Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Εαρινό εξάμηνο 2022-23

Όραση Υπολογιστών

2η Εργαστηριακή Άσκηση

Εκτίμηση Οπτικής Ροής (Optical Flow), Εξαγωγή Χαρακτηριστικών σε Βίντεο για Αναγνώριση Δράσεων, Συνένωση Εικόνων (Image Stitching)

Δημήτριος Κοκκίνης 03118896

dimkok00@gmail.com

Χριστίνα Ρεντίφη 03118217

christired@gmail.com

Μέρος 1: Παρακολούθηση Προσώπου και Χεριών με Χρήση της Μεθόδου Οπτικής Ροής των Lucas-Kanade

Στο πρώτο μέρος καλούμαστε να αναπτύξουμε ένα σύστημα παρακολούθησης (tracking) περιοχών ενδιαφέροντος σε μια αλληλουχία εικόνων που σχηματίζουν ένα βίντεο. Συγκεκριμένα, η αλληλουχία εικόνων, οι οποίες προέρχονται από ένα απόσπασμα βίντεο νοηματικής γλώσσας, απεικονίζουν μια γυναίκα να κουνάει τα χέρια και το κεφάλι της.

1.1 Ανίχνευση Δέρματος Προσώπου και Χεριών

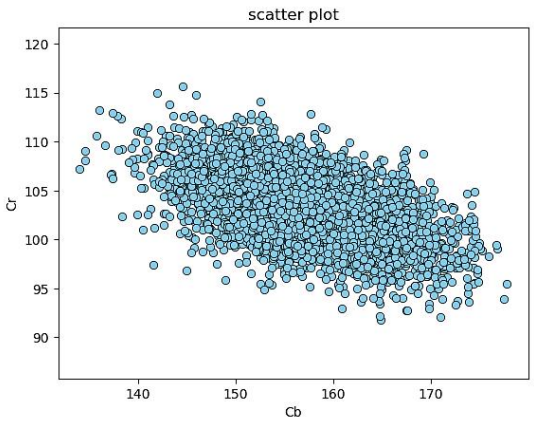
Πρώτο βήμα για την ανάπτυξη του συστήματος, είναι ο αρχικός εντοπισμός των θέσεων των περιοχών ενδιαφέροντος, των περιοχών δηλαδή που χρειάζεται να εστιάσει ο υπολογιστής. Οι περιοχές αυτές είναι το αριστερό χέρι, το δεξί χέρι και το πρόσωπο. Το κοινό χαρακτηριστικό των περιοχών αυτών είναι το δέρμα και συγκεκριμένα το χρώμα του δέρματος.

Η ιδέα της χρησιμοποίησης του χρώματος για να κάνουμε 'mask' τις περιοχές του προσώπου και των χεριών βασίζεται στο γεγονός πως το χρώμα είναι αμετάβλητο σε διαβαθμίσεις, περιστροφές και μετατοπίσεις. Αναλυτικότερα, για τη μοντελοποίηση του χρώματος του δέρματος θα δουλέψουμε με τον χρωματικό χώρο YCbCr, ο οποίος χρησιμοποιεί τη φωτεινότητα (Y), το μπλε στοιχείο σχετικά με το πράσινο (Cb) και το κόκκινο σχετικά με το πράσινο (Cr). Σε αντίθεση με το χρωματικό χώρο RGB, ο οποίος δεν διαχωρίζει το χρώμα από την ένταση της φωτεινότητας, ο YCbCr πλεονεκτεί στη περίπτωση αυτή διότι λόγο του διαχωρισμού που κάνει ανάμεσα στο χρώμα και τη φωτεινότητα, μας δίνει τη δυνατότητα να ανιχνεύσουμε διαφορές στο χρώμα σε διάφορες συνθήκες φωτισμού.

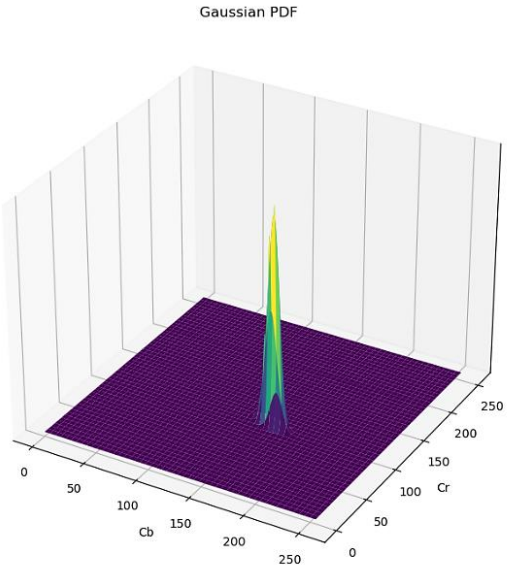
Για να μπορέσει ο υπολογιστής να αποφανθεί για το αν ένα pixel ανήκει ή όχι σε περιοχή δέρματος, θα πρέπει πρώτα να γνωρίζει πως 'μοιάζει' το χρώμα του δέρματος. Αυτή η διαδικασία περιγράφεται από τη μοντελοποίηση του χρώματος του δέρματος από ένα Gaussian μοντέλο

$$P(\mathbf{c} = \text{skin}) = \frac{1}{\sqrt{|\Sigma|\,(2\pi)^2}} e^{-\frac{1}{2}(\mathbf{c}-\boldsymbol{\mu})\Sigma^{-1}(\mathbf{c}-\boldsymbol{\mu})'}$$

το οποίο, μόλις γίνει fit στα δεδομένα χρώματος που μας δίνεται στο υλικό του εργαστηρίου, θα μοντελοποιεί ουσιαστικά τη πιθανότητα το χρώμα του pixel που εξετάζουμε να ανήκει σε χρώμα δέρματος ή όχι. Η εκπαίδευση της Γκαουσιανής γίνεται βρίσκοντας το διάνυσμα της μέσης τιμής και της συνδιακύμανσης από τα δεδομένα δέρματος που μας δίνονται. Στο κώδικα χρησιμοποιήσαμε τις συναρτήσεις της βιβλιοθήκης numpy: mean() και cov().

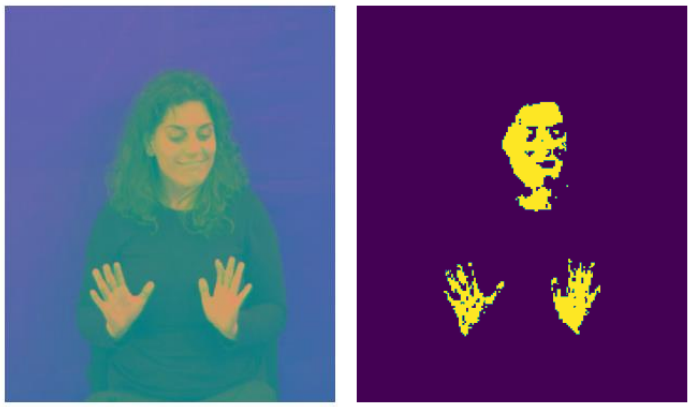


a) Random samples from Gaussian Model



b)Gaussian pdf from chroma samples of skin

Ξεκινάμε διαβάζοντας την εικόνα στο χρωματικό χώρο YCbCr. Έπειτα, υπολογίζουμε για κάθε pixel της εικόνας, τη πιθανότητα να ανήκει σε δέρμα. Αποκτούμε τη δυαδική εικόνα (1 = skin, 0 = no skin) προσθέτοντας ένα κατώφλι ώστε να κρατήσουμε μόνο τα pixel που έχουν μεγάλη πιθανότητα να ανήκουν σε σημεία δέρματος. Δοκιμάζοντας διάφορες τιμές κατωφλίου, τα αποτελέσματα φάνηκαν να προσεγγίζουν το επιθυμητό αποτέλεσμα στο διάστημα [0.02-0.12] και θεωρήσαμε ως καλύτερη επιλογή την τιμή 0.08 καθώς κρατάει την άκρως απαραίτητη πληροφορία του δέρματος χωρίς θόρυβο.



(α) β)

Σχήμα 1: Ανίχνευση περιοχών δέρματος: (α) Πρώτο frame στο χρωματικό χώρο YCbCr. (β) Δυαδική εικόνα ανίχνευσης δέρματος έπειτα από κατωφλιοποίηση της εικόνας πιθανοτήτων.

Πλέον η εικόνα μας έχει θετικές τιμές στα pixel που ανήκουν στο δέρμα, ωστόσο υπόκειται από θόρυβο, κυρίως εντός των περιοχών που μας ενδιαφέρουν. Για να γεμίσουμε τις τρύπες που εμφανίζονται θα εφαρμόσουμε στην εικόνα μια διαδικασία μορφολογικής επεξεργασίας η οποία αποτελείται από opening με ένα μικρό δομικό στοιχείο (2,2) αρχικά και στη συνέχεια closing με μεγάλο δομικό στοιχείο (30,30).

Απαιτείται προσοχή στο όριο τιμής του μεγέθους του δομικού στοιχείου για το closing καθώς από ένα σημείο και έπειτα τείνει να ενώνει περιοχές μεταξύ τους δημιουργώντας μια μάζα και έτσι αποκόπτει το διαχωρισμό των τριών περιοχών που μας ενδιαφέρουν. Δοκιμάζοντας λοιπόν διάφορες τιμές για τα μεγέθη των δομικών στοιχείων, καταλήξαμε στα 2x2 και 30x30.

Η εικόνα που έχει προκύψει έως τώρα είναι μια δυαδική εικόνα στο διάστημα [0,1] όπου έχει άσσους μόνο στα σημεία που βρίσκονται στη περιοχή δέρματος. Για να διαχωρίσουμε τις τρεις διαφορετικές περιοχές (πρόσωπο, αριστερό και δεξί χέρι), θα χρησιμοποιήσουμε τη συνάρτηση label η οποία θα αναγνωρίσει τα διαφορετικά pattern της εικόνας και θα αναθέσει διαφορετικούς αριθμούς για κάθε ένα μοτίβο. Ως αποτέλεσμα θα έχουμε μια εικόνα η οποία έχει διαφορετικούς αριθμούς (διαφορετικά χρώματα) για κάθε μια περιοχή.

Τέλος, για να μπορέσουμε να σχεδιάσουμε ορθογώνια γύρω από τις περιοχές, χρησιμοποιούμε τη συνάρτηση cv2.findContours() η οποία θα βρει το περίγραμμα του σχήματος για κάθε μια περιοχή και θα αποκτήσουμε τις συντεταγμένες καθώς και τις διαστάσεις του ορθογωνίου bounding box με τη συνάρτηση cv2.rectangle().

(α) (β) (γ)

Σχήμα 2: Διαδικασία εύρεσης περιοχών προσώπου και χεριών: (α) Σχηματισμός συνεκτικών περιοχών έπειτα από μορφολογική επεξεργασία. (β) Διαχωρισμός των τριών διαφορετικών περιοχών. (γ) Τελική ανίχνευση των περιοχών που βρίσκονται εντός των ορθογωνίων.

Συνοψίσαμε όλη τη διαδικασία που περιγράψαμε παραπάνω, σε μια συνάρτηση find_bounding() που δέχεται ως είσοδο μια εικόνα, τη μέση τιμή και τη συνδιακύμανση του γκαουσιανού μοντέλου και έχει ως έξοδο τις συντεταγμένες του πάνω αριστερά σημείου του ορθογωνίου και τις διαστάσεις του. Η συνάρτηση αυτή με είσοδο τη πρώτη εικόνα του βίντεο έχει ως έξοδο τις εξής συντεταγμένες:

```
[(94, 274, 64, 78), (208, 270, 52, 80), (156, 104, 70, 116)]
```

Οι συντεταγμένες αυτές είναι αρκετά κοντά με αυτές που μας δόθηκαν ως προτεινόμενες.

1.2 Παρακολούθηση Προσώπου και Χεριών

Στο σημείο αυτό, έχοντας βρει τις αρχικές θέσεις των περιοχών του προσώπου και των χεριών, θα συνεχίσουμε το σύστημα παρακολούθησης αναπτύσσοντας τον αλγόριθμο Lucas-Kanade για εύρεση οπτικής ροής.

Ως οπτική ροή ονομάζουμε την κίνηση (διαφορά) ανάμεσα σε δύο διαφορετικά frames. Δηλαδή, ένα αντικείμενο που έχει κίνηση, θα βρίσκεται σε διαφορετικά σημεία της εικόνας μεταξύ του προηγούμενου και τους αμέσως επόμενου frame. Οπτική ροή λοιπόν είναι μια προσέγγιση της οπτικής κίνησης βασισμένη σε τοπικές παραγώγους σε μια αλληλουχία εικόνων.

1.2.1

Αρχικά θα υλοποιήσουμε τον αλγόριθμο των Lucas-Kanade για την εύρεση της οπτικής κίνησης των ορθογωνίων των περιοχών προσώπου και χεριών. Ο υπολογισμός αυτός της μετατόπισης θα χρησιμοποιηθεί παρακάτω για την πλήρη παρακολούθηση των περιοχών αυτών.

Ας δούμε πρώτα πως λειτουργεί ο αλγόριθμος αυτός.

Έχουμε δύο διαδοχικά frames μιας ακολουθίας εικόνων και στη περίπτωση αυτή θεωρούμε δύο διαδοχικά frames της περιοχής του δεξιού χεριού και συγκεκριμένα για ένα σημείο το οποίο αποτελεί και γωνία (οι γωνίες βρίσκονται σε σημεία με έντονη υφή οπότε είναι πιο εύκολη η ανίχνευση της οπτικής ροής). Υποθέτουμε μια αρχική μετατόπιση (συνήθως 0), δηλαδή το σημείο από τη θέση x,y μετακινήθηκε στη θέση $x + \delta_x$, $y + \delta_y$, $t + \delta_t$. Εφόσον πρόκειται για το ίδιο σημείο, η ένταση της φωτεινότητας και στα δύο σημεία θα είναι ίδια: $I(x, y, t) = I(x + \delta_x, y + \delta_y, t + \delta_t)$ και αναπτύσσοντας κατά σειρά Taylor παίρνουμε:

$$\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t} = 0 \qquad (1)$$

όπου v_x, v_y τα μέρη της ταχύτητας της εικόνας (optical flow).

Η μέθοδος Lucas-Kanade για τη μέτρηση της οπτικής ροής λύνει τις βασικές εξισώσεις οπτικής κίνησης (1) για όλα τα pixel σε μια γειτονιά, χρησιμοποιώντας το κριτήριο των ελάχιστων τετραγώνων. Αποδεικνύεται πως η λύση ελάχιστων τετραγώνων για τη βελτίωση της εκτίμησης της οπτικής ροής σε κάθε σημείο είναι:

$$\mathbf{u}(\mathbf{x}) = \begin{bmatrix} (G_\rho * A_1^2)(\mathbf{x}) + \epsilon & (G_\rho * (A_1 A_2))(\mathbf{x}) \\ (G_\rho * (A_1 A_2))(\mathbf{x}) & (G_\rho * A_2^2)(\mathbf{x}) + \epsilon \end{bmatrix}^{-1} \cdot \begin{bmatrix} (G_\rho * (A_1 E))(\mathbf{x}) \\ (G_\rho * (A_2 E))(\mathbf{x}) \end{bmatrix}$$

όπου

$$A(\mathbf{x}) = \begin{bmatrix} A_1(\mathbf{x}) & A_2(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial I_{n-1}(\mathbf{x}+\mathbf{d}_i)}{\partial x} & \frac{\partial I_{n-1}(\mathbf{x}+\mathbf{d}_i)}{\partial y} \end{bmatrix}$$
$$E(\mathbf{x}) = I_n(\mathbf{x}) - I_{n-1}(\mathbf{x}+\mathbf{d}_i)$$

Επομένως, η διαδικασία για τον υπολογισμό της μετατόπισης μιας περιοχής περιλαμβάνει αρχικά τον υπολογισμό του u για κάθε σημείο του (ή μόνο για features στη περίπτωση της εργασίας) και στη συνέχεια προσθήκη του υπολογισθέντος διανύσματος στην αρχική μας εκτίμηση.

Η συνέλιξη με τον γκαουσιανό πυρήνα γίνεται για να δώσουμε μεγαλύτερη βαρύτητα στα pixel που βρίσκονται πιο κοντά στο pixel που εξετάζουμε διότι παρόλο που κάνουμε την υπόθεση πως τα pixel εντός παραθύρου έχουν σταθερή μετατόπιση, είναι πιο πιθανό τα σημεία που βρίσκονται σε πολύ κοντινή απόσταση από το pixel που εξετάζουμε να ακολουθούν την ίδια κίνηση που εκτελέστηκε από αυτό.

Ουσιαστικά στόχος είναι η ελαχιστοποίηση αυτού του τετραγωνικού σφάλματος και η προσέγγιση της πραγματικής μετατόπισης d.

Η παραπάνω διαδικασία υλοποιείται με τη συνάρτηση lucas_kanade() η οποία δέχεται ως είσοδο δύο διαδοχικά frames από τα παράθυρα της περιοχής ενδιαφέροντος, ανιχνευθείσες γωνίες εντός του παραθύρου, κλίμακα του γκαουσιανού παραθύρου, τη σταθερά κανονικοποίησης ε και την αρχική εκτίμηση της μετατόπισης και επιστρέφει το υπολογισμένο διάνυσμα οπτικής ροής. Η υλοποίηση έγινε για κάθε περιοχή ξεχωριστά.

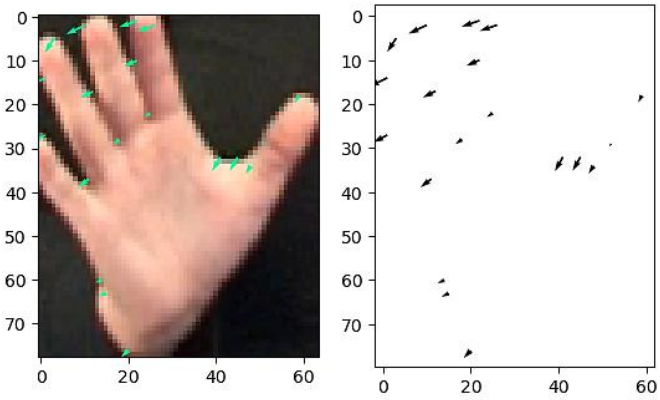
Στα παρακάτω παραδείγματα εικόνων θα γίνει ξεκάθαρη η διαδικασία υπολογισμού της οπτικής ροής.

Ξεκινάμε από τη περιοχή του αριστερού χεριού, 'κόβοντας' παράθυρα που ορίζουν τα bounding boxes και εν συνεχεία βρίσκουμε γωνίες στο πρώτο frame.

(α) (β) (γ)

Σχήμα 3: (α) Περιοχή αριστερού χεριού frame 1. (β) Περιοχή αριστερού χεριού frame 2. (γ) Γωνίες που ανιχνεύθηκαν στη περιοχή του χεριού

Έπειτα υπολογίζουμε τα διανύσματα οπτική ροής και τα εμφανίζουμε πάνω στο δεύτερο frame.

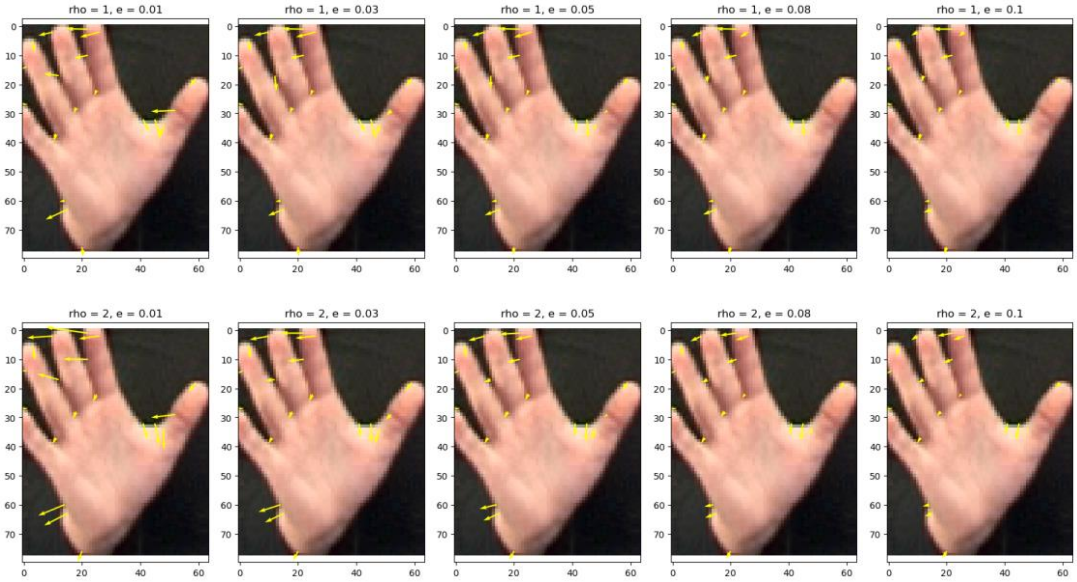


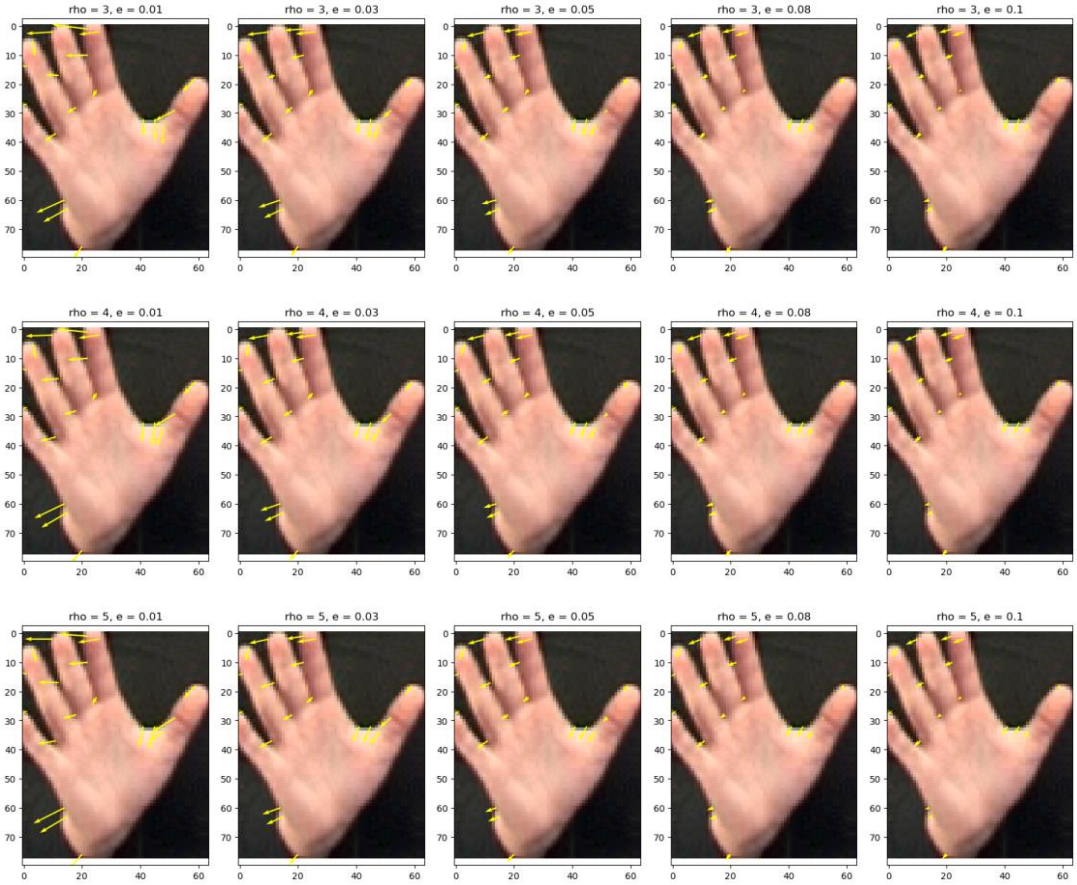
Σχήμα 4: Οπτική ροή αριστερού χεριού μεταξύ δύο διαδοχικών καρέ

Οι παράμετροι που χρησιμοποιήθηκαν για τον συγκεκριμένο υπολογισμό των μετατοπίσεων είναι:

- $\rho = 5$
- $\epsilon = 0.1$
- Αρχική εκτίμηση $\mathrm{d} = 0$

Δοκιμάζοντας διάφορες τιμές των παραμέτρων της κλίμακας ρ και του ϵ παρατηρούμε τα εξής αποτελέσματα στον υπολογισμό της οπτικής ροής.





Σχήμα 5: Το αποτέλεσμα των τιμών των παραμέτρων ρ και ε στην ανίχνευση κίνησης

Αυτό που αντιλαμβανόμαστε είναι πως καθώς το ρ και το ϵ αυξάνονται, τα διανύσματα μικραίνουν. Τι σημαίνει αυτό. Η εν λόγω σταθερά ϵ είναι μια σταθερά κανονικοποίησης και έχει σκοπό να βελτιώσει το αποτέλεσμα σε επίπεδες περιοχές με μειωμένη υφή και πληροφορία. Άρα αναμένουμε ότι αύξηση της σταθεράς αυτής θα οδηγεί σε πιο 'ευσταθή αποτελέσματα.

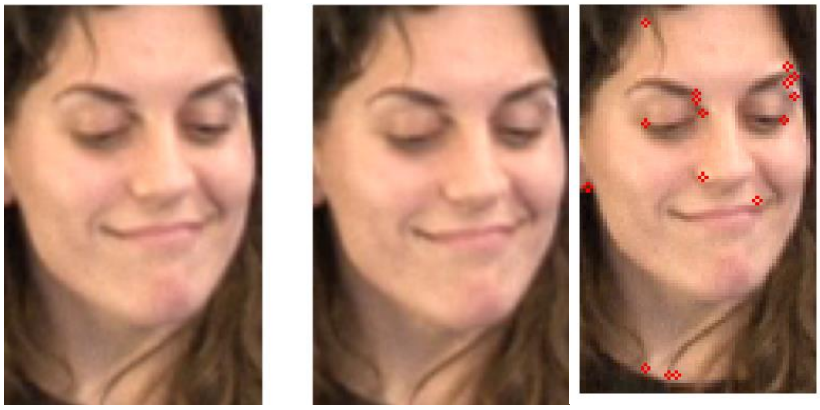
Παρόμοια με τη σταθερά ϵ, φαίνεται πως και το εύρος ρ καθώς αυξάνεται επιφέρει πιο 'ευκρινή' και σταθερά αποτελέσματα χωρίς να υπεισέρχεται θόρυβος, όπου θόρυβο ονομάζουμε τις μικρές κινήσεις που δεν αφορούν τη συνολική αληθινή κίνηση της περιοχής που μας ενδιαφέρει.

Στο σχήμα 5, το ρ αυξάνεται κατά στήλη και το ϵ κατά γραμμή. Φαίνεται ξεκάθαρα πως για την ελάχιστη τιμή του ϵ τα διανύσματα οπτικής ροής είναι τελείως random από όπου δεν μπορεί να αποφανθεί μια συνολική πραγματική κίνηση.

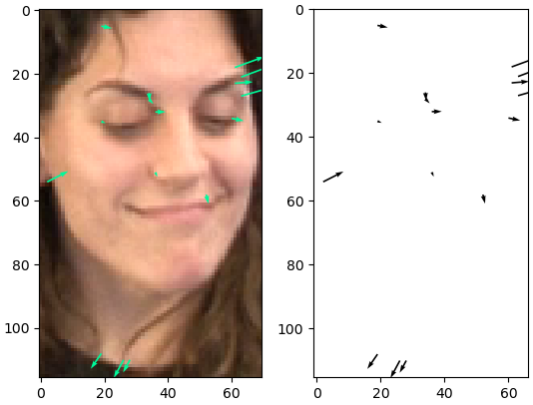
Βέβαια, η επιλογή των παραμέτρων αυτών διαφέρει αναλόγως και το πρόβλημα που αντιμετωπίζουμε και το επιθυμητό αποτέλεσμα που θέλουμε να πετύχουμε.

Ακολουθούμε την ίδια διαδικασία και για τις άλλες περιοχές.

Περιοχή προσώπου:



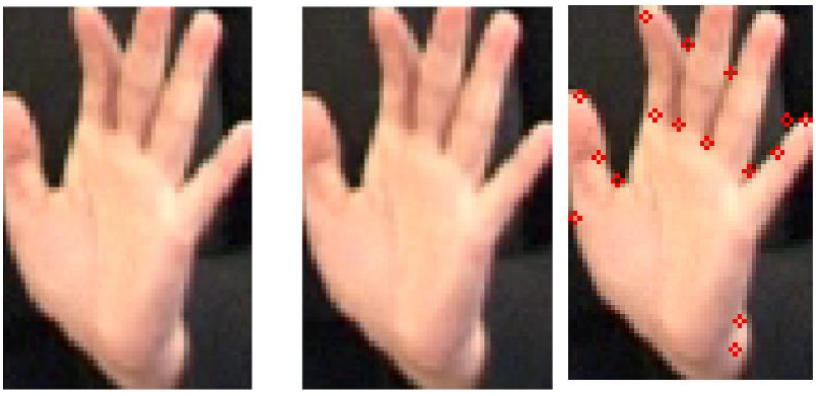
(α) (β) (γ)



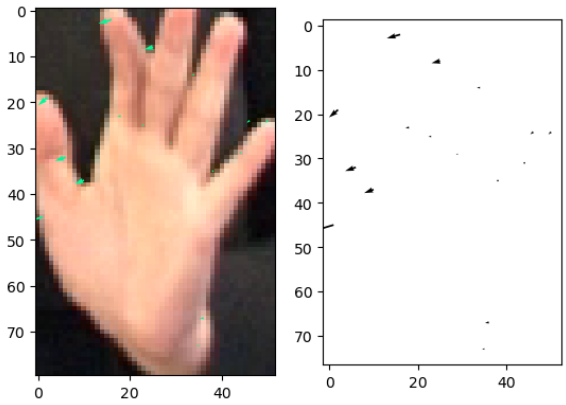
(δ) (ε)

Σχήμα 6: Ανίχνευση οπτικής ροής στη περιοχή του προσώπου: (α) Πρώτο frame (β) Δεύτερο frame (γ) Εντοπισμός σημείων ενδιαφέροντος. (δ)Διανύσματα οπτικής ροής πάνω στην εικόνα. (ε) Δισδιάστατο οπτικό πεδίο κίνησης

Περιοχή δεξιού χεριού:



(α) (β) (γ)



(δ) (ε)

Σχήμα 7: Ανίχνευση οπτικής ροής στη περιοχή του δεξιού χεριού: (α) Πρώτο frame (β) Δεύτερο frame (γ) Εντοπισμός σημείων ενδιαφέροντος. (δ)Διανύσματα οπτικής ροής πάνω στην εικόνα. (ε) Δισδιάστατο οπτικό πεδίο κίνησης

1.2.2 Υπολογισμός της Μετατόπισης των Παραθύρων από τα Διανύσματα Οπτικής Ροής

Για να καταλάβουμε τη κατεύθυνση των bounding boxes χρειαζόμαστε ένα αντιπροσωπευτικό διάνυσμα, οπότε είναι αναγκαίο από τα 20 περίπου διανύσματα μετατόπισης που έχουμε για το κάθε ορθογώνιο να εξάγουμε ένα διάνυσμα το οποίο θα συνοψίζει τη κίνηση που εκτελείται από το εκάστοτε ορθογώνιο bounding box.

Η διαδικασία που ακολουθούμε είναι να κρατάμε τα διανύσματα τα οποία έχουν ενέργεια μεγαλύτερη από ένα κατώφλι και στη συνέχεια να παίρνουμε τη μέση τιμή των διανυσμάτων αυτών. Η ενέργεια ορίζεται ως: $\|d\|^2 = d_x{}^2 + d_y{}^2$

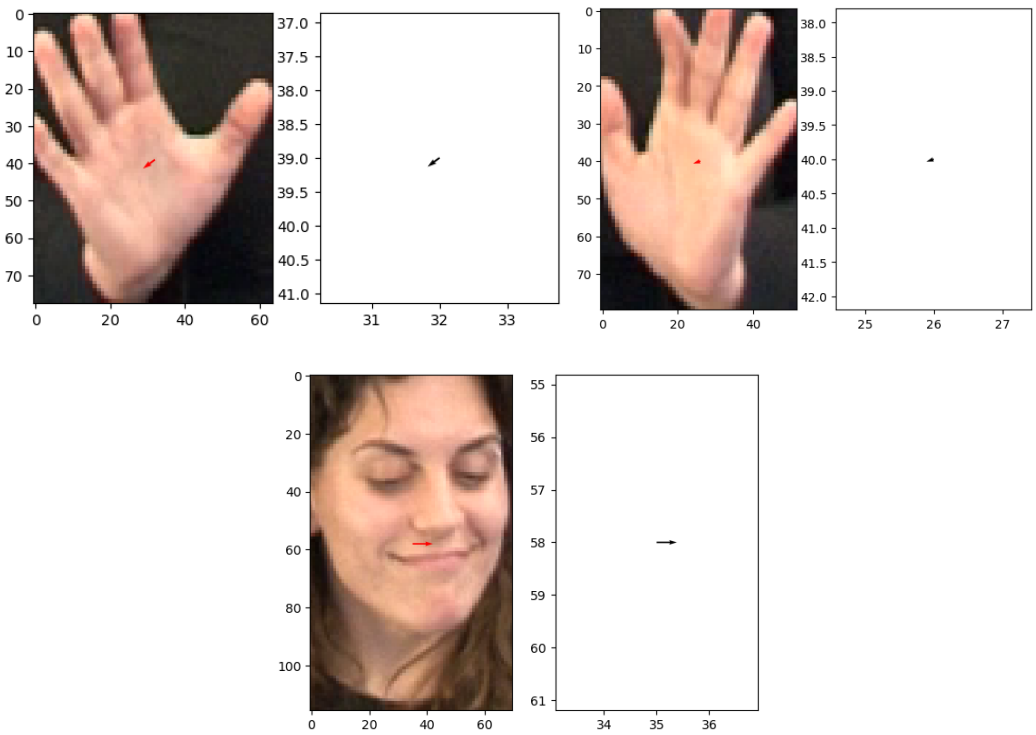
Πειραματίζοντας με τις τιμές κατωφλίου καταλήγουμε στη τιμή 0.2, με την οποία έχουμε τα τελικά διανύσματα μετατόπισης:

Final vector displacement of bounding box for left hand: -4, 3

Final vector displacement of bounding box for face: 1, 0

Final vector displacement of bounding box for right hand: -3, 1

Αποτελέσματα:



Σχήμα 8: Τελικά διανύσματα οπτικής κίνησης

Όσον αφορά το threshold της ενέργειας, πειραματιστήκαμε με διάφορες τιμές και το συμπέρασμα που βγάλαμε ήταν πως για πολύ μικρές τιμές κατωφλίου, το κριτήριο εξαγωγής συνολικών μετατοπίσεων είναι ουσιαστικά η μέση τιμή όλων των διανυσμάτων χωρίς να υπάρχει κάποιο φιλτράρισμα. Αντίθετα, όταν το κατώφλι λάβει πολύ μεγάλες τιμές τότε η συνολική μετατόπιση εξάγεται ως η μέση τιμή διανυσμάτων με μεγάλη ενέργεια, το οποίο έχει ως αποτέλεσμα η εξαγόμενη μετατόπιση να είναι μεγαλύτερη της πραγματικής, δηλαδή η ανίχνευση κίνησης αποτυγχάνει λόγω μεγάλων μετατοπίσεων των bounding boxes, καθώς λαμβάνουμε υπόψη μόνο τα μεγάλα διανύσματα.

Μια ενδεικτική περιοχή τιμών του κατωφλίου ενέργειας είναι [0.2, 0.5]

Τελικό σύστημα παρακολούθησης με τη μέθοδο Lucas-Kanade.

Σχήμα 9: Σύστημα παρακολούθησης προσώπου και χεριών με τη μέθοδο Lucas-Kanade. Διάφορες ενδιάμεσες φάσεις της ανίχνευσης

Η πρώτη εικόνα αποτελεί στιγμιότυπο από βίντεο που παρέχεται στο φάκελο Images/ OneScale-Tracking

Οι παράμετροι που χρησιμοποιήθηκαν για το παραπάνω αποτέλεσμα είναι:

- $\epsilon = 0.1$
- $\rho = 5$
- energy_threshold $= 0.2$

Κάναμε διάφορους πειραματισμούς για να καταλήξουμε σε αυτό το αποτέλεσμα αλλάζοντας κυρίως τις τιμές του ρ και ϵ. Παρατηρήσαμε πως για μικρές τιμές του ϵ

γίνεται λιγότερο λεπτομερή ανίχνευση της κίνησης και συγκεκριμένα οι μετατοπίσεις του ορθογωνίου από ένα frame στο επόμενο ήταν τραχείς και απότομες με μεγάλες αλλαγές στη κατεύθυνση, ενώ όσο αυξάναμε το ϵ η κίνηση των bounding_boxes ήταν πιο smooth.

Η παρακολούθηση συνολικά δεν είναι εντελώς ακριβής και αυτό έχει να κάνει με διάφορους παράγοντες όπως η αρχική ανίχνευση δέρματος αλλά και ο αριθμός και η ποιότητα των γωνιών σχετικά με τα οποία ανιχνεύουμε τη κίνηση.

Η μέθοδος αυτή, δυσκολεύεται να εντοπίσει απότομες αλλαγές στη κίνηση όπως η απότομη αλλαγή στη κατεύθυνση του αριστερού χεριού προς το τέλος του βίντεο. Αυτό ήταν αναμενόμενο διότι ο αλγόριθμος Lucas-Kanade είναι σχεδιασμένος να υπολογίζει την οπτική ροή ανάμεσα σε διαδοχικά frames τα οποία παρεκκλίνουν λίγα pixel.

Η σημασία της παραμέτρου ρ είναι η κλίμακα ανίχνευσης της κίνησης. Για πολύ μικρές τιμές της κλίμακας, τα διανύσματα οπτικής ροής μικραίνουν και ο αλγόριθμος καταλαβαίνει πολύ μικρή κίνηση, καθώς εστιάζει πολύ κοντά στη περιοχή του pixel ανιχνεύοντας κάθε απειροελάχιστη κίνηση που υφίσταται. Αντίθετα, καθώς αυξάνουμε τη κλίμακα ο αλγόριθμος εστιάζει σε γενικότερες, συνολικές κινήσεις. Έτσι αντιλαμβάνεται το ορθογώνιο κομμάτι της εικόνας ως ένα αυτούσιο κομμάτι και δεν μπορεί να ξεχωρίσει τα επιμέρους χαρακτηριστικά.

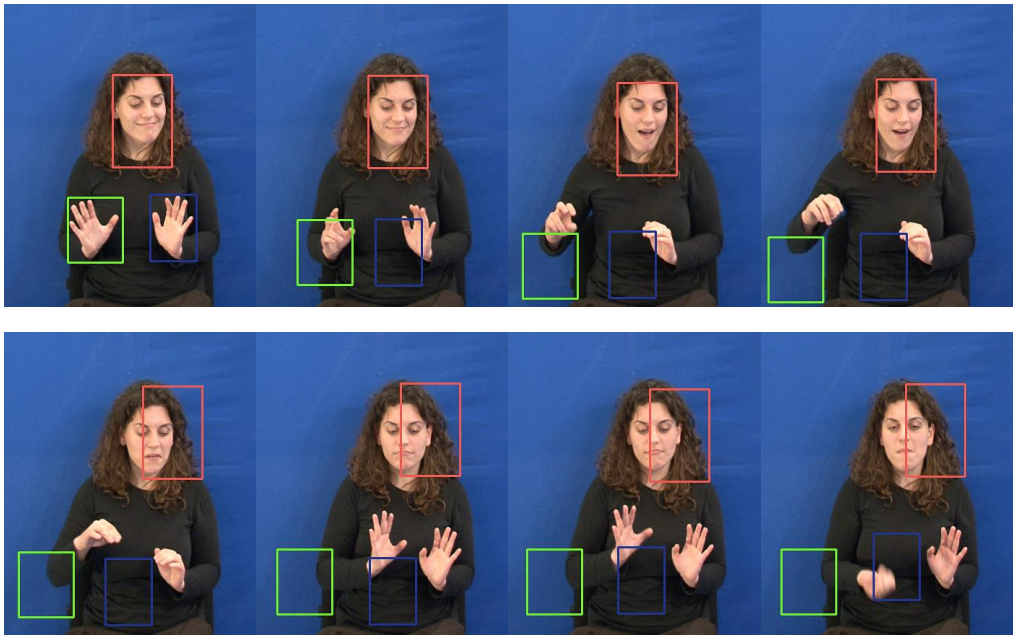
1.2.3 Πολυ-Κλιμακωτός Υπολογισμός Οπτικής Ροής

Το μειονέκτημα του μονοκλιμακωτού Lucas-Kanade είναι πως ανιχνεύει πολύ μικρές κινήσεις, συνεπώς αδυνατεί να ανιχνεύσει απότομες αλλαγές στη κατεύθυνση κίνησης όπως αναφέραμε.

Η πολυκλιμακωτή έκδοση του Lucas-Kanade αναλύει τις εικόνες σε γκαουσιανές πυραμίδες και στη συνέχει υπολογίζει την οπτική ροή από τις μικρές τις πιο μεγάλες κλίμακες χρησιμοποιώντας τη λύση της μικρής κλίμακας για το επόμενο επίπεδο. Με αυτό το τρόπο επιτυγχάνεται ανίχνευση κινήσεων που υπερβαίνουν κατά πολύ το ένα pixel και παράλληλα εκτός από τις μικρές κινήσεις είναι εφικτό να ανιχνευθούν και μεγάλες κινήσεις.

Ο αριθμός των επιπέδων της πυραμίδας δεν πρέπει να υπερβαίνει τα 4 επίπεδα καθώς από εκεί και έπειτα η συνολική μετατόπιση ενδέχεται να είναι ιδιαίτερα αυξημένη.

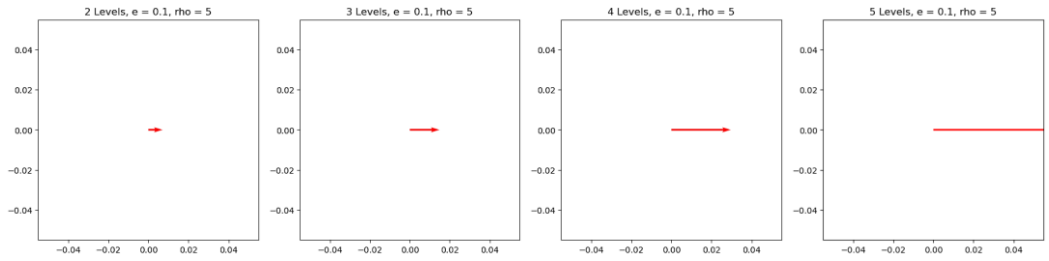
Σύστημα παρακολούθησης βασισμένο στον πολυκλιμακωτό Lucas-Kanade.



Σχήμα 10: Σύστημα παρακολούθησης προσώπου και χεριών με τη πολυκλιμακωτή μέθοδο Lucas-Kanade. Διάφορες ενδιάμεσες φάσεις της ανίχνευσης.

Από τα στιγμιότυπα φαίνεται πως η κίνηση των bounding boxes τώρα είναι πιο γρήγορη καθώς όπως αναφέραμε, με τη πολυκλιμακωτή μέθοδο είναι δυνατό να ανιχνευθούν κινήσεις σε διάφορες κλίμακες οπότε όλες συνεισφέρουν στη τελική κίνηση. Μετα από αρκετά frames τα ορθογώνια των χεριών παύουν πλέον να τα ακολουθούν καθώς έχουν χάσει οπτική επαφή.

Για καλύτερη κατανόηση της επίδρασης που έχει ο αριθμός των επιπέδων της πυραμίδας στην ανίχνευση οπτικής ροής, οπτικοποιούμε τα διανύσματα οπτικής ροής για το αριστερό μόνο χέρι για διάφορες τιμές επιπέδων.



Σχήμα 11: Επίδραση του αριθμού των επιπέδων της πυραμίδας στην ανίχνευση του διανύσματος οπτικής ροής για το αριστερό χέρι

Φαίνεται και στο σχήμα πως καθώς αυξάνεται ο αριθμός των επιπέδων, αυξάνεται και η συνολική μετατόπιση της κίνησης του χεριού.

Μέρος 2: Εντοπισμός Χωρο-χρονικών Σημείων Ενδιαφέροντος και Εξαγωγή Χαρακτηριστικών σε Βίντεο Ανθρωπίνων Δράσεων

Στο μέρος αυτό καλούμαστε να κατηγοριοποιήσουμε βίντεο ανθρωπίνων δράσεων σε τρεις κλάσεις: walking, running και handwaving. Το υλικό που θα χρησιμοποιήσουμε βρίσκεται στον φάκελο part2 – SpatioTemporal. Επιλέγουμε από τα δοθέντα βίντεο τυχαία τρία, ένα από κάθε κατηγορία και πάνω σε αυτά εργαζόμαστε για το 2.1 μέρος. Σε πρώτο στάδιο υλοποιούμε δύο διαφορετικές προσεγγίσεις για τον εντοπισμό χωρο-χρονικών σημείων ενδιαφέροντος στα βίντεο που επιλέξαμε, δηλαδή σημείων που στην διάρκεια του χρόνου παρουσιάζουν κάποιες μεταβολές (κίνηση). Η μία προσέγγιση για τον εντοπισμό των σημείων αυτών είναι ο 3D Harris Detector, ο οποίος αποτελεί μια γενίκευση του γνωστού 2D Harris Stephens Detector με τον οποίο ασχοληθήκαμε στο προηγούμενο εργαστήριο. Η δεύτερη προσέγγιση είναι ο Gabor Detector, ο οποίος βασίζεται στο χρονικό φιλτράρισμα των βίντεο με την χρήση ενός ζεύγους Gabor φίλτρων. Σε επόμενο στάδιο χωρίζουμε τα συνολικά βίντεο που μας δίνονται στον φάκελο, σε train και test βίντεο. Αφού εφαρμόσουμε τους δύο ανιχνευτές μας σε όλα τα βίντεο, χρησιμοποιήσουμε τα σημαντικότερα σημεία που ο καθένας από αυτούς εντοπίζει, ώστε να εξάγουμε τοπικούς περιγραφητές γύρω από αυτά. Συγκεκριμένα υλοποιούμε τρεις τοπικούς περιγραφητές: τους HOG, HOF και τον συνδυασμό αυτών, τον HOG-HOF. Έπειτα χρησιμοποιούμε την Bag of Visual Words αναπαράσταση (BoVW) για κάθε βίντεο, βασισμένη στα διανύσματα τοπικών περιγραφητών που εξαγάγαμε προηγουμένως και οι τελικές αυτές 'global representations' χρησιμοποιούνται ως ορίσματα σε έναν ειδικά σχεδιασμένο svm ταξινομητή πολλαπλών κλάσεων για την τελική κατηγοριοποίηση των βίντεο στις τρεις ανθρώπινες δράσεις που αναφέραμε παραπάνω. Συνοψίζοντας λοιπόν, υλοποιούμε δύο ανιχνευτές για τον εντοπισμό των χωρο-χρονικών σημείων ενδιαφέροντος και 3 περιγραφητές για καθέναν από αυτούς, δηλαδή στο σύνολο έχουμε 6 διαφορετικούς συνδυασμούς που θα μας δώσουν τα τελικά διανύσματα χαρακτηριστικών BoVW βάση των οποίων θα γίνει η ταξινόμηση.

2.1 Χωρο-χρονικά Σημεία Ενδιαφέροντος

Για το μέρος αυτό επιλέγουμε τυχαία ένα βίντεο από κάθε κλάση δράσεων. Αρχικά διαβάζουμε τα βίντεο με την συνάρτηση read_video() την οποία κάνουμε απευθείας import από το έτοιμο υλικό του εργαστηρίου που μας δίνεται, cv23_lab2_2_utils. Στην συνάρτηση αυτή ορίζουμε αρχικά το όνομα του βίντεο, έπειτα το πλήθος των frames στα οποία θέλουμε να το χωρίσουμε (επιλέγουμε 200 frames) και την ταχύτητα αναπαραγωγής του βίντεο (παίρνουμε την default τιμή 0 που αντιστοιχεί στην αληθινή ταχύτητα).

2.1.1

Υλοποιούμε αρχικά τον Harris 3D Detector ο οποίος αποτελεί μια προέκταση του γνωστού 2D Harris-Stephens Detector εισάγοντας επιπλέον την διάσταση του χρόνου (t).

Η σχέση που περιγράφει αυτόν τον ανιχνευτή είναι η εξής:

$$M(x, y, t; \sigma, \tau) = g(x, y, t; s\sigma, s\tau) * (\nabla L(x, y, t; \sigma, \tau)(\nabla L(x, y, t; \sigma, \tau))^T)$$

η οποία σε μορφή πινάκων γράφεται:

$$M(x, y, t; \sigma, \tau) = g(x, y, t; s\sigma, s\tau) * \begin{pmatrix} L_x^2 & L_x L_y & L_x L_t \\ L_x L_y & L_y^2 & L_y L_t \\ L_x L_t & L_y L_t & L_t^2 \end{pmatrix}$$

Ορίζουμε την συνάρτηση Harris() μέσα στην οποία γίνονται οι κατάλληλοι υπολογισμοί ώστε να μπορούμε να υπολογίσουμε έπειτα το κριτήριο σημαντικότητας ανιχνευτή Harris 3D. Μέσα στην συνάρτηση αυτή γίνονται οι εξής υπολογισμοί:

Σε κάθε frame [x,y] των βίντεο μας πραγματοποιούμε αρχικά εξομάλυνση σε χωρική κλίμακα sσ εφαρμόζοντας έναν δισδιάστατο Gaussian kernel. Στη συνέχεια πραγματοποιούμε εξομάλυνση όλων των frames σε χρονική κλίμακα sτ συνελίσσοντας διαδοχικά το χωρικά εξομαλυμένο βίντεο μας, με έναν μονοδιάστατο Gaussian kernel, κατά τον άξονα του χρόνου (t).

Σημειώνουμε ότι ο χωρο-χρονικός Gaussian πυρήνας μπορεί να υλοποιηθεί ως δύο διαδοχικά φιλτραρίσματα λόγω της διαχωρισιμότητάς του:

$$g(x, y, t; s\sigma, s\tau) = \frac{1}{\sqrt{2\pi(s\tau)^2}} e^{-\frac{t^2}{2(s\tau)^2}} * \frac{1}{2\pi(s\sigma)^2} e^{-\frac{x^2+y^2}{2(s\sigma)^2}}$$

$$g(x, y, t; s\sigma, s\tau) = \frac{1}{(2\pi)^{\frac{3}{2}} * (s\tau) * (s\sigma)^2} e^{-\frac{t^2}{2(s\tau)^2}} * e^{-\frac{x^2+y^2}{2(s\sigma)^2}}$$

Αφού έχουμε εξομαλύνει χωρο-χρονικά τα βίντεο, βρίσκουμε τις πρώτες παραγώγους (gradients) κάθε voxel προς τις τρεις διαστάσεις (gradient_x, gradient_y, gradient_t) συνελίσσοντας με το 1D derivative filter $[-1, 0, 1]^T$, στους αντίστοιχους άξονες κάθε φορά (ό άξονας 0 αντιστοιχεί στην διάσταση του χρόνου (t), ο άξονας 1 αντιστοιχεί στην διάσταση του ύψους (y) και ο άξονας 2 αντιστοιχεί στην διάσταση του πλάτους (x)).

Έπειτα υπολογίζουμε τις δεύτερες παραγώγους (Lx, Ly, Lt) κάθε voxel συνελίσσοντας τις αντίστοιχες πρώτες παραγώγους με το derivative filter μας στους κατάλληλους άξονες.

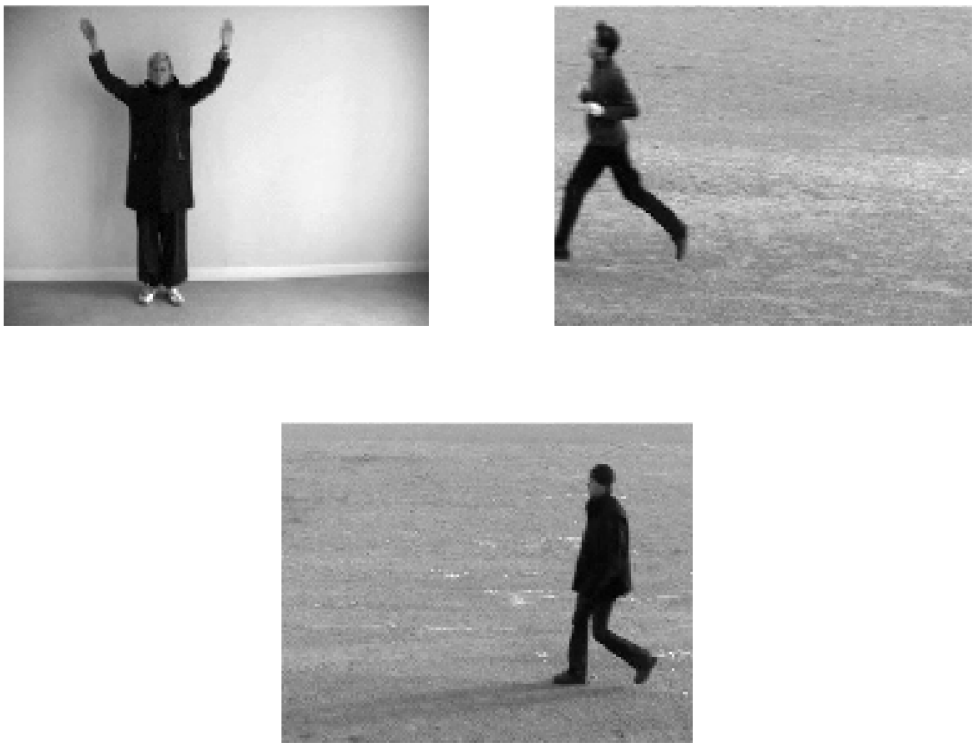
Αφού έχουμε υπολογίσει τα παραπάνω, επιστρέφουμε με την συνάρτηση μας την ορίζουσα του πίνακα δεύτερων παραγώγων (ο οποίος φαίνεται στην σχέση που παραθέσαμε παραπάνω) καθώς επίσης και το ίχνος αυτού.

Ορίζουμε τώρα μια νέα συνάρτηση criterion_Harris() η οποία δέχεται ως ορίσματα το video, την χωρική κλίμακα σ, την χρονική κλίμακα τ και τον παράγοντα κλιμάκωσης s. Χρησιμοποιώντας την ορίζουσα και το ίχνος που υπολογίζονται από την Harris(), η συνάρτηση criterion_Harris() επιστρέφει το 3D κριτήριο γωνιότητας

$$H(x, y, t) = det(M(x, y, t)) - k \cdot trace^3(M(x, y, t))$$

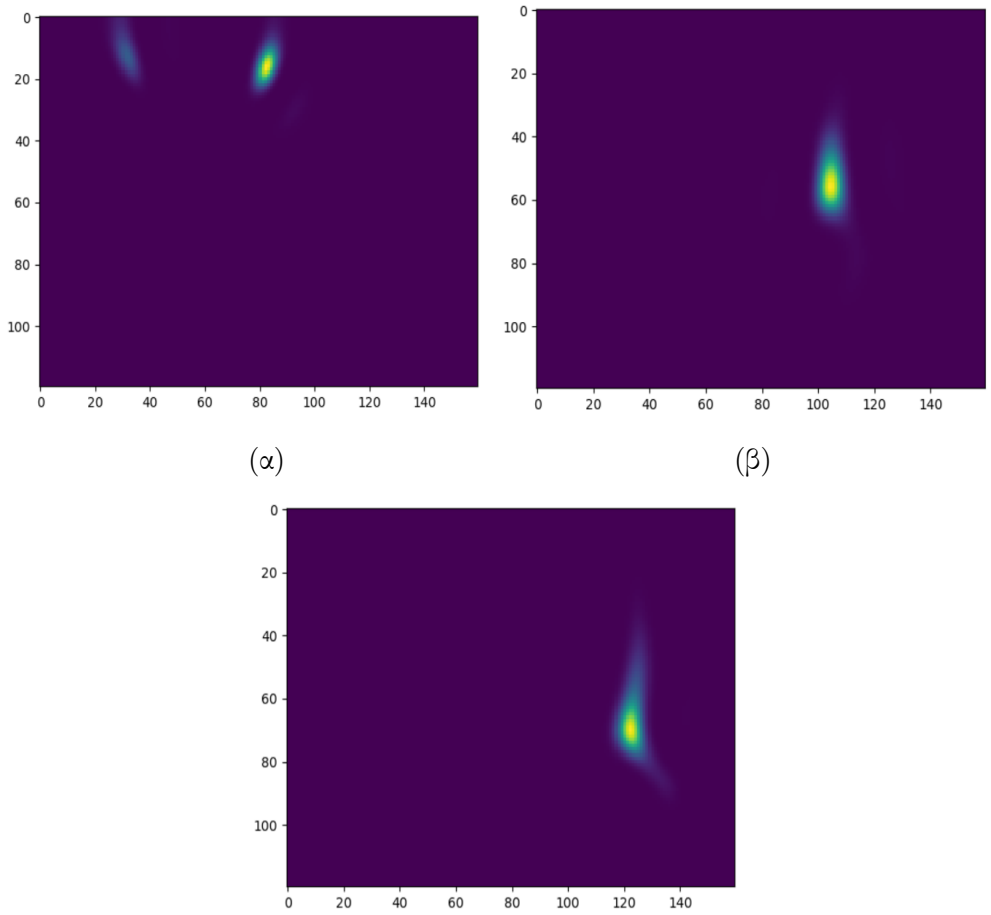
Στους παραπάνω υπολογισμούς χρησιμοποιούμε τις ενδεικτικές τιμές: s=2, σ=4, τ=1.5

Επιλέγουμε τυχαία να απεικονίσουμε το frame 25 των τριών βίντεό μας :



Σχήμα 12: Frame 25 sample video δράσεων

Η απεικόνιση του 3D Harris κριτηρίου για τα συγκεκριμένα frames είναι η εξής:



Σχήμα 13: 3D Harris criterion (α) handwaving, (β) running, (γ)walking

Σημειώνουμε ότι για την απεικόνιση αυτή, κανονικοποιήσαμε όλες τις τιμές των voxel από 0 έως 255 και επιλέξαμε να απεικονίσουμε με πιο έντονο χρώμα (τιμές κοντά στο 255) τις περιοχές που εντοπίζεται κίνηση. Για τον λόγο αυτό δημιουργήσαμε το κριτήριο με τις «αντίθετες» τιμές των voxel που λαμβάνουμε αρχικά.

Για να αντιληφθούμε καλύτερα τον τρόπο με τον οποίο ανιχνεύεται η κίνηση μέσω του συγκεκριμένου κριτηρίου, εμφανίζουμε υπό την μορφή 'animation' τα διαδοχικά frames του κάθε Harris criterion. Τα 'animation' αυτά επισυνάπτονται στον φάκελο 'lab2_part2_material' μαζί με όλα τα υπόλοιπα αρχεία του μέρους 2.

2.1.2

Υλοποιούμε τώρα τον Gabor Detector μας. Αρχικά εξομαλύνουμε όλα τα frames [x,y] με έναν δισδιάστατο gaussian kernel g(x,y,σ) στον χώρο διαστάσεων (x,y). Όπως αναφέραμε ο Gabor ανιχνευτής βασίζεται στο χρονικό φιλτράρισμα του βίντεο μας και αυτό μπορεί να επιτευχθεί λόγω διαχωρισιμότητας, με την χρήση δύο Gabor κρουστικών αποκρίσεων:

$$h_{ev}(t;\tau,\omega) = \cos(2\pi t\omega)\exp(-t^2/2\tau^2) \text{ και } h_{od}(t;\tau,\omega) = \sin(2\pi t\omega)\exp(-t^2/2\tau^2)$$

Όπως βλέπουμε από την παραπάνω σχέση, κάθε μία από τις δύο αποκρίσεις που χρησιμοποιούμε έχει υποστεί εξομάλυνση μέσω ενός gaussian παράγοντα exp(-$t^2/2\tau^2$) εντός χρονικού παραθύρου [-2τ,2τ]. Οι παράγοντες cos(2πtω) και sin(2πtω) δείχνουν τις διαφορετικές κατευθυντικότητες της χρονικής μεταβολής, με την συχνότητα ω να ισούται με 4/τ.

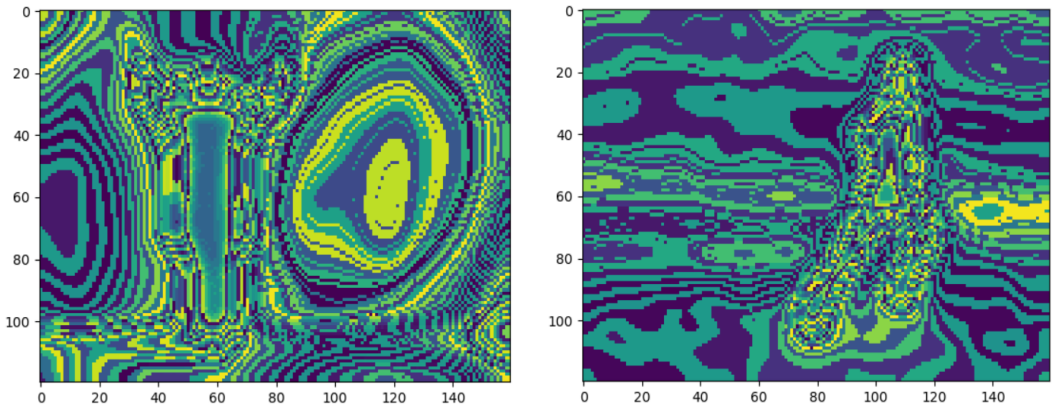
Ορίζουμε την συνάρτηση GaborDetector() η οποία δέχεται σαν όρισμα την χωρική κλίμακα τ βάσει της οποίας ορίζουμε το χρονικό παράθυρο φιλτραρίσματος ίσο με [-2τ,2τ]. Μέσα σε αυτήν υπολογίζουμε τις εξομαλυμένες hev και hod αποκρίσεις, τις οποίες κανονικοποιούμε έπειτα με την L1 νόρμα τους αντίστοιχα, ώστε οι τιμές όλων των pixelνα αθροίζουν στην μονάδα. Για να συμφωνούν οι τιμές των κριτηρίων μας, μεταθέτουμε τις τιμές όλων των voxel του κριτηρίου μας σε κλίμακα από 0 εως 255.

Κατασκευάζουμε την συνάρτηση criterion_Gabor() η οποία δέχεται σαν όρισμα το video, την χωρική κλίμακα σ, την χρονική κλίμακα τ και τον παράγοντα κλιμάκωσης s και επιστρέφει το κριτήριο σημαντικότητας:

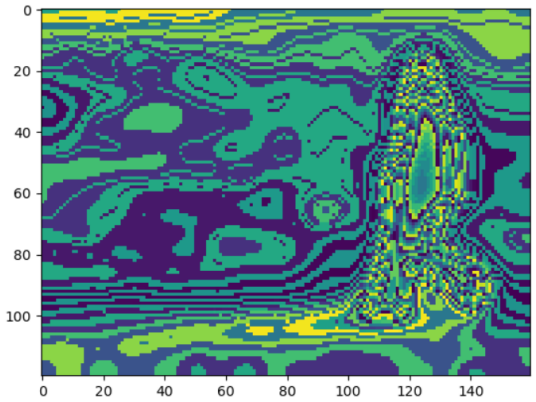
$$H(x,y,t) = (I(x,y,t) * g * h_{ev})^2 + (I(x,y,t) * g * h_{od})^2$$

Στο κριτήριο αυτό χρησιμοποιούμε χωρικά εξομαλυμένες τιμές των voxel, με gaussian πυρήνα g(x,y;σ). Οι παράγοντες (I(x,y,t)*g*hev)^2 και (I(x,y,t)*g*hod)^2 δείχνουν την τετραγωνική ενέργεια της εξόδου προς την κάθε κατεύθυνση αντίστοιχα.

Η απεικόνιση του κριτηρίου Gabor για τυχαία frame των βίντεό μας φαίνεται παρακάτω:



(α) (β)



(γ)

Σχήμα 14: Gabor criterion (α) handwaving (β) running (γ)walking

Αντίστοιχα με πριν δημιουργούμε τα animation που προκύπτουν από όλα τα frames των κριτηρίων που υπολογίσαμε ώστε να δούμε τον τρόπο με τον οποίο ο ανιχνευτής αυτός αντιλαμβάνεται την κίνηση. Τα animation αυτά (όπως και όλα όσα θα δημιουργήσουμε παρακάτω) βρίσικονται στον φάκελο 'lab2_part2_material'

2.1.3

Για κάθε ένα από τα δύο είδη κριτηρίων που υπολογίσαμε παραπάνω, εφαρμόζουμε ένα κατώφλι, απορρίπτοντας έτσι τα σημεία που ανήκουν σε σχετικά ομαλές περιοχές. Η διαδικασία αυτή υλοποιείται μέσω της συνάρτησης find_coords() που κατασκευάζουμε, η οποία παίρνει σαν ορίσματα το κριτήριο του βίντεο, την χωρική κλίμακα σ και την γωνία κατωφλιοποίησης theta_corn και επιστρέφει τις συντεταγμένες (x,y,t) καθώς επίσης και την κλίμακα σ των σημείων που ικανονποιούν την ανισότητητα :

Criterion[x,y,t]>theta_corn*Criterion_max

Βλέποντας τις τιμές των pixel για τα frame των κριτηρίων που εμφανίζουμε, εστιάζοντας στις τιμές των περιοχών κίνησης που μας ενδιαφέρουν, επιλέγουμε πειραματικά ως καλύτερες τιμές τις theta_corn = 0.25 και theta_corn = 0.705 για τον Harris και Gabor ανιχνευτή αντίστοιχα.

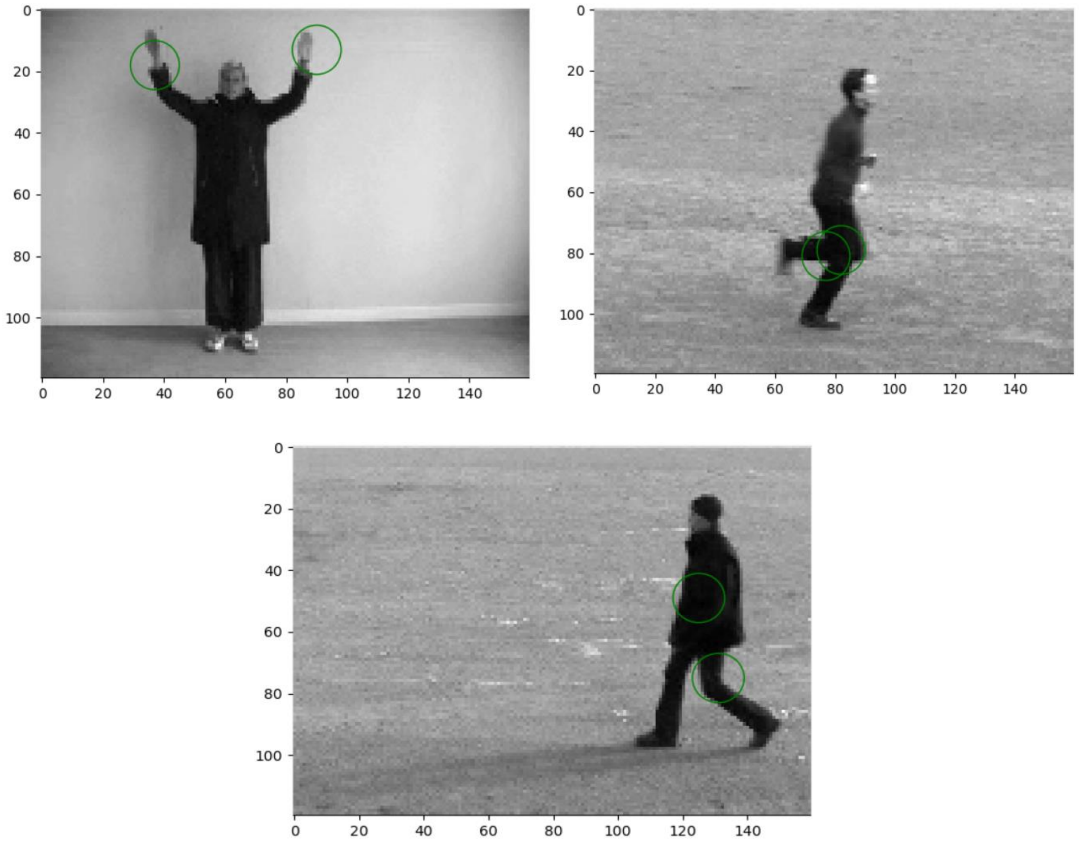
2.1.4

Σε επόμενο βήμα θα μειώσουμε ακόμα περισσότερο το πλήθος των σημείων που θα χρησιμοποιήσουμε μετέπειτα για την εξαγωγή τοπικών περιγραφητών, κρατώντας τα 500 πρώτα σημεία με τις υψηλότερες τιμές για το κάθε κριτήριο αντίστοιχα. Η τελική επιλογή γίνεται μέσω της συνάρτησης final_points που κατασκευάζουμε, η οποία δέχεται σαν ορίσματα το κριτήριο που έχουμε υπολογίσει για το βίντεό μας, τα σημεία ενδιαφέροντος που υπερβαίνουν το κατώφλι που εμείς ορίσαμε προηγουμένως, και το πλήθος των σημαντικότερων σημείων (με τις μεγαλύτερες τιμές) που θέλουμε να

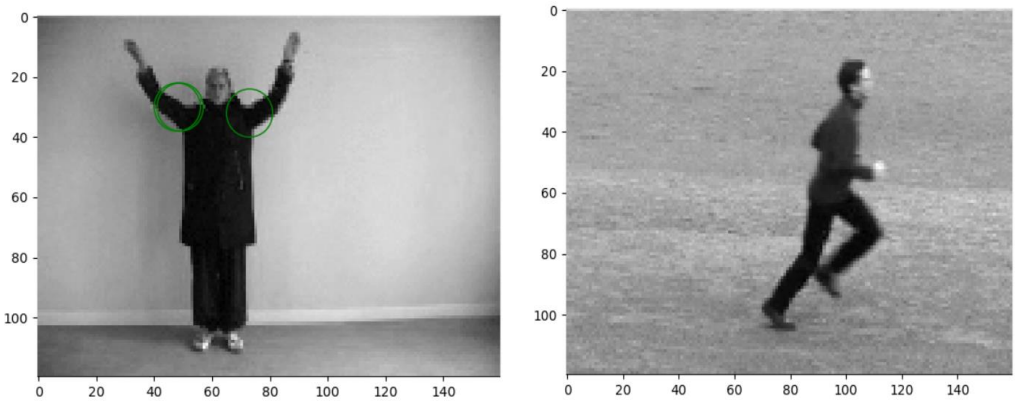
κρατήσουμε τελικά από τα προηγούμενα σημεία που είχαμε εντοπίσει. Ορίζουμε να κρατάμε τα σημεία με τις 500 μεγαλύτερες τιμές.

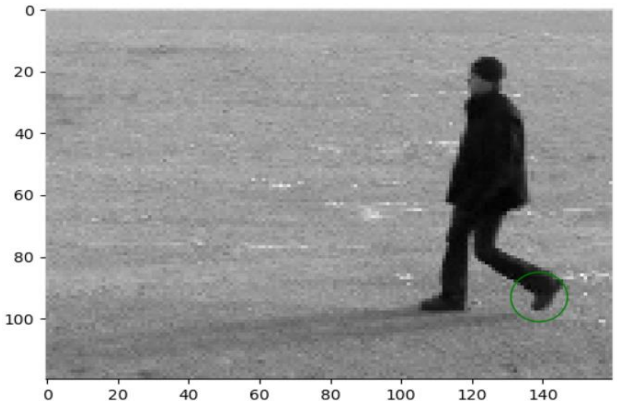
Αντίστοιχα με την find_coords() και η final_points επιστρέφει ένα Nx4 πίνακα που περιέχει τις συντεταγμένες των τελικών σημείων και την χωρική κλίμακά σ στην οποία αυτά εντοπίζονται. Στην περίπτωσή μας είναι N=500.

Χρησιμοποιώντας την συνάρτηση show_detection() από τον βοηθητικό κώδικα του εργαστηρίου απεικονίζουμε στα βίντεό μας τα τελικά σημεία ενδιαφέροντος που επιλέγουμε:



Σχήμα 15: 3D Harris Detection



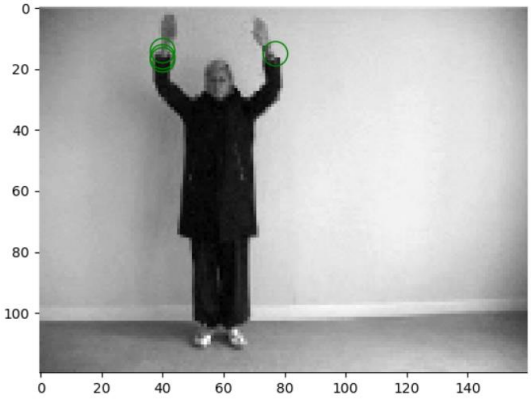


Σχήμα 16: Gabor Detection

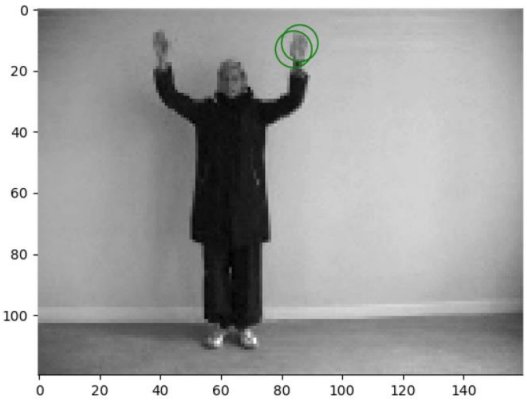
Παρατηρούμε ότι ο Harris ανιχνευτής εντοπίζει σημεία-γωνίες τα οποία παρουσιάζουν χρονική μεταβολή (κίνηση) ενώ με τον Gabor ανιχνευτή εντοπίζονται σημεία που παρουσιάζουν μεταβολή της φωτεινότητας στην διάρκεια του χρόνου ανεξάρτητα από το εάν αυτά αποτελούν γωνίες στα frames. Έτσι θα μπορούσαμε να πούμε πως με τον Gabor ανιχνεύουμε γενικότερα την κίνηση, ως αλλαγή της φωτεινότητας των voxel. Όμως αυτό συνεπάγεται πως μαζί με τα αντικείμενα που παρουσιάζουν κίνηση είναι πιθανό να εντοπίζονται και οι σκιές αυτών που κινούνται μαζί τους κάτι το οποίο μπορεί πολλές φορές δυσχεραίνει την ορθή αναγνώριση των κινήσεων ή δράσεων. Από την άλλη ο 3D Harris ανιχνευτής εστιάζει στην κίνηση ακμών στην διάρκεια του χρόνου, εντοπίζοντας έτσι με μεγαλύτερη ακρίβεια τα αντικείμενα που κινούνται μέσα στο βίντεο.

Σε κάθε περίπτωση και οι δύο ανιχνευτές εντοπίζουν με ικανοποιητική ακρίβεια τις πιο 'έντονες' κινήσεις, οι οποίες είναι στα χέρια και στα πόδια έναντι των κινήσεων του κορμού ή προσώπου.

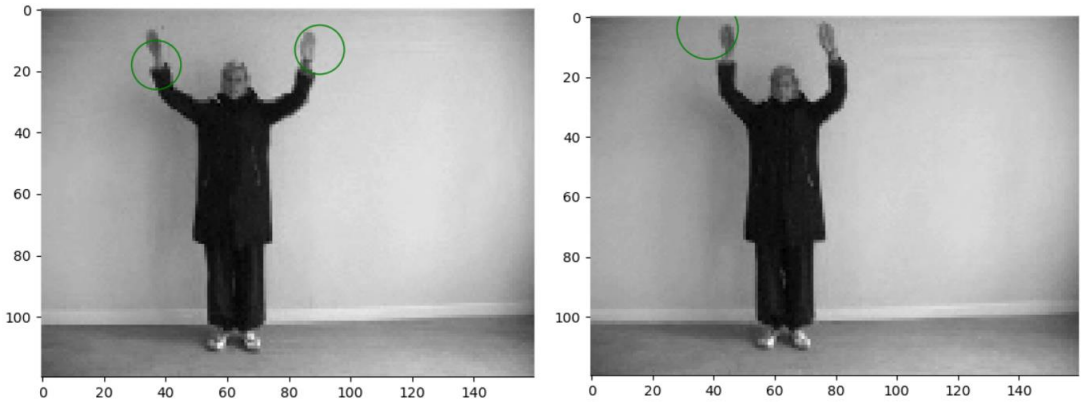
Πειραματιζόμαστε με διαφορετικές τιμές χωρικής κλίμακας σ για τον Harris 3D Detector :



(α)



(β)



(γ) (δ)

(ε)

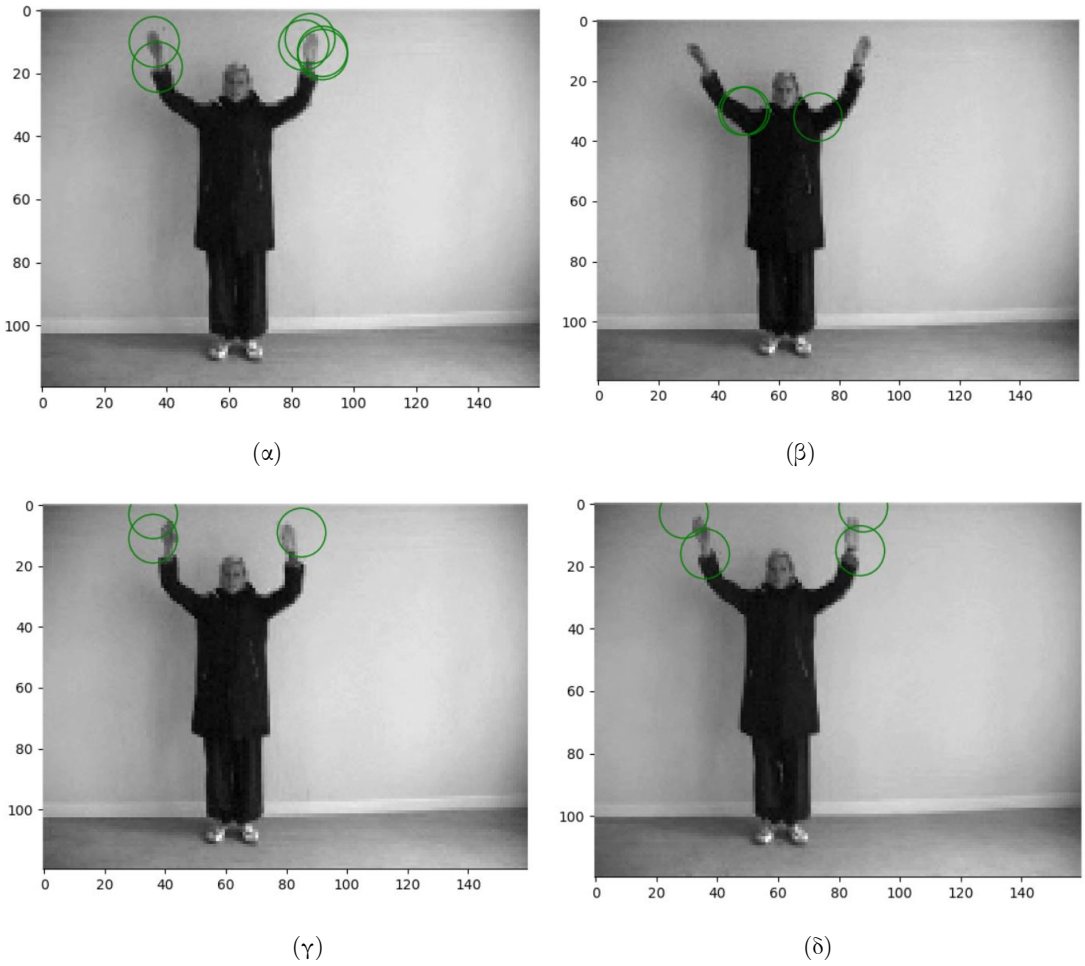
Σχήμα 17: Ανίχνευση Harris με α) σ=2 β) σ=3 γ) σ=4 δ) σ=5 ε) σ=6

Τυπώνουμε επίσης το πλήθος των σημείων ενδιαφέροντος που ανιχνεύονται για τις διαφορετικές κλίμακες έπειτα από την κατωφλιοποίηση του κριτηρίου:

```
Found  9686 interest points with spatial scale sigma 2
Found  16214 interest points with spatial scale sigma 3
Found  24844 interest points with spatial scale sigma 4
Found  36993 interest points with spatial scale sigma 5
Found  48559 interest points with spatial scale sigma 6
```

Όπως βλέπουμε από τα παραπάνω αποτελέσματα, για μεγαλύτερη χωρική κλίμακα σ, αυξάνονται τα σημεία ενδιαφέροντος που ανιχνεύονται, ωστόσο αυτά χάνουν την ακρίβεια που έχουμε εντός μιας σχετικά μικρότερης κλίμακας η οποία μπορεί να ανιχνεύσει με μεγαλύτερη 'λεπτομέρεια' τα σημεία κίνησης (Σχήμα 17). Θεωρούμε πως η ενδεικτική τιμή σ=4 δίνει αρκετά ικανοποιητικά αποτελέσματα όπως μπορούμε να δούμε και από το Σχήμα 17.γ

Δοκιμάζουμε για σταθερή χωρική κλίμακα σ=4 να αλλάξουμε την χρονική κλίμακα τ :



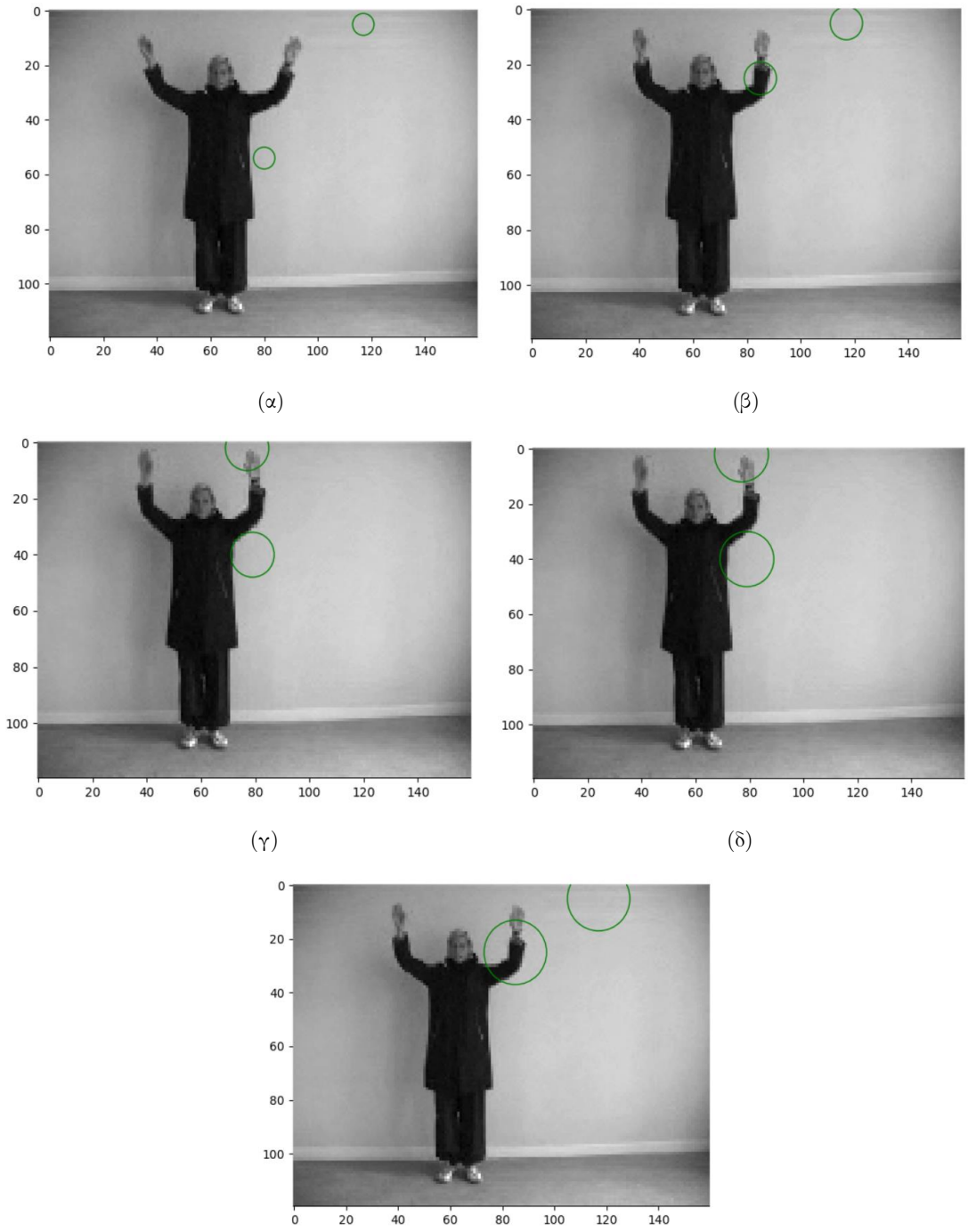
Σχήμα 18: Ανίχνευση Harris με σ=4 και α)τ=1.0, β)τ= 1.5 γ)τ=2.0 ,δ)τ=2.5

Το πλήθος των σημείων που εντοπίζονται τώρα είναι το εξής:

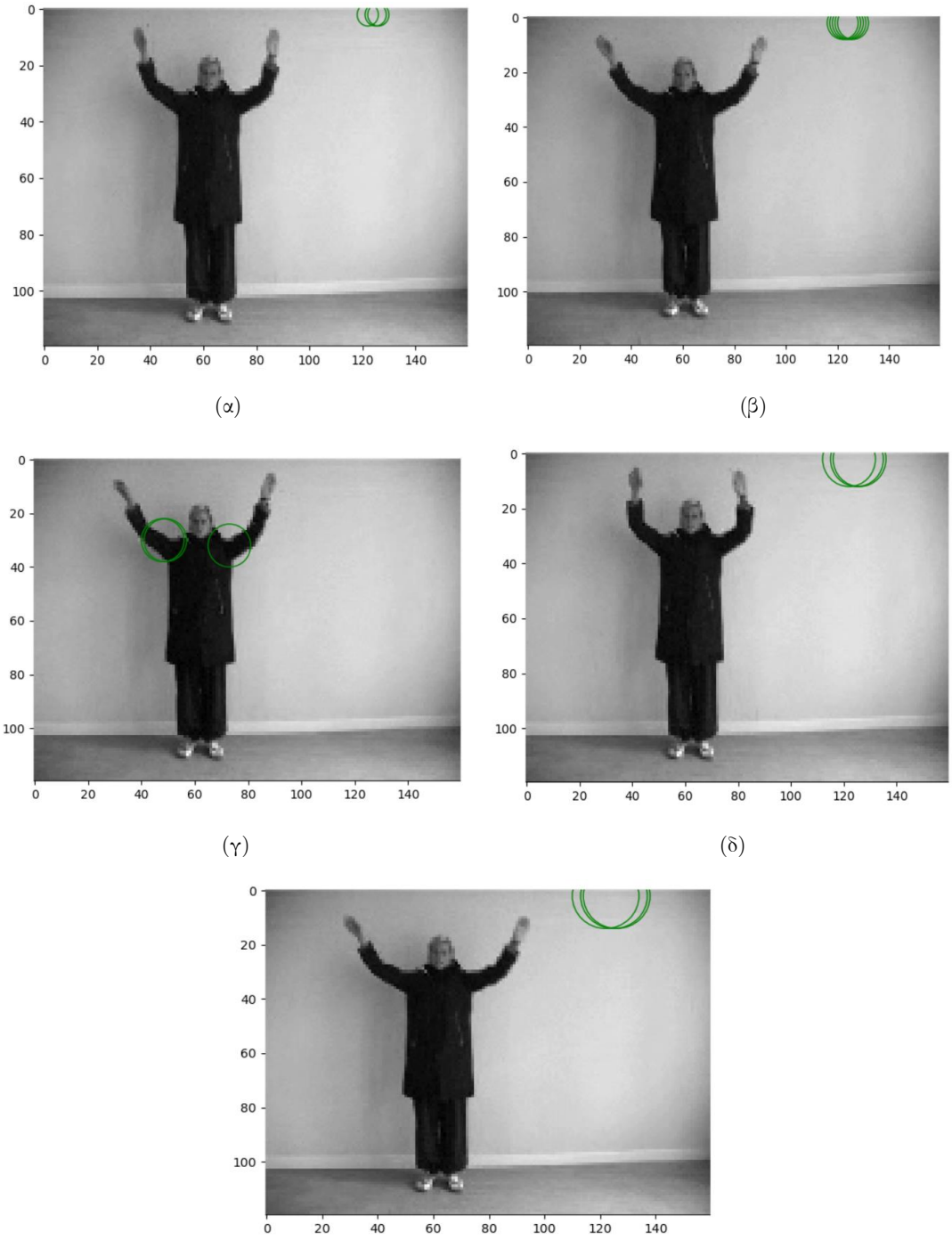
```
Found  24210 interest points with time scale taf 1.0 and spatial scale sigma 4
Found  24844 interest points with time scale taf 1.5 and spatial scale sigma 4
Found  25718 interest points with time scale taf 2.0 and spatial scale sigma 4
Found  27125 interest points with time scale taf 2.5 and spatial scale sigma 4
```

Βλέπουμε ότι για αυξανόμενη χωρική κλίμακα εντοπίζονται περισσότερα σημεία ενδιαφέροντος, χωρίς ωστόσο η ακρίβεια της ανίχνευσης να χάνεται τόσο σε σχέση με την αυξομείωση της χωρικής κλίμακας. Επομένως μπορούμε να πούμε πως ο Harris Detector εξαρτάται τόσο από την χωρική όσο και από την χρονική κλίμακα για το αποτέλεσμα της ανίχνευσης που θα δώσει, και με κατάλληλο συνδυασμό των δύο αυτών, μπορούμε να πετύχουμε σχετικά μεγάλη ακρίβεια στην ανίχνευσή μας.

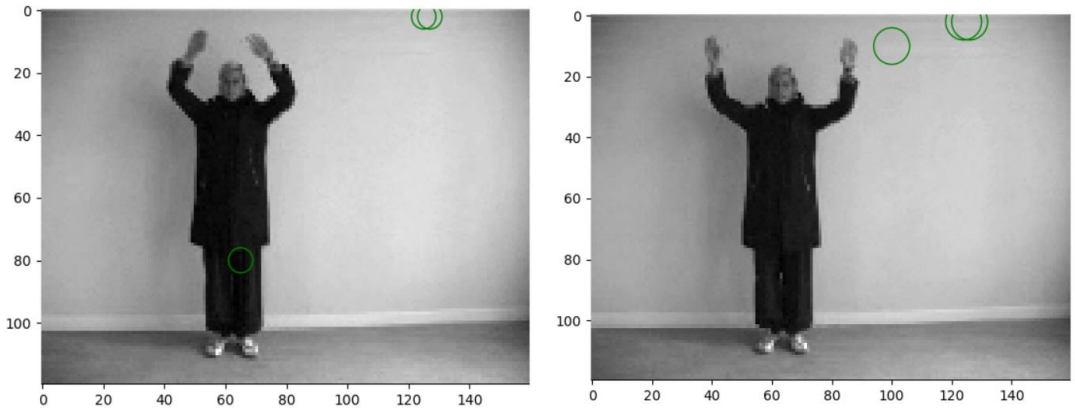
Για διάφορους συνδυασμούς χωρικών και χρονικών κλιμάκων στον Gabor ανιχνευτή μας παίρνουμε τα εξής αποτελέσματα:



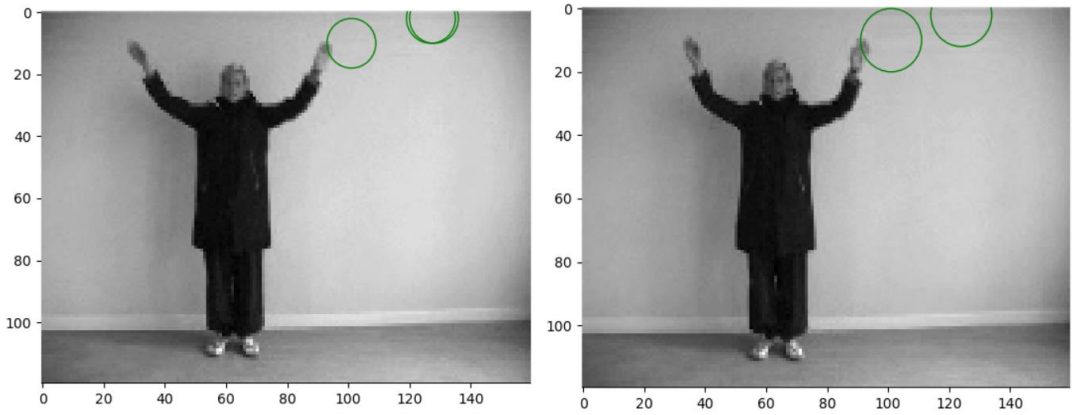
Σχήμα 19: Ανίχνευση Gabor με τ=1 και: α) σ=2 , β) σ=3, γ) σ=4, δ) σ=5, ε) σ=6



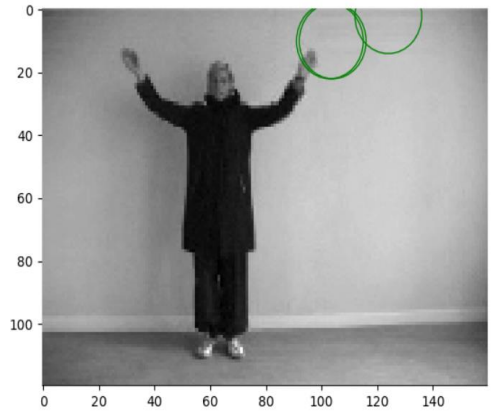
Σχήμα 20: Ανίχνευση Gabor με τ=1.5 και: α) σ=2 , β) σ=3, γ) σ=4, δ) σ=5, ε) σ=6



(α) (β)

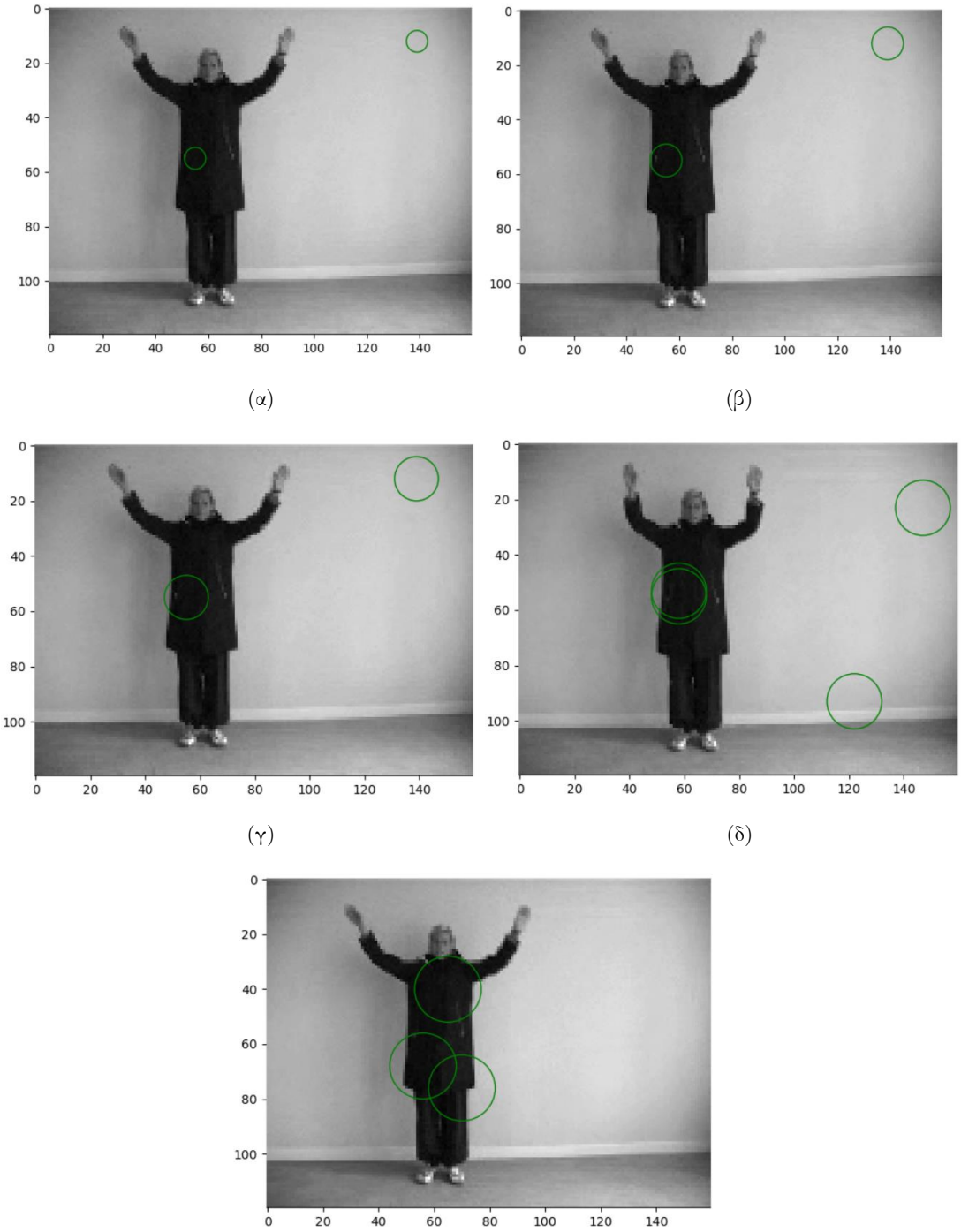


(γ) (δ)



(ε)

Σχήμα 21: Ανίχνευση Gabor με τ=2 και: α) σ=2 , β) σ=3, γ) σ=4, δ) σ=5, ε) σ=6



Σχήμα 22: Ανίχνευση Gabor με τ=2.5 και: α) σ=2 , β) σ=3, γ) σ=4, δ) σ=5, ε) σ=6

Τυπώνουμε και εδώ το πλήθος των σημείων ενδιαφέροντος που ανιχνεύονται για τις διαφορετικές κλίμακες έπειτα από την κατωφλιοποίηση του κριτηρίου:

```
Found  1087488 interest points with time scale taf 1.0 and spatial scale sigma 2
Found  1087488 interest points with time scale taf 1.0 and spatial scale sigma 3
Found  1087488 interest points with time scale taf 1.0 and spatial scale sigma 4
Found  1087488 interest points with time scale taf 1.0 and spatial scale sigma 5
Found  1087488 interest points with time scale taf 1.0 and spatial scale sigma 6
Found  1077269 interest points with time scale taf 1.5 and spatial scale sigma 2
Found  1077269 interest points with time scale taf 1.5 and spatial scale sigma 3
Found  1077269 interest points with time scale taf 1.5 and spatial scale sigma 4
Found  1077269 interest points with time scale taf 1.5 and spatial scale sigma 5
Found  1077269 interest points with time scale taf 1.5 and spatial scale sigma 6
Found  1103160 interest points with time scale taf 2.0 and spatial scale sigma 2
Found  1103160 interest points with time scale taf 2.0 and spatial scale sigma 3
Found  1103160 interest points with time scale taf 2.0 and spatial scale sigma 4
Found  1103160 interest points with time scale taf 2.0 and spatial scale sigma 5
Found  1103160 interest points with time scale taf 2.0 and spatial scale sigma 6
Found  1130549 interest points with time scale taf 2.5 and spatial scale sigma 2
Found  1130549 interest points with time scale taf 2.5 and spatial scale sigma 3
Found  1130549 interest points with time scale taf 2.5 and spatial scale sigma 4
Found  1130549 interest points with time scale taf 2.5 and spatial scale sigma 5
Found  1130549 interest points with time scale taf 2.5 and spatial scale sigma 6
```

Παρατηρούμε ότι το πλήθος των σημείων ενδιαφέροντος που ανιχνεύονται, αυτήν την φορά εξαρτάται από την χρονική κλίμακα τ και όχι από την χωρική κλίμακα σ, κάτι το οποίο αναμέναμε καθώς όπως είπαμε η ανίχνευση βασίζεται στο χρονικό φιλτράρισμα του βίντεο βάσει των δύο χρονικών αποκρίσεων Gabor, hev και hod. Βλέπουμε αντίστοιχα με τον ανιχνευτή Harris και την χωρική του κλίμακα, πως και εδώ για αυξανόμενη χρονική κλίμακα εντοπίζονται και περισσότερα σημεία ενδιαφέροντος τα οποία όμως είναι λιγότερο ‘ακριβή’ ως προς την ανίχνευση των σημείων κίνησης.

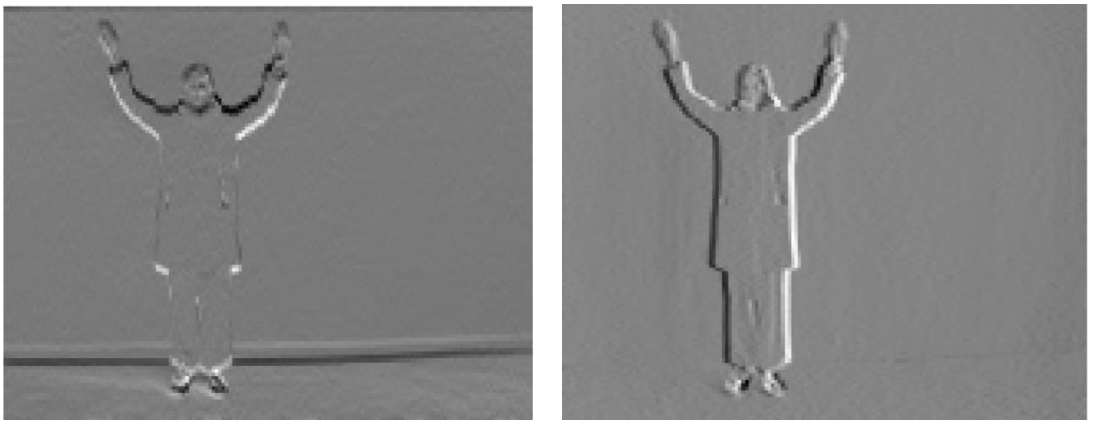
2.2 Χωρο-χρονικοί Ιστογραφικοί Περιγραφητές

Έχοντας υλοποιήσει τους παραπάνω ανιχνευτές, μπορούμε να τους χρησιμοποιήσουμε σε οποιοδήποτε βίντεο για να εξάγουμε τους τοπικούς περιγραφητές τους γύρω από τα σημεία ενδιαφέροντος που αυτοί εντοπίζουν. Με τον όρο τοπικό περιγραφητή εννοούμε μια γειτονιά γύρω από κάποιο σημείο ενδιαφέροντος. Στο μέρος αυτό θα ασχοληθούμε με τρεις τοπικούς περιγραφητές:

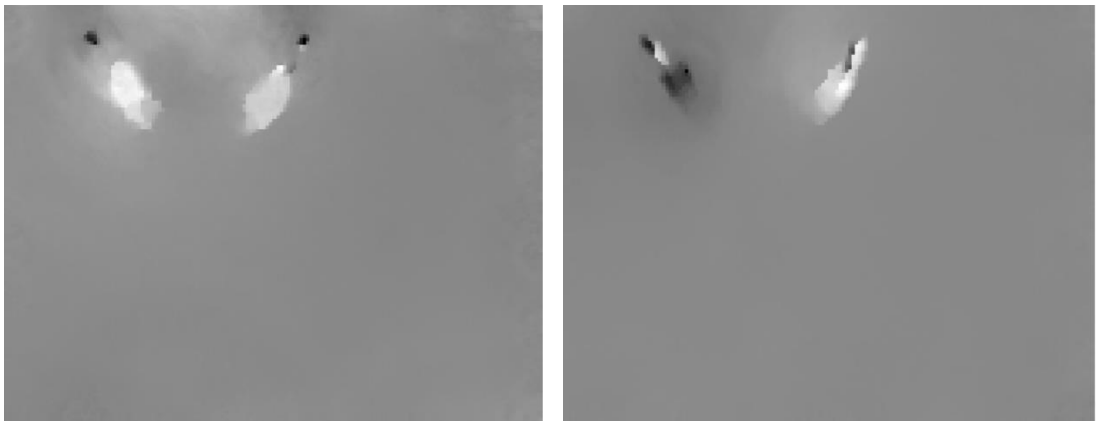
- HOG (Histogram of Oriented Gradients)
- HOF (Histogram of Oriented Flows)
- HOG-HOF που αποτελεί συνδυασμό των δύο παραπάνω.

Για να εξάγουμε τους παραπάνω περιγραφητές, απαραίτητη προϋπόθεση είναι να υπολογίσουμε για κάθε frame το διάνυσμα κλίσης (gradient) και την οπτική ροή TV-L1. Για τον σκοπό αυτό κατασκευάζουμε τις συναρτήσεις get_gradient() και get_optical_flow() με ορίσματα το υπό εξέταση βίντεο κάθε φορά.

Το αποτέλεσμα που παίρνουμε από την εφαρμογή τους στο ‘video handwaving’ που χρησιμοποιήσαμε στο μέρος 2.1 είναι το εξής:



(α) (β)



(α) (β)

Σχήμα 23: α) Optical Flow x of frame 15 , β) Optical Flow y of frame 15

Αποθηκεύουμε τα αντίστοιχα animation που προκύπτουν για τα διανύσματα κλίσης και οπτικής ροής.

Ορίζουμε τις τρεις συναρτήσεις που θα υπολογίζουν τους περιγραφητές μας:

- HOG_desc()
- HOF_desc()
- HOG_HOF_desc()

Κάθε μία από αυτές δέχεται σαν όρισμα το video που θέλουμε να κατηγοριοποιήσουμε, τα τελικά σημεία ενδιαφέροντος που επιλέγουμε για το βιντεο αυτό (προκύπτουν από την final_points), το πλήθος των χαρακτηριστικών nbins βάσει του οποίου θέλουμε να εξάγουμε το ιστόγραμμά μας, το πλέγα grid στο οποίο θέλουμε να χωρίζουμε κάθε φορά το υπό ανάλυση διανυσματικό πεδίο μας και την γειτονιά neighb που ορίζει ένα patch γύρω από κάθε σημείο ενδιάφεροντος. Τα grid και neighb είναι της μορφής np.array([n,m]). Για κάθε σημείο ενδιαφέροντος που εισάγουμε στην συνάρτηση μας, θέλουμε να εξάγουμε τον τοπικό περιγραφητή του γύρω από μια (τετραγωνική) περιοχή 4xσ. Για τον σκοπό θα απομονώσουμε γύρω από αυτό ένα patch διαστάσεων 4xσ οι οποίες ορίζονται από το όρισμα neighb. Επειδή κάποια σημεία ενδιαφέροντος μπορεί να

ανιχνεύονται στα όρια του frame, πραγματοποιούμε αρχικά zero padding του βίντεο σύμφωνα με τις διαστάσεις του patch-γειτονιάς που έχουμε ορίσει.

Στους υπολογισμούς θέτουμε το μέγεθος του πλέγματος grid ίδο με np.array([8,8]) και μέγεθος γειτονιάς neighb ίσο με np.array([4,sigma]). Επίσης ορίζουμε nbins = 3.

Ο εκάστοτε τοπικός περιγραφητής προκύπτει με την βοήθεια της συνάρτησης orientation_histogram() που μας δίνεται στο υλικό του εργαστηρίου cv23_lab2_2_utils. Η συνάρτηση αυτή υπολογίζει κάθε φορά τον τοπικό περιγραφητή γύρω από ένα σημείο ενδιαφέροντος και δέχεται ως ορίσματα τα διανυσματικά πεδία γύρω από το σημείο αυτό (patches), καθώς επίσης και το πλήθος των χαρακτηριστικών nbins και το πλέγμα grid (τα δύο τελευταία προσδιορίζονται ως όρισμα της εξωτερικής συνάρτησης 'desc' του περιγραφητή μας). Η συνάρτηση orientation_histogram() επιστρέφει το μονοδιάστατο διάνυσμα του τοπικού περιγραφητή που είναι μεγέθους n x m x nbins. Έτσι για τα συνολικά N σημεία ενδιαφέροντος οι συνάρτησεις HOG_desc και HOF_desc επιστρέφουν έναν πίνακα μεγέθους N x (n x m x nbins), δηλαδή έναν πίνακα του οποίου κάθε σειρά περιέχει τον τοπικό περιγραφητή ενός σημείου ενδιαφέροντος. Ομοίως κατασκευάζουμε την συνάρτηση HOG_HOF_desc με την διαφορά ότι αυτή επιστρέφει έναν πίνακα N x 2x(n x m x nbins), διότι κάθε σειρά περιέχει το εκάστοτε διάνυσμα του τοπικού περιγραφητή HOG ακολουθούμενο διαδοχικά από το αντίστοιχο διάνυσμα του τοπικού περιγραφητή HOF.

Χωρίζουμε τα βίντεο που μας δίνονται στον φάκελο του μέρους 2 σε train και test data, όπως μας υποδεικνύεται στο training_videos.txt. Αποθηκεύουμε τα πρώτα σε έναν πίνακα desc_train και τα δεύτερα σε έναν πίνακα desc_test. Οι πίνακες αυτοί είναι μονοδιάστατοι και έχουν μέγεθος όσο και το πλήθος Ntrain και Ntest των βίντεο αντίστοιχα. Κάθε στοιχείο τους είναι ένα τρισδιάστατος πίνακας που αναπαριστά ένα sample video.

Για καθένα από τα train και test video εξάγουμε αρχικά το label του, δηλαδή την ετικέτα που δηλώνει ποια από τις τρεις ανθρώπινες δράσεις (running, handwaving, walking) αναπαριστά. Αποθηκεύουμε τα labels στις λίστες train_labels και test_labels αντίστοιχα, οι οποίες έτσι περιέχουν Ntrain και Ntest στοιχεία κατά αντιστοιχία.

Δημιουργούμε τρεις train λίστες και τρεις test λίστες, μία για τον κάθε περιγραφητή μας σε συνδυασμό με τον Harris ανιχνευτή:

train_list_Harris HOG, train_list_Harris_HOF, train_list_Harris_HOG_HOF και test_list_Harris_HOG, test_list_Harris_HOF, test_list_Harris_HOG_HOF.

Κάθε μία από τις τρεις πρώτες λίστες περιέχει Ntrain στοιχεία, καθένα από τα οποία είναι ένας πίνακας μεγέθους N x (μέγεθος αντίστοιχου περιγραφητή), όπου N το πλήθος των σημείων ενδιαφέροντος που χρησιμοποιούμε στον κάθε περιγραφητή. Οι τρεις τελευταίες λίστες ορίζονται αντίστοιχα, με πλήθος στοιχείων Ntest.

Για λόγους πληρότητας κατασκευάζουμε και τις έξι αντίστοιχες labeled λίστες, των οποίων κάθε στοιχείο περιέχει επιπλέον μία στήλη, η οποία καθορίζει την ετικέτα του βίντεο που περιγράφει. Από τις labeled λίστες εξάγουμε τις unlabeled λίστες train_Harris_HOG_videos, test_Harris_HOG_videos, train_Harris_HOF_videos,

test_Harris_HOF_videos, train_Harris_HOG_HOF_videos, test_Harris_HOG_HOF_videos.

Βάσει αυτών μπορούμε τώρα να εξάγουμε την τελική αναπαράσταση (global representation) κάθε βίντεο με τη χρήση της BoVW (Bag of Visual Words). Για τον σκοπό αυτό χρησιμοποιούμε την έτοιμη συνάρτηση bag_of_words() που μας δίνεται στο υλικό του εργαστηρίου. Για κάθε περιγραφητή, η συνάρτηση αυτή δέχεται ως ορίσματα τις αντίστοιχες λίστες train_videos και test_videos αυτού, και τον αριθμό D των οπτικών λέξεων, δηλαδή των κεντροειδών του K-means.

Επιλέγουμε τιμή D=3 για τον κώδικά μας.

Εξάγουμε τα 6 BoVW διανύσματα για τον ανιχνευτή Harris.

2.3.3

Εισάγουμε τα διανύσματα BoVW που βρήκαμε προηγουμένως στην συνάρτηση svm_train_test() μαζί με τα αντίστοιχα labels τους. Η συνάρτηση αυτή υλοποιεί έναν SVM ταξινομητή κατάλληλα προσαρμοσμένο για πολλαπλές κλάσεις και επιστρέφει την ακρίβεια της ταξινόμησης καθώς επίσης και τις προβλέψεις που έγιναν.

Έτσι για τις τρεις διαφορετικές global representations του Harris 3D Detector παίρνουμε τα εξής αποτελέσματα:

- HOG Harris

```
Accuracy of classification using BoW of Harris Detector with HOG descriptors: 0.5833333333333334

Predictions of classification using BoW of Harris Detector with HOG descriptors:
 ['running', 'running', 'running', 'handwaving', 'handwaving', 'handwaving', 'handwaving', 'handwaving', 'running', 'running',
'running', 'handwaving']

Real labels of test samples:
 ['running', 'running', 'running', 'running', 'handwaving', 'handwaving', 'handwaving', 'handwaving', 'walking', 'walking', 'wa
lking', 'walking']
```

- HOF Harris

```
Accuracy of classification using BoW with HOF descriptors: 0.5

Predictions of classification using BoW with HOF descriptors:
 ['walking', 'handwaving', 'walking', 'running', 'handwaving', 'handwaving', 'handwaving', 'handwaving', 'running', 'walking',
'handwaving', 'running']

Real labels of test samples:
 ['running', 'running', 'running', 'running', 'handwaving', 'handwaving', 'handwaving', 'handwaving', 'walking', 'walking', 'wa
lking', 'walking']
```

- HOG_HOF Harris

```
Accuracy of classification using BoW with HOG_HOF descriptors: 0.5

Predictions of classification using BoW with HOG descriptors:
 ['walking', 'handwaving', 'walking', 'running', 'handwaving', 'handwaving', 'handwaving', 'handwaving', 'running', 'walking',
'handwaving', 'running']

Real labels of test samples:
 ['running', 'running', 'running', 'running', 'handwaving', 'handwaving', 'handwaving', 'handwaving', 'walking', 'walking', 'wa
lking', 'walking']
```

Βλέπουμε πως ο HOG περιγραφητής δίνει τα καλύτερα αποτελέσματα ενώ οι άλλοι δύο περιγραφητές δίνουν ακρίβεια της τάξεως 50% κάτι το οποίο δεν είναι ικανοποιητικό. Ένα μέρος των εσφαλμένων προβλέψεων οφείλεται στο γεγονός ότι τα βίντεό μας είναι αρκετά θορυβώδη, ώστε ακόμα και έπειτα από την εξομάλυνση που εφαρμόζουμε σε αυτά να μην μπορούμε να έχουμε πολύ καλή ακρίβεια στην ανίχνευση των σημείων ενδιαφέροντος.

2.3.4

Εκτελούμε τα ίδια βήματα με προηγουμένως και για τον Gabor ανιχνευτή μας, τα αποτελέσματα που παίρνουμε είναι της τάξης του 33%.

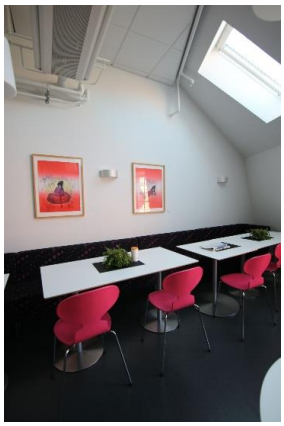
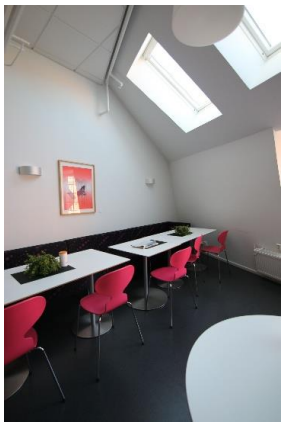
Δεδομένου ότι η ανίχνευση για τα τυχαία βίντεο που χρησιμοποιήσαμε ήταν αρκετά ακριβής, πιθανότερη αιτία των χαμηλών ποσοστών ακρίβειας είναι κάποια λάθος υλοποίηση των περιγραφητών.

Μέρος 3: Συνένωση Εικόνων (Image Stitching) για Δημιουργία Πανοράματος

Στο μέρος αυτό καλούμαστε να δημιουργήσουμε ένα πανόραμα, συνενώνοντας διαδοχικές εικόνες ενός δωματίου που πάρθηκαν από το ίδιο σημείο με απλή περιστροφή της κάμερας.

Η διαδικασία που θα περιγράψουμε αφορά τη συνένωση δύο εικόνων και στη συνέχεια θα ακολουθήσει η δημιουργία του πανοράματος.

Αρχικά διαβάζουμε τις δύο πρώτες εικόνες.

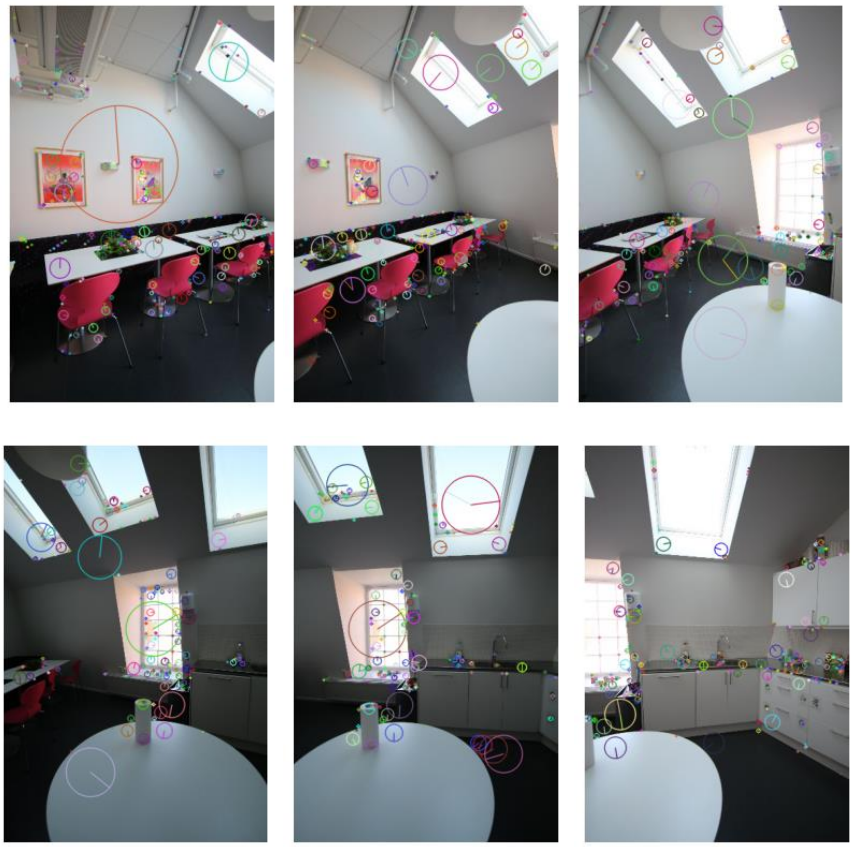
Για τη συνένωση των δύο εικόνων, είναι απαραίτητο το οπτικό πεδίο της πρώτης εικόνας να επικαλύπτεται με το οπτικό πεδίο της δεύτερης εικόνας έτσι ώστε να μπορούν να συνδυαστούν ώστε να προκύψει μια τελική εικόνα που θα αποτελεί ένα ευρύ πλάνο του αριστερού και του δεξιού οπτικού πεδίου.

Πρώτο βήμα στη συνένωση των δύο εικόνων και κατ' επέκταση στην εύρεση του πανοράματος είναι η εξαγωγή και το ταίριασμα SIFT χαρακτηριστικών ανάμεσα στις εικόνες.

Ο λόγοι που χρησιμοποιούμε τα χαρακτηριστικά SIFT είναι οι εξής:

- Τα χαρακτηριστικά SIFT είναι ανεξάρτητα από τη κλίμακα και το προσανατολισμό, που σημαίνει πως μπορούμε να εξάγουμε ίδια χαρακτηριστικά από εικόνες που έχουν υποστεί αλλαγές στον προσανατολισμό ή τη κλίμακα
- Μπορούμε να ανιχνεύσουμε σε διαφορετικές κλίμακες κάτι που οδηγεί σε πιο ακριβές ταίριασμα μεταξύ των εικόνων.
- Παράλληλα με την ανίχνευση των χαρακτηριστικών, ο αλγόριθμος SIFT υπολογίζει και έναν περιγραφητή για κάθε χαρακτηριστικό, ο οποίος κωδικοποιεί τη πληροφορία εμφάνισης και σχήματος του εκάστοτε χαρακτηριστικού. Οι

περιγραφητές αυτοί είναι ανθεκτικοί σε μεταβολές τη φωτεινότητας και συμβάλλουν σε ένα αποδοτικό τρόπο ταιριάσματος των εικόνων.



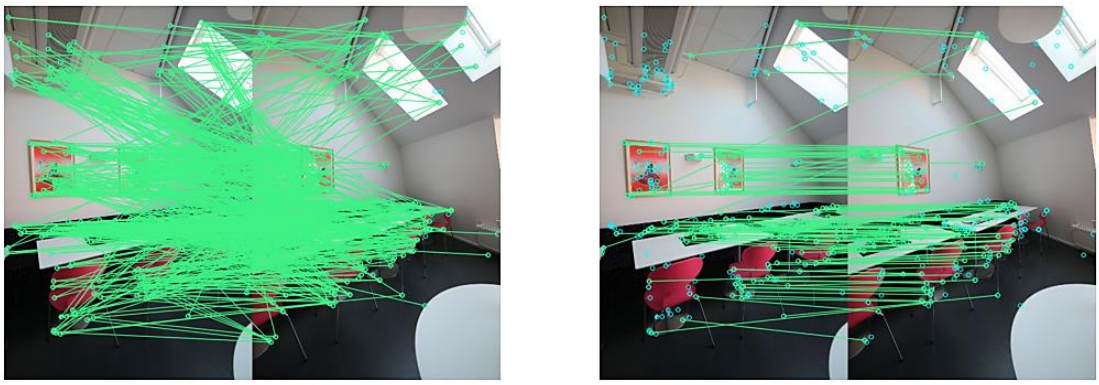
Σχήμα 24: SIFT keypoints για τις 6 εικόνες με τις οποίες θα δημιουργηθεί το πανόραμα

Στη συνέχεια, θα πρέπει να ταιριάξουμε τα χαρακτηριστικά ανάμεσα στις εικόνες, έτσι ώστε να αναγνωρίσουμε τις εικόνες που επικαλύπτει η μια την άλλη. Συγκεκριμένα, θα βρούμε την αντιστοίχιση κάθε χαρακτηριστικού της εικόνας A με τα δύο κοντινότερα της εικόνας B και έπειτα με εφαρμογή του κριτηρίου Lowe θα κρατήσουμε μόνο τις πιο έγκυρες αντιστοιχίσεις.

Για την προσέγγιση των κοντινότερων γειτόνων θα κάνουμε χρήση της βιβλιοθήκης FLANN. Ο αλγόριθμος αυτός αρχικά δημιουργεί μια δομή δεδομένων από indexes με σκοπό την επιτάχυνση της αναζήτησης κοντινότερων γειτόνων. Ύστερα, ο αλγόριθμος για να βρει το κοντινότερο ταίριασμα, ψάχνει το index για την αναζήτηση των ομοιότερων χαρακτηριστικών με βάση μια απόσταση. Για να το πετύχει αυτό χρησιμοποιεί αλγόριθμο KDTREE με αριθμό δέντρων ίσο με 5.

Για να βελτιώσουμε τη ποιότητα των αντιστοιχίσεων εφαρμόζουμε το κριτήριο Lowe με κατώφλι 0.8 για να απορριφθούν οι περισσότερες λανθασμένες αντιστοιχίσεις. Η ιδέα αυτού του κριτηρίου είναι να απορρίψει τα matches που προέρχονται από τους πρώτους και δεύτερους κοντινότερους γείτονες οι οποίοι βρίσκονται σε πολύ κοντινή απόσταση μεταξύ τους. Αν ο λόγος απόστασης μεταξύ του κοντινότερου

και δεύτερου κοντινότερου γείτονα είναι μικρότερος από το κατώφλι , το συγκεκριμένο ταίριασμα απορρίπτεται.

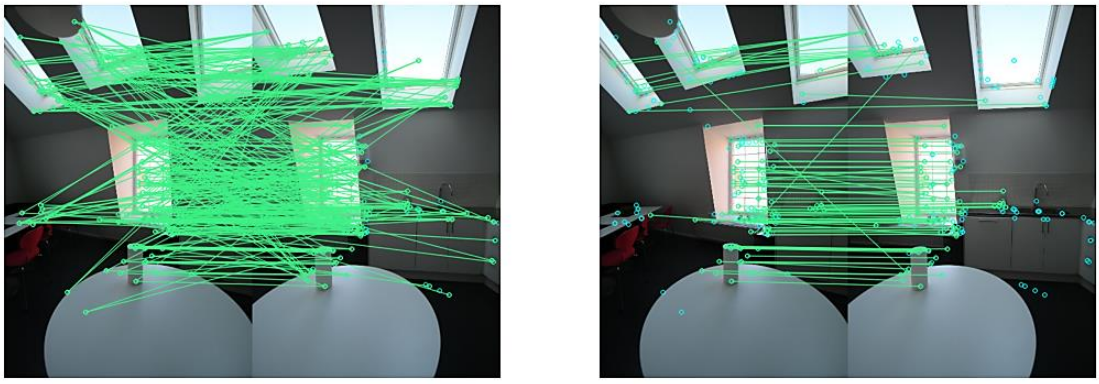


(α) (β)

Σχήμα 25: Matching features μεταξύ της πρώτης και της δεύτερης εικόνας : (α) Όλα τα ταιριάσματα που επιστρέφει ο FLANN. (β) Τα τελικά matches έπειτα από εφαρμογή κριτηρίου Lowe

Όπως φαίνεται και από την εικόνα, με το κριτήριο Lowe καταφέρνουμε να απορρίψουμε αρκετά λανθασμένα ταιριάσματα. Συγκεκριμένα, ενώ αρχικά είχαμε 361 matches έπειτα από τη κατωφλιοποίηση μένουν 138 συνολικά matches.

Αντίστοιχα αποτελέσματα έχουμε και για το ταίριασμα των υπόλοιπων εικόνων.



(α) (β)

Σχήμα 26: Ταίριασμα μεταξύ της 4ης και της 5ης εικόνας του σετ εικόνων

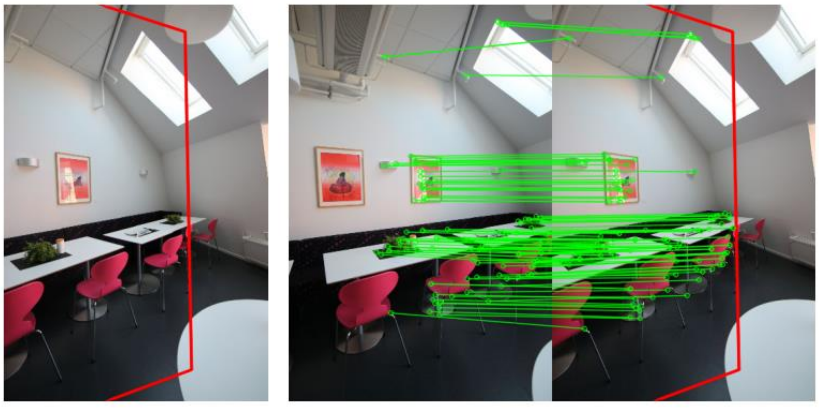
Πλέον, έχοντας βρει τις αντιστοιχίσεις μεταξύ των δύο εικόνων προχωράμε στον υπολογισμό της ομογραφίας H_{BA} για τη μετάβαση της εικόνας A στο πλαίσιο αναφοράς της εικόνας B.

Για τον υπολογισμό της ομογραφίας θα χρησιμοποιήσουμε τον αλγόριθμο RANSAC για να απορρίψουμε τους περισσότερους outliers δηλαδή τα ανιχνευμένα matches τα οποία δεν είναι σωστά. Για το σκοπό αυτό θα θέσουμε τιμή κατωφλίου ίση με 5.

Για τις εικόνες 1 και 2 λαμβάνουμε τον εξής πίνακα ομογραφίας μαζί με το σχήμα μετασχηματισμού της εικόνας 1 ώστε να ταιριάξει στην εικόνα 2.

Homography Matrix

1.67953423e+00	3.18765044e-02	-1.95564542e+02
4.66032589e-01	1.42490832e+00	-1.06989993e+02
1.81544824e-03	1.68164580e-05	1.00000000e+00



(α) (β)

Σχήμα 27: Homography and Perspective transform: (α) Τα όρια της εικόνας A στο πεδίο αναφοράς της εικόνας B. (β) Τα matches της εικόνας A με το μετασχηματισμό της στην εικόνα B

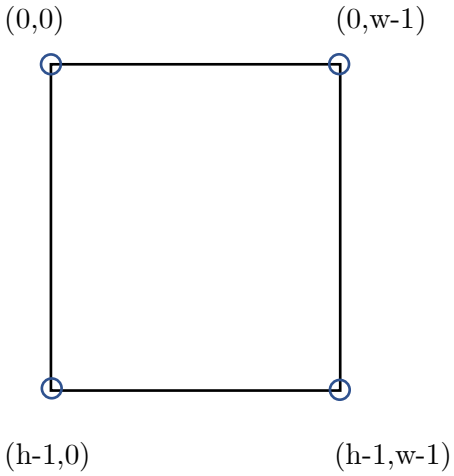
Η ομογραφία βρέθηκε χρησιμοποιώντας τη συνάρτηση findHomography της βιβλιοθήκης cv2. Συγκεκριμένα, πρώτα εντοπίζουμε τις τοποθεσίες των keypoints που ταίριαξαν επιτυχώς μεταξύ των δύο εικόνων και τις περνάμε στη συνάρτηση για να αποκτήσουμε τον 3x3 πίνακα ομογραφίας.

Τέλος δημιουργούμε τη συνάρτηση stitchImages() συνοψίζοντας τα βήματα 5 και 6.

Η συνάρτηση αυτή λαμβάνει ως είσοδο της δύο εικόνες που πρόκειται να συνενωθούν μαζί με τον πίνακα μετασχηματισμού που λάβαμε στο προηγούμενο βήμα και τη μέθοδο ένωσης των δύο εικόνων, δηλαδή είτε να συνενώσουμε την αριστερή εικόνα με τη δεξιά οπότε θα έχουμε left_to_right είτε τη δεξιά εικόνα με την αριστερή όπου θα είναι right_to_left.

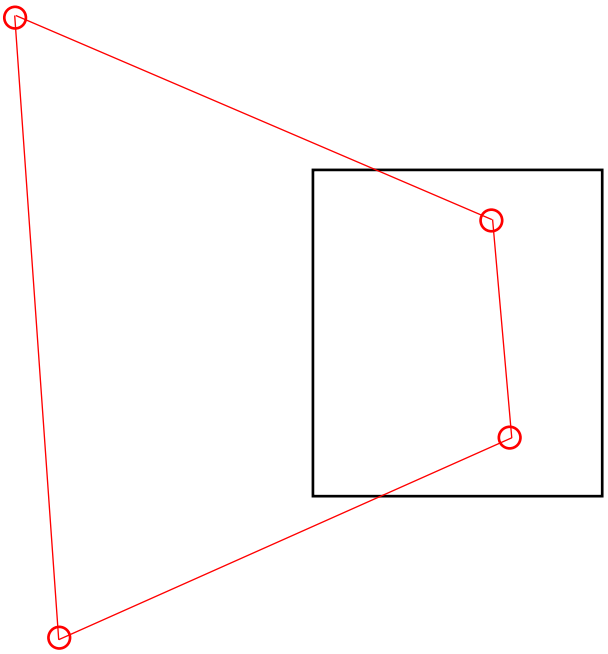
Τα βήματα της συνάρτησης είναι τα εξής:

I. Όρισε τις αρχικές 4 γωνίες της εικόνας που πρόκειται να μετασχηματιστεί



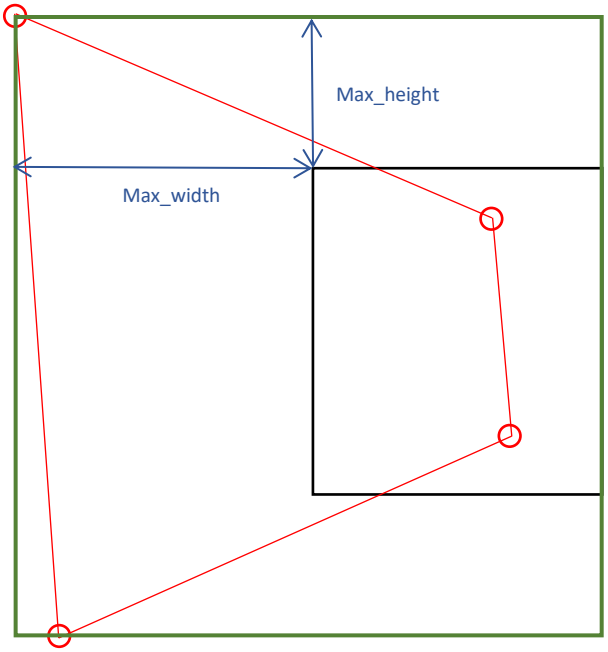
II. Λάβε τις συντεταγμένες των σημείων της μετασχηματισμένης εικόνας εκπεφρασμένες ως προς τις συντεταγμένες αυτές μέσω της συνάρτησης perspectiveTransform().
dst = cv2.perspectiveTransform(pts, H) όπου H ο πίνακας ομογραφίας.
Οι συντεταγμένες που λαμβάνουμε για τη συνένωση της πρώτης με τη δεύτερη εικόνα από αριστερά είναι:

[-195.56454 -106.98999]
[-176.53903 664.9049]
[258.67023 503.64767]
[249.61089 37.480137]



III. Με βάση τις νέες συντεταγμένες εκπεφρασμένες ως προς την αρχική εικόνα, θα ορίσουμε τις διαστάσεις της μετασχηματισμένης εικόνας έτσι ώστε να χωρέσουμε τη μετασχηματισμένη εικόνα σε ένα νέο πλαίσιο αναφοράς όπου δεν θα υπάρχουν αρνητικά x και y.

Ουσιαστικά, το νέο πλαίσιο αναφοράς πλέον θα ορίζεται ως εξής:

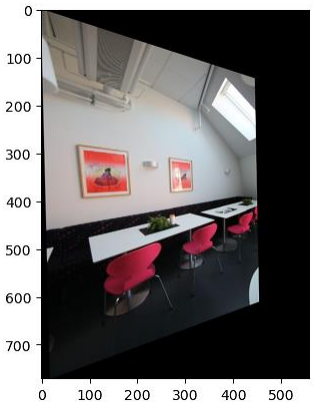


Άρα το νέο πλαίσιο αναφοράς, το ορίζει η γωνία που 'ξεφεύγει περισσότερο' από τις αρχικές συντεταγμένες. Στη περίπτωση της σύνθεσης από αριστερά προς δεξιά το ορίζει η πάνω αριστερή γωνία, στη περίπτωση από δεξιά προς αριστερά κατά πάσα πιθανότητα η πάνω δεξιά ή η κάτω δεξιά γωνία.

Αυτό το πετυχαίνουμε εξετάζοντας τις νέες συντεταγμένες που προέκυψαν (dst) ως προς το ύψος και τα πλάτος.
Συγκεκριμένα, για σύνθεση από αριστερά προς δεξιά, θα έχουμε αρνητικά πλάτη στα αριστερά οπότε θα πρέπει να αυξήσουμε τις διαστάσεις του πλάτους κατά το μεγαλύτερο πλάτος που προκύπτει και αντίστοιχα να προσθέσουμε τα offset των υψών που προεξέχουν. Για τη περίπτωση από δεξιά προς αριστερά απλά αυξάνουμε το πλάτος των αρχικών διαστάσεων καθώς η μετασχηματισμένη εικόνα θα προεξέχει από δεξιά της εικόνας αναφοράς.

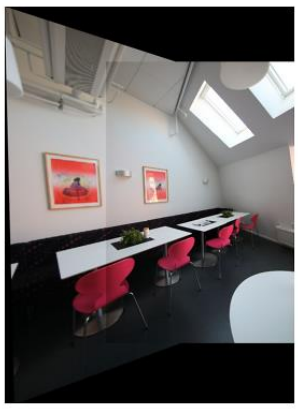
IV. Εφόσον έχουμε ορίσει τις διαστάσεις της μετασχηματισμένης εικόνας (πάνω στην οποία θα χτιστεί το πανόραμα) αλλάζουμε τις συντεταγμένες της ώστε να είναι εκπεφρασμένες στις νέες διαστάσεις. Δηλαδή θέλουμε να ορίσουμε το που θα βρίσκεται η μετασχηματισμένη εικόνα μας στο νέο πλαίσιο αναφοράς.

Ύστερα, λαμβάνουμε ένα νέο πίνακα ομογραφίας για τη μεταφορά των αρχικών σημείων στα νέα σημεία της εικόνας και εκτελούμε τη συνάρτηση warpPerspective() που θα μετασχηματίσει την εικόνα στη τελική μορφή της. Για την συνένωση της πρώτης με τη δεύτερη εικόνα, η μετασχηματισμένη εικόνα φαίνεται παρακάτω:



Όπως φαίνεται και στο σχήμα, οι αρχικές διαστάσεις της εικόνας (547, 364) έχουν γίνει τώρα (771, 559).

V. Η εικόνα που έχει προκύψει είναι η εικόνα A μετασχηματισμένη έτσι ώστε όταν προστεθεί στο πλαίσιο αναφοράς της εικόνας B να προκύψει μια πανοραμική εικόνα. Αυτή είναι η ιδέα και του τελευταίου βήματος όπου ορίζουμε στη μετασχηματισμένη εικόνα τα όρια των pixel στα οποία θα παρεμβάλλουμε την εικόνα B.



Για τη δημιουργία του τελικού πανοράματος συνοψίζουμε όλη τη διαδικασία που περιγράψαμε παραπάνω, στη συνάρτηση Make_Panorama() η οποία παίρνει δύο εικόνες ως εισόδους μαζί με το τρόπο συνένωσης και τη τιμή του threshold για το Low criterion (σε κάποιες περιπτώσεις για τη σταθερή τιμή κατωφλίου 0.8 δεν έμενα πολλά σημεία).

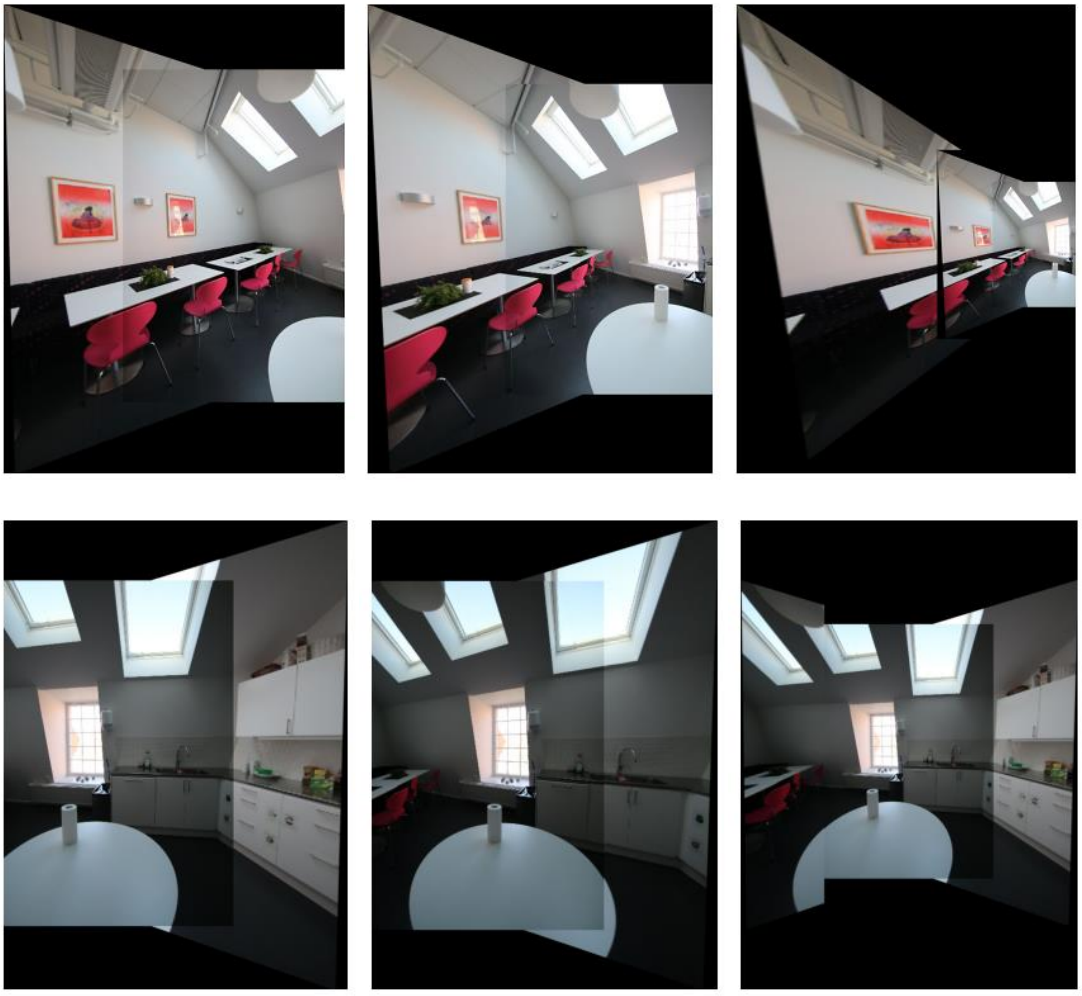
Το τελικό πανόραμα δημιουργήθηκε ως εξής:

- Ένωση εικόνας 1 με εικόνα 2 (πανόραμα 1)
- Ένωση εικόνας 2 με εικόνα 3 (πανόραμα 2)
- Ένωση πανοράματος 1 με πανόραμα 2 (πανόραμα 3)

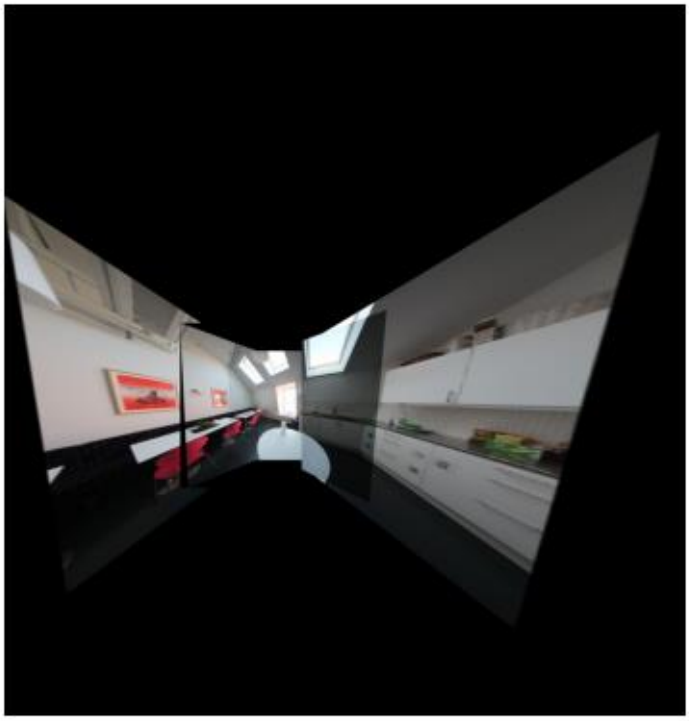
- Ένωση εικόνας 6 με εικόνα 5 (πανόραμα 4)
- Ένωση εικόνας 5 με εικόνα 4 (πανόραμα 5)
- Ένωση πανοράματος 5 με πανόραμα 4 (πανόραμα 6)

- Τελικό πανόραμα: Ένωση πανοράματος 6 με πανόραμα 3

Παρακάτω φαίνονται όλα τα βήματα της δημιουργίας του τελικού πανοράματος



Τελικό Πανόραμα



Κόβοντας τα ενδιάμεσα πανοράματα που προκύπτουν έχουμε: