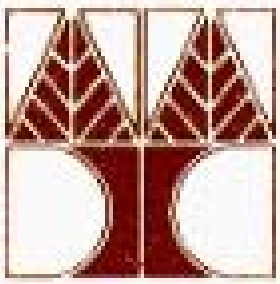
**Ατομική Διπλωματική Εργασία**

# ΕΠΕΞΕΡΓΑΣΙΑ ΒΙΝΤΕΟ ΓΙΑ ΑΞΙΟΛΟΓΗΣΗ ΕΓΚΕΦΑΛΙΚΩΝ ΕΠΕΙΣΟΔΙΩΝ

**Μάριος Παναγίδης**

# ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ



## ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

**Ιούνιος 2005**

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ**
**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

# ΕΠΕΞΕΡΓΑΣΙΑ ΒΙΝΤΕΟ ΓΙΑ ΑΞΙΟΛΟΓΗΣΗ ΕΓΚΕΦΑΛΙΚΩΝ ΕΠΕΙΣΟΔΙΩΝ

**Μάριος Παναγίδης**

Επιβλέπων Καθηγητής
Κωνσταντίνος Σ. Παττίχης

Η Ατομική αυτή Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Ιούνιος 2005

# Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέπων καθηγητή Δρ. Κωνσταντίνο Σ. Παττίχη, αναπληρωτή καθηγητή στο Τμήμα Πληροφορικής του Πανεπιστημίου Κύπρου για την καθοδήγηση που μου πρόσφερε για την διεκπεραίωση της διπλωματικής μου εργασίας.

Επίσης, θα ήθελα να ευχαριστήσω τους Δρ. Μάριο Σ. Παττίχη, επίκουρο καθηγητή στο Image and video Processing and Communications Lab του τμήματος Μηχανικής και Μηχανικής Υπολογιστών του Πανεπιστημίου του Νέου Μεξικό και το Δρ. Ευθύβουλο Κυριάκου επισκέπτη λέκτορα του τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου για την καθοδήγηση και πληροφορίες που μου έδωσαν κατά την διάρκεια της εργασίας αυτής. Θέλω επίσης να ευχαριστήσω θερμά τον καθηγητή Andrew Nikolaides για την ιατρική καθοδήγηση σχετικά με το θέμα που μελετήσαμε.

Τέλος, θα ήθελα να ευχαριστήσω το προσωπικό του Ινστιτούτου Νευρολογίας και Γενετικής Κύπρου και ιδιαίτερα τη Νίκη Γεωργίου για την προθυμία και διευκολύνσεις που μου πρόσφεραν καθόλη τη διάρκεια της διπλωματικής μου εργασίας.

# Περίληψη

Στην μελέτη αυτή παρουσιάζουμε την διαδικασία που ακολουθήσαμε για την ψηφιοποίηση αναλογικού βίντεο υπερηχογραφήματος της καροτίδας, καθώς επίσης και την δημιουργία ενός συστήματος για την επεξεργασία βίντεο για τον υπολογισμό οπτικής ροής των περιστατικών αυτών.

Είχαμε διάθεσή μας, μια μεγάλη βιβλιοθήκη από αναλογικά βίντεο αποθηκευμένα σε κασέτες S-VHS, τα οποία ψηφιοποιήσαμε. Για την διαδικασία της ψηφιοποίησης επιλέχθηκε η καταλληλότερη κάρτα ανάκτησης εικόνων βίντεο (Acer Media) καθώς επίσης ο αριθμός των πλασίων ανά δευτερόλεπτο (25fps), με ανάλυση 720*576, για επιλογή 8bit εικόνων. Δόθηκε ιδιαίτερη έμφαση στην διαδικασία της ψηφιοποίησης για τον λόγο ότι θα γίνει μόνο μια φορά και η ποιότητα της επεξεργασίας θα βασιστεί στην ποιότητα πληροφορίας που προσφέρει η ψηφιοποίηση. Ψηφοποιήθηκαν συνολικά 120 βίντεο, με μέσο χρόνο διάρκειας 10 δευτερόλεπτα, συνολικής χωρητικότητας 35GB.

Για την ανάλυση των βίντεο χρησιμοποιούμε την μέθοδο υπολογισμού οπτικής ροής, υλοποιημένη από τους Horn και Schunck και τροποποιημένη από τον Barron [2]. Για τον έλεγχο ορθότητας του αλγορίθμου και της υλοποίησης που είχαμε στην διάθεσή μας, χρησιμοποιήσαμε τις ακολουθίες εικόνων (φαινομενική κίνηση με αντικείμενο ένα δένδρο, αυτοκίνητο, τετράγωνο) και παραμέτρους spatial gradient για χ και y, που προτείνονται στην δημοσίευση [2]. Τα αποτελέσματα της προτεινόμενης υλοποίησης, ήταν σύμφωνα με την δημοσίευση [2]. Ακολούθως, εφαρμόζουμε τον αλγόριθμο σε ακολουθίες εικόνων από βίντεο αθηρωσκλήρωσης με τεχνητή κίνηση. Υπήρξε συμφωνία μεταξύ της αναμενώμενης και υπολογισθείσας κίνησης στις κατευθύνσεις x και y.

Ακολούθως ο αλγόριθμος οπτικής ροής εφαρμόστηκε σε πραγματικά βίντεο υπερηχογραφήματος καρωτιδικής πλάκας. Τα αποτελέσματα είχαν επιβεβαιωθεί με την βοήθεια ειδικού γιατρού.

Η μελέτη τελειώνει με τα συμπεράσματα και εισηγήσεις για μελλοντική εργασία.

# ΠΕΡΙΕΧΟΜΕΝΑ

# Κεφάλαιο 1

## Ανάλυση του Προβλήματος



### 1.1 Γενικά

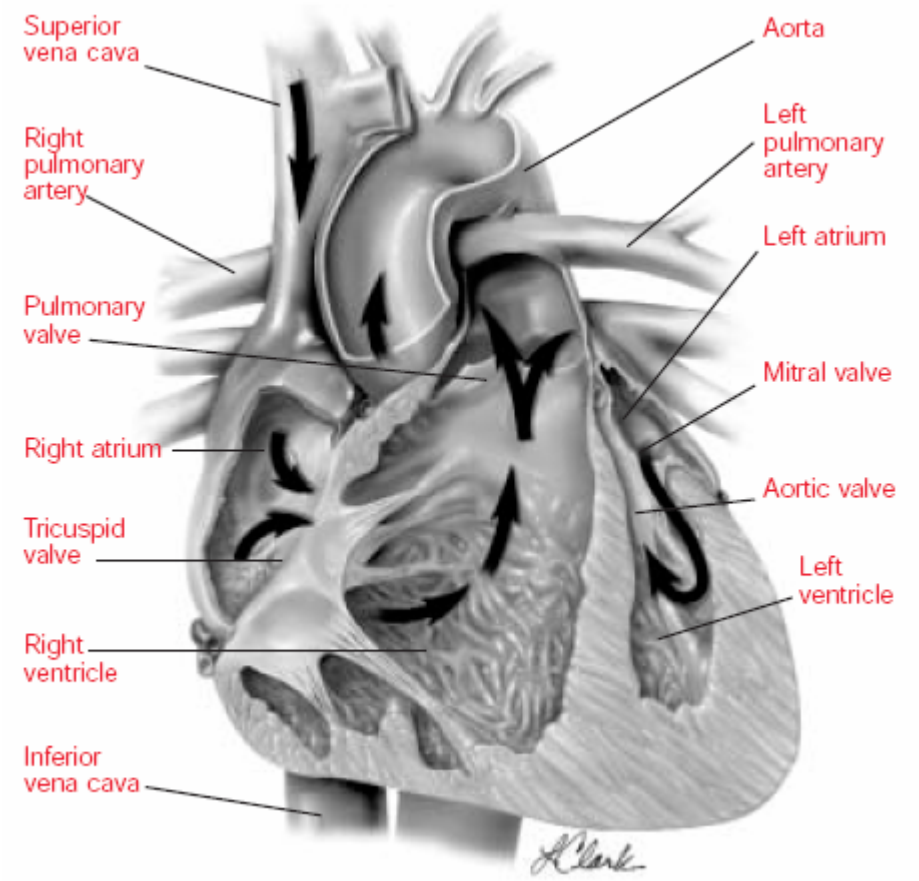
Η επεξεργασία βίντεο μπορεί να δώσει σημαντικές πληροφορίες όσο αφορά την κίνηση πλάκας ασθενών αθηροσκλήρωσης παρέχοντας έτσι στο ιατρικό προσωπικό ένα αξιόλογο εργαλείο για την πρόβλεψη εγκεφαλικών επεισοδίων. Παρακάτω γίνεται ανάλυση βασικών παραμέτρων του προβλήματος τόσο από την ιατρική σκοπιά όσο και από την σκοπιά της επεξεργασία βίντεο.

Οι πληροφορίες που παρουσιάζονται είναι βάση της βιβλιογραφίας [1].

### 1.1.1 Τι είναι η καρδία

Μια συνηθισμένη ανθρώπινη καρδία είναι μια δυνατή, μυική αντλία λίγο μεγαλύτερη από μια γροθιά. Κάθε μέρα, κατά μέσο όρο η ανθρώπινη καρδιά «κτυπά» (συσπάσεις) 100,000 φορές και αντλέι περίπου 2,000 γαλόνια αίματος. Η καρδιά αντλεί αίμα συνεχώς στο κυκλοφοριακό σύστημα. Το κυκλοφοριακό

σύστημα είναι το δίκτυο από ελαστικούς σωλήνεςπου μεταφέρουν το αίμα σε όλο το σώμα. Περιλαμβάνει την καρδιά, τους πνεύμονες, αρτηρίες, arterioles(small arteries) και capillaries(very tiny blood vessels). Αυτά τα αγγεία μεταφέρουν οξυγονομένο – θρεπτικό- πλούσιο αίμα σε όλα τα μέρη του σώματος. Το κυκλοφοριακό σύστημα επίσης περιλαμβάνει αιμοφόρα αγγεία(small veins) και φλέβες. Αυτά είναι τα αγγεία που μεταφέρουν το αίμα αφού χρησιμοποιήθηκαν οι θρεπτικές ουσίες και το οξυγόνο.




**Σχ 1.1** Διατομή καρδίας [1]

### 1.1.2 Αθηροσκλήρωση: Σημαντικός παράγοντας για καρδιαγγειακές Ασθένειες

Η αθηροσκληρωση είναι μια αργή, πολύπλοκη ασθένεια που αρχίζει από την παιδική ηλικία και συχνά εξελίσσεται όταν το άτομο εξελιχτεί σε ενήλικα. Σε ορισμένα άτομα εξελίσσεται πολύ γρήγορα, ακόμα και στην τρίτη δεκαετία. Πολλοί επιστήμονες

πιστεύουν ότι αρχίζει με ζημιά στο βαθύτερο επίπεδο της αρτηρίας. Αυτό το επίπεδο λέγεται ενδοθήλιο [1].

Οι τρεις αποδεδειγμένες αιτίες του αρτηριακού τοιχώματος είναι:

1) ψηλά επίπεδα χοληστερόλης και τριγλυκεριδίων στο αίμα
2) ψηλή πίεση
3) το κάπνισμα

Ειδικότερα το κάπνισμα, χειροτερεύει την αθηροσκλήρωση και επιταχύνει την ανάπτυξη στις στεφανιαίες αρτηρίες, την αορτή και αρτηρίες των ποδιών. (Οι στεφανιαίες αρτηρίες φέρνουν αίμα στο μυ της καρδίας, η αορτή είναι το μεγάλο αγγείο μέσω του οποίου η καρδιά αντλεί το αίμα που πηγαίνει στο σώμα).

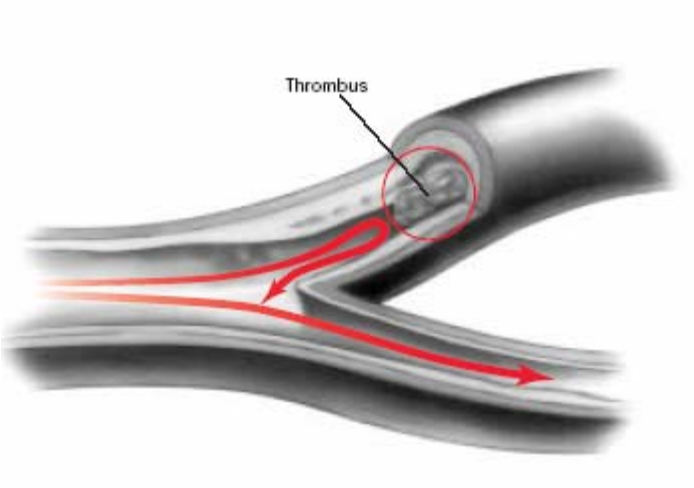
**Σχήμα 1.2**. Στην αθηροσκλήρωση, το μέγεθος της πλάκας που δημιουργείται στις αρτηρίες με το πέρασμα του χρόνου μπορεί να φτάσει στο σημείο που να μειώσει αρκετά την ροή του αίματος[1].

Λόγω της ζημιάς του ενδοθήλιου, του λίπους, της χοληστερόλης, κυτταρικά απολύματα, ασβέστιο και άλλων ουσιών, κατακάθονται στα τοιχώματα της αρτηρίας. Όλα αυτά προκαλούν ένα ερεθισμό στα κύτταρα του αρτηριακού τοιχώματος, και παράγουν άλλες ουσίες που μεγαλώνουν περισσότερο την κατασκευή αυτών των κυττάρων. Αυτά τα κύτταρα και το άλλο υλικό (πλάκες) μπορούν να γίνουν αρκετά μεγάλα ώστε να χοντρήνουν το ενδοθήλιο.

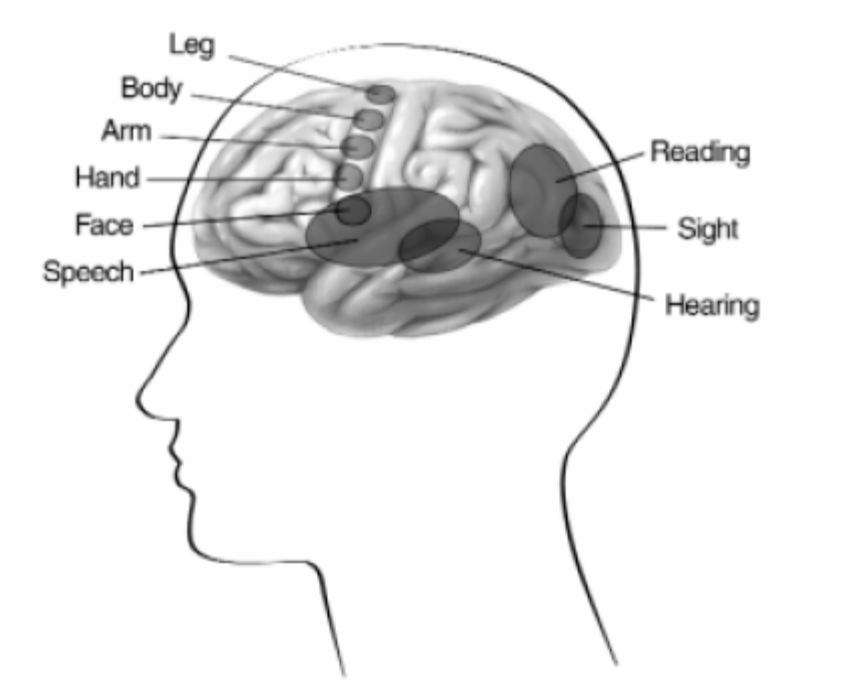
### 1.1.3 Εγκεφαλικό

Το εγκεφαλικό είναι μια καρδιαγγειακή πάθηση. Επηρεάζει τα αγγεία του αίματος που προμηθεύουν με αίμα τον εγκέφαλο.

Το εγκεφαλικό σημβαίνει όταν ένα αγγείο του αίματος που φέρνει οξυγόνο και θρεπτικά συστατικά στον εγκέφαλο, σπάζει ή εμποδίζεται από ένα θρόμβο αίματος ή κάποιο άλλο σώμα. Λόγω αυτής της ρήξης ή εμποδίου, μέρος του εγκέφαλου δεν παίρνει το αίμα και οξυγόνο που χρειάζεται. Χωρίς οξυγόνο, τα νευρικά κύτταρα δεν μπορούν να δουλέψουν και πεθαίνουν μέσα σε λίγα λεπτά. Όταν τα νευρικά κύτταρα δεν δουλεύουν, το μέρος του σώματος που ελέγχουν δεν δουλεύει. Τα καταστρεπτικά αποτελέσματα ενός εγκεφαλικού είναι συχνά μόνιμα, γιατί τα νεκρά κύτταρα του εγκεφάλου δεν μπορούν να αντικατασταθούν.




**Σχήμα.1.3** Όταν μια αρτηρία του εγκεφάλου δεν μπορεί να στείλει αίμα, τότε συμβαίνει το εγκεφαλικό [1].




**Σχ. 1.4** Διαφορετικά τμήματα του εγκεφάλου ελέγχουν διαφορετικές λειτουργίες [1].

### 1.1.4 Διάγνωση εγκεφαλικού

Όταν κάποιος παρουσιάσει συμπτώματα για εγκεφαλικό, ο γιατρός πρέπει πρώτα να μαζέψει πληροφορίες για να κάνει την διάγνωση. Πρέπει πρώτα να μαζέψει το ιστορικό του ασθενή όσο αφορά το διαβήτη, υπέρταση, καρδειακές ασθένειες, ασθένεις των αγγείων του αίματος και άλλες νευρολογικές ασθένειες. Μετρά την πίεση και στα δυο χέρια, ελέγχει τον παλμό, την καρδιά, ακούει για θορύβους πάνω στον αυχένα, ελέγχει τα μάτια και κάνει νευρολογική εξέταση.

Ακολουθούν βασικές εξετάσεις, και εξετάσεις που ο κάθε γιατρός χρειάζεται να κάνει, έτσι ώστε σύμφωνα με την δική του αντίληψή να κάνει την διάγνωση.

### 1.1.5 Είδη εξετάσεων για διάγνωση εγκεφαλικών

Η αναγνώριση συμπτωμάτων είναι ένας τρόπος διάγνωσης εγκεφαλικών. Μια σωστή διάγνωση όμως, δεν σταματά μόνο εκεί. Γίνονται επιπρόσθετες εξετάσεις, λόγω του ότι τα συμπτώματα μπορεί να μην είναι απαραίτητα αποτέλεσμα εγκεφαλικού.

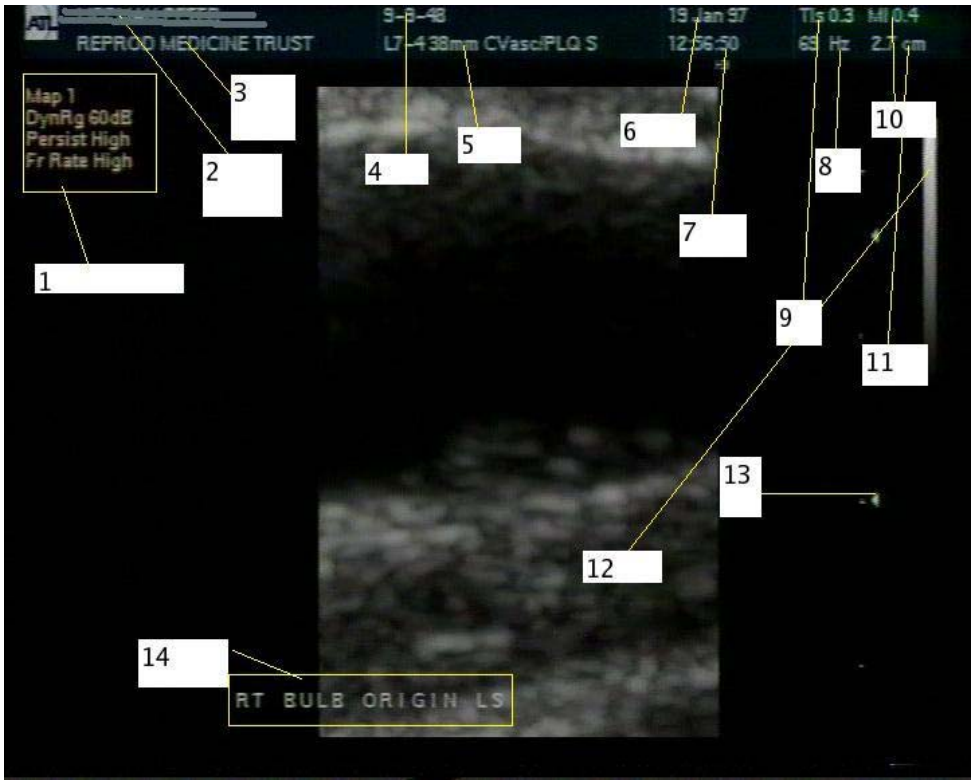
Αυτές οι εξετάσεις είναι ασφαλείς, ανώδυνες και μπορούν να γίνουν χωρίς να είναι απαραίτητο ο ασθενής να είναι στο νοσοκομείο.

Οι εξετάσεις χωρίζονται σε τρεις κατηγορίες:

- Εξετάσεις που χαρτογραφούν τον εγκέφαλο, και δημιουργούν εικόνες όπως τις συνηθισμένες ακτίνες Χ.
- Εξετάσεις που μετρούν την ηλεκτρική δραστηριότητα του εγκεφάλου και δίνουν χρήσιμες πληροφορίες στο πως λειτουργεί και να δείξουν που ακριβώς δεν υπάρχει ομαλή λειτουργία.
- Τέλος, εξετάσεις μέτρησης της ροής του αίματος, που μετρούν την ροή και ανιχνεύουν φραγές στα αγγεία του αίματος. Είναι χρήσιμες στο να δείχνουν περιοχές με σημαντική αθηροσκλήρωση στις αρτηρίες της καροτίδας. Ο γιατρός αποφασίζει με βάση την κάθε περίπτωση κατά πόσον αυτές οι εξετάσεις είναι χρήσιμες, και αν ναι, ποιες από αυτές να χρησιμοποιήσει.

### 1.1.6 Εξετάσεις που δείχνουν την ροή του αίματος

Η εξέταση υπερήχων Doppler, εξετάζει την χρήση ηχητικών κυμάτων ψηλών συχνοτήτων για να ανιχνεύσει εμπόδια στην αρτηρία της καρωτίδας. Ένα εργαλείο, το Doppler probe, που παράγει υπερήχους, τοποθετείται πολύ κοντά στην αρτηρία της καρωτίδας. Τα υπερηχητικά κύματα από το εργαλείο αυτό ταξιδεύουν μέσω του αυχένα και αναπηδούν πάνω στα κινούμενα κύτταρα του αίματος. Το ανακλώμενο ηχητικό κύμα, επιστρέφει στο εργαλείο με διαφορετική συχνότητα, και ανιχνεύεται από το εργαλείο αυτό. Η αλλαγή στη συχνότητα των ηχητικών κυμάτων σχετίζεται με την ταχύτητα των κυττάρων του αίματος και κατά συνέπεια την οπτική ροή.




**Σχήμα 1.5** Εικόνα από εξέταση υπερήχων

1. Πληροφορίες για τι είδους greyscale χρησιμοποποείται στην εικόνα. Επίσης, εκεί που παίρνουν τους υπερήχους δείχνουν τα frames πάνω δεξιά
2. Το όνομα του ασθενούς
3. Όνομα του Ινστιτούτου
4. Ημερομηνία γεννήσεως
5. Η κεφαλή σάρωσης που χρησιμοποιείται(scan head). Εδώ είναι το Linera 7-4 38mm
6. Ημερομηνία εξέτασης
7. Ώρα στο σύστημα του μηχανήματος
8. Συχνότητα: ρυθμός των πλαισίων, είναι ανάλογα με το εργαλείο (probe)

9. και 10. Είναι η ένδειξη θερμότητας και η ένδειξη μηχανικής. Δείχνει ανάλογα με το εργαλείο τι θερμοκρασία έχει το σώμα. Tis: soft tissue or Tip : Bone thermal index or Tic: για αρτηρίες μέσα στον εγκέφαλο.

Συνήθως είναι Tis kai Tip.

11. Το βάθος που εισχωρεί ένας υπερήχος, ανάλογα με το μηχάνημα. Για εξέταση της κοιλιακής χώρας είναι μέχρι 16cm. Για καρωτίδα είναι μέχρι τα 6-7 cm.

12. Ένδειξη για αποχρώσεις του γκρίζου. Αλλάζει η ένδειξη αυτή αναλόγως με το μάτι του χρήστη. Ο κάθε χρήστης κάνει τις δικές του ρυθμίσεις.
13. Τρίγωνο είναι για εστίαση (focal marker). Βοηθά τον χρήστη για να βρει την καλύτερη θέση. Πρέπει να είναι ακριβώς κάτω από την αρτηρία.
14. Το κείμενο που περιγράφει σε ποια αρτηρία βρισκόμαστε.

## 1.2 Σκοπός της μελέτης

Η μελέτη αυτή στοχεύει στην δημιουργία μιας βιβλιοθήκης από βίντεο υπερηχογραφήματος καρωτίδας σε ψηφιακή μορφή και την ανάπτυξη ενός συστήματος για την επεξεργασία και εξαγωγή της οπτικής ροής σχετικά με την κίνηση των αντικειμένων μέσα στα βίντεο.

Συγκεκριμένα, η διπλωματική αυτή εργασία θα ασχοληθεί με τα ακόλουθα θέματα:

1. Μελέτη του προβλήματος.
2. Επιλογή λογισμικού και υλικού για την ψηφιοποίηση αναλογικού βίντεο.
3. Εξαγωγή εικόνων από βίντεο και προετοιμασία δεδομένων για χρήση από την υλοποίηση του αλγορίθμου οπτικής ροής Horn and Schunck [2].
4. Περιγραφή αλγορίθμου Horn and Schunck.
5. Έλεγχος υλοποίησης αλγορίθμου, παραδείγματα χρήσης με τεχνητές εικόνες και αποτελέσματα με πραγματικά δεδομένα.
6. Εξαγωγή συμπερασμάτων.

## 1.3 Δομή της μελέτης

Στο παρόν κεφάλαιο γίνεται μια εισαγωγή στη μελέτη από την πλευρά των ιατρικών παραμέτρων του προβλήματος. Δίνεται επίσης παράδειγμα πλαισίου από βίντεο με πλάκα και γίνεται επεξήγηση των πληροφοριών που προσφέρονται.

Στο κεφάλαιο 2 δίνεται ο ορισμός και χαρακτηριστικά του βίντεο, των προτύπων βίντεο για τηλεοράσεις και τα πρότυπα αποθήκευσης. Παρουσιάζονται επίσης χαρακτηριστικά των σημάτων βίντεο με στόχο την καλύτερη κατανόηση την επιλογή μεθοδολογίας και υλικού για την ψηφιοποίηση του αναλογικού βίντεο.

Στο κεφάλαιο 3 παρουσιάζονται οι μεθοδολογίες μετατροπής αναλογικού βίντεο σε ψηφιακό. Γίνεται περεταίρω ανάλυση της χρωματικηής δειγματοληψίας 4:2:0, την οποία και χρησιμοποιούμε.Έπειτα, δίνονται παραδείγματα μετατροπής RGB σε YUV χρωματοχώρο. Ακολούθως αφού αναφερθούν οι απαιτήσεις για το σύστημα που θα υλοποιήσουμε, παρουσιάζονται συγκρίσεις από δοκιμές που κάναμε για την επιλογή λογισμικού και υλικού που χρησιμοποιήθηκαν κατά την ψηφιοποίηση.

Τέλος, γίνεται περιγραφή της μεθοδολογίας και των εντολών που είναι απαραίτητες για την παροχή βέλτιστων αποτελεσμάτων.

Στο κεφάλαιο 4 παρουσιάζεται η μεθοδολογία προετοιμασίας των δεδομένων για επεξεργασία. Παρουσιάζονται επίσης οι λειτουργίες χωρικής και χρονικής αποκοπής, μετατροπής εικόνων και τέλος παρουσιάζεται το γραφικό περιβάλλον του συστήματος επιλογής βίντεο και προετοιμασίας δεδομένων.

Στο κεφάλαιο 5, γίνεται εισαγωγή στην οπτική ροή και εκτιμησης κίνησης. Παρουσιάζονται επίσης προβλήματα που προκύπτουν από την οπτική ροή, όπως το πρόβλημα της εγγραφής, ανοίγματος και αντιστοίχησης. Ακολούθως παρουσιάζονται τα μοντέλα οπτικής ροής και εισαγωγή στον αλγόριθμο που χρησιμοποιήσαμε, τον αλγόριθμο υπολογισμού της οπτικής ροής, Horn and Schunck. Στο κεφάλαιο αυτό, γίνεται εισαγωγή στον υπολογισμό λάθους και περιγραφή του περιβάλλοντος εργασίας.

Στο κεφάλαιο 6, παρουσιάζεται ο αλγόριθμος του Horn and Schunck με περισσότερη λεπτομέρεια. Επίσης, γίνεται παράθεση των ελέγχων ορθότητας της υλοποίησης του αλγορίθμου με τεχνητά δεδομένα. Στην συνέχεια γίνεται παρουσίαση των αποτελεσμάτων με πραγματικά δεδομένα. Τέλος, παρουσιάζονται τα αποτελέσματα με διαφορετικές μεθοδολογίες περιορισμού του λάθους και θορύβου.

Στο κεφάλαιο 7, παρουσιάζονται τα συμπεράσματα από την μελέτη αυτή και ορισμένα συμπεράσματα που έχουν εξαχθεί. Δίδονται επίσης εισηγήσεις για μελλοντική έρευνα και μελέτη.

# Κεφάλαιο 2

## Βιντεο και χαρακτηριστικά



### 2.1 Εισαγωγή

Το κεφάλαιο αυτό έχει ως σκοπό να δώσει τον ορισμός και χαρακτηριστικά του βίντεο, των προτύπων βίντεο για τηλεοράσεις και τα πρότυπα αποθήκευσης. Αυτό είναι απαραίτητο για την κατανόηση των παραμέτρων που πρέπει να ληφθούν υπόψη κατά την ψηφιοποίηση.

Επίσης, παρουσιάζονται τα χαρακτηριστικά των σημάτων βίντεο με στόχο την καλύτερη κατανόηση των συνδεσμολογιών που δοκιμάζουμε στην ψηφιοποίηση.
Οι πληροφορίες που παρουσιάζονται είναι βάση της βιβλιογραφίας [3,5,6,8,9,10,11].

### 2.2 Τί είναι το βίντεο

Ένα σήμα βίντεο είναι είναι μια ακολουθία διδιάστατων εικόνων που προβάλλονται από μια τρισδιάστατη σκηνή στο επίπεδο της εικόνας μιας βιντεοκάμερας. Οι τιμές

χρώματος σε οποιοδήποτε σημείο σε ένα πλαίσιο του βίντεο αναπαριστά το ανακλώμενο ή παραγώμενο φως σε ένα συγκεκριμένο τρισδιάστατο σημείο στην σκηνή την οποία παρατηρούμε.

Το σήμα αυτό καταγράφει το ανακλώμενο ή παραγώμενη ένταση του φωτός από τα αντικείμενα σε μια σκηνή η οποία παρατηρείται από ένα σύστημα παρατήρησης. Γενικά, οι αλλαγές στην φωτεινότητα διαφέρουν στον χρόνο και στο χώρο. Σε κάθε σκηνή το σύστημα παρατήρησης που μπορεί να είναι και μια κάμερα, συλλαμβάνει μόνο τα μήκη κύματος για το οποία το σύστημα είναι ευαίσθητο και μπορεί να τα αναγνωρίσει.

## 2.3 Φως και χρώμα

Το φως είναι ένα ηλεκτρομαγνητικό κύμα, με μήκη κύματος από 380 έως 780 νανόμετρα, στα οποία το ανθρώπινο μάτι είναι ευαίσθητο. Μια φωτεινή πηγή μπορεί να εκπέμπει ενέργεια σε ένα φάσμα μήκων κύματος, και η ένταση μπορεί να μεταβάλλεται στο χρόνο και το χώρο. Υπάρχουν δύο ειδών φωτεινές πηγές, αυτόφωτες και ετερόφωτες. Οι αυτόφωτες παράγουν ένα ηλεκτρομαγνητικό κύμα και οι ετερόφωτες που ανακλούν ένα προσπίπτων κύμα. Το χρώμα που αντιλαμβανόμαστε εξαρτάτε από το μήκος κύματος της εκπεμπόμενης ενέργειας. Το φως που παράγεται από αυτόφωτες πηγές ακολουθεί ένα αθροιστικό κανόνα: το αντιλαμβανόμενο χρώμα ενός μείγματος αυτόφωτων πηγών, εξαρτάται από το άθροισμα του φάσματος όλων των φωτεινών πηγών. Για παράδειγμα ο συνδυασμός κόκκινου, πράσινου και μπλε σε σωστές αναλογίες παράγει το άσπρο χρώμα.

Οι ετερόφωτες φωτεινές πηγές είναι εκείνες που ανακλούν ένα προσπύπτων φως, το οποίο με τη σειρά του μπορεί να προέκυψε από ετερόφωτη πηγή. Το χρώμα του ανακλώμενου φωτός εξαρτάται στο φασματικό περιεχόμενο του προσκύπτοντως φωτός και το φάσμα του μήκους κύματος που απορροφάται. Το ανακλώμενο φως ακολουθεί τον αφαιρετικό κανόνα: το αντιλαμβανόμενο φως ενός μείγματος ετερόφωτων φωτεινών πηγών εξαρτάται από το υπόλοιπο, των μη απορροφούμενων μήκων κύματος. Για παράδειγμα, για το ανακλώμενο άσπρο, μια επιφάνεια που απορροφά μήκος

κύματος 700nm (κόκκινο) εμφανίζεται σαν κυανό. Άρα το κυανό είναι το συμπλήρωμα του ερυθρού.

## 2.4 Ανθρώπινη αντίληψη για το χρώμα

Υπάρχουν δυο παράγοντες που περιγράφουν την αίσθηση του χρώματος στον άνθρωπο: η φωτεινότητα (luminance) και οι χρωματικές συνιστώσες (chrominance). Η φωτεινότητα αναφέρεται στην αντιλαμβανόμενη φωτεινότητα του φωτός, που είναι ανάλογη του συνόλου της ολικής ενέργειας του ορατού φάσματος. Η φωτεινότητα, περιγράφει τον αντιλαμβανόμενο τόνο του φωτός, που εξαρτάται από τη σύνθεση του μήκους κύματος του φωτός. Οι χρωματικές συνιστώσες με την σειρά τους αναλύονται σε δυο παραμέτρους: τη χροιά και την καθαρότητα. Η χροιά ορίζει τον τόνο του χρώματος, που εξαρτάται από την κορυφή του μήκους κύματος του φωτός, η καθαρότητα περιγράφει πόσο αγνό είναι το χρώμα, το οποίο εξαρτάται από τη διάχυση ή το εύρος φάσματος του φάσματος του φωτός.

## 2.5 Η τριχρωματική θεωρία του φωτός

Η τριχρωματική θεωρία παρουσιάστηκε αρχικά από τον Maxwell το 1855. Ουσιαστικά υποστηρίζει ότι τα περισσότερα χρώματα μπορούν να παραχθούν από ανάμιξη πρωτεύοντων χρωμάτων. Έστω $C_k$, k=1, 2, 3, αναπαριστούν τα χρώματα των κύριων τριών χρωματικών πηγών, και C ένα δοθέν χρώμα. Τότε, αυτό που υποστηρίζει η θεωρία του Maxwell είναι ότι

$$C = \sum T_k C_k \qquad (2.1)$$

όπου το $T_k$ είναι οι ποσότητες από τα τρία κύρια χρώματα που απαιτούνται για να ταυτιστούν με το C.

Το πιο δημοφιλές σύνολο για αυτόφωτες φωτεινές πηγές περιέχουν το κόκκινο, πράσινο και μπλε, γνωστό και ως RGB (Red Green Blue). Το πιο κοινό πρωτεύον

σύνολο για ετερόφωτες φωτεινές πηγές περιέχει το κυανό, πορφυρό και κίτρινο, γνωστό και ως CMY. Μια ενδιαφέρον παρατήρηση είναι το γεγονός πως το RGM και το CMY είναι συμπληρωματικά.

## 2.6 Προσδιορισμός Χρώματος μέσω της φωτεινότητας και των χρωματικών συνιστωσών

Το RGB χρησιμοποιείται κυρίως για να αναπαραστήσει συνδυασμούς φωτεινότητας και χρωματικών συνιστωσών μιας φωτεινής πηγής. Σε πολλές εφαρμογές, θέλουμε να περιγράψουμε ένα χρώμα αναφορικά με την φωτεινότητα και τις χρωματικές συνιστώσες ξεχωριστά, για να δώσουμε την δυνατότητα πιο αποδοτικής επεξεργασίας και μεταφορά χρωματικού σήματος. Για τον λόγο αυτό έχουν αναπτυχθεί χρωματικές συντεταγμένες, στις οποίες μια συντεταγμένη αναπαριστά την φωτεινότητα και οι άλλες δυο αντίστοιχα χαρακτηρίζουν την χροιά και την καθαρότητα.. Ένα παράδεγμα χρωματικών συντεταγμένων είναι και το CIE XYZ, στο οποίο το Y άμεσα μετρά την ένταση της φωτεινότητας. Συγκριτικά με το RGB το CIE RGB δίνεται από [8]:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 2.365 & -0.515 & 0.005 \\ -0.897 & 1.426 & -0.014 \\ -0.468 & 0.089 & 1.009 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad (2.2)$$

Το βίντεο ουσιαστικά καταγράφει το εκπεμπόμενη ή/και την ανακλώμενη ένταση του φωτός, από τα αντικείμενα στην σκηνή τα οποία παρατηρούνται από ένα σύστημα παρατήρησης. Γενικά, η ένταση μεταβάλλεται τόσο ως προς το χρόνο όσο και στο χώρο.

## 2.7 Composite Vs Component Βίντεο

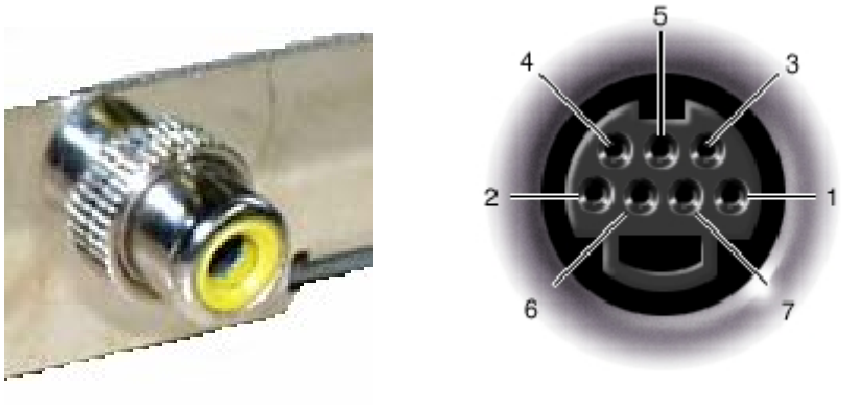
Στην ιδανική περίπτωση, ένα έγχρωμο βίντεο θα έπρεπε να ορίζεται από τρεις συναρτήσεις ή σήματα, κάθε ένα από αυτά να περιγράφει ένα συνιστών χρώμα, είτε στη τριχρωματική αναπαράσταση είτε στην αναπράσταση με φωτεινότητα και χρωματικές συνιστώσες. Ένα βίντεο με αυτή τη μορφή, είναι γνωστό σαν *component video*.

Υπάρχει επίσης και composite μορφή, στην οποία τρία χρωματικά σήματα πολυπλέκονται σε ένα μοναδικό σήμα.

Η δημιουργία ενός composite σήματος στην ιδιότητα ότι τα χρωματικά σήματα έχουν σημαντικά μικρότερο εύρος φάσματος από το συστατικό της φωτεινότητας. Ρυθμίζοντας κάθε χρωματική συνισταμένη σε μια συχνότητα που είναι πάνω από το άνω άκρο της συχνότητας της φωτεινότητας, και προσθέτοντας το αποτέλεσμα των σημάτων στο αρχικό σήμα της φωτεινότητας, δημιουργούμε έτσι ένα composite σήμα που περιέχει πληροφοίες για την φωτεινότητα και τις χρωματικές συνιστώσες.

Η μορφή composite χρησιμοποιείται για αποθήκευση ορισμένων αναλογικών κασέτων, όπως οι VHS. Είναι συμβατή με σήμα grey-scale, και επίσης απαλοίφει την ανάγκη για συγχρονισμό διάφορων χρωματικών συνιστωσών κατά την επεξεργασία έγχρωμου βίντεο. Το composite σήμα, έχει εύρος ζώνης κατά πολύ μικρότερο από το άθροισμα σημάτων component, άρα μπορεί να σταλεί και να αποθηκευτει πιο αποδοτικά. Αυτά τα πλεονεκτήματα μπορούν να επιτευχθούν εις βάρος της ποιότητας του βίντεο, με artifacts.

Συμβιβάζοντας ρυθμό μετάδοσης και ποιότητας του βίντεο ανακαλύφθηκε το S-video, το οποίο αποτελείται από δυο συνιστώσες: την φωτεινότητα και μια μόνο συνιστώσα chrominance, που είναι η πολύπλεξη των δυο αρχικών χρωματικών συνιστωσών.




**Σχήμα 2.1** Στα αριστερά, το composite input και δεξιά το s-video input [10]

Όπως έχουμε ήδη αναφέρει τα πρότυπα που αναλύσαμε παραπάνω ασχολούνται με την μετάδοση τηλεοπτικών σημάτων video. Για τα τηλεοπτικά σήματα υπάρχουν τα πρότυπα NTSC, PAL και SECAM.

## 2.8 Το πρότυπο NTSC

Το πρώτο τηλεοπτικό σύστημα μετάδοσης έγχρωμου video δημιουργήθηκε στις Η.Π.Α το 1953. Βασίστηκε στο πρότυπο NTSC (**N**ational **T**elevision **S**ystem **C**ommittee) και χρησιμοποιείται σήμερα σε πολλές πολιτείες της αμερικανικής ηπείρου αλλά και σε πολλές ασιατικές χώρες συμπεριλαμβανομένης και της Ιαπωνίας. Το πρότυπο προδιαγράφει 525 γραμμές ανά καρέ και τα βασικά του χαρακτηριστικά φαίνονται στον παρακάτω πίνακα.

| **ΣΥΣΤΗΜΑ** | **NTSC M** |
|---|---|
| Γραμμές ανά πεδίο (Lines / Field) | 525/60 |
| Οριζόντια Συχνότητα (Horizontal Frequency) | 15.734 kHz |
| Κατακόρυφη Συχνότητα (Vertical Frequency) | 60 Hz |
| Συχνότητα χρωματικού φέροντος (Color Sub carrier Frequency) | 3.579545 MHz |
| Εύρος Ζώνης Video (Video Bandwidth) | 4.2 MHz |
| Συχνότητα ηχητικού φέροντος (Sound Carrier) | 4.5 MHz |

**Πίνακας 2.1** Χαρακτηριστικά προτύπου NTSC [10]

## 2.9 Το πρότυπο PAL

Το πρότυπο PAL (Phase Alternating Line) εμφανίστηκε στην αρχή της δεκαετίας του 1960 και εφαρμόστηκε στις περισσότερες χώρες με εξαίρεση τη Γαλλία. Το πρότυπο χρησιμοποιεί ένα μεγαλύτερο εύρος ζώνης από το NTSC που του επιτρέπει να έχει καλύτερη ποιότητα εικόνας. Το PAL προδιαγράφει 625 γραμμές ανά καρέ και τα βασικά του χαρακτηριστικά φαίνονται στον πίνακα 2.2.

| **ΣΥΣΤΗΜΑ** | **PAL B,G,H** | **PAL I** | **PAL D** | **PAL N** | **PAL M** |
|---|---|---|---|---|---|
| Γραμμές ανά πεδίο (Lines/Field) | 625/50 | 625/50 | 625/50 | 625/50 | 525/60 |
| Οριζόντια Συχνότητα (Horizontal Frequency in kHz) | 15.625 | 15.625 | 15.625 | 15.625 | 15.750 |
| Κατακόρυφη Συχνότητα (Vertical Frequency in Hz) | 50 | 50 | 50 | 50 | 60 |
| Συχνότητα χρωματικού φέροντος (Color Sub carrier Frequency in MHz) | 4.433618 | 4.433618 | 4.433618 | 3.582056 | 3.575611 |
| Εύρος Ζώνης Video (Video Bandwidth in MHz) | 5.0 | 5.5 | 6.0 | 4.2 | 4.2 |
| Συχνότητα ηχητικού φέροντος (Sound Carrier in MHz) | 5.5 | 6.0 | 6.5 | 4.5 | 4.5 |

**Πίνακας 2.2** Πίνακας σύγκρισης προτύπων [10]

## 2.10 Το πρότυπο SECAM

Το πρότυπο SECAM (**S**equential **C**ouleur **A**vec **M**emoire or **S**equential **C**olor with **M**emory) εμφανίστηκε επίσης στις αρχές του 1960 και εφαρμόστηκε στην Γαλλία. Το πρότυπο χρησιμοποιεί το ίδιο εύρος ζώνης με το σύστημα PAL αλλά μεταδίδει την χρωματική πληροφορία σειριακά. Το πρότυπο έχει προκαθορίσει επίσης 625 γραμμές ανά καρέ [8].

| **ΣΥΣΤΗΜΑ** | **SECAM B,G,H** | **SECAM D,K,K1,L** |
|---|---|---|
| Γραμμές ανά πεδίο (Lines/Field) | 625/50 | 625/50 |
| Οριζόντια Συχνότητα (Horizontal Frequency in kHz) | 15.625 | 15.625 |
| Κατακόρυφη Συχνότητα (Vertical Frequency in Hz) | 50 | 50 |
| Εύρος Ζώνης Video (Video Bandwidth in MHz) | 5.0 | 6.0 |
| Συχνότητα ηχητικού φέροντος (Sound Carrier int MHz) | 5.5 | 6.5 |

**Πίνακας 2.3.** Σύγκριση προτύπου SECAM [9]

## 2.11 Αποθηκευτικά μέσα

| | BetaSP | SVHS/Hi-8 | LaserDisc | VHS/8mm |
|---|---|---|---|---|
| Χρώμα (Color) | Component | Y/C | Composite | Composite |
| Ανάλυση σε γραμμές (Lines of Resolution) | 360 | 400 | 240 | 240 |
| Εύρος Ζώνης Σήματος (Signal Bandwidth) | 7.5 MHz | 4.5 MHz | 4.5 MHz | 2.5 MHz |

**Πίνακας 2.4.** Σύγκριση συστηματων αποθήκευσης [10]

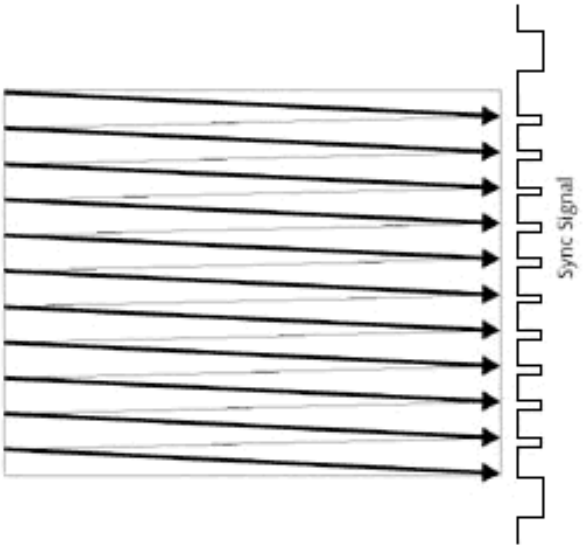
Από τη σκοπιά του αναλογικού σήματος έχουμε να παρατηρήσουμε τη μεγάλη διαφορά στο χρησιμοποιούμενο εύρος ζώνης του επαγγελματικού συστήματος απαράμιλλης ποιότητας Beta σε σύγκριση με αυτό του κοινού οικιακού VHS.

## 2.12 Σάρωση

Σάρωση είναι μια μορφή δειγματοληψίας ενός συνεχώς μεταβαλλόμενου, διδιάστατου σήματος.

### 2.12.1 Προοδευτική σάρωση ή Raster scanning

Είναι ο πιο συνήθες τρόπος που χρησιμοποιείται για χωρική δειγματοληπτική αναπαράσταση. Μετατρέπει τη δυδιάστατη φωτεινότητα εικόνας σε μονοδιάστατη κυματομορφή. Η εικόνα μιας σκηνής στην βιντεοκάμερα, σαρώνεται μια γραμμή κάθε φορά, από αριστερά στα δεξιά, και από πάνω προς τα κάτω.



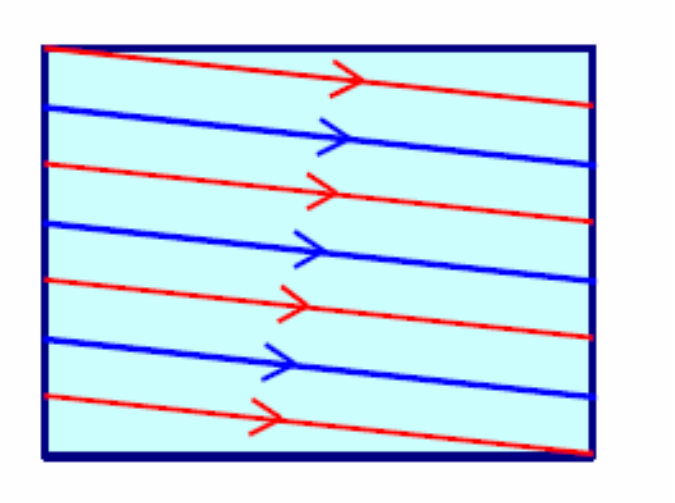

**Σχήμα 2.2** Προοδευτική σάρωση [3]

Αυτό που γίνεται ακριβώς είναι η μετατροπή των μεταβολών φωτεινότητας κατά μήκος κάθε γραμμής σε ένα ηλεκτρικό σήμα. Αφου η γραμμή σαρωθεί από αριστερά προς δεξιά, υπάρχει ένας χρόνος ανίχνευσης προς τα πίσω και σάρωση της επόμενης γραμμής από αριστερά προς δεξιά, όπως φαίνεται και στο σχήμα. 2.2.

### 2.12.2 Περιπλεκόμενη σάρωση (Interlaced Scanning)

Η επιλογή των αριθμών των γραμμών σάρωσης, περιλαμβάνει ένα αντάλλαγμα μεταξύ αλληλοσυγκρουόμενων απαιτήσεων εύρους ζώνης(bandwidth), τρεμόσβημα και ανάλυσης. Η περιπλεκόμενη σάρωση δοκιμάζει να πετύχει αυτά τα ανταλλάγματα, χρησιμοποιόντας πλαίσια εικόνων που συνθέτονται σε δυο πεδία που δειγματοληπτούνται σε διαφορετικούς χρόνους, τέτοια ώστε δυο συνεχόμενες γραμμές ενός πλαισίου εικόνας να ανήκουν σε διαφορετικά πεδία. Αυτό αντιπροσωπεύει ανταλλάγματα κάθετα-χρονικά σε χωρική και χρονική ανάλυση. Για παράδειγμα, αυτό επιτρέπει σε αργά κινούμενα αντικείμενα να γίνονται αντιληπτά με ψηλότερη κάθετη λεπτομέρεια, ενώ σε γρήγορα κινούμενα αντικείμενα να γίνονται αντιληπτά με ψηλότερο χρονικά ρυθμό, περίπου στη μισή κάθετη ανάλυση. Έτσι τα χαρακτηριστικά του ματιού ταυτίζονται γιατί το ανθρώπινο μάτι με αργή κίνηση μπορεί να αντιληφθεί

τις λεπτομέρειες, ενώ με γρηγορότερη κίνηση το ανθρώπινο μάτι δεν αντιλαμβάνεται τόσο εύκολα τις λεπτομέρειες.[5]



**Σχήμα 2.3** Περιπλεκόμενη σάρωση[5]

# Κεφάλαιο 3

## ΨΗΦΙΟΠΟΙΗΣΗ ΑΝΑΛΟΓΙΚΟΥ ΒΙΝΤΕΟ



### 3.1 Εισαγωγή

Στο κεφάλαιο αυτό γίνεται παρουσίαση των μεθοδολογιών μετατροπής αναλογικού βίντεο σε ψηφιακό. Η διαδικασία ψηφιοποίησης αναλογικού βίντεο αποτελείται από τις διεργασίες του φιλτραρίσματος, δειγματοληψίας, κβαντισμού και κωδικοποίησης.
Γίνεται περεταίρω ανάλυση της χρωματικηής δειγματοληψίας 4:2:0, την οποία και χρησιμοποιούμε. Έπειτα, δίνονται παραδείγματα μετατροπής RGB σε YUV χρωματοχώρο.

Γίνεται παρουσίαση των απαιτήσεων του γιατρού για το σύστημα που θα υλοποιήσουμε και παρουσιάζονται με λεπτομέρεια οι δοκιμές σε λογισμικό και υλικό προς ικανοποίηση των απαιτήσεων αυτών.

Οι πληροφορίες που παρουσιάζονται είναι βάση της βιβλιογραφίας [3,5,6,8,9,10,11].

### 3.1.1 Φιλτράρισμα

Η διαδικασία του φιλτραρίσματος γίνεται πριν την δειγματοληψία. Το φιλτράρισμα μπορεί να μειώσει τις περιττές και ανεπιθύμητες συχνότητες στο σήμα, καθώς επίσης και το θόρυβο. Η πιο απλή μέθοδος φιλτραρίσματος είναι με το να πάρουμε τους μέσους όρους των τιμών φωτεινότητας μέσα σε μια περιοχή και να αντικαταστήσουμε την τιμή φωτεινότητας του αρχικού σημείου με αυτή που υπολογίσαμε.

Στα σήματα βίντεο, το φιλτράρισμα που μπορεί να γίνει στο σήμα της φωτεινότητας, δύναται να είναι και διαφορετικό από αυτό των χρωματικών συνιστωσών.

### 3.1.2 Δειγματοληψία

Μετά το φιλτράρισμα το βίντεο δειγματοληπτείται κατά ένα αριθμό σημείων.Ο ελάχιστος αριθμός που πρέπει ένα αναλογικό σήμα να δειγματοληπτείται ονομάζεται Nyquist συχνότητα και αναλογεί στο διπλάσιο της μεγαλύτερης συχνότητας του σήματος. Για το NTSC αυτό είναι 2x4.2= 8.4 MHz ενώ για το PAL 2x5 = 10 MHz. Συνήθως η δειγματοληψία γίνεται σε πιο μεγάλες τιμές για να μην χάνουμε πληροφορία.

### 3.1.3 Κβαντισμός

Το σήμα που έχει δειγματοληφθεί, το οποίο είναι απλά μια ακολουθία από τιμές φωτεινότητας, κβαντοποιείται. Αυτό σημαίνει ότι όταν η τιμή εισόδου είναι σε ένα φάσμα τιμών, αναθέτουμε και βγάζουμε ως έξοδο μια μόνο τιμή που αναπαριστά αυτό το φάσμα τιμών. Έτσι μια διακριτή αναπαράσταση των τιμών επιτυγχάνεται. Η διαδικασία της κβαντοποίησης οδηγεί σε απώλεια πληροφορίας, αφού το

κβαντοποιημένο αποτέλεσμα είναι λιγότερο ακριβές από την αρχική τιμή. Η διαφορά της αρχικής τιμής και αυτής μετά την κβαντοποίηση ονομάζεται σφάλμα κβαντισμού. Η επιλογή στον αριθμό των επιπέδων αποτελεί ένα πάρε δώσε μεταξύ ακρίβειας της αναπαράστασης και το πόσο ορατός μέσα στο σήμα είναι ο θόρυβος.

### 3.1.4 Κωδικοποίηση

Το τελεταυταίο βήμα στην μετατροπή αναλογικού βίντεο σε ψηφιακό είναι η κωδικοποίηση. Πιο απλά, στην ψηφιακή αναπαράσταση. Για παράδειγμα, στην PCM κωδικοποίηση (Pulse Code Modulation), τα εικονοστοιχεία αναπαρίστανται με 8-bit κωδικοποιημένες λέξεις που αναθέτουν στο κάθε εικονοστοιχείο $2^8 = 256$ πιθανές τιμές από το 0 μέχρι το 256.

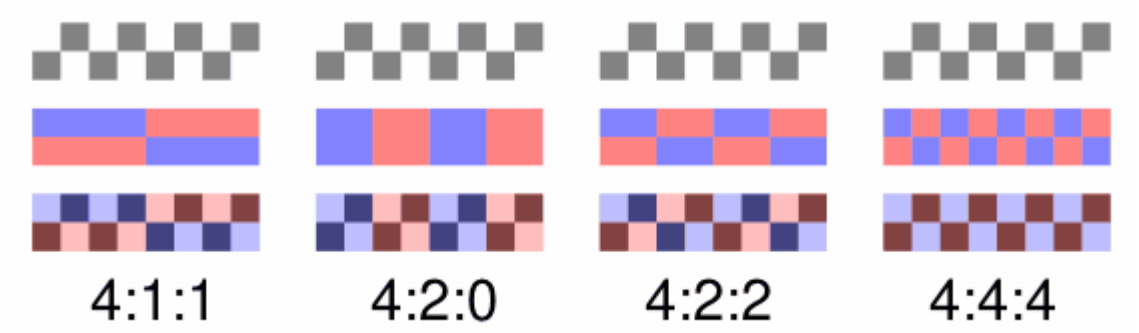
## 3.2 Χρωματική δειγματοληψία και η μορφή 4:2:0

Στην ψηφιακή επεξεργασία εικόνας, ονομάζουμε δειγματοληψία χρώματος (chroma sampling) ως την χρήση λιγότερης ανάλυσης για τις χρωματικές συνισταμένες σε μια εικόνα σχετικά με την φωτεινότητα. Αυτό χρησιμοποιείται όταν ένα αναλογικό component βίντεο ή ένα YUV σήμα δειγματοληπτείται ψηφιακά.

Επειδή το μάτι είναι λιγότερο ευαίσθητο στις αλλαγές του χρώματος παρά στην φωτεινότητα, οι χρωματικές συνισταμένες δεν είναι απαραίτητο να είναι τόσο καλά ορισμένες όσο η φωτεινότητα. Αρκετά συστήματα βίντεο, δειγματοληπτούν τις χρωματικές διαφορές των καναλιών του χρώματος σε λιγότερη βαθμό παρά την φωτεινότητα. Αυτό μειώνει το συνολικό κόστος σε εύρος φάσματος του σήματος του βίντεο χωρίς ουσιαστική απώλεα από την ποιότητα της εικόνας. Οι χαμένες τιμές ξαναστέλνονται από το επόμενο δείγμα για το συγκεκριμένο κανάλι.

Η δειγματοληψία σε ένα σύστημα βίντεο γίνεται συνήθως ως ένα κλάσμα τριών μερών. Οι τρεις παράγοντες είναι: ο αριθμός τη φωτεινότητας Y, ακολουθώμενο από τον αριθμό των δειγμάτων από το ένα από τις δυο χρωματικές συνισταμένες U/Cb και μετά το V/Cr, για κάθε περιοχή δειγματοληψίας. Για σκοπούς ποιοτικής σύγκρισης, μόνο το κλάσμα αυτών των τιμών είναι σημαντικός. Έτσι, το 4:4:4 θα μπορούσε να

ήταν 1:1:1, όμως παραδοσιακά οι τιμές φωτεινότητας έχουν πάντα τιμή 4, και οι υπόλοιπες τιμές παίρνουν ανάλογες τιμές.



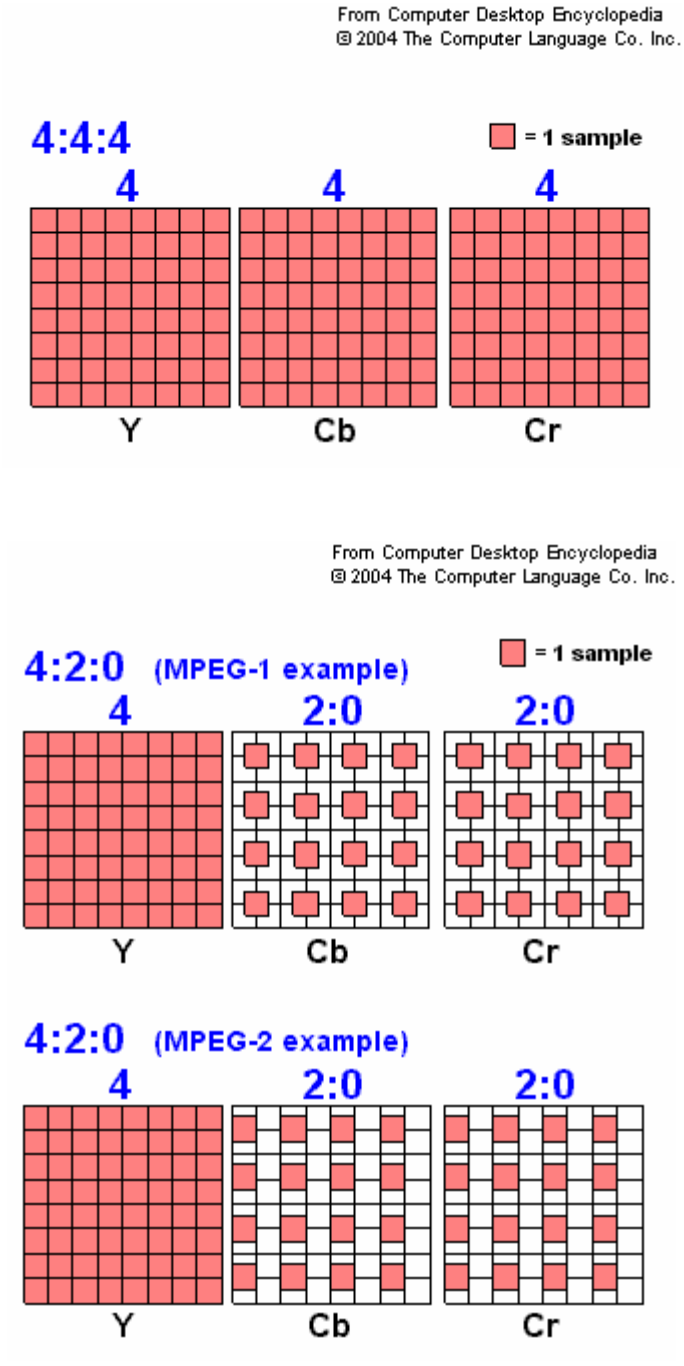

**Σχήμα 3.1** Δειγματοληψία βίντεο [9]

### 3.2.1 Πρότυπο 4:2:0

Το 4:2:0 δεν σημαίνει ότι δεν υπάρχει V ή Cb πληροφορία, αλλά σημαίνει ότι σε κάθε γραμμή, μόνο ένα χρωματικό κανάλι είναι αποθηκευμένο με μισή οριζόντια ανάλυση.

Ο ρυθμός δειγματοληψίας για το Y, Cr, Cb είναι 13.5 MHz, 3.375MHz και 0 MHz αντίστοιχα. Το κανάλι που αποθηκεύεται, δεν αποθηκεύεται στο επόμενο. Έτσι, στην μια γραμμή είναι 4:2:0 και στην άλλη, 4:0:2. Αυτό οδηγεί στον υποδιπλασιασμό της οριζόντιας και κάθετης ανάλυσης, δίνοντας το ένα τέταρτο της συνολικής ανάλυσης. Το PAL/SECAM ταιριάζουν με αυτό το χρωματικό σύστημα. Το ασυμπίεστο βίντεο σε αυτή τη μορφή με 8 bit κβαντισμό χρησιμοποιεί 6 bytes για κάθε macropixel.

Αυτός ο χρωματικός χώρος χρησιμοποιείται στα DVD, MPEG-2 εφαρμογές, PAL, DV και DVCAM.




**Σχήμα 3.2** Δειγματοληψία βίντεο [9]

### 3.3 RGB Χρωματοχώρος

Αυτός είναι ο πιο κοινός χώρος που χρησιμοποιείται σήμερα. Χρησιμοποιείται σε πολλές μορφές εικόνων χωρίς συμπίεση. Στον RGB ( Red Green Blue) υπάρχουν τρία κανάλια, κάθε ένα από αυτά δείχνει την ποσότητα από ένα χρώμα που έχει το τελικό

χρώμα. Ο συνδυασμός τους μας δίνει όλα τα χρώματα που μπορούν να είναι ορατά από το ανθρώπινο μάτι.

### 3.4 YUV Χρωματοχώρος

Ο χώρος αυτός χρησιμοποιείται από την αναλογική τηλεόραση παγκόσμια( PAL/SECAM, NTSC). Το χρωματικό μοντέλο αυτό διαφέρει από το RGB, το οποίοο είναι αυτό που βλέπουμε και αυτό που συλλαμβάνει η κάμερα. Αυτό έγινε γιατί στην δεκαετία του 1950, αποφασίστηκε να συνεχίσουν να επιτρέπουν στις μαυρόασπρες τηλεοράσεις να αποκωδικοποιούν μονοχρωματικά σήματα.

Το Y συμβολίζει την φωτεινότητα. Τα U και V δίνουν τις πληροφορίες του χρώματος και είναι σήματα χρωματικής αφαίρεσης του μπλε μείον την φωτεινότητα (B-Y) και ερυθρό μείον την φωτεινότητα (R-Y). Μέσα από μια διαδικασία που ονομάζεται «μετατροπή χρωματοχώρου», η βιντεοκάμερα μετατρέπει τα RGB δεδομένα που συλλαμβάνουν οι αισθητήρες της σε YUV σήμα. Όταν επιστρέψει στην οθόνη το σήμα, πρέπει πάλι να μετατραπεί σε RGB.

Το YUV είναι μαθηματικά ίσο με το RGB αν και τα χρωματικά κανάλια B-Y και R-Y περιέχουν την μισή από την ανάλυση της φωτεινότητας.

**Παράδειγμα μετατροπής RBG σε YUV**

$$
\begin{aligned}
Y &= 0.299R + 0.587G + 0.114B \\
U &= 0.492\,(B-Y) \\
V &= 0.877\,(R-Y)
\end{aligned}
\qquad (3.1)
$$

Το οποίο μπορεί επίσης να αναπαρασταθεί και ως:

$$
\begin{aligned}
Y &= 0.299R + 0.587G + 0.114B \\
U &= 0.147R - 0.289G + 0.436B \\
V &= 0.615R - 0.515G - 0.100B
\end{aligned}
\qquad (3.2)
$$

**Παράδειγμα μετατροπής από YUV σε RGB**

$$R = Y + 1.140V$$
$$G = Y - 0.395U - 0.581V$$
$$B = Y + 2.032U \qquad (3.3)$$

Αρχικά οι τηλεοράσεις συνδύαζαν την φωτεινότητα και τα χρωματικά κανάλια σε ένα κανάλι με ένα καλώδιο, γνωστό και ως composite video.
Παρατηρούμε πως η χρωματική δειγματοληψία συμπιέζει την πληροφορία.

## 3.5 Μεθοδολογία ψηφιοποίησης αναλογικού βίντεο

Τα βίντεο υπερήχων τα οποία είχαμε να επεξεργαστούμε, είχαν αποθηκευτεί σε S-VHS βίντεο κασέτες.

Για την μετατροπή του αναλογικού S-VHS βίντεο σε ψηφιακό, λάβαμε υπόψη ότι οι μηχανές και το λογισμικό που θα χρησιμοποιούσαμε θα μας παρείχαν τη δυνατότητα να κάνουμε την μετατροπή χωρίς να γίνεται συμπίεση (RAW) , να μην χάνονται πλαίσια(frames) κατά την διαδικασία αυτή, και τέλος, τα γράμματα που φαίνονται στο βίντεο να είναι ευανάγνωστα από τον γιατρό.

Η επεξεργασία βίντεο το οποίο έχει καταγραφεί αναλογικά, απαιτεί ειδικό εξοπλισμό:

- Καλώδιο επικοινωνίας συσκευής βίντεο με τον υπολογιστή
  S-Video, TV-in – για αναλογικό βίντεο
  USB, Firewire(IEEE1394, i-Link) – για ψηφιακό βίντεο
- Κάρτα σύλληψης (frame grabber card, όπως η MATROX meteor II) για ψηφιοποίηση ή λογισμικό ψηφιοποίησης.
- Ισχυρός επεξεργαστής
- Μεγάλη χωρητικότητα μνήμης RAM
- Ταχύς σκληρός δίσκος με μεγάλη χωρητικότητα

| | |
|---|---|
| Composite analog | Συνδυασμός χρωματικών συνιστωσών και φωτεινότητας. |
| Component analog | Διαχωρισμός των χρωματικών συνιστωσών και της φωτεινότητας.<br>S-Video( Y/C)<br>YCrCb |
| Composite digital | Ψηφιοποιημένο composite analog video |
| Component digital | Ψηφιακό σήμα με χρωματικά κανάλια R,G,B κόκκινο, πράσινο, μπλε. |

**Πίνακας. 3.1** Τύποι σήματος βίντεο.

### 3.6 Απαιτήσεις ως προς την ποιότητα, θέση της κάμερας για το βίντεο που θα ψηφιοποιηθεί

Αρχικά, με την Νίκη Γεωργίου , είδαμε τι είδους πλάκες θα θέλαμε να ψηφιοποιήσουμε. Ο γιατρός στις απαιτήσεις του, διευκρίνησε ότι θέλει μικρά κομάτια βίντεο, και όχι όλες τις κασέτες γιατί περιέχουν πολλές άχρηστες πληροφορίες.

Το πρώτο βίντεο που αναλύσαμε ήταν από την κασέτα cardiac1. Παρατήρηση της κύριας αρτηρίας την χρονική περίοδο 02:35-3:25.

Οι παρατηρήσεις του ιατρικού προσωπικού του Ινστιτούτου ήταν οι ακόλουθες:

1. Προτιμούμε οριζόντια τοιχώματα.
2. Εγκάρσια τομή της αρτηρίας.
3. Θέλουμε να βλέπουμε καθαρά την πλάκα.


**Σχήμα.3.3** Εγκάρσια τομή αρτηρίας. Παρατηρούμε το πρόσθιο τοίχωμα, οπίσθιο τοίχωμα και την πλάκα.

Στην συνέχεια, καθορίσαμε συγκεκριμένα τις απαιτήσεις του γιατρού.

1. Πρέπει να φαίνεται το όνομα του ασθενή καθαρά.
2. Η ημερομηνία γέννησης να φαίνεται καθαρά.
3. Ένα βίντεο πρέπει να έχει κατά προτίμηση τουλάχιστον τρεις παλμούς.

Ακολούθως, ο γιατρός υπέδειξε την σωστή διαδικασία που πρέπει να γίνεται στην εξέταση υπερήχων.

### 3.7 Διαδικασία εξέτασης υπερήχων

1. Ο ασθενής μένει ακίνητος για είκοσι δευτερόλεπτα.

2. Ο γιατρός καλεί τον ασθενή να παραμείνει ακίνητος χωρίς αναπνοή για όλο το διάστημα της λήψης.
3. Ο γιατρός, χωρίς να μετακινήσει τα χέρια του και διατηρώντας την ισορροπία του σώματος του με τους αγκώνες, λέει στον βοηθό του να γίνει η σύλληψη βίντεο με τους υπερήχους.

### 3.8 Επιλογή υλικού

Το πρόβλημα χωρίζεται σε επιλογή κατάλληλου λογισμικό και υλικού.
Από πλευρά του διαθέσιμου υλικού εξετάσαμε το Dazzle Digital Video Creator(Laptop), Aver Media, Matrox Meteor-II frame grabber. Βλέπε συγκριτικούς πίνακες και πίνακες υλικού στο Παράρτημα Ε.

### 3.9 Επιλογή λογισμικού

Για την επιλογή λογισμικού συγκριναμε C++, Windows Movie Player, Adobe Premiere, mplayer/mencoder.
Βλέπε Παράρτημα Ε (Πίνακας Λογισμικού).

### 3.10 Δοκιμή με Matrox Meteor II – Βιβλιοθήκη Matrox-C++ σε Windows 2000

Οι πρώτες δοκιμές ψηφιοποίησης εγιναν σε περιβάλλον Windows 2000, με κάρτα βίντεο Matrox Metero II. Χρησιμοποιήσαμε την βιβλιοθήκη της matrox σε συνδυασμό με C++.

Στα πειράματα σύλληψης, δεν είχαμε σταθερά απολέσματα στον αριθμό πλαισίων. Σε ορισμένα πειράματα ο αριθμός πλαισίων ήταν 13 πλαίσια ανά δευτερόλεπτο και σε άλλα ο αριθμός πλαισίων έφτανε τα 20 πλαίσια ανά δευτερόλεπτο. Σε αρκετές περιπτώσεις φτάναμε σε αποτυχημένη σύλληψη.
Ένα μεγάλο μειονέκτημα ακόμα και σε περιπτώσεις επιτυχούς σύλληψης με 20 πλαίσια ανά δευτερόλεπτο, ήταν τα δυσανάγνωστα γράμματα στο ψηφιοποιημένο βίντεο. Στις απαιτήσεις του ο γιατρός, έθεσε ως απαραίτητη προυπόθεση ευανάγνωστα γράμματα

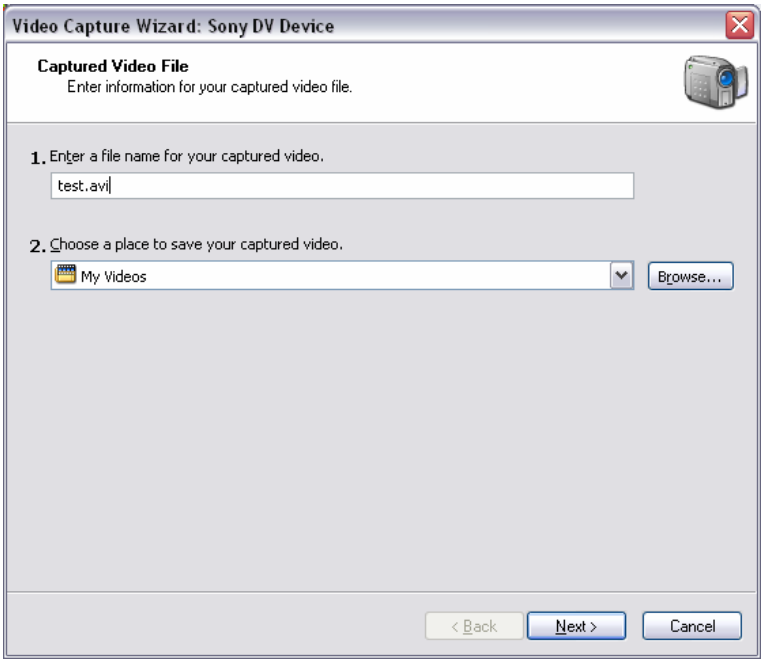
στην ψηφιοποιημένη μορφή του βίντεο για να αναγνωρίζει την κάθε περίπτωση ασθενούς.

Ο γιατρός απέρριψε την επιλογή αυτή λόγω των δυσανάγνωστων γραμμάτων στο βίντεο.

### 3.11 Δοκιμή με Dazzle και Microsoft Windows Movie Maker

Ακολούθως, χρησιμοποιήσαμε το Dazzle και το δωρεάν λογισμικό των Microsoft Windows XP Windows Movie Maker.

1. Από την συσκευή βίντεο, με S-video output, χρησιμοποιήσαμε το S-video input στο Dazzle.
2. Στην συνέχεια από το IEEE 1394 (FireWire) output, χρησιμοποιήσαμε το IEEE 1394 (FireWire) input σε φορητό υπολογιστή.
3. Το λειτουργικό σύστημα Microsoft Windows XP αναγνωρίζει αυτόματα την συσκευή και ενεργοποιεί το πρόγραμμα Microsoft Windows Movie Maker.
4. Ακολούθως, ενεργοποιείται η αρχική οθόνη για νέα σύλληψη βίντεο, όπου επιλέγουμε το όνομα του αρχείου βίντεο και τον κατάλογο στο οποίο θα αποθηκευτεί.

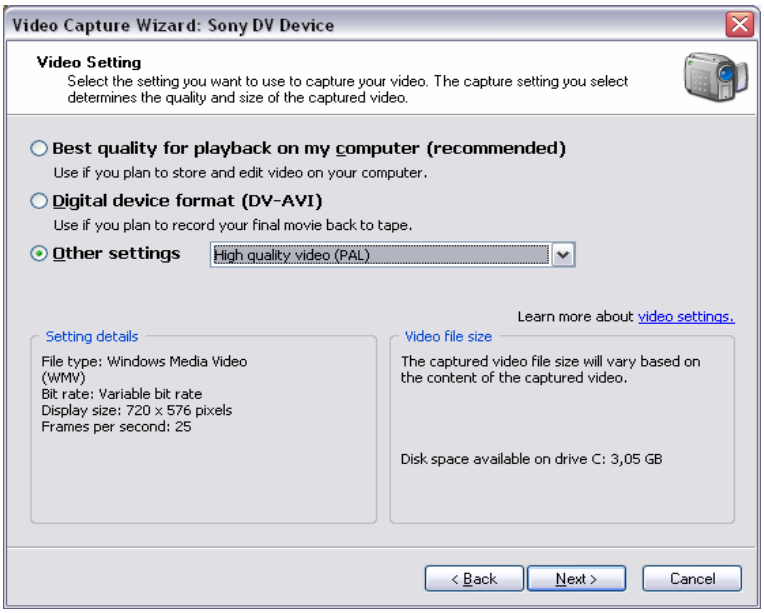



**Σχήμα 3.4:** Εισαγωγή ονόματος στο λογισμικό Microsoft Movie Maker

5. Στην συνέχεια δίνονται επιλογές για τον τρόπο με τον οποίο θα κωδικοποιηθεί το βίντεο. Για καλύτερη ποιότητα προσαρμωσμένη στις ρυθμίσεις του βίντεο που θέλουμε να συλλάβουμε, επιλέγουμε «άλλες επιλογές».

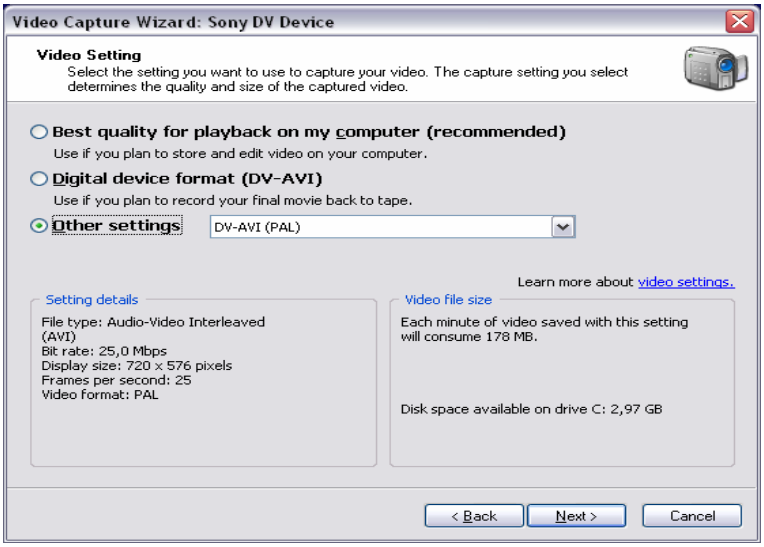

**Σχήμα 3.5:** Επιλογή ρυθμίσεων βίντεο στο λογισμικό Microsoft Movie Maker

6. Από τις άλλες επιλογές αρχικά επιλέξαμε υψηλής ποιότητας PAL για βίντεο 20 δευτερολέπτων. Το μέγεθος του αρχείου βίντεο που παράγεται είναι 14,1 MB (14.822.816 bytes). Είναι σαφές πως υπάρχει μεγάλο ποσοστό συμπίεσης.




**Σχήμα 3.6:** Ειδικές ρυθμίσεις στο λογισμικό Microsoft Movie Maker

7. Στο επόμενο πείραμα που δοκιμάσαμε, αντικαταστήσαμε την επιλογή για υπηλής ποιότητας PAL, σε DVI-AVI PAL το οποίο δημιούργησε αρχείο 70,9 MB (74.431.488 bytes). Και σε αυτή την επιλογή υπάρχει μεγάλο ποσοστό συμπίεσης.

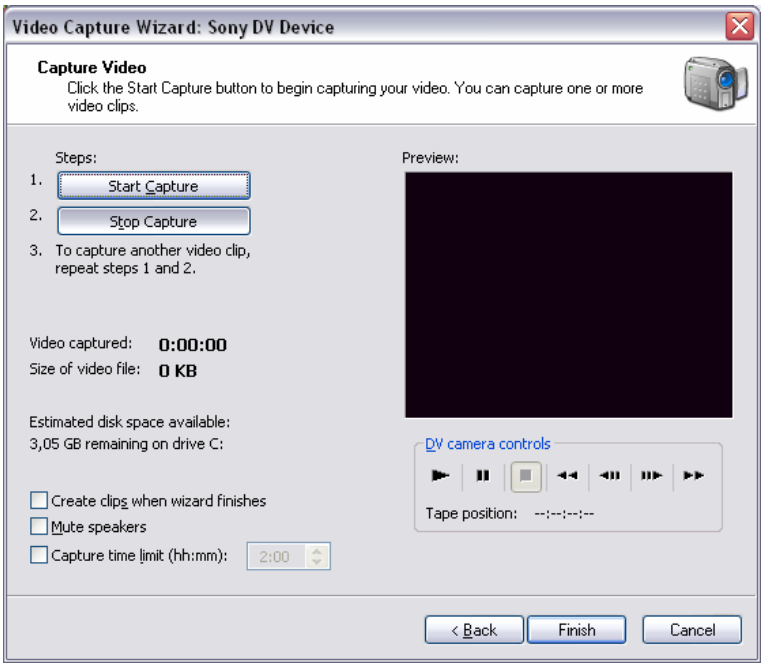



**Σχήμα 3.7:** Ειδικές ρυθμίσεις στο λογισμικό Microsoft Movie Maker

8. Ακολούθως επιλέγουμε να έχουμε έλεγχο της σύλληψης χωρίς προεσκόπηση του βίντεο λόγω του ότι μειώνεται η ποιότητα του βίντεο(χάνονται πλαίσια)


**Σχήμα 3.8** Ρυθμίσεις σύλληψης στο λογισμικό Microsoft Movie Maker

9. Τέλος, με την επιλογή για αρχή της σύλληψης αρχίζει η διαδικασία σύλληψης. Στο σημείο που θέλουμε να σταματήσει το πρόγραμμα την σύλληψη επιλέγουμε την κατάλληλη επιλογή.




**Σχήμα 3.9:** Επιλογή έναρξης και τέλους σύλληψης στο λογισμικό Microsoft Movie Maker

Το αναμενώμενο μέγεθος του αρχείου βίντεο που παράγεται υπολογίζεται ως ακολούθως:

Δεδομένα:

| | |
|---|---|
| Μέγεθος: | 720*576 (μέγεθος PAL) |
| Πλαίσια ανά δευτερόλεπτο: | 25fps |
| Χρόνος βίντεο: | 20s |
| Υπολογισμός: | (720*576)*(8*3)*25*20 / 8(bits ανα byte) = 622080000 |
| | = 593.26MB |

Οι πίνακες περιγραφής, δείχνουν τα προβλήματα-μειονεκτήματα και θετικά στοιχεία-πλεονεκτήματα που είχαμε με την κάθε επιλογή λογισμικύ-υλικού που πειραματιστήκαμε. Ένα παράδειγμα αποτελέσματος σύλληψης φαίνεται στο βίντεο με όναμα carotid_best.wmv.

### 3.12 Δοκιμή σε περιβάλλον LINUX και το λογισμικό mplayer-mencoder

Στην αναζήτηση για την καλύτερη επιλογή ψηφιοποίησης δοκιμάσαμε σε περιβάλλον LINUX το λογισμικό mplayer-mencoder.

Το λογισμικό mplayer/mencoder, χρησιμοποιείται μέσω της γραμμής εντολών. Υποστηρίζει τα περισσότερα πρότυπα συμπίεσης και αποσυμπίεσης, με κατάλληλη εγκατάσταση στο λειτουργικό σύστημα LINUX. Όσο αφορά τις δυνατότητες του σαν λογισμικό επεξεργασίας βίντεο, θεωρείται πλήρης.

Για την ψηφιοποίηση των κασέτων S-VHS, έπρεπε να λάβουμε υπόψη τα ακόλουθα:

1. Το βίντεο των υπερήχων έγινε με το πρότυπο PAL. Βάση αυτού, έπρεπε το μέγεθος του βίντεο να είναι 720*576. ( κανονικά το PAL έχει 625 γραμμές, όμως αυτές οι γραμμές δεν είναι μέρος του βίντεο αλλά πληροφορίες για τηλεοπτικούς δέκτες)
2. Το ψηφιοποιημένο βίντεο δεν πρέπει να συμπιεστεί.
3. Να αποθηκευτεί με ένα πρότυπο που να αναγνωρίζεται και σε περιβάλλον Microsoft Windows, για λόγους συμβατότητας.

| **NTSC** | **PAL/SECAM** |
|---|---|
| 2:1 interlaced | 2:1 interlaced |
| 525 lines per frame | 625 lines per frame |
| 60 refreshes per second | 50 refreshes per second |
| 30 frames per second | 25 frames per second |

**Πίνακας 3.2** Πρότυπα βίντεο [5]

| | **VHS (Video Home System ή Video Helican Scan)** | **Betacam SP** | **S-VHS ( Super VHS)** |
|---|---|---|---|
| Μορφή σήματος | Αναλογικό | Αναλογικό | Αναλογικό |
| Τύπος σήματος | Composite | Composite | Component |
| Μέγεθος κασέτας | ½ ίντζα | ½ ίντζα | ½ ίντζα |
| Ανάλυση | 300 γραμμές ανά πεδίο | 360 γραμμές ανά πεδίο | 480 γραμμές ανά πεδίο |
| Βασική χρήση | Κλασικές βιντεοκασέτες | Αποθήκευση αναλογικού βίντεο | Βελτιωμένη μορφή VHS |

**Πίνακας 3.3** Διαδεδομένα βίντεο formats

### 3.13 Μεθοδολογία ψηφιοποίησης σε περιβάλλον LINUX, χρήση mplayer-mencoder

1. Σύνδεση composite εξόδου από την συσκευή αναπαραγωγής βίντεο στην είσοδο της κάρτας βίντεο Aver Media EZ Maker
2. Σύνδεση composite εξόδου από την συσκευή αναπαραγωγής βίντεο στην είσοδο της τηλεόρασης για προβολή και προεσκόπηση του βίντεο προς ψηφιοποίηση.
3. Σε ένα κέλυφος (Shell), στον κατάλογο στον οποίο θέλουμε να αποθηκευτεί το βίντεο εκτελούμε την εντολή :

```
mencoder –tv driver=v412:width=720:height=578 tv:// -o <filename.avi> -ovc raw –nosound
```

όπου :

mencoder είναι το όνομα της εφαρμογής, μέρος του mplayer πακέτου. -tv driver=v4l2 η επιλογή αυτή χρησιμοποιεί το video for linux 2. Το δύο στο τέλος εννοεί την δεύτερη έκδοση του βίντεο για LINUX η οποία είναι ενημερωμένη και διορθώνει αρκετά προβλήματα της αρχικής έκδοσης. width=720:height=578: Οι προδιαγραφές για PAL βίντεο.

tv:// -o filename.avi: δηλώνει το αρχείο εξόδου

-ovc raw: δηλώνουμε πως θέλουμε το βίντεο να παραμείνει ασυμπίεστο. -nosound: για καλύτερα αποτελέσματα και καλύτερη ποιότητα βίντεο δεν κάνουμε σύλληψη του ήχου

4. Σε κάθε σύλληψη, το βίντεο αποθηκεύεται με την ακόλουθη μορφή: cardiac(αριθμός κασέτας)_(αριθμός σύλληψης). Στην περίπτωση που το βίντεο σταματήσει προς στιγμή για να εστιάσει ο χειριστής σε κάποιο σημείο και συνεχίζει με την ίδια πλάκα, τότε προσθέτουμε σαν επέκταση του ονόματος ένα αύξων αριθμό, αρχίζοντας από το ένα.
5. Συνήθως στην διαδικασία εξέτασης υπερήχων, γίνεται μέτρηση της πλάκας. Αυτό γίνεται τις περισσότερες φορές είτε στην αρχή, είτε στο τέλος μιας εξέτασης. Μερικές φορές κατά την διάρκεια εστίασης σε μια πλάκα. Οι μετρήσεις καταγράφονται μαζί με την παρακολούθηση της πλάκας, με τις ίδιες ρυθμίσεις και με όνομα το αρχικό όνομα ακολουθώμενο από το χαρακτηριστικό measurement.
6. Οποιαδήποτε παρατήρηση είναι χρήσιμη καταγράφεται σε ξεχωριστό μέρος του πίνακα μετρήσεων.Ένα παράδειγμα του πίνακα είναι το ακόλουθο.

| FILENAME | TIME ON TAPE | MEASUREMENT file name and time on tape | OBSERVATIONS |
|---|---|---|---|
| 21_okt_cardiac1_0001.avi | 02:35-03:25 | cardiac1_0001_measurements1.avi (1:38-2:09) | Η Νίκη μας δείχνει την κύρια αρτηρία |
| 21_okt_cardiac1_0002.avi | 32:28-32:34 | Cardiac1_0002_measurement_1.avi (32:35) | Ο γιατρός μας δείχνει τους κτύπους της καρδίας |
| 21_okt_cardiac1_0003.avi | 33:25-33:29 | Cardiac1_0002_measurement_1.avi (32:35) | **Ο γιατρός πιστεύει ότι αυτό το βίντεο είναι ιδανικό.** Μετά από προσεκτική παρατήρηση είπε πως υπάρχει κίνηση στα τοιχώματς της αρτηρίας, γι'αυτό να μην το χρησιμοποιήσουμε αυτό. |
| 21_okt_cardiac1_0004.avi | 33:08-33:25 | Cardiac1_0002_measurement_1.avi (32:35) | Τα τοιχώματα είναι σταθερά και παρατηρούμε την κίνηση στην |

| | | | |
|---|---|---|---|
| | | | πλάκα.<br>Σε μερικές περιπτώσεις έχουμε κίνηση στα εικονοστοιχεία. Στο βίντεο αυτό, παρατηρούμε το μεσαίο και το αριστερό μέρος της πλάκας στο οποίο βλέπουμε κυματοειδή κίνηση.<br>Αυτή η κίνηση δεν είναι αληθινή και πρέπει να το σημειώσουμε. Αυτό προκύπτει κατά την εξέταση υπερήχων το probe παίρνει το σήμα πίσω διαγώνια |

**Πίνακας 3.4** Παράδειγμα πίνακα μετρήσεων

7. Για την αποθήκευση του ψηφιοποιημένου βίντεο χρησιμοποιήσαμε αρχικά τον σκληρό δίσκο του υπολογιστή στον οποίο γίνεται η ψηφιοποίηση. Για την μεταφορά από το Ινστιτούτο για επεξεργασία χρησιμοποιήσαμε USB flash memory καθώς επίσης και USB hard disks.Για να χρησιμοποιηθούν οι συσκευές αυτές στο LINUX, χρησιμοποιήσαμε τις πιο κάτω εντολές:

```
fdisk –l
```

παρουσιάζει όλες τις συσκευές που είναι ενωμένες στον υπολογιστή.

```
mkdir /mnt/Win_Fat32
```

:

Δημιουργία φακέλου για χρήση κατά την λειτουργία της εξωτερικής μνήμης.

```
mount /dev/sda /mnt/Win_Fat32:
```

ενεργοποιεί την πρόσβαση και την χρήση της εξωτερικής μνήμης.Τέλος, αφού αποθηκευτούν τα δεδομένα εκτελούμε

```
umount /dev/sda:pnm:pgm
```

το οποίο απενεργοποιεί την πρόσβαση της εξωτερικής μνήμης.

### 3.14 Χαρακτηριστικά ψηφιοποιημένου βίντεο

Χρησιμοποιώντας την επιλογή –ovc raw το LINUX δεν εφαρμόζει συμπίεση στο βίντεο. Χρησιμοποιεί το YV12 codec, το οποίο είναι ένα διδιάστατο YCrCb 4:2:0. Δεν εφαρμόζει συμπίεση σε μεγάλο βαθμό.

Για κάθε εικόνα, τα δεδομένα για την φωτεινότητα για όλα τα εικονοστοιχεία αποθηκεύονται ακολουθιακά, ακολουθώμενα από τις πληροφορίες για το χρώμα. Αυτή είναι και η κύρια διαφορά μεταξύ RGB και YV12. Ο τρόπος δηλαδή που αποθηκεύονται στην μνήμη. Οι εικόνες αυτές αποθηκεύουν κάθε συνιστών χρώμα ξεχωριστά. Το γεγονός αυτός δίνει μεγάλη επιτάχυνση αφού όλα τα bytes του κάθε επιπέδου μπορούν να χρησιμοποιηθούν με τον ίδιο τρόπο. Επίσης, δίνει επιτάχυνση στη χρήση γιατί αν ένα φίλτρο δεν χρησιμοποιά όλα τα επίπεδα, δεν θα τροποποιήσει όσα δεν χρησιμοποιεί.

Σε όλες τις εικόνες αντιστοιχείται ένα μέγιστο. Το μέγιστο που δίνεται, μπορεί να περιγραφτεί ως το μήκος μιας γραμμής. Αυτό που είναι ενδιαφέρον είναι πως αυτό το μέγιστο, δεν είναι απαραίτητο να είναι ίσο με το πλάτος της εικόνας.
Για παράδειγμα, αν κόψουμε κάτι από την εικόνα, το μόνο που αλλάζει είναι το πλάτος της εικόνας. Το μέγιστο και η μέτρηση των bytes παραμένει η ίδια.
Η εικόνα τότε είναι κάπως έτσι:

rrrrrrrrrrrrrrrpppp

rrrrrrrrrrrrrrrpppp

rrrrrrrrrrrrrrrpppp

όπου ‘r’ είναι τα εικονοστοιχεία μέσα στην εικόνα που χρησιμοποιούμε και ‘p’ είναι το γέμισμα που προστίθεται μετά από κάθε γραμμή.

Στο YV12, το επίπεδο Y, έχει πάντα αριθμό bytes πολλαπλάσιο του 16, όμως είναι δυνατό να έχει mod2 πλάτους εικόνες γιατί το μέγεθος των γραμμών είναι διαφορετικό από το μέγιστο που ανάφερα πιο πάνω. Το μέγιστο είναι πάντα πολλαπλάσιο του 8.

# Κεφάλαιο 4

## Ανάλυση και Προετοιμασία Δεδομένων για Επεξεργασία



### 4.1 Εισαγωγή

Στο κεφάλαιο αυτό, θεωρώντας πως έχουμε πλέον ψηφιοποιήσει το αναλογικό βίντεο, το επόμενο στάδιο είναι η μετατροποπή των δεδομένων στην καλύτερη μορφή για επεξεργασία.

Στην κατοχή μας, έχουμε υλοποιήσεις σε γλώσσα C, των μεθόδων Horn and Schunck (original), Horn and Schunck(modified), Lucas and Kanade, Uras et al, Nagel, Anandan, Singh, Waxman και Fleet and Jepson από την δημοσίευση των J.L. Barron. D.J. Fleet and S.S. Beauchemin Performance of Optical Flow Techniques [2].

Οι υλοποιήσεις παίρνουν σαν είσοδο ακολουθία εικόνων σε μορφή SUN rastefile σε τιμές φωτεινότητας του γκρίζου. Υπάρχουν δηλαδή 32 byte επικεφαλίδες (headers) οκτώ ακέραιοι και ακολουθούν τα δυαδικά δεδομένα υπό μορφή unsigned char τιμών για κάθε εικονοστοιχείο. Ο δεύτερος και τρίτος αριθμός στην επικεφαλίδα δηλώνουν τον αριθμό των στηλών και γραμμών για κάθε εικόνα στην ακολουθία. Λόγω του ότι τα προγράμματα μπορούν να διαβάζουν τις επικεφαλίδες, μπορούν να επεξεργαστούν εικόνες διαφορετικών μεγεθών. Στα προγράμματα υπάρχουν δυο σταθερές PIC_X, PIC_Y. Αυτές πρέπει να είναι πάντα στο ίδιο μέγεθος με αυτό των εικόνων εισόδου. Οι εικόνες δεν είναι απαραίτητο να είναι τετράγωνες αν και στα περισσότερα παραδείγματα στην δημοσίευση είναι με τετράγωνες εικόνες.

Στα προγράμματα υπάρχει επίσης η επιλογή **–B**. Αυτή η επιλογή επιτρέπει να καθορίσουμε το μέγεθος των εικόνων απευθείας από την είσοδο. Αυτό είναι χρήσιμο αν σαν είσοδο έχουμε εικόνες οι οποίες δεν έχουνε επικεφαλίδες και έχουν μόνο δυαδικά δεδομένα με τις τιμές φωτεινότητας της εικόνας.

Οι εικόνες έχουν ένα συγκεκριμένο τύπο ονοματολογίας. Αποτελούνται από το όνομα της εικόνας και σαν επέκταση έχουν ένα αύξων αριθμό. Αυτό γίνεται για να διαβάζεται η ακολουθία από τις εικόνες χωρίς να αλλάζει η σειρά.
Οι πληροφορίες που παρουσιάζονται είναι βάση της βιβλιογραφίας [2], [4], [7], [11].

### 4.2 Εξαγωγή εικόνων από βίντεο

Το λογισμικό mplayer – mencoder μας επιτρέπει να μετατρέψουμε βίντεο από όλες τις μορφές που υποστηρίζει σε εικόνες jpeg, tga, pnm: ppm/pgm/pgmyuv, yuv4mpeg. Οι επιλογές για τις μορφές που μπορεί ο Mplayer να μετατρέψει ένα βίντεο, δίνονται σαν έξοδος όταν εκτελέσουμε την εντολή

```
mplayer –vo help.
```

Από τις επιλογές που υποστηρίζει ο Mplayer, επιλέξαμε τη μορφή PGM.
Η εντολή που κάνει την μετατροπή είναι η ακόλουθη:

```
mplayer όνομα-βίντεο.avi –vo
```

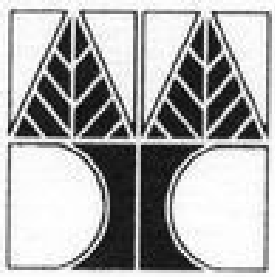
Οι PGM εικόνες είναι πολύ εύκολες στον χειρισμό. Οι εικόνες αυτές αναπαριστούν μια εικόνα σε τιμές φωτεινότητας του γκρίζου. Υπάρχουν πολλές υβριδικές μορφές. Γενικά όμως μπορούμε να σκεφτούμε τις PGM εικόνες σαν πίνακες τυχαίων ακεραίων. Το όνομά τους σημαίνει “Portable Gray Map”. Ένα αρχείο PGM μπορεί να περιγράφει μία ή και περισσότερες εικόνες.

Κάθε PGM εικόνα αποτελείται από τα ακόλουθα:

1. Ένα μαγικό αριθμό για να αναγνωρίζεται ο τύπος του αρχείου. Για το PGM ο μαγικός αριθμός είναι δύο χαρακτήρες “P5”.
2. Κενό ( TAB, κενοί χαρακτήρες κτλ.)
3. Το πλάτος σε ASCII αριθμούς.
4. Κενό
5. Το ύψος της εικόνας σε ASCII αριθμούς.

6. Κενό
7. Η μέγιστη τιμή σε ASCII αριθμούς. Στην περίπτωσή μας 255.
8. Καινούρια γραμμή ή άλλος χαρακτήρας που δείχνει κενό
9. Ένας πίνακας από γραμμές, μεγέθους που δίνεται από τον αριθμό του ύψους, από πάνω προς τα κάτω. Κάθε γραμμή αποτελείται από γραμμές πλήθους ίσο με το πλάτος, από αριστερά στα δεξιά. Κάθε τιμή είναι ένας αριθμός από το 0 μέχρι το 255, με το 0 να δείχνει το μαύρο και το 255 το λευκό. Κάθε τιμή παίρνει χώρο ένα byte. Το πιο σημαντικό bit είναι το πρώτο.
10. Κάθε τιμή μέσα στην εικόνα είναι ένας αριθμός που αντιστοιχεί στην τιμή φωτεινότητας.
11. Σειρές χαρακτήρων που αρχίζουν από "#" μέχρι την επόμενη γραμμή είναι σχόλια και αγνοούνται.

Ένα παράδειγμα PGM εικόνας είναι το ακόλουθο.



**Σχήμα 4.1** Εικονίδιο λογότυπου Πανεπιστιμίου Κύπρου [12].

Η πιο πάνω εικόνα διαβάζεται ως εξής:

P5

20 20

255

238 209 212 208 212 233 211 213 215 217 221 218 222 217 237 227 220 222 222 239

224 212 239 235 181 156 189 235 238 181 180 239 236 205 163 172 228 235 213 205

226 230 255 238 123 86 151 252 255 197 194 255 255 173 87 115 230 255 233 210

224 223 255 197 105 99 126 215 255 194 190 255 227 143 94 92 191 252 230 210

223 228 247 122 82 161 34 160 255 192 189 255 184 44 148 99 114 246 232 211

223 237 194 103 71 74 47 136 227 196 194 234 150 46 71 66 106 196 236 212

223 216 165 51 99 129 84 45 198 192 195 209 58 85 120 110 49 161 220 217

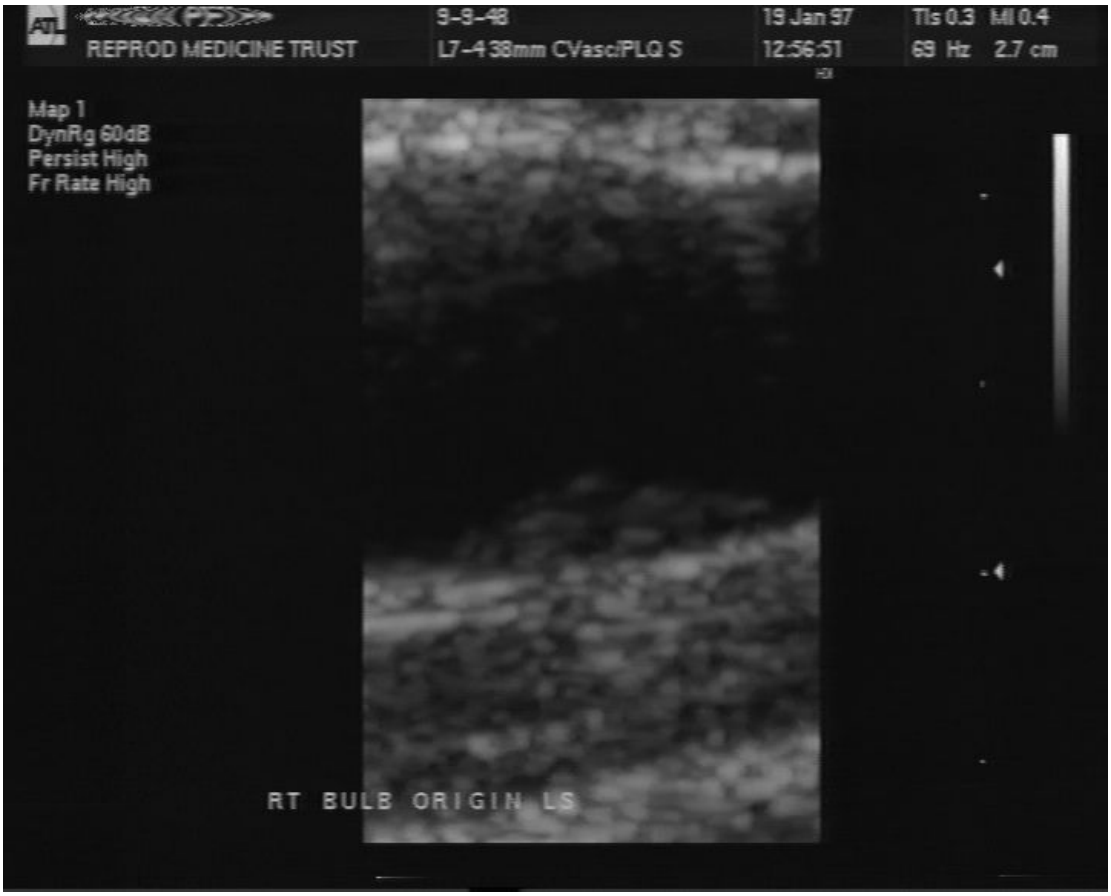
227 191 82 89 68 138 35 78 129 184 190 137 77 33 143 67 89 92 198 222
216 127 81 35 108 77 70 35 85 142 147 88 36 67 69 110 42 81 148 212
233 109 112 105 102 175 98 107 125 153 155 131 107 102 174 101 97 107 98 230
224 214 229 217 205 158 74 68 55 104 104 57 74 82 171 200 208 200 193 225
218 232 255 253 254 240 141 24 7 61 59 4 27 142 236 253 253 255 231 220
221 234 254 249 248 254 237 100 9 77 73 11 117 246 255 250 251 255 231 225
219 234 255 252 253 250 254 154 15 74 66 21 177 255 251 253 252 255 230 223
218 235 255 252 253 252 255 181 21 77 63 38 205 255 251 252 252 255 231 227
217 235 255 253 253 252 255 164 15 80 63 30 192 255 252 253 251 255 228 226
216 235 255 251 251 255 242 108 10 81 68 12 140 253 251 251 252 255 228 226
216 236 255 254 254 244 154 33 9 77 63 7 54 192 255 254 254 255 232 227
215 210 227 220 222 161 64 48 40 95 79 36 37 80 188 226 229 233 206 227
248 235 235 232 239 231 223 229 226 233 229 221 223 214 229 234 232 228 226 248

### 4.3 Χωρική αποκοπή

Η εντολή στον mplayer-mencoder που επιλέγει κομμάτι του βίντεο είναι η ακόλουθη:

```
mencoder –ovc raw –vf crop = 400:200 –nosound filename.avi  –o test1.avi
```

Στην επιλογή αυτή, χρησιμοποιείται το mencoder μέρος του πακέτου. Για παράδειγμα, το πιο κάτω βίντεο διαστάσεων 720x578 (σχήμα 4.2)




**Σχήμα 4.2** Εγκάρσια τομή πλάκας. Πλήρες δείγμα.

μετατρέπεται σε βίντεο διαστάσεων 400x200 με κέντρο το κέντρο του βίντεο πριν την αποκοπή.

**Σχήμα 4.3** Εγκάρσια τομή πλάκας, επιλογή πλάκας.

Η επιλογή του κέντρου αποκοπής μπορεί να γίνει και με επιλογή δική μας με την εντολή:

```
mencoder -ovc raw -vf crop=400:200:300:0 -nosound 21_Ok.avi -o test3.avi
```

που δίνει το βίντεο που φαίνεται πιο κάτω.


**Σχήμα 4.4** Παράδειγμα χωρικής αποκοπής με επιλογή κέντρου αποκοπής.

### 4.4 Χρονική αποκοπή

Η χρονική αποκοπή χρησιμοποιείται και στο λογισμικό με το γραφικό περιβάλλον που δημιουργήθηκε για τον σκοπό αυτό.
Αυτό γίνεται με την εντολή:

```
mencoder filename.avi -ss 0 -endpos 8 -ovc raw -noskip -o test4.avi
```

Η παράμετροι –ss, -endpos δηλώνουν το χρονικά αρχικό και τελικό σημείο, -ovc raw δηλώνει ότι θα γίνει χωρίς συμπίεση η έξοδος του βίντεο, -noskip δίνει έμφαση στο να μην χάνονται πλαίσια κατά την κωδικοποίηση. Τέλος test4.avi είναι το νέο αρχείο που δημιουργείται.

## 4.5 Δημιουργία γραφικού περιβάλλοντος για μετατροπή βίντεο

Κατόπιν εντολής του γιατρού, χρειαστήκαμε γραφικό περιβάλλον για αναπαραγωγή και επεξεργασία βίντεο. Για τον σκοπό αυτό, ήταν απαραίτητο το λογισμικό να είναι τρέχει σε περιβάλλον Microsoft Windows και να κάνει τα ακόλουθα:

1. Να μπορεί να αναπαράγει τα βίντεος που ψηφιοποιήσαμε.
2. Να επιλέγει κομμάτια βίντεο και να τα αποθηκεύει σε νέο αρχείο χωρίς απώλεια πληροφορίας.

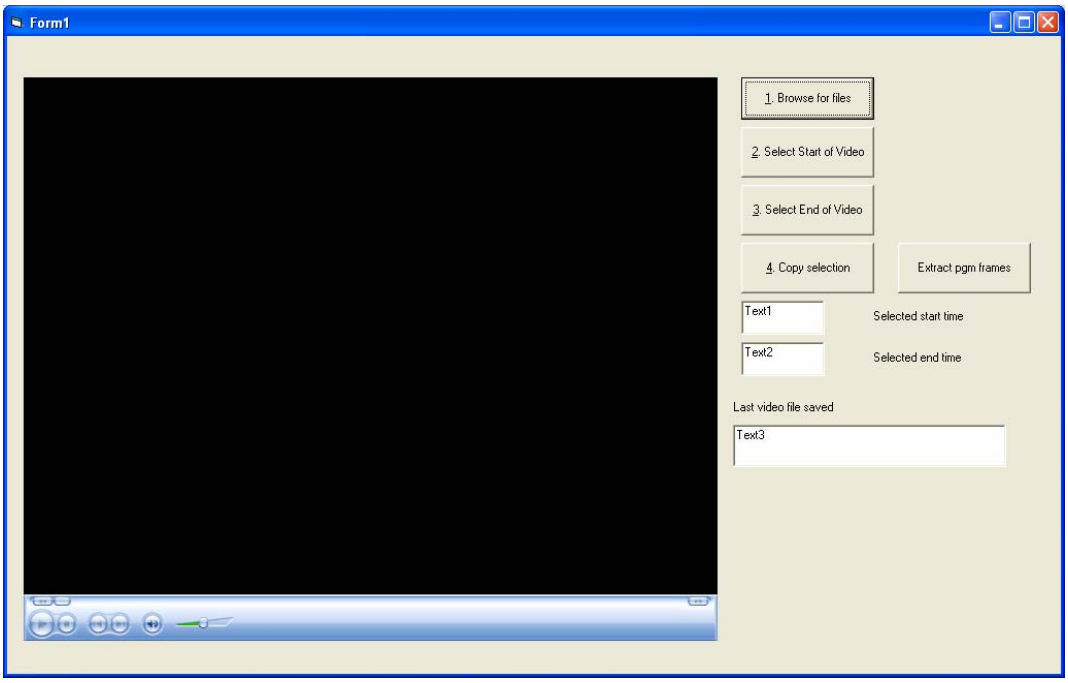
Σκοπός του λογισμικού αυτού, είναι να επιλέγει ο γιατρός κομμάτια που θεωρεί σημαντικά για επεξεργασία.

Για την υλοποίηση του λογισμικού δοκιμάσαμε την Borland Delphi, Microsoft Visual Basic και Matlab 6.5.

Λόγω του ότι πρόσφατα το λογισμικό mplayer-mencoder έγινε συμβατό με τα Microsoft Windows προτιμήσαμε να διατηρήσουμε το ίδιο λογισμικό. Για το γραφικό περιβάλλον χρησιμοποιήσαμε την Microsoft Visual Basic 6.0. Οι εντολές που χρησιμοποιά το γραφικό περιβάλλον είναι scripts που εκτελούν λειτουργίες του Mplayer-mencoder.

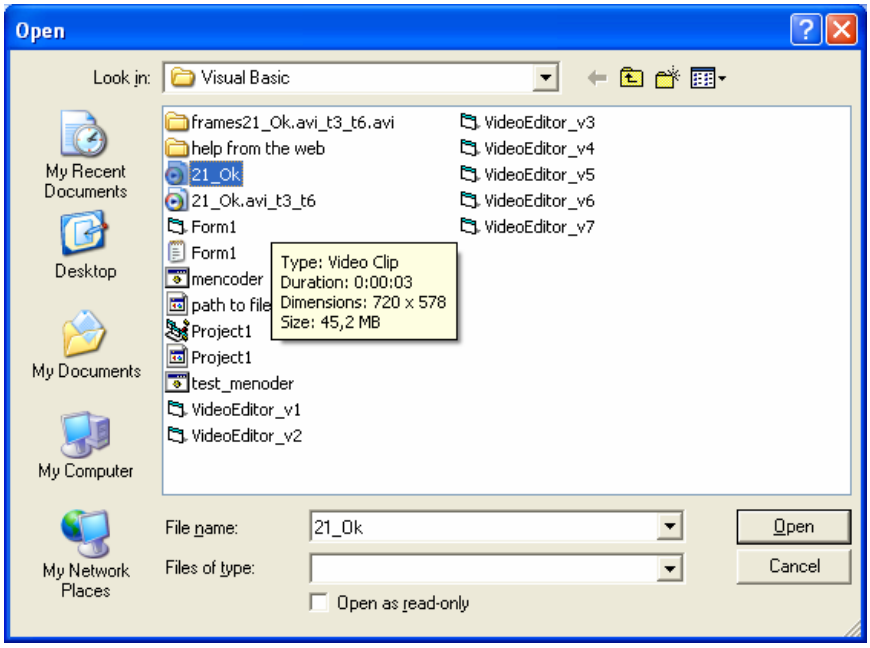
Επιπρόσθετα από τις αρχικές απαιτήσεις, υλοποιήθηκε η δυνατότητα εξαγωγής των εικόνων από το βίντεο σε PGM μορφή.

### Αρχική οθόνη



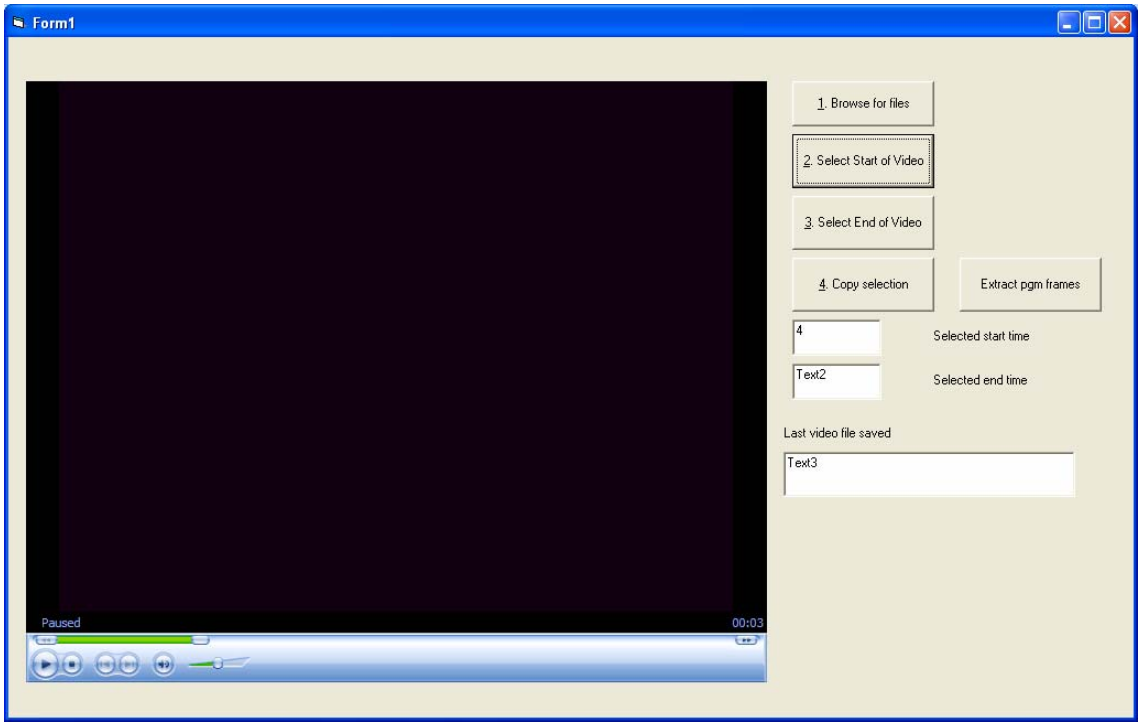

**Σχήμα 4.5** Αρχική οθόνη συστήματος.

### Αναζήτηση και επιλογή αρχείου βίντεο



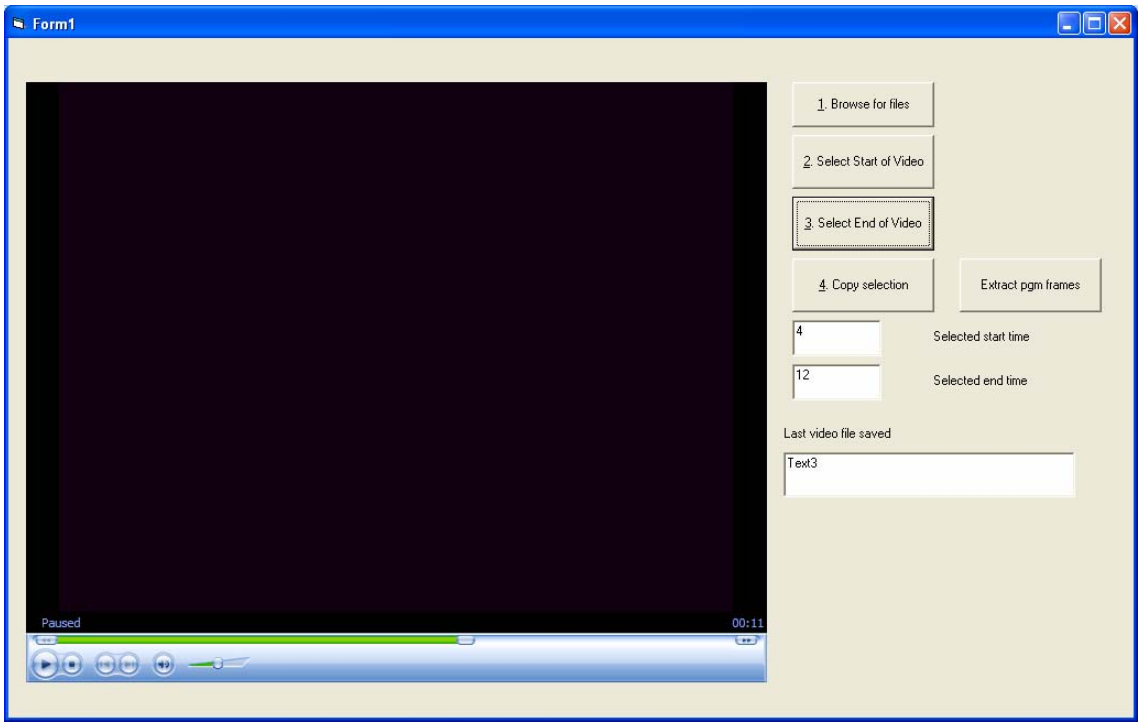

**Σχήμα 4.6** Επιλογή βίντεο για αναπαραγωγή στο σύστημα.

**Επιλογή αρχικού σημείου**



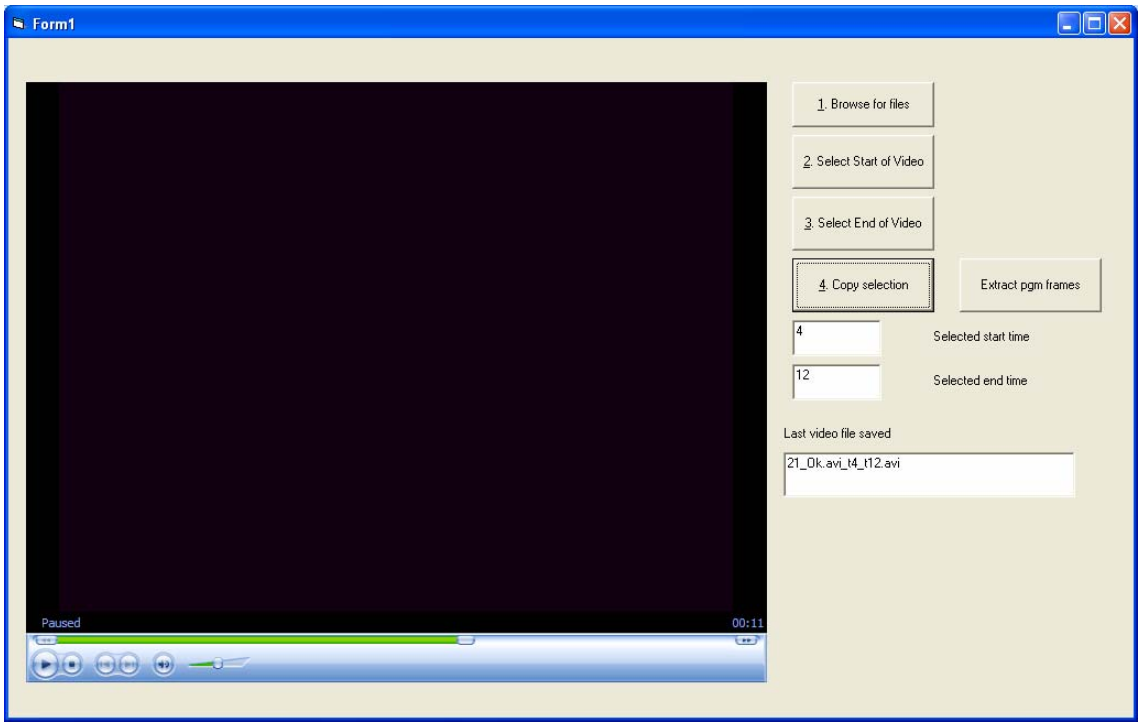

**Σχήμα 4.7** Επιλογή αρχικού σημείου. Ενώ αναπαράγεται το βίντεο, ο χρήστης έχει την δυνατότητα να επιλέξει χρονικά το σημείο αποκοπής. Η επιλογή φαίνεται με διακεκομμένη γραμμή.

**Επιλογή τελικού σημείου**




**Σχήμα 4.8** Επιλογή τελικού σημείου. Ενώ αναπαράγεται το βίντεο, ο χρήστης έχει την δυνατότητα να επιλέξει χρονικά το σημείο αποκοπής. Η επιλογή φαίνεται με διακεκομμένη γραμμή.

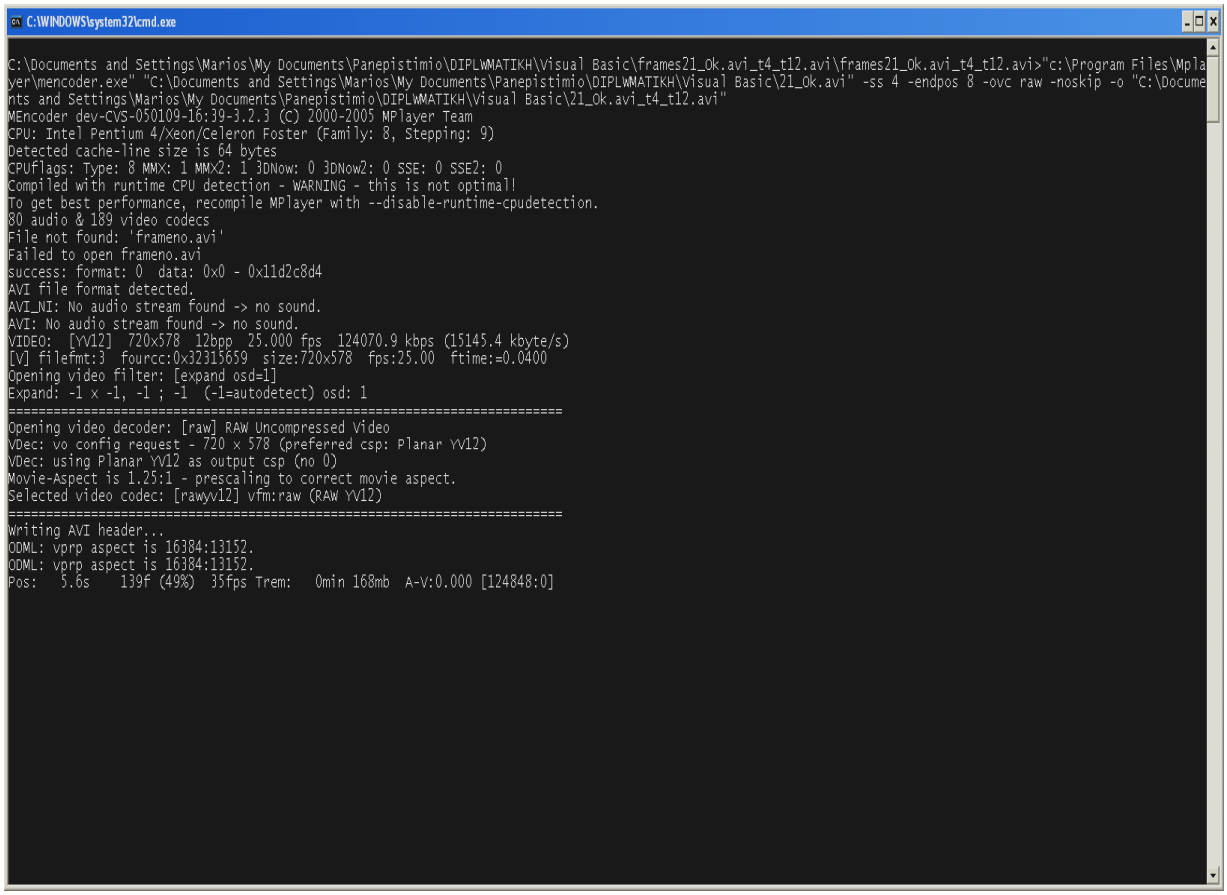
**Εξαγωγή επιλεγμένου βίντεο**




**Σχήμα 4.9** Επιλογή αντιγραφής αρχείου.Χρησιμοποιώντας την προεπιλεγμένο αρχικό και τελικό χρόνο, το σύστημα αντιγράφει το μέρος του βίντεο που είναι στο διάστημα αυτό. Η επιλογή φαίνεται με διακεκομμένη γραμμή.

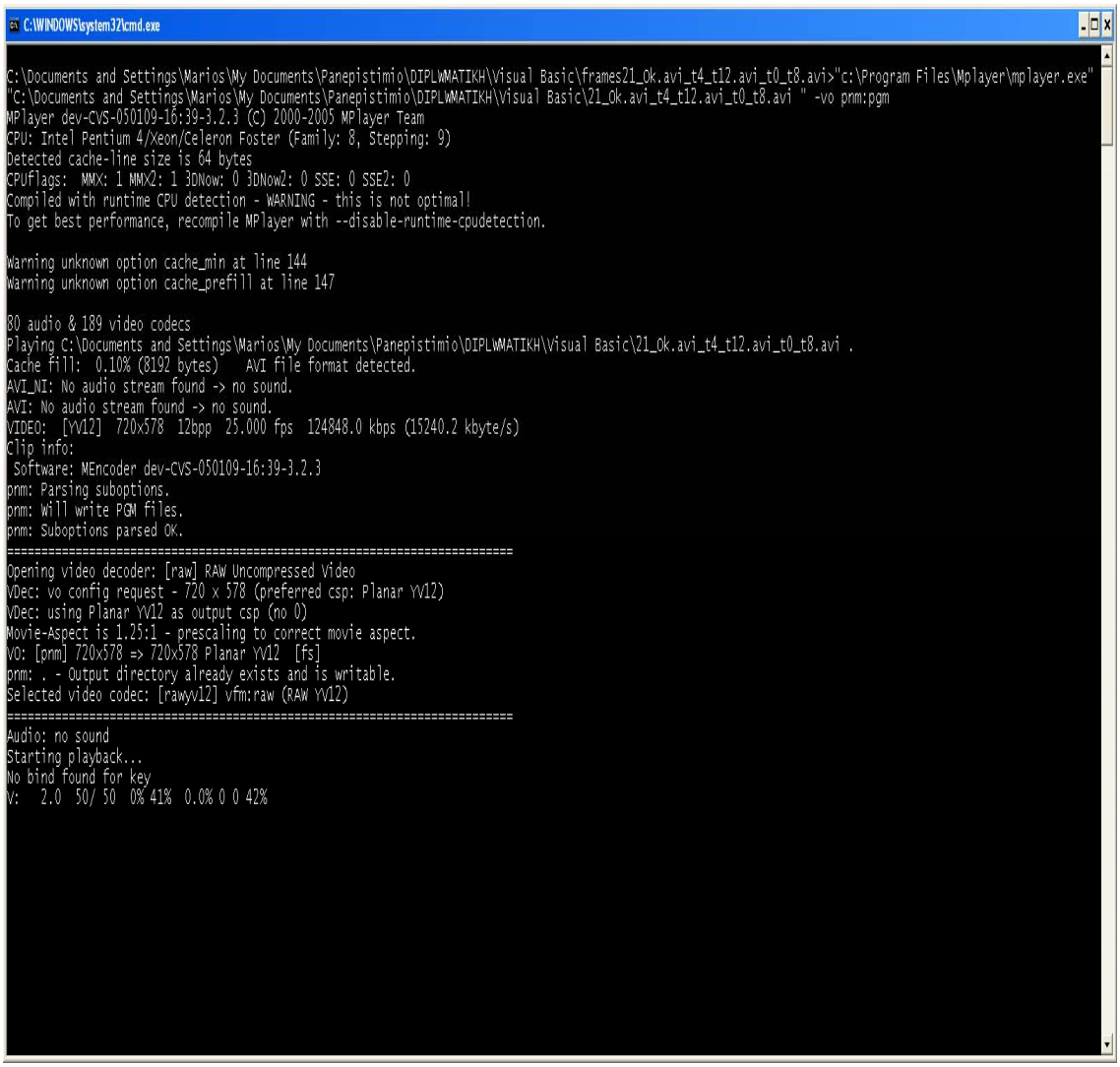
Σημειώνω εδώ πως το όνομα του νέου αρχείου βίντεο που δημιουργείται έχει όνομα το όνομα του αρχικού αρχείου, ακολούθως τον αρχικό χρόνο ακολουθώμενο από τον τελικό χρόνο. Η χρησιμότητα της μορφής αυτής είναι για να ξέρουμε ανά πάσα στιγμή από που προήλθε το βίντεο και την χρονική περίοδο που επιλέξαμε.
Με την επιλογή “Copy Selection” δημιουργειται ένα αρχείο batch, το οποίο περιέχει μέσα την εντολή. Έτσι μπορούμε και σε αυτή την περίπτωση να διατηρούμε τις απαραίτητες πληροφορίες για να επαναδημιουργήσουμε το νέο αρχείο αν χρειαστεί.

## Αντιγραφή βίντεο




**Σχήμα 4.10** Παράδειγμα διεργασίας αντιγραφής βίντεο

## Εξαγωγή εικόνων από βίντεο




**Σχήμα 4.11** Παράδειγμα διεργασίας εξαγωγής εικόνων

## 4.6 Μετατροπή PGM εικόνων σε SUN rastefiles

### 4.6.1 Μετατροπή με χρήση scripts

Για την μετατροπή των εικόνων από PGM σε SUN rasterfiles, χρησιμοποιήσαμε την βιβλιοθήκη ImageMagick και την χρήση scripts. Το script το οποίο μετατρέπει μια εικόνα από PGM σε SUN rasterfile είναι το ακόλουθο:

```
#!bin/bash
COUNTER=0
COUNTER=$(ls -la|grep ".pgm"|awk 'END {print NR}')
for i in $(ls *.pgm| sort -nr)
do
x=(${i%.pgm})
convert $i $x.ras
echo x: $x
echo i: $i
echo counter : $COUNTER
let COUNTER-=1
done
```

**Σχήμα 4.12** Παράδειγμα χρήσης script

Όπως βλέπουμε πιο πάνω γίνεται χρήση της εντολής convert. Η εντολή αυτή που ανήκει στην ImageMagick, παίρνει σαν είσοδο το όνομα της αρχικής εικόνας και το όνομα της εικόνας που θέλουμε μετά την μετατροπή. Διαβάζει την επέκταση και κάνει την μετατροπή. Για παράδειγμα αν έχουμε ως αρχική εικόνα με όνομα image.ras και τελική εικόνα την image2.pgm θα διαβάσει το ras και pgm και θα κάνει μετατροπή από SUN rasterfile σε pgm.

Το script μετατρέπει κάθε εικόνα και διατηρεί το όνομα. Επιπρόσθετα, μετρά το πλήθος των εικόνων που μετάτρεψε.

### 4.6.2 Μετατροπή σε MATLAB

Για τον ίδιο λόγο υλοποιήθηκε με τον ίδιο αλγόρθμο σε MATLAB πρόγραμμα μετατροπής των εικόνων.

```
function r = im2ras(A)
% Pairnei to onoma enos directory kai
% to metatrepei ola ta 'pgm' se Sun rasterfile format
% Usage:  im2ras path


% Get all the files from the directory
files = dir(A);

% Find the number of the files in the directory
number_of_file= size(files);
number_of_files=number_of_file(1);
count=0;
for i= 1:number_of_files

   f_name = files(i).name;
   [token,rem] = strtok(f_name,'.');
 f_name_format=rem;
 f_name_filename=token;
 if (strcmp(f_name_format,'.pgm')==1)
     pgm_image= imread(f_name);
     ras_name= strcat(f_name_filename,'.ras');
     imwrite(pgm_image,ras_name,'ras');
    count=count+1;
 end %end of if
end
fprintf(1,' Number of files converted : %d',count);
```

**Σχήμα 4.13** Παράδειγμα μετατροπής εικόνων .pgm σε .ras μορφή σε περιβάλλον MATLAB.

# Κεφάλαιο 5

## Οπτική ροή – Εκτίμηση Κίνησης



### 5.1 Εισαγωγή

Η εκτίμηση κίνησης, είναι ένα από τα βασικά προβλήματα στην επεξεργασία ψηφιακών βίντεο. Στο κεφάλαιο αυτό, γίνεται εισαγωγή στην οπτική ροή και εκτιμησης κίνησης. Παρουσιάζονται επίσης προβλήματα που προκύπτουν από την οπτική ροή, όπως το πρόβλημα της εγγραφής, ανοίγματος και αντιστοίχησης.

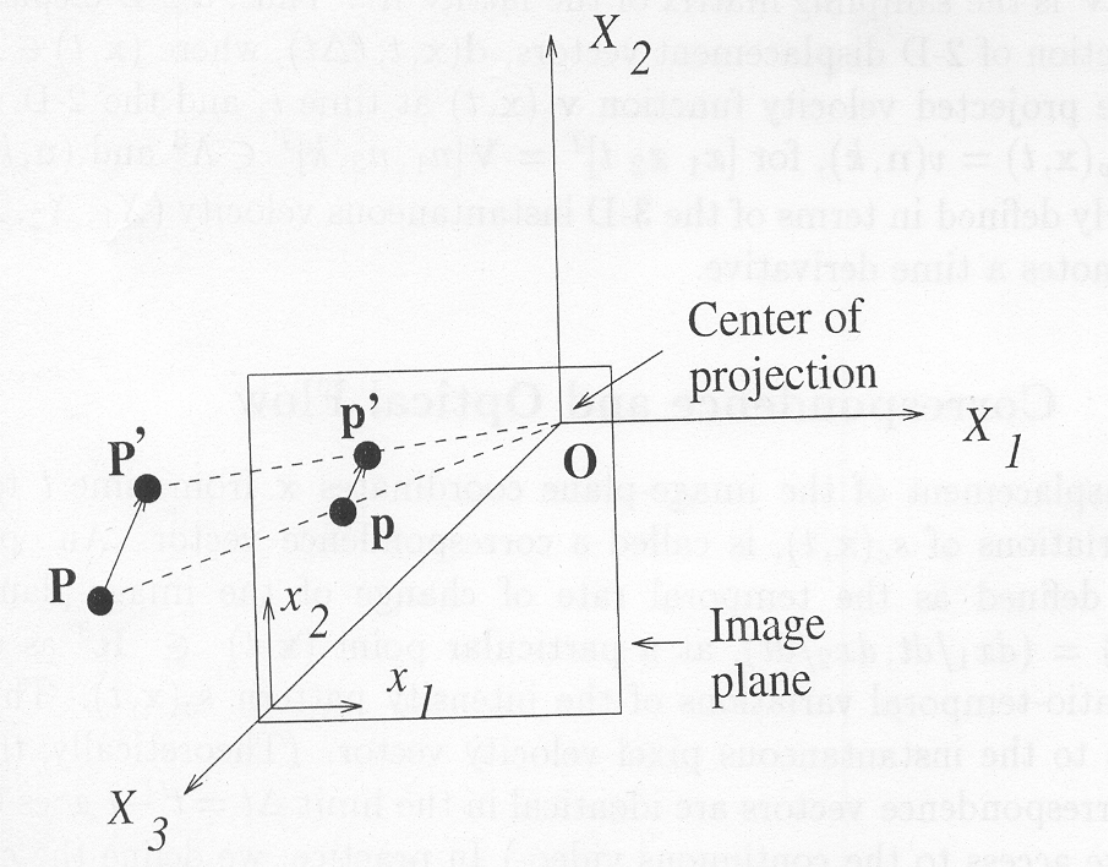
Ακολούθως παρουσιάζονται τα μοντέλα οπτικής ροής και εισαγωγή στον αλγόριθμο που χρησιμοποιήσαμε, τον αλγόριθμο υπολογισμού της οπτικής ροής, Horn and Schunck.

Λόγω του μεγάλου ποσοστού λάθους που παρουσιάζεται λόγω θορύβου και κατά την διάρκεια του κβαντισμού, στο κεφάλαιο αυτό γίνεται εισαγωγή στον υπολογισμό λάθους.

Τέλος, γίνεται αναφορά στα περιβάλλοντα εργασίας που χρησιμοποιήθηκαν κατά την μελέτη αυτή και το προτεινόμενο λογισμικό για την ψηφιοποίηση και επεξεργασία των βίντεο.

## 5.2 Κίνηση δύο διαστάσεων

Η κίνηση δύο διαστάσεων καλείται προβαλώμενη κίνηση. Αναφέρεται στην ορθογραφική προβολή της τρισδιάστατης κίνησης σε μια εικόνα. Η τρισδιάστατη κίνηση μπορεί να χαρακτηριστεί σε θέματα είτε τρισδιάστατης στιγμιαίας ταχύτητας, ή τρισδιάστατη επανατοποθέτηση των σημείων του αντικειμένου.




**Σχήμα 5.1**: Τρισδιάστατη Vs δισδιάστατη κίνηση [8]

Η προβαλλόμενη μετατόπιση μεταξύ t και t′ = t + l Δt, όπου l ένας ακέραιος και Δt είναι το χρονικό διάστημα του δείγματος, μπορεί να οριστεί για όλα τα $(x,t) \in \Re^3$, δίνοντας έτσι τη συνάρτηση του διανύσματος μετατόπισης $d_c$= (x,t, l Δt) των συνεχών μεταβλητών σε χώρο και χρόνο. Το 2-Δ διάνυσμα της μετατόπισης αναφέραιτε σε μια αναπαράσταση δείγματος που δίνεται από:

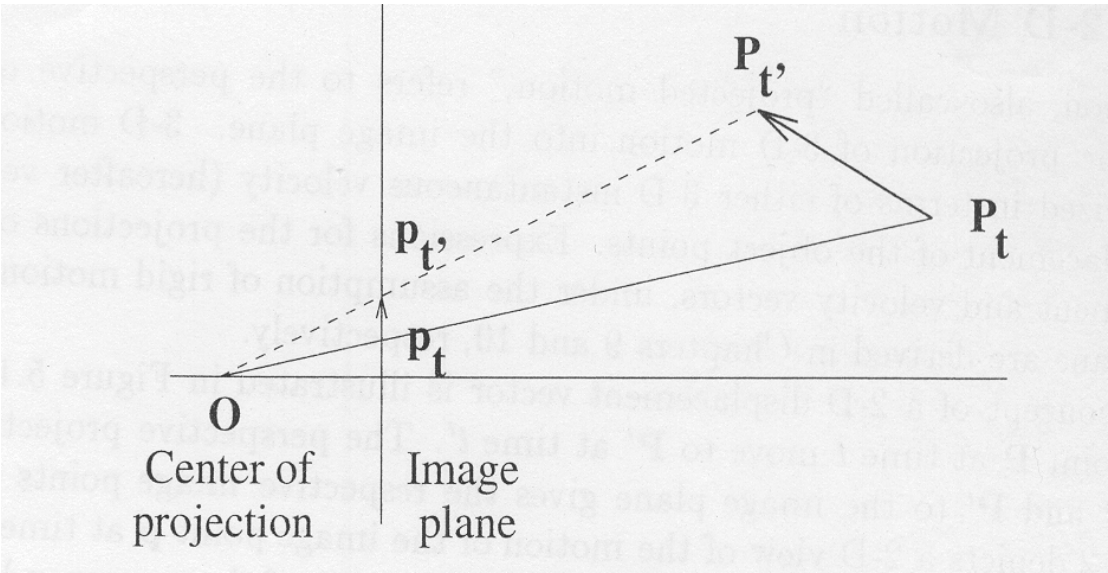
**Equation 1**

$$d_p(x,t,l\,\Delta t) = d_c(x,t,l\,\Delta t)\ ,\ (x,t) \in \Lambda^3\ ,$$

ή ισοδύναμα:

$$d_p(x,t,l\,\Delta t) = d_p(x,t,l\,\Delta t)|_{[x1\ x2\ t]^T = \mathbf{V}[n1\ n2\ k]^T},\ (\mathbf{n}, k) \in Z^3\ ,$$

όπου **V** είναι ο πίνακας δειγματοληψίας του πίνακα που περιγράφει τα εικονοστοιχεία $\Lambda^3$. Έτσι, το δισδιάστατο πεδίο μετατόπισης είναι μια συλλογή από δισδιάστατα διανύσματα μετατόπισης, d(x,t, l Δt) όπου $(x,t) \in \Lambda^3$.




**Σχήμα 5.2**: Η προβολή της κινησης [8]

## 5.3 Αντιστοίχηση και οπτική ροή

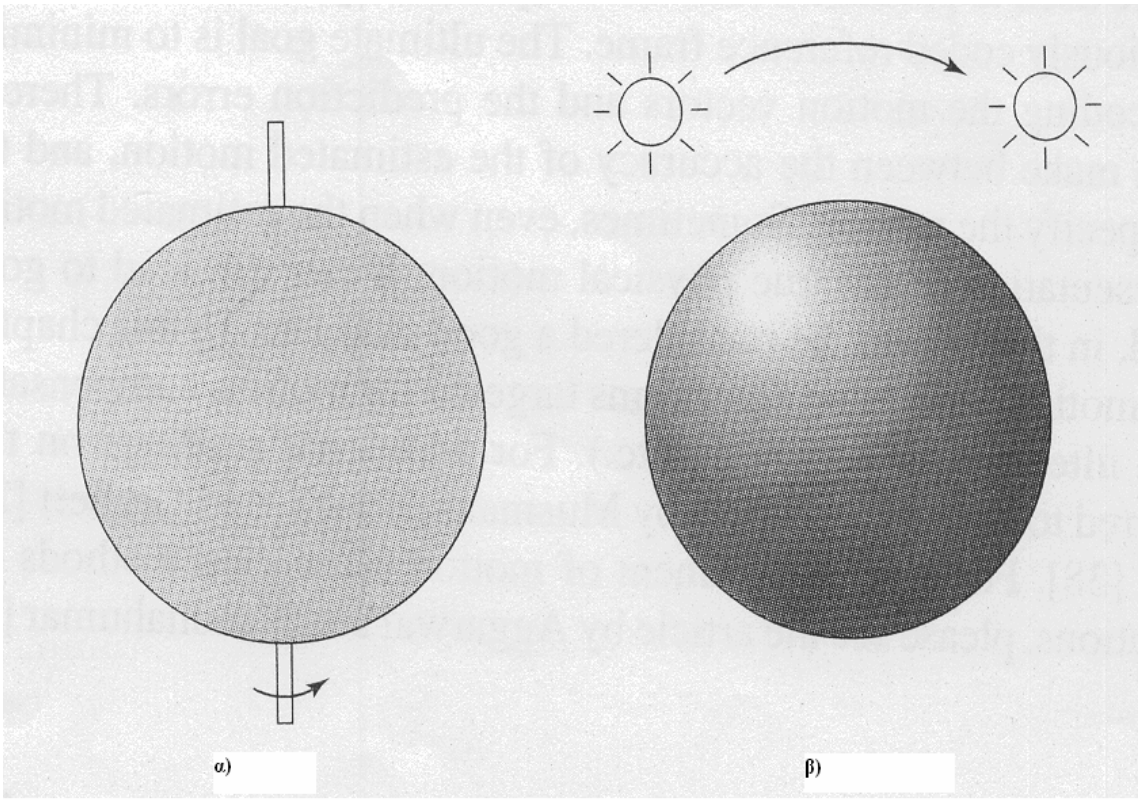
Η μετατόπιση της εικόνας ως προς τον άξονα των χ, από t σε t′, βασισμένο στις μεταβολές του $s_c(x, t)$ καλείται διάνυσμα αντιστοίχησης (correspondence vector). Το διάνυσμα της οπτικής ροής ορίζεται ως ο ρυθμός χρονικής μεταβολής των συντεταγμένων του σημείου στο επίπεδο της εικόνας, (v1,v2) = (dx1/dt, dx2/dt), όπου $(x,t) \in \Re^3$, όπως ορίζεται από τις μεταβολές σε χώρο και χρόνο του προτύπου που ακολουθεί η φωτεινότητα $s_c(x,t)$. Δηλάδη, ανταποκρίνεται στις στιγμιαίες ταχύτητες

του διανύσματος της ταχύτητας. Θεωρητικά, τα διανύσματα οπτικής ροής και αντιστοίχησης είναι ίδια στο όριο Δt=t′-t όταν αυτό τείνει στο μηδέν). Πρακτικά, ορίζουμε σαν το πεδίο αντιστοίχησης και οπτικής ροής, σαν ένα πεδίο διανυσματικό από μετατοπίσεις των εικονοστοιχείων ή την ταχύτητα αντίστοιχα, βασισμένο στις ορατές μεταβολές των φωτεινοτήτων στην εικόνα πάνω στον χωρο-χρονικό πίνακα $\Lambda^3$ .

Το πεδίο αντιστοιχίας και η οπτική ροή είναι γνωστά και ως φαινομενική δισδιάστατη μετατόπιση «apparent 2-D displacement» και φαινομενική δισδιάστατη ταχύτητα «apparent 2-D velocity».

Γενικά, το πεδίο αντιστοίχησης είναι διαφορετικό από την δισδιάστατη μετατόπιση και η οπτική ροή διαφορετική από την 2-Δ ταχύτητα για τους ακόλουθους λόγους:

1. Έλλειψη πληροφορίας στο χωρικό άνυσμα της εικόνας: Πρέπει να υπάρχει αρκετή πληροφορία μέσα σε μια κινούμενη περιοχή όσο αφορά το χρώμα για να παρατηρήσουμε την πραγματική κίνηση. Για παράδειγμα, όπως βλέπουμε και στην αριστερή εικόνα, μια σφαίρα με ομοιόμορφη ένταση γυρίζει γύρω από τον άξονα της. Η κίνηση αυτή δεν δημιουργεί οπτική ροή, έτσι δεν την παρατηρούμε.




**Σχήμα 5.3 α)**Σφαίρα με ομοιόμορφη ένταση γυρίζει γύρω από τον άξονα της. Η

κίνηση αυτή δεν δημιουργεί οπτική ροή **(β)** Σφαίρα με λεία επιφάνεια, η κίνηση της φωτεινής πηγής προκαλεί φαινομενική κίνηση στην σφαίρα [8]

2. Αλλαγές στην εξωτερική φωτεινότητα: Μια ορατή οπτική ροή μπορεί να μην αντικατοπτρίζει πάντα την πραγματική κίνηση. Για παράδειγμα, αν η εξωτερική ένταση αλλάζει από πλαίσιο σε πλαίσιο, όπως φένεται στην δεξιότερη σφαίρα. Αυτό μας κάνει να νομίζουμε ότι κινείται η σφαίρα, κάτι που δεν είναι αλήθεια, απλά μετακινείται η φωτεινή πηγή.

Συμπερασματικά, η 2-Δ μετατόπιση και τα πεδία της ταχύτητας είναι προβολές πάνω στα αντίστοιχα 3-Δ πεδία στην εικόνα, όπου τα πεδία της αντιστοίχησης και οπτικής ροής είναι η ταχύτητα και η συναρτήσεις μετατόπισης που παρατηρούνται από την μεταβολη της έντασης της εικόνας σχετικά με το χρόνο. Αφού μόνο αυτά μπορούμε να παρατηρήσουμε, υποθέτουμε ότι η 2-Δ κίνηση είναι το ίδιο.

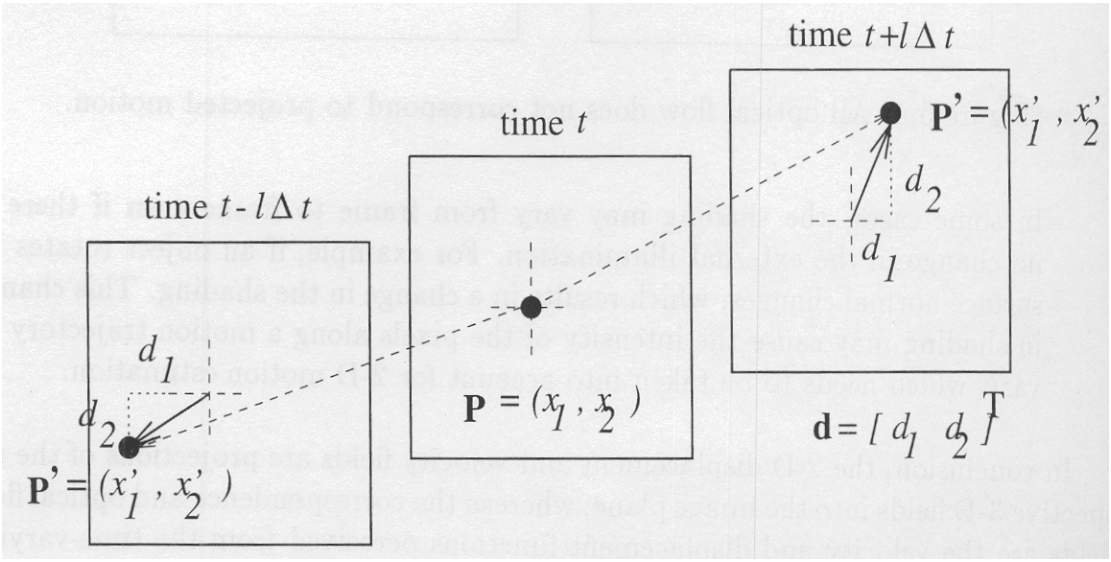
### 5.4 2-Δ Εκτίμηση κίνησης

Το πρόβλημα της 2-Δ εκτίμησης κίνησης, μπορεί να εκφραστεί ως:

1. Η εκτίμηση των διανυσμάτων αντιστοίχησης στην εικόνα $d(x, t; l\,\Delta t) = [d1(x, t; l\,\Delta t)]^T$ μεταξύ t και t+Δt, για κάθε $(x,t) \in \Lambda^3$ , όπου l είναι ακέραιος
2. Την εκτίμηση των διανυσμάτων οπτικής ροής $v(x,t) = [v1(x,t)\ v2(x,t)]^T$ για κάθε $(x,t) \in \Lambda^3$ . Τα διανύσματα αντιστοίχησης και οπτικής ροής συνήθως διαφέρουν από εικονοστοιχείο σε εικονοστοιχείο πχ λόγω επιτάχυνσης των αντικειμένων.

### 5.5 Το πρόβλημα αντιστοίχησης

Το πρόβλημα αυτό μπορεί να οριστεί ως ένα πρόβλημα ευθείας ή αντίστροφης εκτίμησης κίνησης πρόβλημα, αναλόγως αν το διάνυσμα κίνησης έχει οριστεί από t σε t+ l Δt ή από t σε t-l Δt όπως φαίνεται στην πιο κάτω εικόνα.




**Σχήμα 5.4** Ευθεία ή αντίστροφη εκτίμηση κίνησης [8]

**Ευθεία εκτίμηση κίνησης**: Δοθέντος των χωροχρονικών δειγμάτων $s_p(x,t)$ σε χρόνους t και t+ l Δt, που σχετίζονται με

$s_p(x_1, x_2,t) = s_c(x_1+d1(x,t; l \Delta t), x2 + d2(x,t; l \Delta t), t+ l \Delta t)$

ή ισοδύναμα

$s_p(x_1, x2 ,t) = s_{k+l}(x_1+d1(x), x_2 + d_2(x))$, τέτοιο ώστε $t=k\Delta t$

μπορούμε να βρούμε την πραγματική τιμή του διανύσματος αντιστοίχησης $d(x) = [d_1(x) d_2(x)]^T$, όπου οι χρονικές παραμέτροι του d(x) δεν λαμβάνονται υπόψη.

**Αντίστροφη εκτίμηση κίνησης:** Αν ορίσουμε τα διανύσματα αντιστοίχησης από t έως t- l Δt τότε το μοντέλο 2-Δ κίνησης γίνεται:

$S_k(x_1, x_2 ,t) = s_{k-l}(x_1+d_1(x), x_2 + d_2(x))$, τέτοιο ώστε $t=k\Delta t$

**Πρόβλημα Registration**

Το πρόβλημα registration είναι μια ειδική περίπτωση του προβήματος της αντιστοίχισης, όπου δυο πλαίσια μεταφέρονται σχετικά μεταξύ τους, για παράδειγμα πολλαπλές θέσεις μιας στατικής σκηνής με μια κάμερα που αλλάζει συντεταγμένες.

## 5.6 Εκτίμηση οπτικής ροής

Δοθέντωντων των δειγμάτων $s_p(x_1,x_2,t)$ σε ένα 3-Δ πίνακα θέσεων $\in \Lambda^3$, καθορίζουμε την 2-Δ ταχύτητα v(x,t) για κάθε $(x,t) \in \Lambda^3$. Η εκτίμηση των διανυσμάτων

της οπτικής ροής και αντιστοίχησης από δυο πλαίσια είναι ισοδύναμα, με $d(x,t; l\,\Delta t) = v(x,t)\, l\, \Delta t$, υποθέτωντας ότι η ταχύτητα παραμένει σταθερή καθόλη τη διάρκεια της χρονικής μετατόπισης. Σημειώνουμε εδώ πως κάποιος πρέπει να κοιτάξει περισσότερα από δύο πλαίσια ανά χρονική στιγμή για να υπολογίσουμε την οπτική ροή στην παρουσία επιτάχυνσης.

Η 2-Δ εκτίμηση κίνησης, που οριζεται είτε ως πρόβλημα αντιστοίχησης ή οπτικής ροής, όταν βασίζεται μόνο σε δυο πλαίσια, έχει ασάφειες όταν δεν εφαρμοστούν κάποιες υποθέσεις για την φύση της κίνησης.

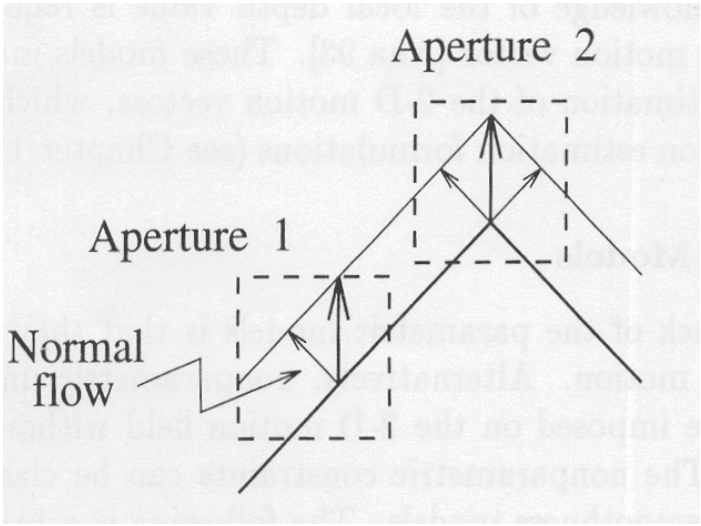
Η ασάφεια βρίσκεται λόγω των ακόλουθων:

1. Δεν μπορεί να εφαρμοστεί αντιστοίχηση για επικαλυμένα ή μη επικαλυμένα σημεία του φόντου. Αυτό είναι γνωστό ως το πρόβλημα του κλεισίματος(occlusion).
2. Αν τα συστατικά της μετατόπισης ή της ταχύτητας σε κάθε εικονοστοιχείο μεταχειρίζονται σαν ξεχωριστές μεταβλητές, τότε ο αριθμός των αγνώστων γίνεται διπλάσιος από τον αριθμό των συστατικών στη διαφορά των πλαισίων. Αυτό οδηγεί στο πρόβλημα του ανοίγματος (aperture).
3. Η εκτίμηση κίνησης είναι εξαιρετικά ευαίσθητη στην παρουσία θορύβου στις εικόνες του βίντεο. Η παραμικρή παρουσία θορύβου μπορεί να οδηγήσει σε μεγάλη απόκλιση στην εκτίμηση της κίνησης.

## 5.7 Το πρόβλημα του ανοίγματος (The aperture problem)

Το πρόβλημα του ανοίγματος είναι μια επαναδιατύπωση του γεγονότος ότι η λύση στο 2-Δ πρόβλημα εκτίμησης κίνησης δεν είναι μοναδική. Αν τα διανύσματα κίνησης σε κάθε εικονοστοιχείο θεωρούνται ως ξεχωριστές μεταβλητές, τότε υπάρχουν διπλάσιοι άγνωστοι απ'ότι εξισώσεις. Ο αριθμός των εξισώσεων είναι ίσος με τον αριθμό των εικονοστοιχείων στην εικόνα, όμως κάθε εικονοστοιχείο στο διάνυσμα κίνησης έχει δυο συστατικά.

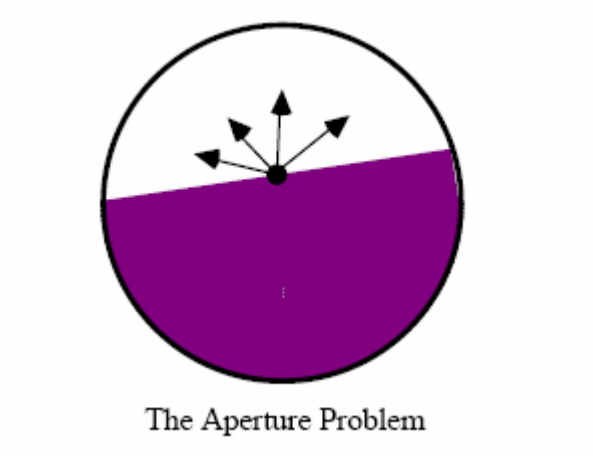
Έστω ότι έχουμε μια γωνιά ενός κινούμενου αντικειμένου, το οποίο κινείται προς τα πάνω. Αν υπολογίσουμε την κίνηση με βάση ένα τοπικό παράθυρο, στο άνοιγμα 1, τότε δεν είναι δυνατό να καθορίσουμε αν η εικόνα κινείται πάνω ή αν κινείται κάθετα στην γωνιά. Η κίνηση στην διεύθυνση κάθετα της γωνιάς ονομάζεται κανονική ροή.




**Σχήμα 5.5** Παράδειγμα προβλήματος του ανοίγματος [8]

Εντούτοις, αν κοιταξουμε στο άνοιγμα 2, τότε είναι δυνατό να υπολογίσουμε την σωστή κίνηση, αφού η εικόνα έχει κλίση σε δυο κάθετες κατευθύνσεις μέσα στο άνοιγμα. Έτσι, είναι δυνατό να υπερκαλύψουμε το πρόβλημα του ανοίγματος υπολογίζοντας την κίνησης στηριζόμενοι στο μπλοκ από τα εικονοστοιχεία που περιέχουν ικανοποιητική μεταβολή στις τιμές φωτεινότητας.

Για να το καταλάβουμε καλύτερα αυτό, ας δούμε και το πιο κάτω παράδειγμα. Θεωρούμε πως είμαστε σε ένα πλοίο και βλέπουμε προς τα έξω. Έξω υπάρχει το η θάλασσα που είναι φουρτουνιασμένη και το πλοίο που κινείται πάνω κάτω.




**Σχήμα 5.6** Παράδειγμα προβλήματος του ανοίγματος [5]

Αν παρατηρήσουμε ότι η άκρια κινείται προς το πάνω, η αληθινή κίνηση μπορεί να είναι μια από τις υπόλοιπες ενδεικνυόμενες κατευθύνσεις.

## 5.8 Μοντέλα 2-Δ πεδίου κίνησης

Λόγω των αδυναμιών της φύσης του προβλήματος , οι αλγόριθμοι εκτίμησης κίνησης χρειάζονται κάποιες επιπρόσθετες υποθέσεις(μοντέλα) για την δομή του 2-Δ πεδίου κίνησης.

### 5.8.1 Παραμετρικά μοντέλα

Τα παραμετρικά μοντέλα έχουν ως στόχο να περιγράψουν την ορθογραφική ή προοπτική προβολή της 3-Δ κίνησης (μετατόπιση ή ταχύτητα) μιας επιφάνειας πάνω στην εικόνα. Γενικά, εξαρτώνται από την αναπαράσταση της 3-Δ επιφάνειας. Για παράδειγμα, το 2-Δ πεδίο κίνησης που προκύπτει από 3-Δ σταθερή κίνηση μιας επίπεδης επιφάνειας με ορθογραφική προβολή, μπορεί να περιγραφεί με μια ένα ανάλογο μοντέλο 6 παραμέτρων, ενώ με προοπτική προβολή μπορεί να περιγραφεί με μη γραμικό μοντέλο 8 παραμέτρων.

Μια υποκλάση των παραμετρικών μοντέλων είναι τα καλούμενα quasi-παραμετρικά μοντέλα που χρησιμοποιούν το βάθος κάθε τρισδιάστατου σημείου σαν ένα ξεχωριστό άγνωστο. Ακολούθως, οι έξι 3-Δ παράμετροι κίνησης περιορίζουν το τοπικό διάνυσμα ροής να είναι μεταξύ συγκεκριμένης γραμμής, ενώ το να γνωρίζουμε για το εύρος της τιμής απαιτείται για να καθορίσουμε ακριβώς την τιμή του διανύσματος κίνησης. Αυτά τα μοντέλα μπορούν να χρησιμοποιηθούν ως περιορισμοί για να συντονίσουν την εκτίμηση των 2-Δ διανυσμάτνω κίνησης, που οδηγεί σε ταυτόχρονη 2-Δ και 3-Δ διατύπωση εκτίμηση κίνησης.

### 5.8.2 Μη παραμετρική μοντέλα

Το μεγαλύτερο μειονέκτημα των παραμετρικών μοντέλων είναι πως εφαρμόζονται μόνο σε 3-Δ σταθερή κίνηση. Εναλλακτικά, μη παραμετρικοί ομοιόμορφοι περιορισμοί (εξομάλυνση) μπορούν να εφαρμοστούν στο 2-Δ πεδίο

κίνσης χωρίς την εφαρμογή 3-Δ μοντέλων σταθερής κίνησης. Οι μη παραμετρικοί περιορισμοί μπορούν να διαχωριστούν σε ντετερμινιστικοί και στοχαστικοί.

**Μέθοδοι βάση εξίσωσης οπτικής ροής (OFE):** Οι μέθοδοι που βασίζονται στην οπτική ροή προσπαθούν να εκτιμήσουν το πεδίο της οπτικής ροής σχετικά με τις τιμές φωτεινότητας στην εικόνα σχετικά με το χώρο και το χρόνο. Στις μονοχρωματικές εικόνες, η εξίσωση οπτικής ροής πρέπει να χρησιμοποιηθεί σε συνδυασμό με ένα κατάλληλο περιορισμό εξομάλυνσης, που απαιτεί το διάνυσμα μετατόπισης να διαφέρει ελαφρά σε μια γειτονιά.

Για έγχρωμες εικόνες,η εξίσωση οπτικής ροής μπορεί να εφαρμοστεί για κάθε χρώμα ξεχωριστά, που μπορεί να περιορίσει το διάνυσμα μετατόπισης σε τρεις διαφορετικές κατευθύνσεις. Εντούτοις, στις περισσότερες περιπτώσεις ο περιορισμός εξομάλυνσης χρειάζεται για να πετύχουμε ικανοποιητικά αποτελέσματα. Γενικοί περιορισμοί εξομάλυνσης οδηγούν σε ανακρίβή εκτίμηση κίνησης στα όρια του κλεισίματος. Πιο εξελιγμένοι κατευθυνόμενοι περιορισμοί εξομάλυνσης επιτρέπουν ξαφνικές ασυνέχειες στον πεδίο της κίνησης

**Μέθοδοι κίνησης μπλοκ:** Υποθέτουμε πως η εικόνα αποτελείται από κινούμενα μπλοκ. Στην προσέγγιση συσχέτισης φάσης, ο γραμμικός όρος της διαφοράς φάσης Fourier μεταξύ δυο συνεχόμενων εικόνων καθορίζει την εκτίμηση κίνησης. Στο ταίριασμα του μπλοκ, ψάχνει για την θέση του καλύτερου μπλοκ σταθερού μεγέθους που ταιριάζει, στο επόμενο ή και προηγούμενο πλαίσιο, με βάση την απόσταση. Η βασική προσέγγιση και των δυο μεθόδων εφαρμόζεται μόνο σε ομογενή κίνηση, εντούτοις, το γενικοποιημένο ταίριασμα μπλοκ μπορεί να συνδιαστεί με άλλους χωρικούς μετασχηματισμούς.

**Αναδρομικοί μέθοδοι εικονστοιχείου:** Οι μέθοδοι αυτοί είναι προβλεπτικοί-διορθωτικοί εκτμητές μετατόπισης. Η πρόβλεψη μπορεί να θεωρηθεί ς η τιμή του υπολογισμού κίνησης στην προηγούμενη θέση του εικονοστοιχείου ή ως γραμικός συνδυασμός εκτιμήσεων σε μια γειτονιά του εικονοστοιχείου που εξετάζουμε. Η ενημέρωση γίνεται βάση την ελαχιστοποίηση της κλίσης του μετατοπισμένου πλαισίου διαφοράς στο εικονοστοιχείο αυτό. Το βήμα της πρόβλεψης θεωρείται γενικά ως ένας

ενδεχόμενος περιορισμό εξομάλυνσης. Η επέκταση αυτής της προσέγγισης σε εκτίμηση με μπλοκ, οδηγεί στις στρατηγικές εκτίμησης Wiener.

**Μέθοδοι Bayesian:** Αυτές οι μέθοδοι εκμεταλλεύονται τους περιορισμούς εξομάλυνσης που βασίζονται στις πιθανότητες, συνήθως στην μορφή ενός πεδίου Gibbs για να εκτιμήσουν την μετατόπιση. Το μειονέκτημα αυτών, είναι η ανάγκη για μεγάλες υπολογιστικές δυνατότητες.

### 5.9 Μέθοδοι βάση οπτικής ροής

Έστω $s_c(x1,x2,t)$ να δηλώνει την συνεχή κατανομή έντασης στο χώρο και στον χρόνο. Αν η ένταση παραμένει σταθερή κατά μήκος της τροχιάς της κίνησης, έχουμε

$$\frac{ds_c(x_1, x_2, t)}{d_t} = 0 \quad (5.1)$$

όπου $x_1$ και $x_2$ διαφέρει από t, σύμφωνα με την τροχιά της κίνησης. Η πιο πάνω εξίσωση είναι μια παράγωγος και δηλώνει τον ρυθμό αλλαγής της έντασης κατά μήκος της τροχιάς. Χρησιμοποιώντας τον αλυσιδωτό κανόνα της διαφοροποίησης, εκφράζεται ως:

$$\frac{\partial s_c(x,t)}{\partial x_1} v_1(x,t) + \frac{\partial s_c(x,t)}{\partial x_2} v_2(x,t) + \frac{\partial s_c(x,t)}{\partial t} = 0 \quad (5.2)$$

Όπου

**Equation 2**

v1(x,t) = dx1/dt και v2(x,t)=dx2/dt δείχνουν τα συστατικά του διανύσματος συντεταγμένων ταχύτητας σχετικά με τις συνεχής χωρικές συντεταγμένες. Η έκφραση αυτή είανι γνωστή και σαν εξίσωση οπτικής ροής ή σαν ο περιορισμός οπτικής ροής. Διαφορετικά μπορεί να εκφραστεί ως :

$$\left\langle \nabla s_c(x;t), v(x,t) \right\rangle + \partial s_c(x;t)/\partial t = 0 \quad (5.3)$$

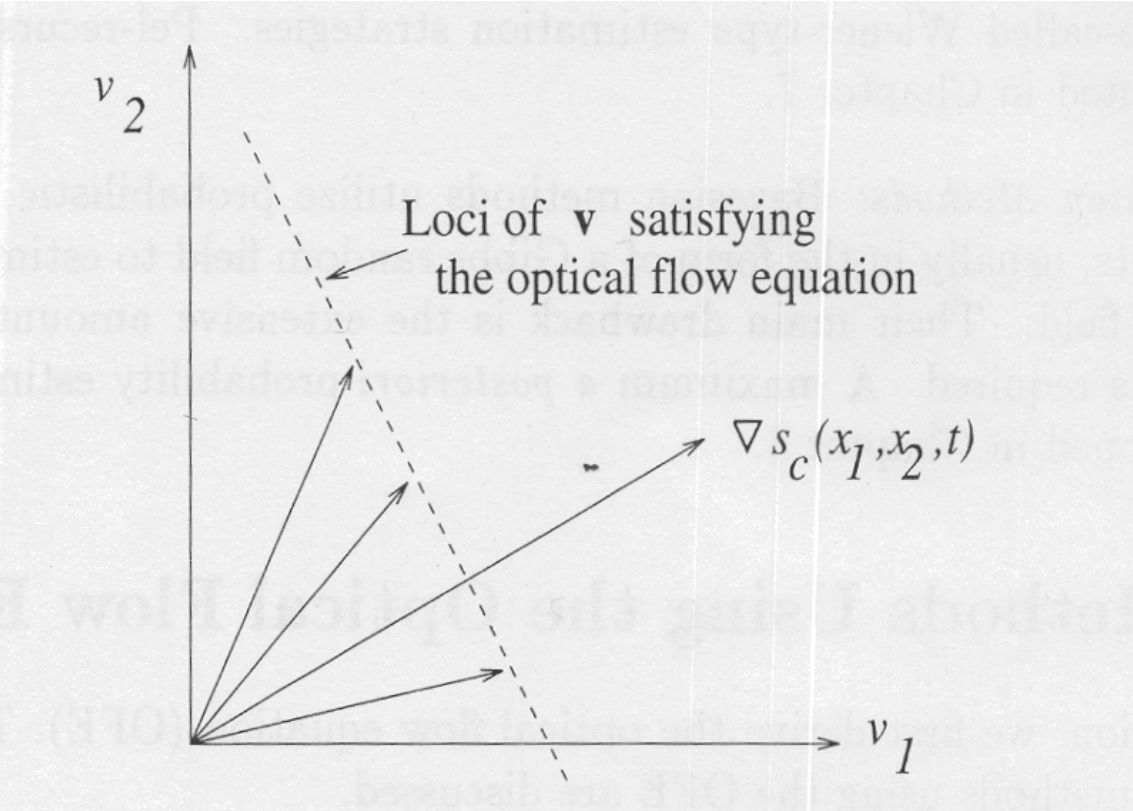
Όπου $\nabla s_c(x;t) = [(\partial s_c(x;t)/\partial x1) * (\partial s_c(x;t)/\partial x2)]^T$

Και <.,> δηλώνει το γινόμενο εσωτερικού διανύσματος.

Φυσικά, ο ορισμός της EOP δεν είναι αρκετός για να καθορίσει μοναδικά την 2-Δ ταχύτητα. Η EOP, οδηγεί σε μια φυσική εξίσωση με δυο αγνώστους, το v1(x,t) και v2(x,t), για κάθε θέση (x,t). Με βάση την εξίσωση πιο πάνω μπορούμε να δούμε ότι μπορούμε να υπολογίσουμε το διάνυσμα ροής που είναι στην κατεύθυνση του χωρικού διανύσματος

$$\frac{\nabla s_c(x;t)}{\left\| \nabla s_c(x;t) \right\|} \quad (5.4)$$

που καλείται κανονική ροή v⊥(x,t), επειδή η συνιστώσα που είναι ορθογώνιο στην χωρική κλίση της εικόνας χάνεται με το εσωτερικό γινόμενο.




**Σχήμα 5.7** Διανύσματα οπτικής ροής [8]

Αυτό φαίνεται και στην εικόνα, όπου όλα τα διανύσματα που οι κορυφές τους βρίσκονται πάνω στην διακεκομμένη γραμμή ικανοποιούν την σχέση EOP.

Έτσι η EOP εφαρμόζει τον περιορισμό ότι η συνιστώσα του διανύσματος ροής στην κατεύθυνση της χωρικής κλίσης της φωτεινότητας της εικόνας για κάθε θέση, που είναι συνεπής με το πρόβλημα του ανοίγματος. Παρατηρούμε ότι η EOP απαιτή ότι πρώτα η ένταση της εικόνας πρέπει να είναι διαφορήσιμη, και δεύτερο, η μερική παράγωγος της φωτεινότητας να υπάρχει.

### 5.10 Μέθοδος Horn και Schunck [2]

Οι Horn και Schunck ψάχνουν να βρουν το πεδίο της κίνησης που ικανοποιεί την EOP με την ελάχιστη απόκλιση των εικονοστοιχείων ανάμεσα στα διανύσματα ροής. Έστω,

$$E_{of}(v(x,t))= \left\langle \nabla s_c\ (x;t), v(x,t)\ \right\rangle + \partial s_c(x;t)/\partial t \quad (5.5)$$

να δηλώνει το λάθος στην EOP. Παρατηρούμε ότι η EOP, ικανοποιήται όταν $E_{of}(v(x,t))=0$. Στην παρουσία κλεισίματος και θορύβου, στοχεύουμε να ελαχιστοποιήσουμε το τετράγωνο του $E_{of}(v(x,t))$ για να εφαρμόσουμε τον περιορισμό της οπτικής ροής.

Η απόκλιση εικονοστοιχείου από εικοστοιχείο των διανυσμάτων ταχύτητας μπορεί να ποσοτικοποιηθεί από το άθροισμα των μεγέθων των τετραγώνων των χωρικών κλίσεων των συστατικών του διανύσματος ταχύτητας, που δίνεται από

$$E^2{}_s(v(x,t))= \left\|\nabla v1(x,t)\right\|^2 + \left\|\nabla v2(x,t)\right\|^2 = (\frac{\partial v1}{\partial x1})^2 + (\frac{\partial v1}{\partial x2})^2 + (\frac{\partial v2}{\partial x1})^2 + (\frac{\partial v2}{\partial x2})^2 \quad (5.6)$$

όπου υποθέτουμε ότι οι χωρικές και χρονικές συντεταγμένες είναι συνεχές μεταβλητές. Μπορεί εύκολα να αποδειχθεί ότι όσο πιο εξομαλυμένο είναι το πεδίο της ταχύτητας, τόσο πιο μικρό είναι το $E^2{}_s(v(x,t))$.

Ακολούθως η μέθοδος του Horn και Schunck ελαχιστοποιεί ένα σταθμικό άθροισμα του λάθους στην EOP και ένα μέτρο της απόκλισης εικονοστοιχείου από εικονοστοιχείο του πεδίου ταχύτητας:

$$\min_{v(x,t)} \int_A (E_{of}(v) + a^2 E_s{}^2(v))\, dx \quad (5.7)$$

για να υπολογίσουν το διάνυσμα της ταχύτητας στο σημείο x, όπου το A δείχνει την συνεχή στήριξη της εικόνας. Η παράμετρος $\alpha^2$, ελέγχει την δύναμη του περιορισμού εξομάλυνσης. Μεγάλες τιμές του $\alpha^2$ αυξάνουν την επιρροή του περιορισμού.

Η ελαχιστοποίηση της συνάρτησης ελαχίστου με χρήση άλγεβρικη ανάλυση μεταβλητών, απαιτεί την λύση δυο εξισώσεων.

$$\begin{aligned}(\frac{\partial s_c}{\partial x1})^2\,\hat{v}1(x,t)+\frac{\partial s_c}{\partial x1}\frac{\partial s_c}{\partial x2}\hat{v}2(x,t)&=a^2\nabla^2\,\hat{v}1(x,t)-\frac{\partial s_c}{\partial x1}\frac{\partial s_c}{\partial t}\\ \frac{\partial s_c}{\partial x1}\frac{\partial s_c}{\partial x2}\hat{v}1(x,t)+(\frac{\partial s_c}{\partial x2})^2\,\hat{v}2(x,t)&=a^2\nabla^2\,\hat{v}2(x,t)-\frac{\partial s_c}{\partial x2}\frac{\partial s_c}{\partial t}\end{aligned} \qquad (5.8)$$

ταυτόχρονα, όπου το $\nabla^2$ δηλώνει την Laplacian και το $\hat{.}$ δηλώνει την εκτίμηση της εκάστοτε ποσότητας. Στην υλοποίηση των Horn και Schunck, οι παράμετροι Laplace των συστατικών της ταχύτητας, έχουν υπολογιστεί από FIR φίλτρα που επιτρέπουν υψηλές συχνότητες για να φτάσουν στην επανάληψη Gauss-Seidel:

$$\hat{v}1^{(n+1)}(x,t)=\hat{v}1^{-(n)}(x,t)-\frac{\partial s_c}{\partial x1}\frac{\frac{\partial s_c}{\partial x1}v1^{-(n)}(x,t)+\frac{\partial s_c}{\partial x2}v2^{-(n)}(x,t)+\frac{\partial s_c}{\partial t}}{a^2+(\frac{\partial s_c}{\partial x1})^2+(\frac{\partial s_c}{\partial x2})^2}$$

$$\hat{v}2^{(n+1)}(x,t)=\hat{v}2^{-(n)}(x,t)-\frac{\partial s_c}{\partial x2}\frac{\frac{\partial s_c}{\partial x1}v1^{-(n)}(x,t)+\frac{\partial s_c}{\partial x2}v2^{-(n)}(x,t)+\frac{\partial s_c}{\partial t}}{a^2+(\frac{\partial s_c}{\partial x1})^2+(\frac{\partial s_c}{\partial x2})^2} \qquad (5.9)$$

Όπου n είναι ο αριθμός επανάληψης, η οριζόντια γραμμή πάνω από τα σύμβολα δεικνύει το σταθμικό τοπικό μέσο όρο, χωρίς το εικονοστοιχείο που είμαστε, και όλες οι μερικές υπολογίζονται στο σημείο (x,t).

Οι αρχικές εκτιμήσεις των ταχυτήτων $v1^{(0)}(x,t)$ και $v2^{(0)}(x,t)$ συνήθως παίρνουν τιμή μηδέν. Η πιο πάνω διατύπωση υποθέτει μια συνεχή χωρο-χρονική κατανομή έντασης. Όλα τα χρονικά και χωρικά διανύσματα πρέπει να υπολογιστούν αριθμητικά από τις εικόνες του δείγματος.

## 5.11 Υπολογισμός διανυσμάτων χρησιμοποιώντας πεπερασμένες διαφορές

Μια προσέγγιση για να υπολογίσουμε τις μερικές παραγώγους από μια διακριτή εικόνα s(n1,n2,k) είναι να τις προσεγγίσουμε με τις αντίστοιχες ευθείες ή αντίστροφες πεπερασμένες διαφορές. Για να έχουμε εύρωστες εκτιμήσεις για τις μερικές, μπορούμε να υπολογίσουμε τον μέσο όρο από την ευθείες και αντίστροφες πεπερασμένες διαφορές , τις μέσες διαφορές. Επιπρόσθετα μπορούμε να υπολογίσουμε τον τοπικό μέσο των μέσων διαφορών για να εξαλείψουμε τις παρενέργειες του θορύβου που παρατηρείται. Οι Horn και Schunck πρότειναν τον μέσο όρο τεσσάρων πεπερασμένων διαφορών για να πετύχουν:

$$\frac{\partial s_c(x1,x2,t)}{\partial x1} \approx \frac{1}{4}\{s(n1+1,n2,k)-s(n1,n2,k)+s(n1+1,n2+1,k)-s(n1,n2+1,k)+s(n1+1,n2$$
$$-s(n1,n2,k+1)+s(n1+1,n2+1,k+1)-s(n1,n2+1,k+1)\}$$

$$\frac{\partial s_c(x1,x2,t)}{\partial x2} \approx \frac{1}{4}\{s(n1,n2+1,k)-s(n1,n2,k)+s(n1+1,n2+1,k)-s(n1+1,n2,k)+s(n1,n2+$$
$$-s(n1,n2,k+1)+s(n1+1,n2+1,k+1)-s(n1+1,n2,k+1)\}$$

$$\frac{\partial s_c(x1,x2,t)}{\partial t} \approx \frac{1}{4}\{s(n1,n2,k+1)-s(n1,n2,k)+s(n1+1,n2,k+1)-s(n1+1,n2,k)+s(n1,n2+$$
$$-s(n1,n2+1,k)+s(n1+1,n2+1,k+1)-s(n1+1,n2+1,k)\}$$

(5.10)

Η χωρική και χρονική εξομάλυνση του βίντεο με Gaussian κανονική κατανομή συνήθως βοηθά την εκτίμηση της κλίσης.

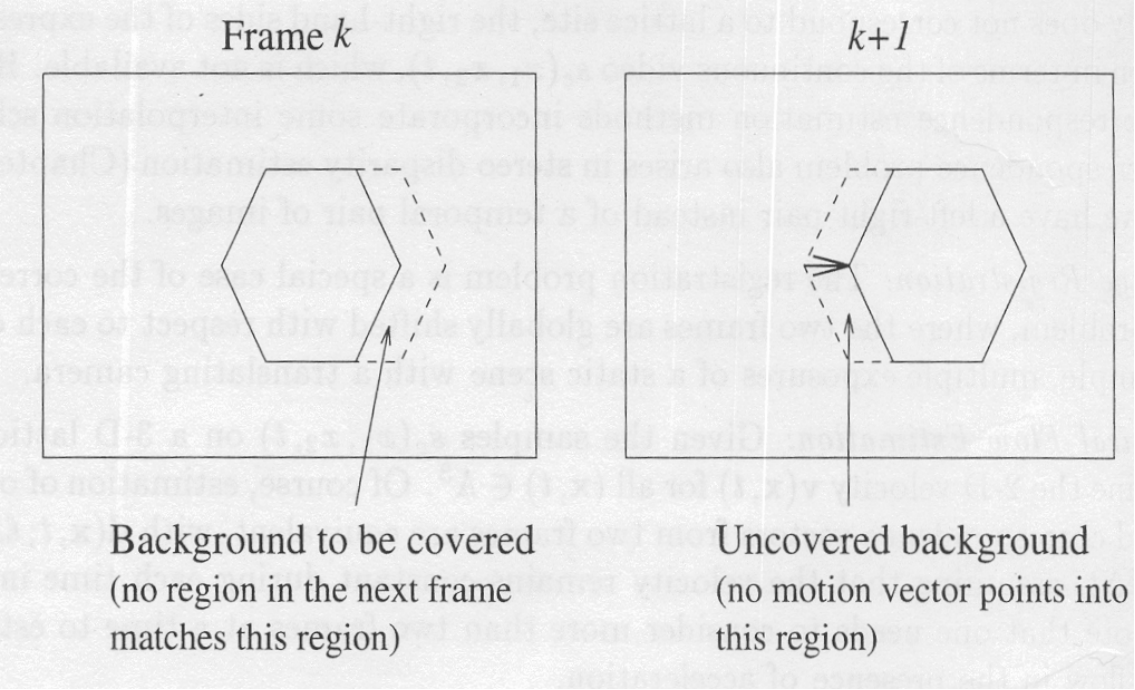
## 5.12 Προσαρμοστικές μέθοδοι

Η μέθοδος των Horn και Schunck εφαρμόζει τους περιορισμούς οπτικής ροής και εξομάλυνσης γενικά σε όλη την εικόνα, ή σε ένα παράθυρο εκτίμησης κίνησης. Αυτό έχει δυο ανεπιθύμητες παρενέργειες:

1. Ο περιορισμός της εξομάλυνσης δεν ευσταθεί στην κατεύθυνση κάθετα σε ένα σύνορο κλεισίματος. Έτσι, ένας γενικός περιορισμός εξομάλυνσης θολώνει τις γωνιές της κίνησης. Για παράδειγμα, αν ένα αντικείμενο κινείται μπροστά παό ένα σταθερό φόντο, τότε υπάρχει μια ξαφνική αλλαγή στο πεδίο της ταχύτητας του συνόρου του αντικειμένου. Οι ακμές της κίνησης μπορούν να διατηρηθούν αν εφαρμόσουμε τον περιορισμό εξομάλυνσης μόνο στις κατευθύνσεις εκεί που η ένταση των εικονοστοιχείων δεν αλλάζουν σημαντικά.

2.Η μη προσαρμοστική μέθοδος εφαρμόζει τον περιορισμό της οπτικής ροής στις περιοχές με άνοιγμα, όπου δεν έπρεπε να εφαρμοστεί. Αυτό μπορεί να επιτευχθεί με προσαρμοστική μεταβολή του ‘α’ για να ελέγξουμε την σχετική δύναμη της οπτικής ροής και των περιορισμών εξομάλυνσης.

Για παράδειγμα, στις περιοχές ανοίγματος, όπως στην εικόνα πιο κάτω, ο περιορισμός οπτικής ροής μπορεί να αφαιρεθεί πλήρως, και ο περιορισμός εξομάλυνσης να εφαρμοστεί στο μέγιστο.




**Σχήμα 5.8** Παράδειγμα προβλήματος του κλεισίματος [8]

## 5.13 Μέθοδος Horn and Schunck-Περιγραφή Αλγορίθμου

Για να καταλάβουμε καλύτερα τις παραμέτρους που αλλάζουν στα πειράματα της υλοποίησης των Horn και Schunck, κάποιους μαθηματικούς τύπους θα τους ορίσουμε διαφορετικά παρά στο πως είναι ορισμένα πιο πάνω.

Η επεξήγηση του αλγορίθμου θα γίνει με τον τρόπο που είναι χωρισμένη η δημοσίευση των Horn and Schunck Determining Optical Flow.

Η οπτική ροή δεν μπορεί να υπολογιστεί τοπικά, καθώς ενώ μπορεί να γίνει ανεξάρτητη εκτίμηης για μια ακολουθία εικόνων για ένα σημείο, δεν μπορεί να γίνει αυτό όμως για την ταχύτηττα γιατί έχει δυο συστατικά. Για να γίνει αυτό χρειάζεται ένα νέος περιορισμός. Η μέθοδος αυτή για να υπολογίσει το σχέδιο της οπτικής ροής υποθέτει ότι η ταχύτητα του σχεδίου φωτεινότητας ποικίλει παντού στην εικόνα. Η υλοποίηση αυτή είναι επαναληπτική και υπολογίζει σωστά την οπτική ροή για ακολουθία εικόνων.

### 5.13.1 Εισαγωγή

Η οπτική ροή είναι η κατανομή ταχυτήτων κίνησης σχεδίων από τιμές φωτεινότητας σε μια εικόνα. Η οπτική ροή μπορεί να προκύψει από την σχετική κίνηση μεταξύ αντικειμένων και παρατηρητή. Συνεπώς, η οπτική ροή μπορεί να δώσει σημαντικές πληροφορίες σχετικά με την χωρική διαρύθμιση των αντικειμένων που παρατηρούνται και τον ρυθμό αλλαγής της διαρύθμισης. Οι ασυνέχειες στην οπτική ροή μπορούν να βοηθήσουν στον χωρισμό των εικόνων σε περιοχές που αναφέρονται στα ξεχωριστά αντικείμενα.

Η οπτική ροή δεν μπορεί να υπολογιστεί για ένα σημείο μιας εικόνας ξεχωριστά από τα γειτονική εικονοστοιχεία, χωρίς επιπρόσθετους περιορισμούς, επειδή το πεδίο της ταχύτητας για κάθε σημείο έχει δυο συστατικά ενώ η αλλαγή στις τιμές φωτεινότητας για κάθε εικονοστοιχείο στην επιφάνεια της εικόνας λόγω κίνησης έχει μόνο ένα περιορισμό.

### 5.13.2 Σχέση οπτικής ροής και ταχύτητας των αντικειμένων

Η σχέση μεταξύ τις οπτικής ροής και των ταχυτήτων των αντικειμένων στον τρισδιάστατο κόσμο δεν είναι εμφανής. Αντιλαμβανόμαστε την κίνηση όταν μια εικόνα που αλλάζει προβάλλεται σε μια σταθερή οθόνη. Αντίστροφα, ένα κινούμενο αντικείμενο μπορεί να είναι η αιτία για μια σταθερό μοτίβο φωτεινότητας.

Προς αποφυγή μεταβολών στην φωτεινότητα λόγω μεταβολών σκίασης, αρχικά τέθηκε η υπόθεση ότι η επιφάνεια είναι επίπεδη. Επίσης, ότι η φωτεινότητα που προκύπτει είναι ομογενής σε όλη την επιφάνεια. Η φωτεινότητα σε ένα σημείο, είναι ανάλογη της αντανάκλασης της επιφάνειας στο σχετικό σημείο του αντικειμένου. Επισης, υπάρχει η υπόθεση ότι αρχικά, η ανάκλαση διαφέρει ομαλά και δεν έχει χωρικές ασυνέχειες. Ο τελευταίος περιορισμός, μας διασφαλίζει ότι η φωτεινότητα της εικόνας είναι διαφορήσιμη. Εξαιρούμε περιπτώσεις όπου τα αντικείμενα κλείνουν το ένα το άλλο.

### 5.13.3 Περιορισμοί-Εξίσωση οπτικής ροής Horn and Schunck

Η εξίσωση που θα προκύψει, σχετίζει την αλλαγή της φωτεινότητας σε μια εικόνα για ένα σημείο και την κίνησης του μοτίβου φωτεινότητας. Έστω, η φωτεινότητα σε ένα σημείο (x,y) μιας εικόνας στον χρόνο t, να δίνεται από την έκφραση E(x,y,t). Έστω ότι το σχέδιο κινείται. Η φωτεινότητα σε ένα σγμείο είναι σταθερή, άρα:

$$\frac{dE}{dt} = 0 \quad (5.11)$$

Προκύπτει ότι:

$$\frac{\partial E}{\partial t}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t} = 0 \qquad (5.12)$$

Αν θεωρήσουμε ότι

$$u = \frac{dx}{dt} \; \kappa\alpha\iota \; v = \frac{dy}{dt} \qquad (5.13)$$

Τότε προκύπτει μια γραμμική εξίσωση με δυο αγνώστους u και v τω.

$E_x u + E_y u + E_t = 0$ (5.14)

Όπου τα E σχετικά με το x,y και t αναφέρονται στα μερικά ολοκληρώματα της φωτεινότητας της εικόνας σχετικά με τα x,y,t αντίστοιχα. Ο περιορισμός για την τοπική ροή εκφράζεται ως εξής:

$(E_x, E_x) \cdot (u,v) = - E_t$ (5.15)

Τα συστατικά για την κίνηση προς την κατεύθυνση του διανύσματος της φωτεινότητας δίνεται από

$$\frac{E_1}{\sqrt{E_x^2 + E_y^2}} \qquad (5.16)$$

### 5.13.4 Περιορισμός εξομάλυνσης

Εάν κάθε σημείο του σχεδίου φωτεινότητας μπορεί να κινετείται ανεξάρτητα, τότε υπάρχει μικρή πιθανότητα να υπολογιστούν οι ταχύτητες. Συνήθως μελετούμε αδιαφανή αντικείμενα πεπερασμένου μεγέθους που κάνουν σταθερή κίνηση ή παραμόρφωση. Σε αυτή την περίπτωση γειτονικών σημείων σε αντικείμενα με παρόμοιες ταχύτητες και το πεδίο της ταχύτητας των μοτίβων φωτεινότητας στην εικόνα διαφέρει ομαλά σχεδόν παντού. Ασυνέχειε στην ροή μπορούν να αναμένονται όπου ένα αντικείμενο παρεμποδίζει ένα άλλο. Ένας αλγόριθμος βασισμένος σε περιορισμό εξομάλυνσης, αναμένεται να έχει δυσκολίες στις άκριες.

Ένας επιπρόσθετος περιορισμός για το πρόβλημα αυτό, είναι να περιοριστεί το τετράγωνο του μεγέθους του διανύσματος της οπτικής ροής:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \quad και \quad \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}$$

### 5.13.5 Κβαντοποίηση και Θόρυβος

Έστω η φωτεινότητα $E_{i,j,k}$ στην διασταύρωση της i σειρά, j στήλη και k εικόνα. Στην ιδανική περίπτωση κάθε μέτρηση θα έπρεπε να ήταν ένας μέσος όρος γύρω από

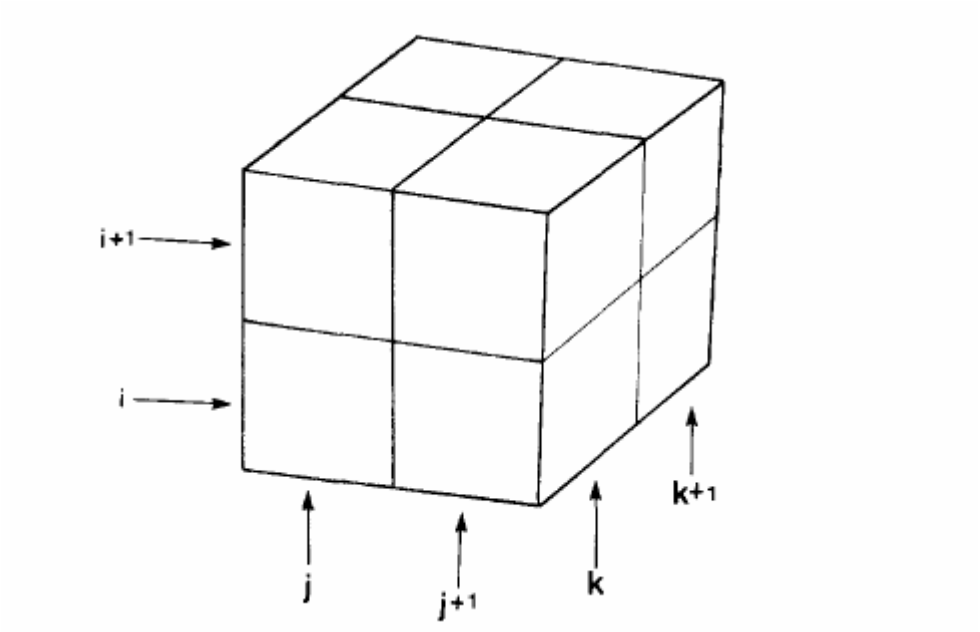
την περιοχή ενός κελιού του εικονοστοιχείου και κατά μήκος του διαστήματος του χρόνου.

Οι υπολογισμοί των $E_x$, $E_y$, $E_t$, πρέπει να αναφέρονται στο ίδιο σημείο την ίδια χρονική στιγμή. Ο υπολογισμός του αλγορίθμου των Horn και Schunck γίνεται για ένα σημείο στο κέντρο ενός κύβου διαμορφωμένου από οκτώ μετρήσεις. Η σχέση στο χρόνο και χώρο μεταξύ των μετρήσεν φαίνεται στο Σχ. Κάθε ένας υπολογισμός είναι ο μέσος όρος από τέσσερις διαφορέςαπό γειτονικές μετρήσεις στον κύβο:

$$E_x \approx \frac{1}{4}\{E_{i,j+1,k} - E_{i,j,k} + E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i+1,j,k+1}\}$$

$$E_y \approx \frac{1}{4}\{E_{i+1,j,k} - E_{i,j,k} + E_{i,j+1,k} + E_{i+1,j,k} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i,j+1,k+1}\} \qquad (5.17)$$

$$E_t \approx \frac{1}{4}\{E_{i,j,k+1} - E_{i,j,k} + E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j+1,k} + E_{i+1,j+1,k+1} - E_{i+1,j+1,k}\}$$




**Σχ 5.10.** Οι τρεις μερικές παράγωγοι των φωτεινοτήτων. Το j αντιπροσωπεύει την κατεύθυνση του x, το i το y και το k τον χρόνο.[4]

### 5.13.6 Υπολογισμός τιμών Laplace για τα τις ταχύτητες ροής

Για να προσεγγίσουμε τις τιμές Lapace για τα u,v ακολουθούμε την πιο κάτω μορφή:

$\nabla^2 u \approx k(\bar{u}_{i,j,k} - u_{i,k,k})$ $\kappa\alpha\iota$ $\nabla^2 v \approx k(\bar{v}_{i,j,k} - v_{i,k,k})$

Όπου οι τοπικοί μέσοι όροι $\bar{u}$ $\kappa\alpha\iota$ $\bar{v}$ ορίζονται ως ακολούθως:

$$\bar{u}_{i,j,k} = \frac{1}{6}\{u_{i-1,j,k} + u_{i,j+1,k} + u_{i,j-1,k}\} + \frac{1}{12}\{u_{i-1,j-1,k} + u_{i-1,j+1,k} + u_{i+1,j+1,k} + u_{i+1,j-1,k}\},$$

$$\bar{v}_{i,j,k} = \frac{1}{6}\{v_{i-1,j,k} + v_{i,j+1,k} + v_{i,j-1,k}\} + \frac{1}{12}\{v_{i-1,j-1,k} + v_{i-1,j+1,k} + v_{i+1,j+1,k} + v_{i+1,j-1,k}\}$$

(5.18)

Η ανάθεση των βαρών των γειτονικών εικονοστοιχείων φαίνεται πιο κάτω:

| | | |
|---|---|---|
| 1/12 | 1/6 | 1/12 |
| 1/6 | -1 | 1/6 |
| 1/12 | 1/6 | 1/12 |

**Σχήμα 5.11** Οι τιμές Laplace, υπολογίζονται με τη αφαλιρεση της τιμής σε ένα εικονοστοιχείο από ένα ζυγισμένο μέσο όρο τιμών στα γειτωνικά εικονοστοιχεία.[4]

### 5.13.7 Περιορισμός λάθους

Για την ελαχιστοποίηση του λάθους στην εξίσωση για την αλλαγή της φωτεινότητας της εικόνας

$E_b = E_x u + E_y v + E_t$

και το μέτρο της απώλειας από την εξομάλυνση στην ροή της ταχύτητας

$$E_c^2 = (\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2 \quad (5.19)$$

Πρακτικά υπάρχει θόρυβος στις μετρήσεις φωτεινότητας κατά τον κβαντισμό. Πρέπει να ορίσουμε το βάρος για το μέγεθος του λάθους το ποίο είναι ανάλογο του θορύβου. Ο παράγοντας λάθους ορίζεται ως «$\alpha^2$».

Το ολικό λάθος δίνεται από:

$$E^2 = \iint (a^2 E_c^2 + E_b^2) dxdy \qquad (5.20)$$

Η ελαχιστοποίηση γίνεται όταν βρούμε κατάλληλες τιμές για την ταχύτητα της οπτικής ροής (u,v).

$$\begin{aligned} E_x^2 u + E_x E_y v &= \alpha^2 \nabla^2 u - E_x E_{t,} \\ E_y^2 v + E_x E_y v &= \alpha^2 \nabla^2 v - E_x E_{t,} \end{aligned} \qquad (5.21)$$

Χρησιμοποιώντας την προσέγγιση Laplace προκύπτει :

$$\begin{aligned} (\alpha^2 + E_x^2) u + E_x E_y v &= (a^2 \overline{u} - E_x E_t) \\ (\alpha^2 + E_y^2) v + E_x E_y u &= (a^2 \overline{v} - E_y E_t) \end{aligned} \qquad (5.22)$$

Η ορίζουσα του πίνακα συντελεστών είναι $a^2(a^2 + E_x^2 + E_y^2)$.

Λύοντας ως προς u και v:

$$\begin{aligned} (a^2 + E_x^2 + E_y^2) u &= +(a^2 + E_y^2)\overline{u} - E_x E_y \overline{v} - E_x E_t \\ (a^2 + E_x^2 + E_y^2) v &= -E_x E_y \overline{u} + (a^2 + E_x^2)\overline{v} - E_y E_t \end{aligned} \qquad (5.23)$$

## 5.14 Περιβάλλον εργασίας για τις δοκιμές

Το τελικό περιβάλλον εργασίας που χρησιμοποιήσαμε ήταν το περιβάλλον LINUX εκδόσεις MANDRAKE και FEDORA CORE 3. Επίσης, με την εγκατάτασταση μιας πρόσφατης έκδοσης cygwin, το πρόγραμμα mplayer μπορεί να δουλέψει αν κάνουμε αντιγραφή των εκτελέσιμων αρχείων στον κατάλογο **C:\cygwin\usr\local\bin.** Αυτό μας επιτρέπει μέσω της γραμμής εντολών να τρέξουμε το πρόγραμμα mplayer-mencoder για με τις εντολές και παραμέτρους όπως δίνονται μέχρι τώρα.

## 5.15 Χρήσιμο λογισμικό

Για την διεκπεραίωση της διπλωματικής εργασίας, έγινε χρήση λογισμικού το οποίο φάνηκε ιδιαίτερα χρήσιμο.

Το Xnview, το οποίο χρησιμοποιείται για την μετατροπή εικόνων, παρουσίαση εικόνων και επεξεργασία.

Mplayer/mencoder για LINUX και Microsoft Windows. Για επεξεργασία βίντεο.

Η βιβλιοθήκη ImageMagick και την χρήση των εντολών convert και identify. Η πρώτη εντολή μετατρέπει ένα τύπο εικόνας σε ένα άλλο και η δεύτερη εντολή μας δίνει όλες τις πληροφορίες που μπορεί να αφορούν μια εικόνα.

Όλο το λογισμικό που χρησιμοποιήθηκε και αναφέρεται πιο πάνω, είναι δωρέαν.

# Κεφάλαιο 6

## Αποτελέσματα Αλγορίθμου οπτικής ροής Horn και Schunck



### 6.1 Εισαγωγή

Για την εκτίμηση κίνησης χρησιμοποιήσαμε όπως έχει αναφερθεί προηγουμένως τον αλγόριθμο των Horn and Schunck 1981, τροποποιημένο από τον John Barron το 1992.

Για να επαληθεύσουμε την ορθότητα του αλγορίθμου θα παρουσιάσουμε εδώ το πως ο αλγόριθμος δουλεύει προγραμματιστικά και τι αποτελέσματα δίνει. Ακολούθως θα τρέξουμε κάποια δοκιμαστικά με πλάκα με τεχνητή κίνηση και τέλος θα δούμε το πως τρέχει ο αλγόριθμος με πραγματικά δεδομένα.

Δοθείσας μιας ακολουθίας εικόνων, υπολογίζουμε το 2-Δ πεδίο ταχύτητας ή την οπτική ροή(φαινομενική κίνηση). Εάν στις εικόνες αυτές, υπάρχει μικρή χρονική διαφορά, τότε αυτό μας επιτρέπει να χρησιμοποιήσουμε τεχνικές που χρησιμοποιούν διαφορικές εξισώσεις, γνωστές και ως μέθοδοι οπτικής ροής.

Το διάνυσμα οπτικής ροής v(x,y), έχει δυο συστατικά. Το v και το u, που περιγράφουν την κίνηση ενός σημείο ως προς την κατεύθυνση x και y στην εικόνα.

Οι Horn και Schunck συνδύασαν τον περιορισμό των κλίσεων με ένα όρο γενικής εξομάλυνσης για να περιορίσουν το υπολογισμένο πεδίο της ταχύτητας v(x,t) = (u(x,t), v(x,t)), ελαχιστοποιώντας το:

$$\int_D (\nabla I.v + I_t)^2 + \lambda^2(\|\nabla u\|_2^2 + \|\nabla v\|_2^2)dx \quad (6.1)$$

που ορίζεται σε ένα πεδίο ορισμού, όπου το μέγεθος του **λ** αντανακλά την επιρροή του ορίσματος της εξομάλυνσης. Στην δημοσίευση Performance of Optical Flow Techniques J.L. Barron, D.J.Fleet and S.S.Beauchemin έχει χρησιμοποιηθεί τιμή λ=0.5

ενώ αρχικά η προτινόμενη τιμή ήταν 100. Επαναληπτικές εξισώσεις χρησιμοποιούνται για να περιορίσουν και να εξασφαλίσουν την ταχύτητα της εικόνας:

$$u^{k+1} = u^{k-1} - \frac{I_x[I_x u^{-k} + I_y v^{-k} + I_t]}{a^2 + I_x^2 + I_y^2} \quad (6.2)$$

$$v^{k+1} = v^{k-1} - \frac{I_y[I_x u^{-k} + I_y v^{-k} + I_t]}{a^2 + I_x^2 + I_y^2}$$

όπου το k δηλώνει τον αριθμό επανάληψης, $u^0$, $v^0$ δηλώνουν τους υπολογισμούς της αρχικής ταχύτητας που ορίζονται μηδέν, και $u^{-\kappa}$, $v^{-\kappa}$ δηλωνουν τους γειτονικούς μέσους όρους των $u^\kappa$, $v^\kappa$.Το μέγιστο των επαναλήψεων είναι εκατόν.

Η αρχική υλοποίηση χρησιμοποιούσε πρώτου βαθμού διαφορές γιαν α υπολογίσουν τις παραγώγους της έντασης. Επειδή όμως αυτό ήταν πηγή λάθους, προστέθηκε στην μέθοδο προ-εξομάλυνση και 4-σημείων κεντιρκές διαφορές για διαφοροποίηση με συντελεστές της μάσκας:

$$\frac{1}{12}(-1, 8, 0, -8, 1). \quad (6.3)$$

Στην υλοποίηση χρησιμοποιήθηκε χωροχρονικό προ-φιλτράρισμα Gaussian, με τυπική απόκλιση 1,5 εικονοστοιχεία σε χώρο και 1,5 εικόνες στο χρόνο, δειγματολειμένο σε τρεις τυπικές αποκλίσεις.

Το πιο βασικό σημείο για τον αλγόριθμο των Horn and Schunck, είναι πως επιπρόσθετα του περιορισμού οπτικής ροής, απαιτείται το υπολογισμένο το διάνυσμα της ταχύτητας, να διαφέρει ομαλά στην επιφάνεια της εικόνας.
Η ομαλότητα αυτή ορίζεται με βάση το μέγεθος του χωρικού διανύσματος του **v.**

## 6.2 Επαλήθευση αλγορίθμου

Πρέπει να σημειώσουμε αρχικά ότι πρέπει πριν τρέξει ο αλγόριθμος να επιβεβαιώσουμε κάποιες απλές παραμέτρους που χρησιμοποιά η υλοποίηση.

Πρώτα, το μέγεθος των εικόνων πρέπει να συμφωνεί με τις διαστάσεις PIC_X και PIC_Y που είναι σταθερές στην υλοποίηση σε C. Επίσης, αναλόγως των τιμών του

sigma θα χρειαστούμε ανάλογο αριθμό εικόνων. Πχ για τιμές 0.7 και κάτω χρειαζόμαστε μόνο 9 πλαίσια ενώ για πιο μεγάλες τιμές χρειαζόμαστε 15.

Για να μεταγλωττίσουμε με μια γραμμή το πρόγραμμα χωρίς την χρήση του Makefile με δύο εντολές που διατίθεται από τον John Barron, μπορούμε να χρησιμοποιήσουμε την εντολή:

```
cc –g horn.c –lm –o horn
```

**Ακολουθία κινούμενου δένδρου**

Ο αλγόριθμος επεξεργάζεται την ακολουθία εικόνων του κινούμενου δένδρου με την ακόλουθη εντολή:

```
$ ./horn newbinarytreet. 0.5 1.5 20 100 testdata/default_test_data/TREE_DATA/trans result
-B 150 150 -C testdata/default_test_data/CORRECT_FLOWS/correct_trans20 -MH -T 5.0
```

Η εντολή αυτή δίνεται από το αρχείο runtreet.

Οι εικόνες αυτές είναι τεχνητές και δείχνουν μια συνθετική κίνηση της κάμερας σε σχέση με ένα σταθερό δένδρο.

Όταν τρέξουμε την υλοποίηση του John and Barron μας δίνει σαν έξοδο ένα αρχείο με επέκταση ".F". Αυτο το αρχείο περιέχει τα πεδία ροής. Μέσω του προγράμματος που είναι μαζί στο πακέτο χρήσης της υλοποιησης **psflow** μπορούμε να μετατρέψουμε τα πεδία ροής σε postscript αρχείο για να τα δούμε και να τα τυπώσουμε.

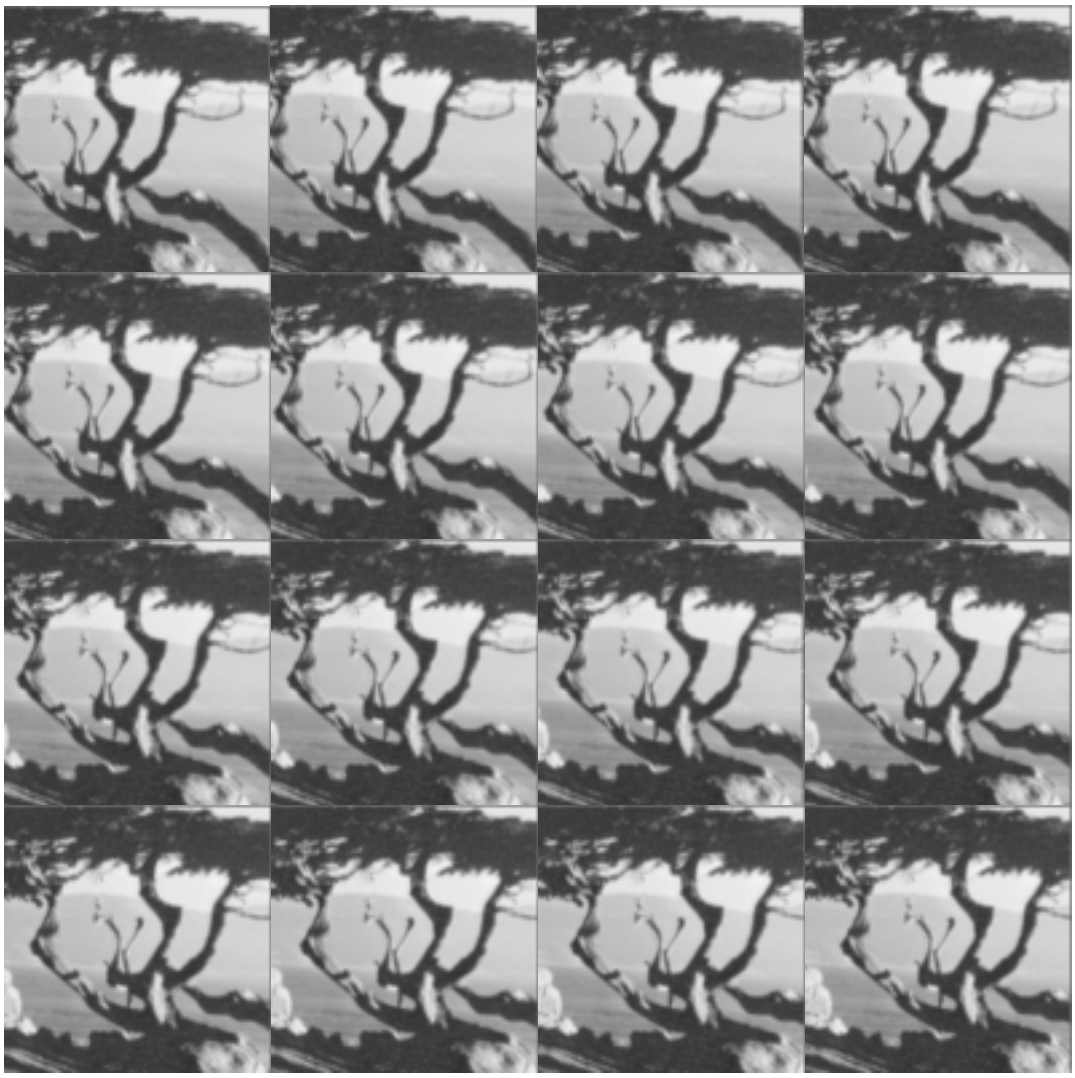
Το λογισμικό psflow χρησιμοποιείται ως ακολούθως:

```
./psflow -s2 -m1.5 -f horn.modified.imageh_l.F image.ps
```

Ο πιο πάνω τρόπος χρήσης του psflow, είναι όπως υποδυκνύεται από τους συγγραφείς της δημοσίευσης. Αν δεν θα γίνει χρήση σε latex τότε δεν χρησιμοποιούμε την παράμετρο –f και η εντολή γίνεται

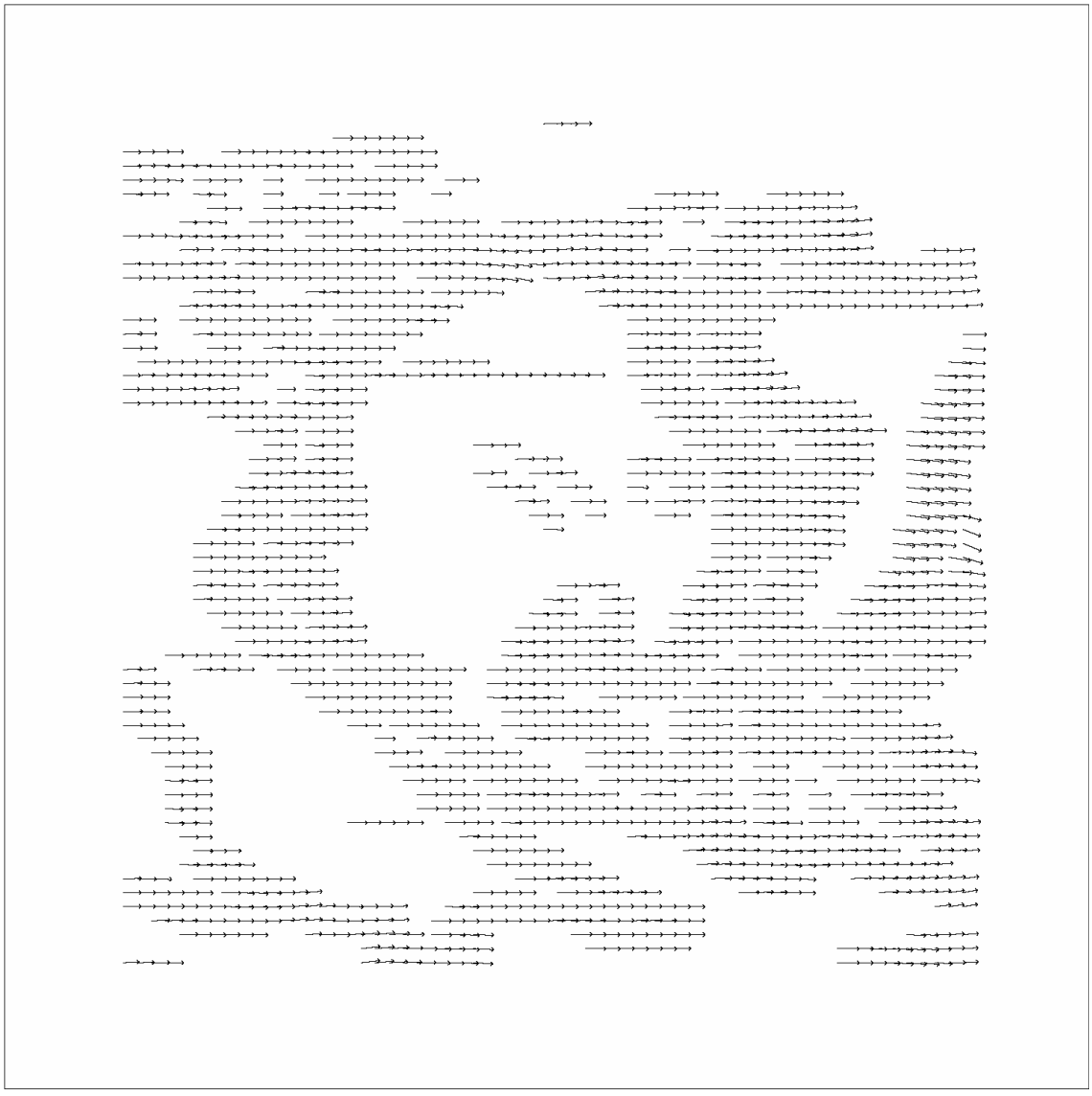
```
./psflow -s2 -m1.5 –n horn.modified.imageh_l.F image.ps
```

Παρακάτω παρουσιάζονται οι εικόνες της ακολουθίας του κινούμενου δέντρου.

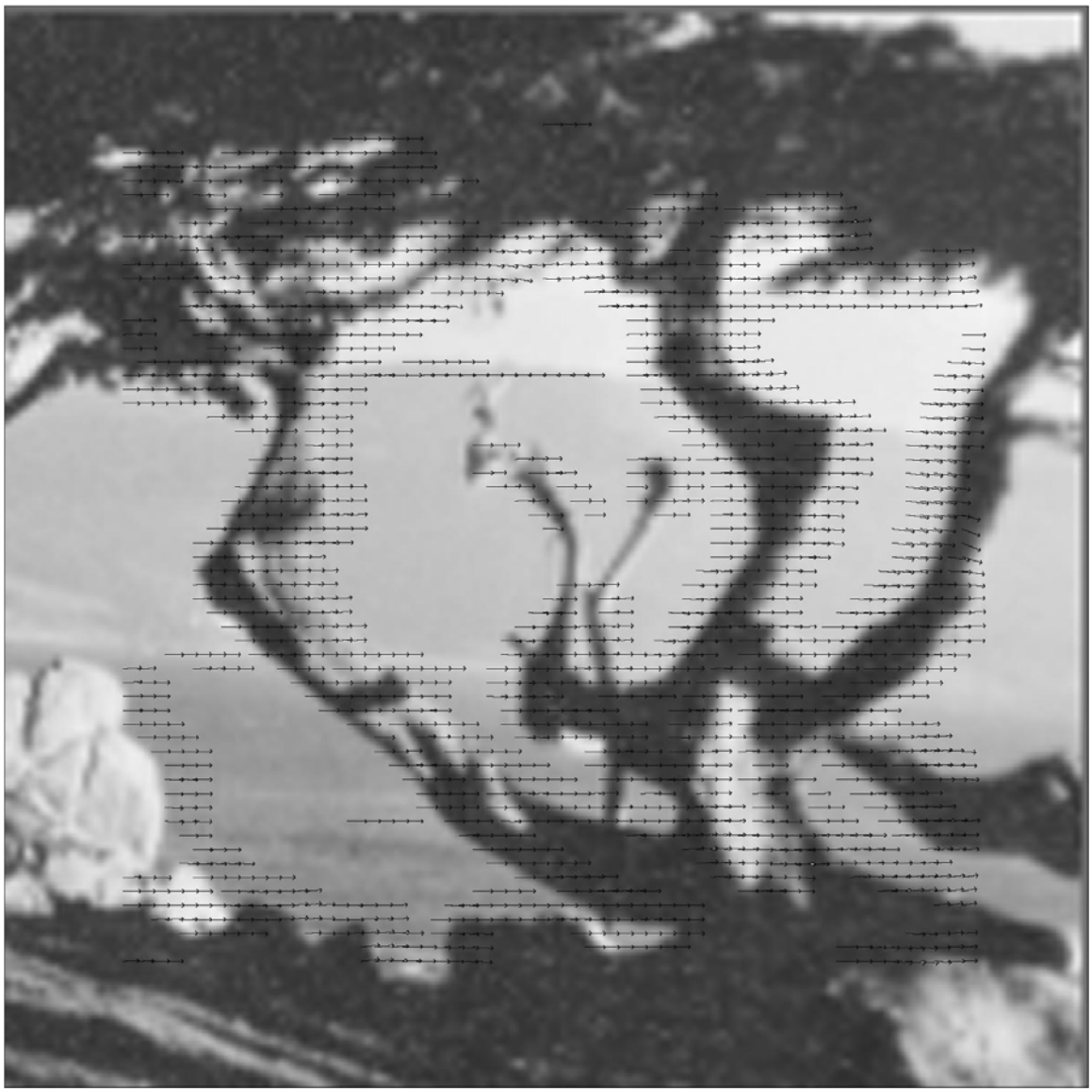


**Σχήμα 6.1** Ακολουθία κινούμενου δένδρου. Η κάμερα κινείται από δεξιά προς αριστερά, επομένως η φαινομενική κινηση του δένδρου είναι από αριστερά προ δεξιά.

Το postscript αρχείο δίνει το πιο κάτω αποτέλεσμα:



**Σχήμα 6.2** Αν υπερκαλύψουμε την εικόνα 20 με τα διανύσματα οπτικής ροής μπορούμε να δούμε την κίνηση των σημείων.



**Σχήμα 6.3** Παρατηρούμε πως η κάμερα κινείται αριστερά, όμως η φαινομενική κίνηση είναι στα δεξιά. Βλέπουμε πως η έξοδος επαληθεύει την κίνηση.

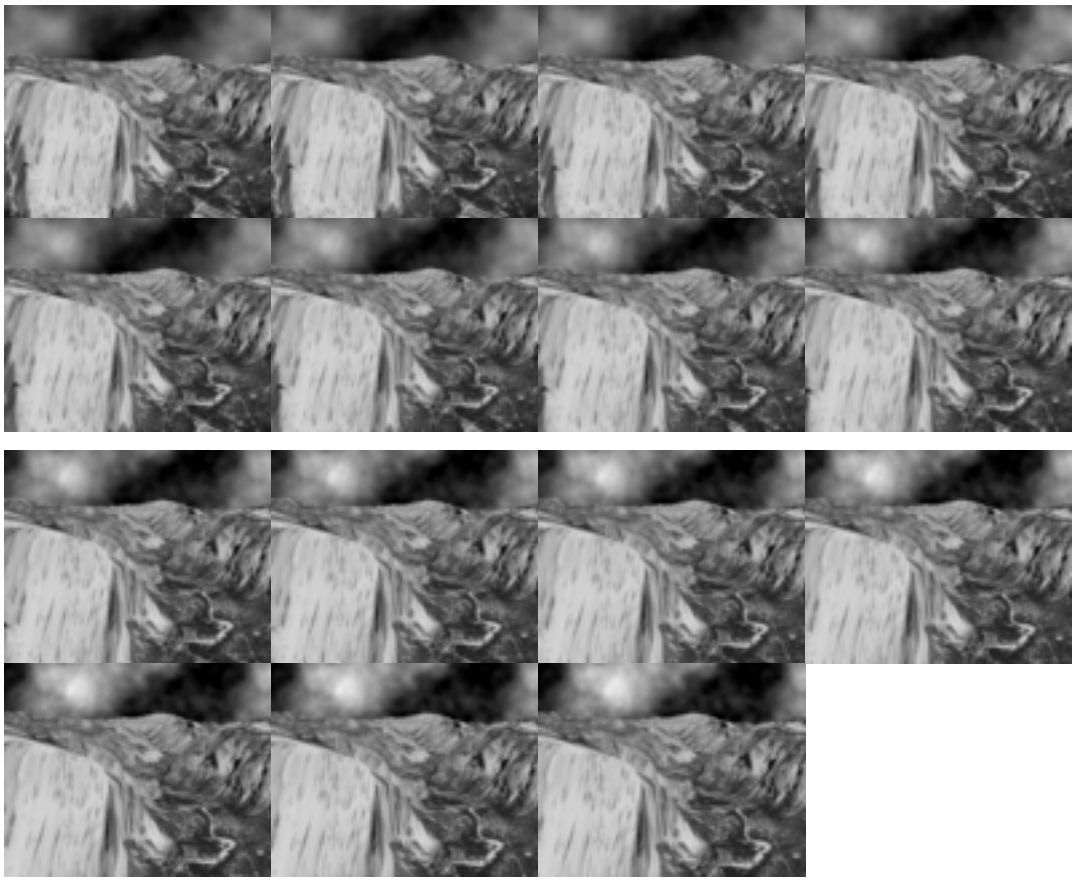
Στην υλοποίηση με κατάλληλες αλλαγές στον κώδικα, μπορούμε να πάρουμε στατιστικά δεδομένα που αφορούν το λάθος, την τυπική απόκλιση, το ποσοστό των σημείων που κινούνται και το λάθος.

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| New2binarytreet.20 | 62.383034 | 2.131299 | 53.164471 | 47.664925 | 66.724510 |
| New2binarytreet.21 | 62.335129 | 2.113307 | 52.633701 | 47.584126 | 66.064919 |
| New2binarytreet.22 | 62.263584 | 2.131326 | 52.136524 | 43.026672 | 67.419960 |

**Πίνακας 6.1** Πίνακας αποτελεσμάτων για τρία διαδοχικά πλαίσια της εικόνας του κινούμενου δένδρου. Παρουσιάζεται το λάθος που υπολογίζεται είναι το λάθος που προκύπτει από το θόρυβο και τον κβαντισμό, η τυπική απόκλιση, το ποσοστό κίνησης στην εικόνα και το μέγιστο και ελάχιστο λάθος κλίσης.

**Ακολουθία Yosemite**

Ακολούθως τρέχουμε το παράδειγμα με τις εικόνες από το Yosemite.
Η ακολουθία με τις εικόνες Yosemite είναι η πιο κάτω:



**Σχήμα 6.4** Στις εικόνες αυτές ο παρατηρητής κινείται μπροστά ενώ τα σύνεφα κινούνται προς τα δεξιά.

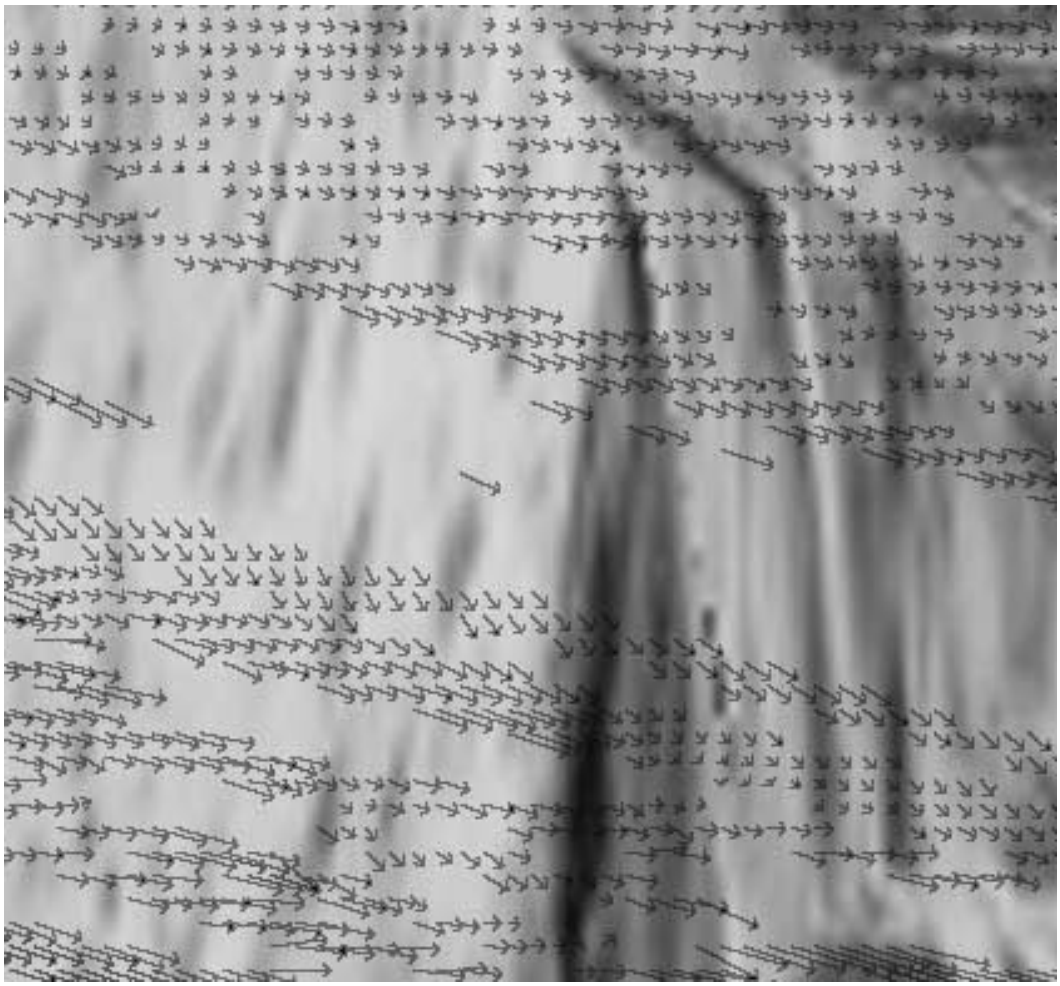
Η εντολή για να τρέξει ο αλγόριθμος τις εικόνες Yosemite είναι η ακόλουθη:

```
$ ./horn yos. 0.5 1.5 9 100 testdata/default_test_data/YOSEMITE_DATA result -C
testdata/default_test_data/CORRECT_FLOWS/correct_yos -MH -T 5.0 -B 312 252 > VID
EO_yos1.txt
```

Στο πιο κάτω σχήμα μπορούμε να δούμε την οπτική ροή της ακολουθίας Yosemite.

**Σχήμα 6.5** Αν υπερκαλύψουμε την εικόνα yos.9 μπορούμε να παρατηρήσουμε ποια σημεία κινούνται.

Λόγω μεγέθους θα παρουσιάσω μόνο μέρος της εικόνας για να φανούν καλύτερα τα διανύσματα οπτικής ροής.



**Σχήμα 6.6** Υπέρθεση οπτικής ροής πάνω στην αρχική εικόνα

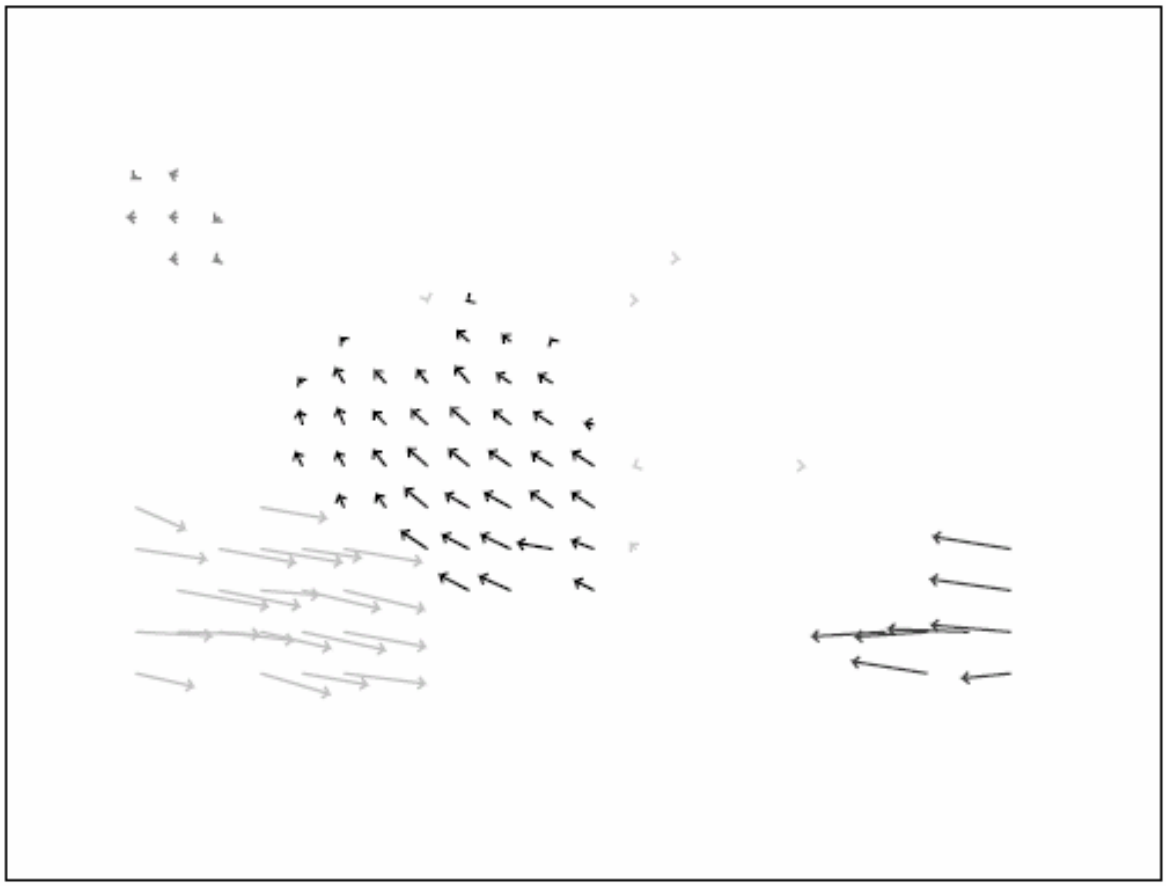
| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| yos.9 | 53.950443 | 21.454185 | 47.634243 | 0.589503 | 86.098305 |

**Πίνακας 6.2** Πίνακας αποτελεσμάτων για το πλαίσιο 9. Παρουσιάζεται το λάθος που υπολογίζεται είναι το λάθος που προκύπτει από το θόρυβο και τον κβαντισμό, η τυπική απόκλιση, το ποσοστό κίνησης στην εικόνα και το μέγιστο και ελάχιστο λάθος κλίσης.

**Η ακολουθία του ταξί του Αμβούργου**

Η ακολουθία του ταξί του Αμβούργου δίνεται πιο κάτω:

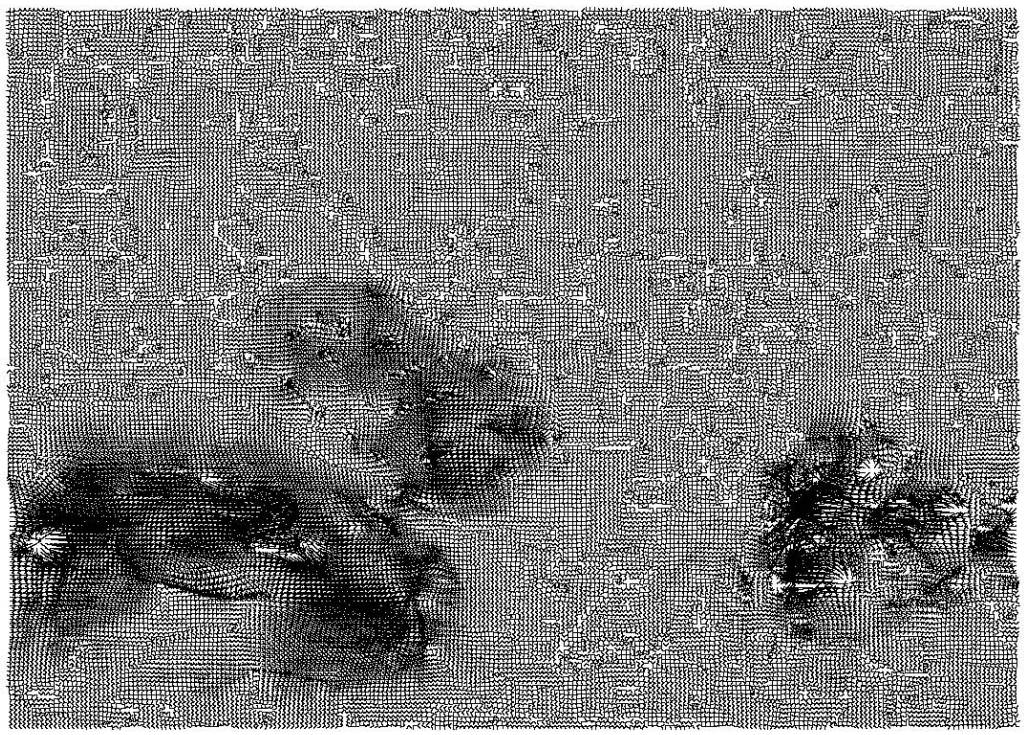
Δοκιμή από την δημοσίευση [A Phase-based Approach to the Estimation of the Optical Flow Field Using Spatial Filtering].



**Σχήμα 6.7** Αποτέλεσμα αλγορίθμου horn.Γραφική αναπαράσταση της οπτικής ροής.

Αρχικά, η γραμμή εντολής για τον υπολογισμό της οπτικής ροής για την εικόνα 10 είναι η εξής:

```
$ ./horn taxi_pgm. 0.5 1.5 10 100 testdata/default_test_data/TAXI_
DATA/ result/ -PGM > taxi.txt
```



**Σχήμα 6.8** Το αποτέλεσμα μας δίνει κίνηση παντού.

Το μέρος των διανυσμάτων ταχυτήτων είναι τα ακόλουθα:

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| taxi.10 | 13.615398 | 20.646772 | 100 | 0.000 | 85.821686 |

**Πίνακας 6.3** Πίνακας αποτελεσμάτων για το πλαίσιο 10 του παραδείγματος ταξί. Παρουσιάζεται το λάθος που υπολογίζεται είναι το λάθος που προκύπτει από το θόρυβο και τον κβαντισμό, η τυπική απόκλιση, το ποσοστό κίνησης στην εικόνα και το μέγιστο και ελάχιστο λάθος κλίσης.

Παρατηρούμε επίσης στα στατιστικά που δίνει σαν έξοδο ο αλγόριθμος πως υπάρχουν 100% ταχύτητες στην εικόνα.

Για να περιορίσουμε τα αντικείμενα που υπολογίζεται η κίνηση, χρησιμοποιούμε την παράμετρο κατωφλίωσης των Ix και Iy.

Αυτό επιτυγχάνεται με την ακόλουθη εντολή.

```
$ ./horn taxi_pgm. 0.5 1.5 9 10 testdata/default_test_data/TAXI_DA A result -PGM -T 5
>taxi.txt
```

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| taxi.10 | 8.982820 | 16.312113 | 22.936972 | 0.000 | 70.608612 |

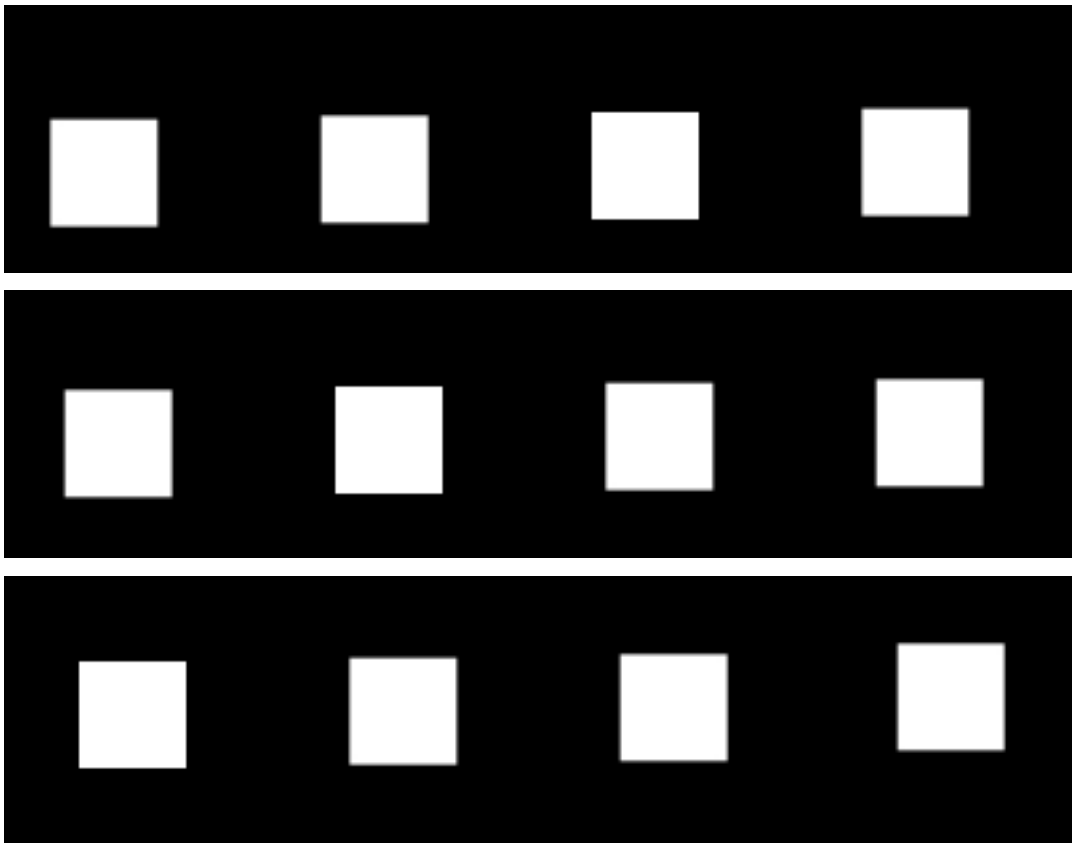
**Πίνακας 6.4** Αποτελέσματα με κατωφλίωση. Το ποσοστό κίνησης μειώνεται στο 22.93 % της εικόνας

Όταν υπερθέσουμε πάνω από την εικόνα taxi.10 παρατηρούμε την εικόνα 6.9:

**Σχήμα 6.9** Η κίνηση που παρατηρείται περιορίζεται κοντά στα αντικείμενα για τα οποία θέλουμε να μετρήσουμε την κίνηση.

**Η ακολουθία του Square2**

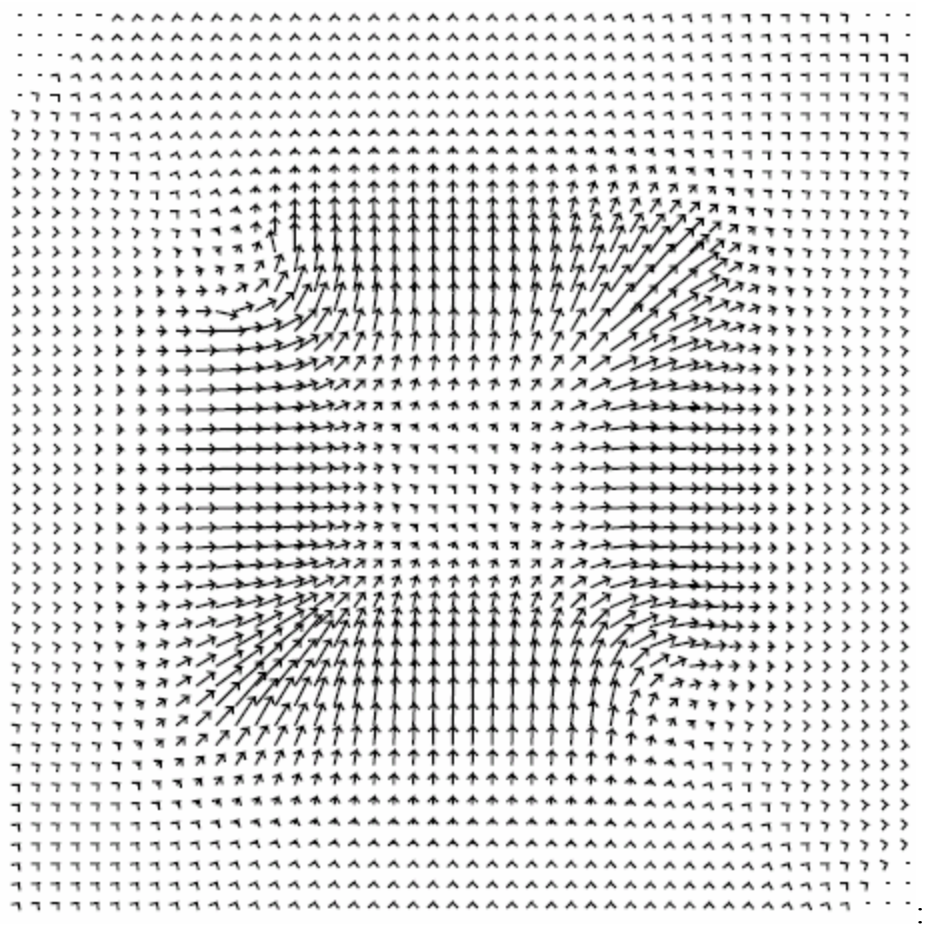
Στην ακολουθία αυτή, ένα σύνολο από εινονοστοιχεία που σχηματίζουν τετράγωνο με τιμή φωτεινότητας 255 κινείται διαγώνια προς πάνω δεξιά σε φόντο με τιμή φωτεινότητας 0.



**Σχήμα 6.10** Σχήμα κινούμενου κύβου. Ο κύβος κινείται από αριστερά στα άνω δεξιά [2].

Από την δημοσίευση των Barron, Fleet και Beauchemin Performance of Optical Flow Techniques, βλέπουμε το αποτέλεσμα της οπτικής ροής για το square2.

Το πεδίο της ροής φένεται πιο κάτω

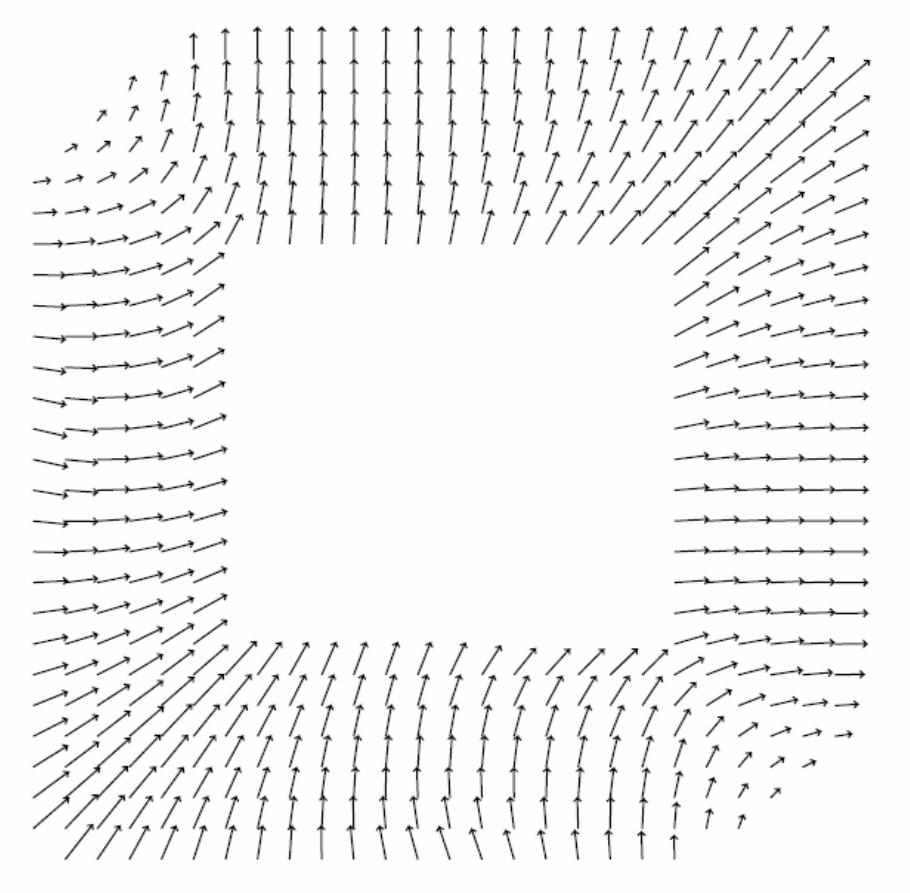


**Σχήμα 6.12** Γραφική αναπαράσταση οπτικής ροής για το παράδειγμα κινούμενου κύβου.

Όταν τρέξουμε την υλοποίηση του αλγορίθμου horn με την εντολή:

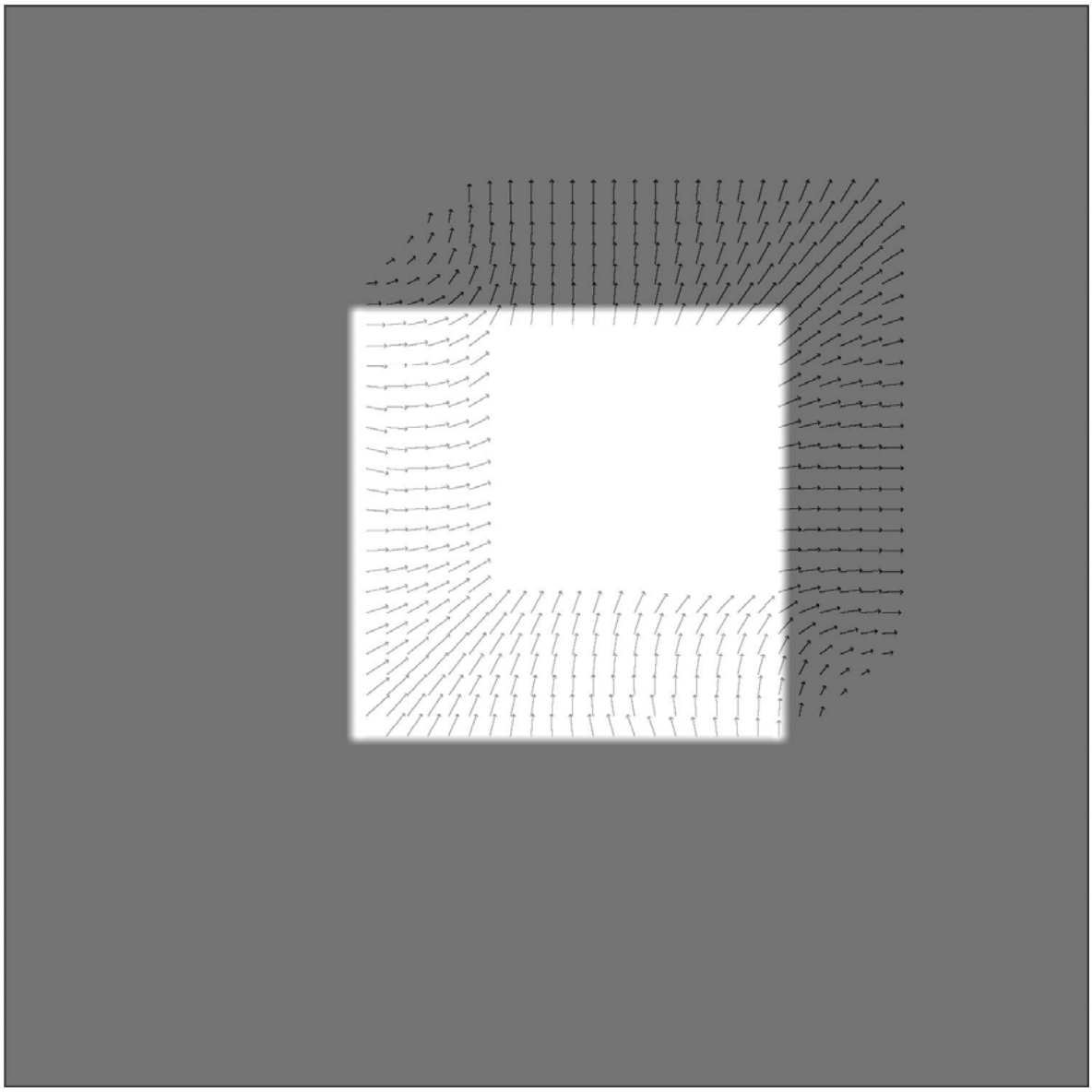
```
$ ./horn square2_p. 0.5 1.5 9 100 testdata/default_test_data/SQUARE_DATA result -C
testdata/default_test_data/CORRECT_FLOWS/correct_square2 -MH -T 1.0 -PGM >sq
uare.txt
```

δίνει το ακόλουθο σχήμα οπτικής ροής που συμφωνεί με την οπτική ροή στην δημοσίευση.



**Σχήμα 6.13** Αποτέλεσμα οπτικής ροής κινούμενου κύβου.

Όταν υπερθέσουμε την εικόνα square2.10 με το πεδίο της οπτικής ροής παίρνουμε την πιο κάτω εικόνα.



**Σχήμα 6.14** Υπέρθεση οπτικής ροής στην τελική εικόνα υπολογισμού κίνησης.

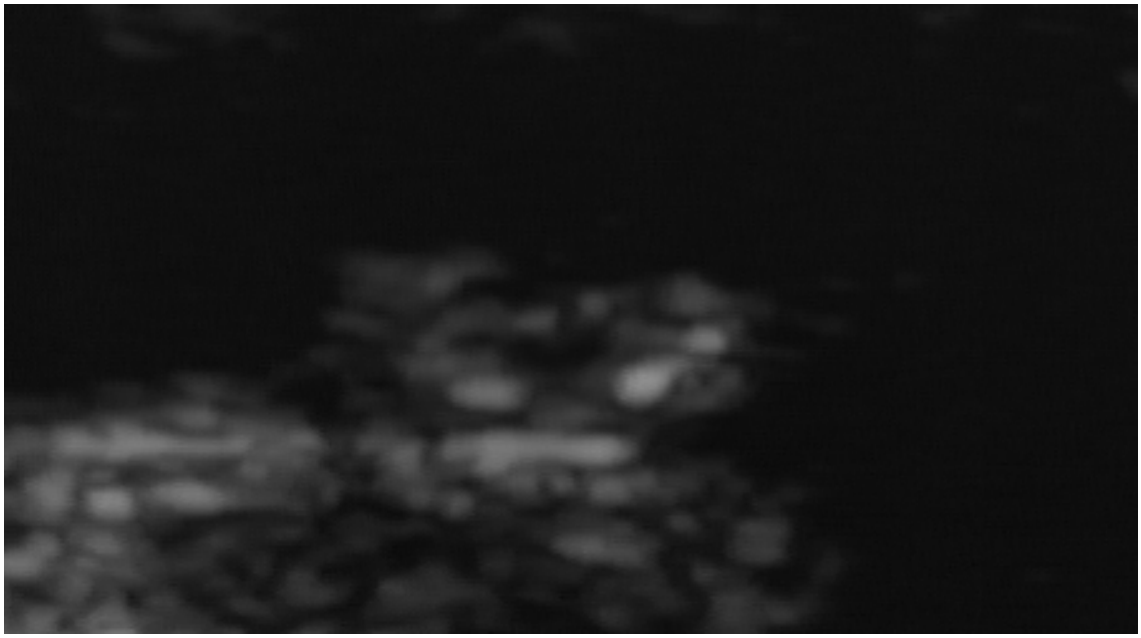
Τα στατιστικά δεδομένα που προκύπτουν είναι:

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| Square2.9 | 53.866962 | 5.205721 | 38.406635 | 26.198874 | 63.792873 |
| Square2.10 | 52.712601 | 4.878706 | 38.406635 | 26.277588 | 62.384987 |
| Square2.11 | 53.469463 | 10.809710 | 38.503086 | 2.274344 | 81.908951 |
| Square2.12 | 64.702454 | 19.053194 | 48.707561 | 0.865472 | 87.956985 |
| Square2.13 | 73.163521 | 18.428545 | 51.523918 | 3.034543 | 89.217712 |

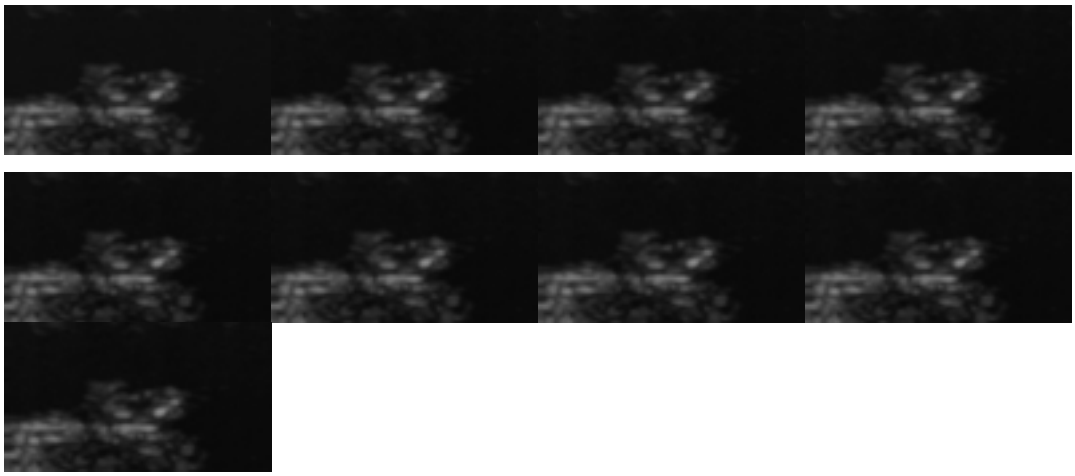
**Πίνακας 6.5** Αποτελέσματα κινούμενου κύβου για 5 διαδοχικά πλαίσια.

### 6.3 Δοκιμές με τεχνητά δεδομένα

Η πιο κάτω πλάκα, κινείται δεξιά κατά ένα εικονοστοιχείο σε κάθε πλαίσιο.Η κίνηση είναι τεχνητή και αφορά όλα τα εικονοστοιχεία.



**Σχήμα 6.15** Εικόνα πλάκας



**Σχήμα 6.16** Ακολουθία από πλάκες, με κίνηση δεξιά.

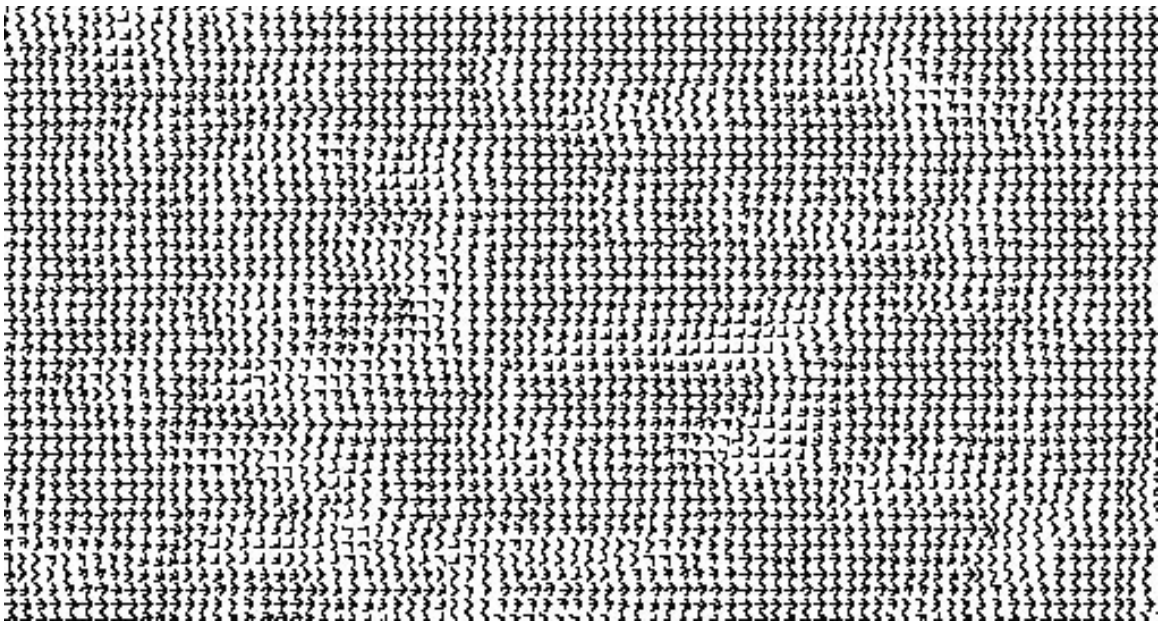
```
./horn kinoumeni. 0.5 0.5 4 10 testdata/kinoumeni-plaka-deksia result -PGM
>kinoumeni_deksia.txt
```

Πιο κάτω μπορούμε να δούμε αριθμητικά το διάνυσμα της ταχύτητας σε κάθε εικονοστοιχείο.

| | | |
|---|---|---|
| Velocities [8][113] | x: 0.814535 | y: 0.004529 |
| Velocities [8][114] | x: 0.984969 | y: -0.020026 |
| Velocities [8][115] | x: 0.835628 | y: -0.027326 |
| Velocities [8][116] | x: 0.836381 | y: -0.011214 |
| Velocities [8][117] | x: 0.992012 | y: -0.000580 |
| Velocities [8][118] | x: 0.993431 | y: 0.002215 |
| Velocities [8][119] | x: 0.991969 | y: -0.003064 |
| Velocities [8][120] | x: 0.915676 | y: -0.011577 |
| Velocities [8][121] | x: 0.990322 | y: 0.014279 |
| Velocities [8][122] | x: 0.851892 | y: 0.018966 |
| Velocities [8][123] | x: 0.817494 | y: -0.048065 |
| Velocities [8][124] | x: 0.762619 | y: 0.001941 |
| Velocities [8][125] | x: 0.801930 | y: 0.067614 |
| Velocities [8][126] | x: 0.819944 | y: 0.007996 |
| Velocities [8][127] | x: 0.913260 | y: 0.081423 |
| Velocities [8][128] | x: 0.893754 | y: 0.022762 |
| Velocities [8][129] | x: 0.878974 | y: 0.002272 |

| | | |
|---|---|---|
| Velocities [8][130] | x: 0.905870 | y: -0.010936 |
| Velocities [8][131] | x: 0.988412 | y: -0.026818 |
| Velocities [8][132] | x: 0.994376 | y: -0.011824 |
| Velocities [8][133] | x: 0.987707 | y: 0.004229 |
| Velocities [8][134] | x: 0.998749 | y: 0.015981 |
| Velocities [8][135] | x: 0.971339 | y: 0.014934 |
| Velocities [8][136] | x: 0.998909 | y: 0.003823 |
| Velocities [8][137] | x: 0.976661 | y: 0.011848 |
| Velocities [8][138] | x: 0.973862 | y: 0.005273 |
| Velocities [8][139] | x: 0.980252 | y: 0.006517 |
| Velocities [8][140] | x: 0.998280 | y: -0.002675 |
| Velocities [8][141] | x: 1.000718 | y: -0.001961 |
| Velocities [8][142] | x: 0.984291 | y: -0.025812 |
| Velocities [8][143] | x: 0.969897 | y: -0.038464 |
| Velocities [8][144] | x: 0.963710 | y: -0.038901 |
| Velocities [8][145] | x: 0.968951 | y: -0.029473 |
| Velocities [8][146] | x: 0.981714 | y: -0.001957 |
| Velocities [8][147] | x: 0.998978 | y: -0.010683 |
| Velocities [8][148] | x: 0.998561 | y: -0.016372 |
| Velocities [8][149] | x: 0.994766 | y: -0.020485 |
| Velocities [8][150] | x: 0.972306 | y: -0.000359 |
| Velocities [8][151] | x: 0.995819 | y: 0.006232 |
| Velocities [8][152] | x: 0.940938 | y: -0.011809 |
| Velocities [8][153] | x: 0.993867 | y: -0.029823 |
| Velocities [8][154] | x: 0.978872 | y: -0.041192 |
| Velocities [8][155] | x: 0.953603 | y: -0.000052 |
| Velocities [8][156] | x: 0.962700 | y: -0.000192 |
| Velocities [8][157] | x: 0.986780 | y: -0.026757 |
| Velocities [8][158] | x: 0.998671 | y: -0.015891 |
| Velocities [8][159] | x: 0.998630 | y: -0.003095 |
| Velocities [8][160] | x: 0.994558 | y: 0.012235 |
| Velocities [8][161] | x: 0.948443 | y: 0.000742 |
| Velocities [8][162] | x: 0.975102 | y: -0.029103 |
| Velocities [8][163] | x: 0.950105 | y: 0.034721 |
| Velocities [8][164] | x: 0.935741 | y: -0.004410 |
| Velocities [8][165] | x: 0.988456 | y: -0.015418 |

**Πίνακας 6.6** Παρατηρούμε πως υπάρχει οριζόντια κίνηση, με φορά δεξιά σε όλη την εικόνα. Η έξοδος υπολογισμού της οπτικής ροής φαίνεται πιο κάτω.



**Σχήμα 6.17** Γραφική αναπαράσταση οπτικής ροής για την πλάκα με κίνηση δεξιά.

Ο υπολογισμός της οπτικής ροής για τις εικόνες kinoumeni.10 – kinoumeni.15 δίνει τα πιο κάτω στατιστικά δεδομένα.

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| Kinoumeni.10 | 39.394768 | 8.003906 | 100.000000 | 0.731149 | 47.609051 |
| Kinoumeni.11 | 40.502640 | 7.020696 | 100.000000 | 1.680861 | 47.379803 |
| Kinoumeni.12 | 40.502640 | 7.020696 | 100.000000 | 1.680861 | 47.379803 |
| Kinoumeni.13 | 41.314411 | 6.186968 | 100.000000 | 2.926701 | 47.189297 |
| Kinoumeni.14 | 41.314411 | 6.186968 | 100.000000 | 2.926701 | 47.189297 |
| Kinoumeni.15 | 41.932590 | 5.467310 | 100.000000 | 4.437870 | 46.985939 |

**Πίνακας 6.7** Πίνακας αποτελεσμάτων για την κινούμενη πλάκα

Χρησιμοποιούμε τώρα την παράμετρο –T για κατωφλίωση των υπολογισμένων ταχυτήτων του διανύσματος χωρικής έντασης.

Η εντολή είναι:

```
$ ./horn kinoumeni. 0.5 0.5 4 10 testdata/kinoumeni-plaka-deksia result -PGM -T
3.5 >kinoumeni_T3.5.txt
```

Μέρος των διανυσμάτων ταχύτητας είναι τα πιο κάτω:

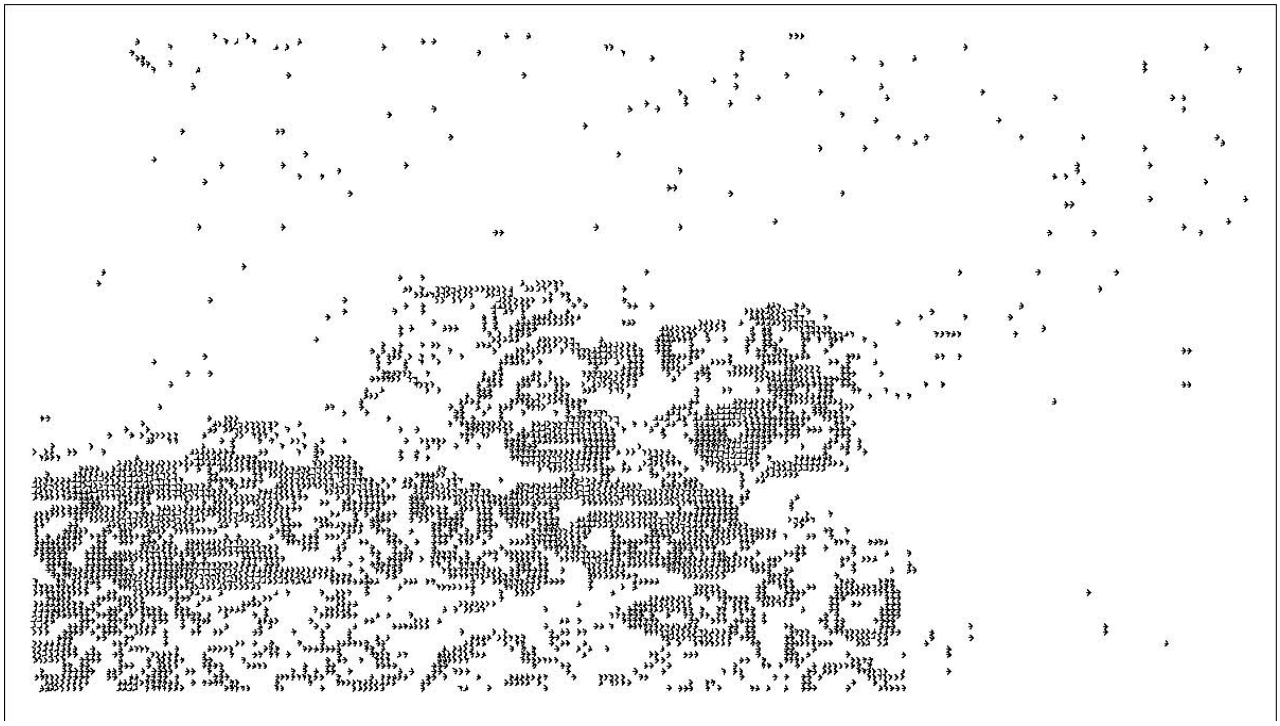
| | | |
|---|---|---|
| Velocities [8][69] | x: 0.790789 | y: 0.109780 |
| Velocities [8][71] | x: 0.654628 | y: -0.217156 |
| Velocities [8][72] | x: 0.587635 | y: -0.094654 |
| Velocities [8][83] | x: 0.349763 | y: -0.057000 |
| Velocities [8][84] | x: 0.420281 | y: 0.036049 |
| Velocities [8][87] | x: 0.809498 | y: -0.088035 |
| Velocities [8][176] | x: 0.897932 | y: 0.084981 |
| Velocities [8][184] | x: 0.849710 | y: 0.142945 |
| Velocities [8][278] | x: 0.578527 | y: -0.047123 |
| Velocities [8][279] | x: 0.555816 | y: -0.073102 |
| Velocities [8][280] | x: 0.545115 | y: -0.010432 |
| Velocities [8][281] | x: 0.574243 | y: 0.003965 |
| Velocities [8][282] | x: 0.602248 | y: 0.044685 |
| Velocities [8][323] | x: 0.967674 | y: -0.022648 |
| Velocities [8][331] | x: 0.948891 | y: 0.031795 |
| Velocities [8][369] | x: 0.717962 | y: -0.093326 |
| Velocities [9][44] | x: 0.906903 | y: 0.093436 |
| Velocities [9][45] | x: 0.901282 | y: 0.098131 |
| Velocities [9][72] | x: 0.633596 | y: -0.263384 |
| Velocities [9][73] | x: 0.637343 | y: -0.249271 |
| Velocities [9][74] | x: 0.554713 | y: -0.044848 |
| Velocities [9][75] | x: 0.522230 | y: -0.038913 |
| Velocities [9][80] | x: 0.354832 | y: 0.187562 |
| Velocities [9][82] | x: 0.256368 | y: -0.019027 |
| Velocities [9][83] | x: 0.285874 | y: -0.000630 |
| Velocities [9][84] | x: 0.368514 | y: -0.019624 |
| Velocities [9][85] | x: 0.497616 | y: -0.132219 |
| Velocities [9][103] | x: 0.947422 | y: 0.073902 |
| Velocities [9][121] | x: 0.901219 | y: -0.081474 |
| Velocities [9][200] | x: 0.983214 | y: 0.023434 |

| | | |
|---|---|---|
| Velocities [9][209] | x: 0.993533 | y: 0.008480 |
| Velocities [10][44] | x: 0.898408 | y: 0.138097 |
| Velocities [10][53] | x: 0.703092 | y: -0.184595 |
| Velocities [10][75] | x: 0.661002 | y: -0.237843 |
| Velocities [10][76] | x: 0.643191 | y: -0.215421 |
| Velocities [10][77] | x: 0.597427 | y: 0.208492 |
| Velocities [10][79] | x: 0.364250 | y: 0.043208 |
| Velocities [10][80] | x: 0.448244 | y: 0.332239 |
| Velocities [10][85] | x: 0.606772 | y: -0.278320 |
| Velocities [10][86] | x: 0.709215 | y: -0.234449 |
| Velocities [10][102] | x: 0.840236 | y: 0.150906 |
| Velocities [10][146] | x: 0.965540 | y: 0.017947 |
| Velocities [10][150] | x: 0.944605 | y: 0.054276 |
| Velocities [10][153] | x: 0.924206 | y: -0.054110 |
| Velocities [11][43] | x: 0.931641 | y: 0.095018 |
| Velocities [11][53] | x: 0.602224 | y: -0.038369 |
| Velocities [11][55] | x: 0.695114 | y: -0.222318 |
| Velocities [11][56] | x: 0.741461 | y: -0.250016 |
| Velocities [11][86] | x: 0.755251 | y: -0.239995 |
| Velocities [11][87] | x: 0.808886 | y: -0.123527 |
| Velocities [11][88] | x: 0.907343 | y: 0.033742 |
| Velocities [11][95] | x: 0.524749 | y: -0.007084 |
| Velocities [11][96] | x: 0.535726 | y: -0.004237 |
| Velocities [11][97] | x: 0.586228 | y: 0.036781 |
| Velocities [11][98] | x: 0.671824 | y: 0.225343 |
| Velocities [11][99] | x: 0.650919 | y: 0.242345 |
| Velocities [11][210] | x: 0.731830 | y: -0.174534 |
| Velocities [11][228] | x: 0.970400 | y: -0.042458 |
| Velocities [11][256] | x: 0.981129 | y: -0.018741 |
| Velocities [11][279] | x: 0.836344 | y: 0.087446 |
| Velocities [11][321] | x: 0.940240 | y: -0.058571 |
| Velocities [12][43] | x: 0.957805 | y: 0.061429 |
| Velocities [12][56] | x: 0.729279 | y: -0.261232 |
| Velocities [12][57] | x: 0.711457 | y: -0.214418 |
| Velocities [12][93] | x: 0.643121 | y: 0.256903 |
| Velocities [12][94] | x: 0.598471 | y: 0.279875 |
| Velocities [12][97] | x: 0.522989 | y: 0.016936 |
| Velocities [12][98] | x: 0.529366 | y: 0.041990 |
| Velocities [12][132] | x: 0.965044 | y: 0.021789 |
| Velocities [12][212] | x: 0.568014 | y: -0.250135 |

| | | |
|---|---|---|
| Velocities [12][213] | x: 0.583497 | y: -0.261896 |
| Velocities [12][214] | x: 0.506236 | y: -0.038117 |

**Πίνακας 6.8** Παρατηρούμε εδώ πως με την κατωφλίωση μεγάλο μέρος των διανυσμάτων ταχυτήτων χάθηκε.

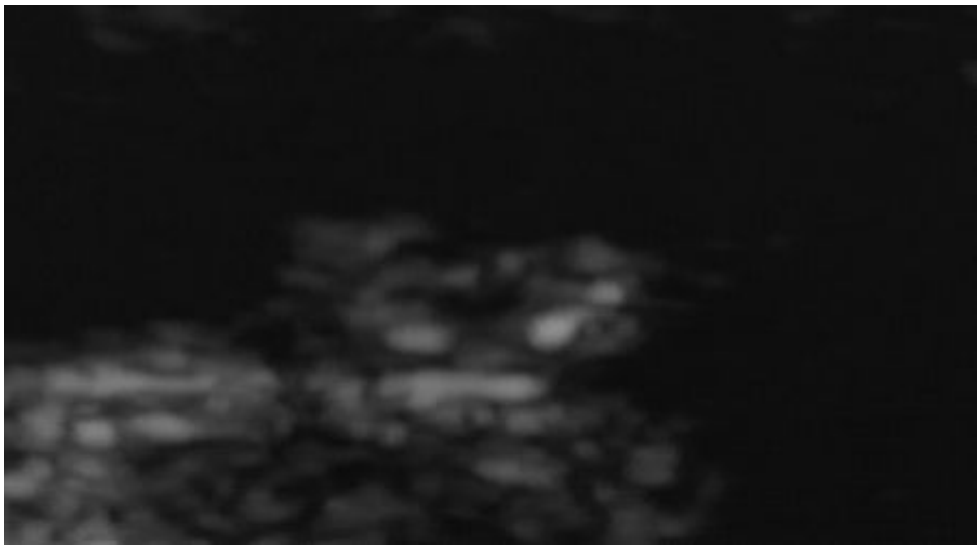
Τα διανύσματα ροής γραφικά φαίνονται πιο κάτω:



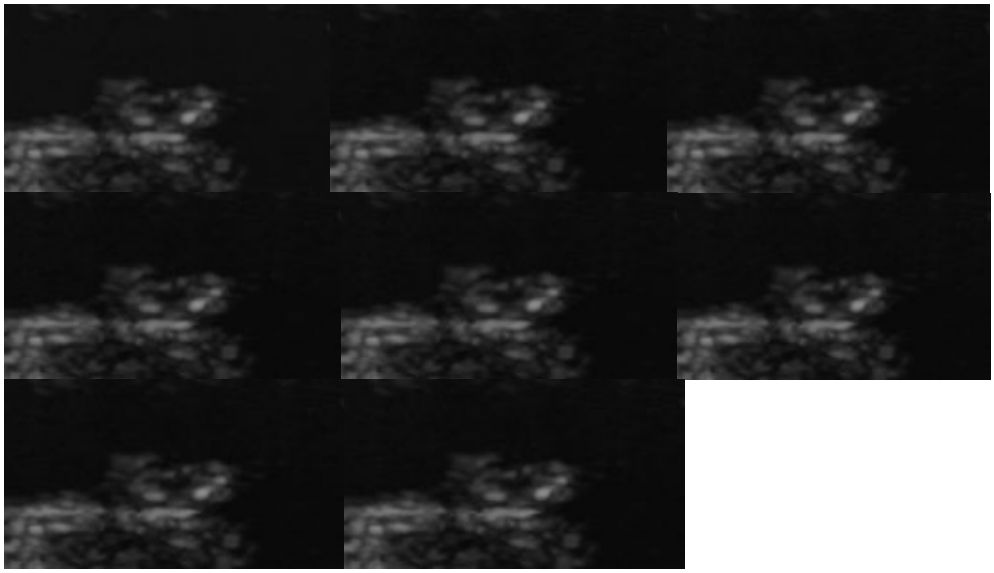
**Σχήμα 6.18** Τα διανύσματα ροής επικεντρώνονται στα αντικείμενα.

**Μέρος πλάκας κινείται ένα εικονοστοιχείο δεξιά.**

Πιο κάτω βλέπουμε την διατομή μιας αρτηρίας και την πλάκα που υπάρχει σε αυτήν. Στην δοκιμή αυτή, το κομμάτι της πλάκας κινείται οριζόντια με φορά δεξιά.



**Σχήμα 6.19** Παράδειγμα πλάκας.

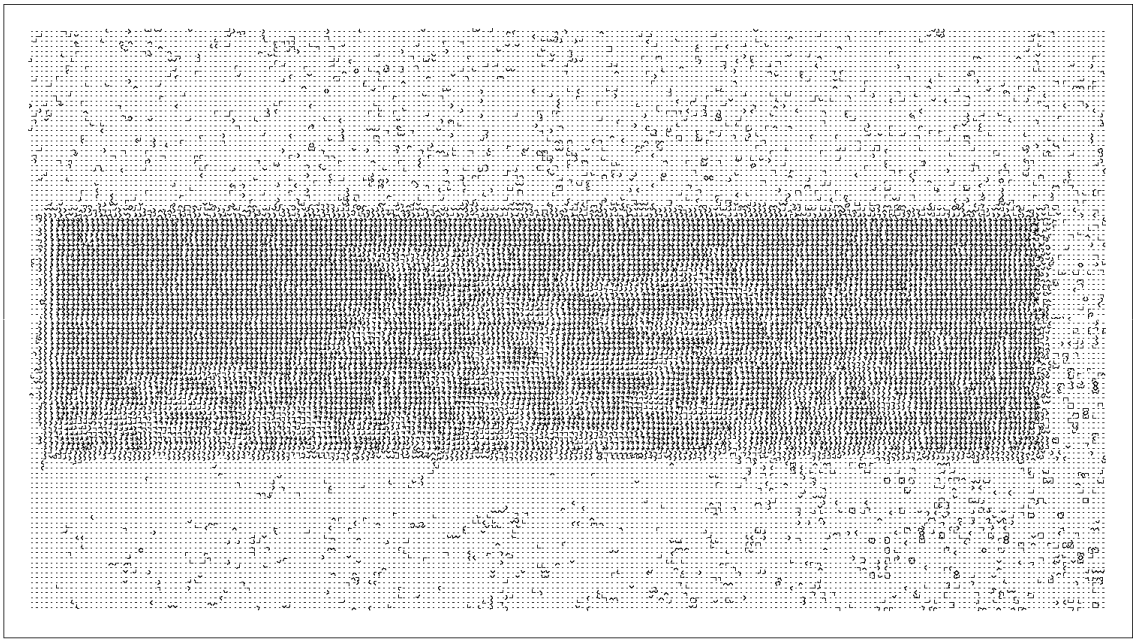


**Σχήμα 6.20** Ακολουθία εικόνων πλάκας στην οποία μέρος της πλάκας κινείται δεξιά.

Για να υπολογίσουμε την οπτική ροή χρησιμοποιούμε την εντολή:

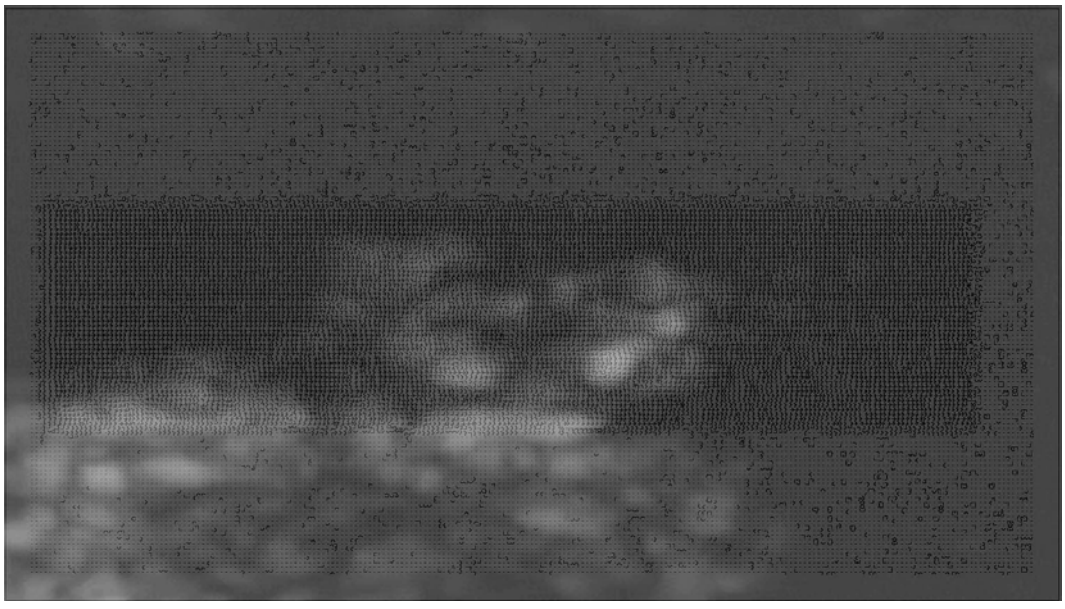
```
$ ./horn kinoumeni-meros. 0.5 0.5 4 5 testdata/kinoumeni-plaka-meros-deksia/ result -PGM
```

Το αποτέλεσμα του υπολογισμού της οπτικής ροής είναι όπως φαίνεται πιο κάτω.



**Σχήμα 6.21** Γραφική αναπαράσταση οπτικής ροής. Η κίνηση είναι μόνο στο σημείο που μετακινήσαμε. Η κίνηση στα υπόλοιπα σημεία είναι θόρυβος.

Με υπέρθεση στην εικόνα 4 για την οποία υπολογίζεται η οπτική ροή παίρνουμε την σχηματική αναπαράσταση του υπολογισμού.



**Σχήμα 6.22** Υπέρθεση τελικής εικόνας και οπτικής ροής.

Αν δούμε και τα αριθμητικά δεδομένα σε περιοχές που δεν υπάρχει η τεχνητή κίνηση:

| | | |
|---|---|---|
| Velocities [81][64] | x: 0.036464 | y: -0.009585 |
| Velocities [81][65] | x: 0.003203 | y: -0.009063 |
| Velocities [81][66] | x: 0.024512 | y: -0.009912 |
| Velocities [81][67] | x: 0.044611 | y: -0.032989 |
| Velocities [81][68] | x: 0.078017 | y: -0.040373 |
| Velocities [81][69] | x: 0.001820 | y: -0.016462 |
| Velocities [81][70] | x: 0.038230 | y: -0.036785 |
| Velocities [81][71] | x: 0.064857 | y: 0.009464 |
| Velocities [81][72] | x: 0.012214 | y: -0.056048 |
| Velocities [81][73] | x: 0.117370 | y: -0.049948 |
| Velocities [81][74] | x: 0.102551 | y: -0.018606 |
| Velocities [81][75] | x: 0.003350 | y: 0.000195 |
| Velocities [81][76] | x: 0.001073 | y: 0.010529 |
| Velocities [81][77] | x: 0.001602 | y: 0.021599 |
| Velocities [81][78] | x: 0.000977 | y: 0.039815 |
| Velocities [81][79] | x: 0.055418 | y: 0.047419 |
| Velocities [81][80] | x: 0.016273 | y: 0.026210 |
| Velocities [81][81] | x: 0.006351 | y: 0.026278 |
| Velocities [81][82] | x: 0.045887 | y: 0.044318 |
| Velocities [81][83] | x: 0.063284 | y: 0.030142 |
| Velocities [81][84] | x: 0.003724 | y: 0.003808 |
| Velocities [81][85] | x: 0.004095 | y: 0.024599 |
| Velocities [81][86] | x: 0.057694 | y: 0.013618 |
| Velocities [81][87] | x: 0.041232 | y: 0.040774 |
| Velocities [81][88] | x: 0.058005 | y: 0.047984 |
| Velocities [81][89] | x: 0.107015 | y: 0.035858 |
| Velocities [81][90] | x: 0.075077 | y: 0.041289 |
| Velocities [81][91] | x: 0.032696 | y: 0.039807 |
| Velocities [81][92] | x: 0.001269 | y: 0.000053 |
| Velocities [81][93] | x: 0.000675 | y: 0.000162 |
| Velocities [81][94] | x: 0.000477 | y: -0.003755 |

**Πίνακας 6.9** Στα σημεία όπου δεν υπάρχει κίνηση, παρατηρούνται αμελητέες τιμές λόγω θορύβου.

Και τα αριθμητικά δεδομένα όπου υπάρχει κίνηση, συμφωνούμε με τα αποτελέσματα του αλγορίθμου, αν και οι παρατηρήσεις μας επιβεβαιώνουν το λάθος που υπολογίζεται από τους στατιστικούς υπολογισμούς.

| | | |
|---|---|---|
| Velocities [83][67] | x: 0.582237 | y: 0.183295 |
| Velocities [83][68] | x: 0.833450 | y: 0.217675 |
| Velocities [83][69] | x: 0.756815 | y: -0.125280 |
| Velocities [83][70] | x: 0.798704 | y: -0.028102 |
| Velocities [83][71] | x: 0.600533 | y: -0.096908 |
| Velocities [83][72] | x: 0.591048 | y: 0.168408 |
| Velocities [83][73] | x: 0.938070 | y: 0.130267 |
| Velocities [83][74] | x: 0.752457 | y: 0.131068 |
| Velocities [83][75] | x: 0.514997 | y: -0.016451 |
| Velocities [83][76] | x: 0.435742 | y: 0.024966 |
| Velocities [83][77] | x: 0.430830 | y: 0.047081 |
| Velocities [83][78] | x: 0.555127 | y: -0.177231 |
| Velocities [83][79] | x: 0.828130 | y: 0.228096 |
| Velocities [83][80] | x: 0.604334 | y: 0.128853 |
| Velocities [83][81] | x: 0.652618 | y: 0.166324 |
| Velocities [83][82] | x: 0.548456 | y: -0.069967 |
| Velocities [83][83] | x: 0.572912 | y: 0.029929 |
| Velocities [83][84] | x: 0.965506 | y: -0.082639 |
| Velocities [83][85] | x: 0.965274 | y: -0.054990 |
| Velocities [83][86] | x: 0.707139 | y: 0.149838 |
| Velocities [83][87] | x: 0.475014 | y: 0.138528 |
| Velocities [83][88] | x: 0.417218 | y: 0.303220 |
| Velocities [83][89] | x: 0.687487 | y: 0.245238 |
| Velocities [83][90] | x: 0.894599 | y: -0.160543 |
| Velocities [83][91] | x: 0.804439 | y: -0.252823 |
| Velocities [83][92] | x: 0.916995 | y: 0.143816 |
| Velocities [83][93] | x: 0.480435 | y: 0.102258 |
| Velocities [83][94] | x: 0.443328 | y: -0.069742 |
| Velocities [83][95] | x: 0.584570 | y: -0.164665 |

**Πίνακας 6.10** Οι ταχύτητες στα σημεία με κίνηση είναι κοντά στο 1 εικονοστοιχείο ανά πλαίσιο.

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| Kinoumeni-meros-deksia.4 | 12.537430 | 17.618612 | 100.000000 | 0.000000 | 71.210838 |

**Πίνακας 6.11** Τα στατιστικά δεδομένα για την εικόνα kinoumeni-meros-deksia.5

## 6.4 Αποτελέσματα με πραγματικά δεδομένα

|Για την ανάλυση με πραγματικά δεδομένα θα δείξουμε πως τρέχει ο αλγόριθμος σε μια ακολουθία εικόνων που προέκυψαν από μέρος βίντεο το οποίο έχει επεξεργαστεί και έχει επιλεχθεί μόνο η πλάκα. Για την χρονική αποκοπή έγινε χρήση του γραφικού περιβάλλοντος και για την χωρικήαποκοπή έγινε χρήση της γραμμής εντολών.

### Πλάκα μεγέθους 450*250 cardiac1_0001_t11_t14_crop

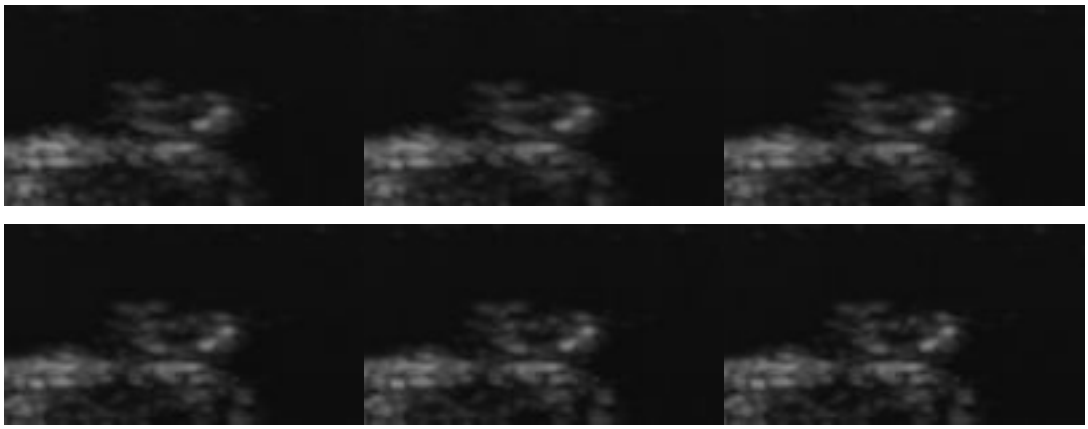
Όπως συμπαιρένουμε από το όνομα του βίντεο, η πλάκα είναι μέρος της κασέτας cardiac1 το πρώτο βίντεο (0001) και μετρήθηκε το εντέκατο δευτερόλεπτο μέχρι το δέκατοτέταρτο.
Αν κάνουμε αναπαραγωγή του βίντεο θα δούμε πως καταγράφτηκαν τρεισήμισυ παλμοί. Εμείς θέλουμε μόνο ένα.
Με το γραφικό περιβάλλον για επιλογή βίντεο, επιλέγουμε δύο παλμούς. Από 1-2 δευτερόλεπτα και από 2-3 δευτερόλεπτα.
Χρησιμοποιώντας το εργαλείο Xnview και παρατηρώντας την κίνηση της πλάκας πλαίσιο με πλαίσιο παρατηρούμε πως ορθά έχουν επιλεγεί οι πλάκες και πως σε κάθε σύνολο πλαισίων καταγράφτηκε ένας πλήρης παλμός κάθε φορά για 25 εικόνες.

Θα αναλύσουμε αρχικά τον παλμό σε χρόνο 1-2 στο βίντεο cardiac1_0001_t11_t14_crop.

Να σημειωθεί εδώ πως ενώ στα παραδείγματα ο αριθμός των επαναλήψεων ήταν πάντα 100, εμείς δε θα χρησιμοποιήσουμε τόσο μεγάλο αριθμό επαναλήψεων για να αποφύγουμε λάθη που προκύπτουν από την εφαρμογή του μέσου όρου.

**Παλμός 1-2**

Το μέρος της ακολουθίας εικόνων που θα επεξεργαστούμε είναι οι πιο κάτω:



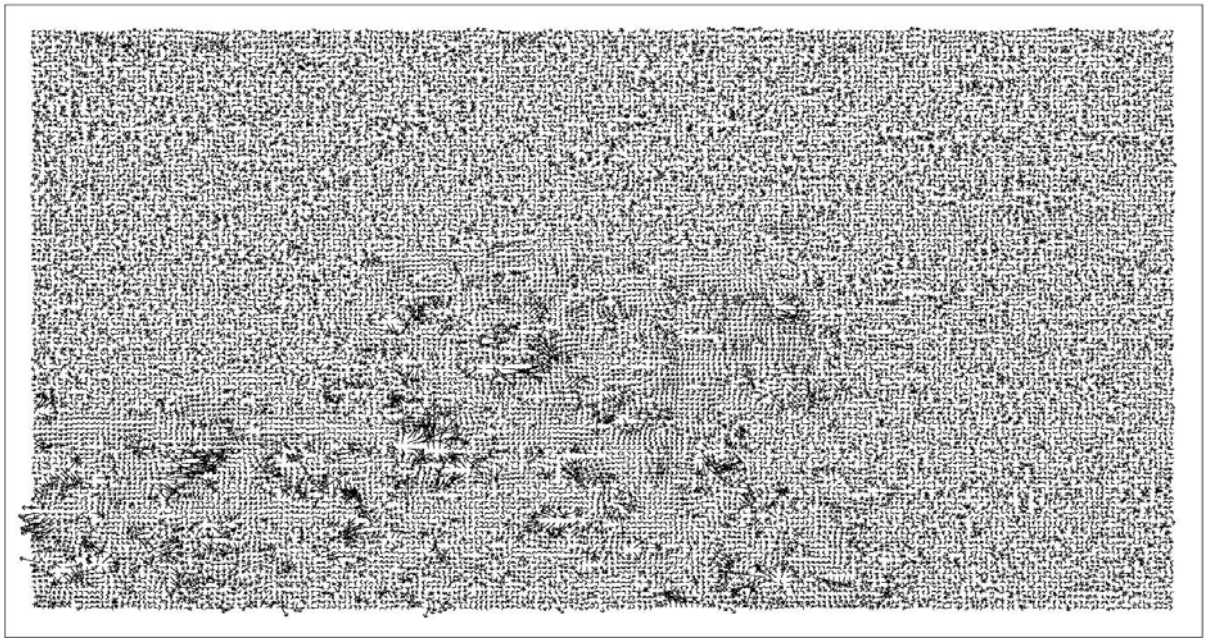
**Σχήμα 6.23** Ακολουθία εικόνων από πραγματικά δεδομένα.

Χρησιμοποιώντας το script prepare_no_threshold ονομάζουμε τις εικόνες t1-t2-simple γιατί θα χρησιμοποιήσουμε τις προεπιλεγμένες επιλογές της υλοποίησης που χρησιμοποιούνται και στα παραδείγματα. Η μόνη διαφορά είναι πως οι εικόνες που διαβάζουμε τώρα είναι εικόνες PGM.

Η εντολή που χρησιμοποιήσαμε είναι η πιο κάτω:

```
$ ./horn t1_t2_simple. 0.5 0.5 9 20
../croped/cardiac1_0001_t11_t14_crop/framescardiac1_0001_t11_t14_cr
op.avi_t1_t2.avi result -PGM
```

Το αποτέλεσμα που παίρνουμε είναι ο υπολογισμός της οπτικής ροής σε όλα τα σημεία. Για καλύτερη αναπαράσταση και υπολογισμό των τιμών της οπτικής ροής θα χρησιμοποιήσουμε την παράμετρο κατωφλίωσης των Ix και Iy.



**Σχήμα 6.24** Γραφική αναπαράσταση οπτικής ροής για τον παλμό 1-2.Παρατηρείται κίνηση παντού, όμως στα αντικείμενα είναι περισσότερη η κίνηση.

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| t1_t2_simple.9 | 30.768013 | 16.058172 | 100 | 0.062557 | 85.929764 |

**Πίνακας 6.12** Τα στατιστικά δεδομένα της εικόνας 9 χωρίς κατωφλίωση

Για να καταλάβουμε καλύτερα το πως εφαρμόζεται η κατωφλίωση θα σημειώσω το πως χρησιμοποιείται μέσα στον κώδικα και γιατί:

Δεδομένης της εξίσωσης της οπτικής ροής

$$Ix * u + Iy * v + It = 0 \quad (6.1)$$

Κατωφλιώνει την κίνηση ως προς χ και y με τον εξής τρόπο:

$$\text{Αν}\,(\kappa\alpha\tau\acute{\omega}\phi\lambda\iota)^2 > \mathrm{I}_x^2 + \mathrm{I}_y^2 \qquad (6.2)$$

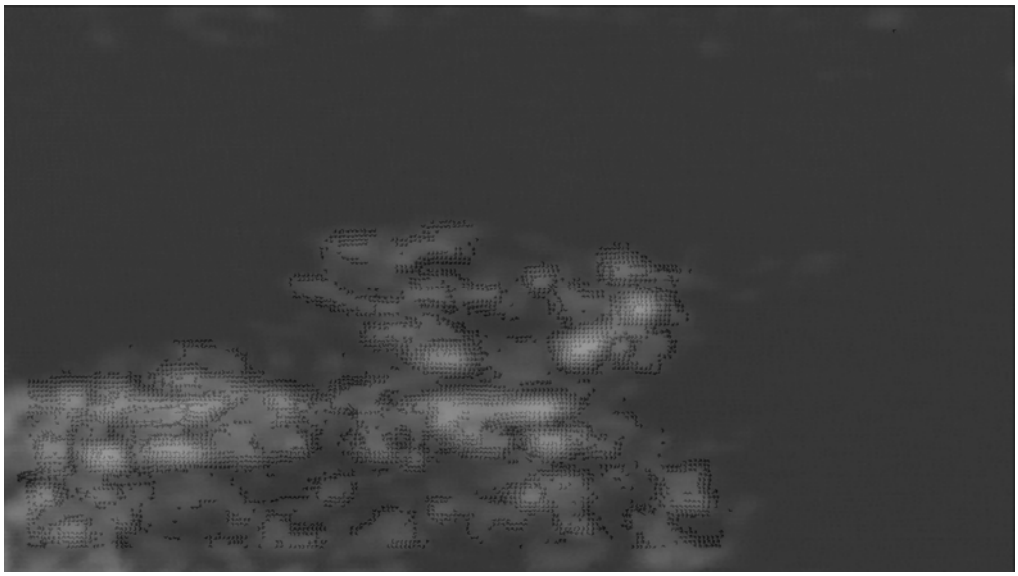
τότε μην λάβεις υπόψη την ταχύτητα σε αυτό το σημείο.

Η κατωφλίωση έγινε με την εντολή:

```
$ ./horn t1_t2_simple. 0.5 0.5 9 20
../croped/cardiac1_0001_t11_t14_crop/framescardiac1_0001_t11_t14_cro
p.avi_t1_t2.avi result -PGM -T 3.5  >t1_t2_simple2.txt
```

**Σχήμα 6.25** Το αποτέλεσμα της κατωφλίωσης

Αν υπερθέσουμε το αποτέλεσμα της οπτικής πάνω στην εικόνα 9 για την οποία έχουμε υπολογίσει την οπτική ροή θα πάρουμε το εξής αποτέλεσμα:



**Σχήμα 6.26** Υπέρθεση αποτελέσματος οπτικής ροής πάνω στην εικόνα 9

Αν επιλέξουμε μόνο το μέρος της πλάκας θα παρατηρήσουμε καλύτερα το πως φαίνεται η οπτική ροή πάνω στην πλάκα:

**Σχήμα 6.27** Επιλογή μέρους της πλάκας για καλύτερη παρατήρηση κίνησης.

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| t1_t2_simple.9 | 30.768013 | 16.058172 | 100 | 0.062557 | 85.929764 |
| t1_t2_simple.9 | 28.520704 | 15.439303 | 15.267439 | 0.275536 | 79.755295 |

**Πίνακας 6.13** Τα στατιστικά δεδομένα με κατωφλίωση 3.5

Παρατηρούμε τώρα πως υπάρχει βελτίωση στον υπολογισμό της οπτικής ροής αφού το λάθος μειώθηκε κατά 2%. Για να δούμε αν η κατωφλίωση είναι ένας τρόπος για να μειώσουμε το λάθος για να το εφαρμόσουμε και στα υπόλοιπα βίντεο θα συγκρίνουμε για την πλάκα αυτή για κάθε εικόνα τα στατιστικά δεδομένα για κατωφλίωση 0.5, 1.5, 2.5, 3.5 και χωρίς κατωφλίωση.
Η γενική εντολή που χρησιμοποιείται σε όλες τις επαναλήψεις είναι η πιο κάτω:

```
$ ./horn t1_t2_simple. 0.5 0.5 9 20
../croped/cardiac1_0001_t11_t14_crop/framescardiac1_0001_t11_t14_cro
p.avi_t1_t2.avi result -PGM
```

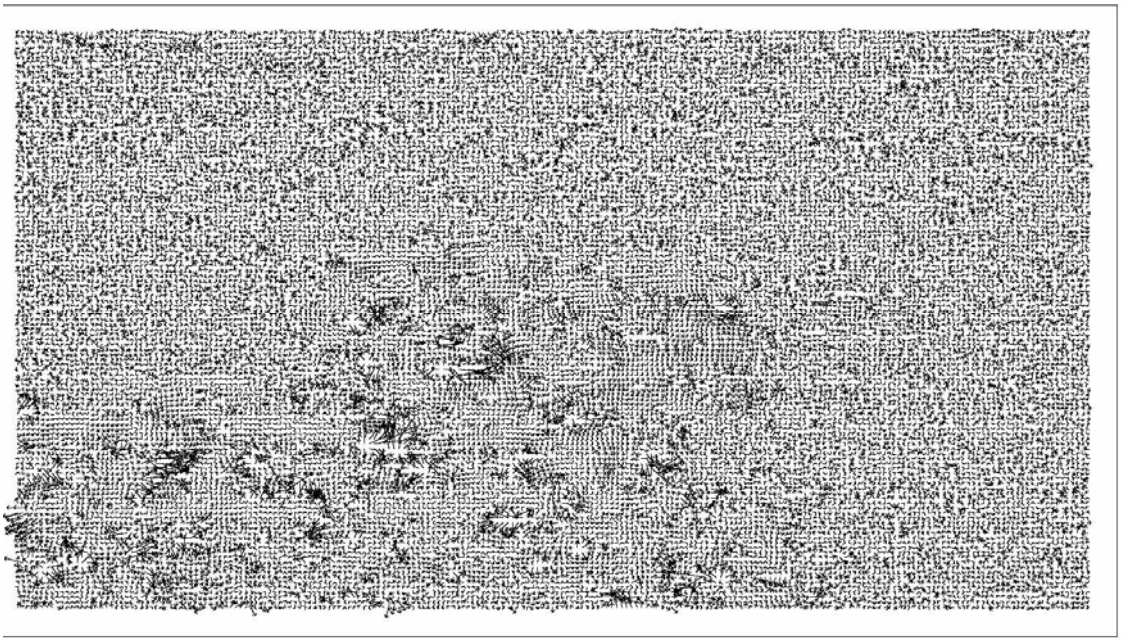
| Εικόνα | | Computed Statistics | | | | |
|---|---|---|---|---|---|---|
| | | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| T_0 | t1_t2_simple.9 | 30.768013 | 16.058172 | 100.000 | 0.062557 | 85.929764 |
| T_0.5 | | 31.132004 | 16.496899 | 73.555473 | 0.062557 | 85.538063 |
| T_1.5 | | 31.718307 | 17.111757 | 30.553587 | 0.062557 | 82.954735 |
| T_2.5 | | 30.192474 | 16.299166 | 21.535902 | 0.062557 | 82.334152 |
| T_3.5 | | 28.520704 | 15.439303 | 15.267439 | 0.275536 | 79.755295 |
| | | | | | | |
| T_0 | t1_t2_simple.10 | 31.564863 | 16.397015 | 100.000 | 0.096913 | 85.639160 |
| T_0.5 | | 32.027020 | 16.850525 | 73.27385 | 0.096913 | 85.639160 |
| T_1.5 | | 32.964836 | 17.422112 | 30.68553 | 0.096913 | 82.643280 |
| T_2.5 | | 31.383797 | 16.463024 | 21.68261 | 0.096913 | 80.608063 |
| T_3.5 | | 29.700678 | 15.720729 | 15.50080 | 0.096913 | 80.608063 |
| | | | | | | |
| T_0 | t1_t2_simple.11 | 30.860270 | 16.143349 | 100.0000 | 0.092787 | 86.015778 |
| T_0.5 | | 31.285545 | 16.528875 | 73.20394 | 0.092787 | 86.015778 |

| | | | | | | |
|---|---|---|---|---|---|---|
| T_1.5 | | 32.056515 | 17.073284 | 30.50238 | 0.092787 | 83.349411 |
| T_2.5 | | 30.355640 | 15.995273 | 21.36555 | 0.092787 | 79.541878 |
| T_3.5 | | 28.820177 | 15.159111 | 15.15026 | 0.092787 | 77.817123 |
| | | | | | | |
| T_0 | t1_t2_simple.12 | 28.635122 | 15.216942 | 100.0000 | 0.148038 | 84.267700 |
| T_0.5 | | 28.600115 | 15.522531 | 73.21083 | 0.148038 | 84.267700 |
| T_1.5 | | 27.345297 | 15.607519 | 30.51321 | 0.148038 | 81.232872 |
| T_2.5 | | 25.428062 | 14.560266 | 21.47189 | 0.148038 | 77.490585 |
| T_3.5 | | 23.882149 | 13.664008 | 15.34621 | 0.148038 | 77.490585 |
| | | | | | | |
| T_0 | t1_t2_simple.13 | 29.142576 | 15.190546 | 100.00000 | 0.000000 | 82.071053 |
| T_0.5 | | 29.192642 | 15.464832 | 72.67320 | 0.000000 | 81.528130 |
| T_1.5 | | 28.393284 | 15.570748 | 30.51026 | 0.000000 | 76.940155 |
| T_2.5 | | 26.479801 | 14.456971 | 21.44629 | 0.000000 | 75.401459 |
| T_3.5 | | 25.032881 | 13.643538 | 15.4899 | 0.000000 | 69.133171 |

**Πίνακας 6.14** Ο συγκριτικός πίνακας συγκρίνει για κάθε πλαίσιο τις παρατηρήσεις για το λάθος, τυπική απόκλιση, ελάχιστο λάθος και μέγιστο λάθος γωνίας.Οι μετρήσεις έγιναν με κατωφλίωση και χωρίς. Οι τιμές κατωφλίωσης φαίνονται παό την παράμετρο T_χ, όπου χ=0.5, 1.5, 2.5, 3.5.

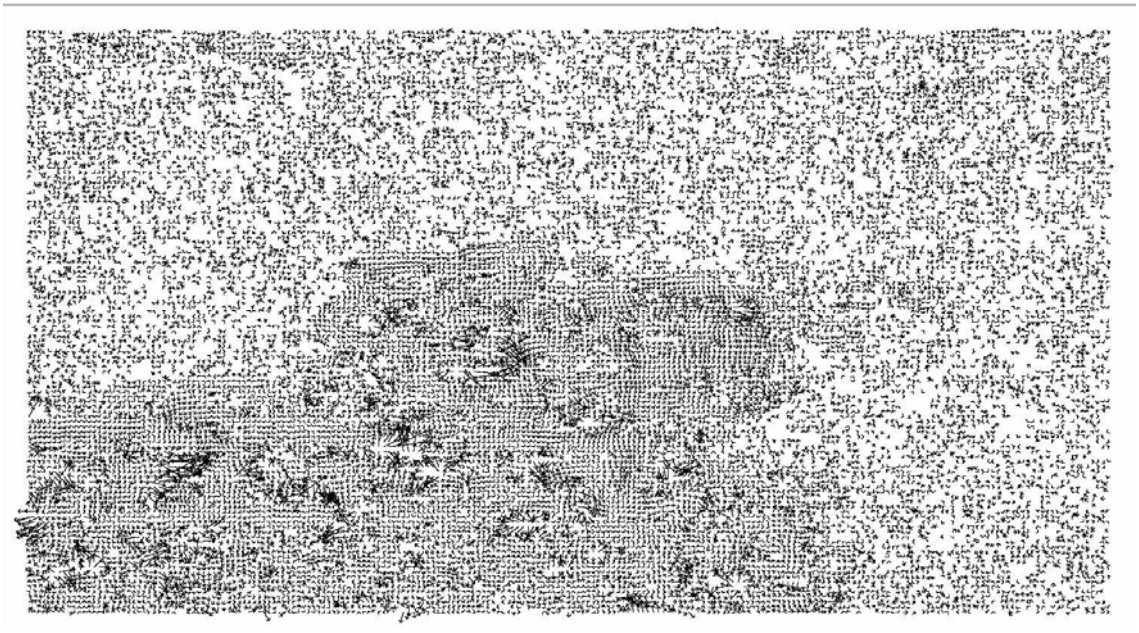
Για να αντιληφθούμε καλύτερα το πως επηρεάζει η κατωφλίωση, δίνω εδώ την γραφική αναπαράσταση της οπτικής ροής για κάθε επίπεδο κατωφλίωσης.

T_0:



**Σχήμα 6.28** Γραφική αναπαράσταση οπτικής ροής χωρίς κατωφλίωση

T_0.5:

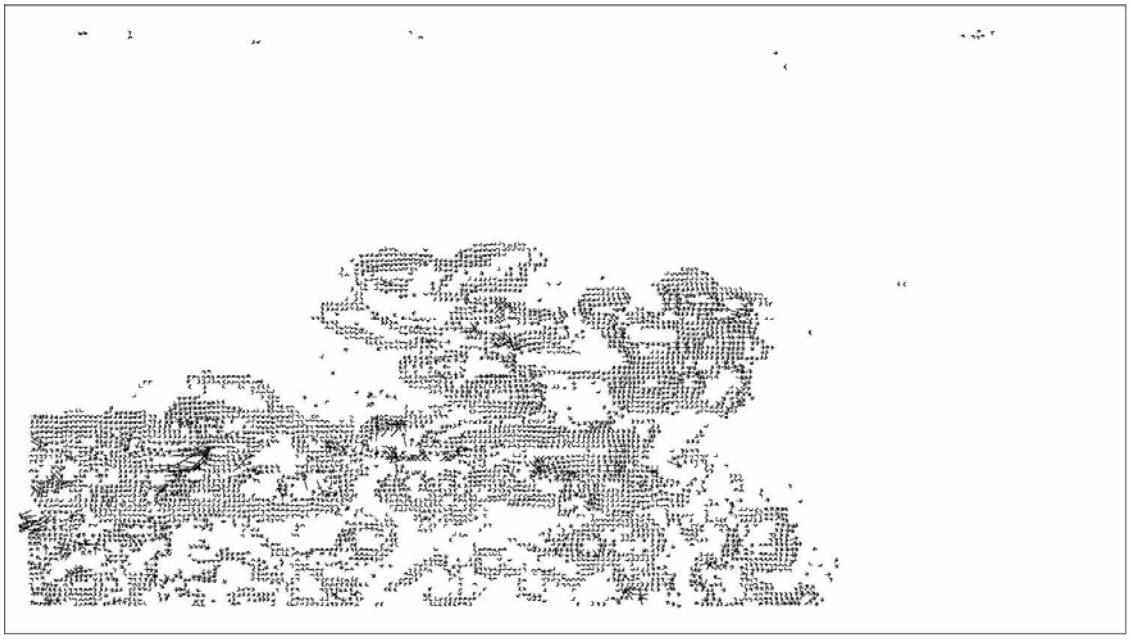


**Σχήμα 6.29** Γραφική αναπαράσταση οπτικής ροής με κατωφλίωση 0.5

T_1.5:

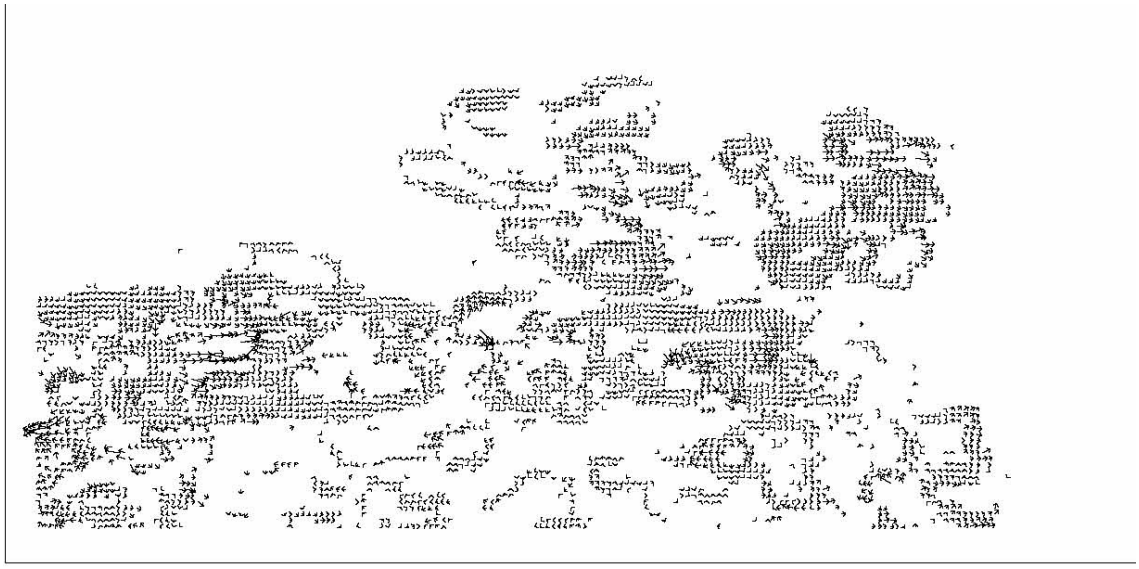
**Σχήμα 6.30** Γραφική αναπαράσταση οπτικής ροής με κατωφλίωση 1.5

T_2.5:



**Σχήμα 6.31** Γραφική αναπαράσταση οπτικής ροής με κατωφλίωση 2.5

T_3.5:



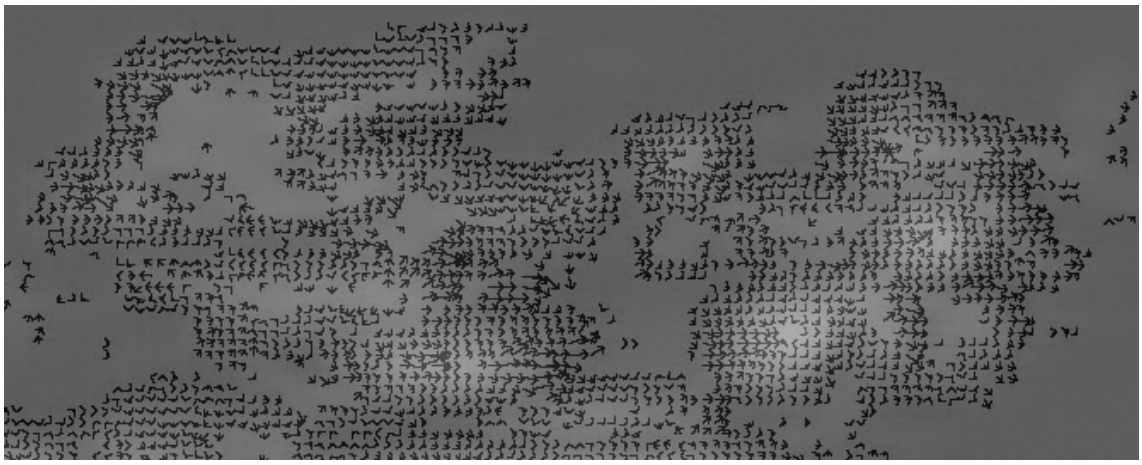
**Σχήμα 6.32** Γραφική αναπαράσταση οπτικής ροής με κατωφλίωση 3.5

**Παλμός 2-3**

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| T2_t3_simple.9 | 31.126579 | 16.595524 | 24.440702 | 0.155766 | 81.334167 |
| T2_t3_simple.10 | 27.897076 | 15.712643 | 24.880854 | 0.128204 | 79.677284 |
| T2_t3_simple.11 | 27.250982 | 14.998837 | 25.263893 | 0.150658 | 78.225952 |
| T2_t3_simple.12 | 27.600996 | 14.947286 | 25.525818 | 0.128204 | 75.034317 |
| T2_t3_simple.13 | 27.269917 | 14.914660 | 25.555359 | 0.148038 | 76.332466 |

**Πίνακας 6.15** Αποτελέσματα παλμού 2-3

Για την εικόνα 13 η οπτική ροή όταν υπερτεθεί πάνω στην αρχική εικόνα παίρνουμε το πιο κάτω αποτέλεσμα:



**Σχήμα 6.33** Υπέρθεση, επιλογή και μεγέθυνση εικόνας 13 στον παλμό 2-3.

Για κάθε σημείο πάνω στην εικόνα, η έξοδος της υλοποίησης μας επιτρέπει να δούμε την ταχύτητα του. Η ταχύτητα μετριέται σε pixel/frame.

## 6.5 Κατωφλίωση φωτεινοτήτων

Στην προσπάθεια για περιορισμό του λάθους, έχουμε δοκιμάσει την κατωφλίωση στις τιμές φωτεινότητας μιας πλάκας. Ο κώδικας που χρησιμοποιήσαμε και τα scripts, μετέτρεπαν αυτόματα τις τιμές φωτεινότητας από ένα κάτω όριο και ένα άνω όριο σε 0 ή 255 ανάλογα αν ήταν μέσα στο πεδίο τιμών που είχαμε ορίσει. Η τεχνική της κατωφλίωσης των φωτεινοτήτων περιορίζει σημαντικά το λάθος.

Συγκεκριμένα:

| Εικόνα | Computed Statistics | | | | |
|---|---|---|---|---|---|
| | Error | St Dev | Density | Minimum angle error | Maximum angle error |
| t1_t2_threshold.13 | 6.942002 | 16.695005 | 100.000000 | 0.000000 | 89.636559 |

**Πίνακας 6.16** Στα αποτελέσματα κατωφλίωσης τιμών φωτεινότητας, παρατηρούμε μείωση του λάθους.

Λόγω της φύσης του προβλήματος η ιδέα για κατωφλίωση των τιμών φωτεινοτήτων δεν θα δοκιμαστεί στο παρόν στάδιο λόγω του ότι το υλικό της πλάκας έχει δύο φύσεις. Η

μία, είναι η σκληρή και η άλλη η μαλακή. Η σκληρή πλάκα είναι σταθερή και έχει μικρές πιθανότητες να αποκολληθεί από το τοίχωμα. Η σκληρή πλάκα έχει σκούρο χρώμα.
Από την άλλη, εχουμε τις μαλακές και ανοιχτόχρωμες πλάκες που δεν είναι σταθερές και υπάρχει μεγάλη πιθανότητα να αποκολληθούν.

Για την κατωφλίωση έχει υλοποιηθεί λογισμικό στην γλώσσα C το οποίο υπολογίζει και παρουσιάζει όλες τις τιμές φωτεινότητας με τον αριθμό εμφάνισής τους σε μια εικόνα, καθώς επίσης την μέγιστη και ελάχιστη τιμή. Ο κώδικας αυτός βρίσκεται στο παράρτημα 1.

# Κεφάλαιο 7

## Συμπεράσματα και Μελλοντική Εργασία



### 7.1 Συμπεράσματα

Για κάθε βήμα της διπλωματικής εργασίας, μπορούμε να καταλήξουμε σε συμπεράσματα που είτε βασίστηκαν σε υπάρχουσα έρευνα, είτε σε κάτι καινούριο.

- Όσο αφορά την ψηφιοποίηση του αναλογικού βίντεο, έχουμε καταλήξει στο συμπέρασμα πως δεν είναι απαραίτητο να χρησιμοποιούμε την καλύτερη δυνατή ποιότητα σε υλικό για να πετύχουμε την καλύτερη δυνατή ποιότητα σε αρχεία βίντεο. Το γεγονός πως έχουμε στην διάθεσή μας βίντεο το οποίο μπορεί να αναπαρασταθεί από τιμές φωτεινότητας 0-255, μας επιτρέπει να χρησιμοποιήσουμε composite αντί component. Το γεγονός αυτό μας εφησυχάζει για τον λόγο ότι υπάρχει ασυμβατότητα με ορισμένους συνδυασμούς υλικού και λογισμικού σε ότι αφορά το component. Το γεγονός πως δεν το χρειαζόμαστε, αλλά είναι αρκετό το composite μας επέτρεψε να συνεχίσουμε στο στάδιο της επεξεργασίας του βίντεο. Ο αριθμός των πλαισίων ανά δευτερόλεπτο πρέπει να είναι 25 για να συμφωνεί με το πρότυπο PAL.
- Σε ότι αφορά την επιλογή λογισμικού για επεξεργασία βίντεο, το λογισμικό mplayer που χρησιμοποιήσαμε και στην σύλληψη, αποδείχτηκε να είναι το καλύτερο από πλευράς ποιότητας, αξιοπιστίας και επαναχρησιμοποίησης. Το γεγονός πως δεν υπάρχει γραφικό περιβάλλον εργασίας, θεωρείται θετικό γιατί μπορέσαμε να δημιουργήσουμε ένα απλό γραφικό περιβάλλον με μόνο ότι χρειαζόμαστε για την επεξεργασία.
- Για την μορφή των εικόνων που αποτελούν το βίντεο, επιλέξαμε να χρησιμοποιήσουμε την PGM για τον λόγο ότι είναι εύκολη μορφή για εκμάθηση και χρήση. Επίσης, δεν επιβάλλει κάποιας μορφής συμπίεσης, ενώ επιπρόσθετα είναι προσαρμοσμένη για εικόνες με τιμές φωτεινότητας της κλίμακας του γκρίζου.

- Στην προσπάθεια να δημιουργήσουμε γραφικό περιβάλλον για το λογισμικό mplayer/mencoder δοκιμάσαμε την χρήση Delphi, Visual Basic και Matlab. Η χρήση της Visual Basic αποδείχτηκε να είναι η πιο εύκολη και γρήγορη.
- Χρειάζεται μεγάλη προσοχή στην ψηφιοποίηση, καθότι όταν έγινε ψηφιοποίηση σε Windows Movie Maker, αν και η ποιότητα του βίντεο ήταν αρκετά καλή στο μάτι, εντούτοις στο βίντεο εφαρμόστηκε συμπίεση μεγάλου βαθμού, πράγμα που στην επεξεργασία βίντεο θεωρείται μεινέκτημα. Ο λόγος είναι ότι μπορεί να χαθεί πληροφορία όταν εφαρμοστεί συμπίεση(συμπίεση με απώλειες) και επίσης κατά την επεξεργασία θα χρειαζόμαστε κάθε φορά να συμπιέζουμε, να αποσυμπιέζουμε πράγμα που αυξάνει τον χρόνο επεξεργασίας.
- Για την επεξεργασία της πλάκας, δεν χρειαζόμαστε ολόκληρο το βίντεο. Μπορούμε κάλλιστα να αποκόψουμε μόνο την πλάκα και να την επεξεργαστούμε. Αυτό θα περιορίσει τα στατιστικά λάθους αφού θα επικεντρωθούμε για την μείωση του λάθους μόνο σε περιοχές που μας ενδιαφέρουν.
- Δεν πρέπει να γίνεται κατωφλίωση σε τιμές φωτεινότητας που δεν είμαστε σίγουροι ότι δεν χάνουμε χρήσιμη πληροφορία.
- Χρησιμοποιώντας κατωφλίωση των συνισταμένων Ix και Iy, περιορίζει το συνολικό λάθος στον υπολογισμό της οπτικής ροής.

## 7.2 Μελλοντική Εργασία

Όπως έχει ήδη αναφερθεί στην εισαγωγή του προβλήματος υπάρχουν πολλοί τομείς που πρέπει να γίνει έρευνα για την σωστή επίλυση του προβλήματος. Πιο συγκεκριμένα:

- Σύγκριση απόδοσης τεχνικών εκτίμησης κίνησης για επιλογή βέλτιστης υλοποίησης σύμφωνα με τις ιδιαιτερότητες των βίντεο που υπάρχουν στην διάθεσή μας.
- Εκτίμηση κίνησης η οποία να βασίζεται στην δυναμική των δομικών συστατικών. Αυτό θα απαιτεί περισσότερη εμβάθυνση από πλευράς του φυσικομαθηματικού υπόβαθρου που πρέπει να έχει ο αναλυτής των βίντεο.
- Υπολογισμός και μείωση θορύβου.

- Νευρωνικά δίκτυα και η αυτόματη αναγνώριση κίνησης και επιλογής πλάκας. Σύγκριση με υπάρχουσα γνώση και υπολογισμός πιθανότητας να φύγει κομμάτι.
- Σταθερότητα κατασκευής. Ακολουθίες Fibonacci και μέθοδοι που χρησιμοποιούνται από πολιτικούς μηχανικούς για τον υπολογισμό της σταθερότητας.
- Τρισδιάστατη μοντελλοποίηση της πλάκας με χρήση γραφικών.
- Επέκταση του επεξεργαστή βίντεο χωρίς την χρήση συστατικών.
- Δημιουργία βάσης δεδομένων για διατήρηση των στατιστικών στοιχείων κάθε βίντεο για χρήση και υπολογισμό κίνησης, λάθους κτλ.
- Επεξεργασία σε YUV, RGB χρωματοχώρο.

Είναι εμφανές πως το θέμα της επεξεργασίας βίντεο για πρόβλεψη εγκεφαλικών είναι πολυδιάστατο, πράγμα που δίνει την δυνατότητα σε πολλά άτομα να ψάξουν την διάσταση που τους ενδιαφέρει για να προσφέρουν στην συνολική επίλυση του προβλήματος.

# Βιβλιογραφία


[1] American Heart Association, “Heart and Stroke Facts”

[2] J.L. Barron, D.J. Fleet and S.S Beauchemin, “Performance of Optical Flow Techniques”, IJCV pp. 43-77, 1994

[3] Barry G. Haskell, Atul Puri, and Arun N. Netraveli, “Digital Video: An introduction to mpeg-2”, Digital Multimedia Standard Series, ITP

[4] Horn and Schunck, , (1981), “Determining Optical Flow”, AI 17, pp. 185-204

[5] Μάριος Παττίχης, “Διαλέξεις Ψηφιακή Επεξεργασία Εικόνας, Κεφάλαιο 10”

[6] Murat Tekalp , “Digital Video Processing”, Prentice Hall Signal Processing Series.

[7] Xenophon Papademetris and Peter N. Belhumeur, “Estimation of motion boundary location and optical flow using dynamic programming”, IEEE ICIP 1996

[8] Yao Wang, John Ostermann, Ya-Qin Zhang, “Video Processing and Communication”, Prentice Hall, Signal Processing Series, Alan V. Oppenheim Series Editor

[9] http://www.refurbished-laptops.biz,

[10] http://www.teleteaching.gr/w3/text-2-1.htm, “Αναλογικά Σήματα Βίντεο”

[11] http://netpbm.sourceforge.net/doc/pgm.html, “PGM format Specification”

[12] http://www.ucy.ac.cy, “Πανεπιστήμιο Κύπρου”

# Παράρτημα Α

Στο παράρτημα αυτό παρατίθεται ο κώδικας σε γλώσσα C, για τον υπολογισμό της οπτικής ροής [2], καθώς επίσης και για άλλες λειτουργίες όπως υπολογισμού τιμών φωτεινότητας, κατωφλίωση, μετατροπή σε PGM.

### Κώδικας σε C

Κώδικας υλοποίησης αλγορίθμου Horn and Schunck

### Horn.c

```
/***********************************************************
  horn.c
  Travis Burkitt, modified by John Barron, 1992
  May  1988  -- written for MASSCOMP
   -- May 8, 1989  modified for SUN 3/60
   --- Implementation of Horn and Schunck's algorithm
          for calculating an optical flow field.
***********************************************************/
#include  <fcntl.h>
#include  <stdio.h>
#include  <math.h>
#include "rasterfile.h"


#define BORDER 2
#define HEAD 32
#define PI  M_PI
#define PIC_X 250
#define PIC_Y 450
#define PIC_T 9
#define FIVE 5
#define PMODE 0644
#define TRUE 1
#define FALSE 0
#define HEAD 32
#define NO_VALUE 100.0
```

```
#define OUTPUT_SMOOTH 1
#define BIG_MAG 1000000.0


unsigned char inpic[PIC_T][PIC_X][PIC_Y];
unsigned char pic[FIVE][PIC_X][PIC_Y];
float floatpic[FIVE][PIC_X][PIC_Y];
unsigned header[HEAD];
float Ix[PIC_X][PIC_Y],Iy[PIC_X][PIC_Y];
float It[PIC_X][PIC_Y],full_vels[PIC_X][PIC_Y][2];
float correct_vels[PIC_X][PIC_Y][2],full_vels1[PIC_X][PIC_Y][2];
float diff_x(),diff_y(),diff_t(),difference(),temp_vels[PIC_X][PIC_Y][2];
float PsiER(),norm(),alpha,fmin1();
int pic_x,pic_y,pic_t,THRESHOLD,STANDARD,BINARY,int_size_x,int_size_y, PGM;
float actual_x,actual_y,size_x,size_y,offset_x,offset_y;
int startx,starty,endx,endy,step,WRITE_SMOOTH;


/*********************************************************************/
/*   Main program                                                    */
/*********************************************************************/
main(argc,argv)
int argc;
char **argv;
{
int fd,fdf,offset,size,start,end,middle,i,j,num;
int fd_correct,no_bytes,numpass,time;
float ave_error,st_dev,density,min_angle,max_angle,sigma,tau;
unsigned char header[HEAD],path1[100],path2[100],path3[100];
char full_name[100],norm_name[100],correct_filename[100];

if(argc < 7 || argc > 19)
        {
     printf("Usage: %s <filename stem> <alpha> <sigma> <central file number> <number of iterations> <input path> <output path> [-S <smooth path> -C <full correct filename> -B <cols> <rows> -T <tau> -H or -MH]\n\n",argv[0]);
        printf("<filename stem> - stem of input filenames\n");
        printf("<alpha> - Lagrange multiplier value\n");
        printf("<sigma> - the standard deviation used in smoothing the files\n");
        printf("   sigma==0.0 means no smoothing: use 2 or 5 unsmoothed images\n");
```

```
            printf("<central file number> - the image file number for which\n");
            printf("   image velocity is to be computed\n");
            printf("<number of iterations> - number of iterations in the velocity calculation\n");
            printf("<input path> - directory where input data resides\n");
            printf("<output path> - directory where computed flow fields put\n");
            printf("-S <smooth path> - directory where smoothed data is to be put\n");
            printf("   if not present smoothed files are not written\n");
            printf("-B <cols> <rows> - use binary file of size <cols>*<rows> characters\n");
            printf("   instead of black and white rasterfiles as image input\n");
            printf("   image size read from rasterfile header if used\n");
            printf("-C <correct filename> - perform error analysis using correct velocity field\n");
            printf("-T <tau> - threshold the computed velocities on spatial intensity gradient\n");
            printf("-H Standard Horn and Schunck\n");
            printf("   perform H and S differencing on 2 input images\n");
            printf(" sigma is ignored for -H\n");
            printf("-MH Non-standard Horn and Schunck - default\n");
            printf("    perform 4-point central differences on 5 input images\n");
            exit(1);
        }

printf("\n-------------------------------\n");
printf("Command line: ");
for(i=0;i<argc;i++) printf("%s ",argv[i]);
printf("\n%d arguments\n",argc-1);

sscanf(argv[2],"%f",&alpha);
sscanf(argv[3],"%f",&sigma);
sscanf(argv[4],"%d",&middle);
sscanf(argv[5],"%d",&numpass);
printf("alpha=%f\n",alpha);
printf("sigma=%f\n",sigma);
printf("Central image: %d\n",middle);
printf("Number of iterations: %d\n",numpass);
strcpy(path1,argv[6]);
strcpy(path2,".");
strcpy(path3,argv[7]);

printf("Input directory: %s\n",path1);
printf("Output directory: %s\n",path3);
 PGM = FALSE;
```

```
 BINARY = FALSE;
 STANDARD = FALSE;
 THRESHOLD = FALSE;
 WRITE_SMOOTH = FALSE;
i=8;
strcpy(correct_filename,"unknown");
while(i<argc)
{
if(strcmp("-H",argv[i])==0)
        {
        STANDARD = TRUE;
        i++;
        }
else
if(strcmp("-MH",argv[i])==0)
        {
        STANDARD = FALSE;
        i++;
        }
else
if(strcmp("-C",argv[i])==0)
        {
        strcpy(correct_filename,argv[i+1]);
        i+=2;
        }
else
if(strcmp("-T",argv[i])==0)
        {
        sscanf(argv[i+1],"%f",&tau);
        i+=2;
        THRESHOLD = TRUE;
        }
else
if(strcmp("-S",argv[i])==0)
        {
        strcpy(path2,argv[i+1]);
        WRITE_SMOOTH = TRUE;
        i+=2;
        }
else
```

```
if(strcmp("-B",argv[i])==0)
        {
        sscanf(argv[i+1],"%d",&pic_y);
        sscanf(argv[i+2],"%d",&pic_x);
        BINARY = TRUE;
        i += 3;
        }

 else
if(strcmp("-PGM",argv[i])==0)
        {
         PGM = TRUE;
        i += 1;
        }
else
        {
        printf("Invalid option %s specified as argument %d -  program terminates\n",argv[i],i);
        exit(1);
        }
}

if(WRITE_SMOOTH) printf("Smoothed data directory: %s\n",path2);
else printf("Smoothed images not written\n");
fflush(stdout);

printf("Correct velocity file: %s\n",correct_filename);
fflush(stdout);
if(strcmp(correct_filename,"unknown")!=0)
{
/* Read the correct velocity data */
if((fd_correct = open(correct_filename,O_RDONLY))==NULL)
        {
        printf("Fatal error in opening file %s\n",correct_filename);
        printf("fd_correct: %d\n",fd_correct);
        exit(1);
        }
no_bytes = 0;
no_bytes += read(fd_correct,&actual_y,4);
no_bytes += read(fd_correct,&actual_x,4);
no_bytes += read(fd_correct,&size_y,4);
```

```
no_bytes += read(fd_correct,&size_x,4);
no_bytes += read(fd_correct,&offset_y,4);
no_bytes += read(fd_correct,&offset_x,4);
if(offset_x != 0.0 || offset_y != 0.0 || actual_x != size_x || actual_y != size_y)
        {
        printf("Fatal error: something wrong with correct velocity data\n");
        printf("Actual y: %f Actual x: %f\n",actual_y,actual_x);
        printf("Size y: %f Size x: %f\n",size_y,size_x);
        printf("Offset y: %f Offset x: %f\n",offset_y,offset_x);
        exit(1);
        }
int_size_y = size_y;
int_size_x = size_x;
for(i=0;i<int_size_x;i++)
        no_bytes += read(fd_correct,&correct_vels[i][0][0],int_size_y*8);
printf("\nFile %s opened and read\n",correct_filename);
printf("Size of correct velocity data: %d %d\n",int_size_y,int_size_x);
printf("%d bytes read\n",no_bytes);
fflush(stdout);
}

if(!STANDARD)
{
size = 6*sigma+1;
if(size%2==0) size = size+1;
offset = size/2+2; /* Add 2 as neighbourhood size offset */
start = middle-offset;
end = middle+offset;
printf("Size: %d Offset: %d Start: %d End: %d\n",size,offset,start,end);
printf("%d images required\n",end-start+1);
if((end-start+1) > PIC_T)
        {
        printf("Fatal error: not enough room for the images\n");
        exit(1);
        }
if(end < start)
        {
        printf("Fatal error: specify images in ascending order\n");
        exit(1);
        }
```

```
read_and_smooth3D(path1,argv[1],sigma,floatpic,pic,inpic,start,middle,end,header);
if(sigma!=0.0 && WRITE_SMOOTH)
writefiles(path2,argv[1],pic,sigma,pic_t,pic_x,pic_y,middle-2,middle+2,header);
}
else
{
start = middle;
end = middle+1;
offset = 2;
readfiles(path1,argv[1],inpic,pic_t,&pic_x,&pic_y,start,end,header);
/* Copy the input images unchanged */
for(i=0;i<pic_x;i++)
for(j=0;j<pic_y;j++)
        {
        pic[0][i][j] = inpic[0][i][j];
        pic[1][i][j] = inpic[1][i][j];
        pic[2][i][j] = 0;
        pic[3][i][j] = 0;
        pic[4][i][j] = 0;
        floatpic[0][i][j] = (int) inpic[0][i][j];
        floatpic[1][i][j] = (int) inpic[1][i][j];
        floatpic[2][i][j] = 0.0;
        floatpic[3][i][j] = 0.0;
        floatpic[4][i][j] = 0.0;
        }
}

printf("Number of Columns: %d Number of Rows: %d\n",pic_y,pic_x);
if(pic_x > PIC_X || pic_y > PIC_Y)
        {
        printf("Fatal error: images are too big\n");
        exit(1);
        }

printf("\n");
fflush(stdout);
if(STANDARD==TRUE)
{
if(THRESHOLD) sprintf(full_name,"%s/horn.original.%sF-%4.2f",path3,argv[1],tau);
```

```
else sprintf(full_name,"%s/horn.original.%sF",path3,argv[1]);
}
else
{
if(THRESHOLD) sprintf(full_name,"%s/horn.modified.%sF-%4.2f",path3,argv[1],tau);
else sprintf(full_name,"%s/horn.modified.%sF",path3,argv[1]);
}

printf("Output full velocities go to file %s\n",full_name);
printf("\n");
fflush(stdout);

startx = 0;
starty = 0;
endx = pic_x-1;
endy = pic_y-1;

startx += BORDER;
starty += BORDER;
endx -= BORDER;
endy -= BORDER;

num = 2;
step = 1;

time = 0;
if(STANDARD)
        {
        calcIx(Ix,floatpic,time);
        calcIy(Iy,floatpic,time);
        calcIt(It,floatpic,time);
        }
else compute_ders(Ix,Iy,It,floatpic,pic_t,pic_x,pic_y,offset);
if(FALSE)
{
if(STANDARD) printf("Standard Horn and Schunck Derivatives\n");
else printf("4 point central difference derivatives\n");
print_ders(Ix,Iy,It);
}
```

```
for(i=0;i<PIC_X;i++)
for(j=0;j<PIC_Y;j++)
        {
        full_vels[i][j][0] = full_vels[i][j][1] = 0.0;
        full_vels1[i][j][0] = full_vels1[i][j][1] = 0.0;
        }

/* Perform iterations */
for(i=0;i<numpass;i+=2)
        {
        printf("%3dth iteration\n",i);
        fflush(stdout);
        calc_vels(full_vels,full_vels1,Ix,Iy,It);
        printf("The improvement: %f\n",difference(full_vels,full_vels1,pic_x,pic_y));
        fflush(stdout);
if(strcmp(correct_filename,"unknown")==0)
        {
        rearrange(full_vels1,temp_vels);
        calc_statistics(correct_vels,int_size_x,int_size_y,temp_vels,
                pic_x,pic_y,2*offset,&ave_error,&st_dev,&density,&min_angle,&max_angle);
        printf("Error: %f St Dev: %f Density: %f\n",ave_error,st_dev,density);
        fflush(stdout);
        }
        printf("%3dth iteration\n",i+1);
        calc_vels(full_vels1,full_vels,Ix,Iy,It);
        printf("The improvement: %f\n",difference(full_vels,full_vels1,pic_x,pic_y));
if(strcmp(correct_filename,"unknown")==0)
        {
        rearrange(full_vels,temp_vels);
        calc_statistics(correct_vels,int_size_x,int_size_y,temp_vels,
                pic_x,pic_y,2*offset,&ave_error,&st_dev,&density,&min_angle,&max_angle);
        printf("Error: %f St Dev: %f Density: %f\n",ave_error,st_dev,density);
        fflush(stdout);
        }
        }
if(THRESHOLD) threshold(full_vels,Ix,Iy,tau,pic_x,pic_y);

if((fdf=creat(full_name,0644))!=NULL)
        output_velocities(fdf,"Full",full_vels,pic_x,pic_y,2*offset);
else printf("Error in opening %s file\n\n",full_name);
```

```
printf("\nVelocities computed and output\n");
fflush(stdout);

if(strcmp(correct_filename,"unknown")==0)
{
calc_statistics(correct_vels,int_size_x,int_size_y,full_vels,
    pic_x,pic_y,2*offset,&ave_error,&st_dev,&density,&min_angle,&max_angle);
printf("\n\nComputed Statistics\n");
printf("Error: %f St Dev: %f\n",ave_error,st_dev);
printf("Density: %f\n",density);
printf("Minimum angle error: %f Maximum angle error: %f\n",min_angle,max_angle);
}
fflush(stdout);
}



/*****************************************************************/
/* Compute x derivatives                                         */
/*****************************************************************/
calcIx(Ex,floatpic,t)
float Ex[PIC_X][PIC_Y];
float floatpic[FIVE][PIC_X][PIC_Y];
int t;
{
int i,j;

printf("****** calculating Ex ******\n");
for(i=startx;i<=endx;i+=step)
for(j=starty;j<=endy;j+=step)
        {
        Ex[i][j] = (floatpic[t][i+step][j] + floatpic[t][i+step][j+step] +
                    floatpic[t+1][i+step][j] + floatpic[t+1][i+step][j+step])/4.0
                   -(floatpic[t][i][j] + floatpic[t][i][j+step] +
                    floatpic[t+1][i][j] + floatpic[t+1][i][j+step])/4.0;
        if(Ex[i][j] > BIG_MAG)
                {
                printf("Ex too large at i=%d j=%d Ex=%f\n",i,j,Ex[i][j]);
                exit(1);
                }
```

```
        }
}


/*****************************************************************/
/* Compute y derivatives                                         */
/*****************************************************************/
calcIy(Ey,floatpic,t)
float Ey[PIC_X][PIC_Y];
float floatpic[FIVE][PIC_X][PIC_Y];
int t;
{
int i,j;

printf("****** calculating Ey ******\n");
for(i=startx;i<=endx;i+=step)
for(j=starty;j<=endy;j+=step)
        {
        Ey[i][j] = (floatpic[t][i][j+step] + floatpic[t][i+step][j+step] +
                    floatpic[t+1][i][j+step] + floatpic[t+1][i+step][j+step])/4.0
                   -(floatpic[t][i][j] + floatpic[t][i+step][j] +
                    floatpic[t+1][i][j] + floatpic[t+1][i+step][j])/4.0;
        if(Ey[i][j] > BIG_MAG)
                {
                printf("Ey too large at i=%d j=%d Ey=%f\n",i,j,Ey[i][j]);
                exit(1);
                }
        }
}

/*****************************************************************/
/* Compute t derivatives                                         */
/*****************************************************************/
calcIt(Et,floatpic,t)
float Et[PIC_X][PIC_Y];
float floatpic[FIVE][PIC_X][PIC_Y];
int t;
{
int i,j;
```

```
printf("****** calculating Et ******\n");
for(i=startx;i<=endx;i+=step)
for(j=starty;j<=endy;j+=step)
        {
        Et[i][j] = (floatpic[t+1][i][j] + floatpic[t+1][i+step][j] +
                    floatpic[t+1][i][j+step] + floatpic[t+1][i+step][j+step])/4.0
                   -(floatpic[t][i][j] + floatpic[t][i+step][j] +
                    floatpic[t][i][j+step] + floatpic[t][i+step][j+step])/4.0;
        if(Et[i][j] > BIG_MAG)
                {
                printf("Et too large at i=%d j=%d Ex=%f\n",i,j,Et[i][j]);
                exit(1);
                }
        }
}


/*****************************************************************/
/* Compute average u value in neighbour about i,j                */
/*****************************************************************/
vels_avg(vels,ave)
float vels[PIC_X][PIC_Y][2],ave[PIC_X][PIC_Y][2];
{
int i,j;
for(i=startx+1;i<endx;i++)
for(j=starty+1;j<endy;j++)
        {
        ave[i][j][0] = (vels[i-1][j][0]+vels[i][j+1][0]+
                            vels[i+1][j][0]+vels[i][j-1][0])/6.0 +
                       (vels[i-1][j-1][0]+vels[i-1][j+1][0]+
                        vels[i+1][j+1][0]+vels[i+1][j-1][0])/12.0;
        ave[i][j][1] = (vels[i-1][j][1]+vels[i][j+1][1]+
                            vels[i+1][j][1]+vels[i][j-1][1])/6.0 +
                       (vels[i-1][j-1][1]+vels[i-1][j+1][1]+
                            vels[i+1][j+1][1]+vels[i+1][j-1][1])/12.0;
        }

/* Copy average values of neighbourhoods for boundaries */
for(i=startx;i<=endx;i++)
        {
```

```
            ave[i][starty][0] = ave[i][starty+1][0];
            ave[i][endy][0] = ave[i][endy-1][0];
            ave[i][starty][1] = ave[i][starty+1][1];
            ave[i][endy][1] = ave[i][endy-1][1];
            }
for(j=starty+1;j<=endy;j++)
            {
            ave[startx][j][0] = ave[startx+1][j][0];
            ave[endx][j][0] = ave[endx-1][j][0];
            ave[startx][j][1] = ave[startx+1][j][1];
            ave[endx][j][1] = ave[endx-1][j][1];
            }
/* Corner Points */
ave[startx][0][0] = ave[startx+1][1][0];
ave[startx][0][1] = ave[startx+1][1][1];
ave[startx][endy][0] = ave[startx+1][endy-1][0];
ave[startx][endy][1] = ave[startx+1][endy-1][1];
ave[endx][0][0] = ave[endx-1][1][0];
ave[endx][0][1] = ave[endx-1][1][1];
ave[endx][endy][0] = ave[endx-1][endy-1][0];
ave[endx][endy][1] = ave[endx-1][endy-1][1];
}



/*****************************************************************/
/* Compute u,v values                                            */
/*****************************************************************/
calc_vels(vels,vels1,Ex,Ey,Et)
float vels[PIC_X][PIC_Y][2],vels1[PIC_X][PIC_Y][2];
float Ex[PIC_X][PIC_Y],Ey[PIC_X][PIC_Y],Et[PIC_X][PIC_Y];
{
int i,j,k;
float mag,ave[PIC_X][PIC_Y][2];

printf("****** Computing Velocity ******\n");
fflush(stdout);
vels_avg(vels1,ave);
for(i=startx;i<=endx;i+=step)
for(j=starty;j<=endy;j+=step)
```

```
            {
             vels[i][j][0] = ave[i][j][0]-Ex[i][j]*
               (Ex[i][j]*ave[i][j][0]+Ey[i][j]*ave[i][j][1]+Et[i][j])
               /(alpha*alpha+Ex[i][j]*Ex[i][j]+Ey[i][j]*Ey[i][j]);
             vels[i][j][1] = ave[i][j][1]-Ey[i][j]*
               (Ex[i][j]*ave[i][j][0]+Ey[i][j]*ave[i][j][1]+Et[i][j])
               /(alpha*alpha+Ex[i][j]*Ex[i][j]+Ey[i][j]*Ey[i][j]);
             /* if((int)vels[i][j][0]!=0)
                printf("Vels(%d,%d,0) %d \n",i,j,(int)vels[i][j][0]);
                if((int)vels[i][j][1]!=0)
                printf("Vels(%d,%d,1) %d \n",i,j,(int)vels[i][j][1]);
             */
             mag = sqrt(vels[i][j][0]*vels[i][j][0]+vels[i][j][1]*vels[i][j][1]);
             if(mag > 5.0 && FALSE)
               {
                printf("Velocity magnitude of %f at %d %d is over 5.0\n",mag,i,j) ;
               }
            }
}



/*****************************************************************/
/* Print derivative information                                  */
/*****************************************************************/
print_ders(Ex,Ey,Et)
float Ex[PIC_X][PIC_Y],Ey[PIC_X][PIC_Y],Et[PIC_X][PIC_Y];
{
int i,j;

printf("********************  Ex  ********************\n");
for(i=50;i<=60;i+=step)
            {
            for(j=50;j<=60;j+=step) printf("%5.1f ",Ex[i][j]);
            printf ("\n");
            }

printf("********************  Ey  ********************\n");
for(i=50;i<=60;i+=step)
            {
```

```
            for(j=50;j<=60;j+=step) printf("%5.1f ",Ey[i][j]);
            printf("\n");
            }

printf("********************  Et  ********************\n");
for(i=50;i<=60;i+=step)
            {
            for(j=50;j<=60;j+=step) printf("%5.1f ",Et[i][j]);
            printf("\n");
            }
}




/*************************************************************************/
/* Compute spatio-temporal derivatives                                   */
/*************************************************************************/
compute_ders(Ix,Iy,It,floatpic,pic_t,pic_x,pic_y,n)
float Ix[PIC_X][PIC_Y];
float Iy[PIC_X][PIC_Y];
float It[PIC_X][PIC_Y];
float floatpic[PIC_T][PIC_X][PIC_Y];
int n,pic_t,pic_x,pic_y;
{
int i,j;
float kernel[5];

for(i=0;i<PIC_X;i++)
for(j=0;j<PIC_X;j++)
            {
            Ix[i][j] = Iy[i][j] = It[i][j] = 0.0;
            }

calc_diff_kernel(kernel);
for(i=n;i<pic_x-n;i++)
for(j=n;j<pic_y-n;j++)
            {
            Ix[i][j] = diff_x(floatpic,kernel,i,j,2);
            Iy[i][j] = diff_y(floatpic,kernel,i,j,2);
```

```
            It[i][j] = diff_t(floatpic,kernel,i,j,2);
            /* printf("i:%d j:%d Ix:%f Iy:%f It:%f\n",i,j,Ix[i][j],Iy[i][j],It[i][j]); */
            }
printf("Spatio-Temporal Intensity Derivatives Computed\n");
fflush(stdout);
}

/*****************************************************************************/
/* Compute a 4 point central difference kernel                               */
/*****************************************************************************/
calc_diff_kernel(diff_kernel)
float diff_kernel[5];
{
diff_kernel[0] = -1.0/12.0;
diff_kernel[1] = 8.0/12.0;
diff_kernel[2] = 0.0;
diff_kernel[3] = -8.0/12.0;
diff_kernel[4] = 1.0/12.0;
}


/************************************************************
  Apply 1D real kernels in the x direction
************************************************************/
float diff_x(floatpic,kernel,x,y,n)
float floatpic[PIC_T][PIC_X][PIC_Y];
float kernel[5];
int x,y,n;
{
int i;
float sum;

sum = 0.0;
for(i=(-n);i<=n;i++)
            {
            sum += kernel[i+2]*floatpic[2][x+i][y];
            }
return(sum);
}
```

```
/************************************************************
  Apply 1D real kernels in the y direction
************************************************************/
float diff_y(floatpic,kernel,x,y,n)
float floatpic[PIC_T][PIC_X][PIC_Y];
float kernel[5];
int x,y,n;
{
int i;
float sum;

sum = 0.0;
for(i=(-n);i<=n;i++)
        {
        sum += kernel[i+2]*floatpic[2][x][y+i];
        }
return(sum);
}


/************************************************************
  Apply 1D real kernels in the t direction.
************************************************************/
float diff_t(floatpic,kernel,x,y,n)
float floatpic[PIC_T][PIC_X][PIC_Y];
float kernel[5];
int x,y,n;
{
int i;
float sum;

sum = 0.0;
for(i=(-n);i<=n;i++)
        {
        sum += kernel[i+2]*floatpic[2+i][x][y];
        }
return(sum);
}
```

```
/*************************************************************************/
/* Read input images and perform Gaussian smoothing.                     */
/*************************************************************************/
readfiles(path,s,pic,pic_t,pic_x,pic_y,start,end,header)
char s[100],path[100];
int pic_t,*pic_x,*pic_y,start,end;
unsigned char header[HEAD];
unsigned char pic[PIC_T][PIC_X][PIC_Y];
{
char fname[100];
int i,j,k,fp,fd,time,no_bytes;
unsigned char temp_save[PIC_X][PIC_Y];
int ONCE;
int ints[8];
 int M,N,Q;
 char *ptr;
 char headers[100];
 FILE *fm;
 int l=0;
printf("Reading Files...\n");
time = -1;
if((end-start) < 0)
            {
             printf("\nSpecified time for writing file incorrect\n");
             exit(1);
            }
 ONCE = TRUE;
 for(i=start;i<=end;i++)
   {
    time++;
    no_bytes = 0;
    sprintf(fname,"%s/%s%d",path,s,i);
            if(!BINARY && !PGM)
              {
               if((fp=open(fname,O_RDONLY)) >0)
                 {
                       if(ONCE)
                         {
                          no_bytes += read(fp,ints,HEAD);
                          (*pic_y) = ints[1];
```

```
                (*pic_x) = ints[2];
                ONCE = FALSE;
               }//end of once
              else no_bytes += read(fp,header,HEAD);
          }//end of fopen
              else
                {
                 printf("File %s does not exist in readfiles.\n",fname);
                 exit(1);
                }


        for(j=0;j<(*pic_x);j++)
          no_bytes += read(fp,&pic[time][j][0],(*pic_y));
        printf("File %s read (%d bytes)\n",fname,no_bytes);
        no_bytes = 0;
        fflush(stdout);
         }//end of !BINARY !PGM

        if(BINARY){
         if((fp=open(fname,O_RDONLY)) >0){
        for(j=0;j<(*pic_x);j++)
          no_bytes += read(fp,&pic[time][j][0],(*pic_y));
        printf("File %s read (%d bytes)\n",fname,no_bytes);
        no_bytes = 0;
        fflush(stdout);
         }
         else
           {
                 printf("File %s does not exist in readfiles.\n",fname);
                 exit(1);
                }
        }//end of if BINARY

        /*****************************************************************/
        if(PGM){
          printf("PGM detected\n");
if ((fm=fopen(fname,"r")) == NULL) {
  fprintf(stderr,"Can't open %s.\n",fname);
  exit(1);
```

```
 }

 fgets(headers,100,fm);
 if ( (headers[0]!=80) ||    /* 'P' */
      (headers[1]!=53) ) {   /* '5' */
      fprintf(stderr,"Image %s is not PGM.\n",fname);
      exit(1);
 }

 fgets(headers,100,fm);
 while(headers[0]=='#')
   fgets(headers,100,fm);

 M=strtol(headers,&ptr,0);
 N=atoi(ptr);

 fgets(headers,100,fm);

 Q=strtol(headers,&ptr,0);
 (*pic_x)=N;
 (*pic_y)=M;

 if (fread(&pic[time], (M*N), 1, fm) == 0) {
            fflush(stdout);
   fprintf(stderr,"Image %s is wrong size.\n",fname);
   exit(1);
 }

            }//end of if PGM
            /*****************************************************************/

  }//end of for
} /* End of readfiles */


/*********************************************************************/
/*   Write smoothed files                                            */
/*********************************************************************/
writefiles(path,s,result,sigma,pic_t,pic_x,pic_y,start,end,header)
char s[100],path[100];
```

```
float sigma;
int pic_t,pic_x,pic_y,start,end;
unsigned char result[FIVE][PIC_X][PIC_Y];
unsigned char header[HEAD];
{
char fname[100];
int i,j,k,fp,time,no_bytes;

printf("\nWriting smoothed files...\n");

time = -1;
for (i=start;i<=end;i++)
        {
        no_bytes = 0;
        time++;
        sprintf(fname,"%s/smoothed.%s%d-%3.1f",path,s,i,sigma);
        if((fp=creat(fname,0644))!=NULL)
                {
                no_bytes += write(fp,&header[0],HEAD); /* Write 32 byte raster header */
                for(j=0;j<pic_x;j++)
                        no_bytes += write(fp,&result[time][j][0],pic_y);
                printf("File %s written (%d bytes)\n",fname,no_bytes);
                }
        else
                {
                printf("File %s cannot be written\n",fname);
                exit(1);
                }
        fflush(stdout);
        close(fp);
        }
} /* End of writefiles */


/*****************************************************************/
/* Smooth the image sequence using a 3D separable Gaussian filter.  */
/*****************************************************************/
read_and_smooth3D(path,stem,sigma,floatpic,pic,inpic,start,middle,end,header)
char stem[100],path[100];
float sigma;
```

```
unsigned char pic[FIVE][PIC_X][PIC_Y];
unsigned char inpic[PIC_T][PIC_X][PIC_Y];
unsigned char header[HEAD];
float floatpic[FIVE][PIC_X][PIC_Y];
int start,end,middle;
{
int fd,n,size,i,j,k,time,frame;
char name[100];

for(k=0;k<FIVE;k++)
for(i=0;i<PIC_X;i++)
for(j=0;j<PIC_Y;j++)
        floatpic[k][i][j] = 0.0;

pic_t = end-start+1;


readfiles(path,stem,inpic,pic_t,&pic_x,&pic_y,start,end,header);

printf("Number of columns: %d Number of rows: %d\n",pic_y,pic_x);
if(pic_x > PIC_X || pic_y > PIC_Y)
        {
        printf("Fatal error: images are too big\n");
        exit(1);
        }
fflush(stdout);

time = -1;
frame = middle-2;
if(OUTPUT_SMOOTH) printf("Start: %d End: %d Middle: %d\n",start,end,middle);
for(frame=middle-2;frame<=middle+2;frame++)
        {
        time++;
        convolve_Gaussian(inpic,floatpic,pic,sigma,pic_t,pic_x,pic_y,
                start,middle-2+time,time);
        }
if(sigma!=0.0) printf("Input files smoothed\n");
}

/*********************************************************************/
```

```
/* Perform 3D Gaussian smoothing by separable convolution                */
/************************************************************************/
convolve_Gaussian(inpic,floatpic,pic,sigma,pic_t,pic_x,pic_y,start,frame,time)
unsigned char inpic[PIC_T][PIC_X][PIC_Y];
unsigned char pic[FIVE][PIC_X][PIC_Y];
float floatpic[FIVE][PIC_X][PIC_Y];
int pic_x,pic_y,pic_t,frame,time,start;
float sigma;
{
float mask[100],term,product,sum;
int size,i,j,k,offset,a,b;
float pic0[PIC_X][PIC_Y],pic1[PIC_X][PIC_Y];


if(OUTPUT_SMOOTH)
{
printf("\nStart of 3D convolution\n");
printf("Time: %d Frame: %d\n",time,frame);
}

fflush(stdout);
size = (int) 6*sigma+1;
if(size%2==0) size = size+1;
offset = size/2;
if(pic_t < size)
        {
        printf("\nFatal error: not enough images\n");
        exit(1);
        }
sum = 0.0;

if(sigma != 0.0)
for(i=0;i<size;i++)
        {
        mask[i] = (1.0/(sqrt(2.0*3.141592654)*sigma))*
                exp(-(i-offset)*(i-offset)/(2.0*sigma*sigma));
        sum = sum+mask[i];
        }
else { mask[0] = 1.0; sum = 1.0; }
```

```
if(OUTPUT_SMOOTH && sigma!=0.0)
{
printf("Size: %d Offset: %d\nMask values: ",size,offset);
for(i=0;i<size;i++)
        printf("%f ",mask[i]);
printf("\nSum of mask values: %f\n",sum);
}
for(i=0;i<pic_x;i++)
for(j=0;j<pic_y;j++)
        pic0[i][j] = pic1[i][j] = 0.0;


if(sigma != 0.0)
{
for(i=0;i<pic_x;i++)
for(j=0;j<pic_y;j++)
        {
        term = 0.0;
        for(a=-offset;a<=offset;a++)
                {
                term = term +(inpic[a+frame-start][i][j]*mask[a+offset]);
                }
        pic0[i][j] = term;
        }
if(OUTPUT_SMOOTH) printf("Convolution in t direction completed\n");
for(i=offset;i<pic_x-offset;i++)
for(j=offset;j<pic_y-offset;j++)
        {
        term = 0.0;
        for(a=-offset;a<=offset;a++)
                {
                term = term + (pic0[i+a][j]*mask[a+offset]);
                }
        pic1[i][j] = term;
        }
if(OUTPUT_SMOOTH) printf("Convolution in x direction completed\n");

for(i=offset;i<pic_x-offset;i++)
for(j=offset;j<pic_y-offset;j++)
        {
```

```
            term = 0.0;
            for(b=-offset;b<=offset;b++)
                        {
                        term = term + (pic1[i][j+b])*mask[b+offset];
                        }
            floatpic[time][i][j] = term;
            if(term > 255.0) term = 255.0;
            if(term < 0.0) term = 0.0;
            pic[time][i][j] = (int) (term+0.5);
            }
if(OUTPUT_SMOOTH) printf("Convolution in y direction completed\n");
}
else /* No smoothing */
{
printf("No smoothing: frame=%d frame-start=%d time=%d\n",frame,frame-start,time);
fflush(stdout);
for(i=0;i<pic_x;i++)
for(j=0;j<pic_y;j++)
            {
            pic[time][i][j] = inpic[frame-start][i][j];
            floatpic[time][i][j] = inpic[frame-start][i][j];
            }
}

if(OUTPUT_SMOOTH) printf("End of Convolution\n");
fflush(stdout);
}


/*************************************************************
  Output full velocities using old Burkitt format
*************************************************************/
output_velocities(fdf,s,full_velocities,pic_x,pic_y,n)
float full_velocities[PIC_X][PIC_Y][2];
char s[100];
int fdf,n;
{
float x,y;
int i,j,bytes,no_novals,no_vals,NORMAL;
```

```
NORMAL = FALSE;
if(strcmp(s,"Normal")==0) NORMAL = TRUE;
if(fdf==NULL)
        {
        printf("\nFatal error: full velocity file not opened\n");
        exit(1);
        }
/* original size */
y = pic_x;
x = pic_y;
write(fdf,&x,4);
write(fdf,&y,4);

/* size of result data */
y = pic_x-2*n;
x = pic_y-2*n;
write(fdf,&x,4);
write(fdf,&y,4);

/* offset to start of data */
y = n;
x = n;
write(fdf,&x,4);
write(fdf,&y,4);
bytes = 24;

no_novals = no_vals = 0;
/* Prepare velocities for output, i.e. rotate by 90 degrees */

 printf("FULL VELOCITIES START NOW \n");
for(i=n;i<pic_x-n;i++)
for(j=n;j<pic_y-n;j++)
        {
        if(full_velocities[i][j][0] != NO_VALUE &&
          full_velocities[i][j][1] != NO_VALUE)
                {
                no_vals++;
                x = full_velocities[i][j][0];
                y = full_velocities[i][j][1];
                printf("Velocities [%d][%d] \t x: %f \t",i, j, y);
```

```
                printf("y: %f\n", x);

                full_velocities[i][j][0] = y;
                full_velocities[i][j][1] = -x;
                }
        else
                {
                no_novals++;
                }
        }
for(i=n;i<pic_x-n;i++)
        {
        bytes += write(fdf,&full_velocities[i][n][0],(pic_y-2*n)*8);
        }
close(fdf);
printf("\n%s velocities output from output_velocities: %d bytes\n",s,bytes);
printf("Number of positions with velocity: %d\n",no_vals);
printf("Number of positions without velocity: %d\n",no_novals);
printf("Percentage of %s velocities: %f\n",s,
        no_vals/(1.0*(no_vals+no_novals))*100.0);
fflush(stdout);
}


/*************************************************************/
/*  Compute error statistics                                 */
/*************************************************************/
calc_statistics(correct_vels,int_size_x,int_size_y,full_vels,
        pic_x,pic_y,n,ave_error,st_dev,density,min_angle,max_angle)
float full_vels[PIC_X][PIC_Y][2],*ave_error,*density,*st_dev;
float correct_vels[PIC_X][PIC_Y][2],*min_angle,*max_angle;
int n,pic_x,pic_y,int_size_x,int_size_y;
{
int full_count,no_full_count,i,j,a,b,total_count;
float sumX2,temp,uva[2],uve[2];

full_count = no_full_count = total_count = 0;
sumX2 = 0.0;
(*min_angle) = HUGE_VAL;
(*max_angle) = -HUGE_VAL;
```

```
(*ave_error) = (*st_dev) = (*density) = 0.0;
for(i=n;i<pic_x-n;i++)
{
for(j=n;j<pic_y-n;j++)
            {
             /*  if((int)full_vels[i][j][0]!=0 && (int)full_vels[i][j][0]!=100)
              printf("Vels(%d,%d,0) %d \n",i,j,(int)full_vels[i][j][0]);
             if((int)full_vels[i][j][1]!=0 && (int)full_vels[i][j][1]!=100)
               printf("Vels(%d,%d,1) %d \n",i,j,(int)full_vels[i][j][1]);
             */

            if(full_vels[i][j][0] != NO_VALUE && full_vels[i][j][1] != NO_VALUE)
             {
             full_count++;
             uve[0] = full_vels[i][j][0]; uve[1] = full_vels[i][j][1];
             uva[0] = correct_vels[i][j][0]; uva[1] = correct_vels[i][j][1];
             temp = PsiER(uve,uva);
             (*ave_error) += temp;
             sumX2 += temp*temp;
             if(temp < (*min_angle)) (*min_angle) = temp;
             if(temp > (*max_angle)) (*max_angle) = temp;
             }
            else no_full_count++;
            total_count++;
            }
}

if(full_count != 0) (*ave_error) = (*ave_error)/full_count;
else (*ave_error) = 0.0;
if(full_count > 1)
{
temp = fabs((sumX2 - full_count*(*ave_error)*(*ave_error))/(full_count-1));
(*st_dev) = sqrt(temp);
}
else (*st_dev) = 0.0;
(*density) = full_count*100.0/(total_count*1.0);

if((*ave_error) == 0.0) { (*min_angle) = (*max_angle) = 0.0; }

if(FALSE)
```

```
{
printf("\nIn calc_statistics\n");
printf("%d full velocities\n",full_count);
printf("%d positons without full velocity\n",no_full_count);
printf("%d positions in total\n",total_count);
fflush(stdout);
}
}

/*************************************************************
 Full Image Velocity Angle Error
*************************************************************/
float PsiER(ve,va)
float ve[2],va[2];
{
float nva;
float nve;
float v,r,temp;
float VE[3],VA[3];

VE[0] = ve[0];
VE[1] = ve[1];
VE[2] = 1.0;

VA[0] = va[0];
VA[1] = va[1];
VA[2] = 1.0;

nva = norm(VA,3);
nve = norm(VE,3);
v =  (VE[0]*VA[0]+VE[1]*VA[1]+1.0)/(nva*nve);

/**  sometimes roundoff error causes problems **/
if(v>1.0 && v < 1.0001) v = 1.0;

r = acos(v)*180.0/PI;

if (!(r>=0.0 && r< 180.0))
{
printf("ERROR in PSIER()...\n r=%8.4f  v=%8.4f  nva=%8.4f nve= %8.4f\n",r,v,nva,nve);
```

```
printf("va=(%f,%f) ve=(%f,%f)\n",va[0],va[1],ve[0],ve[1]);
}

return r;
}


/*************************************************************
  returns the norm of a vector v of length n.
*************************************************************/
float norm(v,n)
float v[];
int n;
{
int i;
float sum = 0.0;

for (i=0;i<n; i++)
      sum += (v[i]*v[i]);
sum = sqrt(sum);
return sum;
}

/*************************************************************/
/* Compute the difference between two flow fields      */
/*************************************************************/
float difference(v1,v2,pic_x,pic_y)
float v1[PIC_X][PIC_Y][2],v2[PIC_X][PIC_Y][2];
int pic_x,pic_y;
{
int i,j,n;
float sum,t1,t2;

sum = 0.0;
n = 0;
for(i=0;i<pic_x;i++)
for(j=0;j<pic_y;j++)
            {
            t1 = v1[i][j][0]-v2[i][j][0];
            t2 = v1[i][j][1]-v2[i][j][1];
```

```
            sum += sqrt(t1*t1+t2*t2);
            n++;
            }
sum = sum/n;
return(sum);
}


/*****************************************************************/
/* Threshold the computed velocities on the basis of the spatial  */
/* intensity gradient.                                            */
/*****************************************************************/
threshold(full_vels,Ix,Iy,tau,pic_x,pic_y)
float full_vels[PIC_X][PIC_Y][2],Ix[PIC_X][PIC_Y],Iy[PIC_X][PIC_Y];
float tau;
int pic_x,pic_y;
{
int i,j,count;
float gradient2,tau2;

count = 0;
tau2 = tau*tau;
for(i=0;i<pic_x;i++)
for(j=0;j<pic_y;j++)
            {
            gradient2 = Ix[i][j]*Ix[i][j]+Iy[i][j]*Iy[i][j];
            /* gradient2 = fmin1(Ix[i][j]*Ix[i][j],Iy[i][j]*Iy[i][j]); */
            if(gradient2 < tau2)
                        {
                        count++;
                        full_vels[i][j][0] = NO_VALUE;
                        full_vels[i][j][1] = NO_VALUE;
                        }
            }
printf("Threshold: %f\n",tau);
printf("%d velocities thresholded\n",count);
}

float fmin1(x,y)
float x,y;
```

```
{
if(x<y) return(x);
else return(y);
}

/*************************************************************/
/* Rearrange velocity field v1 (in output format) and place  */
/* the result in v2 so that error analysis can be performed. */
/*************************************************************/
rearrange(v1,v2)
float v1[PIC_X][PIC_Y][2],v2[PIC_X][PIC_Y][2];
{
int i,j;
for(i=0;i<PIC_X;i++)
for(j=0;j<PIC_Y;j++)
        {
        if(v1[i][j][0] != NO_VALUE && v1[i][j][1] != NO_VALUE)
                {
                v2[i][j][0] =  v1[i][j][1];
                v2[i][j][1] = -v1[i][j][0];
                }
        else
                {
                v2[i][j][0] = NO_VALUE;
                v2[i][j][1] = NO_VALUE;
                }
        }
}
```

## Getmaxvalue.c

```
#include <stdio.h>

#include "ReadImage.c"
#include "WriteImage.c"

int main(argc,argv)
int argc;
char *argv[];
{
 int i, j;
 int N, M, Q;
 int **fimage;
 FILE *fp;
 int max_val=0;
 int min_val=255;
 int start=0;
 int end=0;
 int table[256];
 int init=0;
 int count=0;
 int pos1=0;
 int pos2=0;

 int request=0;

 for(init=0;init<256;init++){
  table[init]=0;
 }
 ReadImage(argv[1], &fimage, &M, &N, &Q);


 for(i=0; i<N; i++){
  for(j=0; j<M; j++){
    if(fimage[i][j] > max_val){
      max_val=fimage[i][j];
    }
    if(fimage[i][j] < min_val){
      min_val=fimage[i][j];
```

```
    }
    table[fimage[i][j]]+=1;
  count = count +1;

  if(argc>2){
    pos1=(int)atoi(argv[3]);
    pos2=(int)atoi(argv[2]);
    if((i==pos1)&&(j==pos2))
      request=fimage[i][j];
  }

  }

}


 init=0;
 for(init=0;init<256;init++){
  printf("Value: %d\t number of appearances : %d\n",init, table[init]);
 }

 if(argc>2){
  printf("selected pixel value at(%d,%d) is: %d\n", pos1, pos2,request);
 }
 printf("min value = %d\n", min_val);
 printf("max value = %d\n", max_val);
 printf("total number: %d\n",count);


 return (0);

}
```

**ReadImage.c**

```
void ReadImage(fname, fimage, M, N, Q)
char fname[200];
int ***fimage;
int *M, *N, *Q;
{
 int i,j;
 unsigned char  *image;
 char header [100], *ptr;
 FILE  *fd;

 if ((fd=fopen(fname,"r")) == NULL) {
  fprintf(stderr,"Can't open %s.\n",fname);
  exit(1);
 }

 fgets(header,100,fd);
 if ( (header[0]!=80) ||    /* 'P' */
      (header[1]!=53) ) {   /* '5' */
      fprintf(stderr,"Image %s is not PGM.\n",fname);
      exit(1);
 }

 fgets(header,100,fd);
 while(header[0]=='#')
   fgets(header,100,fd);

 *M=strtol(header,&ptr,0);
 *N=atoi(ptr);

 fgets(header,100,fd);

 *Q=strtol(header,&ptr,0);

 image = (unsigned char *) malloc( ((*M)*(*N)) * sizeof(unsigned char));

 *fimage=(int **)malloc( (*N) * sizeof(int *));
 for(i=0; i < (*N); i++)
   (*fimage)[i]=(int *)malloc( (*M) * sizeof(int));
```

```
 if (fread(image, (*M)*(*N), 1, fd) == 0) {
   fprintf(stderr,"Image %s is wrong size.\n",fname);
   exit(1);
 }

 /* Convert the unsigned characters to integers */

 for(i=0; i < (*N); i++){
   for(j=0; j < (*M); j++){
    (*fimage)[i][j]=(int)image[i*(*M)+j];
    //     if((int)image[i*(*M)+j] !=255)
    // printf("(%d,%d):%d ",i,j, (int)image[i*(*M)+j]);
 }
   //printf("\n");
 }
 free(image);

 fclose(fd);

}
```

**WriteImage.c**

```
void WriteImage(fname, fimage, M, N, Q)
char fname [100];
int **fimage;
int  M, N, Q;
{
 int i, j;
 unsigned char *image;
 FILE  *fd;

 image = (unsigned char *) malloc( (M*N) * sizeof(unsigned char));

 for(i=0; i<N; i++)
   for(j=0; j<M; j++)
     image[i*M+j]=(unsigned char)fimage[i][j];

 if ((fd=fopen(fname,"w")) == NULL) {
   fprintf(stderr,"Can't output %s.\n",fname);
   exit(0);
 }

 fprintf(fd,"P5\n");
 fprintf(fd,"%d %d\n", M, N);
 fprintf(fd,"%d\n", Q);

 if (fwrite(image, M*N, 1, fd) != 1) {
   fprintf(stderr,"Can't write image %s.\n",fname);
   exit(0);
 }

 free(image);

 fclose(fd);

}
```

**Threshold.c**

```
#include <stdio.h>

#include "ReadImage.c"
#include "WriteImage.c"

int main(argc,argv)
int argc;
char *argv[];
{
 int i, j;
 int N, M, Q;
 int **fimage;
 FILE *fp;
 int start=0;
 int end=0;
 ReadImage(argv[1], &fimage, &M, &N, &Q);
 start=(int)atoi(argv[3]);
 end = (int)atoi(argv[4]);
 /* example: thresholding */
 printf("start: %d end: %d \n",start,end);

 for(i=0; i<N; i++)
   for(j=0; j<M; j++)
     if(fimage[i][j] < end && fimage[i][j]>start )
       fimage[i][j] = 0;
     else
       fimage[i][j] = 255;


 WriteImage(argv[2], fimage, M, N, Q);

 return (0);

}
```

# Παράρτημα Β

Στο παράρτημα αυτό, παρατίθεται ο κώδικας script, ο οποίος χρησιμοποιείται για μετατροπές εικόνων καθώς επίσης και για κατωφλίωση/προετοιμασία δεδομένων.

**SCRIPTS**

### Convert_pgm_to_jpg.sh

```
#!bin/bash
COUNTER=0
COUNTER=$(ls -la|grep ".pgm"|awk 'END {print NR}')
for i in $(ls *.pgm| sort -nr)
do
x=(${i%.pgm})
convert $i $x.jpeg
echo x: $x
echo i: $i
echo counter : $COUNTER
let COUNTER-=1
done
```

### convert_to_pgm.sh

```
#!bin/bash
COUNTER=0
COUNTER=$(ls -la|grep ".ras"|awk 'END {print NR}')
for i in $(ls *.ras| sort -nr)
do
x=(${i%.ras})
convert $i $x.pgm
echo x: $x
echo i: $i
echo counter : $COUNTER
let COUNTER-=1
done
```

### prepare_and_threshold.sh

```
#!bin/bash
```

```
echo Please, enter the name for the new images
read NAME

echo Please, enter the minimum threshold
read MIN
echo Please, enter the maximum threshold
read MAX
COUNTER=0
COUNTER=$(ls -la|grep ".pgm"|awk 'END {print NR}')
let COUNTER-=1
for i in $(ls *.pgm| sort -nr)
do
x=(${i%.pgm})
./threshold $i $NAME.$COUNTER $MIN $MAX
echo x: $x
echo i: $i
echo counter : $COUNTER
let COUNTER-=1
done
```

### prepare_no_threshold.sh

```
#!bin/bash

echo Please, enter the name for the new images
read NAME
COUNTER=0
COUNTER=$(ls -la|grep ".pgm"|awk 'END {print NR}')
let COUNTER-=1
for i in $(ls *.pgm| sort -nr)
do
x=(${i%.pgm})
cp $i $NAME.$COUNTER $MIN $MAX
echo x: $x
echo i: $i
echo counter : $COUNTER
let COUNTER-=1
done
```

# Παράρτημα Γ

Στο παράρτημα αυτό παρατίθεται ο κώδικας υλοποίησης του συστήματος σε Visual Basic.

**Κώδικας Visual Basic**

**Project.vbp**

```
Option Explicit


Public Function GetTimes(Time_ As Integer, tipos As Integer)
Static start_time As String
Static end_time As String
Static end_final As String

If tipos = 0 Then
start_time = CStr(Time_)
Text1.Text = start_time
End If

If tipos = 1 Then
end_time = CStr(Time_)
Text2.Text = end_time
End If

If tipos = 3 Then

If CInt(start_time) > CInt(end_time) Then
MsgBox ("error: StartTime of video greater than EndTime of Video. Reselect Video
")
End If



If CInt(start_time) < CInt(end_time) Then
```

```
Dim intFileHandle As Integer
Dim quote As String
Dim timestart1 As String
Dim timestart2 As String
Dim timestart As String
Dim timeend1 As String
Dim timeend2 As String
Dim timeend As String
Dim start_time_integer As Integer
Dim end_time_integer As Integer
Dim end_final_time_integer As Integer


  quote = """"
intFileHandle = FreeFile
timestart1 = GetString(start_time, ",", "F")
timestart2 = GetString(start_time, ",", "B")
timestart = timestart1 '+ "." + timestart2

timeend1 = GetString(end_time, ",", "F")
timeend2 = GetString(end_time, ",", "B")
timeend = timeend1 + "." + timeend2

start_time_integer = CInt(start_time)
end_time_integer = CInt(end_time)
end_final_time_integer = end_time_integer - start_time_integer

end_final = CStr(end_final_time_integer)

Open "mencoder.bat" For Output As #intFileHandle
Print #intFileHandle, quote + "c:\Program Files\Mplayer\mencoder.exe" + quote + "
" + quote + CommonDialog1.FileName + quote + " " + "-ss" + " " + start_time + " " +
"-endpos" + " " + end_final + " " + "-ovc raw -noskip -o" + " " + quote +
```

```
CommonDialog1.FileName + "_t" + start_time + "_" + "t" + end_time + ".avi" + quote
  Text3.Text = CommonDialog1.FileTitle + "_t" + start_time + "_" + "t" + end_time + ".avi"
  Close #intFileHandle
  Shell "mencoder.bat"

End If

End If

End Function


Public Function GetString(tmpStr As String, tmpDiv As String, Mode As String) As String
Dim tmpRetStr As String
Dim tmpLen As Integer
Dim tmpErr As Integer
 ' ******************************************************
 ' tmpString = The whole string
 ' tmpDiv = The Divider chr
 ' if mode = "F" then get the string in front of the div
 ' if mode = "B" then get the string in back of the div
 '
 ' Return values:
 '  Errors
 '    Err1 = wrong mode ("F" & "B" ok)
 '    Err2 =  No Div, did not found a divider
 '    Err3 = No F, did not find any string in front of the div
 '    Err4 = No B, did not find any string in back of the div
 '    Err5 = No String
 '  Normal
 '    String
```

```
' ******************************************************
' Example: GetString("Red=Green","=","F") will retrun "Red"
' *** for string
tmpErr = 0
If Len(tmpStr) = 0 Then
  tmpErr = 5
  GoTo GetStringErr
End If

' *** test for chr in tmpDiv
If Len(tmpDiv) = 0 Then
  tmpErr = 2
  GoTo GetStringErr
End If

' *** test for div in string
If InStr(1, tmpStr, tmpDiv) = 0 Then
  tmpErr = 2
  GoTo GetStringErr
End If

' *** process "F"
If Mode = "F" Then
   tmpRetStr = Left(tmpStr, (InStr(1, tmpStr, tmpDiv) - 1))
   If Len(tmpRetStr) > 0 Then
     GetString = tmpRetStr
     Exit Function
   Else
     tmpErr = 3
     GoTo GetStringErr
   End If
End If

If Mode = "B" Then
```

```
    tmpRetStr = Right(tmpStr, Len(tmpStr) - (InStr(1, tmpStr, tmpDiv)))
    If Len(tmpRetStr) > 0 Then
      GetString = tmpRetStr
      Exit Function
    Else
      tmpErr = 4
      GoTo GetStringErr
    End If

  End If
    tmpErr = 1
    GoTo GetStringErr
  Exit Function
GetStringErr:
GetString = "Err" & tmpErr
End Function
```

```
Public Sub Command1_Click()
CommonDialog1.ShowOpen
 If CommonDialog1.FileName = "" Then Exit Sub
WindowsMediaPlayer1.Url = CommonDialog1.FileName
End Sub

Public Sub Command2_Click()
'Shell "C:\Documents and Settings\Marios\My Documents\Panepistimio\DIPLWMATIKH\Visual Basic\test_menoder.bat"
'Shell "c:\Program Files\Mplayer\mencoder.exe" + CommonDialog1.FileName + "-ss" + VideoStartTime + "-endpos" + VideoEndTime + "-ovc raw -noskip -o" + "C:\Documents and Settings\Marios\My Documents\Panepistimio\DIPLWMATIKH\Visual Basic\selection1_21_Ok.avi"
```

```
'    Dim intFileHandle As Integer
'    Dim quote As String
'    Dim s_StartTime As String
'    Dim s_EndTime As String
'   quote = """"
'    intFileHandle = FreeFile
'    Open "mencoder.bat" For Output As #intFileHandle
'
'    Print #intFileHandle, quote + "c:\Program Files\Mplayer\mencoder.exe" + quote + "" + quote + CommonDialog1.FileName + quote + " " + "-ss" + " " + Form_Load.Command3_Click + " " + "-endpos" + " " + VideoEndTime + " " + "-ovc raw -noskip -o" + " " + quote + "C:\Documents and Settings\Marios\My Documents\Panepistimio\DIPLWMATIKH\Visual Basic\" + CommonDialog1.FileTitle + VideoStartTime + "_" + VideEndTime + ".avi" + quote
'    Close #intFileHandle
'Shell "mencoder.bat"
Dim a As Integer
a = GetTimes(1, 3)



End Sub

Public Sub Command3_Click()
Dim VideoStartTime As Integer
Dim a As Integer
VideoStartTime = WindowsMediaPlayer1.Controls.CurrentPosition
a = GetTimes(VideoStartTime, 0)

End Sub

Public Sub Command4_Click()
Dim VideoEndTime As Integer
Dim a As Integer
```

```
VideoEndTime = WindowsMediaPlayer1.Controls.CurrentPosition

a = GetTimes(VideoEndTime, 1)
End Sub

Private Sub Command5_Click()
Dim frame_dir As String
Dim frame_dir_p1 As String
Dim frame_dir_p2 As String
Dim frame_dir_p3 As String
Dim new_path As String
 Dim quote As String
      quote = """"

frame_dir_p1 = Text1.Text
frame_dir_p2 = Text2.Text
frame_dir_p3 = Text3.Text
frame_dir = CurDir & "\frames" & frame_dir_p3

MkDir (frame_dir)
new_path = frame_dir




 Dim intFileHandle As Integer
  intFileHandle = FreeFile
 Open "extract.bat" For Output As #intFileHandle
   Print #intFileHandle, quote + "c:\Program Files\Mplayer\mplayer.exe" + quote + " "
+ quote + CurDir + "\" + frame_dir_p3 + " " + quote + " -vo" + " " + "pnm:pgm"
   Close #intFileHandle
```

```
FileCopy CurDir + "\extract.bat", new_path + "\extract.bat"
Kill CurDir + "\extract.bat"
    ChDir new_path
   Shell "extract.bat"


End Sub
```

# Παράρτημα Δ

Στο παράρτημα αυτό, παρατίθεται ο κώδικας σε MATLAB για την μετατροπή εικόνων SUN rasterfiles, σε pgm εικόνες.

**Κώδικας σε MATLAB για μετατροπή sun rasterfiles σε pgm εικόνες**

**Im2ras.m**

```
function r = im2ras(A)
% Pairnei to onoma enos directory kai
% to metatrepei ola ta 'pgm' se Sun rasterfile format
% Usage:  im2ras path



% Get all the files from the directory
files = dir(A);


% Find the number of the files in the directory
number_of_file= size(files);
number_of_files=number_of_file(1);
count=0;
for i= 1:number_of_files


  f_name = files(i).name;
  [token,rem] = strtok(f_name,'.');
 f_name_format=rem;
 f_name_filename=token;
 if (strcmp(f_name_format,'.pgm')==1)
    pgm_image= imread(f_name);
    ras_name= strcat(f_name_filename,'.ras');
    imwrite(pgm_image,ras_name,'ras');
   count=count+1;
 end %end of if
end
fprintf(1,' Number of files converted : %d',count);
```

# Παράρτημα Ε

Στο παράρτημα αυτό, παρατίθενται οι πίνακες σύγκρισης λογισμικού και υλικού κατά την επιλογή βέλτιστης μεθοδολογίας ψηφιοποίησης αναλογικού βίντεο.

## Συγκριτικοί πίνακες υλικού και λογισμικού

### Συγκριτικός πίνακας 1

**H/W table**

| **Υλικό** | **Πλεονεκτήματα** | **Μειονεκτήματα** |
|---|---|---|
| Dazzle Digital Video Creator(Laptop) | Εύκολο στην χρήση. | Η χρήση του Dazzle δεν συνίσταται καθώς γίνεται συμπίεση από την θύρα IEEE 1394. |
| Aver Media | Συμβατή κάρτα βίντεο με το LINUX | |
| Matrox Meteor-II frame grabber. | Αναγνωρισμένη κάρτα σύλληψης βίντεο. | Απώλεια πλαισίων σε μεγάλο βαθμό.<br>Χαμηλής ποιότητας γράμματα στο βίντεο.<br>Δεν παρέχει δυνατότητα S-Video In. |

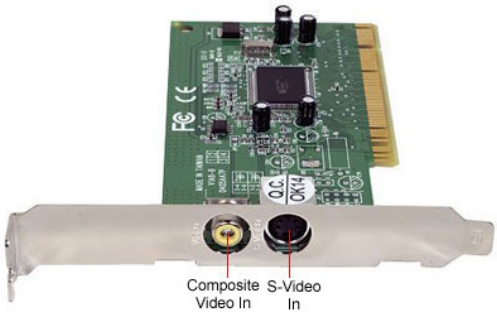
## Συγκριτικός πίνακας 2

| Product name | **Matrox Meteor II** | **SCM Microsystems Dazzle Hollywood DV-Bridge** |
|---|---|---|
| Product image |  |  |
| Main Specs | | |
| Product Description | Matrox Meteor II - video input adapter - PCI | Dazzle Hollywood DV-Bridge video input adapter - FireWire |
| Device Type | Video input adapter - plug-in card | Video input adapter - external |
| Interface Type | PCI | IEEE 1394 (FireWire) |
| Dimensions (WxDxH) | 18.8 cm 10.7 cm | - |
| Video Input | Video capture adapter – plug-in card | Video capture adapter - IEEE 1394 (FireWire) - external |
| Analogue Video Format | NTSC, PAL | NTSC, PAL |
| Analogue video Signal | Composite video | S-Video, composite video |
| Digital Video Format | - | DV |
| Data Transfer Rate | - | 3.1 MBps |
| Video Compression Format | YUV 4:1:1, YUV 4:2:2 | - |
| Audio Input Support | - | Standard |
| Features | - | DV input |
| OS Required | Microsoft Windows 2000 / NT4.0, Microsoft Windows 98/ME | Apple MacOS 9.0.4, Microsoft Windows Millennium Edition, Microsoft Windows 98 Second Edition / Windows 2000 |
| Cables Included | - | 1 x IEEE 1394 cable - external - 1.8 m |
| Manufacturer | - | 1 year warranty |

| | | |
|---|---|---|
| Warranty | | |
| Prices | | |
| Product name | Matrox Meteor II | SCM Microsystems Dazzle Hollywood DV-Bridge |

## Πίνακας υλικού: Aver Media DVD EZ Maker PCI Video Card

**Specifications**




- **Input Signal:** S-Video input
  Composite Video Input
- **DVD Writing Formats:**
  MPEG-2 VBR (5Mbps Average, 8.1Mbps Maximum)
  PCM Audio 48KHz/16 bits 2 Channels
- **VCD Writing Format:**
  MPEG-1 Video 1152 kbps
  MPEG-1 Layer 2 Audio 224 kbps

**Video Input**

| | |
|---|---|
| Video input type | Video capture adapter |
| Analog video signal | S-Video,Composite video |
| Audio Input Support | Standard |
| Digital video capture resolution | 720 x 480 |
| Digital video input format | MPEG-2 |
| Analog video format | NTSC |

**General**

| | |
|---|---|
| Compatibility | PC |

| | | |
|---|---|---|
| | Depth | 4.9 in |
| | Width | 0.7 in |
| | Height | 4.1 in |
| | Weight | 3.2 oz |
| **Expansion / Connectivity** | Expansion Slot(s) Total (Free) | -1,None -1,-1 |
| | Port(s) Total (Free) / Connector Type:Port(s) Total (Free) / Connector Type Port(s) Total (Free) / Connector Type | 1 Display / video S- video input: 4 pin mini- DIN, 1 Display / video Composite video input: RCA,1 Audio Line-in:RCA |
| **Software / System Requirements** | Min operating system | Microsoft Windows 2000 / XP |
| | Min Processor Type | 800 MHz |
| | Interface devices | CD- ROM, Sound card,Graphics card |

| | | |
|---|---|---|
| | Software type | Ulead VideoStudio, Drivers & Utilities, Ulead DVD MovieFactory |
| **Miscellaneous** | Recording Standard:Compliant Standards | CE,FCC Class B certified |

**Πίνακας Λογισμικού**

**S/W table**

| **Λογισμικό** | **Πλεονεκτήματα** | **Μειονεκτήματα** |
|---|---|---|
| Matrox S/W (Matrox Meteor II κάρτα βίντεο) | Καλή ποιότητα στο βίντεο, χωρίς συμπίεση. Διαθέσιμος C++ κώδικας. | Χάσαμε πολλά πλαίσια(frames), Κακή ποιότητα στα γράμματα του βίντεο. |
| Windows Movie Maker (Χρήση με Dazzle) | Εύκολο στην χρήση. Πολύ καλή ποιότητα βίντεο. Τα γράμματα στο βίντεο είναι ευανάγνωστα. | Γίνεται συμπίεση, γι'αυτό δεν ήταν αποδεκτό. Παράδειγμα επαλήθευσης. Για 32 δευτερόλεπτα βίντεο, μέγεθος πλαισίου 640*380, 8bit greyscale, 25 frames το δευτερόλεπτο(PAL standard) υπολογίζεται να απαιτείται περίπου 227 MB, ενώ στην πραγματικότητα με το λογισμικό αυτό παίρνουμ αρχείο μεγέθους 2.683 MB. Είναι εμφανής η |

| | | |
|---|---|---|
| | | συμπίεση. |
| Adobe Premiere ( Χρήση με AverMedia) | Μεγάλο φάσμα δυνατοτήτων και επιλογών για επεξεργασία βίντεο. | Πρόβλημα στην χρήση του. Δεν ήταν εφικτή η σύλληψη βίντεο από την κάρτα βίντεο |
| Linux mplayer/mencoder(Χρήση με AverMedia) | Χωρίς συμπίεση.<br>Πολύ καλή ποιότητα βίντεο.<br>Ευκρίνεια στα γράμματα μέσα στο βίντεο.<br>Η απώλεια σε πλαίσια είναι αμελητέα (περίπου 1-5 frames για κάθε σύλληψη βίντεο) | Δεν υπάρχει διαθέσιμο μέχρι στιγμής γραφικό περιβάλλον για πιο εύκολη χρήση. |