Ψηφιακή Επεξεργασία Εικόνας

$2^{η}$ Εργαστηριακή Αναφορά

Εργαστηριακή Ομάδα *76*

Γιουμερτάκης Απόστολος, 2017030142
Κατσούπης Ευάγγελος, 2017030077

Mean Filter

Ζητήθηκε η κατασκευή μιας συνάρτησης, Compute_Mean(), για την κατασκευή ενός φίλτρου μέσου όρου (Mean filter), η οποία δέχεται ως όρισμα την αρχική εικόνα με τον θόρυβο και το μέγεθος του πίνακα kernel. Ως έξοδο, επιστρέφει την εικόνα μετά την εφαρμογή του φίλτρου, απαλαγμένη, στο μέγιστο, από θόρυβο, στο ίδιο μέγεθος με την αρχική.

Λειτουργία της συνάρτησης:
Γίνεται επέκταση των σειρών και στηλών της εικόνας, σύμφωνα με τις διαστάσεις του kernel και στην συνέχεια γέμισμα με μηδενικά (zero padding). Ο αλγόριθμος που εφαρμόζεται για την επέκταση των σειρών και των στηλών τις εικόνας είναι:

$$IMG\_ROWS = \left\lfloor \frac{KERNEL\_ROWS}{2} \right\rfloor$$

$$IMG\_COLUMNS = \left\lfloor \frac{KERNEL\_COLUMNS}{2} \right\rfloor$$

Παρακάτω φαίνεται το παράδειγμα για μάσκα $3X3$. Αριστερά η αρχική εικόνα χωρίς padding (Πίνακας 1) και δεξιά με zero padding (Πίνακας 2).

$a_{1,1}$	$\cdots$	$a_{1,i}$
$a_{2,1}$	$\cdots$	$a_{2,i}$
$a_{3,3}$	$\cdots$	$a_{3,i}$

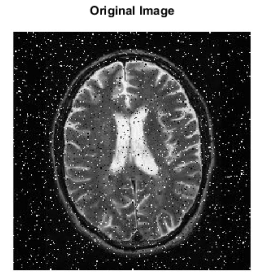
0	0	0	0	0
0	$a_{1,1}$	$\cdots$	$a_{1,i}$	0
0	$a_{2,1}$	$\cdots$	$a_{2,i}$	0
0	$a_{3,3}$	$\cdots$	$a_{3,i}$	0
0	0	0	0	0

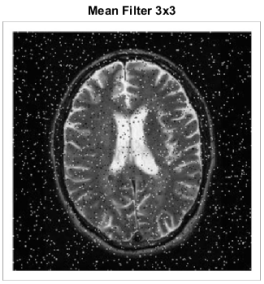
Η εφαρμογή της επέκτασης είναι απαραίτητη, καθώς είναι αναγκαίο όλα τα pixels της αρχικής εικόνας να βρίσκονται στο κέντρο του kernel, το οποίο είναι αδύνατο να επιτευχθεί χωρίς padding.

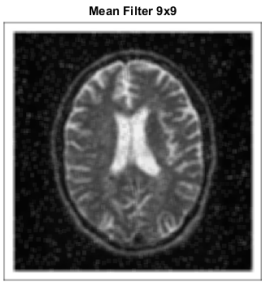
Για κάθε pixel της αρχικής εικόνας, παίρνουμε ένα παράθυρο με μέγεθος όσο και το kernel, το οποίο περιέχει τις τιμές φωτεινότητας των γειτονικών pixels καθώς και την τιμή του ίδιου στο κέντρο του πίνακα. Στην συνέχεια, υπολογίζεται ο μέσος όρος όλων των pixels στο παράθυρο και η τιμή που προκύπτει αντικαθίσταται στην αρχική εικόνα. Στην περίπτωση μας, η τιμή αυτή τοποθετείται σε έναν νέο πίνακα, ο οποίος επιστρέφεται στο τέλος της συνάρτησης. Η διαδικασία αυτή επαναλαμβάνεται σειριακά, κατά μήκος όλων των σειρών και στηλών της εικόνας.

Για την διεκπεραίωση της άσκησης χρησιμοποιήθηκαν τρία(3) διαφορετικά kernels, μεγέθους $3X3$, $5X5$, $9X9$ αντίστοιχα. Τα φίλτρα αυτά εφαρμόστηκαν σε δύο(2) εικόνες όπως φαίνεται και παρακάτω.

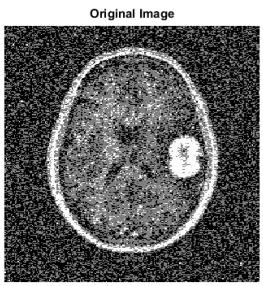
1^{η} Εικόνα:

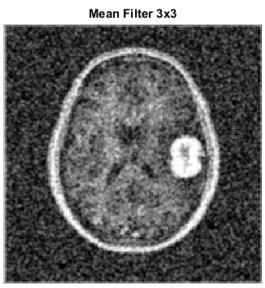


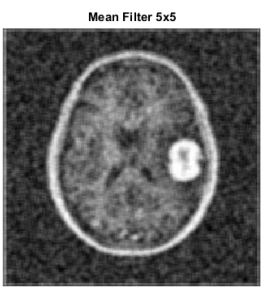


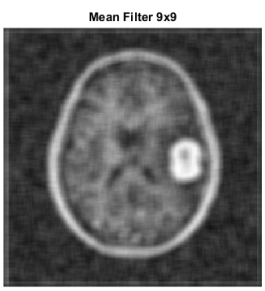


2^{η} Εικόνα:









Παρατηρόντας τις δύο(2) παραραπάνω εικόνες, βλέπουμε πως όσο μεγαλώνει το μέγεθος του kernel, τόσο πιο ¨καθαρή' γίνεται η εικόνα, δηλαδή τόσο περισσότερο απομακρύνεται ο θόρυβος. Όμως, η εικόνα αλλοιώνεται και καταλήγει να γίνεται πιο θολή καθώς αυξάνεται το μέγεθος του kernel.

Συγκεκριμένα, για kernel μεγέθους $3X3$, οι εικόνες είναι ήδη αρκετά καθαρές σε σχέση με την αρχική χωρίς να είναι θολές. Για kernel μεγέθους $5X5$, έχουμε περαιτέρω μείωση του θορύβου αλλά ταυτόχρονα η εικόνα γίνεται ακόμα πιο θολή. Τέλος, στην περίπτωση που το kernel έχει μέγεθος $7X7$, στην πρώτη (1η) εικόνα ο θόρυβος έχει σχεδόν εξαφανιστεί και στην δεύτερη (2η) έχει μειωθεί σημαντικά, αλλά και οι εικόνες έχουν αλλοιωθεί.

Median Filter

Για την δημιουργία της συνάρτησης Compute_Median(), δόθηκαν σαν ορίσματα, η εικόνα αναφορά αλλά και το μέγεθος του Kernel ($Compute\_Median(imageName, [kernel\_rows \;\; kernel\_columns])$). Και σε αυτή την περίπτωση γίνεται padding της εικόνας και συγκεκριμένα replicate padding, σύμφωνα με τα διαστάσεις του kernel. Έστω με μάσκα διαστάσεων $5X7$, η padded εικόνα θα εμπλουτιστεί με 4 νέες γραμμές και 6 στήλες συμμετρικά. Οι τιμές των νέων στοιχείων θα είναι ίδιες με τις ακριανές τιμές της εικόνας αναφοράς.

Για επεξήγηση παρατίθεται το παρακάτω παράδειγμα, όπου στον πρώτο πίνακα βλέπουμε τα αρχικά στοιχεία στης εικόνας αναφοράς και στον δεύτερο, τα στοιχεία μετά το padding.

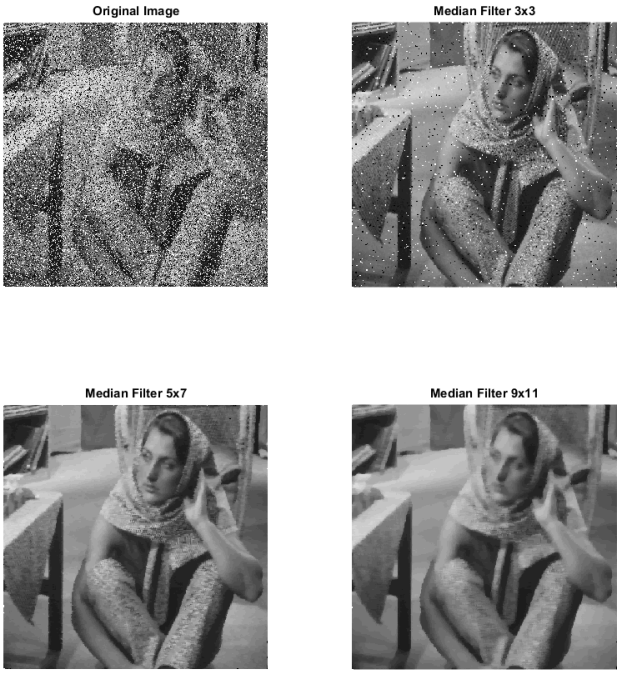
$a_{1,1}$	$\cdots$	$a_{1,i}$
$a_{2,1}$	$\cdots$	$a_{2,i}$
$a_{3,3}$	$\cdots$	$a_{3,i}$

$a_{1,1}$	$a_{1,1}$	$\cdots$	$a_{1,i}$	$a_{1,i}$
$a_{1,1}$	$a_{1,1}$	$\cdots$	$a_{1,i}$	$a_{1,i}$
$a_{2,1}$	$a_{2,1}$	$\cdots$	$a_{2,i}$	$a_{2,i}$
$a_{3,3}$	$a_{3,3}$	$\cdots$	$a_{3,i}$	$a_{3,i}$
$a_{3,3}$	$a_{3,3}$	$\cdots$	$a_{3,i}$	$a_{3,i}$

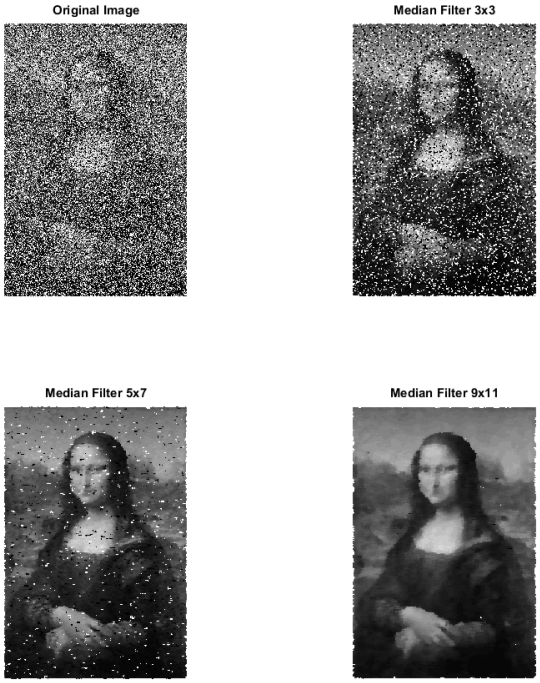
Έχοντας στα χέρια μας την νέα εικόνα και το μέγεθος του Kernel, σειριακά πέρνουμε 'παράθυρα', μεγέθους Kernel, της padded εικόνας με κεντρικό στοιχείο το αντίστοιχο της αρχικής εικόνας. Για κάθε τέτοιο παράθυρο βρίσκουμε την διάμεσό του (το στειχείο το οποία είναι μεγαλύτερο ή ίσο από τα μισά στοιχεία του παραθύρου και μικρότερο ή ίσο από τα υπόλοιπα) και αποθηκέυουμε την τιμή σε νέο πίνακα όπου και θα επιστραφεί από την συνάρτηση.

Με την παραπάνω μεθοδολογία ελένχθηκαν δύο εικόνες, για Kernel με μεγέθη $3X3,\ 5X7,\ 9X11$.

1^{η} Εικόνα:



2^{η} Εικόνα:



Παρατηρείται, και στις δυο περιπτώσεις η τεράστια επιδραση του φίλτρου. Αν αναλύσουμε τις τιμές τις αρχικής εικόνας, παρουσία θορύβου, παρατηρείται πως γειτονία από pixel που έχει επηρεαστεί από θορύβο, περιέχει πληροφορία με μεγάλες διαφορές στις τιμές της (μεγάλη συχνότητα). Όταν εφαρμόζεται το Median Filter για αυτή την γειτονία (παράθυρο), απορρίπτονται οι ακρέες τιμές καθώς αποθηκεύετε η διάμεσος των στοιχείων που ελέγχουμε κάθε φορά. Με χρήση μικρού (πχ $3X3$) φίλτρου, υπάρχει πιθανότητα οι τιμές του παραθύρου που ελέγχεται, να περιέχουν όλες ακραίες τιμές, με αποτέλεσμα η διάμεσός τους να έχει την ίδια συμπεριφορά και να μην οδηγεί σε βελτίωση. Όσο μεγαλύτερο είναι το παράθυρο, τόσο θα εξομαλύνεται η πληροφορία που θα αποθηκεύεται, καθώς θα λαμβάνονται παραπάνω τιμές υπόψιν.

Αξίζει να σημειωθεί πως το παραπάνω, έχει και το μειονέκτημα της αλλοίωσης της πληροφορίας που μας ενδιαφέρει. Αυτό γίνεται πιο εμφανές στην 1^{η} εικόνα, όπου για μέγεθος Kernel $3X3$ υπάρχει παραπάνω θόρυβος σε σχέση με τα υπόλοιπα μεγέθη, αλλά έχουμε και πιο εμφανείς λεπτομέρειες σε σημεία με μεγάλη συγχότητα (πχ πρόσωπο).

Εν κατακλείδι, αντιλαμβανόμαστε πώς το Median φίλτρο είναι εξαιρετικό στο να αποκόπτει πληροφορία με μεγάλες συγχότητες με το μειονέκτημα ότι επιδρά και στην χρήσιμη πληροφορία. Στο παράδειγμα της Mona Lisa όπου ο γνήσιος πίνακας περιέχει αρκετή λεπτομέρεια σε μικρά σημεία, μετά την επεξεργασία έχουμε σαν αποτέλεσμα, εικόνα όπου απουσιάζει αρκετός θόρυβος, αλλά και αρκετή πληροφορία σε σημεία του προσόπου και των ρούχων (σε αντίθεση με τον ουρανό στο background).

Min-Max Filters

Με παρόμοιο τρόπο σχηματίστηκαν και οι συναρτήσεις $Compute\_Min()$ και $Compute\_Max()$ και με ίδια ορίσματα. Σε αντίθεση με τις άλλες συναρτήσεις, το padding έγινε με συμμετρικό τρόπο (symmetric padding). Παράδειγμα της παραπάνω διαδικασίας είναι η εξής:

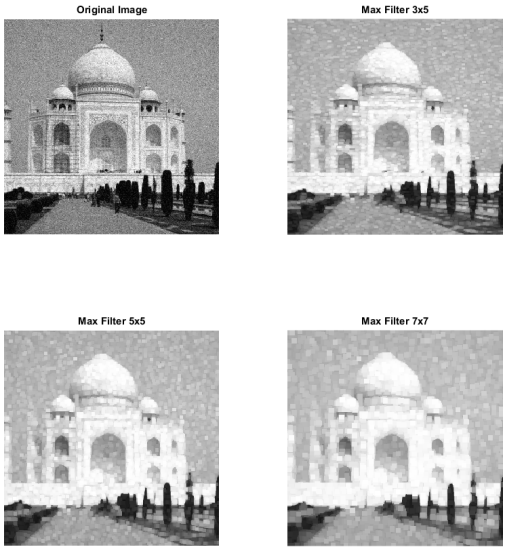
$a_{1,1}$	$\cdots$	$a_{1,i}$
$a_{2,1}$	$\cdots$	$a_{2,i}$
$a_{3,3}$	$\cdots$	$a_{3,i}$

$a_{2,2}$	$a_{2,1}$	$a_{2,1}$	$\cdots$	$a_{2,i}$	$a_{2,i}$	$a_{2,i-1}$
$a_{1,2}$	$a_{1,1}$	$a_{1,1}$	$\cdots$	$a_{1,i}$	$a_{1,i}$	$a_{1,i-1}$
$a_{1,2}$	$a_{1,1}$	$a_{1,1}$	$\cdots$	$a_{1,i}$	$a_{1,i}$	$a_{1,i-1}$
$a_{2,2}$	$a_{2,1}$	$a_{2,1}$	$\cdots$	$a_{2,i}$	$a_{2,i}$	$a_{2,i-1}$
$a_{3,2}$	$a_{3,1}$	$a_{3,3}$	$\cdots$	$a_{3,i}$	$a_{3,i}$	$a_{3,i-1}$
$a_{3,2}$	$a_{3,1}$	$a_{3,1}$	$\cdots$	$a_{3,i}$	$a_{3,i}$	$a_{3,i-1}$
$a_{2,2}$	$a_{2,1}$	$a_{2,1}$	$\cdots$	$a_{2,i}$	$a_{2,i}$	$a_{2,i-1}$

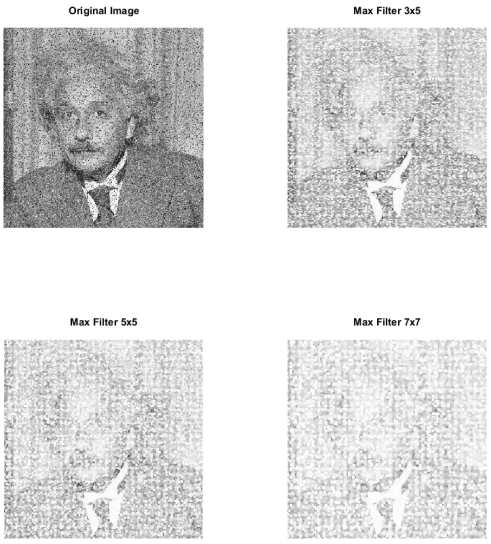
Δημιουργώντας την νέα εικόνα, με τρόπο όμοιο με τους παραπάνω αλγόριθμους, υπολογίζουμε το μέγιστο και το ελάχιστο στοιχείο του κάθε παραθύρου και το αποθηκεύουμε την τιμή αυτή στην νέα εικόνα, για τον αντίστοιχο αλγόριθμο.

Τα αποτελέσματα έπειτα από επεξεργασία των εικόνων αναφοράς για επεξεργασία με max filter:

$1^{η}$ Εικόνα:



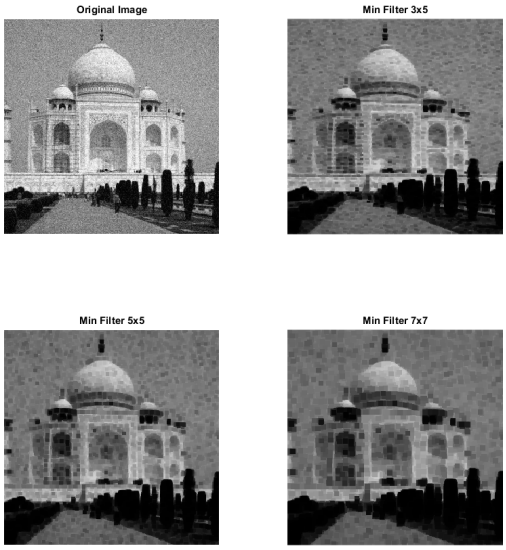
$2^{η}$ Εικόνα:



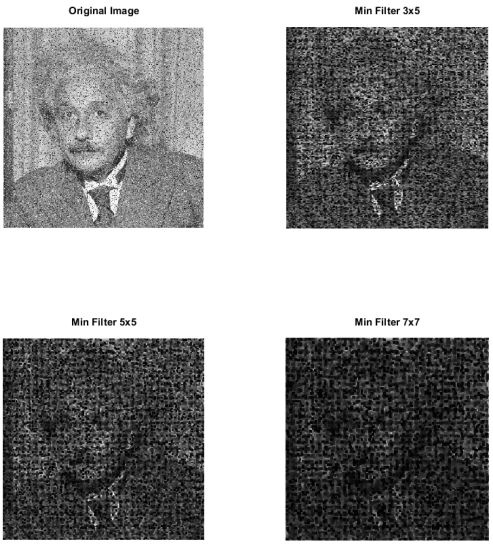
Παρατηρούμε πως οι εικόνες είναι πιο φωτεινές, λόγου του ότι αποθηκέυεται μέγιστες τιμές κάθε παραθύρου. Όσο πιο μεγάλο το παράθυρο τόσο πιο μεγάλο έυρος τιμών που θα ελέγχεται και άρα η μεγαλήτερη τιμή εξ αυτών θα αποθηκέυεται. Το Max φίλτρο είναι καλό για ανίχνευση 'φωτεινών', σε σχέση με την υπόλοιπη εικόνα, λεπτομερειών. Παράδειγμα σε αυτό, μπορεί να χαρακτηριστεί η πρώτη εικόνα όπου μπορούμε να αντιληφθούμε που βρήσκονται τα κτίρια στο συνολό της.

Αντίσοιχα, για επεξεργασία με min filter:

1^{η} Εικόνα:



2^{η} Εικόνα:



Όμοια με το προαναφερθέν φίλτρο, σε αυτή την περίπτωση ανιχνέυονται οι χαμηλές (σκουρόχρωμες), σε σχέση με την υπόλοιπη εικόνα, τιμές πληροφορίας, οι οποίες και εδώ, είναι ανάλογες με το μέγεθος του Kernel.

Και στις δύο περιπτώσεις, παρατηρούμε, πως τα φίλτρα δεν επιδρούν θετικά στην συνολική εικόνα σε σχέση με τον θόρυβο, καθώς η εικόνα που προκύπτει, έχει αποθηκευμένες και τις τιμές του θορύβου.