασκευασμένο από το Matlab

Το πρότυπο αυτό όπως φαίνεται στην Εικόνα 100 παρουσιάζει δύο ελάχιστα με 180^{o} διαφορά που αντιστοιχούν στις δύο μαύρες κουκίδες στην περιφέρεια. Στο πρότυπο και πάλι συνυπολογίστηκε η συμβολή του θορύβου με την συνέλιξη με μία γκαουσιανή. Είμαστε λοιπόν έτοιμοι να εφαρμόσουμε τη συνάρτηση συσχέτισης όπως και στο κεφάλαιο 5.4.3.



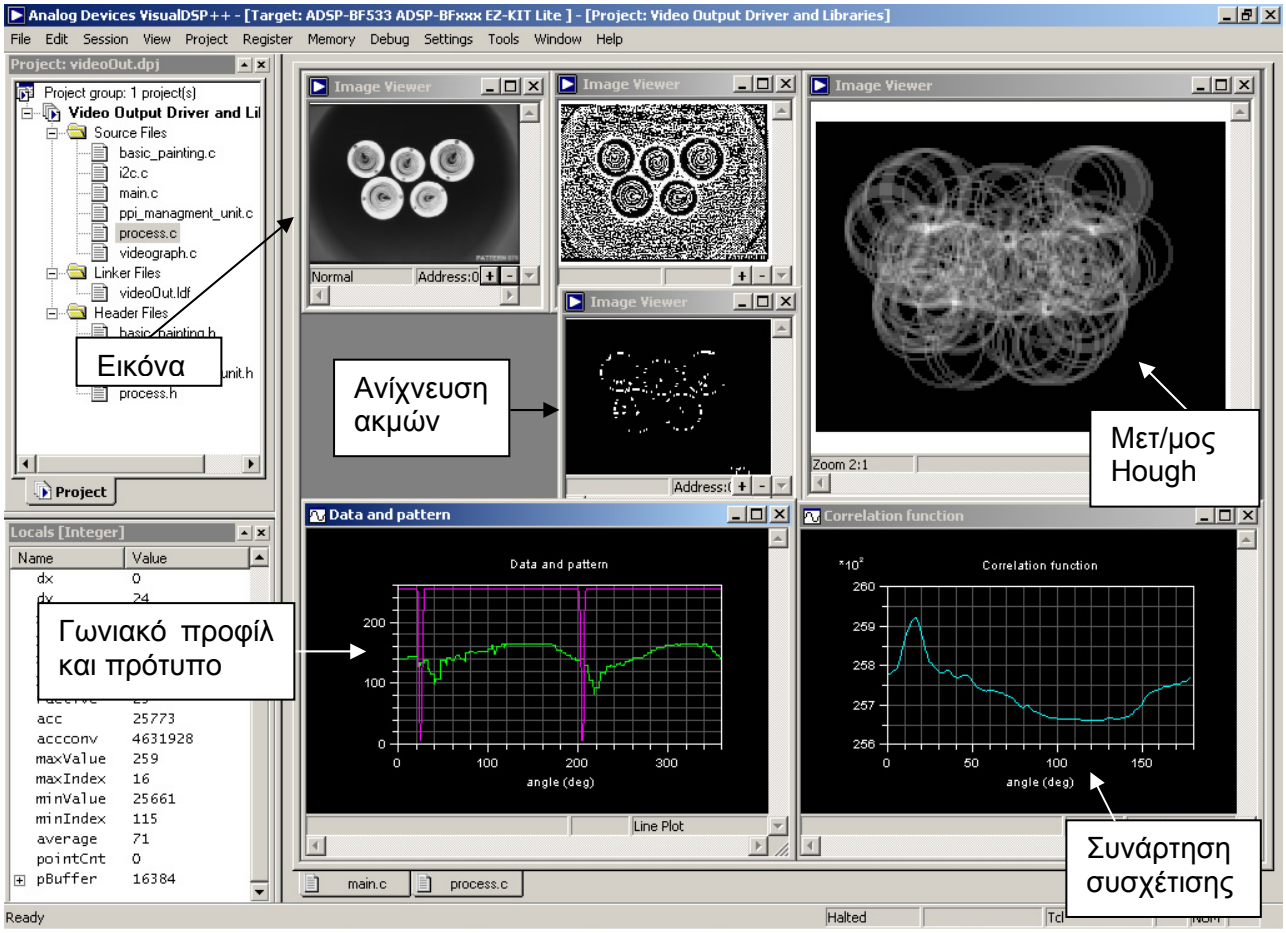
Εικόνα 101. Συσχέτιση σήματος με το πρότυπο

Όπως μπορεί να δει κανείς στην Εικόνα 101 τα αποτελέσματα είναι ιδανικά. Το σήμα καθάρισε δίνοντας ένα σήμα με ένα μοναδικό καθαρό μέγιστο που αντιστοιχεί στη γωνία των οπών.

Για να εκτιμήσουμε αν στην εικόνα βρίσκουμε πράγματι ξεκάθαρες οπές ή όχι, υπολογίζουμε το μέγιστο, το ελάχιστο και την μέση τιμή της συνάρτησης συσχέτισης. Αφαιρούμε από το μέγιστο και τη μέση τιμή την ελάχιστη τιμή (αναφορά στο 0) και αποδεχόμαστε την ανίχνευση της γωνίας αν το μέγιστο είναι τουλάχιστον δύο φορές όσο η μέση τιμή. Αυτό αποδεικνύεται ένα αποτελεσματικό κριτήριο.

6.11 Ολοκλήρωση συστήματος

Το σύστημά μας είναι ικανό να αναγνωρίσει επιτυχώς πολλούς MDTs σε μία εικόνα.



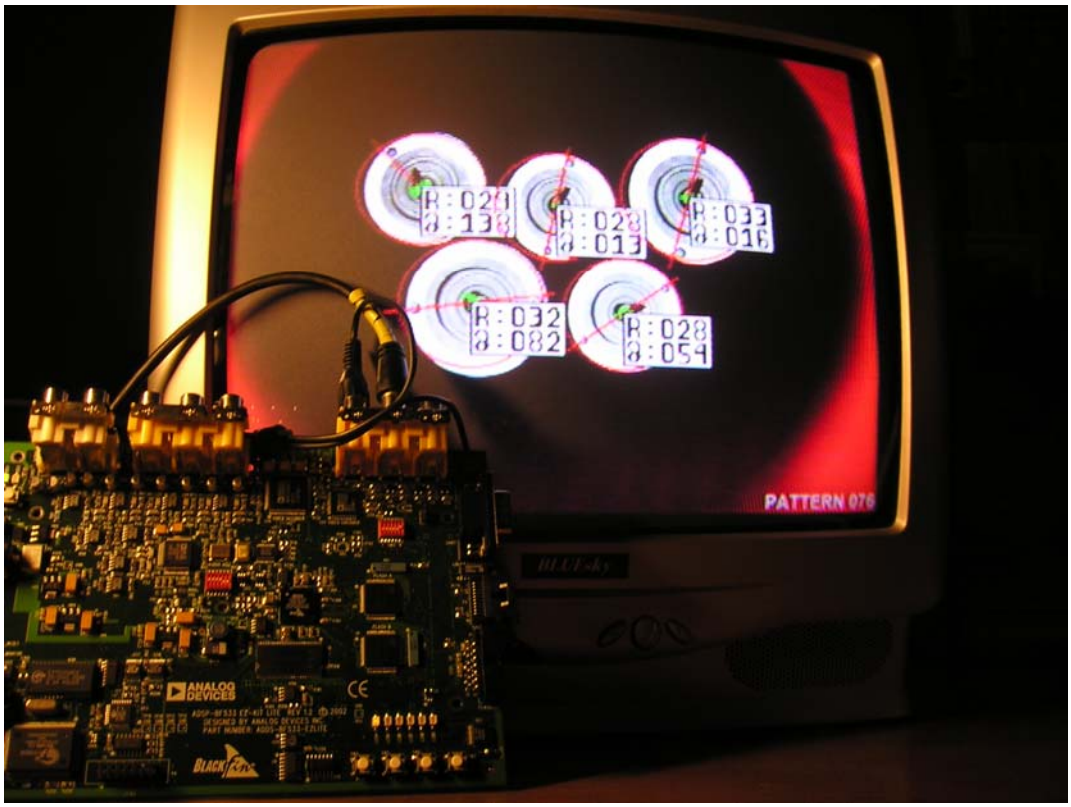
Εικόνα 102. Οι «σκέψεις» του DSP πριν την απόφαση

Συνοψίζοντας τη διαδικασία όπως φαίνεται στην Εικόνα 102 αυτή έχει ως εξής.

1. Στην αρχή το σύστημα βρίσκεται στην κατάσταση αναμονής
2. Με το πάτημα του αριστερότερου πλήκτρου ξεκινάει η σύλληψη εικόνας
3. Στη συνέχεια γίνεται ανόρθωση αναλογιών, σμίκρυνση και αντιγραφή στον buffer του σήματος video. Επίσης ανιχνεύονται οι ακμές με βάση τιμή κατωφλιού επιλεγόμενη από τα δύο πλήκτρα στα δεξιά.
4. Εκτέλεση μετασχηματισμού Hough
5. Αναγνώριση κύκλων στον χώρο Hough. Κάθε σημείο με τιμή ανώτερη ενός threshold μαρκάρεται με ένα πράσινο σταυρό στην οθόνη. Στη συνέχεια γίνεται ομαδοποίηση των σημείων και για κάθε κύκλο αναγνώριση γωνίας οπών πάνω στον δακτύλιο χρησιμοποιώντας συνάρτηση συσχέτισης με πρότυπο. Ζωγραφίζονται με κίτρινο χρώμα κύκλοι στους οποίους δεν ανιχνεύτηκαν οπές και εμφανίζονται στοιχεία μόνο για την ακτίνα τους. Ζωγραφίζονται με κόκκινο χρώμα κύκλοι στους οποίους ανιχνεύτηκαν οπές, ζωγραφίζεται ευθεία στην κατεύθυνση των οπών και εμφανίζονται πλήρη στοιχεία.

Για κάθε ένα από τα παραπάνω βήματα ανάβει ένα ενδεικτικό Led του αναπτυξιακού, με τη βοήθεια του οποίου μπορεί ο χειριστής να εκτιμήσει τον χρόνο που απαιτείται για κάθε εργασία. Με τις υπάρχουσες προδιαγραφές, ο χρόνος για κάθε ένα από τα παραπάνω βήματα είναι σχετικά μικρός.

Τέλος με το δεύτερο από τα αριστερά πλήκτρο, γίνεται ενεργοποίηση και απενεργοποίηση της ένδειξης της διαδρομής πάνω στην οποία υπολογίζεται το γωνιακό προφίλ της φωτεινότητας του δακτυλίου για την ανίχνευση οπών. Η κατάσταση εμφανίζεται στο έκτο Led του συστήματος.



Εικόνα 103. Το σύστημα σε λειτουργία

Το σύστημα δοκιμάζεται επιτυχώς με αρκετές δοκιμαστικές εικόνες και «ζωντανό» video. Οι επιδόσεις του είναι εξαιρετικές όταν υπάρχει μεγάλη αντίθεση με το φόντο ενώ υπάρχει σημαντικό πρόβλημα όταν το φόντο αποτελείται από επαναλαμβανόμενα γεωμετρικά πρότυπα, κάτι το οποίο περιμέναμε λόγω του μετασχηματισμού Hough. Η μεγαλύτερη πηγή σφαλμάτων είναι ο αλγόριθμος ανίχνευσης ακμών που ενισχύει την ύπαρξη θορύβου και εμφανίζει περιορισμένη ευαισθησία στην ανίχνευση εκτεταμένων αλλά λιγότερο απότομων ακμών.

6.12 Προγραμματισμός της Flash

Για να έχουμε μία ολοκληρωμένη εφαρμογή που τρέχει αυτόνομα χωρίς να χρειάζεται κάθε φορά να φορτώσουμε το πρόγραμμα από τον υπολογιστή, έχει προβλεφθεί η ύπαρξη της μνήμης Flash πάνω στο αναπτυξιακό. Το εργαλείο με το οποίο πραγματοποιείται ο προγραμματισμός λέγεται Flash Programmer (Tools > Flash Programmer…). Επιπλέον χρειάζονται κάποιες ρυθμίσεις των ιδιοτήτων του project ώστε να μην εξάγεται πλέον εκτελέσιμο αρχείο (.dxe) αλλά αρχείο για τον Programmer (.ldr). Αυτές οι ρυθμίσεις, όπως και οι απαραίτητη ενεργοποίηση του Boot Kernel που αναλαμβάνει το φόρτωμα του προγράμματος από τις εξωτερικές Flash στην μνήμη του επεξεργαστή, γίνονται μέσα από τα Project Options (Project > Project Options). Η πλήρης διαδικασία περιγράφεται στο αντίστοιχο Engineer to Engineer Note της Analog Devices[84]. Η τελική διάταξη είναι αυτόνομη και εκτελεί το πρόγραμμα κατευθείαν, χωρίς καμία διασύνδεση με PC.

[84] EE-239, Running Programs from Flash on ADSP-BF533 Blackfin® Processors, Steve K, 2004

Κεφάλαιο 7. Αξιολόγηση εργασίας και μελλοντικές επεκτάσεις

Στο κεφάλαιο αυτό γίνεται αξιολόγηση της απόδοσης της όλης διάταξης και ωφέλιμη κριτική σχετικά με τις τεχνικές που χρησιμοποιήθηκαν. Το κεφάλαιο αυτό συμπληρώνεται από προτάσεις για το πως διάφορες τεχνικές που χρησιμοποιήθηκαν μπορούν να αξιοποιηθούν και σε άλλες εφαρμογές όπως και πιθανές εφαρμογές ολόκληρης της διάταξης. Το κεφάλαιο αυτό θα φανεί χρήσιμο σε όποιον θελήσει να αξιοποιήσει περαιτέρω το περιεχόμενο αυτής της διπλωματικής.

7.1 Αξιολόγηση εργασίας

Η εργασία αυτή είναι ολοκληρωμένη σε πολύ μεγάλο βαθμό. Παρουσιάστηκαν αναλυτικά όλες οι τεχνικές και τεχνολογίες οι οποίες χρησιμοποιούνται και στη συνέχεια εφαρμόστηκαν για την λύση του πρακτικού προβλήματος της αναγνώρισης ενός MDT και της γωνίας των οπών στήριξής τους σε μία εικόνα. Το σύστημα που αναπτύχθηκε καλύπτει πλήρως τις απαιτήσεις μας όπως παρουσιάζεται και παρακάτω.

Δυνατότητες και χαρακτηριστικά συστήματος

Αναγνώριση θέσης MDT	Το σύστημα είναι σε θέση όχι μόνο να αναγνωρίζει την θέση ενός MDT αλλά ακόμα και περισσότερων σε μία εικόνα.
Μέτρηση γωνίας οπών στήριξης	Το σύστημα είναι σε θέση να μετρήσει την γωνία των οπών στήριξης αν αυτή είναι ευδιάκριτη για όλους τους MDTs που ανιχνεύει. Σε περίπτωση που ανιχνευθεί MDT αλλά δεν είναι δυνατή η ανίχνευση γωνίας αυτός σημειώνεται με διαφορετικό χρώμα (κίτρινο).
Εποπτική παρουσίαση	Το σύστημα παρουσιάζει άμεσα τα αποτελέσματα σε μία οθόνη. Εμφανίζονται οι ακτίνες των ανιχνευόμενων κύκλων, και η γωνία των οπών στήριξης με ειδική ταμπέλα στο κέντρο του κύκλου που αναγνωρίστηκε. Με πράσινους σταυρούς σημειώνονται τα σημεία πριν το clustering, πληροφορία χρήσιμη κατά την εγκατάσταση.
Εύκολη εγκατάσταση	Το σύστημα εγκαθίσταται με την απλή τοποθέτηση μίας κάμερας ώστε να «βλέπει» τους MDTs. Είναι καλό να ξεχωρίζουν οι MDTs σε σχέση με το φόντο ως προς την φωτεινότητά τους και το φόντο να έχει όσο το δυνατόν πιο ομοιόμορφο χρώμα ή απλώς λευκό θόρυβο (χιονάκι) αντί για επαναλαμβανόμενα γεωμετρικά σχήματα. Η μόνη ρύθμιση που χρειάζεται είναι η ρύθμιση ευαισθησίας που γίνεται με δύο πλήκτρα του αναπτυξιακού.
Φορητή διάταξη	Η διάταξη είναι φορητή και έχει πάρα πολύ εύκολη εγκατάστασή. Συγκεκριμένα τα τρία εξαρτήματα που χρειάζονται είναι μία συνήθης κάμερα, το αναπτυξιακό ADSP-BF533 EZ-KIT LITE™ και μία τηλεόρασης ή άλλη συσκευή ικανή να δείξει σήμα video.
Ταχύτητα	Η ταχύτητα εκτέλεσης είναι κάτι παραπάνω από ικανοποιητική για τις ανάγκες της εφαρμογής. Αν και το πρόγραμμα είναι γραμμένο σε μη-βελτιστοποιημένη C, οι επεξεργαστικές δυνατότητες του BF533 σε συνδυασμό με το σωστό «ζύγισμα» των απαιτήσεων της συσκευής όσον αφορά την ακρίβεια, κάνουν την απόδοσή της ιδανική. Το «κλειδί» για την υψηλή απόδοση είναι η υλοποίηση εξολοκλήρου με ακεραίους. Αν θέλαμε να έχουμε συνεχή ροή σήματος video θα έπρεπε να γίνουν βελτιώσεις στην ταχύτητα εκτέλεσης όπως θα εξηγήσουμε παρακάτω.
Ακρίβεια	Το σύστημα παρουσιάζει ικανοποιητική ακρίβεια. Η σμίκρυνση της λαμβανόμενης από την κάμερα εικόνας προκειμένου να γίνεται γρήγορη επεξεργασία οδηγεί σε μικρές ανακρίβειες. Αν δεν μας αρκεί η ακρίβεια της συσκευής όπως μπορούμε με πολύ μικρές τροποποιήσεις στο πρόγραμμα να πετύχουμε θεαματική αύξησή της, όπως θα εξηγήσουμε παρακάτω.
Χαμηλό κόστος	Το κόστος είναι εξαιρετικά χαμηλό ιδιαίτερα σε σύγκριση με ένα PC. Αυτό ίσως δεν είναι τόσο προφανές λόγω του ότι το αναπτυξιακό κοστίζει περίπου 400€ (τελική τιμή), το λογισμικό περίπου 1400€ (πανεπιστημιακή έκδοση) και σίγουρα χρειάζονται περισσότερες εργατώρες για τον προγραμματισμό σε DSP. Όμως και τα τρία αυτά ανήκουν στην κατηγορία του κόστους που δεν επαναλαμβάνονται NRE (Non – recurring expenses). Το software και το

αναπτυξιακό θα χρησιμοποιηθούν για την ανάπτυξη και άλλων εφαρμογών. Το κόστος τους μοιράζεται σε όλες τις εφαρμογές και η απόσβεση γίνεται εύκολα. Οι εργατώρες επίσης καταναλώνονται μόνο κατά την ανάπτυξη της εφαρμογής και συνεπώς διαιρούνται με τον αριθμό των τεμαχίων που θα παραχθούν για την συγκεκριμένη εφαρμογή. Το συνολικό κόστος μίας συσκευής η οποία θα είχε τα απολύτως απαραίτητα (BF533, 8Mb DRAM, 1Mb FLASH, ρολόι χρονισμού και τους video encoders - decoders) δεν ξεπερνάει αυτή τη στιγμή τα 120€ ακόμα και για λίγα τεμάχια. Το αντίστοιχο κόστος ενός PC με κάρτα σύλληψης video είναι τουλάχιστον διπλάσιο. Η κάμερα και η οθόνη δεν συμπεριλαμβάνονται στην μελέτη γιατί θεωρούμε ότι τα απαιτούν και οι δύο εκδοχές. Συνεπώς έχοντας μία μείωση του κόστους κατά 120€ τουλάχιστον σε σύγκριση με την ισοδύναμη λύση με PC, με 15 τεμάχια έχουμε ουσιαστικά αποσβέσει πλήρως το software και το αναπτυξιακό. Έστω ότι με άλλα 25 τεμάχια εξοφλούμε και την ανάπτυξή του (1500 γραμμές κώδικα). Στην ουσία λοιπόν, μετά από τα πρώτα 40 τεμάχια έχουμε καθαρό κέρδος 120€ ανά τεμάχιο επειδή επιλέξαμε DSP αντί για PC.

7.2 Βελτιώσεις και μελλοντικές επεκτάσεις

Παρόλο που το σύστημα λειτουργεί ικανοποιητικά για τις προδιαγραφές μας, θα μπορούσαν να γίνουν ορισμένες βελτιώσεις.

7.2.1 Καλύτερη ανίχνευση ακμών (edge detection)

Η μεγαλύτερη πηγή σφαλμάτων σε αυτό το σύστημα είναι, αυτή τη στιγμή, ο αλγόριθμος ανίχνευσης ακμών. Ο αλγόριθμος “Canny” δίνει σημαντικά καλύτερα αποτελέσματα και είναι λίγο πιο σύνθετος στην υλοποίηση[85]. Ίσως θα μπορούσαν να εφαρμοστούν και άλλοι αλγόριθμοι που να δίνουν ακόμα καλύτερα αποτελέσματα. Η βελτίωση της ανίχνευσης ακμών θα έδινε ορθή ανίχνευση κύκλων ακόμα και σε περισσότερο θορυβώδη φόντα.

7.2.2 Αυτόματη ανίχνευση κατωφλιού

Για το edge detection χρειάζεται ένα κατώφλι αναφοράς το οποίο στη συσκευή μας ρυθμίζεται μέσω δύο πλήκτρων. Αυτό φυσικά δεν είναι πρόβλημα γιατί μία τέτοια ρύθμιση είναι απαραίτητη μόνο κατά την εγκατάσταση. Στην πράξη, είναι δυνατόν με βάση στατιστικά μεγέθη της εικόνας (ιστόγραμμα) να υπολογιστεί αυτό το κατώφλι αυτόματα, όπως γίνεται και στην συνάρτηση edge του Matlab™. Με την βελτίωση αυτή δεν θα χρειαζόταν καμία ρύθμιση κατά την εγκατάσταση της συσκευής και θα γινόταν ανθεκτική σε περιοδικές αλλαγές του φωτισμού (σκίαση από ανθρώπους, εναλλαγή ημέρας-νύχτας κ.τ.λ.).

7.2.3 Αύξηση ακρίβειας

Αν δεν μας ικανοποιεί η ακρίβεια είναι πάρα πολύ εύκολο να την βελτιώσουμε. Αυτή τη στιγμή το σύστημα δουλεύει πάνω σε ένα πλαίσιο εικόνας (miniFrame) με διαστάσεις 168x128. Με μικρές τροποποιήσεις σε ένα μόνο αρχείο (process.c) μπορεί να βελτιωθεί δραματικά η ακρίβεια του συστήματος τόσο για τον υπολογισμό της θέσης των κύκλων, όσο και για τον υπολογισμό των γωνιών στήριξης. Αυτό όμως θα έχει άμεσες συνέπειες στις απαιτήσεις μνήμης και υπολογιστικής ισχύος. Ο διπλασιασμός για παράδειγμα του μεγέθους πλαισίου σε 336x 256 θα είχε ως άμεση συνέπεια (χοντρικά) τον τετραπλασιασμό των απαιτήσεων μνήμης και υπολογιστικής ισχύος. Η μνήμη φυσικά δεν αποτελεί πρόβλημα. Η υπολογιστική ισχύς επίσης μπορεί να αξιοποιηθεί καλύτερα όπως θα περιγράψουμε παρακάτω. Συνεπώς η αύξηση της ακρίβειας είναι κάτι το οποίο μπορεί να επιτευχθεί άνετα με την ίδια διάταξη με απλή αναβάθμιση του λογισμικού.

[85] http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT6/node2.html

7.2.4 Σύνθετη αναγνώριση προτύπου

Θα μπορούσαμε να αναγνωρίζουμε το πρότυπο του tube με βάση τη μεθοδολογία που αναπτύξαμε στην ενότητα 2.5.1. Το γεγονός ότι δεν απέδωσε αυτή η μέθοδος οφείλεται στην χαμηλή ποιότητα του ανιχνευτή ακμών, στην περιορισμένη ανάλυση της σμικρυμένης εικόνας και στην απαίτησή μας για μία απλή λύση. Αναπτύσσοντας τον ανιχνευτή όπως περιγράφηκε στην ενότητα 2.5.1. μπορούμε να έχουμε δραματική βελτίωση, ανάλογη με αυτή που πετύχαμε με τη συνάρτηση συσχέτισης για την ανίχνευση γωνιών. Το σημαντικό είναι ότι με αυτό το βήμα θα είναι δυνατή η αναγνώριση του MDT ακόμα και σε αρκετά θορυβώδες φόντο ακόμα και με κυκλικά σχέδια.

7.2.5 Συνεχής ροή video

Αν και στις περισσότερες εφαρμογές δεν είναι απαραίτητο, θα ήταν όμορφο να αναπτυσσόταν η εφαρμογή έτσι ώστε να μπορούσε να λαμβάνει video από την κάμερα, να το επεξεργάζεται και να παράγει σήμα video για την τηλεόραση ταυτόχρονα, με ταχύτητα 25 πλαισίων ανά δευτερόλεπτο. Αυτό προγραμματιστικά είναι εύκολο να γίνει. Το μόνο που χρειάζεται είναι να χρησιμοποιηθούν δύο buffers που από τον ένα θα αναπαράγεται το video ενώ στον άλλο θα γράφεται η εικόνα από την κάμερα και την επεξεργασία (double buffering). Αυτό το οποίο ουσιαστικά αναιρεί τη δυνατότητα αυτή είναι η αδυναμία του αναπτυξιακού (hardware) για την ταυτόχρονη λήψη και αποστολή σημάτων μέσω του PPI. Ο νέος επεξεργαστής ADSP-BF561 έχει δύο θύρες PPI και συνεπώς κάνει δυνατή την ταυτόχρονη εγγραφή και ανάγνωση video. Επιπλέον έχει διπλάσια επεξεργαστική ισχύ. Αν λοιπόν εκτελείτω η εφαρμογή στο αναπτυξιακό αυτού του BlackFin θα μπορούσε με μικρές τροποποιήσεις να υποστηρίξει real time video. Θα έπρεπε να σημειώσουμε βέβαια ότι η ταυτόχρονη επεξεργασία και λήψη θα επιβαρύνει τον ελεγκτή DMA και επιπλέον ότι στην ουσία ο επεξεργαστής θα έχει μέγιστο 1/25 = 40ms για να ολοκληρώσει την επεξεργασία και την αναγνώριση του πλαισίου. Κάτι τέτοιο είναι εύκολο να γίνει όπως εξηγούμε παρακάτω. Μία δυνατότητα που μας δίνεται με την αναγνώριση video σε πραγματικό χρόνο, είναι ότι μπορούμε να μετράμε την ταχύτητα μετακίνησης και την ταχύτητα περιστροφής για κάθε MDT του πλάνου.

7.2.6 Βελτίωση ταχύτητας επεξεργασίας

Με τις παρούσες προδιαγραφές, δεν είναι αναγκαία η βελτιστοποίηση της ταχύτητας εκτέλεσης του κώδικα. Αν όμως θελήσουμε να κάνουμε κάποιες από τις βελτιώσεις που αναφέραμε παραπάνω, ταχύτερος κώδικα είναι αναγκαίος. Αυτό μπορεί να γίνει κατά κύριο λόγο με τρεις τρόπους.

7.2.6.1 Χρήση του optimizer

Το compile αυτή τη στιγμή έχει γίνει χωρίς την ενεργοποίηση των optimizations, κυρίως γιατί κατά την φάση του debugging θέλουμε να μπορούμε να εκτελούμε κώδικα βήμα – βήμα. Όταν ενεργοποιείται το optimization, λόγω της ομαδοποίησης και αναδιάταξης του κώδικα για την επίτευξη μέγιστης ταχύτητας, γενικά δεν έχουμε κανονική αντιστοίχηση από τις εντολές σε C στις εντολές σε assembly. Εκτελώντας λοιπόν τον αλγόριθμο βήμα – βήμα θα παρατηρήσουμε μία σχεδόν τυχαία κίνηση του δείκτη θέσης η οποία δυσκολεύει το debugging. Επιπλέον η αύξηση της ταχύτητας που επιτυγχάνεται με την ενεργοποίηση του optimizer, τουλάχιστον στην περίπτωση του δικού μας προγράμματος δεν ήταν τόσο θεαματική όσο θα θέλαμε.

7.2.6.2 Μετατροπή ρουτινών που εκτελούνται συχνά σε assembly

Εδώ μπορεί να μας βοηθήσει σημαντικά ο profiler. Με τη βοήθειά του μπορούμε να εντοπίσουμε τα χρονοβόρα κομμάτια κώδικα και να βελτιστοποιήσουμε την απόδοσή τους ξαναγράφοντάς τα σε assembly. Ο compiler μας δίνει την δυνατότητα να αναμειγνύουμε assembly και C. Με τον τρόπο αυτό μπορούν να αξιοποιηθούν καλύτερα τα hardware loops, αντί για for, και οι προχωρημένες εντολές αριθμητικής του επεξεργαστή. Πρέπει να σημειωθεί ότι είναι αρκετά συνηθισμένο στην ψηφιακή επεξεργασία σήματος, ένα πολύ μεγάλο μέρος (αν όχι όλο) του προγράμματος να γράφεται σε συμβολική γλώσσα. Στην διαδικασία του optimization, ο έμπειρος προγραμματιστής έχει προνόμιο έναντι του compiler, γιατί γνωρίζει το τι σημαίνουν τα δεδομένα που χειρίζεται μία

ρουτίνα. Για παράδειγμα αν ο προγραμματιστής γνωρίζει ότι η μέγιστη τιμή δύο μεταβλητών τύπου char είναι 60, και θέλει να τις προσθέσει τότε προφανώς μπορεί να παραλείψει τον κώδικα που χειρίζεται τα κρατούμενα γιατί 60+60 = 120 < 128, συνεπώς ποτέ δεν έχουμε κρατούμενο. Τέτοιες πληροφορίες δεν μπορεί να τους γνωρίζει ο compiler.

7.2.6.3 Χρήση λιγότερων DMA μεταφορών και μείωση μνήμης

Αρκετή μνήμη χρησιμοποιείται για την αποθήκευση των πλαισίων Video. Επίσης bandwidth των DMA καναλιών αναλώνεται για την μεταφορά τετριμμένων σημάτων συγχρονισμού. Όλα αυτά μπορούν να αποφευχθούν αν χρησιμοποιήσουμε πιο σωστά τα DMA κανάλια και τις δυνατότητες του ελεγκτή PPI. Συγκεκριμένα αν δεν μας χρειάζεται το δεύτερο πλαίσιο για την ακρίβεια της συσκευής μας μπορούμε να συλλάβουμε μόνο το πρώτο πλαίσιο και φυσικά χωρίς τα σήματα συγχρονισμού (active frame). Αυτό θα μειώσει στο μισό τον χρόνο που χρειάζεται για την σύλληψη εικόνας και τον φόρτο των καναλιών DMA για τη συγκεκριμένη λειτουργία. Στη συνέχεια μπορεί να χρησιμοποιηθεί Descriptor Based DMA λειτουργία για τη μεταφορά του σήματος video στον video encoder. Συγκεκριμένα μία γραμμή από κάθε είδος σήματος συγχρονισμού, το μονό ενεργό πλαίσιο και ένας εκτενής πίνακας οδηγιών προς το DMA κανάλι μπορούν να συνθέσουν ένα πλήρες interlaced σήμα video. Αυτό θα οδηγήσει σε ελαχιστοποίηση του μεγέθους της μνήμης που χρησιμοποιείται για την αποθήκευση του video και διπλασιασμό της ταχύτητας ζωγραφικής αφού πλέον οι αλγόριθμοι θα χρειάζεται να ζωγραφίζουν σε ένα μόνο πλαίσιο.

7.2.6.4 Χρήση της Cache

Όταν πλέον έχουμε βελτιστοποιήσει αρκετά τον κώδικα, παρατηρούμε ότι ο περισσότερος χρόνος εκτέλεσης του προγράμματος, σπαταλάται για την εγγραφή και την ανάγνωση της εξωτερικής SDRAM. Αυτό είναι αναμενόμενο, αφού παρόλο που ο επεξεργαστής «τρέχει» στα 600MHz, η εξωτερική μνήμη «τρέχει» στα 133MHz. Συνεπώς αλγόριθμοι οι οποίοι κάνουν εκτεταμένη χρήση εγγραφών και αναγνώσεων από την μνήμη όπως ο μετασχηματισμός Hough θα εκτελούνται πάρα πολύ αργά. Η μόνη λύση για το πρόβλημα αυτό είναι η χρήση της εσωτερικής μνήμης Cache. Αυτή «τρέχει» με την ταχύτητα του πυρήνα. Η εγγραφή σε αυτή από την εξωτερική μνήμη γίνεται μέσω DMA καναλιού. Με αυτό τον τρόπο, όταν για παράδειγμα ο επεξεργαστής είναι απασχολημένος με την επεξεργασία δεδομένων του βήματος Α, μπορεί παράλληλα να έχει δώσει εντολή για cach-άρισμα μία περιοχής της μνήμης που θα χρησιμοποιηθεί στο επόμενο βήμα Β. Με αυτό τον τρόπο είναι δυνατή η επιπλέον επιτάχυνση της ταχύτητας εκτέλεσης του κώδικα.

Μία χρήσιμη και απλή επέκταση που μπορεί να γίνει είναι η αποστολή δεδομένων μέσω της σειριακής θύρας σε ένα PC ή καταγραφικό. Στους βιομηχανικούς αυτοματισμούς, περιβάλλον όπου θα μπορούσε να χρησιμοποιηθεί η συσκευή μας, η καταγραφή μετρήσεων παίζει πάρα πολύ σημαντικό ρόλο και είναι απαραίτητη για κάθε επαγγελματικό εργαλείο.

Η σημαντικότερη όμως μελλοντική επέκταση θα ήταν η ενσωμάτωση του συστήματος αναγνώρισης εικόνας που σχεδιάσαμε και υλοποιήσαμε σε ένα σύστημα αυτομάτου ελέγχου για κάποια διεργασία σχετική με τα MDTs. Θα πρέπει να σημειωθεί ότι ο επεξεργαστής έχει αρκετή επεξεργαστική ισχύ ώστε να μπορεί να εκτελέσει και τον κώδικα ελέγχου του συστήματος αυτοματισμού, αν αυτός δεν είναι υπερβολικά σύνθετο.

7.3 Αξιολόγηση τεχνολογιών

Για την εργασία αυτή συνδυάσαμε πολλές τεχνικές από διαφορετικούς τομείς. Η τεχνογνωσία που αποκτήθηκε μπορεί να βοηθήσει στην λύση πολλών προβλημάτων.

7.3.1 Μετασχηματισμός Radon (Hough)

Για την αναγνώριση κυκλικών σχημάτων στην εικόνα μας χρησιμοποιήσαμε τον μετασχηματισμό Radon. Ο μετασχηματισμός ήταν αποτελεσματικός για την εφαρμογή μας και μας έδωσε ικανοποιητικά αποτελέσματα όσον αφορά την αναγνώριση κυκλικών σχημάτων.

Η υλοποίησή που κάναμε σε DSP δεν εκμεταλλεύεται με τον καλύτερο τρόπο τις δυνατότητες παράλληλης επεξεργασίας των μονάδων MAC και για τη χρήση του απαιτούνται πολλές εγγραφές και αναγνώσεις από τη μνήμη. Επιπλέον η λειτουργία του επιδρά για κάθε σημείο στον χώρο της εικόνας πάνω σε μία ευρεία περιοχή στον χώρο Hough (Radon) πράγμα που κάνει δυσκολότερο το σπάσιμό του σε κομμάτια (διαίρει και βασίλευε) και τη χρήση της cache για την επιτάχυνση διεργασιών, σε σύγκριση με τους κλασικούς FFT. Σίγουρα θα ήταν ενδιαφέρουσα η μελέτη των ιδιοτήτων του μετασχηματισμού Hough (Radon) με τη βοήθεια των οποίων θα μπορούσε να επιταχυνθεί η ταχύτητα εκτέλεσής του.

Ο μετασχηματισμός αυτός είναι πάρα πολύ χρήσιμος και μπορεί να χρησιμοποιηθεί σε πολλές εφαρμογές αναγνώρισης εικόνας και ανακατασκευής σημάτων. Απαιτεί όμως προσεκτικό ζύγισμα σε επίπεδο προδιαγραφών ώστε το υπολογιστικό του κόστος να κυμαίνεται σε λογικά επίπεδα.

Μία εφαρμογή στην οποία θα μπορούσε να εφαρμοστεί ο κυκλικός μετασχηματισμός Radon (Hough) είναι στην αναγνώριση ίριδας για έλεγχο πρόσβασης. Με τη βοήθεια του μετασχηματισμού αυτού, θα αναγνωρίζονται με ευκολία οι κύκλοι που ορίζουν την ίριδα. Στη συνέχεια με μία σπειροειδή διαδρομή από τον εσωτερικό προς τον εξωτερικό κύκλο μπορούμε να έχουμε ένα μονοδιάστατο προφίλ της ίριδας, ανάλογο με αυτό που χρησιμοποιήσαμε κι εμείς για την αναγνώριση των οπών. Στη συνέχεια με την χρήση συναρτήσεων συσχέτισης μπορούμε να εξάγουμε κατά πόσο ταιριάζουν ή όχι τα προφίλ. Το πλεονέκτημα αυτής της τεχνικής είναι ότι δρα ανεξάρτητα από το πόσο διασταλμένη είναι η ίριδα, αφού αναγνωρίζει τον εσωτερικό και εξωτερικό κύκλο της και προσαρμόζεται ανάλογα. Η χρήση μίας κάμερας με μεγάλη ανάλυση είναι απαραίτητη για την επιτυχία αυτής της τεχνικής.

7.3.2 Η επεξεργασία εικόνας για μετρήσεις

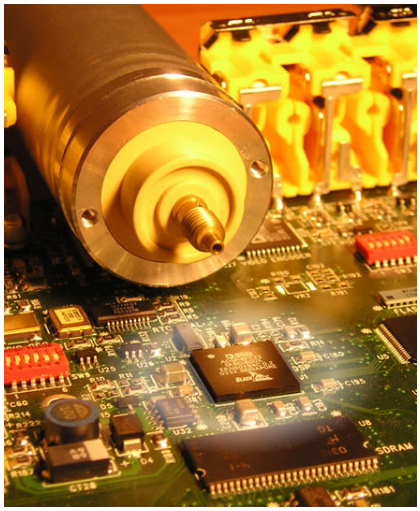
Το πρόβλημά μας ήταν ένα πρόβλημα μέτρησης το οποίο το λύσαμε με επιτυχία με τεχνικές αναγνώρισης εικόνας. Επαληθεύσαμε λοιπόν με την εργασία αυτή ότι η επεξεργασία εικόνας δεν περιορίζεται μόνο σε εφαρμογές που σχετίζονται με τη βελτίωση εικόνων και την ορθή παρουσίαση τους σε κάποιο χειριστή. Η επεξεργασία εικόνας μπορεί να χρησιμοποιηθεί αποτελεσματικά για την μέτρηση ποσοτήτων και μάλιστα παρουσιάζει σημαντικά πλεονεκτήματα έναντι εναλλακτικών τεχνικών.

Οικονομία	Τα έξοδα μίας διάταξης επεξεργασίας είναι πολύ συγκεκριμένα. Μία κάμερα, ένα υπολογιστικό σύστημα και πιθανώς κάποιο monitor. Μία διάταξη που αποτελείται από τα παραπάνω μπορεί να αντικαταστήσει πολύπλοκες μετρητικές διατάξεις με πολύ υψηλό κόστος. Επιπλέον αφού η μέτρηση γίνεται από απόσταση παρουσιάζονται πολύ λιγότερες φθορές από τις αντίστοιχες μηχανικές διατάξεις.
Μη καταστροφική μέθοδος	Η μέτρηση με αναγνώριση εικόνας είναι μη καταστροφική. Σε αρκετές περιπτώσεις οι μηχανικές διατάξεις μέτρησης καταστρέφουν το αντικείμενο το οποίο μετράνε και γι' αυτό χρησιμοποιούνται τεχνικές τυχαίας δειγματοληψίας, με συνέπεια την μείωση της απόδοσης της γραμμής παραγωγής αφού απόρριψη ελαττωματικών προϊόντων γίνεται σε παρτίδες και την μείωση της ποιότητας του τελικού προϊόντος. Με την βοήθεια της επεξεργασίας εικόνας μπορούμε να κάνουμε μετρήσεις στο σύνολο των παραγόμενων προϊόντων εξασφαλίζοντας αυξημένη ποιότητα και απόρριψη μεμονωμένων ελαττωματικών προϊόντων χωρίς την ανάγκη ανθρώπινης επιτήρησης.
Ταχύτητα	Ένα σύστημα αναγνώρισης εικόνας μπορεί να παρουσιάζει αυξημένες επιδόσεις ταχύτητας λόγω της μέτρησης με βάση τα οπτικά χαρακτηριστικά η οποία γίνεται ακαριαία. Για παράδειγμα με μία κάμερα υπερύθρων μπορούμε να μετράμε τη θερμοκρασία ακαριαία με πολύ καλή ακρίβεια σε αντίθεση με κάποια διάταξη που απαιτεί επαφή κατά την οποία πρέπει να περιμένουμε να επέλθει θερμική ισορροπία. Ένα άλλο πλεονέκτημα είναι ότι δεν χρειάζεται να διακόπτεται η παραγωγή

	για να γίνεται μέτρηση. Τα αντικείμενα μπορούν να μετακινούνται και με τη βοήθεια αλγορίθμων ανακατασκευής εικόνας να έχουμε μετρήσεις που προσεγγίζουν με μεγάλη ακρίβεια μετρήσεις πάνω σε ακίνητα αντικείμενα. Επιπλέον ένα καλά σχεδιασμένο σύστημα αναγνώρισης εικόνας μπορεί να παίρνει αποφάσεις πολύ πιο γρήγορα από έναν άνθρωπο για απλά προβλήματα.
Δύο διστάσεις	Η μέτρηση που παίρνουμε με συστήματα αναγνώρισης εικόνας είναι δισδιάστατη. Οι περισσότερες μετρητικές συσκευές μπορούν να μετράνε σε μεμονωμένα σημεία. Αυτό όμως περιορίζει την δυνατότητα λήψης σωστών αποφάσεων και συνεπώς την αξιοπιστία της διάταξης. Για το παράδειγμα που αναφέραμε παραπάνω, τη μέτρηση της θερμοκρασίας, είναι προφανές ότι ένα θερμόμετρο μετράει τη θερμοκρασία σε ένα σημείο. Συνεπώς αν ένα αντικείμενο έχει στο σημείο μέτρησης σωστή θερμοκρασία ενώ σε ένα άλλο όχι, π.χ. λόγω κάποιας ρωγμής, αυτό δεν θα απορριφθεί όπως θα γινόταν με ένα σύστημα επεξεργασίας εικόνας.
Εύκολη προσαρμογή	Όπως είπαμε τα «εξαρτήματα» ενός συστήματος επεξεργασίας εικόνας, είναι μία κάμερα και ένα επεξεργαστικό σύστημα. Προφανώς λοιπόν η προσαρμογή ενός συστήματος αναγνώρισης εικόνας σε νέες απαιτήσεις μίας γραμμής παραγωγής περιλαμβάνει κατά κύριο λόγο εύκολες και οικονομικές αναβαθμίσεις στο πρόγραμμα και όχι αντικατάσταση εξοπλισμού. Επιπλέον λόγω της γενικότητας μερικών μεθόδων επεξεργασίας εικόνας είναι δυνατή η επαναχρησιμοποίηση κώδικα σε μεγάλο βαθμό.
Αυξημένες δυνατότητες	Όπως έγινε προφανές από τα παραπάνω, οι τεχνικές επεξεργασίας εικόνας «απελευθερώνουν τη φαντασία» και κάνουν δυνατές μετρήσεις που δεν είναι δυνατές με κλασικού τύπου μετρητικές διατάξεις. Στο δικό μας πρόβλημα, για παράδειγμα, καταφέραμε να μετρήσουμε τη γωνία σε πολλούς MDTs ταυτόχρονα και σε τυχαία θέση. Αυτό είναι εξαιρετικά δύσκολο να επιτευχθεί με οποιαδήποτε μηχανική διάταξη η οποία θα πολύ μεγαλύτερο κόστος. Ότι μπορεί να δει και να μετρήσει ο άνθρωπος με τα μάτια του μπορεί, με διάφορους βαθμούς δυσκολίας, να τα μετρήσει και ένα σύστημα αναγνώρισης εικόνας. Αυτό θα γίνει φανερό στα επόμενα χρόνια καθώς η έρευνα στον τομέα αυτό θα εξελίσσεται και θα δίνει προσιτές λύσεις σε ακόμα περισσότερα προβλήματα.

7.3.3 Υλοποίηση σε DSP

Την εφαρμογή μας την προτυποποιήσαμε σε ένα PC και στη συνέχεια την υλοποιήσαμε σε μία πλατφόρμα DSP. Με αυτό τον τρόπο αποδείξαμε ότι τα DSPs μπορούν να λύσουν αποτελεσματικά προβλήματα επεξεργασίας video και αναγνώρισης εικόνας.



Εικόνα 104. Τα DSPs λύνουν πραγματικά προβλήματα αναγνώρισης εικόνας

Αυτό είναι κάτι αρκετά «νέο» μιας και στις περισσότερες περιπτώσεις τα DSPs χρησιμοποιούνται για επεξεργασία εικόνας και ήχου ή εφαρμογές τηλεπικοινωνιών κάτι το οποίο καταδεικνύεται και από την έλλειψη αναπτυξιακών με δυνατότητες video. Αυτός ο περιορισμός των εφαρμογών των DSPs σε επεξεργασία απλών σημάτων (ήχος – ακίνητη εικόνα) ήταν κάτι αναπόφευκτο λόγω των σχετικά χαμηλών ταχυτήτων επεξεργασίας που είχαν στο παρελθόν. Η ραγδαία όμως ανάπτυξή τους ανοίγει δρόμους για νέες πιο πλήρεις και προχωρημένες εφαρμογές.

Κατά την ανάπτυξη της εφαρμογής μας εντυπωσίασε η υψηλή ταχύτητα επεξεργασίας του ADSP-BF533. Λόγω της εμπειρίας

σε κλασικούς επεξεργαστές 8-bit οι επιδόσεις αυτού του συστήματος μας εξέπληξαν. Η αίσθηση που έχει κανείς προγραμματίζοντας σε αυτόν τον επεξεργαστή είναι σαν να προγραμματίζει σε ένα πολύ γρήγορο PC το οποίο τρέχει μόνο την εφαρμογή σου. Αυτή η αίσθηση είναι πάρα πολύ σημαντική γιατί σου δίνει τη δυνατότητα να πειραματίζεσαι με ευκολία χωρίς να πρέπει να έχεις συνεχώς στο μυαλό σου θέματα ταχύτητας όπως συμβαίνει σε λιγότερο δυνατούς μικροεπεξεργαστές. Με αυτό τον τρόπο η ανάπτυξη των εφαρμογών γίνεται πιο γρήγορα και δομημένα, οδηγούμαστε γρήγορα σε ένα πρότυπο που δουλεύει σωστά, κάτι που είναι πάρα πολύ χρήσιμο από επιχειρηματικής απόψεως και στη συνέχεια έχουμε τη δυνατότητα να βελτιώσουμε τη ταχύτητα της εφαρμογής, αν αυτό είναι απαραίτητο ώστε να χρησιμοποιήσουμε χαμηλότερες συχνότητες ρολογιού για μικρότερη κατανάλωση ρεύματος και λιγότερο δυνατούς επεξεργαστές για μείωση του κόστους.

Σε συνδυασμό με τα παραπάνω πρέπει να ξανατονίσουμε τη σημασία του να έχεις ένα αυτόνομο υπολογιστικό σύστημα αντί για ένα σύστημα που «τρέχει» πολλά πράγματα όπως συμβαίνει συνήθως σε ένα PC. Το σύστημά μας έχει πολύ μεγαλύτερη αξιοπιστία μιας και το πρόγραμμα που εκτελείται είναι μόνο ένα και ελεγμένο πλήρως από εμάς. Ούτε κάποιο λειτουργικό ούτε άλλες εφαρμογές θα διαταράξουν την λειτουργία του και την επεξεργασία σε πραγματικό χρόνο. Επιπλέον λόγω της διαφορετικής «νοοτροπίας» προγραμματισμού σε DSPs που όπως είπαμε βασίζεται στη «ροή δεδομένων» και όχι σε «εσωτερικές καταστάσεις», έχουμε πολύ μεγαλύτερη αυτοπεποίθηση, ότι αν δουλέψει μία φορά το σύστημα, θα δουλεύει για πάντα. Δεν υπάρχουν κρυμμένες, απρόβλεπτες καταστάσεις που να «κολλήσουν το σύστημα» ούτε υψηλή πολυπλοκότητα που να κάνει την εύρεση λαθών δύσκολη. Τα προγράμματα σε DSPs αποτελούνται από πολλές μικρές ρουτίνες που εκτελούνται επαναλαμβανόμενα σε κάθε «πακέτο» δεδομένων. Με αυτόν τον τρόπο η αξιοπιστία ενός συστήματος DSP είναι πολύ μεγαλύτερη από ένα άλλο σύνθετο υπολογιστικό σύστημα.

Κλείνοντας αυτή τη διπλωματική εργασία, θα ήθελα να τονίσω ότι τα DSPs έχουν φτάσει σε ένα τέτοιο επίπεδο «ωριμότητας» που είναι πλέον προσιτά για την καλύτερη επίλυση ακόμα και απλών προβλημάτων για τα οποία στο παρελθόν δεν θα συνέφερε η χρήση μίας τόσο προχωρημένης τεχνολογίας. Μέχρι πρόσφατα τα DSPs χρησιμοποιούντουσαν μόνο σε εμπορικές εφαρμογές μεγάλων ποσοτήτων και σχετικά απλών απαιτήσεων για την μείωση του συνολικού κόστους. Η αύξηση των υπολογιστικών δυνατοτήτων των DSPs με σταθερό κόστος, η βελτίωση των εργαλείων ανάπτυξης και η διεύρυνση της βιβλιογραφίας κάνουν πλέον την τεχνολογία των DSPs προσιτή σε κάθε επιστήμονα για την λύση τόσο απλών όσο και σύνθετων προβλημάτων.

Δημήτρης Κουζής – Λουκάς

2004

Βιβλιογραφία – αναφορές

Βιβλιογραφία για επιμέρους αντικείμενα μπορεί κανείς να βρει στα αντίστοιχα σημεία της διπλωματικής ως υπότιτλους (footnotes). Συγκεντρωτικά παρουσιάζονται παρακάτω.

1. The Radon Transform and Some of Its Applications, Stanley R. Deans, John Wiley & Sons, 1983
2. The general quadratic Radon transform, Koen Denecker, Jeroen Van Overloop and Frank Sommen, 1998
3. P.V.C. Hough. A Method and Means for Recognizing Complex Patterns. US Patent: 3,069,654, Dec. 1962
4. http://www2.lut.fi/~kyrki/ipcv02/hough/exercise/exercise.html
5. http://eivind.imm.dtu.dk/staff/ptoft/Radon/Radon.html
6. The Radon Transform, Theory and Implementation, Peter Toft, Ph.D. Thesis, 1996
7. Use of the 3D Radon transform to examine the properties of oceanic Rossby waves, Peter G Challenor, Paolo Cipollini and David Cromwell, Journal of Atmospheric and Oceanic Technology, November 2000
8. Σελίδα 53, A Study of the Indian Ocean Circulation using Satellite Observations and Model Simulations, Dr. Bulusu Subrahmanyam, Ph.D Thesis, 2004
9. Cleaning Sky Survey Databases using Hough Transform and Renewal String Approaches, A. J. Storkey, N. C. Hambly, C. K. I. Williams, R. G. Mann, September 2003
10. Automatic Detection of Circular Objects by Ellipse Growing, Kenichi KANATANI, Naoya OHTA, November 2001
11. gtresearchnews.gatech.edu/newsrelease/bakery.htm (bakery.pdf)
12. Digital Image Processing, William K. Pratt, John Wiley & Sons, 2001
13. Video Demystified, Keith Jack, LLH Technology Publishing, 2001
14. Digital Signal Processing Handbook, Vijay K. Madisetti, Douglas B. Williams, CRC Press, 1999
15. Intelligent Image Processing, Steve Mann, University of Toronto, 2002, John Wiley & Sons
16. EE-203, Interfacing the ADSP-BF535/ADSP-BF533 Blackfin® Processor to NTSC/PAL video decoder over the asynchronous port, Thorsten Lorenzen, 2003
17. Table 6.1, p. 93. Video Demystified, Keith Jack, LLH Technology Publishing, 2001
18. Σελίδα 11-18, ADSP-BF533 Blackfin™ Processor Hardware Reference
19. http://www.owlnet.rice.edu/~elec539/Projects97/morphjrks/moredge.html
20. Geodesic Active Regions and Level Set Methods: Contributions and applications in Artificial Vision, Nikos K. Paragios, Ph.D. Thesis, 2000
21. Pattern Classification, Richard Duda, Peter Hart, David Stork, John Wiley and Sons, 2001
22. Στατιστική αναγνώριση προτύπων, Θ. Αλεξόπουλος, Α. Τζαμαριουδάκη, 2003
23. Computer Graphics A programming approach Second Edition by Steven Harrington McGRAW-HILL 1988
24. http://mathworld.wolfram.com/Point-LineDistance2-Dimensional.html
25. http://www.winthrop.edu/news/releases/archivereleases/spring2000/bresenham.htm
26. http://www.udayton.edu/~cps/cps560/notes/Lines/bresnham.htm
27. http://www.cs.wpi.edu/~matt/courses/cs563/talks/history.html
28. http://encyclopedia.thefreedictionary.com/Bresenham%27s%20line%20algorithm
29. http://www.ece.mcmaster.ca/~xwu/
30. http://www.cs.helsinki.fi/group/goa/mallinnus/lines/bresenh.html
31. http://encyclopedia.thefreedictionary.com/Bresenham%27s%20line%20algorithm%20C%20code
32. An efficient antialiasing technique, Xiaolin Wu, July 1991, Computer Graphics, Volume 25, Number 4
33. A Linear Algorithm for Incremental Digital Display of Circular Arcs, Jack Bresenham, 1977, Graphics and Image Processing
34. http://www.cse.iitk.ac.in/~amit/courses/768/98/dash/#hough
35. http://biometrics.cse.msu.edu/

36. http://www.identix.com/
37. http://www.informatics.bangor.ac.uk/~kuncheva/activities/aprmi2004.htm
38. http://www.wordiq.com/definition/Machine_vision
39. Special Purpose Digital Processors (DSP) (F. Mayer-Lindenberg, TUHH) dsp.pdf
40. (http://www.bdti.com) (Ole Wolf, wolf@bdti.com)
41. http://cs-alb-pc3.massey.ac.nz/notes/59304/l12.html
42. http://www.webopedia.com/TERM/P/pipelining.html
43. http://www.x86.org/articles/branch/branchprediction.htm
44. http://home.earthlink.net/~yatescr/papers.htm
45. http://stevehollasch.com/cgindex/coding/ieeefloat.html
46. http://dspvillage.ti.com/docs/catalog/dspplatform/details.jhtml?templateId=5121&path=templatedata/cm/dspdetail/data/vil_getstd_whatis
47. Special Purpose Digital Processors (DSP) F. Mayer-Lindenberg, TUHH (dsp.pdf)
48. http://www.xilinx.com/products/design_resources/dsp_central/info/fpga_overview.htm#comparison
49. www.opencores.org/forums.cgi/cores/2003/09/00043
50. Dr. Tobias Ellermeyer - Reducing NRE and Turnaround Time with Structured ASICs (http://www.micram-me.com/services/asic.htm)
51. http://www.arraydesign.com/design/
52. http://www.eet.com/article/showArticle.jhtml?articleId=21400223
53. http://www.manufacturing.net/pur/article/CA416351.html?stt=001&pubdate=05%2F20%2F04
54. http://www.eetimes.com/story/OEG20010604S0057
55. ADSP-BF533 Blackfin™ Processor Hardware Reference
56. ΕΛΕΚΤΟΡ, Απρίλιος 2004
57. Σελίδα 2-16, ADSP-BF533 Blackfin Processor Hardware Reference
58. Σελίδα 10-24, ADSP-BF53x Blackfin Processor Instruction Set Reference
59. Webopedia: http://www.webopedia.com/TERM/R/RISC.html
60. Σελίδα 1-3, ADSP-BF533 Blackfin Processor Hardware Reference
61. http://www.pattosoft.com.au/jason/Papers/ValueRangeProp/
62. http://www.cs.tcd.ie/Michael.Brady/ISD/MSc%20System%20Design%201.ppt
63. Σελίδα 17-1, ADSP-BF533 Blackfin Processor Hardware Reference
64. Table 4-11. Events That Cause Exceptions. BF533 Hardware manual.
65. Σελίδα 8-13, ADSP-BF533 Blackfin Processor Hardware Reference
66. Σελίδα 8-4: Table 8-1, 8-2, 8-3, ADSP-BF533 Blackfin Processor Hardware Reference
67. Σελίδα 15-16, ADSP-BF533 Blackfin™ Processor Hardware Reference
68. Anomaly [05000092], Σελίδα 4, Blackfin® ADSP-BF533 Anomaly list
69. Σελίδα 8-28, ADSP-BF533 Blackfin™ Processor Hardware Reference
70. Σελίδα 21, Blackfin® Embedded Processor Datasheet
71. Anomaly [05000066], [05000092], Σελίδα 2 και 4, Blackfin® ADSP-BF533 Anomaly list
72. Table 4-10, Σελίδα 4-41, ADSP-BF533 Blackfin™ Processor Hardware Reference
73. Σελίδα 3-17, ADSP-BF533 Blackfin™ Processor Hardware Reference
74. ADSP-BF533 EZ-KIT Lite Evaluation System Manual (p. viii)
75. ADSP-BF533 EZ-KIT Lite Evaluation System Manual (p. 2-8)
76. Σελίδα 21, ADV7171 manual, Analog Devices, 2001
77. Σελίδα 18, ADV7183 manual, Analog Devices, 2002
78. Σελίδα 42, πίνακας “PAL B, D, G, H, I”, ADV7171 manual, Analog Devices, 2001
79. www.uclinux.org
80. uClinux as an Embedded OS on an Embedded Processor, Analog Devices, 2004
81. MATLAB Application Program Interface Reference, V.6, ©COPYRIGHT 1984 - 2001 The MathWorks, Inc.
82. MATLAB External Interfaces, V. 6, © COPYRIGHT 1984 - 2001 by The MathWorks, Inc.
83. Πίνακας 4-11, σελίδα 4-43, ADSP-BF533 Blackfin™ Processor Hardware Reference
84. EE-239, Running Programs from Flash on ADSP-BF533 Blackfin® Processors, Steve K, 2004
85. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT6/node2.html