Όραση Υπολογιστών

1η Εργαστηριακή Άσκηση

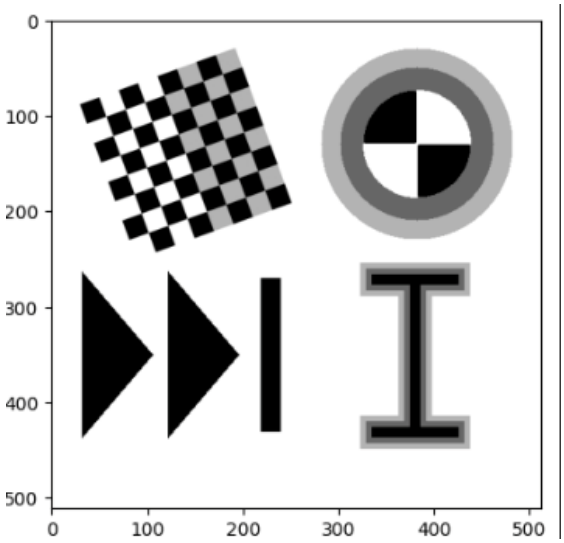
Ομαδα:
Δωροθέα Κουμίδου 03119712
Γιώργος Χαραλάμπους 03119706

Σχολή Ηλεκτρολόγων μηχανικών & Μηχανικών Υπολογιστών

ΜΕΡΟΣ 1ο : ΑΝΙΧΝΕΥΣΗ ΑΚΜΩΝ ΣΕ ΓΚΡΙΖΕΣ ΕΙΚΟΝΕΣ

1.1 Δημιουργία Εικόνων Εισόδου

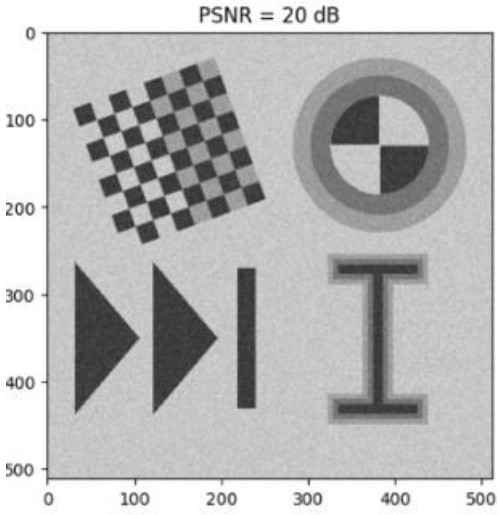
Η εικόνα I ("edgetest_24.png") της οποίας θα προσπαθήσουμε να βρούμε τις ακμές της φαίνεται παρακάτω:

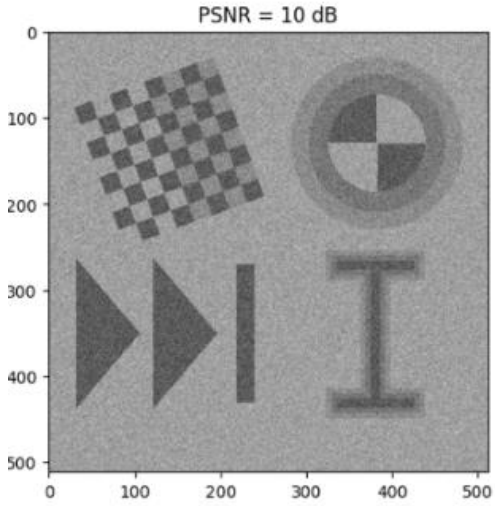


Η συνάρτηση **np.random.normal(),** προσθέτει θόρυβο με PSNR=20dB και 10dB, όπου η τυπική απόκλιση του λευκού θορύβου προκύπτει από τον τύπο που δίνεται στην οδηγία:

$$PSNR = 20\log_{10}\left(\frac{I_{max} - I_{\min}}{\sigma_n}\right)$$

Όπου $I_{\max}$, I_{min} η μέγιστη και ελάχιστη τιμή των pixels της εικόνας αντίστοιχα. Προσθέτοντας τον θόρυβο προκύπτουν οι εικόνες:





Παρατηρείται ότι μεγαλύτερο PSNR δίνει μια καθαρότερη εικόνα με λιγότερο θόρυβο, γιατί έχουμε μικρότερη τυπική απόκλιση και επομένως μικρότερο εύρος γύρω από την μέση τιμή της κατανομής.

1.2 Υλοποίηση Αλγορίθμου Ανίχνευσης Ακμών

Δημιουργούνται οι προσεγγίσεις των εξής κρουστικών αποκρίσεων:
A) Δισδιάστατή Gaussian
B) Laplacian-of-Gaussian (LoG)

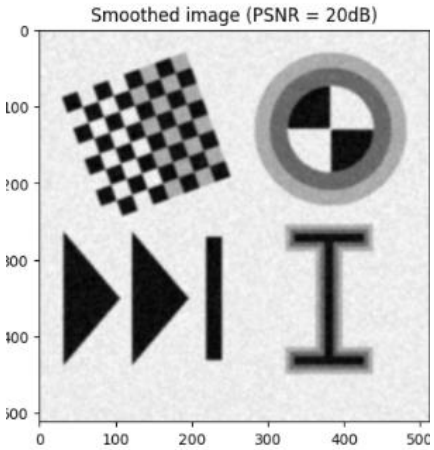
με την βοήθεια των οποίων προσεγγίζεται η Laplacian L της εξομαλυμένης εικόνας με 2 τρόπους, χρησιμοποιώντας τους πιο κάτω τύπους:
A) Γραμμική Laplacian: $L_1 = \nabla^2(G_\sigma * I) = (\nabla^2 G_\sigma) * I$

B) Μη-γραμμική Laplacian: $L_2 = (I\sigma \oplus B) + (I\sigma \ominus B) - 2I\sigma$

Όπου I=αρχική εικόνα

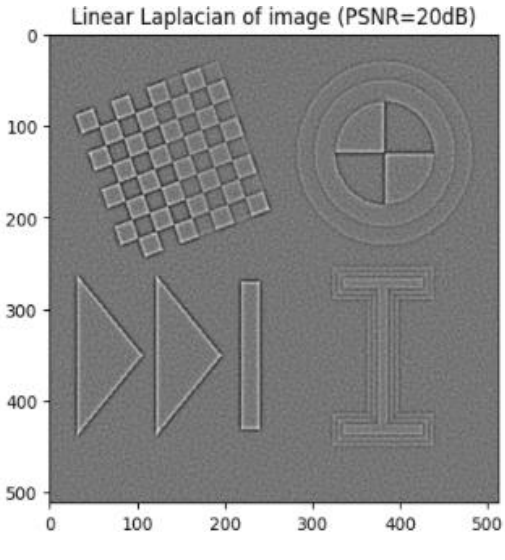
Iσ= εξομαλυμένη εικόνα

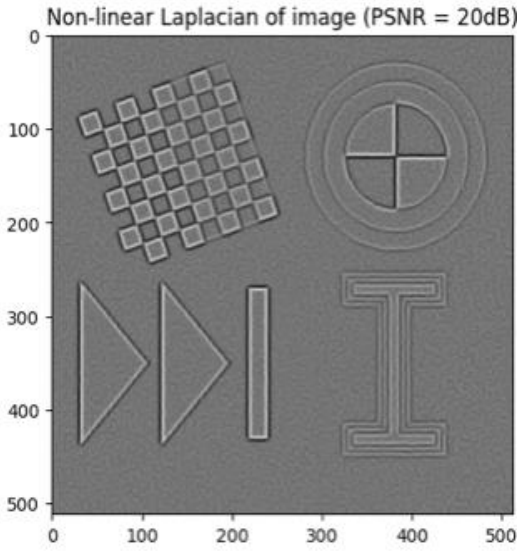
Εξομαλυμένη εικόνα γίνεται με την συνέλιξη της εικόνας αρχικής εικόνας και της δισδιάστατης Gaussian συνάρτησης



```python
#1.2.2
#The 2 types od Laplacian
L1 = cv2.filter2D(image1,-1,LoG(n,sigma))
plt.imshow(L1,cmap='gray')
plt.title("Linear Laplacian of image (PSNR=20dB)")
plt.show()

#Non-Linear (L2)
kernel = np.array([[0,1,0],
                   [1,1,1],
                   [0,1,0]], dtype=np.uint8)

smoothed_img = cv2.filter2D(image1,-1,G2D)

L2 = cv2.dilate(smoothed_img,kernel)+cv2.erode(smoothed_img,kernel)-2*smoothed_img

plt.imshow(L2,cmap='gray')
plt.title("Non-linear Laplacian of image (PSNR = 20dB)")
plt.show()

plt.imshow(smoothed_img,cmap='gray')
plt.title("Smoothed image (PSNR = 20dB)")
plt.show()
```

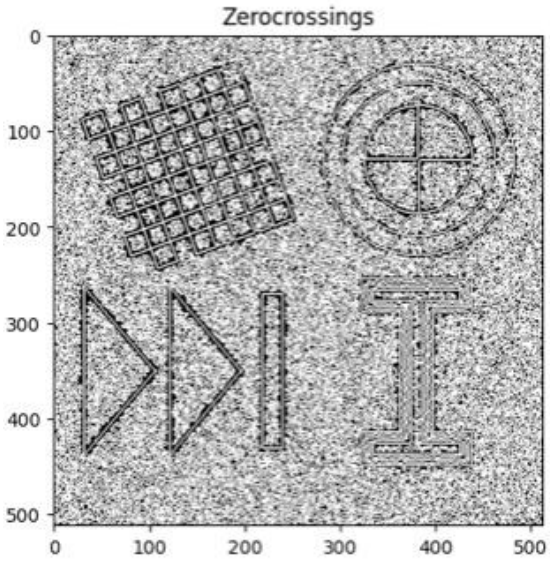
Επιλέγεται PSNR=20dB και εφαρμόζονται η γραμμική και μη γραμμική προσέγγιση της λαπλασιανής. Οι εξομαλυμένες εικόνες που προκύπτουν είναι οι εξής:





Ακολουθεί ο υπολογισμός των σημείων μηδενισμού (zerocrossings) της Laplacian. Για να το πετύχουμε αυτό, υπολογίζουμε την Δυαδική Εικόνα Προσήμου X = (L>=0), και στην συνέχεια το περίγραμμα Y του X: Y = (X $\oplus$ B) − (X $\ominus$ B).

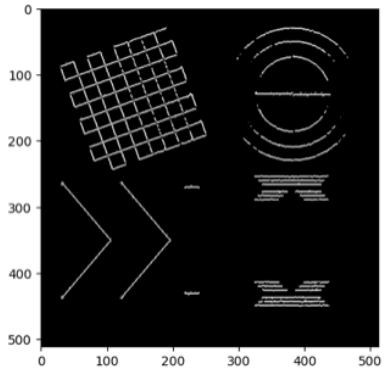
Το αποτέλεσμα του υπολογισμού zerocrossing στις παραπάνω εικόνες είναι το εξής:

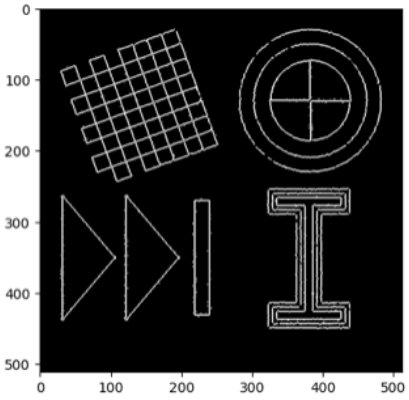


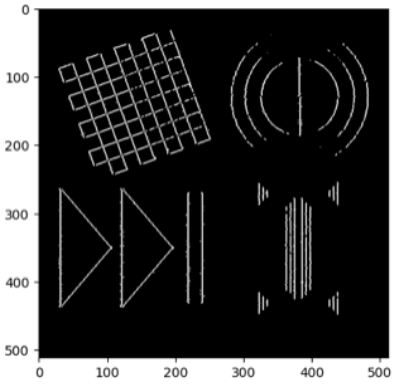
Τέλος, απορρίπτονται τα zerocrossings σε σχετικά ομαλές περιοχές, δηλαδή δεν ικανοποιούν το παρακάτω κριτήριο:

$$Y[i,j]= 1 \text{και } \|\nabla I\sigma[i,j]\| > \theta edge \cdot \max_{x,y} \|\nabla I\sigma\|$$

Η παραπάνω διαδικασία έγινε για άξονα χ και y ξεχωριστά και μετά με το λογικό or παίρνουμε το αποτέλεσμα (1 στα pixels ακμών,0 αλλού):

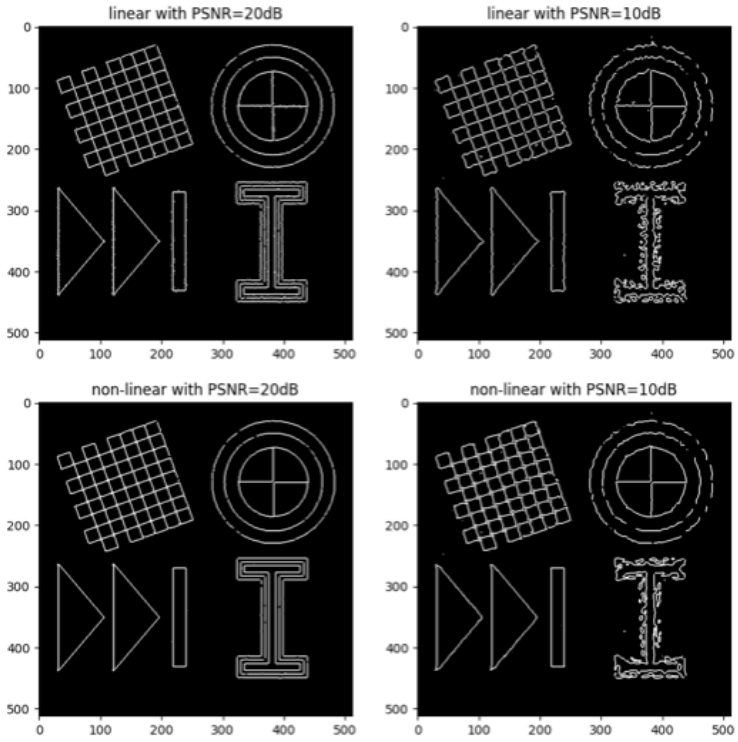






Συνοψίζοντας, δημιουργείται η συνάρτηση Edge_detect με παραμέτρους (εικόνα εισόδου, σ, θ, τύπος προσέγγισης της Laplace) και δίνει έξοδο μια δυαδική εικόνα με τις ανιχνευόμενες ακμές.

Οι παρακάτω εικόνες απεικονίζουν την έξοδο για γραμμικό και μη φίλτρο, καθώς και για διαφορετική τιμή θορύβου.



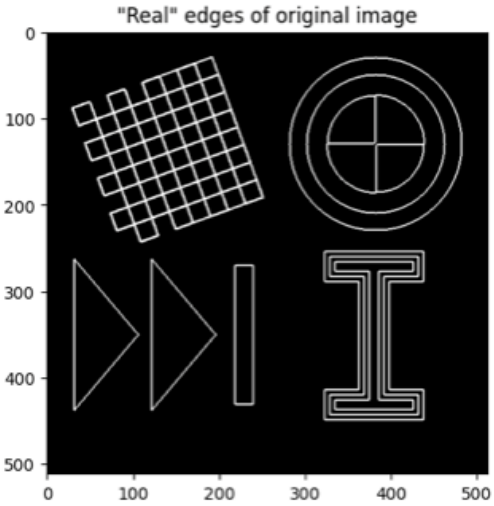
1.3 Αξιολόγηση Αποτελεσμάτων Ανίχνευσης Ακμών:

Για την αξιολόγηση της ποιότητας εντοπισμού ακμών στις εικόνες με θόρυβο, πρέπει να εντοπιστούν οι ακμές στην αρχική φωτογραφία, χρησιμοποιώντας τον παρακάτω τελεστή:

$$M = (I \oplus B) - (I \ominus B)$$

Το μορφολογικό φίλτρο dilate εξομαλύνει την εικόνα ‘γεμίζοντας’ χάσματα και κενά, με αποτέλεσμα η εικόνα να ‘διογκώνεται’.
Αντίθετα το erode εξομαλύνει την εικόνα αφαιρώντας pixels που έχουν μεταξύ τους κενά, με αποτέλεσμα η εικόνα να ‘συρρικνώνεται’.
Η αφαίρεση των αποτελεσμάτων των 2 αυτών φίλτρων (της διογκωμένης και συρρικνωμένης εικόνας) αφήνει τα pixels της διογκωμένης εικόνας που δεν μοιράζεται με την συρρικνωμένη, το οποίο είναι καλή προσέγγιση των ακμών της πραγματικής εικόνας.

Τέλος εφαρμόζοντας ένα threshold στα εναπομείναντα pixels, καταλήγουμε σε μία δυαδική εικόνα που προσεγγίζει πολύ καλά τις πραγματικές ακμές.
Η υλοποίηση της παραπάνω διαδικασίας επιτυγχάνεται με την συνάρτηση real_edges() με threshold=0.1:



Για τον καθορισμό της ποιότητας, υπολογίζουμε την ποσότητα

$$C = [Pr(D|T) + Pr(T|D)] /2$$

Pr(D|T):= ποσοστό ανιχνευθεισών ακμών που είναι αληθινές (precision)

Pr(T|D):= ποσοστό αληθινών ακμών που ανιχνεύθησαν (recall)

Μια πρώτη αξιολόγηση της ποιότητας για τα αρχικά δεδομένα, δίνουν τα εξής αποτελέσματα:

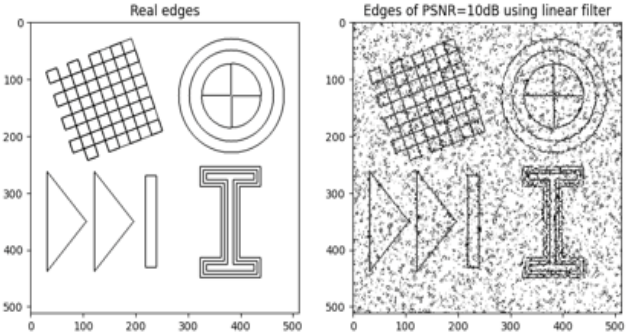
```
edges1 = EdgeDetect(image1,1.5,0.2,'non-linear') #Edges using PSNR=20dB
edges2 = EdgeDetect(image2,3,0.2,'non-linear')  #Edges using PSNR=10dB
```

```
Η ποσοτική αξιολόγηση για PSNR=10dB είναι: 0.6622335423213609
```

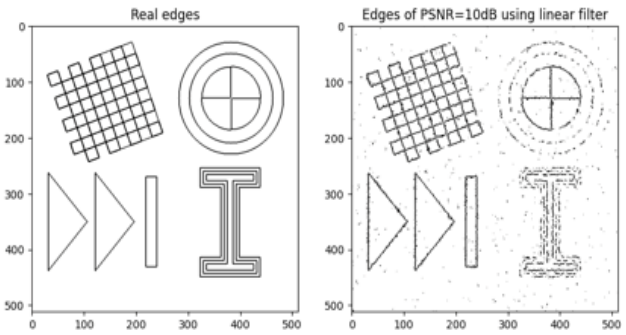
```
Η ποσοτική αξιολόγηση για PSNR=20dB είναι: 0.9207868837154622
```

Παρατίθενται μερικά παραδείγματα καθορισμού της ποιότητας για διιαφορετικές τιμές τυπικής απόκλισης σ [1.5 , 2 , 2.5 ,3] , PSNR [10 ,20] , θ [0.15 , 0.2, 0.25] και filter[linear , non-linear].

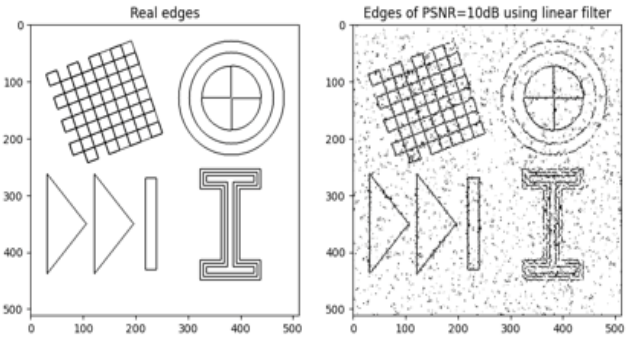
Edge detection of quality=0.52, sigma=1.5, theta edge=0.15



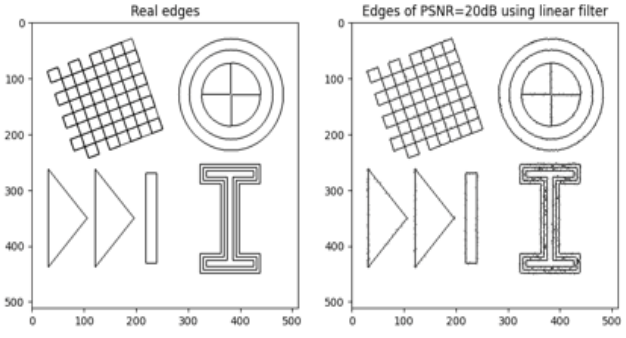
Edge detection of quality=0.674, sigma=1.5, theta edge=0.25



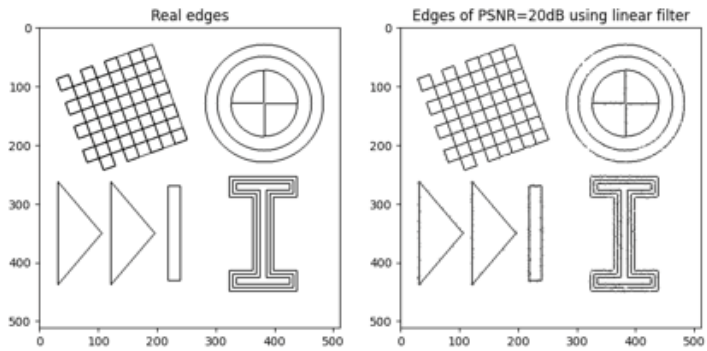
Edge detection of quality=0.605, sigma=1.5, theta edge=0.2



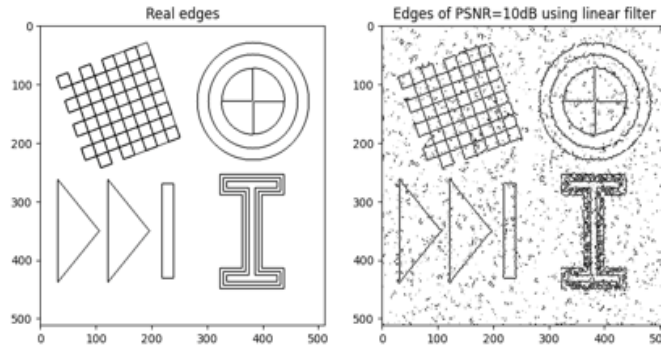
Edge detection of quality=0.884, sigma=1.5, theta edge=0.15



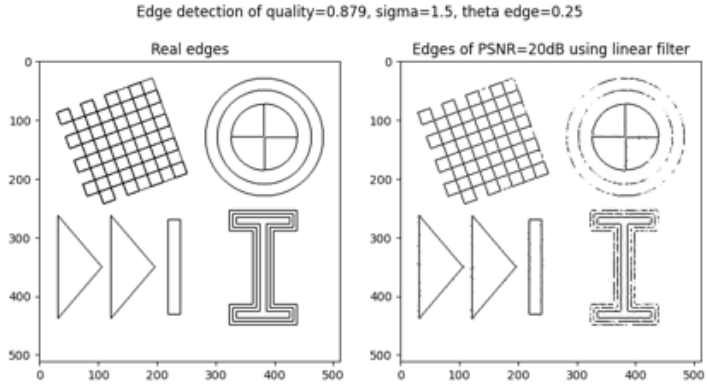
Edge detection of quality=0.909, sigma=1.5, theta edge=0.2



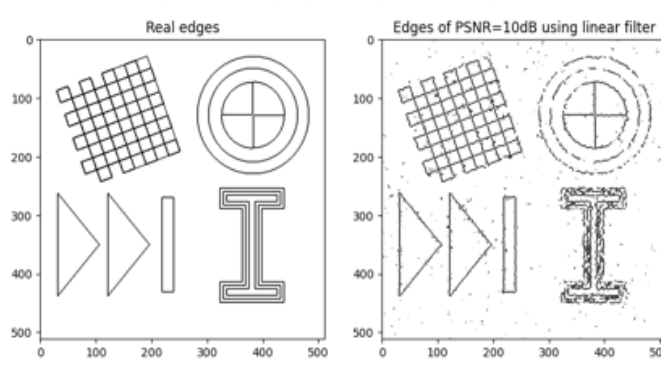
Edge detection of quality=0.593, sigma=2, theta edge=0.15



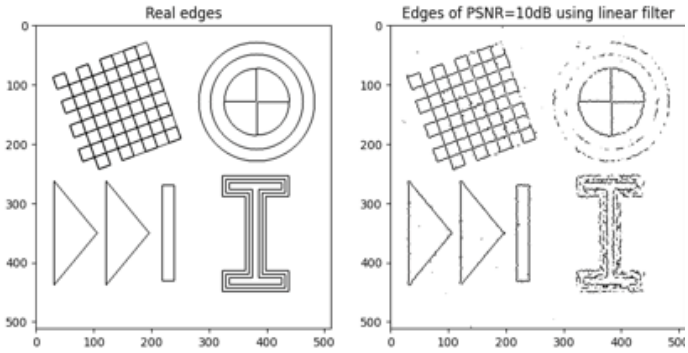
Edge detection of quality=0.879, sigma=1.5, theta edge=0.25



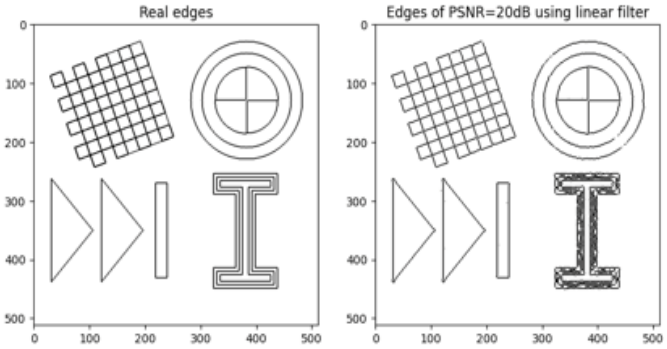
Edge detection of quality=0.68, sigma=2, theta edge=0.2



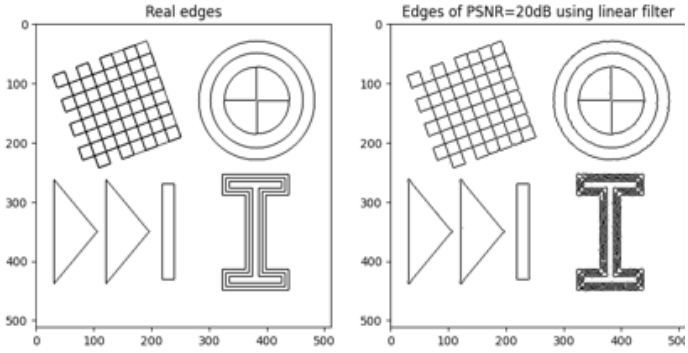
Edge detection of quality=0.707, sigma=2, theta edge=0.25



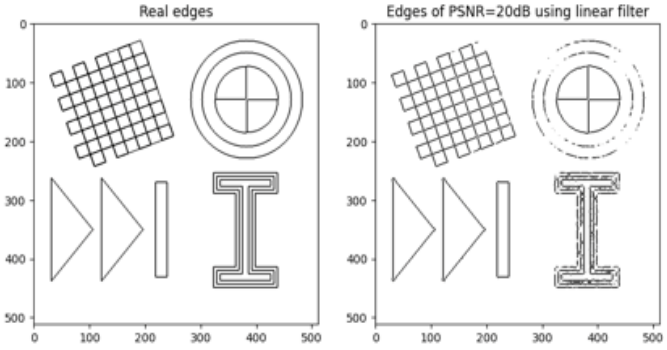
Edge detection of quality=0.845, sigma=2, theta edge=0.2



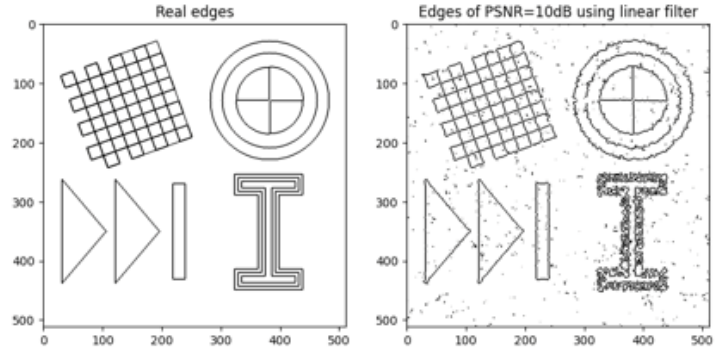
Edge detection of quality=0.831, sigma=2, theta edge=0.15



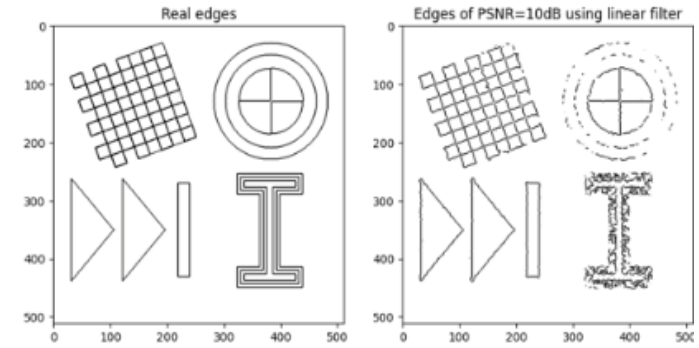
Edge detection of quality=0.867, sigma=2, theta edge=0.25



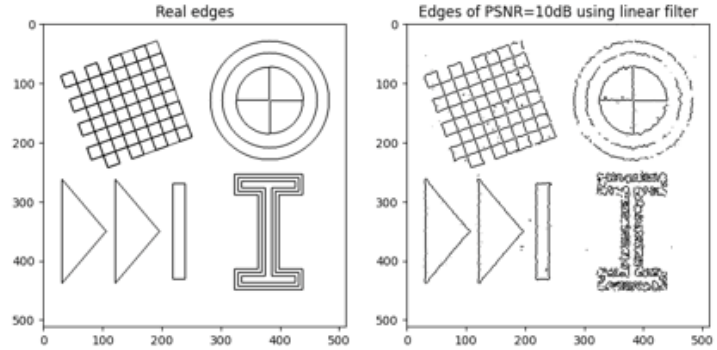
Edge detection of quality=0.631, sigma=2.5, theta edge=0.15



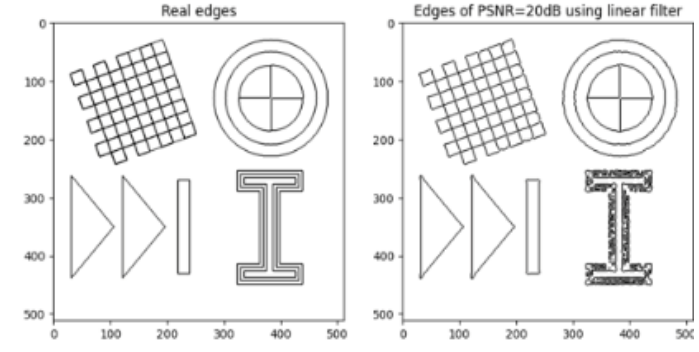
Edge detection of quality=0.672, sigma=2.5, theta edge=0.25



Edge detection of quality=0.673, sigma=2.5, theta edge=0.2

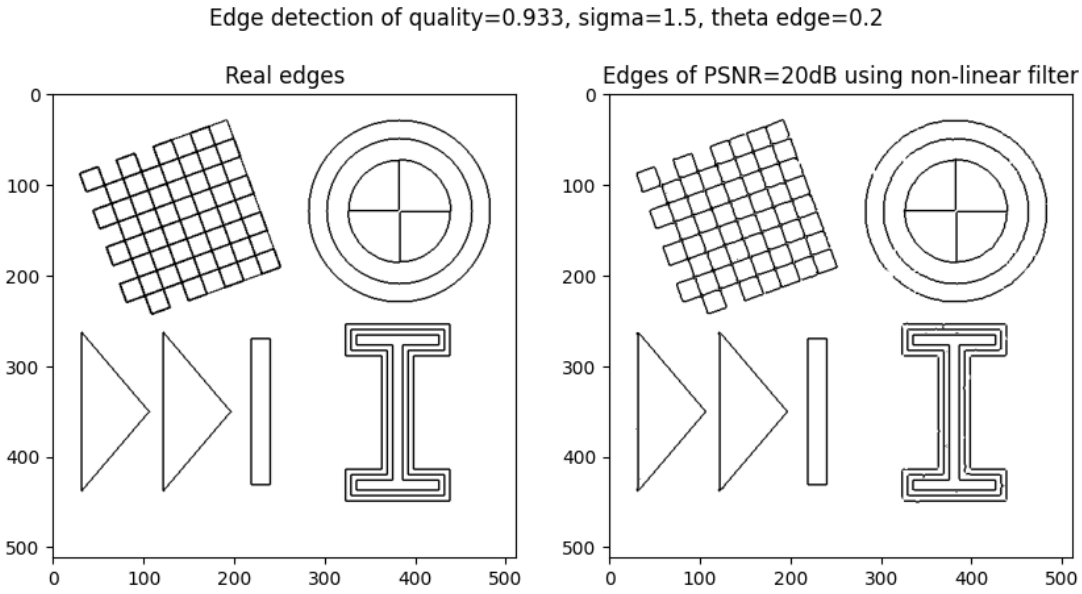


Edge detection of quality=0.798, sigma=2.5, theta edge=0.15



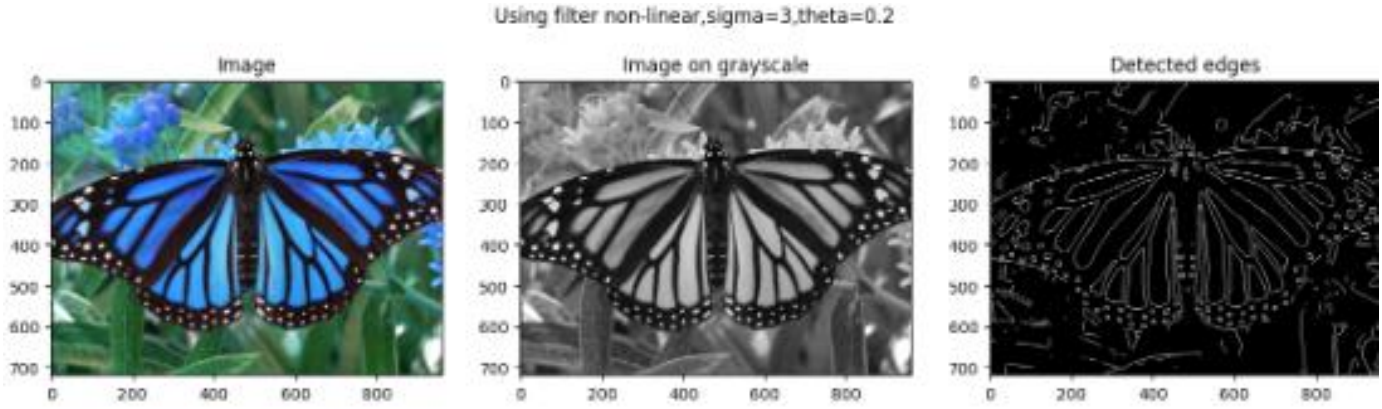
Παρατηρείται ότι η αύξηση της τυπικής απόκλισης σ, μειώνει σημαντικά τον θόρυβο της τελικής εικόνας, συνεπώς οδηγεί με υψηλή ποιότητας ανίχνευση ακμών. Επίσης φαίνεται ότι η χρήση της μη-γραμμικής προσέγγισης της Laplacian δίνει καλύτερα αποτελέσματα από την γραμμική.

Το καλύτερο αποτέλεσμά μας είναι το παρακάτω:



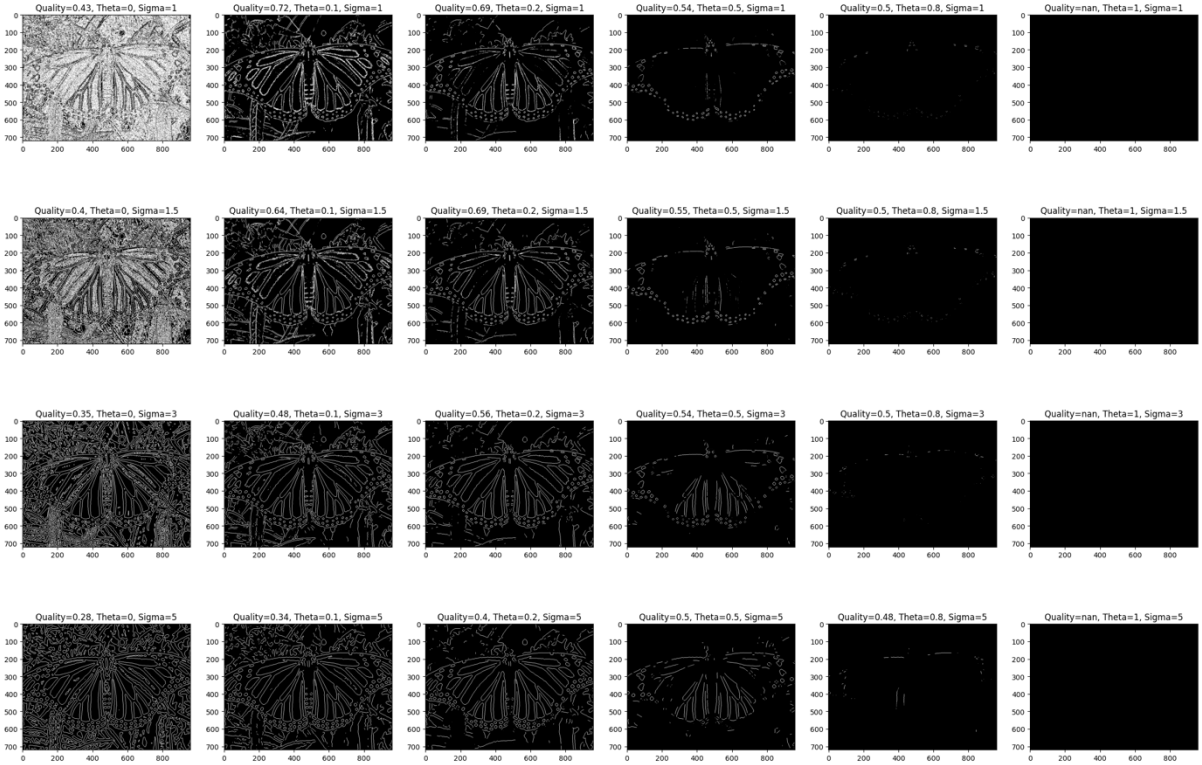
1.4 Εφαρμογή σε πραγματικές εικόνες:

Εφαρμόζεται η συνάρτηση EdgeDetect() στην ασπρόμαυρη εκδοχή της εικόνα ‘butterfly.jpg’.
Τα αποτελέσματα είναι τα παρακάτω:

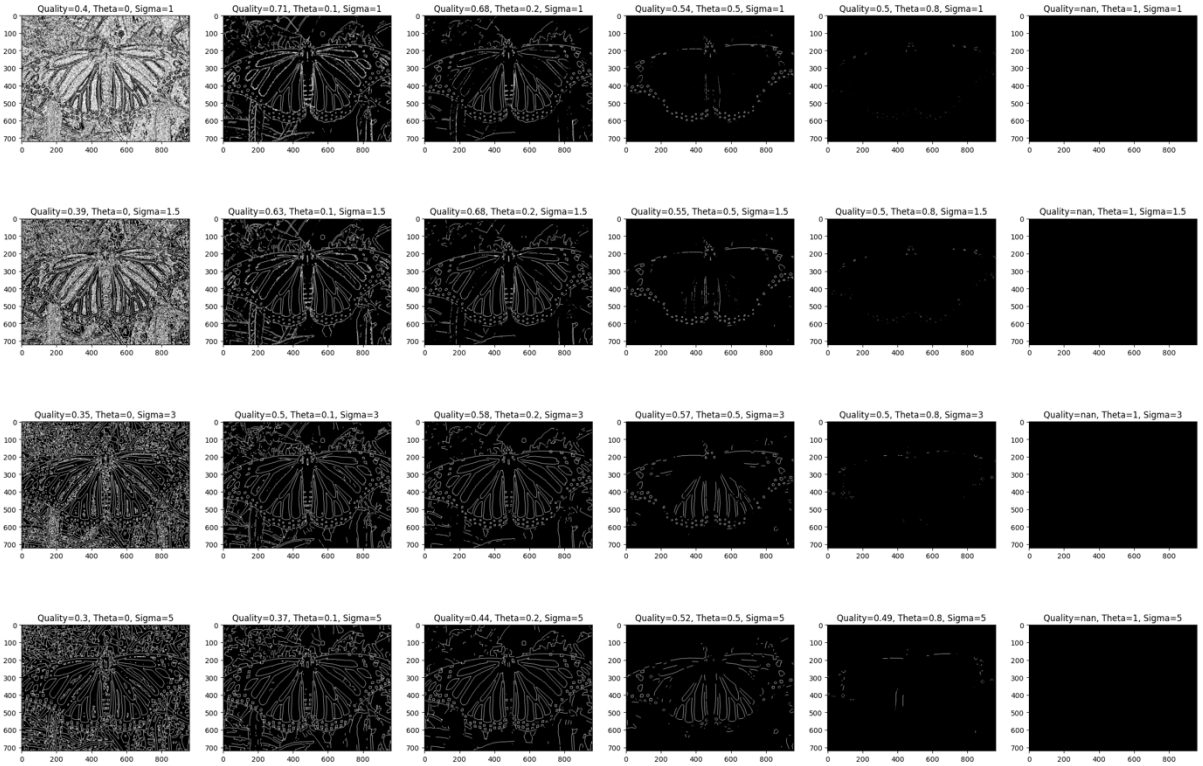


Γίνονται διάφορες δοκιμές για διαφορετικές τιμές σ, threshold και φίλτρα.

Μη γραμμικά φίλτρα



Γραμμικά Φίλτρα

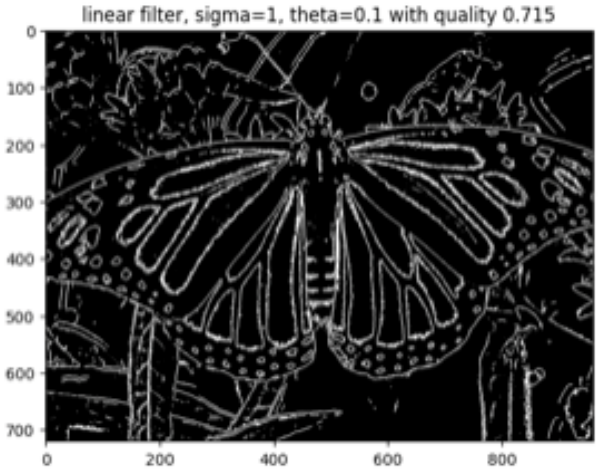


Κάνοντας μια ποιοτική ανάλυση των παραπάνω, παρατηρείται ότι η αύξηση του threshold οδηγεί σε λιγότερη χρήσιμη πληροφορία, καθώς το πλήθος των επιλεγμένων pixels μειώνεται, ενώ για πολύ μικρά thresholds απομένει πολύς θόρυβος στη εικόνα.

Επίσης για πολύ μικρά σ το παράθυρο για το εφαρμοζόμενο φίλτρο είναι αρκετά μικρό, συνεπώς στην εξομαλυμένη εικόνα απομένει ‘άχρηστη’ πληροφορία που την καθιστά δυσανάγνωστη.
Από την άλλη, τα μεγάλα σ οδηγούν σε μεγάλα παράθυρα τα οποία αγνοούν χρήσιμη πληροφορία.

Η χρήση της γραμμικής προσέγγισης της Laplacian επιτυγχάνει καλύτερα αποτελέσματα από την μη γραμμική, καθώς δίνει καλύτερη εξομάλυνση της εικόνας στα αρχικά στάδια της επεξεργασίας.

Για την επιλογή της καλύτερης εικόνας χρησιμοποιήθηκε εντολή if όπου κρατούσε την καλύτερη εικόνα. Παρακάτω φαίνεται το αποτέλεσμα:



ΜΕΡΟΣ 2ο: ΑΝΙΧΝΕΥΣΗ ΣΗΜΕΙΩΝ ΕΝΔΙΑΦΕΡΟΝΤΟΣ

2.1 Ανίχνευση Γωνιών

Για την ανίχνευση των γωνιών στις φωτογραφίες, πρέπει να υπολογιστούν οι ιδιοτιμές λ1 και λ2 (μία για κάθε κατεύθυνση) της κάθε εικόνας και συγκρίνονται οι τιμές μεταξύ τους.

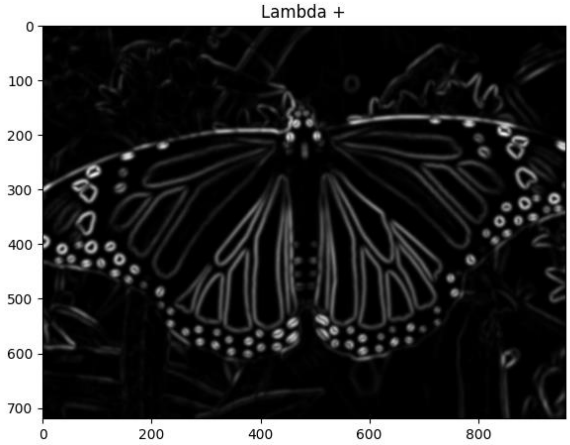
Για τον υπολογισμό, βρίσκονται τα J_1, J_2, J_3 του τανυστή J:

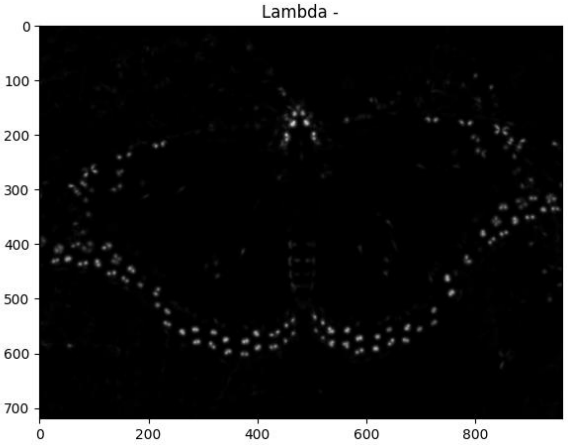
$$J_1(x,y) = G_\rho * \left(\frac{\partial I_\sigma}{\partial x} \cdot \frac{\partial I_\sigma}{\partial x} \right)(x,y)$$
$$J_2(x,y) = G_\rho * \left(\frac{\partial I_\sigma}{\partial x} \cdot \frac{\partial I_\sigma}{\partial y} \right)(x,y)$$
$$J_3(x,y) = G_\rho * \left(\frac{\partial I_\sigma}{\partial y} \cdot \frac{\partial I_\sigma}{\partial y} \right)(x,y)$$

Έπειτα υπολογίζεται τα λ1, λ2 σύμφωνα με την σχέση:

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

Οπτικοποιώντας τα λ1, λ2 για την butterfly.png προκύπτουν τα εξής:





Από θεωρία είναι γνωστό ότι τα σημεία όπου τα λ1 και λ2 έχουν υψηλές και όμοιες τιμές είναι σημείο γωνιάς, ενώ οποιασδήποτε άλλος συνδυασμός των λ1 και λ2 είναι για ακμές ή αδιάφορες περιοχές (flat regions).

Παρατηρείται ότι οι ιδιοτιμές λ1 (λ+) αντιστοιχούν τόσο σε ακμές όσο και σε γωνίες, ενώ οι λ2 (λ-) μονάχα σε γωνίες. Αυτό προκύπτει από την θεωρία, καθώς στις ακμές η μία ιδιοτιμή είναι μεγαλύτερη της άλλης (συνεπώς είναι θετική και ανήκει στα λ+) ενώ μονάχα στα γωνιακά σημεία τα λ+ και λ- έχουν την ίδια τιμή (όπως φαίνεται από τις δύο εικόνες, όπου υπάρχουν κάποια κοινά σημεία).

Υπολογίζοντας το κριτήριο γωνιότητας R:

$$R(x,y) = \lambda_- \lambda_+ - k \cdot (\lambda_- + \lambda_+)^2$$

και εφαρμόζοντας τις συνθήκες Σ1 και Σ2 για τον εντοπισμό των pixels-γωνιών, φαίνονται τα πιο κάτω αποτελέσματα:

Τα παραπάνω αποτελέσματα προκύπτουν για σ=2, p=2.5, k=0.05, theta_corn=0.05 με τη χρήση της συνάρτησης harris_stevens_detector().

Παρατηρείται ότι ο αλγόριθμος βρίσκει αρκετές γωνίες, παρόλα αυτά το αποτέλεσμα δεν είναι βέλτιστο.

2.2 Πολυκλιμακωτή ανίχνευση γωνιών

Για την επίτευξη καλύτερων αποτελεσμάτων στην ανίχνευση γωνιών, μπορούμε να δοκιμάσουμε την μέθοδο που περιγράψαμε στο 2.1 (Harris-Stephens Method) για πολλές τιμές σ,p και να επιλέξουμε μεταξύ των αποτελεσμάτων τους. Για αυτό υπολογίζουμε N διαφορετικά σ,p, τα οποία προκύπτουν από τα αρχικά επί κάποιο παράγοντα κλιμάκωσης s, και εφαρμόζουμε την harris_ stephens_detector() για κάθε ζεύγος σ,p.

(Η διαδικασία αυτή υλοποιείται από την συνάρτηση find_all_corners())

Για την επιλογή των κατάλληλων σημείων ,αρχικά υπολογίζεται η κανονικοποιημένη LoG (Laplacian-of-Gaussian) για κάθε σ όπως περιγράφεται από:

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

και από τα σημεία που προκύπτουν, επιλέγονται αυτά που μεγιστοποιούν την $LoG(x, y, \sigma_i)$ σε μια γειτονιά 2 διαδοχικών κλιμάκων , δηλαδή είναι τοπικό μέγιστο.
(Η διαδικασία αυτή υλοποιείται από τις συναρτήσεις normalize_LoG(), find_all_LoG() και harris_laplace_detector()).

Εφαρμόζοντας την διαδικασία στις εικόνες με αρχικοποίηση σ_0=2, ρ_0=2.5, N=4, s=1.5, προκύπτουν:

Παρατηρείται ότι στις φωτογραφίες όπου οι γωνίες δεν είναι τόσο ευδιάκρατικες η Harris-Laplacian βρίσκει περισσότερα σημεία από την Harris-Stephens.

2.3 Ανίχνευση blobs

Ως 'blobs' ορίζονται περιοχές με ομοιογένεια που διαφέρουν σημαντικά από την γειτονιά τους. Για την ανίχνευσή τους, υπολογίζεται ο πίνακας Hessian:

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

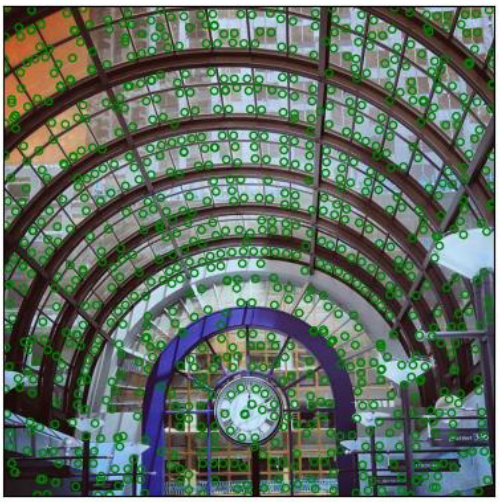
όπου $L_{xx}(x,y,\sigma) = \frac{\partial^2}{\partial x^2}\{I_\sigma(x,y)\}$, $L_{yy}(x,y,\sigma) = \frac{\partial^2}{\partial y^2}\{I_\sigma(x,y)\}$ και $L_{xy}(x,y,\sigma) = \frac{\partial^2}{\partial x \partial y}\{I_\sigma(x,y)\}$.

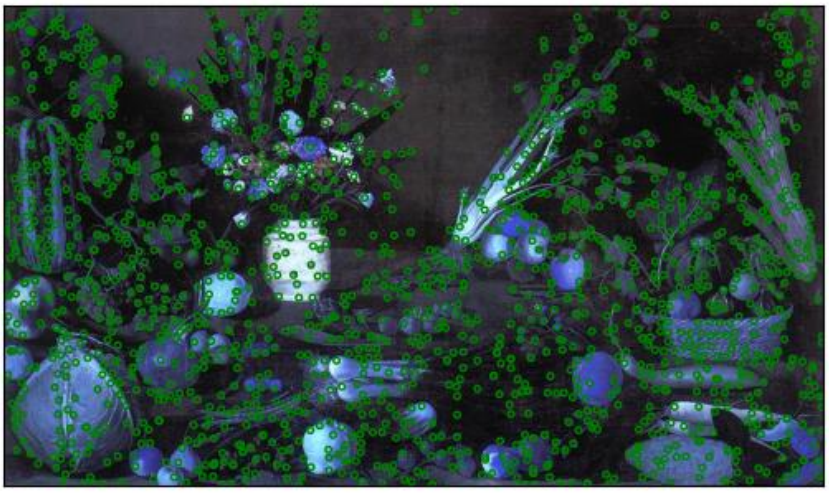
Ως κριτήριο γωνιότητας παίρνουμε το

$$R = \det(H) = Lxx * Lyy - (Lxy)^2$$

και εφαρμόζονται τα κριτήρια Σ1 και Σ2 του ερωτήματος 2.1

Με χρήση της συνάρτησης hessian_detector() υλοποιούνται τα πιο πάνω και τα αποτελέσματα αυτών είναι οι παρακάτω εικόνες με παραμέτρους σ=1.5, theta=0.005:

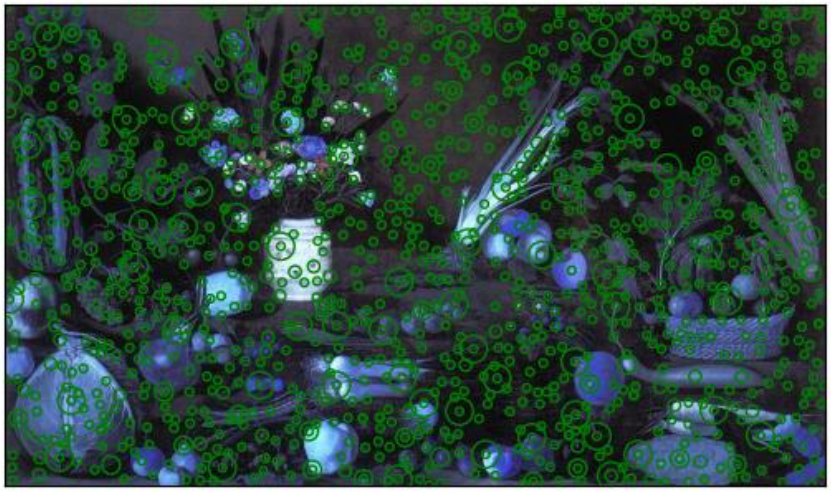




Παρατηρείται ότι η συγκεκριμένη μέθοδος ανίχνευσης έχει καλύτερα αποτελέσματα από τις προηγούμενες αφού έχει αισθητά περισσότερα σημεία ενδιαφέροντος, ευδιάκριτα ή μη.

2.4 Πολυκλιμακωτή Ανίχνευση Blobs

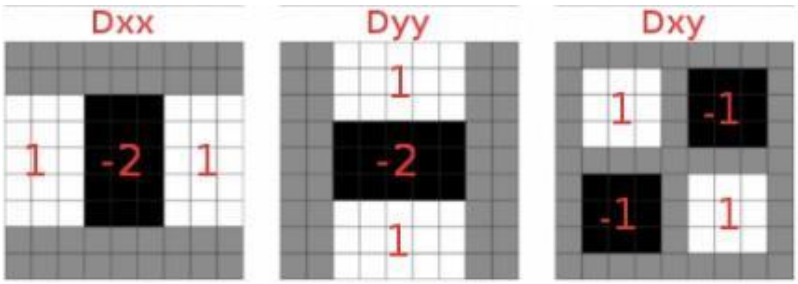
Ακολουθώντας την ίδια διαδικασία με το ερώτημα 2.2 και την συνάρτηση hessian_laplace_detector(), βρίσκονται τα blobs για διάφορες κλίμακες με αρχικοποίηση παραμέτρων σ_o=2, s=1.5, N=4, θ=0.005 και εφαρμόζοντας το κριτήριου του τοπικού μεγίστου στα διαδοχικά LoGs, προκύπτουν τα παρακάτω αποτελέσματα:

Όπως ήταν αναμενόμενο η πολυκλιμακωτή ανίχνευση έχει ακόμα καλύτερο αποτέλεσμα.

2.5 Επιτάχυνση με Χρήση Box Filters και Ολοκληρωτικών Εικόνων

Η πολυκλιμακωτή ανίχνευση blobs απαιτεί μεγάλο υπολογιστικό κόστος, εξαιτίας των πολλαπλών συνελίξεων για διάφορα σ. Για αυτό το λόγο προσεγγίζουμε τους συντελεστές Lxx,Lyy,Lxy της Hessian μήτρας μέσω των box filters Dxx, Dyy, Dxy και των ολοκληρωτικών εικόνων.



Με τη βοήθεια του Παραρτήματος Α, γίνεται η υλοποίηση των box filters με τα αντίστοιχα μεγέθη:

Dxx: ύψος παραθύρου 4 × floor(n/6) + 1, πλάτος παραθύρου 2 × floor(n/6) + 1

Dyy : ύψος παραθύρου 2 × f loor(n/6) + 1, πλάτος παραθύρου 4 × f loor(n/6) + 1

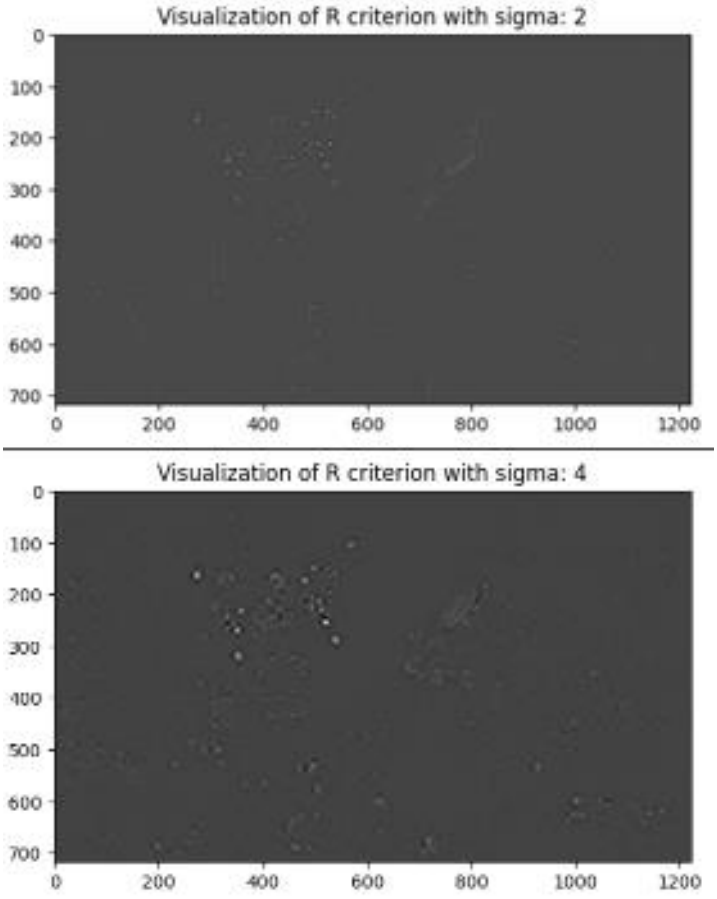
Dxy : ύψος παραθύρου 2 × f loor(n/6) + 1, πλάτος παραθύρου 2 × f loor(n/6) + 1

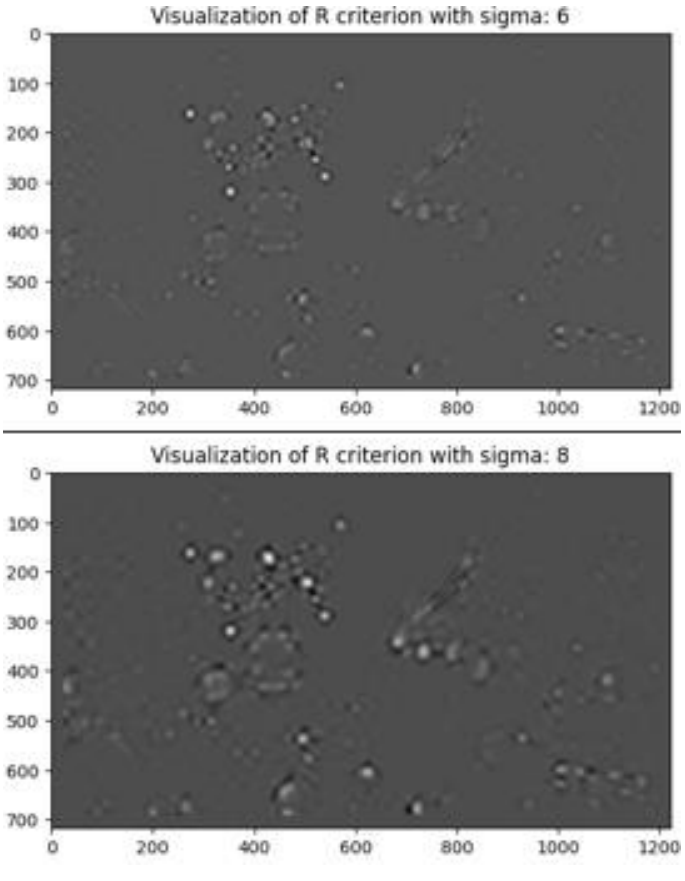
Έπειτα υπολογίζεται το κριτήριο γωνιότητας:

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

Και παρατίθενται οπτικοποιήσεις του R για διαφορετικές τιμές σ.

Παρακάτω οι οπτικοποιήσεις του R για την εικόνα Caravaggio.jpg με σίγμα 2,4,6 και 8:

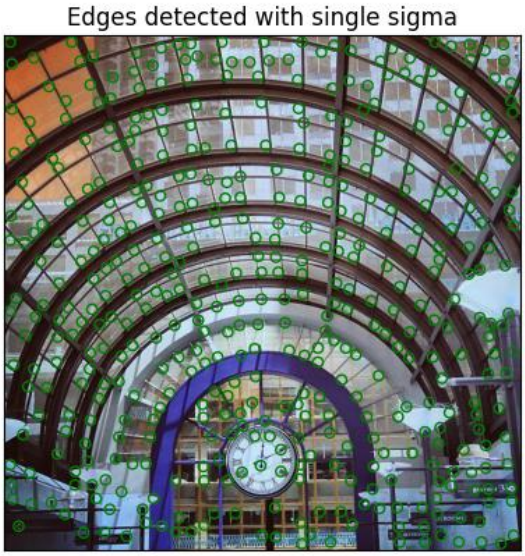


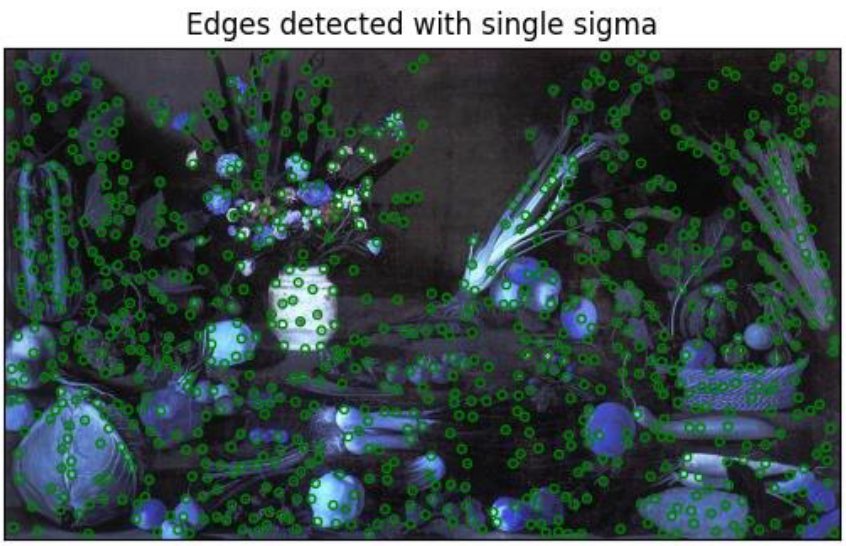


Στο notebook γίνεται οπτικοποίηση του R και για την άλλη εικόνα.

Η αύξηση της τυπικής απόκλισης σ, αλλοιώνει την ποιότητα της εικόνας και ο εντοπισμός των γωνιακών σημείων γίνεται δυσκολότερος.

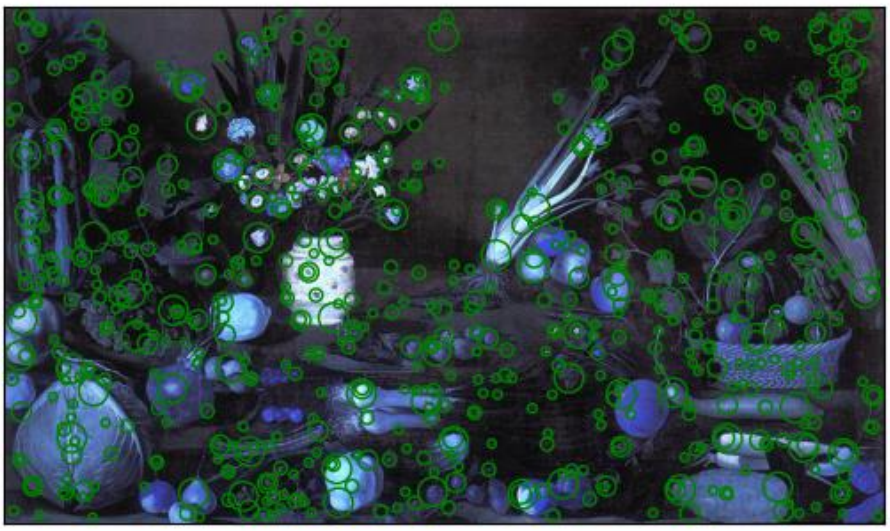
Με τη χρήση της συνάρτησης hessian_aprox(), οι γωνίες που προκύπτουν για σ=2 φαίνονται πιο κάτω:





Η προσέγγιση της hessian δίνει ένα εξίσου σημαντικό αποτέλεσμα .

Όπως και στα προηγούμενα ερωτήματα διερευνάται και η πολυκλιμακωτή ανίχνευση με box filters που επιβεβαιώνει τα συμπεράσματα των προηγουμένων αντίστοιχων ερωτημάτων. Η συνάρτηση box_multiscale() δίνει τα ακόλουθα αποτελέσματα.



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

3.1 Ταίριασμα Εικόνων υπό Περιστροφή και Αλλαγή Κλίμακας

Εξετάζεται η ικανότητα εύρεσης της περιστροφής και της κλίμακας με την χρήση των σημείων ενδιαφέροντος (χρησιμοποιώντας τις συναρτήσεις του προηγούμενου μέρους) και τοπικών περιγραφητών που βρίσκονται στο αρχείο *cv24_lab1_part3_utils.pyc*.

Χρησιμοποιώντας το παράδειγμα από το test_matching_evluation.py, υπολογίζουμε για κάθε περιγραφητή και ανιχνευτή το μέσο σφάλμα εκτίμησης κλίμακας και γωνίας. Οι ανιχνευτές που χρησιμοποιήθηκαν είναι αυτοί των ερωτημάτων 2.1-2.5 και στην περίπτωση των box_filters επιλέγεται η πολυκλιμακωτή ανάλυση.

Έχοντας δημιουργήσει ανώνυμες συναρτήσεις ως lambda functions και για τις 5 συναρτήσεις μας, προκύπτουν τα παρακάτω αποτελέσματα για την χρήση των συναρτήσεων με SURF:

Results for Harris Detector and SURF:
Avg. Scale Error for Image 1: 0.003316394599950531
Avg. Theta Error for Image 1: 0.17663001260578187

Results for Hessian Detector and SURF:
Avg. Scale Error for Image 1: 0.019927129310211854
Avg. Theta Error for Image 1: 8.697976338073397

Results for Multiscale Hessian Detector and SURF:
Avg. Scale Error for Image 1: 0.05616494049951524
Avg. Theta Error for Image 1: 3.017101439784951

Results for Multiscale Box and SURF:
Avg. Scale Error for Image 1: 0.0021564274661461645
Avg. Theta Error for Image 1: 0.13592237796969459

Αντίστοιχα, για τη χρήση των συναρτήσεων με HOG προκύπτουν τα παρακάτω αποτελέσματα:

Results for Harris Detector and HOG:
Avg. Scale Error for Image 1: 0.1344098100010664
Avg. Theta Error for Image 1: 14.114276263215611

Results for Harris Detector and HOG:
Avg. Scale Error for Image 1: 0.21056864504286205
Avg. Theta Error for Image 1: 17.7030804509519

Results for Multiscale Hessian Detector and HOG:
Avg. Scale Error for Image 1: 0.27495909290313536
Avg. Theta Error for Image 1: 19.500789543680042

Results for Multiscale Box and HOG:
Avg. Scale Error for Image 1: 0.10708768877401259
Avg. Theta Error for Image 1: 8.426506599288679

Παρατηρείται ότι ο τοπικός περιγραφητής SURF δίνει προβλέψεις με μικρότερο σφάλμα κλίμακας και γωνίας από τον HOG περιγραφητή. Επιπλέον από τους ανιχνευτές που χρησιμοποιήθηκαν η πολυκλιμακωτή ανίχνευση με box_filters είναι η πιο αποτελεσματική. Η ανίχνευση με χρήση blobs(πολυκλιμακωτή και μη) έχει μεγάλο σφάλμα και στις 2 περιπτώσεις, γι'αυτό και δεν προτείνεται ως ανιχνευτής. Ο έλεγχος του ανιχνευτή harris_laplace δεν περιλαμβάνεται, διότι απαιτεί αρκετό χρόνο.

3.2 Κατηγοριοποίηση Εικόνων

Η αξιολόγηση της επίδοσης(accuracy) των παραπάνω ανιχνευτών θα γίνει μέσω ταξινόμησης εικόνων σε 3 κλάσεις: αυτοκίνητο, ποδήλατο, άνθρωπος. Επιλέγονται μόνο οι πολυκλιμακωτοί ανιχνευτές και ακολουθείται η διαδικασία σύμφωνα με τον δοθέντα κώδικα *example_classification.py*.

Αρχικά με τη συνάρτηση FeatureExtraction() εξαγάγονται τα χαρακτηριστικά για το αντίστοιχο ζεύγος περιγραφητή-ανιχνευτή για όλη τη βάση δεδομένων.

Μετά, γίνεται ο διαχωρισμός σε train και test set μέσω της συνάρτησης createTrainTest και τέλος η ταξινόμηση γίνεται με τη χρήση των συναρτήσεων BagOfWords() και svm().

Τα αποτελέσματα με τη χρήση του SURF είναι τα εξής:

Mean accuracy for Hessian-Laplace with SURF descriptors: 48.828%

Mean accuracy for Multiscale Box with SURF descriptors: 57.793%

Αντίστοιχα, για το HOG:

Mean accuracy for Hessian-Laplace with HOG descriptors: 57.379%

Mean accuracy for Multiscale Box with HOG descriptors: 65.379%

Παρατηρείται ότι η χρήση του περιγραφητή HOG δίνει καλύτερη επίδοση στο μοντέλο σε σύγκριση με το SURF. Επιπλέον η ανίχνευση με τη χρήση multiscale box filters είναι και πάλι η αποτελεσματικότερη. Αξίζει να σημειωθεί ότι, η εξαγωγή χαρακτηριστικών με τον ανιχνευτή hessian_laplace() χρειάζεται περίπου 8 λεπτά, ενώ ο multiscale_box() χρειάζεται περίπου 1 ώρα με πολύ καλύτερη επίδοση. Ο ανιχνευτής harris_laplace() είναι αρκετά χρονοβόρος και δεν προτείνεται(εξαγωγή χαρακτηριστικών 6 ώρες+).

Η εκπαίδευση των μοντέλων έγινε με τη χρήση των ενδεικτικών τιμών παραμέτρων και η επίδοση θα μπορούσε να βελτιωθεί με ένα grid_search() στις παραμέτρους σ, scale , theta και N.

Συμπερασματικά, ο τοπικός περιγραφητής SURF προτείνεται για το ταίριασμα των εικόνων υπό περιστροφή και αλλαγή κλίμακας, ενώ για το πρόβλημα της κατηγοριοποίησης συστήνεται η χρήση του HOG. Αναφορικά με τους ανιχνευτές σημείων ενδιαφέροντος, η πολυκλιμακωτή ανίχνευση με box filters είναι η καταλληλότερη.