ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΟΡΑΣΗ ΥΠΟΛΟΓΙΣΤΩΝ

(2020-2021)

1ο Εργαστηριακό Project

Θέμα: Εντοπισμός Σημείων Ενδιαφέροντος και Εξαγωγή Χαρακτηριστικών σε Εικόνες

Ονοματεπώνυμο:

- Χρήστος Τσούφης

Αριθμός Μητρώου:

- 03117176

Στοιχεία Επικοινωνίας:

- el17176@mail.ntua.gr

Στόχος της Εργασίας

Το project αποτελείται από 3 μέρη. Στο Μέρος 1, δημιουργούνται οι εικόνες εισόδου (1.1), υλοποιούνται Αλγόριθμοι Ανίχνευσης Ακμών σε τεχνητές (1.2) και πραγματικές εικόνες (1.4) και αξιολογούνται τα αποτελέσματα (1.3). Στο Μέρος 2, μελετάται η Ανίχνευση Σημείων Ενδιαφέροντος (Interest Point Detection) σε εικόνες. Συγκεκριμένα, μελετώνται οι περιπτώσεις: Ανίχνευση Γωνιών (2.1) & Πολυκλιμακωτή Ανίχνευση Γωνιών (2.2), Ανίχνευση Blobs (2.3) & Πολυκλιμακωτή Ανίχνευση Blobs (2.4) και Επιτάχυνση με την χρήση Box Filters Ολοκληρωτικών Εικόνων (Integral Images) (2.5). Τέλος, το Μέρος 3 πραγματεύεται Εφαρμογές σε Ταίριασμα και Κατηγοριοποίηση Εικόνων με Χρήση Τοπικών Περιγραφητών στα Σημεία Ενδιαφέροντος. Ειδικότερα, μελετώνται οι περιπτώσεις: Ταίριασμα Εικόνων υπό Περιστροφή και Αλλαγή Κλίμακας (3.1) και Κατηγοριοποίηση Εικόνων (3.2).

Τεχνολογίες & Τρόπος Εκτέλεσης εφαρμογής

Η παρούσα εργασία υλοποιήθηκε σύμφωνα με το αρχείο "cv21_lab0" το οποίο δημιουργεί ένα Python περιβάλλον με όνομα "cv_lab1_env" και χρησιμοποιεί Python 3.7 με χρήση των βιβλιοθηκών numpy, scipy, scikit-learn, tqdm, matplotlib και OpenCV 3.4.2. Το σετάρισμα της εφαρμογής έγινε σε απομονωμένο περιβάλλον conda.

Το project έχει την εξής δομή: Αποτελείται από 3 ξεχωριστά αρχεία, ένα για κάθε μέρος της εργασίας, με αντίστοιχα ονόματα. Επίσης, περιλαμβάνει ξεχωριστά, το αρχείο utils.py το οποίο περιέχει όλες τις συναρτήσεις που χρησιμοποιήθηκαν, το αρχείο EdgeDetect.py που χρησιμοποιείται στο Μέρος 1 και το αρχείο bovw.py που χρησιμοποιείται για το Μέρος 3.

Μέρος 1: Ανίχνευση Ακμών σε Γκρίζες Εικόνες

1.1 *Δημιουργία Εικόνων Εισόδου*

1.1.1 *Πρώτα, η αρχική εικόνα I_0 = "edgetest_10.png" που βρίσκεται στο επιπρόσθετο υλικό στο moodle, διαβάζεται στο αντίστοιχο κομμάτι του κώδικα.*

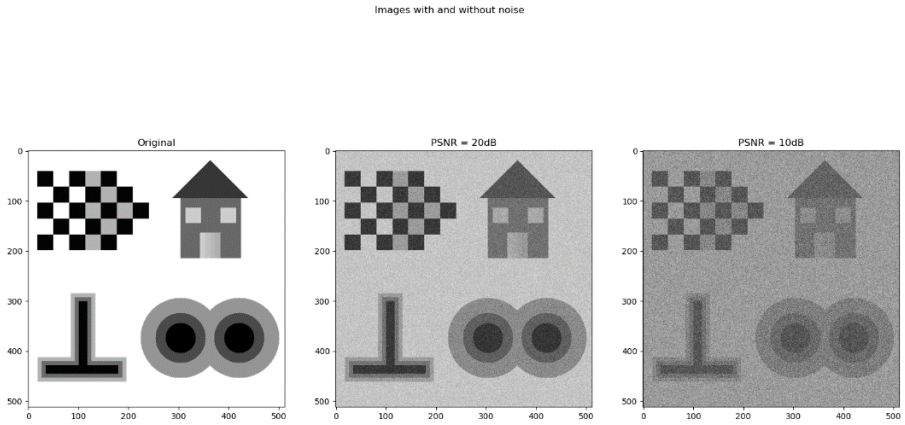
1.1.2 *Έπειτα, γίνεται εισαγωγή προσθετικού θορύβου στις εικόνες για την δημιουργία θορυβωδών εικόνων $I(x,y) = I_0(x,y) + n(x,y)$ που θα χρησιμοποιηθούν σαν είσοδοι στον αλγόριθμο ανίχνευσης ακμών. Ο θόρυβος $n(x, y)$ είναι White Gaussian, με μέση τιμή 0 και τυπική απόκλιση σ_n τέτοια ώστε ο PSNR να έχει συγκεκριμένες τιμές. Ζητείται μάλιστα, η χρήση 2 διαφορετικών τιμών για τον PSNR του θορύβου:*

i) PSNR = 20 dB, ii) PSNR = 10 dB

Ο PSNR ορίζεται ως εξής: $PSNR = 20log_{10}(\frac{I_{max}-I_{min}}{\sigma_n})$ *(dB)*

$$I_{max} = \max_{x,y} I(x,y),\ I_{min} = \min_{x,y} I(x,y)$$

Η παραπάνω διαδικασία υλοποιείται με τη συνάρτηση *imnoise* στο *utils.py* .



1.2 *Υλοποίηση Αλγορίθμων Ανίχνευσης Ακμών*

Ζητείται η κατασκευή μιας συνάρτησης EdgeDetect η οποία υλοποιεί τον αλγόριθμο ανίχνευσης ακμών που περιγράφεται παρακάτω.

1.2.1 *Σε αυτό το ερώτημα ζητείται η δημιουργία κρουστικών αποκρίσεων δύο διακριτών γραμμικών φίλτρων που προσεγγίζουν τα συνεχή φίλτρα με τις εξής κρουστικές αποκρίσεις:*

i) Διδιάστατη Gaussian $G_\sigma(x,y)$
ii) Laplacian-of-Gaussian (LoG) $h(x,y) = \nabla^2 G_\sigma(x,y)$

Σημειώνεται ότι και στις δύο περιπτώσεις χρησιμοποιείται η τυπική απόκλιση (χωρική κλίμακα) σ για την Gaussian και οι πυρήνες έχουν έκταση n x n pixels με n = ceil(3·σ) ·2 + 1.

Συγκεκριμένα, ζητείται η δημιουργία μιας συνάρτησης που να υλοποιεί τα παραπάνω φίλτρα.

Η παραπάνω διαδικασία υλοποιείται με τη συνάρτηση *EdgeDetect* στο αρχείο *EdgeDetect.py* .

1.2.2 *Εδώ περιγράφεται η προσέγγιση της Laplacian L της εξομαλυμένης εικόνας $I_\sigma(x,y) = G_\sigma * I(x,y)$ που ζητείται με 2 διαφορετικές εναλλακτικές:*

i) Γραμμική (L_1): συνέλιξη της I με την LoG h:
$L_1 = \nabla^2(G_\sigma * I) = (\nabla^2 G_\sigma) * I = h * I$
ii) Μη-Γραμμική (L_2): μη-γραμμική εκτίμηση της Laplacian της I_σ με μορφολογικά φίλτρα:
$L_2 = I_\sigma \oplus B + I_\sigma \ominus B - 2I_\sigma$

Η 1η μέθοδος αφορά φιλτράρισμα με την LoG που δίνεται ως η Λαπλασιανή μιας ισοτροπικής Gaussian $G_\sigma(x,y)$:

$$h(x,y) = \nabla^2 G_\sigma(x,y) = \frac{\partial^2 G_\sigma}{\partial x^2} + \frac{\partial^2 G_\sigma}{\partial y^2} = \frac{x^2+y^2-2\sigma^2}{\sigma^4}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

Έστω επίσης, ο πυρήνας του φίλτρου h διαστάσεων n x n όπου $n = \lceil 3 \cdot \sigma \rceil \cdot 2 + 1$.

Οπότε προκύπτει η εικόνα: $L_1(x,y) = h(x,y) * I(x,y)$

Η 2η μέθοδος αφορά φιλτράρισμα με την LoG με μορφολογικά φίλτρα που μπορεί να προσεγγιστεί ως εξής:

$$L_2 = I_\sigma \oplus B + I_\sigma \ominus B - 2I_\sigma$$

Όπου $I_\sigma \oplus B = \bigcup_{b\in B} I_\sigma + B$ $I_\sigma \ominus B = \bigcap_{b\in B} I_\sigma - B$ $I_\sigma = I * G_\sigma$

Και B, είναι το μορφολογικό φίλτρο np.array([[0, 1, 0], [1, 1, 1], [0, 1, 0]], dtype=np.uint8).

Τα δομικά στοιχεία που χρησιμοποιούνται είναι δυαδικές εικόνες δίσκων διαμέτρου n_s.

Η παραπάνω διαδικασία υλοποιείται με τη συνάρτηση *EdgeDetect* στο αρχείο *EdgeDetect.py* .

1.2.3 *Έπειτα, ζητείται η προσέγγιση των zero-crossings (σημεία μηδενισμού) της L. Ένας πιθανός αλγόριθμος για το στάδιο αυτό είναι ο εξής:*

a) Δημιουργία της Δυαδικής Εικόνας Πρόσημου X της L. X = (L>=0)
b) Εύρεση του περιγράμματος της X (συμμετρικό ως προς X και X^c):
$Y = (X \oplus B) - (X \ominus B) \approx \partial X$.

Τα zero-crossings αντιστοιχούν στα σημεία στα οποία η δυαδική εικόνα Y έχει την τιμή 1.

1.2.4 *Ζητείται η απόρριψη των zero-crossings σε σχετικά ομαλές περιοχές. Τελικά επιλέγονται ως ακμές τα zero-croosings Y στα οποία η εξομαλυμένη εικόνα Iς έχει σχετικά μεγάλη κλίση, δηλ. τα pixels (i,j) για τα οποία:*

$$Y[i,j] = 1 \text{ και } \left|\left|\nabla I\sigma(x,y)\right|\right| > \theta_{edge} \max_{x,y} \left|\left|\nabla I\sigma(x,y)\right|\right|$$

Διαισθητικά, για την παραπάνω υλοποίηση, τα σημεία ενδιαφέροντος είναι τα zero-crossings της 2ης Παραγώγου $\nabla^2 I(x,y) = 0$. Όμως, επειδή οι εικόνες έχουν θόρυβο και η διαδικασία παραγώγισης είναι ευαίσθητη στο θόρυβο, πρώτα εξομαλύνεται η εικόνα με Gaussian Filtering με χρήση Gaussian Kernel G_σ και έπειτα εφαρμόζεται η παραγώγιση. Οπότε, $\nabla^2(G_\sigma * I) = (\nabla^2 G_\sigma) * I = h * I$. Σημειώνεται ότι η συνάρτηση h μπορεί να υπολογιστεί ξεχωριστά για να χρησιμοποιηθεί και παρακάτω.

Ύστερα, για την ανίχνευση των zero-crossings, με τον τρόπο που περιγράφεται παραπάνω, χρησιμοποιείται ο τελεστής συνόρου boundary_operator στο αρχείο utils.py . Στη συνέχεια, προκύπτουν τα σημεία όπου Y[i,j] = 1 και με AND με την Q[i,j], όπου είναι αληθής όταν

$$\left|\left|\nabla I\sigma(x,y)\right|\right| > \theta_{edge} \max_{x,y} \left|\left|\nabla I\sigma(x,y)\right|\right|$$

Τα φιλτραρισμένα zero-crossings που προκύπτουν ως η εικόνα Q & Y.

Η παραπάνω διαδικασία υλοποιείται με τη συνάρτηση *EdgeDetect* στο αρχείο *EdgeDetect.py* .

1.3 *Αξιολόγηση των Αποτελεσμάτων Ανίχνευσης Ακμών*

1.3.1 *Ζητείται ο υπολογισμός των "αληθινών" ακμών, χρησιμοποιώντας την αρχική καθαρή εικόνα I_0. Η δυαδική εικόνα αληθινών ακμών T μπορεί να βρεθεί αποκλειστικά εφαρμόζοντας τον απλό τελεστή ακμών (edge operator) στη I_0: $M = (I_0 \oplus B) - (I_0 \ominus B)$ και χρησιμοποιώντας κατωφλιοποίηση για μετατροπή της εξόδου σε δυαδική εικόνα: $T = M > \theta_{realedge}$.*

Η παραπάνω διαδικασία υλοποιείται στο αρχείο *utils.py* .

1.3.2 *Εδώ ζητείται η ποσοτική αξιολόγηση των αποτελεσμάτων ανίχνευσης ακμών. Ειδικότερα, χρησιμοποιώντας τις δυαδικές εικόνες των αληθινών ακμών T και των ακμών D που ανίχνευσε ο αλγόριθμος στην θορυβώδη εικόνα εισόδου, ζητείται ο υπολογισμός του ακόλουθου κριτηρίου ποιότητας του αποτελέσματος ανίχνευσης: $C = [Pr(D|T) + Pr(T|D)]/2$, όπου $Pr(D|T)$ είναι το ποσοστό των ανιχνευθεισών ακμών που είναι αληθινές (Precision) και $Pr(T|D)$ είναι το ποσοστό των αληθινών ακμών που ανιχνεύθησαν (Recall). Αν τα D και T ιδωθούν σαν διακριτά σύνολα (με στοιχεία τα pixels στα οποία η δυαδική εικόνα έχει τιμή 1), τότε ισχύει $Pr(T|D) = card(D \cap T)/card(T)$, όπου $card(\cdot)$ δίνει τον αριθμό των στοιχείων ενός συνόλου.*

Συγκεκριμένα, εδώ υπολογίζονται οι πιθανότητες $Pr(D|T) = \frac{|D \cap T|}{|T|}$ και $Pr(T|D) = \frac{|D \cap T|}{|D|}$ καθώς και ο δείκτης $C = \frac{1}{2}[Pr(D|T) + Pr(T|D)$.

Τα αποτελέσματα φαίνονται παρακάτω:

Noisy image with PSNR 20dB, linear filter. C = 0.8724365230689108

Noisy image with PSNR 20dB, morphological filter. C = 0.9058576849155361

Noisy image with PSNR 10dB, linear filter. C = 0.6403443879898569

Noisy image with PSNR 10dB, morphological filter. C = 0.7339753414745319

Η παραπάνω διαδικασία υλοποιείται στο αρχείο *utils.py* .

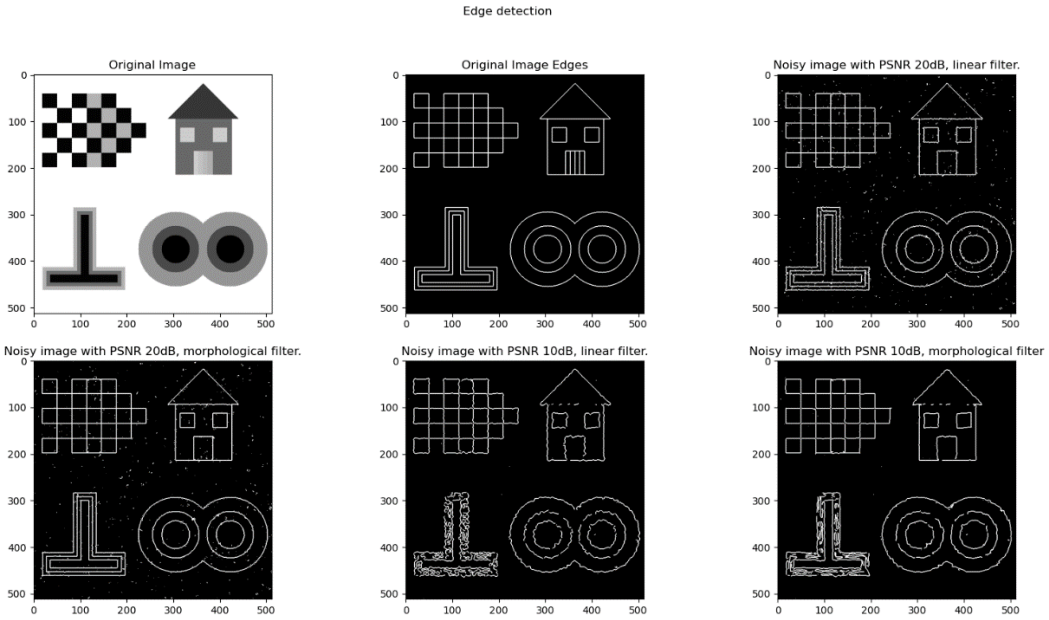
1.3.3 *Στα παραπάνω, για κάθε διαφορετική είσοδο, προτείνεται ο πειραματισμός με τιμές των παραμέτρων σ και* θ_{edge} *ώστε να επιτευχθούν όσο το δυνατόν καλύτερα αποτελέσματα. Για κάθε περίπτωση, ζητείται να παρουσιαστεί το 'καλύτερο' αποτέλεσμα που εξήχθη και να σχολιαστεί. Επίσης, ζητείται ο σχολιασμός της επίδρασης που έχει η μεταβολή των παραμέτρων και του επιπέδου του θορύβου στο αποτέλεσμα ανίχνευσης ακμών. Τέλος, ζητείται η σύγκριση των αποτελεσμάτων για τις διαφορετικές προσεγγίσεις της Laplacian.*

Σχολιασμός αποτελεσμάτων:

Στο Μέρος 1, έγινε εφαρμογή των διαφόρων μεθόδων ανίχνευσης ακμών στην εικόνα "edgetest_10.png" και προέκυψαν τα εξής συμπεράσματα:

- Όταν υπάρχει υψηλό PSNR και προστεθεί λιγότερος θόρυβος, τα αποτελέσματα είναι καλύτερα καθώς η εικόνα με PSNR = 20 dB με γραμμική προσέγγιση της LoG ήταν καλύτερη.
- Όταν αυξάνεται η παράμετρος του thresholding τότε ο αλγόριθμος ανίχνευσης ακμών γίνεται πιο "διακριτικός" και εάν υπερβεί κανείς το όριο, μαυρίζει όλη την εικόνα. Αντιθέτως, χαμηλές τιμές θ_{edge} έχουν περισσότερο θόρυβος στις εικόνες άρα και χειρότερη επίδοση στην ανίχνευση ακμών.
- Παρατηρείται και στις δύο περιπτώσεις ότι η morphological προσέγγιση της LoG δίνει καλύτερα αποτελέσματα από τα μορφολογικά φίλτρα.
- Μεγάλες τιμές του σ έχουν ως αποτέλεσμα το ενδιάμεσο στάδιο, που παράγονται οι L & I_σ να θολώνουν περισσότερο.
- Με Grid Search επιτυγχάνεται η εύρεση των βέλτιστων παραμέτρων, που δίνουν το μέγιστο C.

Επιπλέον, τα αποτελέσματα της ανίχνευσης ακμών απεικονίζονται παρακάτω:

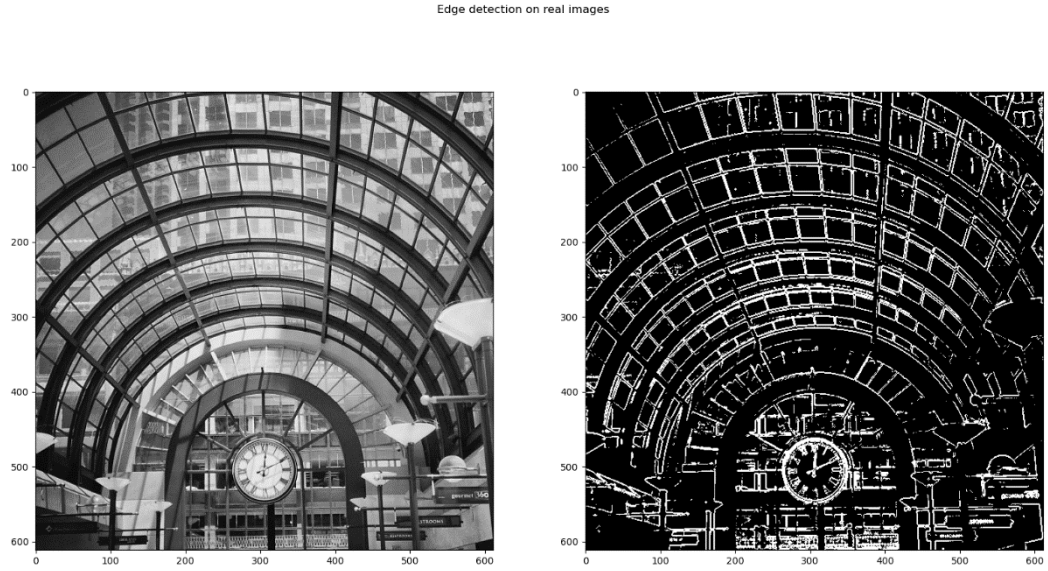


Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part1.py* & *utils.py* .

1.4 *Εφαρμογή των Αλγορίθμων Ανίχνευσης Ακμών σε Πραγματικές εικόνες*

1.4.1 *Ζητείται η εφαρμογή των αλγορίθμων ανίχνευσης ακμών που υλοποιήθηκαν, στην πραγματική εικόνα "urban_edges.jpg" χωρίς την προσθήκη θορύβου.*

Σημειώνεται ότι σαν threshold έχει τεθεί η τιμή $\theta_{real} = 100$. Τα αποτελέσματα που προκύπτουν είναι:



Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part1.py* & *utils.py* .

1.4.2 *Ζητείται και πάλι ο πειραματισμός με τις τιμές των παραμέτρων σ και θ_{edge} ώστε να επιτευχθούν όσο το δυνατόν καλύτερα αποτελέσματα.*

Στην πραγματική εικόνα, η αύξηση της τιμής του threshold ήταν μεγαλύτερη από το αρχικό παράδειγμα ($\theta_{realedge} = 100$). Αυτό δικαιολογείται από το γεγονός ότι η πραγματική εικόνα έχει πιο πολλές μεταβολές οπότε και πολύ μεγαλύτερο gradient.

Σχολιασμός Αποτελεσμάτων:

- Με την αύξηση της τιμής της τυπικής απόκλισης, παρατηρείται στην εικόνα ανίχνευσης ακμών αλλοίωση των ακμών, δηλαδή οι ακμές αρχίζουν να εμφανίζουν μία επιπλέον καμπυλότητα, χωρίς εμφάνιση θορύβου.
- Με τη μείωση της τιμής της τυπικής απόκλισης, παρατηρείται στην εικόνα ανίχνευσης ακμών εμφάνιση θορύβου γεγονός το οποίο είναι ανεπιθύμητο. Όμως, η δυνατότητα διάκρισης των ακμών δεν επηρεάζεται. Το ίδιο συμβαίνει και με τη μείωση της παραμέτρου κατωφλίου.
- Όσο αυξάνεται η τιμή της παραμέτρου κατωφλίου, φαίνεται ότι χάνονται πολλές από τις ακμές. Η απώλεια αυτή, οφείλεται στο γεγονός ότι η αύξηση της παραμέτρου κατωφλίου οδηγεί σε απόρριψη περισσότερων zero-crossings.

Μέρος 2: Ανίχνευση Σημείων Ενδιαφέροντος (Interest Point Detection)

Αυτό το μέρος της εργαστηριακής άσκησης πραγματεύεται την ανίχνευση σημείων ενδιαφέροντος σε εικόνες. Θα υλοποιηθεί ένα σύνολο από ανιχνευτές και θα σχολιαστούν οι διαφορές τους. Για τις ανάγκες του συγκεκριμένου μέρους, δίνονται οι δύο εικόνες ("blood_smear.jpg" & "mars.png") πάνω στις οποίες θα πρέπει να σχεδιαστούν τα αποτελέσματα των ανιχνευτών που υλοποιήθηκαν για ενδεικτικές τιμές των παραμέτρων.

Σημειώνεται ότι για τις ανάγκες της οπτικοποίησης των σημείων ενδιαφέροντος που ανιχνεύονται, χρησιμοποιήθηκε τη συνάρτηση interest_points_visualization ελαφρώς παραλλαγμένη.

2.1 *Ανίχνευση Γωνιών*

Μια διαισθητικά απλή κατηγορία σημείων ενδιαφέροντος είναι η επιλογή σημείων που αντιστοιχούν σε γωνίες της εικόνας. Για την ανίχνευση των γωνιών ζητείται η υλοποίηση της κλασσικής μεθόδου των Harris-Stephens, όπως εξηγείται παρακάτω. Για το ερώτημα αυτό και μόνο, να εφαρμοστεί η αναπτυχθείσα μέθοδος και στην εικόνα ακμών "urban_edges.jpg".

2.1.1 *Ζητείται ο υπολογισμός των στοιχείων J_1, J_2, J_3 του δομικού τανυστή J σε κάθε pixel (x,y) της εικόνας, σύμφωνα με τις σχέσεις:*

$$J_1(x,y) = G_\rho * \left(\frac{\partial I_\sigma}{\partial x} \cdot \frac{\partial I_\sigma}{\partial x}\right)(x,y)$$

$$J_2(x,y) = G_\rho * \left(\frac{\partial I_\sigma}{\partial x} \cdot \frac{\partial I_\sigma}{\partial y}\right)(x,y)$$

$$J_3(x,y) = G_\rho * \left(\frac{\partial I_\sigma}{\partial y} \cdot \frac{\partial I_\sigma}{\partial y}\right)(x,y)$$

όπου $I_\sigma = I * G_\sigma$ και Gρ, Gσ δισδιάστατοι Gaussian πυρήνες ομαλοποίησης με τυπικές αποκλίσεις σ (κλίμακα διαφόρισης) και ρ (κλίμακα ολοκλήρωσης) αντίστοιχα.

Στην Όραση Υπολογιστών, συχνά ζητείται η διάκριση κοινών χαρακτηριστικών μεταξύ όψεων των ίδιων αντικειμένων, όπως λ.χ. στην στερεοπροβολή και το motion estimation. Οι γωνίες μιας εικόνας ορίζονται ως οι περιοχές συνάντησης ακμών στην εικόνα. Πιο διαισθητικά, ζητείται σε μια γειτονιά να ανιχνευθούν μεγάλες αλλαγές προς όλες τις κατευθύνσεις της γειτονιάς. Γενικότερα, οι γωνίες αποτελούν ευσταθή χαρακτηριστικά όταν αλλάζει η προοπτική του αντικειμένου.

Στην ανίχνευση γωνιών αναζητείται το σημείο ενδιαφέροντος κοιτώντας την ένταση σε σημεία κοντά στο query point με χρήση κάποιου παραθύρου. Μεταβάλλοντας την θέση του παραθύρου, αναμένεται μεγάλη μεταβολή προς όλες τις κατευθύνσεις. Ο ανιχνευτής Harris-Stephens αποτελεί μια μαθηματική προσέγγιση που περιγράφεται παρακάτω. Η μεταβολή της φωτεινότητας E(u,v) μιας εικόνας σε ένα σημείο (u,v) ως

$$\mathrm{E(u,v)} = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \approx \begin{pmatrix} u & v \end{pmatrix} \sum_{x,y} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

Όπου $M = \sum_{x,y} w(x,y) \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$ και w(x,y) η συνάρτηση παραθύρωσης και M ο πίνακας των παραγώγων της εικόνας I.

Διαισθητικά τα στοιχεία της $\nabla I(x, y) = (I_x, I_y)$ μπορούν να θεωρηθούν ως κατανομές για την ανίχνευση των γωνιών. Στην κατανομή (I_x, I_y) γίνεται μια προσπάθεια για ταίριασμα των ελλείψεων με βάση τα μέτρα των ιδιοτιμών λ_+, λ_- του M (που αντιστοιχούν στις principal διευθύνσεις). Σε περιπτώσεις που υπάρχουν γωνίες, παρατηρείται ότι $\lambda_+ \approx \lambda_-$ και ότι οι τιμές τους είναι μεγάλες σε αντίθεση με τα flat regions. Ένα καλό μέτρο για την ταξινόμηση γωνιών είναι η ποσότητα:

$$R = |M| - k\,tr^2(M) = \lambda_+\lambda_- - k(\lambda_+ + \lambda_-)^2$$

που ονομάζεται και corner response ενώ το $k \in [0.04, 0.06]$ είναι μια εμπειρικά προσδιορισμένη παράμετρος. Επιπλέον, σημειώνεται ότι έχει γίνει μια μικρή τροποποίηση στον ανιχνευτή ώστε να λαμβάνει τα τρία κανάλια R, G, B της εικόνας και η ανίχνευση τρέχει σε καθένα ξεχωριστά, ενώ σαν αποτέλεσμα επιστρέφει το OR των γωνιών για όλα τα κανάλια. Στις περιοχές που το R έχει μεγάλη τιμή εντοπίζεται μια γωνία. Για την υλοποίηση του Harris detector υπολογίζονται οι δομικοί τανυστές J_1, J_2, J_3 που δίνονται ως

$$J_1 = G_\rho * (I_{\sigma,xx})^2 \qquad J_2 = G_\rho * (I_{\sigma,xx} \cdot I_{\sigma,yy}) \qquad J_3 = G_\rho * (I_{\sigma,yy})^2$$

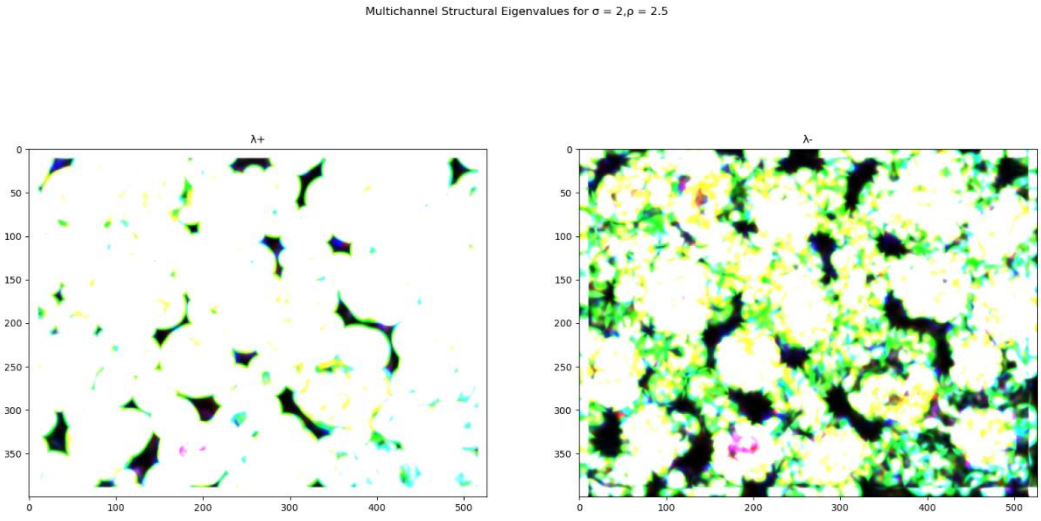
Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part2.py* & *utils.py* .

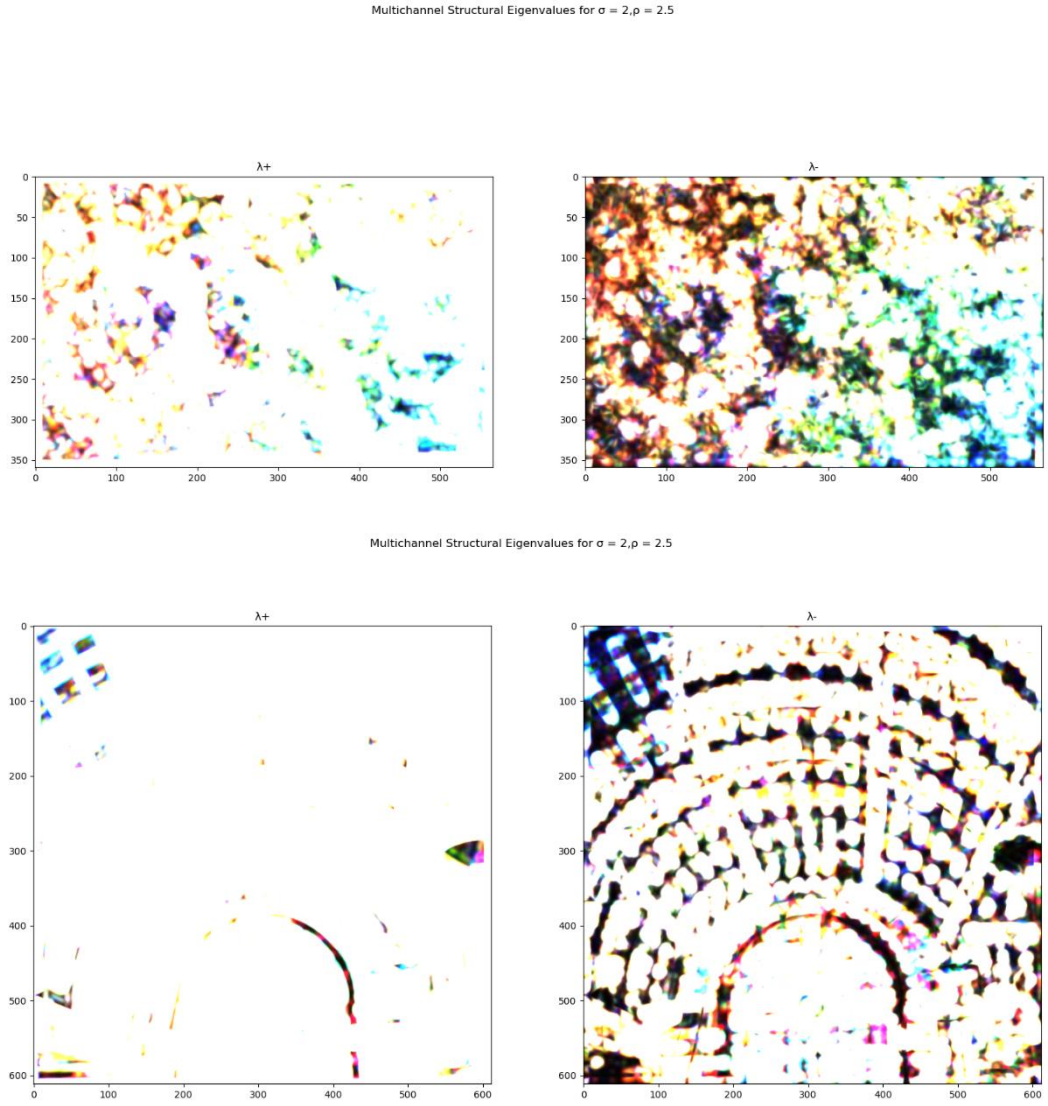
2.1.2 *Ζητείται ο υπολογισμός των ιδιοτιμών λ_-, λ_+, του τανυστή J σε κάθε pixel, σύμφωνα με τις σχέσεις:* $\lambda_\pm = \frac{1}{2}(J_1 + J_3 \pm \sqrt{(J_1 - J_3)^2 + 4J_2^2}$ *καθώς και ο σχεδιασμός σαν γκρίζες εικόνες των λ_-, λ_+. Επίσης, ζητείται σχολιασμός για τη σχέση των τιμών αυτών με τα χαρακτηριστικά της εικόνας.*

Στη συνέχεια υπολογίζονται οι ιδιοτιμές του τανυστή J ως

$$\lambda_\pm = \frac{1}{2}(J_1 + J_3 \pm \sqrt{(J_1 - J_3)^2 + 4J_2^2})$$

Τα αποτελέσματα (σε multichannel) για τις ιδιοτιμές λ_+, λ_- απεικονίζονται ακολούθως:





Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part2.py* & *utils.py* .

2.1.3 *Με βάση τις υπολογισμένες ιδιοτιμές, ζητείται να εξαχθεί το ακόλουθο "κριτήριο γωνιότητας": R* $= \lambda_+\lambda_- - k(\lambda_- + \lambda_+)^2$

Όπου k, είναι μια μικρή θετική σταθερά. Τελικά, να επιλεχθούν σαν γωνίες τα pixels (x,y) τα οποία:

(Σ1) Είναι μέγιστα του R εντός τετραγωνικών παραθύρων που τα περιβάλλουν, το μέγεθος των οποίων εξαρτάται από την κλίμακα σ,

(Σ2) Αντιστοιχούν σε τιμή του R μεγαλύτερη από ένα ποσοστό του ολικού μεγίστου του R, δηλ. ότι $R(x,y) > \theta_{corn} \cdot R_{max}$*, όπου* θ_{corn} *ένα κατάλληλα επιλεγμένο κατώφλι.*

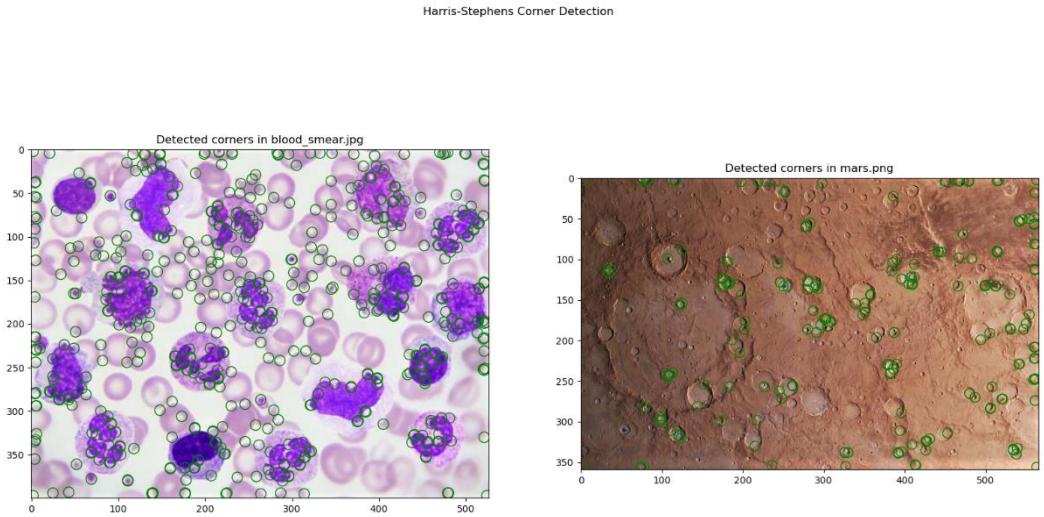
Για το κριτήριο γωνιότητας (cornerness criterion) ως

$$R = \lambda_+\lambda_- - k(\lambda_- + \lambda_+)^2$$

Τα σημεία δίνονται ως η συναλήθευση C1 & C2 όπου

$$C_1[i, j] = (R[i, j] == (R \oplus B(n_s))[i, j]) \qquad C_2[i, j] = (R[i, j] > \theta_{corn} \cdot R_{max})$$

Τα αποτελέσματα για την ανίχνευση γωνιών Harris-Stephens για $\sigma = 2$, $\rho = 2.5$, $\theta_{corn} = 0.005$, $k = 0.05$ απεικονίζονται στη συνέχεια.



Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part2.py* & *utils.py* .

Σχολιασμός αποτελεσμάτων:

- Παρατηρείται ότι ο ανιχνευτής Harris-Stephens κατάφερε με επιτυχία να υπολογίσει τις γωνίες στις δύο εικόνες.

2.2 *Πολυκλιμακωτή Ανίχνευση Γωνιών*

Στο ερώτημα αυτό επεκτείνεται η παραπάνω μέθοδος ώστε να ανιχνεύει γωνίες σε πολλαπλές κλίμακες και να επιστρέφει εκτός από τα σημεία που αντιστοιχούν σε γωνίες και την κλίμακα στην οποία ανιχνεύτηκαν. Η μέθοδος αυτή λέγεται Harris-Laplacian και αποτελείται από δύο στάδια, ένα για την επιλογή σημείων σε κάθε κλίμακα και ένα για την τελική επιλογή των σημείων που παρουσιάζουν μέγιστο σε κάποια μετρική αμετάβλητη ως προς την κλίμακα.

2.2.1 *Το πρώτο στάδιο αποτελείται από την εφαρμογή του αλγορίθμου εύρεσης γωνιών μονής κλίμακας για διαφορετικές κλίμακες ολοκλήρωσης και διαφόρισης:*

$$\sigma_0, \sigma_1, \ldots \sigma_{N-1} = s^0\sigma_0, s^1\sigma_0, \ldots, s^{N-1}\sigma_0$$

$$\rho_0, \rho_1, \ldots \rho_{N-1} = s^0\rho_0, s^1\rho_0, \ldots, s^{N-1}\rho_0$$

όπου σ_0, ρ_0 αρχικές κλίμακες διαφόρισης και ολοκλήρωσης, s ο παράγοντας ολοκλήρωσης και N ο αριθμός των κλιμάκων.

2.2.2 *Το δεύτερο στάδιο αφορά την αυτόματη επιλογή της χαρακτηριστικής κλίμακας για κάθε σημείο που ανιχνεύτηκε στο προηγούμενο βήμα. Αρχικά, υπολογίζεται η κανονικοποιημένη LoG για τις διαφορετικές κλίμακες διαφόρισης του προηγούμενου βήματος:*

$$|LoG(x, \sigma_i)| = \sigma_i^2|L_{xx}(x, \sigma_i) + L_{yy}(x, \sigma_i)|,\ i = 0, \ldots, N-1$$

Στη συνέχεια, απορρίπτονται τα σημεία για τα οποία η κλίμακα που ανιχνεύθηκαν δεν μεγιστοποιεί την LoG μετρική σε μια γειτονιά 2 διαδοχικών κλιμάκων.

Στην πολυκλιμακωτή ανίχνευση γωνιών, τα αποτελέσματα της single-scale ανίχνευσης γωνιών επεκτείνονται σε μια γεωμετρική πρόοδο από κλίμακες διαφόρισης και ολοκλήρωσης. Σκοπός είναι να διατηρηθούν οι ROIs όπου η κλίμακα των χαρακτηριστικών που ανιχνεύτηκε περιέχει την περισσότερη πληροφορία σε σχέση με τις γειτονικές της κλίμακες. Για αυτό, χρησιμοποιείται μια (LoG) με την οποία αξιολογούνται τα διάφορα σημεία. Η μέθοδος ονομάζεται Harris–Laplacian και διατηρεί μόνο ορισμένες γωνίες ανά κλίμακα. Οι κλίμακες δίνονται από τα $\sigma_i = s^i\sigma_0$, $\rho_i = s^i\rho_0$ για i = 0, 1, . . . , N − 1, όπου N το πλήθος των κλιμάκων. Στην προσέγγιση αυτή χρησιμοποιείται ο τανυστής C όπου το στοιχείο C(x,y,i) αντιστοιχεί στο αν υπάρχει γωνία στο (x,y) για την κλίμακα σ_i. Η μετρική LoG για κάθε κλίμακα υπολογίζεται ως εξής:

$$LoG(x,y,i) = \sigma_i^2(L_{xx}(x,y,\sigma_i) + L_{yy}(x,y,\sigma_i)) \qquad \text{όπου } L_{xx} = \frac{\partial^2 I_\sigma}{\partial x^2} \text{ και } L_{yy} = \frac{\partial^2 I_\sigma}{\partial y^2}$$

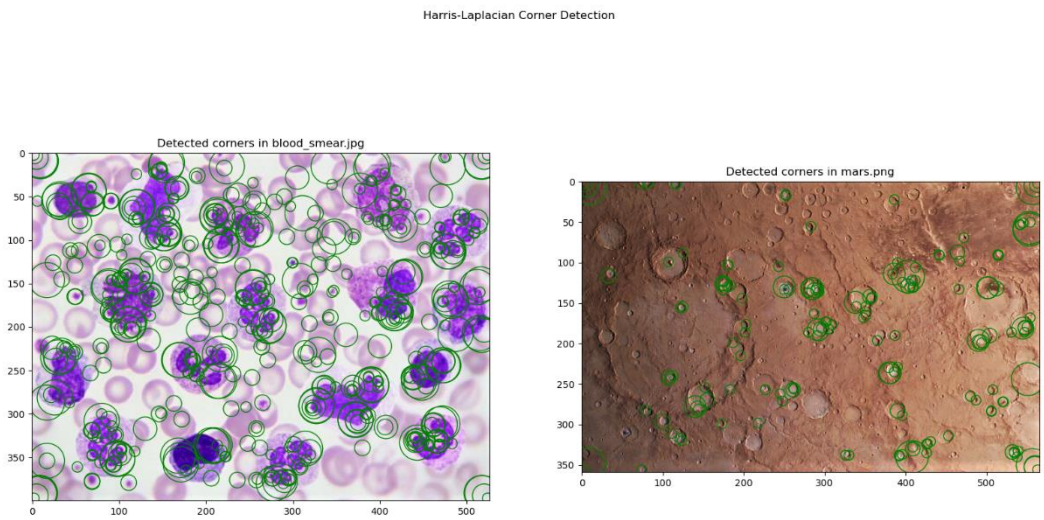
Στη συνέχεια, υπολογίζεται ο πίνακας με τους δείκτες των κλιμάκων για τις οποίες η μετρική |LoG| είναι μέγιστη. Τα στοιχεία του πίνακα αυτού είναι τα:

$$f(x,y,i) = (|LoG(x,y,i)|) \geq \max_{j \neq i, |j-i| \leq 1} \{|LoG(x, y, i)|\}$$

Τέλος, για κάθε κλίμακα, τα corners που θα διατηρηθούν τελικά είναι τα εξής:

$$\hat{C}(x, y, i) := C(x, y, i) \wedge f(x, y, i)$$

Τα αποτελέσματα για τις προτεινόμενες παραμέτρους ($\sigma = 2$, $\rho = 2.5$, $\theta_{corn} = 0.005$, $k = 0.05$, $N = 4$) παρουσιάζονται παρακάτω.



Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part2.py* & *utils.py* .

2.3 *Ανίχνευση Blobs*

Μια από τις σημαντικότερες κατηγορίες σημείων ενδιαφέροντος βασίζονται στην ανίχνευση 'blobs', που ορίζονται ως περιοχές με κάποια ομοιογένεια που διαφέρουν σημαντικά από την γειτονιά τους. Για την εύρεση τέτοιων περιοχών, σε αντιστοιχία με το κριτήριο γωνιότητας της μεθόδου Harris, γίνεται χρήση των μερικών παραγώγων δεύτερης τάξης της εικόνας και συγκεκριμένα η ορίζουσα του πίνακα Hessian:

$$H(x,y) = \begin{bmatrix} L_{xx}(x,y,\sigma) & L_{xy}(x,y,\sigma) \\ L_{xy}(x,y,\sigma) & L_{yy}(x,y,\sigma) \end{bmatrix}$$

Όπου $L_{xx}(x,y,\sigma) = \frac{\partial^2}{\partial x^2}\{I_\sigma(x,y)\}$, $L_{yy}(x,y,\sigma) = \frac{\partial^2}{\partial y^2}\{I_\sigma(x,y)\}$ *και* $L_{xy}(x,y,\sigma) = \frac{\partial^2}{\partial x \partial y}\{I_\sigma(x,y)\}$

2.3.1 *Ζητείται ο υπολογισμός των μερικών παραγώγων δεύτερης τάξης της εικόνας* L_{xx}, L_{xy}, L_{yy} *επιλέγοντας μια τιμή για την κλίμακα σ και η κατασκευή του κριτηρίου* $R(x,y) = det(H(x,y))$ *για κάθε pixel.*

2.3.2 *Ζητείται να επιλεχθούν ως σημεία ενδιαφέροντος, τα σημεία που αποτελούν τοπικά μέγιστα και έχουν τιμή μεγαλύτερη από ένα κατάλληλα ορισμένο κατώφλι, ακριβώς όπως και στην ανίχνευση γωνιών με την μέθοδο Harris.*

Για μια grayscale εικόνα, διαισθητικά ένα blob είναι μια πυκνή περιοχή σε μια εικόνα που τα pixels της περιοχής έχουν παρόμοιες τιμές intensity μεταξύ τους και αρκετά διαφορετικές από τα pixels σε μια μικρή γειτονιά έξω από την περιοχή. Μπορούν επίσης να αντιστοιχούν σε σχήματα ή άλλα αντικείμενα του φυσικού κόσμου.

Η ανίχνευση blobs μοιάζει πολύ με την ανίχνευση γωνιών που υλοποιήθηκε παραπάνω με τη μόνη διαφορά ότι στη μέθοδο εντοπισμού αντί για το κριτήριο γωνιότητας, τώρα χρησιμοποιείται το "κριτήριο ανίχνευσης blob" που και αυτό χρησιμοποιεί τη δεύτερη παράγωγο της φιλτραρισμένης εικόνας όπως φαίνεται και παρακάτω.

Έχουν ήδη οριστεί οι τελεστές L_{xx}, L_{yy}, L_{xy} που επιδρούν πάνω στην φιλτραρισμένη με την δισδιάστατη γκαουσιανή συνάρτηση εικόνα I_σ ως εξής:

$$L_{xx}(x,y,\sigma) = \frac{\partial^2}{\partial x^2}\{I_\sigma(x,y)\},\ L_{yy}(x,y,\sigma) = \frac{\partial^2}{\partial y^2}\{I_\sigma(x,y)\} \text{ και } L_{xy}(x,y,\sigma) = \frac{\partial^2}{\partial x\,\partial y}\{I_\sigma(x,y)\}$$

Οι τελεστές αυτοί χρησιμοποιούνται για να οριστεί ο Hessian πίνακας:

$$H(x,y) = \begin{bmatrix} L_{xx}(x,y,\sigma) & L_{xy}(x,y,\sigma) \\ L_{xy}(x,y,\sigma) & L_{yy}(x,y,\sigma) \end{bmatrix}$$

Το κριτήριο ανίχνευσης blobs, είναι η ορίζουσα αυτού του πίνακα. Δηλαδή:

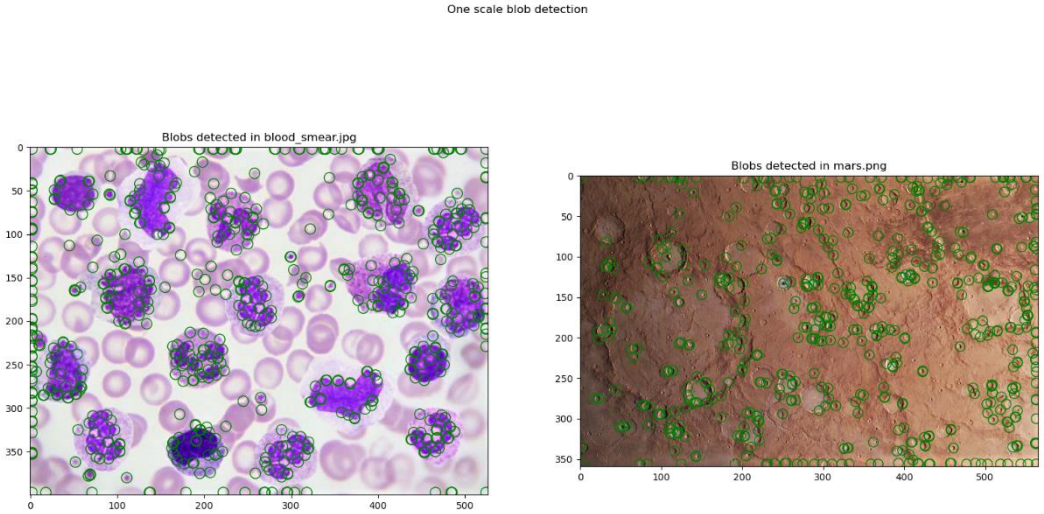
$$R(x,y) = \det(H(x,y)) = L_{xx} \cdot L_{yy} - L_{xy} \cdot L_{xy}$$

Αφού λοιπόν έχει υπολογιστεί το κριτήριο ανίχνευσης blobs R(x,y), η μέθοδος εντοπισμού των blobs θα είναι ακριβώς όπως παραπάνω.

Ως γειτονιά ορίζεται ένα τετραγωνικό παράθυρο, του οποίου το μέγεθος εξαρτάται από την παράμετρο σ, και επιλέγονται, από τα σημεία για τα οποία το R(x,y) δίνει τοπικό μέγιστο στη γειτονιά τους, εκείνα για τα οποία:

$$R(x,y) > \theta_{blob} \cdot \max R(x, y),\ \theta_{blob} \in (0,1)$$

Τα αποτελέσματα που προκύπτουν από την ανίχνευση blobs σε μια κλίμακα απεικονίζονται παρακάτω.



Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part2.py* & *utils.py* .

Σχολιασμός αποτελεσμάτων:

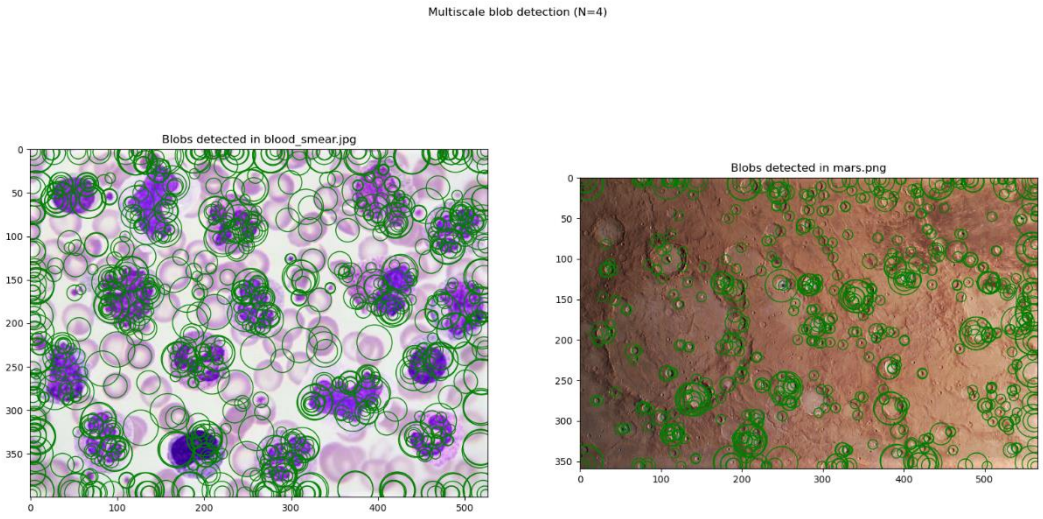
- Παρατηρείται ότι τα blobs και των δύο εικόνων ανιχνεύθηκαν με επιτυχία.
- Η παράμετρος κλίμακας, ακριβώς όπως και στην ανίχνευση γωνιών, καθορίζει το μέγεθος των αντικειμένων για τα οποία αναζητούνται blobs.
- Η παράμετρος θ_{blob} καθορίζει την "ανεκτικότητα" στην εύρεση blobs καθώς, όσο αυξάνεται το threshold, τόσο μειώνονται τα "false alarms" (αύξηση Recall) αλλά ταυτόχρονα αυξάνεται ο κίνδυνος να χαθούν κάποια blobs (μείωση Precision). Το ζήτημα είναι να βρεθεί ο συνδυασμός των Recall, Precision που βελτιστοποιεί το quality factor C γεγονός που όμως δεν μελετάται στην παρούσα άσκηση.
- Παρατηρείται ότι στις άκρες των εικόνων υπάρχουν false alarms. Αυτό συμβαίνει διότι το παράθυρο μικραίνει στη γειτονιά αυτή και συνεπώς αυξάνεται η πιθανότητα να περάσει κάποιο pixel (x,y) το 1ο μέρος ως προς τον εντοπισμό blobs. Μια πιθανή λύση του ζητήματος αυτού θα ήταν η επέκταση της γειτονιάς προς τη μία κατεύθυνση για τα σημεία του πλέγματος της εικόνας.

2.4 *Πολυκλιμακωτή Ανίχνευση Blobs*

2.4.1 *Ζητείται να επαναληφθεί η διαδικασία της πολυκλιμακωτής ανίχνευσης γωνιών, προσθέτοντας ένα δεύτερο στάδιο για την επιλογή σημείων ενδιαφέροντος που παρουσιάζουν μέγιστο σε γειτονικές κλίμακες (Hessian-Laplace). Τα αρχικά σημεία ενδιαφέροντος για κάθε κλίμακα θα επιλεχθούν σύμφωνα με την μέθοδος του προηγούμενου ερωτήματος.*

Με τρόπο ανάλογο με την ανίχνευση γωνιών, μπορούν να ανιχνευθούν blobs σε πολλές διαφορετικές κλίμακες. Οι κλίμακες που εξετάζονται κι εδώ ακολουθούν γεωμετρική πρόοδο. Επειδή τα κριτήρια είναι ακριβώς τα ίδια με πριν, δηλαδή ξεχωριστός υπολογισμός της LoG(x, y, σ) μετρικής για κάθε μία από τις κλίμακες που εξετάζεται και στη συνέχεια επιλογή σημείων που αποτελούν τοπικά μέγιστα σε ένα παράθυρο μήκους 2 ως προς τις κλίμακες, κρίνεται σκόπιμη η απευθείας παράθεση των αποτελεσμάτων.

Τα αποτελέσματα φαίνονται παρακάτω:



Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part2.py* & *utils.py* .

Σχολιασμός αποτελεσμάτων:

- Παρατηρείται ότι τα αποτελέσματα είναι πιο ποιοτικά σε σχέση με την ανίχνευση blobs σε μία κλίμακα.
- Παρατηρείται ότι η αλλαγή της κλίμακας μεταξύ δύο configurations (για $\sigma_0 = 2$ & για $\sigma_0 = 0.2$) οδήγησε σε περισσότερα pixels που κατηγοριοποιήθηκαν ως blobs και παρ' όλα αυτά δεν αλλοίωσε την ανίχνευση ως προς την γεωμετρία των σχημάτων που οριοθετούν αυτά τα blobs.
- Παρατηρείται πως η πολυκλιμακωτή ανίχνευση blobs δίνει σημαντικά καλύτερα αποτελέσματα σε σχέση με την ανίχνευση blobs μιας κλίμακας. Αυτό συμβαίνει καθώς ανιχνεύονται και blobs μεγαλύτερου μεγέθους λόγω των διαφορετικών τιμών των κλιμάκων που χρησιμοποιήθηκαν. Με αύξηση του αριθμού των κλιμάκων αλλά και εκκίνηση από πιο μικρή τιμή του σ μπορούν να επιλεχθούν και ακόμα περισσότερα σημεία.

2.5 *Επιτάχυνση με την χρήση Box Filters και Ολοκληρωτικών Εικόνων (Integral Images)*

Ο υπολογισμός της Hessian για κάθε κλίμακα αντιστοιχεί στην συνέλιξη της εικόνας με αυξανόμενου μεγέθους φίλτρα, που είναι μια υπολογιστικά ακριβή διαδικασία. Για την επιτάχυνση της μεθόδου αυτής, στο Παράρτημα προτάθηκα η προσέγγιση των φίλτρων $2^{ης}$ παραγώγου με Box Filters, δηλαδή με φίλτρα που βασίζονται σε αθροίσματα ορθογωνίων περιοχών, όπως φαίνεται και στην εικόνα της εκφώνησης. Η υλοποίηση τέτοιων αθροισμάτων γίνεται πολύ αποτελεσματικά με χρήση Ολοκληρωτικών Εικόνων.

2.5.1 *Ζητείται ο υπολογισμός της Ολοκληρωτικής Εικόνας.*

2.5.2 *Ζητείται ο υπολογισμός των L_{xx}, L_{yy}, L_{xy} με χρήση της Ολοκληρωτικής Εικόνας σύμφωνα με τα Box Filters (D_{xx}, D_{yy}, D_{xy}) για το επιλεχθέν σ. Στο στάδιο αυτό θα υλοποιηθούν τα φίλτρα της εικόνας 2, όπου κάθε άθροισμα των ορθογωνίων παραθύρων των φίλτρων σταθμίζεται με τον εικονιζόμενο συντελεστή. Για δεδομένο σ, το οποίο αντιστοιχεί σε φίλτρο μεγέθους n x n (με n = 2ceil(3σ)+1), το μέγεθος των παραθύρων υπολογίζεται ως εξής:*

D_{xx} = *ύψος παραθύρου 4 x floor(n/6) + 1, πλάτος παραθύρου 2 x floor(n/6) + 1*

D_{yy} = *ύψος παραθύρου 2 x floor(n/6) + 1, πλάτος παραθύρου 4 x floor(n/6) + 1*

D_{xy} = *ύψος παραθύρου 2 x floor(n/6) + 1, πλάτος παραθύρου 2 x floor(n/6) + 1*

Παρατηρείται ότι για κάθε φίλτρο, είναι θεμιτά τα τοπικά αθροίσματα για ένα σταθερού μεγέθους παράθυρο και το τελικό φίλτρο προκύπτει ως σταθμισμένος συνδυασμός αποκρίσεων για το παράθυρο αυτό.

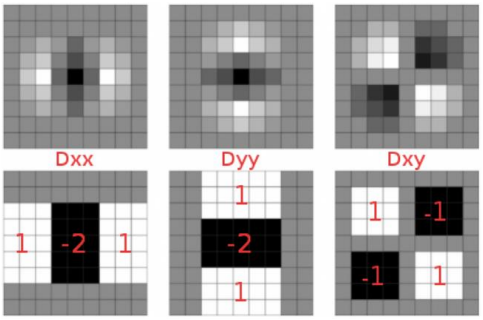
2.5.3 *Ζητείται η εύρεση των σημείων ενδιαφέροντος ως τα τοπικά μέγιστα του κριτηρίου R(x,y) με τον ίδιο τρόπο με τα προηγούμενα ερωτήματα.*

$$R(x,y) = L_{xx}(x,y)L_{yy}(x,y) - (0.9L_{xy}(x,y))^2$$

Επιπλέον, ζητείται να οπτικοποιηθεί ως εικόνα το κριτήριο R της απλής Hessian μεθόδου και το παραπάνω προσεγγιστικό κριτήριο R για κάποια κλίμακα σ. Τέλος, ζητείται να επιλεχθούν 3-4 κλίμακες και να σχολιαστεί η ποιότητα προσέγγισης για αυξανόμενες κλίμακες.

2.5.4 *Ζητείται να επαναληφθεί η διαδικασία της πολυκλιμακωτής ανίχνευσης σημείων ενδιαφέροντος όπως εξηγήθηκε στα προηγούμενα ερωτήματα.*

Σε αυτό το στάδιο θα εκτελεστούν οι ίδιοι αλγόριθμοι χρησιμοποιώντας προσεγγίσεις των L_{xx}, L_{yy}, L_{xy}. Στην περίπτωση της απλής συνέλιξης μιας W×H εικόνας με πυρήνα×K, η συνολική πολυπλοκότητα για τη χωρική συνέλιξη είναι $O(WHK^2)$ στη γενική περίπτωση και O(WHK) αν η συνάρτηση είναι διαχωρίσιμη, ενώ με χρήση FFT μειώνεται σε O((H + W)K log((H + W)K)). Στη γενική περίπτωση τα Box Filters μπορούν να προσεγγίσουν καλά τη συνέλιξη με LoG με χρήση των παρακάτω φίλτρων D_{xx}, D_{yy}, D_{xy}.



Για τη συνέλιξη με rectangular φίλτρο $h : D \to \mathbb{R}$ και σύνολο $R = [a, b] \times [c, d] \subseteq D$ όπου

$$h(x,y) = c_h 1\{(x, y) \in R\}$$

μπορεί να χρησιμοποιηθεί η ολοκληρωτική εικόνα I_C της αρχικής εικόνας όπου

$$I_C(x, y) = \sum_{i \le x, j \le y} I(x, y)$$

και να υπολογιστεί η συνέλιξη $I * h$ ως

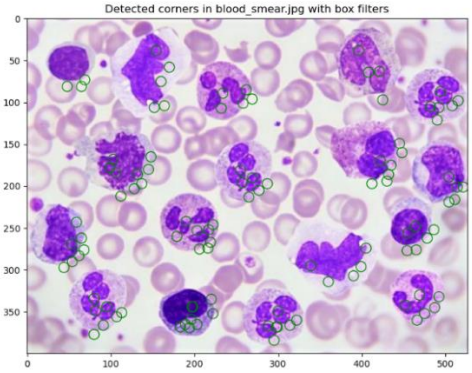
$$(I*h)(x,y) = c_h(I_C(x - a, y - c) + I_C(x - b - 1, y - d - 1) - I_C(x - a, y - d - 1) - I_C(x - b - 1, y - c))$$

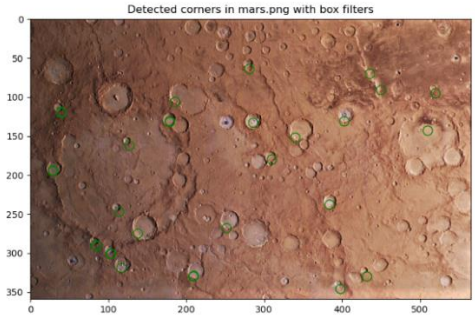
Επομένως μπορούν να "σπάσουν" τα D_{xx}, D_{yy}, D_{yx} σε περιοχές και να εφαρμοστεί η παραπάνω τακτική για να ληφθούν αποτέλεσμα σε $O(W \times H \times F)$ όπου F το πλήθος των χωρίων που γίνεται box φιλτράρισμα, χρόνο με $O(W \times H)$ preprocessing.

Η παραπάνω διαδικασία υλοποιείται στο αρχείο *part2.py* & *utils.py* .

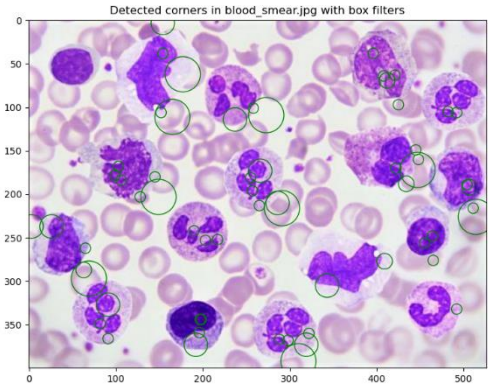
Για τη μέθοδο αυτή τα αποτελέσματα ($\sigma = 3$, $k = 0.05$, $\theta_{corn} = 0.05$, $\rho = 2.5$ και $N = 4$ (για το multiscale) με χρήση Box Filters) των παραπάνω αλγορίθμων συνοψίζονται παρακάτω:

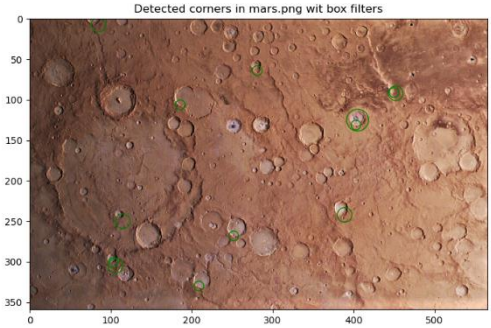
Harris-Stephens Corner Detection with box filters





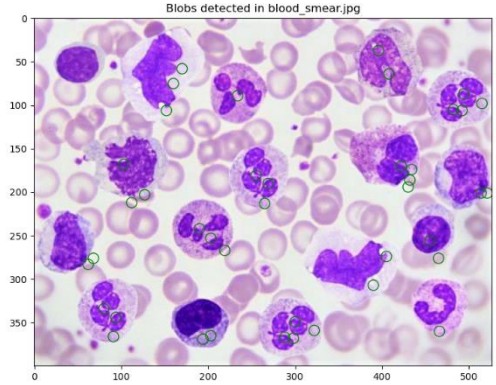
Harris-Laplacian Corner Detection with box filters

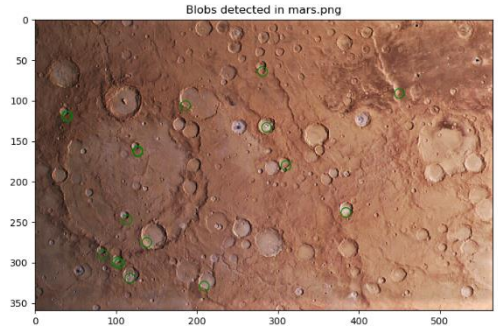




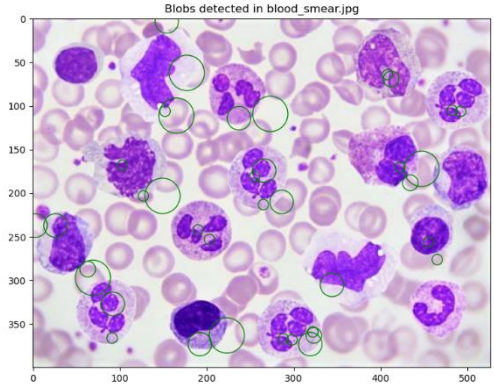
Αποτελέσματα για ανίχνευση μιας κλίμακας και πολυκλιμακωτή ανίχνευση blobs με: $\sigma = \sigma_0 = 3$, $\theta_{blobs} = 0.05$ και $N = 4$ (για το multiscale) με χρήση Box Filters.

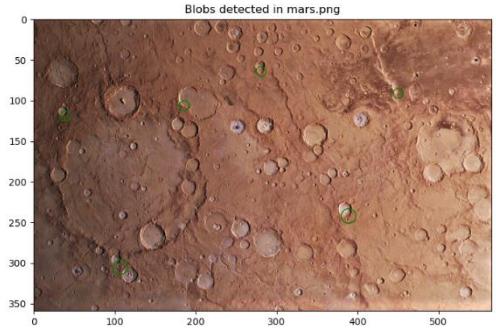
One scale blob detection with box filters





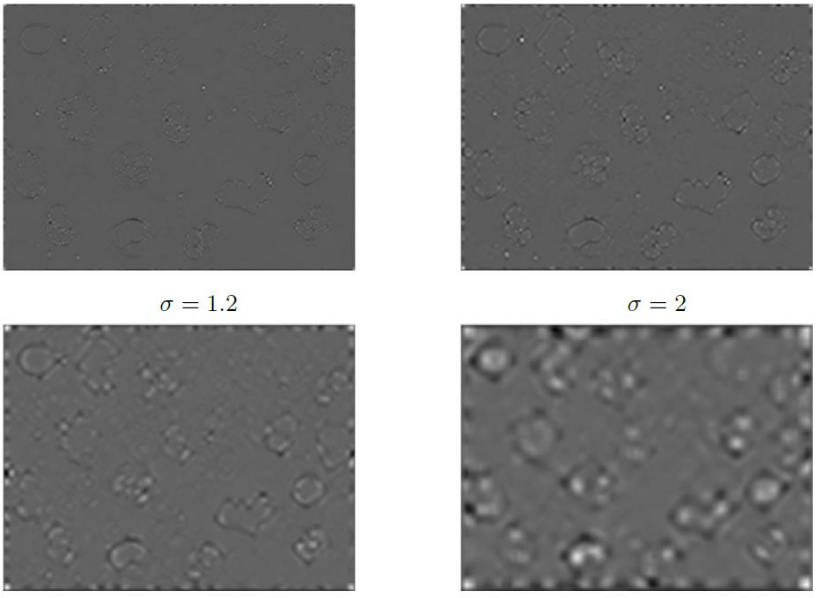
Multiscale blob detection (N=4) with box filters





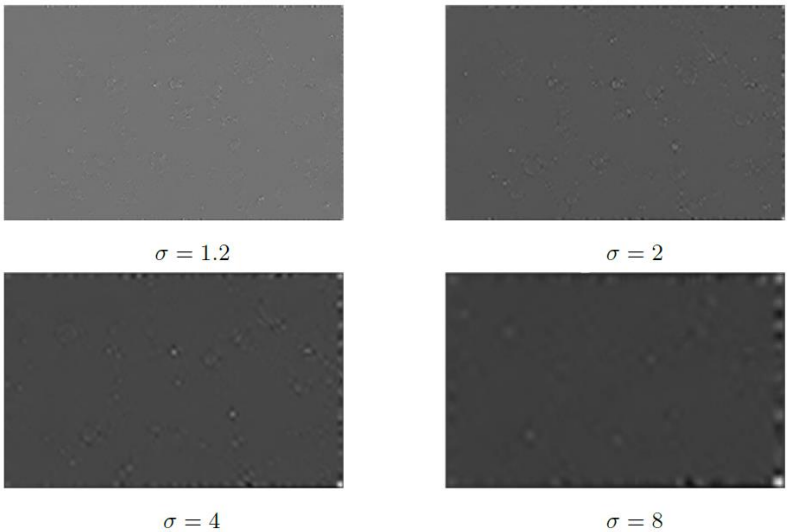
Παρακάτω εμφανίζεται το αποτέλεσμα με χρήση box filters και χωρίς για κλίμακες σ ∈ {1.2, 2, 4, 8} για το κριτήριο γωνιότητας R(x, y) για προσέγγιση με box filters και για γραμμικό φιλτράρισμα με LoG. Παρατηρείται ότι η αύξηση της κλίμακας σ έχει ως αποτέλεσμα το smoothing και των δύο προσεγγίσεων με μικρότερη επίδραση βέβαια στα box filters πράγμα που είναι αναμενόμενο διότι η προσέγγιση των LoG φίλτρων χαλάει όσο αυξάνεται ο πυρήνας του κάθε box filter που είναι ανάλογος στο σ. Γενικότερα, το φιλτράρισμα με box filters δίνει αρκετά ικανοποιητικά αποτελέσματα για $\sigma_{box} > \sigma_{LoG}$ αφού η επίδραση του blurring είναι λιγότερο αισθητή όσο αυξάνεται η κλίμακα σ, όπως άλλωστε διαπιστώνεται στα πειράματα που έγιναν στα προηγούμενα βήματα. Αυτό έχει ως αποτέλεσμα για το ίδιο σ να λαμβάνονται περισσότερα σημεία ενδιαφέροντος στην προσέγγιση με box filters. Οι τιμές της φωτεινότητας είναι ενδεικτικές αφού το R(x, y) συγκρίνεται με το R_{max} για την εξαγωγή των ROIs.

Για το bloos_smear:



$\sigma = 1.2$ $\sigma = 2$ $\sigma = 4$ $\sigma = 8$

Για το mars:



$\sigma = 1.2$ $\sigma = 2$ $\sigma = 4$ $\sigma = 8$

Σχολιασμός αποτελεσμάτων:

- Παρατηρείται από όλα τα παραπάνω γραφήματα πως όσο αυξάνεται η κλίμακα, η ανίχνευση με τα box filters χάνει αρκετά από τα σημεία ενδιαφέροντος. Παρ' όλα αυτά είναι αρκετά όμοια με τα blobs χωρίς αυτή την τεχνική. Η απώλεια αυτή συμβαίνει γιατί εισάγεται ένα σφάλμα ανάλογο με το σ. Οπότε, ναι μεν η χρήση ολοκληρωτικών εικόνων και box filters αυξάνει πολύ την ταχύτητα υπολογισμού του αποτελέσματός μας, αλλά και παράλληλα προσφέρει μια πολύ καλή ανίχνευση για μικρές τιμές της κλίμακας σ. Όμως, για μεγαλύτερες τιμές, η ποιότητα του αποτελέσματος μειώνεται δραματικά. Για το λόγο αυτό χρειάζεται να ελέγχεται αν αξίζει η αλλαγή αυτή μεταξύ ταχύτητας υπολογισμού και υπολογισμού αποτελέσματος.
- Μέσα από όλες αυτές τις εικόνες παρατηρείται πως η πολυκλιμακωτή ανίχνευση σημείων ενδιαφέροντος με τη χρήση box filters και ολοκληρωτικών εικόνων προσφέρει αρκετά καλά αποτελέσματα αντάξια με αυτά των απλών πολυκλιμακωτών ανιχνεύσεων για μικρές τιμές του σ. Αξιοσημείωτο, όμως, είναι πως φαίνονται και κάποιες αστοχίες οι οποίες είναι κυρίως για τις πιο μεγάλες κλίμακες. Αυτές οφείλονται στο ότι η χρήση τον box filters αλλά και των ολοκληρωτικών εικόνων αυξάνει την ταχύτητα αλλά και στο ότι όσο αυξάνεται η κλίμακα μειώνεται η ποιότητα. Βέβαια αξίζει να σημειωθεί πως όσο πιο μεγάλη είναι η αρχική κλίμακα, τόσο λιγότερα θα είναι τα τελικά σημεία ενδιαφέροντος καθώς είναι λιγότερες οι περιοχές που διακρίνονται blobs.

Μέρος 3: Εφαρμογές σε Ταίριασμα και Κατηγοριοποίηση Εικόνων με Χρήση Τοπικών Περιγραφητών στα Σημεία Ενδιαφέροντος

Τα σημεία ενδιαφέροντος δίνουν μια εκτίμηση περιοχών, οι οποίες περιέχουν σημαντικά χαρακτηριστικά της εικόνας. Για τον λόγο αυτό από τις περιοχές αυτές εξάγονται τοπικοί περιγραφητές, που κωδικοποιούν μια γειτονιά γύρω από τα σημεία ενδιαφέροντος. Ως τοπικούς περιγραφητές, θα χρησιμοποιηθούν οι εξής:

- *SURF (Speed Up Robust Features)*
- *HOG (Histogram of Oriented Gradients)*

Και οι δύο περιγραφητές έχουν ως είσοδο μια γειτονιά ενός σημείου ενδιαφέροντος και βασίζονται στην κωδικοποίηση της πληροφορίας υποτμημάτων της γειτονιάς αυτής με χρήση της πρώτης κατευθυντικής παραγώγου. Η μέθοδος των SURF υπολογίζει πρώτα την γενική κατεύθυνση της γειτονιάς, έτσι ώστε να εξαχθούν περιστροφικά ανεξάρτητοι περιγραφητές.

HOG

Τα HOG Features (Histogram of Oriented Gradients) αποτελούν ένα σύνολο χαρακτηριστικών για την περιγραφή της δομής του σχήματος σε μια εικόνα και έχουν χρησιμοποιηθεί με μεγάλη επιτυχία στην αναγνώριση των αντικειμένων, όπως το ανθρώπινο σώμα, που περιέχονται σε μια εικόνα. Τα HOGs παρέχουν μια πυκνή επικαλυπτόμενη περιγραφή των περιοχών μια εικόνας και υπολογίζονται σε ένα πυκνό πλέγμα ομοιόμορφων cells. Τα HOGs χρησιμεύουν στην ομαδοποίηση και κωδικοποίηση των κατευθύνσεων μιας εικόνας σε ιστογράμματα. Για τον υπολογισμό των HOG Features αρχικά λαμβάνεται η παράγωγος της εικόνας I με χρήση Sobel Kernels για την x και y κατεύθυνση αντίστοιχα. Οι τελεστές Sobel δίνονται ως

$$D_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad D_y = D_x^T \qquad g_x = I * D_x \qquad g_y = I * D_y$$

Εν συνεχεία, υπολογίζεται το μέτρο και η φάση της παραγώγου

$$\nabla I(x,y) \approx (g_x(x,y), g_y(x,y)) \qquad M(x,y) = \sqrt{g_x^2 + g_y^2} \qquad A(x,y) = \arg(g_x, g_y)$$

Γύρω από κάθε περιοχή (x_0, y_0) που έχει ενδιαφέρον, εκτείνεται ένα τετραγωνικό χωρίο B(x_0, y_0) το οποίο το μοιράζουμε σε cells και χωρίζουμε σε n_b bins όλες τις γωνίες στο διάστημα 0-360 με βήμα $s = 360/n_b$ για κάθε cell. Η τιμή κάθε bin j ορίζεται ως

$$b(j, x_0, y_0) = \sum_{(u,v) \in C(x_0, y_0)} \mathbf{1}\left\{\lfloor A(u,v)/s \rfloor = j\right\} M(u,v)$$

Στη συνέχεια γίνεται κανονικοποίηση με την L2 Νόρμα

$$\hat{b}(j, x_0, y_0) = \frac{b(j, x_0, y_0)}{\left(\sum_{k=0}^{n_b - 1} b^2(k, x_0, y_0)\right)^{1/2}}$$

Για κάθε παράθυρο γίνονται concat τα HOG όλων των cells και προκύπτουν τα HOG Features για το συγκεκριμένο σημείο. Με τον τρόπο αυτό τα HOG αναδεικνύουν την πληροφορία για το τοπικό σχήμα κωδικοποιώντας τις κατευθύνσεις της κλίσης της εικόνας σε ιστογράμματα. Τέλος, επιτυγχάνουν να είναι αρκετά αμετάβλητα ως προς τις αλλαγές της φωτεινότητας εφαρμόζοντας μια τοπική κανονικοποίηση των ιστογραμμάτων σε επικαλυπτόμενες περιοχές.

SIFT

Ο scale-invariant feature transform (SIFT) είναι ένας αλγόριθμος εντοπισμού χαρακτηριστικών σε εικόνες με ευρύ φάσμα εφαρμογών στην ΟΥ3. Τα σημεία-κλειδιά SIFT των αντικειμένων εξάγονται πρώτα από ένα σύνολο εικόνων αναφοράς και αποθηκεύονται σε μια βάση δεδομένων. Ένα αντικείμενο αναγνωρίζεται σε μια νέα εικόνα, συγκρίνοντας μεμονωμένα κάθε χαρακτηριστικό από τη νέα εικόνα σε αυτή τη βάση δεδομένων και βρίσκοντας υποψήφιες συναρτήσεις με βάση την ευκλείδεια απόσταση των διανυσμάτων χαρακτηριστικών τους. Από το πλήρες σετ αντιστοιχιών, εντοπίζονται υποσύνολα σημείων-κλειδιών που συμφωνούν στο αντικείμενο και η θέση, η κλίμακα και ο προσανατολισμός στη νέα εικόνα για να φιλτράρουν τους καλούς αγώνες. Ο προσδιορισμός συνεπών συστάδων εκτελείται ταχέως με τη χρησιμοποίηση μιας αποτελεσματικής υλοποίησης πίνακα κατακερματισμού του γενικευμένου μετασχηματισμού Hough. Κάθε ομάδα 3 ή περισσότερων χαρακτηριστικών που συμφωνούν σε ένα αντικείμενο και στη θέση του υπόκειται στη συνέχεια σε περαιτέρω λεπτομερή επαλήθευση μοντέλου και, στη συνέχεια, απορρίπτονται. Τέλος, υπολογίζεται η πιθανότητα ότι ένα συγκεκριμένο σύνολο χαρακτηριστικών υποδεικνύει την παρουσία ενός αντικειμένου, δεδομένης της ακρίβειας της προσαρμογής και του αριθμού πιθανών ψευδών αγώνων. Οι αντιστοιχίσεις αντικειμένων που περάσουν όλες αυτές τις δοκιμές μπορούν να αναγνωριστούν ως σωστές με μεγάλη εμπιστοσύνη.

SURF

Ο SURF (Speeded Up Robust Features) είναι και αυτός ένας περιγραφητής που βασίζεται εν πολλοίς στον SIFT. Για την ανίχνευση των σημείων ενδιαφέροντος, το SURF χρησιμοποιεί μια ακέραια προσέγγιση του προσδιοριστή του ανιχνευτή Hessian blob, ο οποίος μπορεί να υπολογιστεί με 3 ακέραιες λειτουργίες χρησιμοποιώντας integral images. Ο descriptor του χαρακτηριστικού βασίζεται στο άθροισμα της απόκρισης κυματιδίων Haar γύρω από το σημείο ενδιαφέροντος. Οι περιγραφητές SURF έχουν χρησιμοποιηθεί για τον εντοπισμό και την αναγνώριση αντικειμένων, προσώπων ή προσώπων, την ανακατασκευή τρισδιάστατων σκηνών, την παρακολούθηση αντικειμένων και την εξαγωγή σημείων ενδιαφέροντος. Η εικόνα μετασχηματίζεται σε συντεταγμένες, χρησιμοποιώντας την τεχνική πυραμίδων πολλαπλών αναλύσεων, για να αντιγράψει την αρχική εικόνα με σχήμα Pyramidal Gaussian ή Laplacian Pyramid για να αποκτήσει μια εικόνα με το ίδιο μέγεθος αλλά με μειωμένο εύρος ζώνης. Αυτό επιτυγχάνει blurring της αρχικής εικόνας (που ονομάζεται και scale-space blurring) και εξασφαλίζει ότι οι ROIs παραμένουν αμετάβλητες.

3.1 *Ταίριασμα Εικόνων υπό Περιστροφή και Αλλαγή Κλίμακας*

Στο μέρος αυτό θα εξεταστεί η ικανότητα εύρεσης της περιστροφής και της κλίμακας με τη χρήση των ανιχνευτών σημείων ενδιαφέροντος που υλοποιήθηκαν παραπάνω καθώς και των τοπικών περιγραφητών. Συγκεκριμένα, για το πείραμα αυτό έχουν χρησιμοποιηθεί 3 εικόνες, οι οποίες έχουν παραμορφωθεί περιστρέφοντάς τες και αλλάζοντας το μέγεθός τους. Συνεπώς, υπάρχουν συνολικά 20 παραμορφώσεις και χρησιμοποιώντας μια από τις εικόνες ως εικόνα αναφοράς, θα γίνει αντιστοίχιση των τοπικών περιγραφητών της με αυτούς των υπόλοιπων παραμορφωμένων εικόνων. Η διαδικασία αυτή ονομάζεται ταίριασμα. Επιλέγοντας ένα σύνολο αντιστοίχισης σημείων – τοπικών περιγραφητών, είναι δυνατόν να εκτιμηθεί ο μετασχηματισμός ομοιότητας μεταξύ των εικόνων και κατ' επέκταση την περιστροφή και διαφορά κλίμακας.

3.1.1 *Η εκτίμηση της περιστροφής και της κλίμακας των εικόνων και η αποτίμησή τους γίνεται αυτόματα από την συνάρτηση* matching_evaluation. *Η συνάρτηση αποτίμησης δέχεται ως ορίσματα μια συνάρτηση για την εξαγωγή σημείων ενδιαφέροντος και μια συνάρτηση για την εξαγωγή τοπικών περιγραφητών και επιστρέφει το μέσο σφάλμα εκτίμησης κλίμακας και το μέσο σφάλμα εκτίμησης γωνίας περιστροφής για καθεμία από τις 6 εικόνες που χρησιμοποιήθηκαν.*

3.1.2 *Για κάθε έναν από τους 5 ανιχνευτές σημείων ενδιαφέροντος που υλοποιήθηκαν παραπάνω, και για τους δύο τοπικούς περιγραφητές, να υπολογιστούν και να σχολιαστεί η ικανότητα εκτίμησης της περιστροφής και κλίμακας των εικόνων για κάθε συνδυασμό.*

Στην ενότητα αυτή διερευνάται η ικανότητα που έχουν οι περιγραφητές που αναφέραμε να ανιχνεύσουν περιστροφές (rotation) ή/και σμικρύνσεις ή μεγενθύσεις (scaling) σε εικόνες. Πιο συγκεκριμένα, έστω $f : A \subseteq Z^2 \rightarrow [0, 255]$ το σήμα μιας grayscale εικόνας και $\hat{f}_{s,\theta}$ το σήμα που περιγράφει την ίδια εικόνα scaled κατά έναν παράγοντα s και περιστραμμένη κατά μια γωνία θ. Αναζητείται η εκτίμηση των παραμέτρων s, θ με αρχική μας πληροφορία τα ROI για την κάθε εικόνα. Η διαδικασία (συνοπτικά) είναι η ακόλουθη:

Algorithm 1: Matching | Rotation, Translation από τα ROI

Input: f, $\hat{f}$

Output: s, theta

1 Βρες ROI για τις εικόνες f, $\hat{f}$ χρησιμοποιώντας έναν από τους ακόλουθους detector (με ή χωρίς box filters):

(I) Harris-Stephens (II) Harris-Laplacian (III) Single-Scale-Blobs (IV) Multiscale-Blobs

2 Βρες descriptors που περιγράφουν τις εικόνες f, $\hat{f}$ με βάση τα ROI χρησιμοποιώντας έναν από τους ακόλουθους τοπικούς περιγραφητές:

(I) SURF (II) SIFT (III) HOG

3 Βρες τα ζεύγη των descriptors

4 Απομόνωσε τους outliers

5 Από τους υπόλοιπους descriptors, υπολόγισε τον ομοιορφικό μετασχηματισμό H

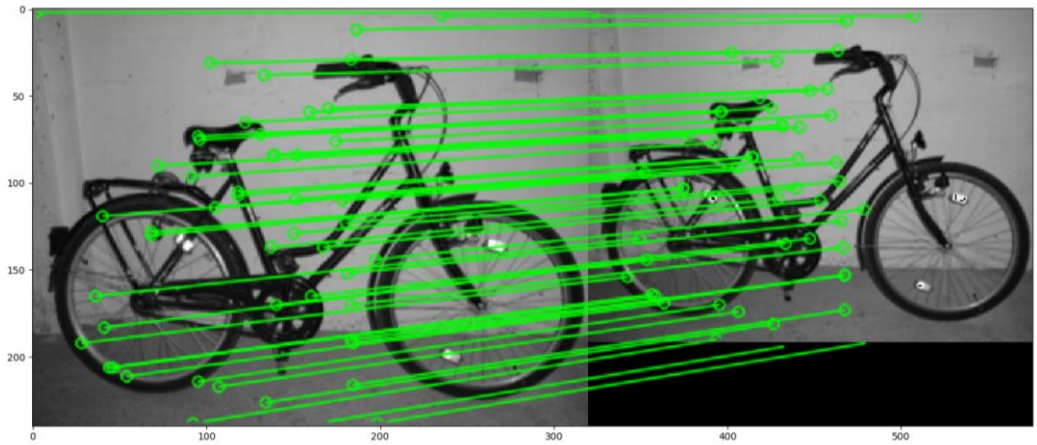
6 Αποσύνθεσε τον ομοιορφικό μετασχηματισμό H σε πιθανούς Translation και Rotation πίνακες

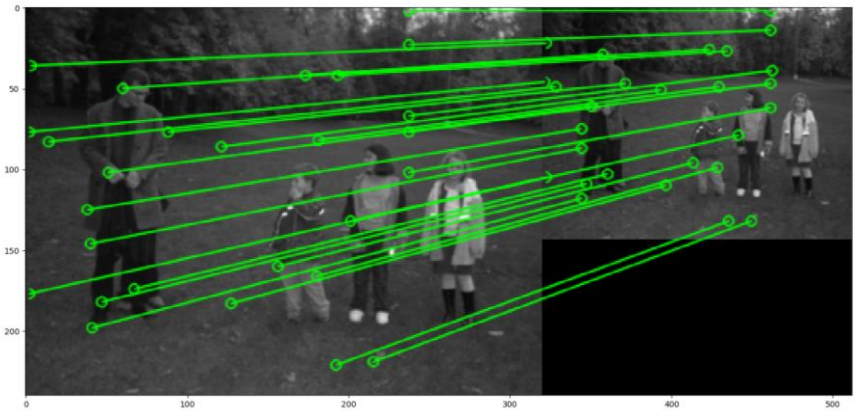
7 Για κάθε πίνακα, αποσύνθεσε και υπολόγισε τα t_x, r_x, t_y, r_y όπου t_x εκφράζει την μετατόπιση κατά τον άξονα x και r_x εκφράζει την περιστροφή ως προς τον άξονα x (αντίστοιχα και για το y).

8 Από τις διάφορες λύσεις, διάλεξε αυτή που ελαχιστοποιεί το: $|f_x - f_y|$

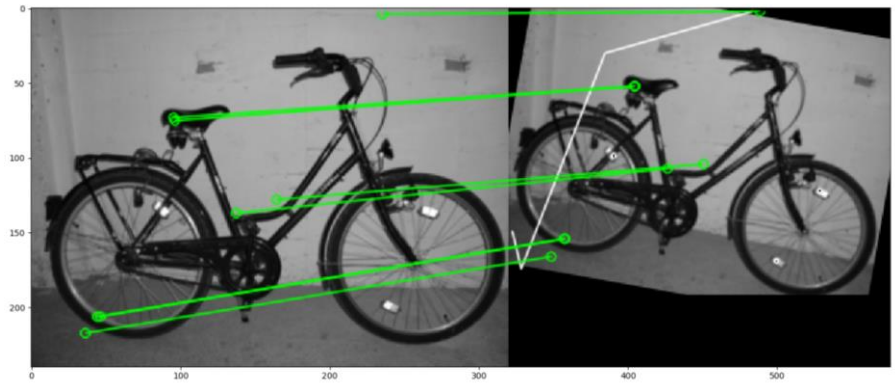
αντίστοιχα το $|r_x - r_y|$

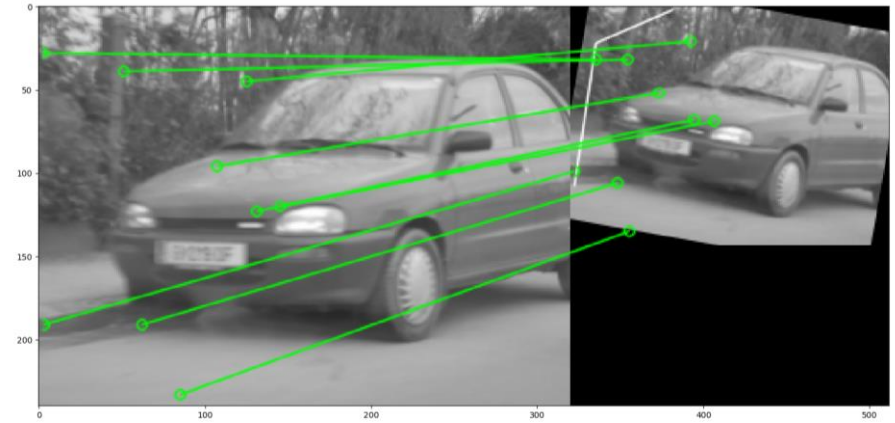
<u>Ανίχνευση scale</u>:

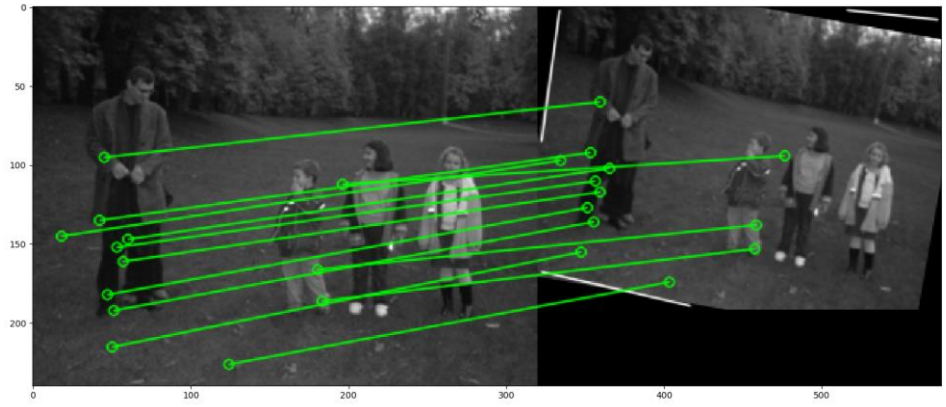




Ανίχνευση rotation:







Πίνακας Αποτελεσμάτων:

Detector	Descriptor	Box Filters	Avg. scale error	Avg. theta error
Harris Laplacian	*SURF*	+	0.66	12.13
Harris Laplacian	*SURF*	-	0.45	11.98
Single Scale blobs	*SURF*	+	0.59	8.73
Single Scale blobs	*SURF*	-	0.36	7.98
Multiscale blobs	*SURF*	+	0.62	8.01
Multiscale blobs	*SURF*	-	0.6	7.35
Harris Laplacian	*SIFT*	+	0.67	11.93
Multiscale blobs	*SIFT*	+	0.68	8.45

Σχολιασμός αποτελεσμάτων:

- Από τον πίνακα, παρατηρείται ότι τα blobs αποτελούν καλύτερο descriptor από τα corners για το πρόβλημα του matching. Τα blobs εφόσον είναι περιοχές με μεγάλη ομοιογένεια στο εσωτερικό τους και μεγάλη ανομοιογένεια στο εξωτερικό τους, αποτελούν σημεία στα οποία είναι αναμενόμενο οι κατευθυντικές παράγωγοι στη γειτονιά τους να έχουν απότομες αλλαγές και συνεπώς να κωδικοποιούν πολλή πληροφορία. Αυτή η διαισθητική ερμηνεία είναι σε αρκετά ικανοποιητική.
- Παρατηρείται ότι οι πολυκλιμακωτές μέθοδοι ανίχνευσης έχουν καλύτερη απόδοση σε σχέση με τις αντίστοιχες μονοκλιμακωτές τόσο στην κλίμακα όσο και στην περιστροφή. Αυτό αποδεικνύεται και από τα παραπάνω, καθώς η ανίχνευση σημείων ενδιαφέροντος γίνεται με μεγαλύτερη απόδοση με τη χρήση πολυκλιμακωτών μεθόδων. Δηλαδή, αντιμετωπίζουν πιο σωστά τις εικόνες με περισσότερες κλίμακες καθώς δεν εξετάζεται μόνο μία τη φορά.
- Επιπλέον, παρατηρείται ότι δεν επιτυγχάνεται πουθενά άριστο αποτέλεσμα. Ενδεικτικά, αξίζει να αναφερθεί ότι δεν έγινε εξαντλητικός έλεγχος στον χώρο των παραμέτρων (hyperparameter tuning). Επίσης, ο matcher που τρέχει είναι ο Flann για λόγους ταχύτητας, όμως το ότι δεν γίνεται εξαντλητικό matching ανεβάζει τα ποσοστά των errors. Επίσης, δεν έχει γίνει διαφοροποίηση στα thresholds μεταξύ των διαφορετικών εικόνων. Σε κάθε περίπτωση, τα αποτελέσματα που προκύπτουν για τις εικόνες παραπάνω είναι σε μεγάλο βαθμό ικανοποιητικά.
- Φαίνεται ότι ο περιγραφητής SURF έχει καλύτερη συμπεριφορά από τον HOG, όσον αφορά την παράμετρο λάθους της γωνίας, ενώ η διαφορά τους όσον αφορά την παράμετρο του λάθους της κλίμακας είναι μικρότερη. Ο περιγραφητής SURF, βρίσκει αρχικά τον Hessian σε κάθε σημείο, στη συνέχεια υπολογίζει τα σημεία ενδιαφέροντος και τέλος, κανονικοποιεί τα ιστογράμματα με βάση την περιστροφή τους. Ο περιγραφητής HOG, υπολογίζει, τα gradients, φτιάχνει τα ιστογράμματα με βάση την περιστροφή τους και ύστερα ενώνει τα blobs και τα κανονικοποιεί.
- Τέλος, σημειώνεται ότι η εύρεση των blobs με τη χρήση box filters και ολοκληρωτικών εικόνων δεν δίνει ικανοποιητικά αποτελέσματα τόσο για την κλίμακα όσο και τη γωνία, διότι ο τρόπος αυτός δεν είναι τόσο ακριβής.

3.2 *Κατηγοριοποίηση Εικόνων*

Στο μέρος αυτό θα αξιολογηθεί η επίδοση και η καταλληλότητα των διαφόρων ανιχνευτών και περιγραφητών σε ένα τυπικό πρόβλημα κατηγοριοποίησης εικόνων. Συγκεκριμένα, δίνεται ένα σύνολο εικόνων από τη βάση Pascal VOC2005. Κάθε εικόνα ανήκει σε μια από τις τρεις ακόλουθες κλάσεις: αυτοκίνητο, άνθρωπος και ποδήλατο. Σκοπός είναι η κατηγοριοποίηση της κάθε εικόνας στη σωστή κλάση, χρησιμοποιώντας σαν χαρακτηριστικά αναγνώρισής τους περιγραφητές που κατασκευάστηκαν παραπάνω.

3.2.1 *Αρχικά, για κάθε εικόνα και από κάθε κλάση θα πρέπει να εξαχθούν τα χαρακτηριστικά με βάση τα οποία θα γίνει η κατηγοριοποίηση. Προτείνεται η χρήση της συνάρτησης FeatureExtraction που δίνεται προκειμένου να εξαχθούν τα χαρακτηριστικά για όλη τη βάση. Για την χρήση διαφορετικών ανιχνευτών / περιγραφητών προτείνεται η χρήση ανώνυμων συναρτήσεων όπως και στο προηγούμενο μέρος. Σημειώνεται ότι ο πειραματισμός αφορά μόνο τις πολυκλιμακωτές εκδόσεις των περιγραφητών.*

3.2.2 *Στη συνέχεια, θα πρέπει να γίνει ο διαχωρισμός των εικόνων στα σύνολα train και test καθώς και η δημιουργία ετικετών που θα δείχνουν την κλάση στην οποία θα ανήκει κάθε εικόνα. Προτείνεται η χρήση της συνάρτησης createTrainTest που δίνεται προκειμένου να γίνει αυτός ο διαχωρισμός χρησιμοποιώντας σαν όρισμα τα χαρακτηριστικά από το προηγούμενο ερώτημα.*

3.2.3 *Κατασκευή αναπαράστασης Bag of Visual Words*

Ο σχετικός κώδικας βρίσκεται στο αρχείο *bovw.py* .

3.2.4 *Το τελικό στάδιο αποτελείται από την τελική κατηγοριοποίηση των εικόνων με βάση την BoVW αναπαράσταση. Για την κατηγοριοποίηση χρησιμοποιείται ένας SVM ταξινομητής κατάλληλα προσαρμοσμένος για πολλαπλές κλάσεις. Η όλη διαδικασία υλοποιείται με τη συνάρτηση svm, η οποία επιστρέφει το αποτέλεσμα της αναγνώρισης καθώς και το συνολικό ποσοστό επιτυχίας. Προτείνεται ο πειραματισμός με τους διαφορετικούς ανιχνευτές / περιγραφητές και ο σχολιασμός του πως αλλάζουν τα ποσοστά αναγνώρισης. Ποιο συνδυασμό τους θα προτείνατε τελικά;*

Στο μέρος αυτό θα αξιολογηθεί η επίδοση και η καταλληλότητα των διαφόρων ανιχνευτών και περιγραφητών σε ένα τυπικό πρόβλημα κατηγοριοποίησης εικόνων στη βάση Pascal VOC2005. Κάθε εικόνα ανήκει σε μια από τις τρεις ακόλουθες κλάσεις: αυτοκίνητο, άνθρωπος και ποδήλατο. Σκοπός είναι η κατηγοριοποίηση της κάθε εικόνας στη σωστή κλάση χρησιμοποιώντας σαν χαρακτηριστικά αναγνώρισης τους περιγραφητές που θα υπολογιστούν χρησιμοποιώντας τους ανιχνευτές που έχουν αναπτύξει στο Μέρος 2.

Αρχικά, εξάγονται τα χαρακτηριστικά (SIFT, SURF ή HOG) από το σύνολο των σημείων ενδιαφέροντος που έχει ανιχνευτεί με τους multi-scale ανιχνευτές σημείων ενδιαφέροντος. Για κάθε ζεύγος εικόνας-ετικέτας (I_k, y_k) εξάγονται τα features σε ένα πίνακα F_k διαστάσεων $n_k \times d$ όπου το n_k εξαρτάται από το πλήθος των ROIs που έχουν ανιχνευτεί και το d από το πλήθος των χαρακτηριστικών, ακριβώς όπως στο βήμα 2 του αλγορίθμου για το matching. Μετά, χωρίζεται το dataset σε train και test με τη βοήθεια της train_test_split που παρέχεται από την sklearn. Το μέγεθος του test dataset είναι 33% σε σχέση με το συνολικό μέγεθος. Στη συνέχεια πρόκειται να παραχθούν image embeddings χρησιμοποιώντας σαν είσοδο τα features. Για το σκοπό αυτό κατηγοριοποιούνται σε clusters κάποια από τα features σε διάφορες περιοχές της εικόνας και έπειτα μετράται η κανονικοποιημένη συχνότητα των τοπικών περιγραφητών κάθε εικόνας σε σχέση με τα κέντρα των clusters. Για το σκοπό αυτό, κατασκευάζεται ο πίνακας:

$$\mathbb{F} = \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_N \end{pmatrix}$$

που έχει όλα τα χαρακτηριστικά (vertically stacked) και μετά λαμβάνεται ένα δείγμα ⌈N/2⌉ γραμμών του πίνακα, έστω $\hat{F}$. Οι γραμμές του $\hat{F}$ χρησιμοποιούνται σαν είσοδος στον αλγόριθμο K-Means (που υπάρχει στη βιβλιοθήκη sklearn) για την εξαγωγή των κέντρων n_c clusters, έστω $z_1, \ldots, z_{nc}$. Στη συνέχεια για κάθε διάνυσμα χαρακτηριστικών υπολογίζεται η κατανομή:

$$N(k,j) = \left|\{f \in \text{row}(F_k) \mid j = \text{argmin}_{r=1,\ldots n_c}\{\|f - z_r\|_2\}\}\right|$$

και κανονικοποιείται με την L_2 νόρμα

$$x(k,j) = \frac{N(k,j)}{\left(\sum_{r=1}^{n_c} N^2(k,r)\right)^{1/2}}$$

Το διάνυσμα $x_k = (x(k, j))_{1\leq j\leq nc}$ αποτελεί την διανυσματική αναπαράσταση (embedding) της εικόνας I_k. Τέλος, τα δεδομένα εισέρχονται σε έναν ταξινομητή SVM για ταξινόμηση.

Παρακάτω αναγράφονται τα αποτελέσματα για τα διάφορα features και ανιχνευτές που δοκιμάστηκαν.

Detector	**Descriptor**	**Αριθμός Κέντρων n_c**	**Accuracy (%)**
Harris-Laplacian	*SURF*	50	68.1
Harris-Laplacian	*SIFT*	50	72.3
Harris-Laplacian	*HOG*	20	64.9
Hessian-Laplacian	*SURF*	50	55.1
Hessian-Laplacian	*SIFT*	50	74.2
Hessian-Laplacian	*HOG*	50	75.0

Σχολιασμός Αποτελεσμάτων:

- Παρατηρείται ότι η καλύτερη επίδοση προέκυψε για Hessian μέθοδο (blobs) για πολυκλιμακωτή ανίχνευση με χαρακτηριστικά HOG, ενώ σε παρόμοια επίδοση κυμαίνονταν αρκετές μέθοδοι ανίχνευσης όπως Harris-Laplacian με SIFT και Hessian-Laplacian με SIFT. Μια ερμηνεία για αυτό το αποτέλεσμα είναι ότι τα blobs αποτελούν τις περιοχές που διαφέρει πολύ η φωτεινότητα δίνοντας μια έννοια "κατεύθυνσης" στη γειτονιά και κατά συνέπεια πιο περιεκτικό ιστόγραμμα κατευθυνόμενων παραγώγων για αυτή τη γειτονιά.
- Δεν φαίνεται να υπάρχει μεγάλη διαφορά ανάμεσα στους συνδυασμούς περιγραφητή - ανιχνευτή. Γενικότερα, σύμφωνα με τα παραπάνω, η διαφορά με είναι πολύ μικρή. Όμως, επειδή υπάρχουν πολλού παράγοντες που επηρεάζουν το αποτέλεσμα, δεν είναι εφικτό να επιλεχθεί κάποιος συνδυασμός μεθόδων που να υπερέχει των άλλων. Ωστόσο, παρατηρείται ότι για τους δύο περιγραφητές η απόδοση των box filters είναι χειρότερη από τους άλλους δύο πολυκλιμακωτούς ανιχνευτές.

Βιβλιογραφία - Αναφορές

[1] Anubhav Agarwal, CV Jawahar, and PJ Narayanan. A survey of planar homography estimation techniques. Centre for Visual Information Technology, Tech. Rep. IIIT/TR/2005/12, 2005.

[2] Ezio Malis and Manuel Vargas. Deeper understanding of the homography decomposition for vision-based control. PhD thesis, INRIA, 2007.

[3] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In BMVC 2008-19th British Machine Vision Conference, pages 275–1. British Machine Vision Association, 2008.

[4] David Mumford, John Fogarty, and Frances Kirwan. Geometric invariant theory, volume 34. Springer Science & Business Media, 1994.

[5] Frazer K Noble. Comparison of opencv's feature detectors and feature matchers. In 2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP), pages 1–6. IEEE, 2016.

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In European conference on computer vision, pages 404–417. Springer, 2006.

[7] Chanop Silpa-Anan and Richard Hartley. Optimised kd-trees for fast image descriptor matching. In 2008 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8. IEEE, 2008

[8] Όραση Υπολογιστών – Maragos_CV_Book2014.pdf & παλαιότερο υλικό