Εργασία 1

ΨΗΦΙΑΚΗ ΕΠΕΞΕΡΓΑΣΙΑ ΕΙΚΟΝΑΣ

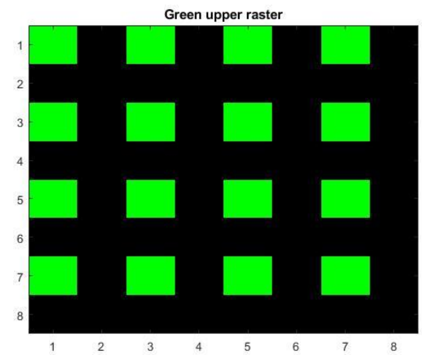
Αντώνιος Σκούρτης (9142) | askourtis@ece.auth.gr | 24/04/2021

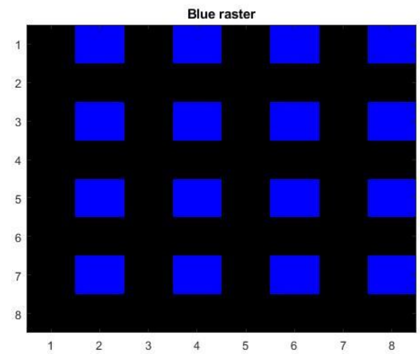
Περιεχόμενα

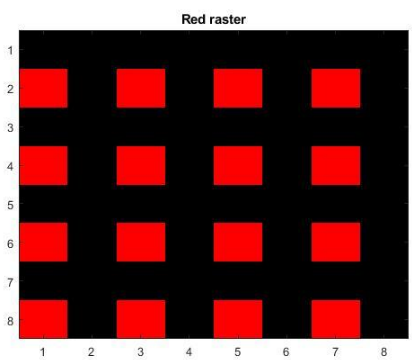
Φίλτρο Bayer

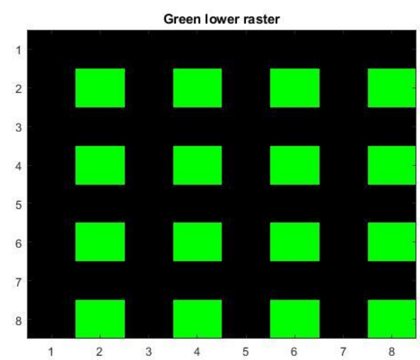
Τα αρχικά δεδομένα δίνονται με τη μορφή ενός μωσαϊκού γνωστό ως φίλτρο Bayer. Το φίλτρο αυτό αποτελείται από τέσσερα διαφορετικά και μη επικαλυπτόμενα μοτίβα.

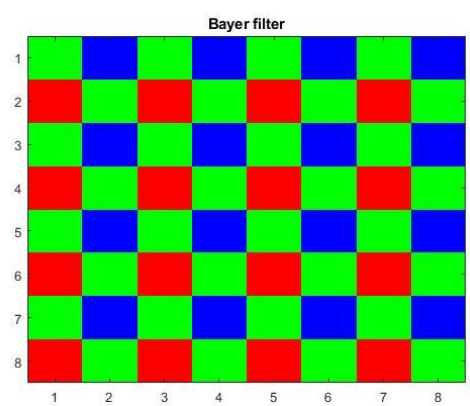
- Κόκκινο μοτίβο
- Μπλε μοτίβο
- Πάνω πράσινο μοτίβο
- Κάτω πράσινο μοτίβο











ΠΡΑΓΜΑΤΙΚΟ ΠΑΡΑΔΕΙΓΜΑ

Χρησιμοποιώντας τα δεδομένα που δόθηκαν για την εργασία και αναθέτοντας στα κατάλληλα pixels τα κατάλληλα χρώματα λαμβάνουμε το παρακάτω αποτέλεσμα.

Προφανώς είναι φανερό τι προσπαθεί να αναπαραστήσει η παραπάνω εικόνα αλλά υπάρχουν πολλά παράσιτα και λανθασμένο συνολικό χρώμα , καθώς η φωτογραφία είναι πρακτικά πράσινη.

Τα παραπάνω αποτελέσματα είναι λογικά καθώς το κάθε pixel έχει μόνο μία απόχρωση ενός βασικού χρώματος δημιουργώντας τα παράσιτα που παρατηρούνται. Επιπλέον τα πράσινα pixels είναι διπλάσια σε αριθμό από τα κόκκινα ή μπλε, επομένως είναι λογικό η φωτογραφία να τείνει προς το πράσινο χρώμα.

Για την επίλυση του προβλήματος αυτού πρέπει να συνδυαστούν τα μονοχρωματικά pixels μεταξύ τους. Ένας, ίσως απλοϊκός, τρόπος είναι να βρεθεί το μέσο χρώμα κάθε pixel βάσει των γειτόνων του. Για την εκπόνηση αυτού, συνελίσεται η εικόνα αυτή με το παρακάτω kernel.

$$K = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Δεδομένου του K είναι λογικό πως μπορεί κάποιο pixel να καταλήξει με φωτεινότητα χρώματος μεγαλύτερη της μονάδας, κάτι το οποίο είναι αδύνατο. Πρέπει λοιπόν το K να κανονικοποιηθεί. Το κανονικοποιημένο $\widehat{K}$ είναι

$$\widehat{K} = \frac{\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}}{9}$$

Μετά την εφαρμογή του $\widehat{K}$ πάνω στην παραπάνω εικόνα, λαμβάνουμε το εξής αποτέλεσμα

Η φωτογραφία έχει εμφανώς λιγότερα παράσιτα. Μετά την εφαρμογή μεγαλύτερου φίλτρου όπως το παρακάτω

$$\widehat{K_{10}} = \frac{ones(10)}{10^2}$$

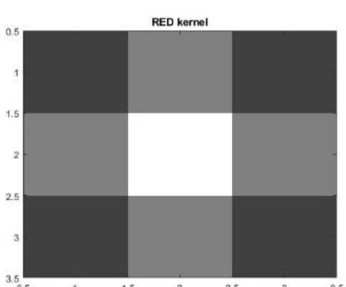
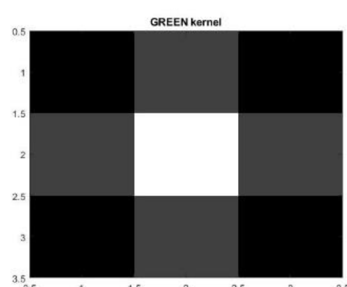
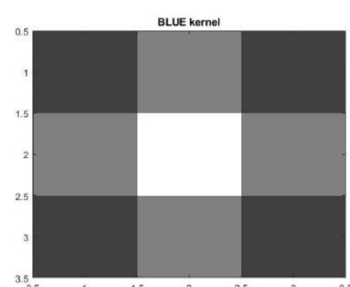
Το αποτέλεσμα είναι μια θολή φωτογραφία με πράσινο χρώμα όπως η παρακάτω

Εφόσον έχουμε δύο φορές πιο πολλά pixels στο πράσινο κανάλι από κάθε άλλο κανάλι, τότε πρέπει το πράσινο κανάλι να διαιρεθεί διά του δύο, για να μην υπερκαλύπτει τα άλλα χρώματα. Εφαρμόζοντας την παραπάνω λογική παράγεται το παρακάτω αποτέλεσμα

Η συγκεκριμένη φωτογραφία είναι μία πολύ καλή βελτίωση της πρώτης προσπάθειας, καθώς δεν υπάρχουν παράσιτα και η αναλογία των χρωμάτων είναι σωστή. Το αποτέλεσμα όμως δεν είναι το βέλτιστο καθώς είναι πολύ θολωμένο και πολύ σκοτεινό.

Η φωτογραφία είναι σκοτεινή καθώς το φίλτρο απλώς υπολογίζει τον μέσο όρο των γειτόνων, οι περισσότεροι από τους οποίους έχουν μηδενικό χρώμα λόγω του φίλτρου Bayer. Επιπλέον είναι θολή καθώς το φίλτρο του μέσου όρου κάνει ακριβώς αυτό, δηλαδή θολώνει μία εικόνα.

Στην πραγματικότητα πρέπει να υπολογιστούν τα κενά pixels μέσω παρεμβολής κάτι το οποίο είναι αρκετά απλό για το φίλτρο Bayer, αφού μπορεί να υπολογιστεί μέσω συνέλιξης και κατάλληλου Kernel. Πιο συγκεκριμένα οι τα kernels για το κάθε κανάλι είναι

		
$K_R = \frac{\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}}{4}$	$K_G = \frac{\begin{bmatrix}0 & 1 & 0\\ 1 & 4 & 1\\ 0 & 1 & 0\end{bmatrix}}{4}$	$K_B = \frac{\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}}{4}$

Εφαρμόζοντας λοιπόν τους παραπάνω kernels παράγεται το παρακάτω αποτέλεσμα

Resulting image

Είναι φανερό λοιπόν πως η παραπάνω φωτογραφία είναι μία πολύ καλή προσέγγιση, το πράσινο χρώμα πλέον βρίσκεται μόνο στα σημεία που πρέπει, και η φωτογραφία δεν είναι θολωμένη, ενώ ταυτόχρονα η φωτεινότητα είναι στα επίπεδα που πρέπει να είναι.

Υποδειγματοληψία εικόνας

Στο παρών κομμάτι της εργασία πρέπει να αλλάξουν οι διαστάσεις της φωτογραφίας μέσω υποδειγματοληψίας. Οι δύο μέθοδοι υποδειγματοληψίας που έπρεπε να υλοποιηθούν είναι οι παρακάτω

- Nearest Neighbor
- Bilinear Interpolation

NEAREST NEIGHBOR

Σε αυτήν την περίπτωση τα χρώματα των pixels που δεν αντιστοιχούν σε ακέραιες θέσεις λαμβάνουν την τιμή του κοντινότερου γείτονά τους. Προφανώς τα pixels που αντιστοιχούν σε ακέραιες θέσεις, διατηρούν το χρώμα τους. Εκτελώντας το παραπάνω για να γίνει η φωτογραφία μεγέθους 240×320. Δεδομένου πως το αρχικό μέγεθος είναι 960×1280 η φωτογραφία μικραίνει τέσσερεις φορές.

Παρατηρείται στο $Image\ 1$ πως η φωτογραφία είναι αναγνωρίσιμη και χωρίς πολλά παράσιτα, αν και όχι ανύπαρκτα. Πιο συγκεκριμένα σε διαγώνιους όπως στο κλαδί, φαίνεται πως η μέθοδος αποτυγχάνει και δημιουργεί άσχημες γωνίες καθώς το χρώμα αλλάζει έντονα στα όρια του κλαδιού.

BILINEAR INTERPOLATION

Το παραπάνω πρόβλημα λύνει σε ικανοποιητικό βαθμό η μέθοδος αυτή. Πιο συγκεκριμένα, πλέον, αντί να υπολογίζεται το χρώμα από τον κοντινότερο γείτονα, το χρώμα υπολογίζεται από το αποτέλεσμα μίας διγραμμικής παρεμβολής μεταξύ των γειτόνων. Το αποτέλεσμα μετά από σμίκρυνση 200×300 είναι

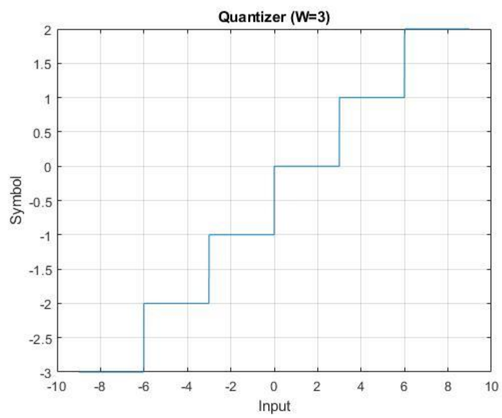
Είναι φανερό, από το $Image\ 2$, πως η μέθοδος αυτή παράγει πιο ομαλά αποτελέσματα από την *nearest neighbor*, χωρίς αυτό να σημαίνει πως είναι τέλειο το αποτέλεσμα.

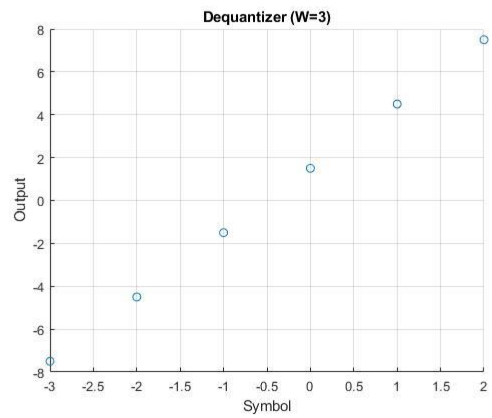
Image 1: Μέθοδος Nearest Neighbor

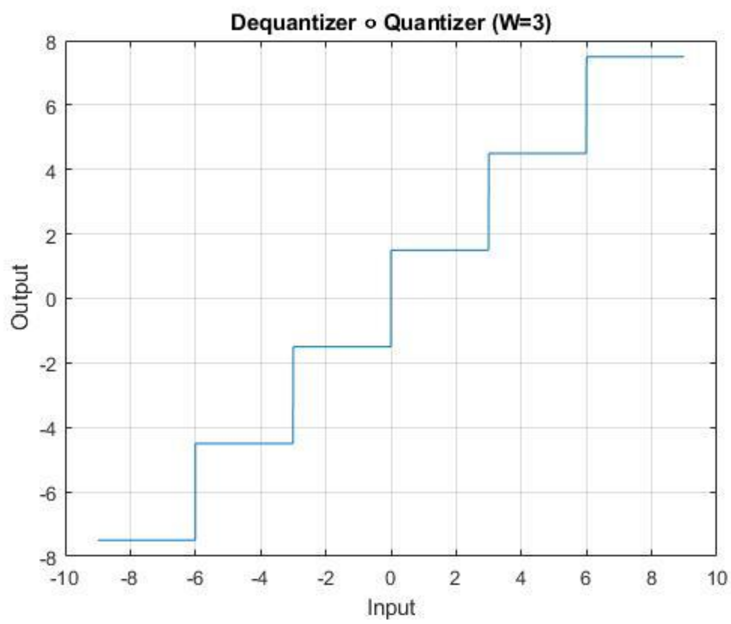
Image 2: Μέθοδος Bilinear Interpolation

Κβαντισμός

Για τις ανάγκες του κβαντισμού υλοποιήθηκε ένας midrise ομοιόμορφος και συμμετρικός κβαντιστής και αποκβαντιστής με πλάτος ζώνης w.





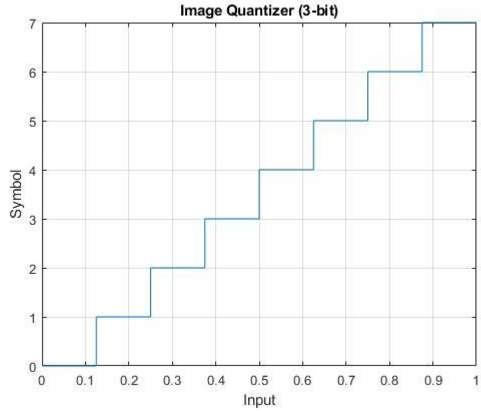


Επομένως κάθε σύμβολο i αντιστοιχεί στο σύνολο εισόδου $[i \times w, (i+1) \times w)$ και στην τιμή εξόδου $i \times w + \frac{w}{2}$.

Όμως η κάθε χρωματική συνιστώσα σε ένα στοιχείο της εικόνας ανήκει πάντα στο διάστημα $[0,1]$ επομένως ο κβαντιστής πρέπει να κβαντίσει σε αυτήν την περιοχή. Επιπλέον, δεδομένου ότι $0 < w < 1$, θα υπάρχουν $\frac{1}{w} + 1$ διαφορετικές ζώνες. Αφού η τελευταία ζώνη θα απαρτιζόταν από μόνο μία τιμή, την μονάδα. Καθώς αυτό είναι σπάταλο, δηλαδή να υπάρχει ολόκληρη ζώνη για μία μόνο τιμή, η οριακή τιμή της μονάδας εντάσσεται στην προηγούμενη ζώνη. Δηλαδή στο σύμβολο $\frac{1}{w} - 1$ θα αντιστοιχεί το διάστημα εισόδου $\left[\left(\frac{1}{w} - 1\right) \times w, 1\right] = [1 - w, 1]$ και η τιμή εξόδου $1 - \frac{w}{2}$. Τέλος στο διάστημα $[0,1]$ θα υπάρχουν τελικά $\frac{1}{w}$ διαφορετικές ζώνες και σύμβολα.

Δεδομένων των παραπάνω, ένας κβαντιστής με n-bits χρειάζεται ζώνη $w = \frac{1}{2^n}$ καθώς έτσι θα υπάρχουν $\frac{1}{\frac{1}{2^n}}$ διαφορετικά σύμβολα δηλαδή 2^n διαφορετικά σύμβολα $i \in \{1, 2, 3, 4, \cdots, 2^n - 1\}$

Άρα το αποτέλεσμα του κβαντισμού της φωτογραφίας με κβαντιστή 3 bit είναι



Πρότυπο PPM

Μετά την αποθήκευση της εικόνας μέσω του προτύπου PPM τα αποτελέσματα είναι τα παρακάτω

Προβλήματα

ΥΠΟΔΕΙΓΜΑΤΟΛΗΨΙΑ

Κατά την υποδειγματοληψία φαίνεται να υπάρχουν διαφορές με την ήδη υπάρχουσα συνάρτηση imresize της Matlab. Πιο συγκεκριμένα φαίνεται τα σφάλματα να συγκεντρώνονται στις ακμές των γεωμετρικών σχημάτων, όπως τα πέταλα των λουλουδιών, τα όρια των κάγκελων αλλά και τα όρια της φωτογραφίας αυτής καθ' αυτής.

Image 3: Σφάλματα της μεθόδου Nearest

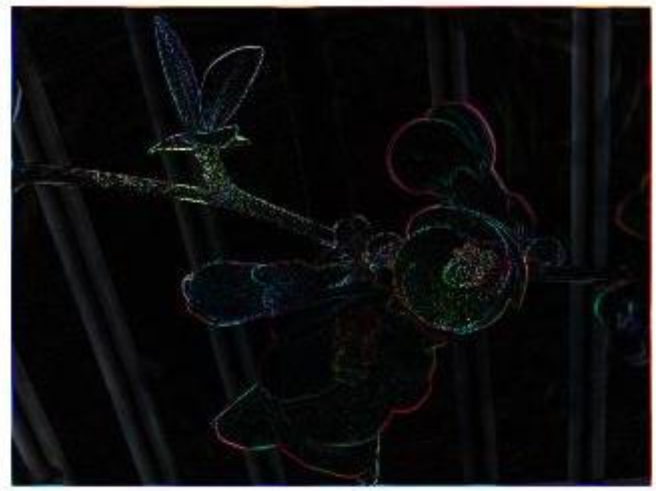


Image 4: Σφάλματα της μεθόδου Linear

Στις φωτογραφίες 3 και 4 όσο πιο φωτεινό είναι το pixel τόσο μεγαλύτερο το σφάλμα σε εκείνο το σημείο και το χρώμα δηλώνει ποιο χρωματικό κανάλι εμφανίζει το σφάλμα.

Παρατηρείται πως και οι δύο μέθοδοι αποκλίνουν από την προ υπάρχουσα συνάρτηση της Matlab και πως οι περισσότερες αποκλίσεις βρίσκονται στα άκρα των σχημάτων.

Ενδιαφέρουσα περίπτωση είναι η έλλειψη σφάλματος στην δεξιά κάτω γωνία της μεθόδου Nearest, καθώς η μέθοδος επεξεργάζεται τα pixels ισάξια και η ύπαρξη μίας συνεκτικής περιοχής χωρίς σφάλμα είναι άξιο απορίας.

ΠΡΟΤΥΠΟ PPM

Είναι εμφανές πως η φωτογραφία που διαβάστηκε από το αρχείο ppm έχει πιο έντονα χρώματα σε σχέση με την τοπικά αποκβαντισμένη εκδοχή της. Μετά από διερεύνηση του προβλήματος παρατηρήθηκε πως ο αποκβαντισμός μέσω της imread γίνεται με διαφορετικό κβαντιστή, καθώς τα σύμβολα $s = 2^3 - 1$ λαμβάνουν την τιμή 1 ενώ τα σύμβολα $s = 0$ λαμβάνουν την τιμή 0. Κάτι τέτοιο δεν μπορεί να υλοποιηθεί με midrise κβαντιστή.

Κώδικας

Το κάθε script της Matlab είναι επαρκώς σχολιασμένος, επομένως δεν σχολιάζεται περεταίρω στην αναφορά. Επιπλέον μπορεί να βρεθεί αναρτημένος και στο αποθετήριό μου στο GitHub.