COMPUTER VISION LAB EXERCISE 3

ΚΩΝΣΤΑΝΤΙΝΟΣ ΚΑΡΑΪΣΚΟΣ
ΑΜ: 1072636

February 11, 2024

Ερώτημα 1

Στην εκφώνηση της άσκησης ο αλγόριθμος SIFT χωρίζεται σε 4 στάδια και έπειτα αναλύονται 5 στάδια. Θα αναφερθούμε στην Εύρεση Τοπικών Σημείων Κλειδιών και στην Ακριβής Εύρεση Τοπικών Σημείων Κλειδιών ως δύο διαφορετικά στάδιο, όπως και αναφέρονται στην ανάλυση των σταδίων.
Τα 5 στάδια του αλγορίθμου SIFT εντοπίζονται στις παρακάτω γραμμές του αρχείου *SIFT_feature.m* :

1. Ανίχνευση Ακροτάτων στο Χώρο Κλίμακας: 20 - 68
2. Εύρεση Τοπικών Σημείων Κλειδιών: 69 - 116
3. Ακριβής Εύρεση Τοπικών Σημείων Κλειδιών: 117 - 189
4. Διόρθωση Κατεύθυνσης: 190 - 260
5. Τοπικός Περιγραφέας Σημείου Κλειδιού: 261 - 344

Ερώτημα 2

Αρχικά διαβάζουμε την εικόνα *cameraman.tiff*, αλλάζουμε το μέγεθός της σε 256x256 και τη μετατρέπουμε σε double.

Ανίχνευση Ακροτάτων στο Χώρο Κλίμακας:
Αρχικοποιούμε τη διασπορά του Gaussian πυρήνα ($sigma0 = sqrt(2)$) και ορίζουμε ότι ο χώρος κλίμακας θα αποτελείται από 3 οκτάβες, με 3 εικόνες ανά οκτάβα.
$D = cell(1, octave)$: δημιουργούμε ένα array άδειων μητρώων ([[],[],[]]). Είναι το array στο οποίο θα αποθηκεύσουμε την πυραμίδα.

$D(i) = mat2cell(zeros(row * 2^(2 - i) + 2, colum * 2^(2 - i) + 2, level), row * 2^(2 - i) + 2, colum * 2^(2 - i) + 2, level)$: δημιουργούμε ένα array μηδενικών σε κάθε κελί του array D, όπου οι διαστάσεις του καθορίζονται από το i. Σε κάθε κελί δημιουργούμε έναν $nxnx3$ array, το οποίο αποτελεί την οκτάβα και τα επίπεδά της.

$temp_img = kron(img, ones(2))$: διπλασιάζουμε την εικόνα σε μέγεθος, δημιουργώντας ουσιαστικά με κάθε pixel μια γειτονία 4pixel από τον εαυτό του.

$temp_img = padarray(temp_img, [1, 1], 'replicate')$: προσθέτει ένα στοιχείο padding σε κάθε διάσταση χρησιμοποιώντας τα ακριανά σημεία.

Κάνουμε plot την αρχική εικόνα και την padded εικόνα.

Δημιουργία πυραμίδας:
Δημιουργούμε την πυραμίδα μέσω δυο for, η εξωτερική δημιουργεί τις οκτάβες και η εσωτερική τα επίπεδα κάθε οκτάβας.
Για κάθε επίπεδο κάθε οκτάβας, αρχικοποιούμε την κλίμακα (scale) και δημιουργούμε ένα 1D Gaussian φίλτρο

με τυπική απόκλιση ίση με scale.
Για να υπολογίσουμε κάθε επίπεδο κάθε οκτάβας, υπολογίζουμε τη συνέλιξη της *temp_img* με το Gaussian φίλτρο, έπειτα τη διαφορά των Gaussian της συνελιγμένης εικόνας με την εικόνα αναφοράς και αποθηκεύουμε την εικόνα στο αντίστοιχο array του προσωρινού πίνακα D.
Αν βρισκόμαστε στο 3ο επίπεδο οποιασδήποτε οκτάβας η τελευταία εικόνα του παρόντος επιπέδου, γίνεται η εικόνα αναφοράς (*temp_img*) του επόμενου επιπέδου, αφού πρώτα αφαιρέσουμε την πρώτη γραμμή των περιφερειακών pixel.
Αφού ολοκληρωθεί μια οκτάβα, αποθηκεύουμε τα επίπεδά της στο i-οστό κελί του πίνακα D, υποδειγματοληπτούμε την εικόνα αναφοράς κατά 2 και έπειτα την κάνουμε padding κατά ένα σημείο σε κάθε μεριά της κάθε διάστασης χρησιμοποιώντας τα ακριανά σημεία.

Εύρεση Τοπικών Σημείων Κλειδιών:

Αρχικοποιούμε τις μεταβλητές που θα χρειαστούμε κατά την εύρεση τοπικών σημείων κλειδιών, την interval που ορίζει σε πιο επίπεδο κάθε οκτάβας θα τελειώσει η αναζήτηση, ενός πίνακα (extrema) στον οποίο θα αποθηκεύουμε τις πληροφορίες των σημείων κλειδιών και τη flag που θα αποτελεί το δείκτη με βάση τον οποίο θα διαπερνάμε τον πίνακα extrema για την αποθήκευση πληροφοριών.
Η αναζήτηση τον τοπικών σημείων κλειδιών γίνεται μέσω 3ων for. Η πρώτη for διαπερνά τις οκτάβες, η δεύτερη τα επίπεδα κάθε οκτάβας (ξεκινώντας από το δεύτερο έως το interval) και η τρίτη τα pixel κάθε επιπέδου. Σε κάθε οκτάβα, οι δύο τελευταίες σειρές και στήλες αγνοούνται λόγω του padding που έχουμε κάνει.
Για κάθε pixel του επιπέδου από το οποίο ξεκινάμε (της εκάστοτε οκτάβας), παίρνουμε μια περιοχή 3x3 από το ίδιο, το προηγούμενο και το επόμενο επίπεδο και εντοπίζουμε την μέγιστη και ελάχιστη τιμή του pixel μεταξύ αυτών των 3ων 3x3 περιοχών. Αν το μεσαίο pixel της 3x3 περιοχής του επιπέδου που ερευνούμε είναι αυτό με την μέγιστη ή την ελάχιστη τιμή, αποθηκεύουμε τις πληροφορίες του στον πίνακα extrema (οκτάβα, επίπεδο, pixel και αν είναι μέγιστο ή ελάχιστο (1, -1 αντίστοιχα)).
Αφού εντοπίσουμε τα τοπικά σημεία κλειδιά, απαλείφουμε από τον πίνακα extrema τις περιττές θέσεις. Υπολογίζουμε τις x, y συντεταγμένες των σημείων κλειδιών κάνοντάς τες scaling ταυτόχρονα και έπειτα τις κανονικοποιούμε ώστε να είναι συμβατές μεταξύ διαφορετικών οκτάβων. Τέλος, κάνουμε plot την εικόνα μας, με τα μαρκαρισμένα τοπικά σημεία κλειδιά.

Ακριβής Εύρεση Τοπικών Σημείων Κλειδιών:

Αρχικοποιούμε το threshold για το φιλτράρισμα των σημείων σε περιοχές ομοιογενούς φωτεινότητας και για τα σημεία που αντιστοιχούν σε ακμές, υπολογίζουμε τον αριθμό των σημείων κλειδιών που βρήκαμε στο προηγούμενο βήμα και δημιουργούμε δύο 2D συνελίξεις για να υπολογίσουμε αργότερα την δεύτερης τάξης παράγωγο στους x, y άξονες.
Με την πρώτη for υπολογίζουμε τη συνάρτηση D για κάθε σημείο κλειδί, για κάθε επίπεδο κάθε οκτάβας. Συγκεκριμένα, υπολογίζουμε τη δεύτερης τάξης παράγωγο για κάθε επίπεδο κάθε οκτάβας και έπειτα ανανεώνουμε το εκάστοτε επίπεδο της πυραμίδας, ενσωματώνοντας πληροφορία από την δεύτερης τάξης παραγώγου ως προς τους άξονες x,y καθώς και της αρχικής πληροφορίας της εικόνας. Αυτό θα μας βοηθήσει στο να φιλτράρουμε μετέπειτα τα σημαντικότερα σημεία κλειδιά.
Με την δεύτερη for, απορρίπτουμε τα σημεία τα οποία ανήκουν σε περιοχές ομοιογενούς ομοιότητας. Για κάθε σημείο κλειδί που εντοπίσαμε (*extr_volume*), υπολογίζουμε τις x, y συντεταγμένες του με βάση τις πληροφορίες που έχουμε αποθηκεύσει στον extrema array, κάνοντάς τες scale με την αντίστοιχη οκτάβα τους. Κάνουμε μία μετατροπή στις συντεταγμένες χρησιμοποιούμε δεικτοδότηση με βάση το 1 και υπολογίζουμε και την z συντεταγμένη, η οποία είναι ουσιαστικά το επίπεδο της οκτάβας που μας ενδιαφέρει. Ανακτούμε την τιμή του pixel που μας ενδιαφέρει από την DoG πυραμίδα και αν είναι μικρότερο κατά απόλυτη τιμή του threshold τα ορίζουμε ως μη ενδιαφέροντα σημεία.
Βρίσκουμε τις θέσεις όπου εντοπίσαμε μη ενδιαφέροντα σημεία και δημιουργούμε ένα μητρώο με ένα μητρώο το οποίο περιέχει διευρυμένα indices για κάθε σημείο. Αυτό το κάνουμε για να εξασφαλίσουμε την εξάλειψη γειτονικών σημείων με το ανεπιθύμητο, τα οποία πολύ πιθανόν να περιέχουν παρόμοια πληροφορία με αυτό. Τέλος, απομακρύνουμε τα ανεπιθύμητα σημεία από τον πίνακα των ακροτάτων και υπολογίζουμε εκ νέου τον αριθμό των ακροτάτων που έχουμε. Υπολογίζουμε τις συντεταγμένες x, y κάνοντάς τες και μετατροπή σε δεικτοδότηση με βάση το 1, προσθέτοντας 1 στο τελικό αποτέλεσμα των υπολογισμών. Έπειτα, υπολογίζουμε τα x,y που αντιστοιχούν στις διαστάσεις της αρχικής εικόνας και προβάλουμε τα σημεία αυτά πάνω στην αρχική εικόνα.
Για κάθε σημείο κλειδί, υπολογίζουμε τις συντεταγμένες x, y που αντιστοιχούν στην πραγματική εικόνα και έπειτα υπολογίζουμε την δεύτερης τάξης παράγωγο στο αντίστοιχο σημείο. Υπολογίζουμε την ορίζουσα, το

ίχνος του Hessian μητρώου και ορίζουμε το threshold. Αν η ορίζουσα είναι αρνητική, δηλαδή η περιοχή που βρισκόμαστε δεν παρουσιάζει κάποια καμπυλότητα, ή το ίχνος του Hessian μητρώου είναι μεγαλύτερο του threshold μαρκάρουμε το pixel ως μη ενδιαφέρον.
Κάνουμε τα ίδια βήματα με πριν για την απομάκρυνση των μη ενδιαφερόντων σημείων, υπολογίζουμε τις συντεταγμένες των τελικών σημείων κλειδιών, με βάση τις διαστάσεις της αρχικής εικόνας μας και τα προβάλουμε πάνω στην αρχική εικόνα.

Διόρθωση Κατεύθυνσης:

Σε αυτό το μέρος θα κάνουμε τον αλγόριθμο να αποκτήσει ανεξαρτησία σε αλλαγές κλίμακας.
Υπολογίζουμε την κλίμακα του εκάστοτε σημείου με βάση το επίπεδο και την οκτάβα στην οποία βρίσκεται, το εύρος της περιοχής γύρω από το σημείο ενδιαφέροντος και τις x,y συντεταγμένες του σημείου στην κλίμακα που βρίσκεται. Έπειτα, ελέγχουμε αν το σημείο κλειδί βρίσκεται εντός ενός έγκυρου εύρους, συγκεκριμένα να μην βρίσκεται πολύ κοντά στα άκρα (πλαίσιο) της εικόνας.
Αν βρίσκεται εντός έγκυρου εύρους, διορθώνουμε το indexing των συντεταγμένων ώστε να ταιριάζει με το indexing με βάση το 1 της Matlab και υπολογίζουμε και το επίπεδο στο οποίο βρίσκεται το pixel. Βρίσκουμε τον όγκο της περιοχής που μας ενδιαφέρει γύρω από το pixel. Δημιουργούμε το κυκλικό Gaussian παράθυρο που θα χρησιμοποιήσουμε για να εξασθενίσουμε τη συνεισφορά των πιο ακριανών pixel από το pixel κλειδί και δημιουργούμε δύο arrays για να αποθηκεύσουμε το μέτρο κλίσης και την κλίση των pixel της περιοχής. Έπειτα, υπολογίζουμε το μέτρο κλίσης και την κλίση των pixel της περιοχής με δύο for, μέσω των τοπικών παραγώγων. Η εξωτερική for διαπερνά τις σειρές της περιοχής που έχουμε ορίσει γύρω από το pixel κλειδί και η εσωτερική της στήλες αυτής της περιοχής. Για τον υπολογισμό του μέτρου κλίσης, υπολογίζουμε την Ευκλείδεια απόσταση και για τον υπολογισμό της κλίσης, τη γωνία της εφαπτομένης (σε μοίρες) μεταξύ των δυο γειτονικών pixel στους άξονες x, y του σημείου κλειδιού. Δημιουργούμε ένα διάνυσμα 1x36 το οποίο αναπαριστά τα bin του ιστογράμματος προσανατολισμού. Έπειτα, για να δημιουργήσουμε το ιστόγραμμα, χρησιμοποιούμε δύο for, όπου η εξωτερική αναφέρεται σε κάθε bin και η εσωτερική σε pixel της κυκλικής περιοχής που ορίσαμε. Αρχικοποιούμε τις μεταβλητές που ορίζουν τα όρια του τωρινού bin. Εάν η κλίση του pixel που εξετάζεται βρίσκεται μέσα στο επιθυμητό εύρος, προσθέτουμε το μέτρο κλίσης του στο bin αφού πρώτα εφαρμόσουμε το Gaussian παράθυρο που δημιουργήσαμε πριν. Τέλος, αποθηκεύουμε στο διάνυσμα *mag_counts* τον αριθμό το μέτρο κλίσης του bin που δημιουργήσαμε.
Βρίσκουμε το bin με τη μέγιστη τιμή στο ιστόγραμμα και έπειτα δημιουργούμε το διάνυσμα kori, στο οποίο αποθηκεύουμε τα indices των σημείων με κλίση τουλάχιστον 80% του μέγιστου. Μετατρέπουμε τα indices του kori σε τιμές γωνιών και τις αποθηκεύουμε στο διάνυσμα kpori. Αποθηκεύουμε της πληροφορίες του pixel σε ένα προσωρινό πίνακα (*temp_extrema*) τον οποίο κάνουμε pad κυκλικά για να εξασφαλίσουμε ότι έχει μήκος ίσο με το μήκος του kori και τέλος αποθηκεύουμε το padded array στο διάνυσμα minor. Διαγράφουμε από το διάνυσμα minor τις μηδενικές τιμές και ανανεώνουμε τον πίνακα extrema καθώς και το μέγεθος τον όγκο των σημείων κλειδιών.

Τοπικός Περιγραφέας Σημείου Κλειδιού:

Στο τελευταίο βήμα του αλγορίθμου δημιουργούμε τους τοπικούς descriptors για κάθε σημείο κλειδί.
Αρχικοποιούμε τις μεταβλητές της περιοχής που θα εξετάσουμε γύρω από τα σημεία κλειδιά και δημιουργούμε ένα μητρώο για να αποθηκεύσουμε τους descriptors για όλα τα σημεία κλειδιά. Για κάθε σημείο κλειδί, δημιουργούμε αρχικά ένα διάνυσμα 1x128 στο οποίο θα αποθηκεύσουμε προσωρινά τον descriptor του σημείου, υπολογίζουμε το μέγεθος της περιοχής γύρο από το σημείο και βρίσκουμε τις συντεταγμένες του (μετατρέποντάς τες σε 1-base indexing). Υπολογίζουμε τους δείκτες x, y των pixel μίας περιοχής γύρω από το σημείο κλειδί με διαστάσεις 8x8, υπολογίζουμε και αποθηκεύουμε τις τιμές των συντεταγμένων κάθε σημείου της περιοχής στο μητρώο sub. Για την περιοχή αυτή, βρίσκουμε το 2x2 μητρώο περιστροφής της. Περιστρέφουμε την περιοχή γύρω από το σημείο κλειδί με βάση το μητρώο περιστροφής. Υπολογίζουμε τις ακέραιες συντεταγμένες των περιστρεμμένων σημείων ώστε να μπορούν να αντιστοιχηθούν στην εικόνα και τις προσαρμόζουμε από συντεταγμένες της εικόνας που βρίσκονταν, σε συντεταγμένες της αρχικής εικόνας. Αποθηκεύουμε σε ένα διάνυσμα (patch), τις τιμές των περιστρεμμένων pixels της περιοχής, από την εικόνα του επιπέδου τις οκτάβας στην οποία ανήκε στο σημείο κλειδί, το οποίο μετατρέπουμε σε ένα 2D μητρώο με διαστάσεις 16x16. Στο επεξεργασμένο παράθυρο εφαρμόζουμε έναν Gaussian πυρήνα για να βρούμε το μέτρο κλίσης κάθε σημείου του παραθύρου, δίνοντας έμφαση στα κεντρικά σημεία και υπολογίζουμε την κλίση τους. Στο τελευταίο βήμα, μας έχει μείνει να υπολογίσουμε τον descriptor για το σημείο κλειδί. Για κάθε υποπεριοχή της τοπικής περιοχής, υπολογίζουμε των αριθμό των pixel που την συντελούν, δημιουργούμε ένα array για να αποθηκεύσουμε τις κατευθύνσεις του και ένα array για τα bins του ιστογράμματος προσανατολισμών. Για κάθε ένα από τα 8 bins, και για κάθε pixel της υποπεριοχής, υπολογίζουμε τις συντεταγμένες του και αν η

κλίση του βρίσκεται εντός συγκεκριμένων μοιρών, αποθηκεύουμε ως κλίση αυτού του pixel τη μέση κλίση του bin και συσσωρεύουμε και το μέτρο κλίσης του. Αφού ερευνήσουμε όλα τα pixel κάθε υποπεριοχής, δημιουργούμε τον descriptor αποθηκεύοντας τα μέτρα κλίσης που βρήκαμε.
Κανονικοποιούμε τον descriptor, ώστε οι τιμές του να αθροίζουν στο 1, περιορίζουμε τις τιμές των στοιχείων του ώστε να μετριάσουμε τυχών outliers, έπειτα κανονικοποιούμε ξανά τον descriptor και αποθηκεύουμε τον τελικό descriptor στο μητρώο που δημιουργήσαμε (feature) το οποίο θα περιέχει όλους τους descriptors. Τέλος, από το σύνολο των σημείων κλειδιών, κρατάμε μόνο αυτά που δεν έχουν ουδέτερο προσανατολισμό, δηλαδή το άθροισμα των μέτρων των στοιχείων του descriptor τους διαφέρει αρκετά από το 0.

Ερώτημα 3

Όπως βλέπουμε από τις γραμμές 25 και 26 του κώδικα, χρησιμοποιούμε 3 οκτάβες με 3 επίπεδα ανά οκτάβα.

Ερώτημα 4

Εναλλακτικοί τρόποι για να εντοπίσουμε της βέλτιστη λύση ενός σημείου κλειδιού:

a) **Harris Corner Detector:**

- Πλεονεκτήματα: ανοσία σε περιστροφές της εικόνας, ανθεκτικός σε φωτομετρικές παραμορφώσεις, μικρό υπολογιστικό κόστος, προσαρμογή στις απαιτήσεις ανά περίπτωση (parameter tuning).
- Μειονεκτήματα: ευαισθησία στο θόρυβο (gradient-based μέθοδος), ευαισθησία στις αλλαγές κλίμακας, αδυναμία στον εντοπισμό περιγραμμάτων ή κηλίδων, απαιτητική σωστή παραμετροποίηση (threshold).

b) **FAST (Features from Accelerated Segment Test):**

- Πλεονεκτήματα: ταχύτητα, ανοσία στις περιστροφές τις εικόνας, χαμηλή υπολογιστική πολυπλοκότητα, μόνο μία παράμετρος προς προσαρμογή (threshold for corner detection).
- Μειονεκτήματα: ευαισθησία στο θόρυβο, περιορισμός κυρίως στον εντοπισμό γωνιών, ευαισθησία στις αλλαγές κλίμακας της εικόνας, εντοπισμών και μη βέλτιστων σημείων κλειδιών, απαιτητική σωστή παραμετροποίηση (threshold).

c) **ORB (Oriented FAST and Rotated BRIEF):**

- Πλεονεκτήματα: γρήγορος και αποδοτικός (συνήθως πιο γρήγορος από τους SIFT και SURF), ανοσία στις περιστροφές και στις αλλαγές κλίμακας, χρήση descriptor που παράγει ο BRIEF που είναι πιο γρήγορος στον υπολογισμό, ανθεκτικός στις φωτομετρικές παραμορφώσεις
- Μειονεκτήματα: μειωμένη ακρίβεια ταυτοποίησης σημείων λόγω περιορισμένης διακριτοποίησης του descriptor, ευαισθησία σε γεωμετρικές παραμορφώσεις, ευαισθησία στο θόρυβο, απαιτητική παραμετροποίηση.

d) **SURF (Speed Up Robust Features):**

- Πλεονεκτήματα: υπολογιστικά αποδοτικός, ανοσία στις περιστροφές και στις αλλαγές κλίμακας τη εικόνας, υπολογιστικά αποδοτικός descriptor, ανθεκτικός στο θόρυβο και στις φωτομετρικές παραμορφώσεις.
- Μειονεκτήματα: ευαισθησία στις περιστροφές της εικόνας, σχετικά μεγάλο μέγεθος descriptor, ευαισθησία σε affine παραμορφώσεις, απαιτητική παραμετροποίηση.

Ερώτημα 5

Τρέχουμε τον SIFT για τις παρακάτω 5 εικόνες.
Εικόνα totem.tiff. Αποτελέσματα:

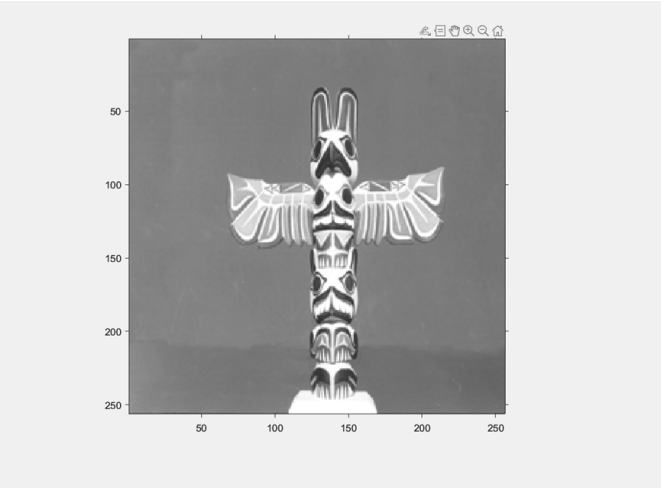

Figure 1: Totem image

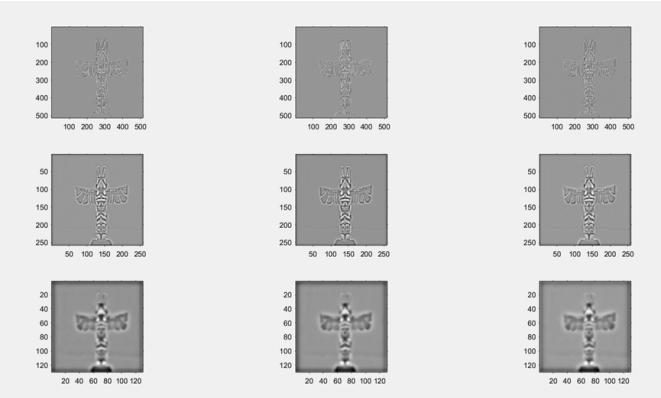

Figure 2: Pyramids of the image

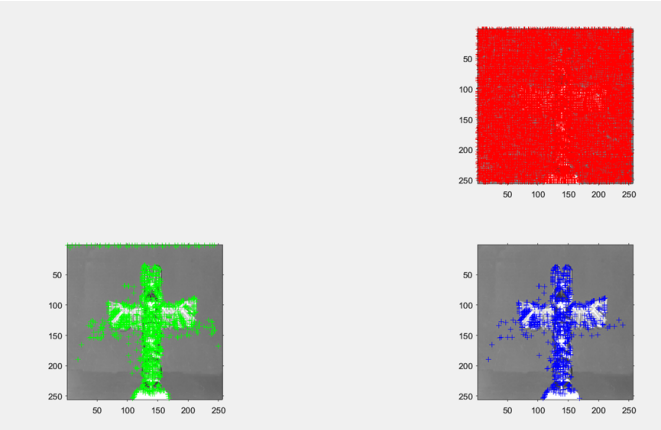

Figure 3: Detected key points

Εικόνα x31_f18.tif. Αποτελέσματα:

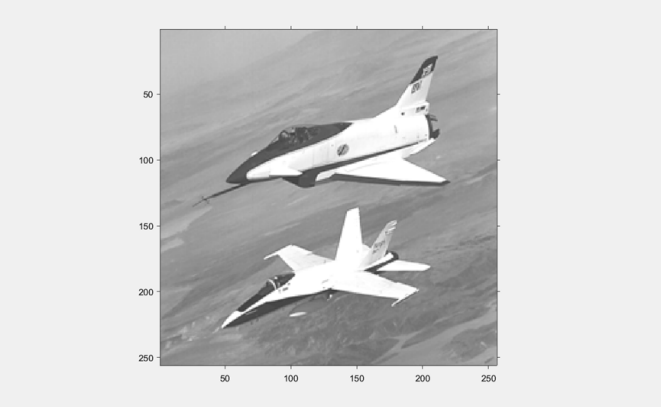

Figure 4: x31 f18 image

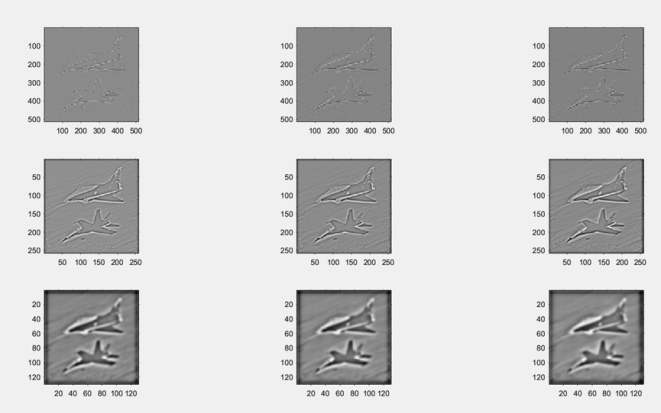

Figure 5: Pyramids of the image

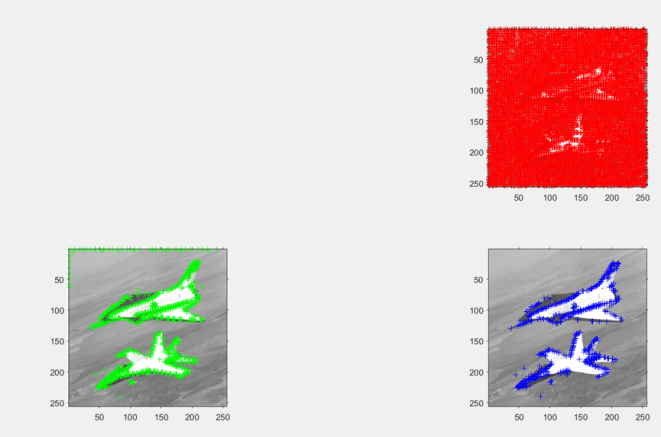

Figure 6: Detected key points

Εικόνα leaves.tif. Αποτελέσματα:

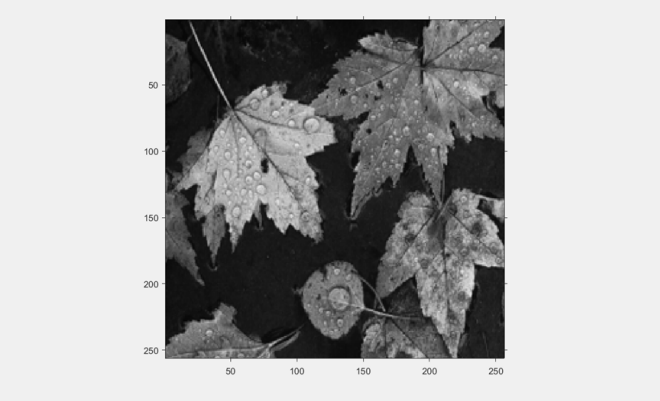

Figure 7: Leaves image

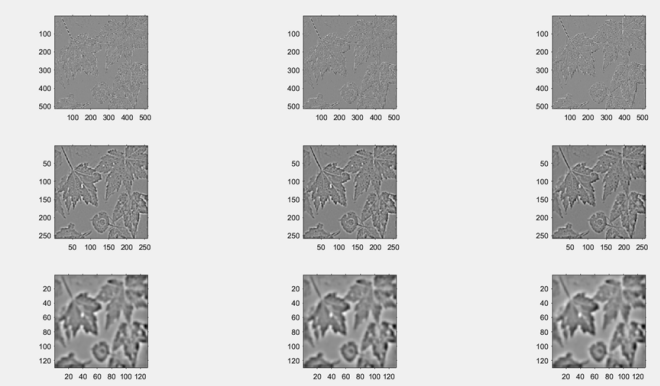

Figure 8: Pyramids of the image

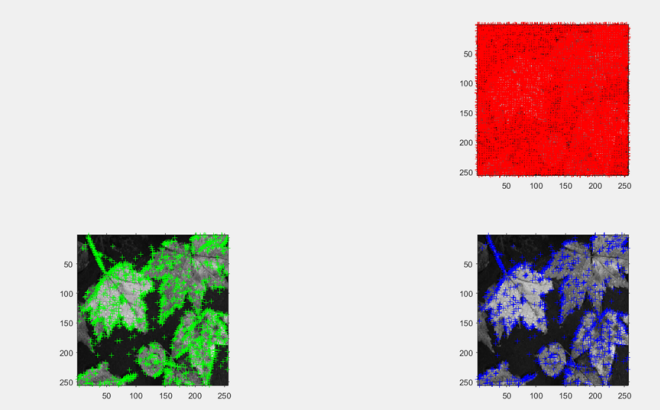

Figure 9: Detected key points

Εικόνα columns.tif. Αποτελέσματα:

Figure 10: Columns image

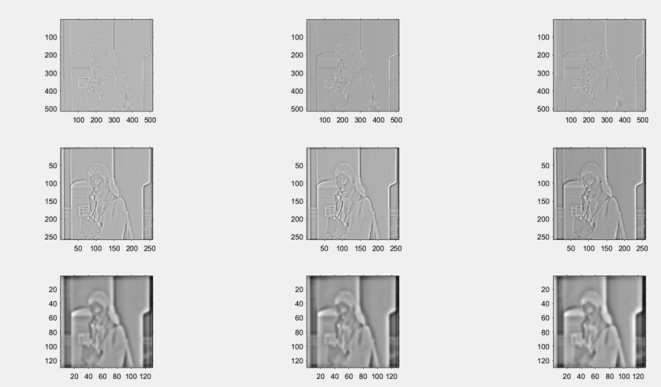

Figure 11: Pyramids of the image

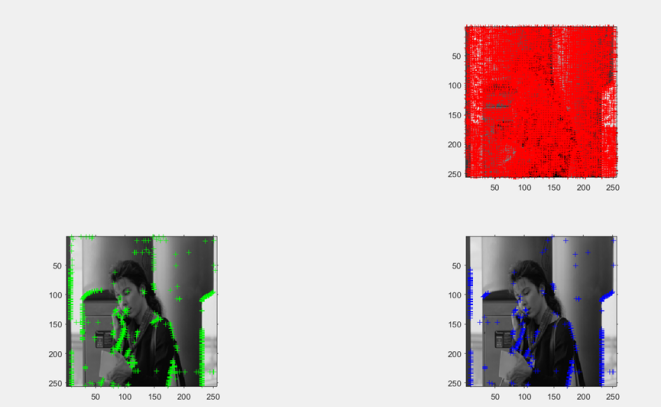

Figure 12: Detected key points

Εικόνα autumn.tif. Αποτελέσματα:

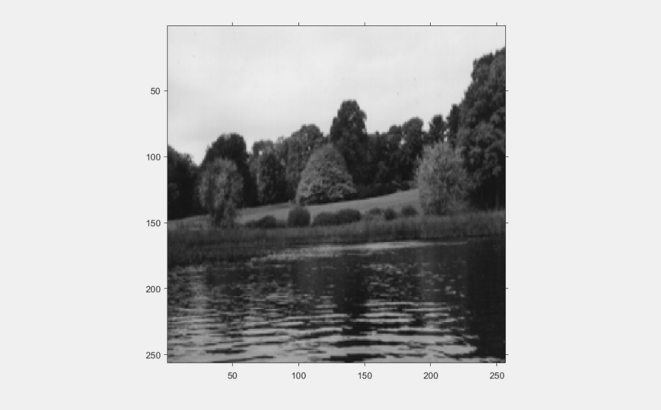

Figure 13: Autumn image

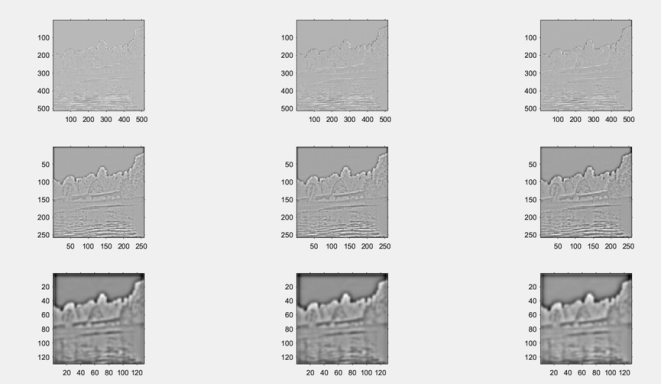

Figure 14: Pyramids of the image

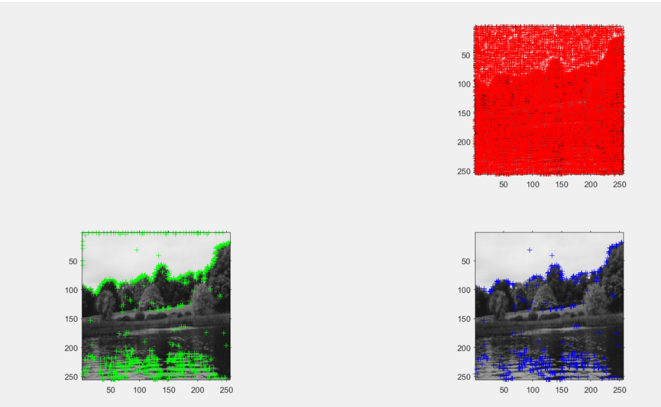

Figure 15: Detected key points

Όπως είναι αναμενόμενο, στην πρώτη οκτάβα, μπορούμε να διακρίνουμε με πολύ καλή ευκρίνεια πάρα πολλές μικρές λεπτομέρειες πάνω στην εικόνα και όσο ανεβαίνουμε οκτάβες, οι λεπτομέρειες γίνονται όλο και πιο δυσδιάκριτες, έως την τελευταία οκτάβα στην οποία μπορούμε να διακρίνουμε σχεδόν μόνο, τα περιγράμματα των αντικειμένων της αρχικής εικόνας. Όσο ανεβαίνουμε δηλαδή οκτάβες, δίνεται μεγαλύτερη έμφαση στα σημεία τα οποία εμφανίζουν μεγαλύτερη αντίθεση ως προς τη φωτεινότητα με το περιβάλλον τους, τα οποία αναμένουμε να εντοπιστούν και ως τελικά σημεία κλειδιά της αρχικής εικόνας μας.
Στην αρχική ανίχνευση τοπικών ακροτάτων, παρατηρούμε ότι εντοπίζουμε πάρα πολύ μεγάλο αριθμό τοπικών ακροτάτων, το οποίο φαίνεται οπτικοποιώντας τα τοπικά αυτά ακρότατα με κόκκινο χρώμα πάνω στην εικόνα. Αυτό συμβαίνει γιατί σε αρχική φάση ελέγχουμε για διαφορές σε φωτεινότητα μεταξύ των pixel σε πολύ μικρές γειτονιές και αφού παρά την οπτική ομοιογένεια των pixel σε πολλές περιοχές υπάρχουν μικρές διαφορές στη φωτεινότητα οι οποίες μεταφράζονται ως ακρότατα. Μετά από δύο είδους φιλτραρίσματα, ένα ομοιογένειας και ένα ακμών, παρατηρούμε ότι μετά το δεύτερο μας μένουν τα περισσότερα από τα σημεία κλειδιά τα οποία αναμέναμε να εντοπίσουμε παρατηρώντας την τελευταία οκτάβα της πυραμίδας μας. Αυτό συμβαίνει καθώς απορρίψαμε όσα σημεία παρουσιάζουν πολύ μικρές διαφορές μεταξύ των γειτονικών τους και όσα σημεία αντιστοιχούν σε ακμές καθώς από αυτά δεν μπορούμε να βγάλουμε ουσιαστικά συμπεράσματα σε προβλήματα αντιστοίχησης σημείων που βασίζονται σε χαρακτηριστικά.
Γενικά, παρατηρούμε ότι ο αλγόριθμος έχει πολύ καλή απόδοση όταν δεν υπάρχει πολύ μεγάλη ομοιογένεια στη φωτεινότητα μεταξύ των αντικειμένων της εικόνας, καθώς και όταν η εικόνα δεν αποτελείται κυρίως από ακμές.

Ερώτημα 6

```
% Code for matching keypoints from their descriptors computed from SIFT

% Create a tilted cameraman image
img=imread('cameraman.tif');
theta=1;
A=[cosd(theta) -sind(theta) 0;
sind(theta) cosd(theta) 0;
0 0 1;];
tform=affine2d(A');
[tilted_img]=imwarp(img,tform,'Interp','Cubic','FillValues',0);

imwrite(tilted_img,'tilted_cameraman.tif','tif');

% Compute the descriptors for each image using SIFT
[imgFeatures,imgExtremas,imgOctaves]=SIFT_algorithm('cameraman.tif');
[tiltedFeatures,tiltedExtremas,tiltedOctaves]=SIFT_algorithm('tilted_cameraman.tif');

cameramanPairs=matchFeatures(imgFeatures',tiltedFeatures');

imgExtremas=imgExtremas';
tiltedExtremas=tiltedExtremas';

[m, n] = size(img);

% Use the computed pairs indexes to calculate the coordinates
% of the paired features from the two images
xImg=floor((imgExtremas(4*cameramanPairs(:,1)-1,:)-1)./(n./(2.^(imgExtremas(4*cameramanPai
yImg=mod((imgExtremas(4*cameramanPairs(:,1)-1,:)-1),m./(2.^(imgExtremas(4*cameramanPairs(:
rxImg=xImg./2.^(imgOctaves-1-imgExtremas(4*cameramanPairs(:,1)-3,:));
ryImg=yImg./2.^(imgOctaves-1-imgExtremas(4*cameramanPairs(:,1)-3,:));

imshow(img)
hold on
plot(ryImg,rxImg,'r+');
```

```
xTilted=floor((tiltedExtremas(4*cameramanPairs(:,1)-1,:)-1)./(n./(2.^(tiltedExtremas(4*cam
yTilted=mod((tiltedExtremas(4*cameramanPairs(:,1)-1,:)-1),m./(2.^(tiltedExtremas(4*cameram
rxTilted=xImg./2.^(tiltedOctaves-1-tiltedExtremas(4*cameramanPairs(:,1)-3,:));
ryTilted=yImg./2.^(tiltedOctaves-1-tiltedExtremas(4*cameramanPairs(:,1)-3,:));

imshow(tilted_img)
hold on
plot(ryTilted,rxTilted,'g+');

matchedImgPoints=[ryImg rxImg];
matchedTiltedPoints=[ryTilted rxTilted];

% Plot the matched features on the two images
figure;
showMatchedFeatures(img,tilted_img,matchedImgPoints,matchedTiltedPoints,'montage');
```

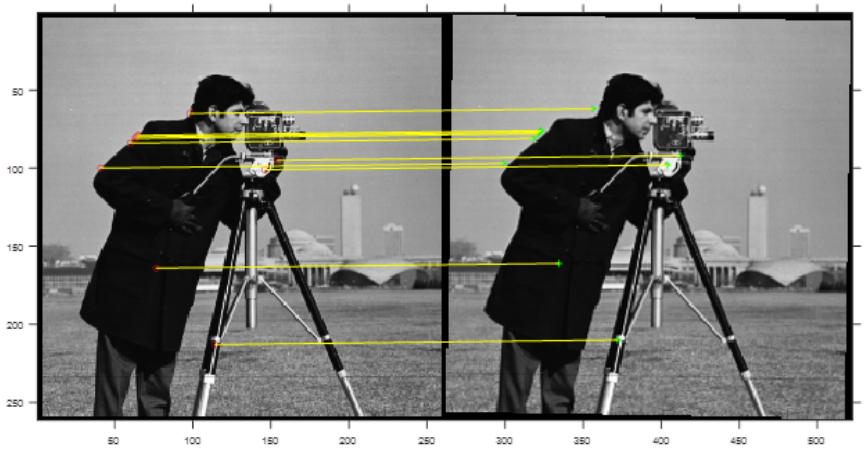

Figure 16: Cameraman matched key points

Ερώτημα 7

Ο RANSAC (Random Sample Consensus) είναι ένας επαναληπτικός αλγόριθμος που χρησιμοποιείται για την εκτίμηση ενός μοντέλου και την αντιστοίχιση σημείων, παρουσία outliers. Τα βήματα του αλγορίθμου, είναι τα παρακάτω:

1. Επιλογή ενός τυχαίου αριθμού δειγμάτων από το αρχικό σύνολο δειγμάτων και εκτίμηση ενός μοντέλου, χρησιμοποιώντας αυτά τα δείγματα.
2. Εκτίμηση όλων των δεδομένων σε σχέση με αυτό το μοντέλο (με βάση κάποια συνάρτηση απώλειας) και χαρακτηρισμός των σημείων που ταιριάζουν καλά στο μοντέλο ως εσωτερικά και των υπολοίπων ως εξωτερικά.
3. Υπολογισμός των εσωτερικών σημείων.
4. Επανάληψη των βημάτων 1-3 για καθορισμένο αριθμό επαναλήψεων ή μέχρι η εκτίμηση του μοντέλου να φτάσει κάποια τιμή.
5. Επιλογή του μοντέλου με τον μεγαλύτερο αριθμό εσωτερικών σημείων.
6. Προαιρετικά, επανεκτίμηση του μοντέλου χρησιμοποιώντας τα εσωτερικά σημεία του καλύτερου μοντέλου.

Προκειμένου να αντιστοιχίσουμε σημεία κλειδιά που βρέθηκαν στην αρχική εικόνα *cameraman.tif* και *tilted_cameraman.tif*, χρησιμοποιήσαμε την συνάρτηση *matchFeatures*() η οποία δεν ακολουθεί την λογική του αλγορίθμου RANSAC.
Η συνάρτηση *matchFeatures*(), υπολογίζει τις αποστάσεις μεταξύ των περιγραφέων, με βάση μία μετρική, των σημείων που εντοπίσαμε στις εικόνες, και επιστρέφει τα ζευγάρια των indexes των περιγραφέων των σημείων που εντόπισε ως πολύ όμοια.

Ερώτημα 8

Ο αλγόριθμος GLOH (Gradient Location and Orientation Histogram) ακολουθεί τα ίδια βήματα με τον αλγόριθμο SIFT (Scale Invariant Feature Transform) για τον εντοπισμό των τοπικών σημείων κλειδιών, διαφοροποιείται στη δημιουργία τον περιγραφέων των εντοπισμένων χαρακτηριστικών. Συγκεκριμένα, ο GLOH υπολογίζει το σε ιστόγραμμα σε log-polar πλέγμα, αντί για ορθογώνιο που χρησιμοποιεί ο SIFT και το κάθε ιστόγραμμά του χωρίζεται σε 16 bins αντί για 8 bins, που χρησιμοποιεί ο SIFT. Καταλήγουμε επομένως να δημιουργούμε ένα διάνυσμα περιγραφέα μεγέθους 1x276, το οποίο φέρνουμε με PCA στο ίδιο μέγεθος με τους περιγραφείς που παράγονται από τον SIFT.
Η δημιουργία του χώρου κλίμακας στον SURF αντί να γίνεται με σταθερού μεγέθους φίλτρο και διαδοχικά μειωμένου μεγέθους εικόνας όπως γίνεται στον SIFT, χρησιμοποιεί την αρχική εικόνα και εφαρμόζει σε αυτή διαφορετικά φίλτρα που αντιστοιχούν σε διαφορετικές κλίμακας. Για τον εντοπισμό της θέσης και της κλίμακας των χαρακτηριστικών ο SURF (Speeded Up Robustness Features), βασίζεται στην χρήση της ορίζουσας του Hessian μητρώου και όχι στην DoG που χρησιμοποιεί ο SIFT και ο GLOH, αλλά ακολουθεί την ίδια σύγκριση τιμών με τα οκτώ γειτονικά pixel του σημείου στο ίδιο επίπεδο και με τα 9 γειτονικά στο πάνω και κάτω από αυτό επίπεδο. Οι συνελίξεις της εικόνας με τα ορθογώνια φίλτρα που χρησιμοποιούνται, υπολογίζονται με χρήση της ολοκληρωτικής εικόνας, τεχνική που απαιτεί σταθερό αριθμό πράξεων άρα και μειωμένο υπολογιστικό κόστος.
Στην ανάθεση προσανατολισμού στα χαρακτηριστικά, ο SURF χρησιμοποιεί τα κυματίδια Haar. Συγκεκριμένα, υπολογίζει τις αποκρίσεις των κυματιδίων εντός μίας γειτονιάς με κέντρο το χαρακτηριστικό και έπειτα εφαρμόζεται ένας Gaussian πυρήνας για εξομάλυνση. Στην φιλτραρισμένη απόκριση, εφαρμόζουμε τέλος ένα παράθυρο το οποίο μας επιστρέφει τον κυρίαρχο προσανατολισμό του χαρακτηριστικού. Βασίζεται για την ανάθεση προσανατολισμού δηλαδή, στις τοπικές μεταβολές της φωτεινότητας της εικόνας, σε αντίθεση με τον SIFT και τον GLOH, που βασίζονται σε ιστογράμματα προσανατολισμών.
Τέλος, για τη δημιουργία περιγραφέων, δημιουργείται μία περιοχή 4x4 γύρω από το χαρακτηριστικό που περιστρέφεται με βάση τον προσανατολισμό του. Για κάθε μία από τις 16 υποπεριοχές, υπολογίζονται 4 ποσότητες που αντιστοιχούν σε αθροίσματα αποκρίσεων των κυματιδίων Haar στους x,y άξονες. Άρα, παράγεται ένας περιγραφέας μήκους 64, σε αντίθεση με τους SIFT και GLOH που το μήκος του περιγραφέα τους είναι 128.

Ερώτημα 9

Πολυπλοκότητα χειρότερης περίπτωσης για τον:

- SIFT: O(M*N+I), όπου MxN οι διαστάσεις της εικόνας και I ο αριθμός των σημείων κλειδιών.
- GLOH: O(M*N+I), όπως και ο SIFT καθώς το βήμα υπολογισμού του ιστογράμματος προσθέτει σταθερό κόστος στον αλγόριθμο.
- SURF: O(M*N+k), όπου k ο αριθμός των σημείων κλειδιών.