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

2.

Αρχικά διαβάζουμε την εικόνα cameraman.tif, 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.