Ψηφιακή Επεξεργασία Εικόνας

1η Εργασία Μαθήματος 2022 - 2023

Αλεξόπουλος Δημήτριος ΑΕΜ 10091
aadimitri@ece.auth.gr

Περιεχόμενα

1 Εισαγωγή: Από τον αισθητήρα στη μνήμη 4
- 1.1 White Balance . . . 4
- 1.2 Παρεμβολή . . . 5
  - 1.2.1 Nearest Neighbor Interpolation . . . 5
  - 1.2.2 Bilinear Interpolation . . . 6
- 1.3 Μετατροπή color space . . . 8

2 Υλοποίηση 10
- 2.1 readdng . . . 10
- 2.2 dng2rbg . . . 10
  - 2.2.1 wbmask . . . 11
  - 2.2.2 transform . . . 12
  - 2.2.3 nearest . . . 12
  - 2.2.4 bilinear . . . 13

3 Αποτελέσματα 14
- 3.1 *Raw* εικόνα . . . 14
- 3.2 *Ccam* εικόνα . . . 14
- 3.3 *Cxyz* εικόνα . . . 15
- 3.4 *Clinear* εικόνα . . . 16
- 3.5 *Csrgb* εικόνα . . . 17
- 3.6 Διαφορετικό *Bayer* . . . 18
- 3.7 Μέθοδος *nearest* . . . 18

Βιβλιογραφία 20

Κατάλογος Σχημάτων

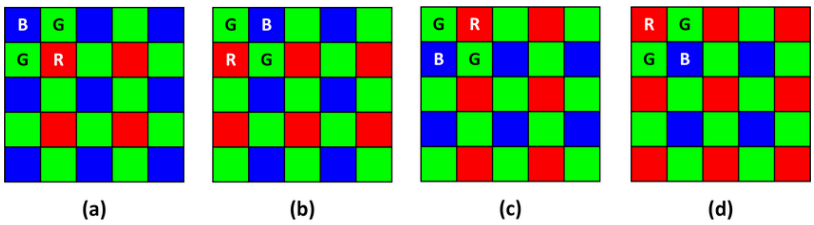
1.1 Bayer patterns: (a) BGGR, (b) GBRG, (c) GRBG, and (d) RGGB. 4
1.2 Red and Blue channels demosaicing 6
1.3 Green Channel demosaicing 6
1.4 Diagonal Cross Interpolation 7
1.5 Horizontal Interpolation 7
1.6 Vertical Interpolation 7
1.7 Green Channel demosaicing 8

3.1 *Raw* εικόνα 14
3.2 *Ccam* εικόνα 15
3.3 *Cxyz* εικόνα 16
3.4 *Clinear* εικόνα 17
3.5 *Clinear* εικόνα 18
3.6 *BGGR*, *GBRG* 18
3.7 *GRBG*, *RGGB* 18
3.8 Μέθοδος *nearest* 19

1 Εισαγωγή: Από τον αισθητήρα στη μνήμη

Αντικείμενο της παρούσας εργασίας είναι η μετατροπή μιας εικόνας από RAW format, όπως αυτή προκύπτει μετά την φωτογραφική λήψη, σε μορφή jpeg, εύχρηστη για την προβολή της και την αποθήκευσή της στην μνήμη.

Η ψηφιακή καταγραφή έγχρωμων εικόνων με τη χρήση RGB αισθητήρων υλοποιείται από πλέγμα αισθητήρων τοποθετημένων σε μια προτυποποιημένη διάταξη, όπως για παράδειγμα αυτή του πρότυπου Bayer που φαίνεται παρακάτω:



Σχήμα 1.1: *Bayer patterns: (a) BGGR, (b) GBRG, (c) GRBG, and (d) RGGB.*

Κατά τη λήψη, λοιπόν, της φωτογραφίας σε κάθε θέση του πλέγματος καταγράφεται ένα από τα τρία κανάλια χρώματος R, G ή B. Ο παραγόμενος πίνακας καταγραφών αποτελεί την εικόνα σε RAW format. Η raw εικόνα έχει διαστάσεις $M_0 \times N_0$ ακριβώς ίσες με τις διαστάσεις του πλέγματος. Οι RGB εικόνες που συνήθως χρησιμοποιούμε, π.χ. μορφής JPEG, αποτελούν προϊόν μετέπειτα υπολογισμών, τους οποίους θα αναλύσουμε και θα υλοποιήσουμε στη συνέχεια.

Επειδή κατά τις φωτογραφικές λήψεις από μηχανές διαφόρων κατασκευαστών δεν υπάρχει ενιαίο πρωτόκολλο, ο ακριβής τρόπος με τον οποίο αποθηκεύεται το RAW format μαζί με τα συνοδευτικά metadata διαφοροποιείται από περίπτωση σε περίπτωση. Για τον λόγο αυτό, στα πλαίσια της εργασίας θα εργαστούμε μόνο με εικόνες που ακολουθούν το κοινό ανοιχτό format της Adobe, το οποίο ονομάζεται Digital Negative με τα αντίστοιχα αρχεία να έχουν την κατάληξη .DNG. Τα metadata που αναφέρθηκαν παραπάνω αφορούν χρήσιμες πληροφορίες για την μετατροπή της raw εικόνας σε RGB και θα αναλυθούν επίσης παρακάτω.

Για την μετατροπή, λοιπόν, αυτή απαιτούνται τρία βασικά στάδια:

1. ρύθμιση του white balance της raw εικόνας,
2. διαδικασία παρεμβολής τιμών R, G και B, ώστε κάθε δείγμα της παραγόμενης εικόνας να έχει τιμή για κάθε ένα από τα τρία κανάλια,
3. μετατροπή από το color space της κάμερας στο color space αναπαραγωγής (display) της εικόνας.

1.1 White Balance

Κατά την λήψη ενός αντικειμένου σε μία φωτογραφία, λευκές περιοχές του αντικειμένου θα πρέπει να φαίνονται λευκές, δηλαδή θα πρέπει το οποιοδήποτε καταγραφόμενο χρώμα α του αντικειμένου να παίρνει στο πλέγμα την μορφή:

$$\alpha = \lambda \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \text{ με } \lambda \in [0,1] \tag{1.1}$$

Επειδή, όμως, το φάσμα του προσπίπτοντος φωτός δεν είναι πάντα ισοκατανεμημένο στις περιοχές R, G και B μπορεί η παραπάνω συνθήκη να ανατραπεί οπότε χρειάζεται να πολλαπλασιαστούν με κατάλληλους διορθωτικούς συντελεστές τα 2 από τα τρία κανάλια. Συνήθως διατηρείται αμετάβλητο το G και πολλαπλασιάζονται με διορθωτικούς συντελεστές τα άλλα δύο (R, B), καθώς μόνο η αναλογία των τριών χρωμάτων έχει σημασία. Οι συντελεστές που χρησιμοποιούνται είναι κοινοί για όλα τα pixels της εικόνας, καθώς υιοθετείται η απλούστευση ότι το φάσμα του προσπίπτοντος φωτός είναι ίδιο σε όλα τα εικονιζόμενα σημεία. Οι συντελεστές white balance είναι αποθηκευμένοι στα metadata της εικόνας.

1.2 Παρεμβολή

Η αρχική raw εικόνα μας είναι ένας δισδιάστατος πίνακας στον οποίον βρίσκεται 'κρυμμένη' η πληροφορία για το χρώμα του κάθε pixel με βάση μία από τις διατάξεις του προτύπου Bayer του σχήματος 1.1. Για την εξαγωγή αυτής της πληροφορίας, χρησιμοποιούμε μεθόδους παρεμβολής και υπολογίζουμε, έτσι, την τιμή του καναλιού $C \in R,G,B$ σε μία θέση x, y του πλέγματος με βάση τις γνωστές τιμές του ίδιου καναλιού σε γειτονικές θέσεις. Οι θέσεις x, y επιλέγονται στο πλέγμα σημείων με πυκνότητα που επιλέγεται από τον χρήστη και δεν είναι απαραίτητο να συμπίπτουν με θέσεις του πρωτογενούς πλέγματος του αισθητήρα.

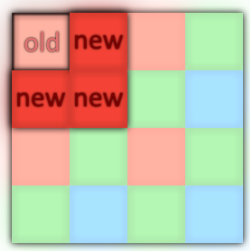
Για τους σκοπούς της εργασίας θα χρησιμοποιήσουμε δύο τεχνικές demosaicing, δηλαδή τεχνικές ανακατασκευής του πλήρους χρώματος μιας εικόνας από τα δείγματα που συλλέγει ένας ψηφιακός φακός κάμερας. Κοινό χαρακτηριστικό και των δύο αποτελεί η διαίρεση του πλέγματος των pixels της εικόνας σε 2×2 blocks στο καθένα από τα οποία, σύμφωνα με το πρότυπο Bayer, υπάρχουν χρώματα που 'λείπουν'.

1.2.1 Nearest Neighbor Interpolation

Σύμφωνα με αυτή τη τεχνική για κάθε 2×2 block από pixels στην εικόνα βρίσκουμε τα χρώματα που 'λείπουν' παρεμβάλοντάς τα με τα κοντινότερα (nearest) pixels του block. Διακρίνουμε, λοιπόν, δύο περιπτώσεις, μία για τα χρώματα που εμφανίζονται μόνο μία φορά σε κάθε block (R, B) και μία για το χρώμα που εμφανίζεται δύο φορές (G).

- **Red and Blue channels demosaicing**

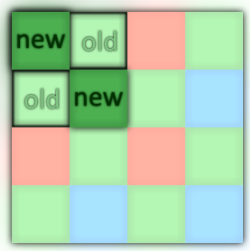
  Σε κάθε ένα από τα 2×2 blocks γεμίζουμε τα τρία pixels που 'λείπουν' με ακριβώς την ίδια τιμή που έχει και το μοναδικό σωστά χρωματισμένο pixel του block. Αυτό γίνεται ξεκάθαρο στην παρακάτω εικόνα:



Σχήμα 1.2: *Red and Blue channels demosaicing*

- **Green Channel demosaicing**

  Σε κάθε ένα από τα 2×2 blocks γεμίζουμε τα δύο pixels που 'λείπουν' με την τιμή του γειτονικού τους στην ίδια γραμμή. Έτσι, το πρώτο pixel που 'λείπει' παίρνει την τιμή του πρώτου σωστά χρωματισμένου pixel και το δεύτερο pixel που 'λείπει' παίρνει την τιμή του δεύτερου σωστά χρωματισμένου pixel. Αυτό γίνεται ξεκάθαρο στην παρακάτω εικόνα:



Σχήμα 1.3: *Green Channel demosaicing*

1.2.2 Bilinear Interpolation

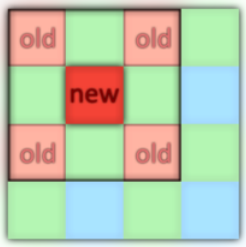
Σύμφωνα με αυτή τη τεχνική και πάλι για κάθε 2×2 block από pixels στην εικόνα βρίσκουμε τα χρώματα που 'λείπουν' παρεμβάλοντάς τα με τα γειτονικά, ωστόσο, τώρα χρησιμοποιούμε τον μέσο όρο τους.

- **Red and Blue channels demosaicing**

  Σε κάθε ένα από τα 2×2 blocks γεμίζουμε τα τρία pixels που 'λείπουν' με την μέση τιμή των γειτονικών τους pixels. Διακρίνουμε τρεις περιπτώσεις, ανάλογα με την θέση του pixel που 'λείπει':

  – **Diagonal Cross Interpolation**

    Σε αυτήν την περίπτωση το pixel που 'λείπει' έχει τέσσερεις σωστά χρωματισμένους γείτονες γύρω του κι άρα η τιμή του θα είναι η μέση τιμή των τεσσάρων αυτών pixels. Αυτό γίνεται ξεκάθαρο στην παρακάτω εικόνα:



Σχήμα 1.4: *Diagonal Cross Interpolation*

– **Horizontal Interpolation**

Σε αυτήν την περίπτωση το pixel που 'λείπει' έχει δύο σωστά χρωματισμένους γείτονες γύρω του, έναν δεξιά του κι έναν αριστερά του, κι άρα η τιμή του θα είναι η μέση τιμή των δύο αυτών pixels. Αυτό γίνεται ξεκάθαρο στην παρακάτω εικόνα:

Σχήμα 1.5: *Horizontal Interpolation*

– **Vertical Interpolation**

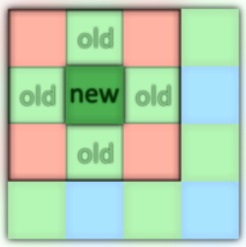
Σε αυτήν την περίπτωση το pixel που 'λείπει' έχει δύο σωστά χρωματισμένους γείτονες γύρω του, έναν από πάνω του κι έναν από κάτω του, κι άρα η τιμή του θα είναι η μέση τιμή των δύο αυτών pixels. Αυτό γίνεται ξεκάθαρο στην παρακάτω εικόνα:

Σχήμα 1.6: *Vertical Interpolation*

- **Green channel demosaicing**

Σε κάθε ένα από τα 2×2 blocks γεμίζουμε τα δύο pixels που 'λείπουν' με την μέση τιμή των γειτονικών τους pixels. Σε αυτήν την περίπτωση το pixel που 'λείπει' έχει τέσσερεις

σωστά χρωματισμένους γείτονες γύρω του κι άρα η τιμή του θα είναι η μέση τιμή των τεσσάρων αυτών pixels. Αυτό γίνεται ξεκάθαρο στην παρακάτω εικόνα:



Σχήμα 1.7: *Green Channel demosaicing*

1.3 Μετατροπή **color space**

Η εικόνα που προκύπτει μετά την παρεμβολή δεν θα έχει τα pixels της στις σωστές συντεταγμένες του RGB χώρου που αναγνωρίζει το λειτουργικό σύστημα. Για τον λόγο αυτό, είναι απαραίτητο να γίνει η μετατροπή από το color space της κάμερας στο color space της οθόνης. Αυτό γίνεται με μια σειρά από γραμμικούς μετασχηματισμούς, κατά τους οποίους σε κάθε pixel της εικόνας εφαρμόζεται ένας 3×3 πίνακας μετασχηματισμού.

Για τον πρώτο μετασχηματισμό αξιοποιούμε τον 3×3 πίνακα $T_{XYZ \to Cam}$ που αποθηκεύει ο κατασκευαστής στα metadata της εικόνας και συνδέει το 3D colorspace της κάμερας με το πρότυπο colorspace XYZ:

$$C_{Cam} = T_{XYZ \to Cam} C_{XYZ} \tag{1.2}$$

Πρέπει να προσέξουμε ότι ο πίνακας αυτός μετασχηματισμού ορίζεται με την αντίθεση κατεύθυνση κι άρα στην υλοποίηση θα πρέπει να αντιστραφεί για να μας δώσει την εικόνα C_{XYZ}:

$$C_{XYZ} = T^{-1}_{XYZ \to Cam} C_{Cam} \tag{1.3}$$

Στη συνέχεια, το πρότυπο CIE ορίζει τον αντίστοιχο πίνακα $T_{XYZ \to RGB}$ μετατροπής του XYZ στο τυποποιημένο σύστημα RGB που χρησιμοποιούν οι συσκευές προβολής (οθόνες, προβολείς):

$$C_{linear} = T_{XYZ \to RGB} C_{XYZ} \tag{1.4}$$

με τον πίνακα μετασχηματισμού να έχει τις τιμές:

$$T_{XYZ \to RGB} = \begin{bmatrix} 3.2406 & -1.5372 & -0.4986 \\ -0.9689 & 1.8758 & 0.0415 \\ 0.0557 & 0.2040 & 1.0570 \end{bmatrix}$$

Πριν από κάθε εφαρμογή μετασχηματισμού, θα πρέπει να προσέξουμε ότι οι γραμμές των πινάκων μετασχηματισμού χρειάζεται να κανονικοποιηθούν ώστε κάθε γραμμή να αθροίζει στο

1. Αυτό συμβαίνει διότι ένα λευκό pixel στο colospace της κάμερας θα πρέπει να παραμείνει λευκό και στο τελικό colorspace. Εφόσον και στα δύο colorspaces το λευκό αναπαρίσταται σε RGB ως $[1 \quad 1 \quad 1]^T$ θα πρέπει να κανονικοποιήσουμε τις γραμμές εξασφαλίζοντας ότι θα ισχύει:

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}_{colorspace_1} = \begin{bmatrix} T_{transform} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}_{colorspace_2} \tag{1.5}$$

Το πρότυπο προβλέπει μια ακόμη μη γραμμική διόρθωση για να πάρουμε την τελική εικόνα και περιγράφεται από την εξίσωση:

$$C_{sRGB} = C_{linear}^{1/2.2} \tag{1.6}$$

Έχουμε τώρα, λοιπόν, εφαρμόσει τους κατάλληλους μετασχηματισμούς στην εικόνα ώστε να έχει τα pixels της στις σωστές συντεταγμένες του RGB χώρου που αναγνωρίζει το λειτουργικό σύστημα και μπορεί να προβάλλει η οθόνη.

2 Υλοποίηση

Κατά την υλοποίηση σε MATLAB της μετατροπής της raw εικόνας μας σε μορφή jpeg που να μπορεί να προβληθεί από την οθόνη, κατασκευάσαμε δύο βασικές συναρτήσεις, readdng και dng2rgb, και κάποιες βοηθητικές συναρτήσεις. Στη συνέχεια θα παρουσιάσουμε λεπτομερέστερα την υλοποίηση αυτών των συναρτήσεων, ακολουθώντας τα βήματα της θεωρητικής ανάλυσης που προηγήθηκε στην εισαγωγή.

2.1 readdng

Κατασκευάζουμε, αρχικά, την συνάρτηση $[rawim, XYZ2Cam, wbcoeffs] = readdng(filename)$ με ορίσματα και εξόδους:

- filename: το path της προς μετατροπή εικόνας
- rawim: ο $M_0 \times N_0$ πίνακας με τις μετρήσεις που προκύπτουν από τον σένσορα της κάμερας (χωρίς τα metadata)
- XYZ2Cam: ο 3×3 πίνακας που εξάγεται από τα metadata και μετατρέπει το colorspace της εικόνας
- wbcoeffs: το 1×3 διάνυσμα που περιέχει τους διορθωτικούς συντελεστές για το white balancing της εικόνας

όπου M_0, N_0 είναι οι διαστάσεις (ύψος και πλάτος αντίστοιχα) της εικόνας μετά την εξαγωγή των metadata.

Σκοπός της συνάρτησης είναι να διαβάσει τα χρήσιμα για την επεξεργασία metadata της εικόνας και στην συνέχεια να τα αποκόψει από αυτήν. Θα δώσει, λοιπόν, στην έξοδο μόνο το μέρος της εικόνας rawim που απαιτείται για την ανάκτηση των χρωμάτων της εικόνας. Επιπλέον, οι τιμές του πίνακα της εικόνας θα έχουν υποστεί σημειακό μετασχηματισμό έτσι ώστε το blacklevel να οδηγηθεί στο 0 και το whitelevel στο 1.

Επειδή λόγω θορύβου μπορεί παρά τις προδιαγραφές των blacklevel και whitelevel η εικόνα να έχει τιμές και έξω από τα όρια, μετά την εφαρμογή του μετασχηματισμού κάνουμε αποκοπή όσων τιμών τύχει να βρεθούν έξω από το διάστημα [0, 1] με την εντολή $rawim = max(0, min(rawim, 1))$;

2.2 dng2rbg

Η συνάρτηση αυτή [Csrgb,Clinear,Cxyz,Ccam] = dng2rgb(rawim,XYZ2Cam,wbcoeffs, bayertype,method,M,N) αποτελεί και την κύρια συνάρτηση για την επεξεργασία της εικόνας μας καλώντας κάποιες βοηθητικές συναρτήσεις. Τα ορίσματα και οι έξοδοι της συνάρτησης είναι τα εξής:

- rawim,XYZ2Cam,wbcoeffs: οι έξοδοι της συνάρτησης readdng που αναλύθηκε παραπάνω

- bayertype: το πρότυπο Bayer της επιλογής μας $\{BGGR, GBRG, GRBG, RGGB\}$ σύμφωνα με τα μοτίβα του σχήματος 1.1
- method: η μέθοδος παρεμβολής της επιλογής μας $\{nearest, bilinear\}$
- M, N: οι διαστάσεις της νέας εικόνας που θέλουμε να παράξουμε από την αρχική $(M \times N)$
- Csrgb: η τελική εικόνα μετά τον μη-γραμμικό μετασχηματισμό της εικόνας *Clinear*
- Clinear: η εικόνα μετά τον μετασχηματισμό της *Cxyz* σύμφωνα με το πρωτόκολλο *CIE*
- Cxyz: η εικόνα μετά τον μετασχηματισμό της *Ccam* σύμφωνα με τον πίνακα μετασχηματισμού *XYZ2Cam*
- Ccam: η *rawim* μετά την διαδικασία παρεμβολής

Η συνάρτηση, αρχικά, πραγματοποιεί το white balancing που περιγράφεται στην ενότητα 1.1 με την χρήση της συνάρτησης *wbmask*(). Στη συνέχεια καλέι ανάλογα με το όρισμα της μεταβλητής *method* μία από τις συναρτήσεις *nearest*() και *bilinear*() που πραγματοποιούν αντίστοιχα τις δύο μεθόδους παρεμβολής της εικόνας που περιγράφονται στην ενότητα 1.2.

Επόμενο βήμα της συνάρτησης είναι η εφαρμογή των μετασχηματισμών από το *colorspace* της κάμερας στο *colorspace* της οθόνης. Για τον σκοπό αυτό ακολουθούμε τα βήματα που παρουσιάστηκαν στην ενότητα 1.3. Προσέχουμε πριν από κάθε μετασχηματισμό να κανονικοποιούμε τις γραμμές του πίνακα μετασχηματισμού ώστε, όπως προαναφέρθηκε, κάθε γραμμή να αθροίζει στο 1 κι έτσι να διατηρούνται ακέραια τα χρώματα της εικόνας. Για να το πετύχουμε αυτο χρησιμοποιούμε την εντολή $XYZ2Cam = XYZ2Cam./repmat(sum(XYZ2Cam,2),1,3)$; καθώς και τις αντίστοιχες εντολές για τους υπόλοιπους πίνακες μετασχηματισμού.

Τέλος, παρ΄όλο που δεν ζητείται από την εκφώνηση της εργασίας, για την ωραιότερη παρουσίαση της τελικής εικόνας πραγματοποιούμε ένα *brightnesscorrection* προσέχοντας πάντα για την αποκοπή όσων τιμών τύχει να βρεθούν έξω από το διάστημα [0, 1]. Ας δούμε, τώρα, λεπτομερέστερα την υλοποίηση των βοηθητικών συναρτήσεων.

2.2.1 wbmask

Η πρώτη βοηθητική συνάρτηση $colormask = wbmask(m, n, wbcoeffs, bayertype)$ χρησιμοποιείται στο white balancing που περιγράφεται στην ενότητα 1.1 κι έχει τα εξής ορίσματα και εξόδους:

- m, n: οι διαστάσεις της εικόνας που υπόκειται σε επεξεργασία
- wbcoeffs: το 1×3 διάνυσμα που περιέχει τους διορθωτικούς συντελεστές
- bayertype: το πρότυπο Bayer της επιλογής μας $\{BGGR, GBRG, GRBG, RGGB\}$ σύμφωνα με τα μοτίβα του σχήματος 1.1
- colormask: ο πίνακας μετασχηματισμού που θα χρησιμοποιηθεί για το *whitebalancing*

Ανάλογα, λοιπόν, με το όρισμα της μεταβλητής *bayertype* η συνάρτηση *wbmask*() κατασκευάζει μια μάσκα, έναν πίνακα δηλαδή μετασχηματισμού, η οποία θα πολλαπλασιαστεί στοιχείο προς στοιχείο με την εικόνα *rawim* ώστε να γίνει το white balancing.

2.2.2 **transform**

Η συνάρτηση $corrected = transform(im, cmatrix)$ χρησιμοποιείται από την $dng2rgb()$ για όλους τους μετασχηματισμους κι έχει ορίσματα και εξόδους:

- im: η εικόνα που θα υποστεί τον μετασχηματισμό
- cmatrix: ο 3×3 πίνακας μετασχηματισμού
- corrected: το αποτέλεσμα του μετασχηματισμού

Με άλλα λόγια, η συνάρτηση αυτή εφαρμόζει σε κάθε ***pixel*** της εικόνας τον πίνακα μετασχηματισμού ή πιο συγκεκριμένα, βρίσκει τα κατάλληλα βάρη της παλιάς χρωματικής παλέτας για να σχηματίσει την νέα παλέτα.

2.2.3 **nearest**

Η συνάρτηση $Ccam = nearest(wbim, bayertype, M0, N0, M, N)$ πραγματοποιεί την παρεμβολή με την μέθοδο Nearest Neighbor Interpolation που παρουσιάστηκε στην ενότητα 1.2.1. και έχει τα εξής ορίσματα και εξόδους:

- wbim: η εικόνα προς παρεμβολή
- M0,N0: οι παλιές διαστάσεις της αρχικής εικόνας (ύψος και πλάτος αντίστοιχα)
- M,N: οι νέες διαστάσεις της επεξεργασμένης εικόνας (ύψος και πλάτος αντίστοιχα)
- Ccam: το αποτέλεσμα της παρεμβολής

Η κύρια δυσκολία σε αυτήν την συνάρτηση είναι η προσαρμογή του πλαισίου της εικόνας στις νέες διαστάσεις, χωρίς να χαθεί η πληροφορία του χρώματος. Για να το πετύχει αυτό, η συνάρτηση κατασκευάζει ένα νέο ***grid*** συντεταγμένων $M \times N$ (για κάθε χρώμα), έτσι ώστε καθένα από τα τέσσερα γωνιακά σημεία του να συμπίπτουν με τα αντίστοιχα γωνιακά σημεία του ***grid*** της εικόνας εισόδου. Θεωρούμε ότι οι φυσικές διαστάσεις της εικόνας δεν μεταβάλλονται.

Υπολογίζουμε, πρώτα, το βήμα για κάθε διάσταση με το οποίο θα πρέπει να κινηθούμε πάνω στο νέο ***grid*** ώστε τα γειτονικά ***pixels*** ενός στοιχείου να βρίσκονται σε αναλογία με τα γειτονικά ***pixels*** του αρχικού ***grid***:

```
mstep = M0/M;
nstep = N0/N;
```

Καθώς, λοιπόν, επισκεπτόμαστε ένα ένα τα ***pixels*** του νέου ***grid*** ορίζουμε για καθένα απ' αυτά τις συντεταγμένες που αντιστοιχούν στο παλιό ***grid*** ως εξής:

```
m0 = round(1+(m-1)*mstep);
n0 = round(1+(n-1)*nstep);
```

Εξασφαλίζουμε, έτσι, ότι δεν θα χαθεί η πληροφορία για το χρώμα της εικόνας, αλλά τα ***pixels*** του νέου ***grid*** θα έχουν γείτονες σε αναλογία με την πυκνότητα που ορίζουν οι νέες διαστάσεις της εικόνας.

Κάνοντας, στη συνέχεια, έναν έλεγχο για το εάν μια συντεταγμένη του ***pixel*** είναι άρτιου ή περιττού αριθμού μπορούμε να ανατρέξουμε με δύο for loops σε ένα ένα στοιχείο του νέου ***grid*** και να εφαρμόσουμε τον αλγόριθμο που περιγράφεται στην ενότητα 1.2.1 ώστε να γεμίσουμε τα ***pixels*** που 'λείπουν' με το χρώμα των κοντινότερων γειτόνων τους, με βάση πάντα το ζητούμενο ***bayertype***.

2.2.4 **bilinear**

Η συνάρτηση $Ccam = bilinear(wbim, bayertype, M0, N0, M, N)$ πραγματοποιεί την παρεμβολή με την μέθοδο Bilinear Interpolation που παρουσιάστηκε στην ενότητα 1.2.2. και έχει τα εξής ορίσματα και εξόδους:

- wbim: η εικόνα προς παρεμβολή
- M0,N0: οι παλιές διαστάσεις της αρχικής εικόνας (ύψος και πλάτος αντίστοιχα)
- M,N: οι νέες διαστάσεις της επεξεργασμένης εικόνας (ύψος και πλάτος αντίστοιχα)
- Ccam: το αποτέλεσμα της παρεμβολής

Για την συνάρτηση αυτή ακολουθούμε ακριβώς την ίδια λογική με την συνάρτηση ***nearest***, δηλαδή προσαρμόζουμε το βήμα διάσχισης των ***pixels*** του νέου ***grid*** σε αντιστοιχία με το παλιό ***grid***. Στη συνέχεια, εφαρμόζουμε τα βήματα του αλγορίθμου που παρουσιάζεται στην ενότητα 1.2.2 και γεμίζουμε τα ***pixels*** που 'λείπουν' παρεμβάλοντας γραμμικά το χρώμα των κοντινότερων γειτόνων τους, με βάση πάντα το ζητούμενο ***bayertype***.

Η ιδιαιτερότητα αυτής της συνάρτησης σε σύγκριση με την προηγούμενη μέθοδο παρεμβολής είναι ότι θα πρέπει να λάβουμε υπόψην τις οριακές περιπτώσεις. Στις περιπτώσεις, δηλαδή, που ένα ***pixel*** βρίσκεται στο περιθώριο της εικόνας υπάρχει περίπτωση κάποιοι γείτονες να απουσιάζουν κι επομένως θα πρέπει η τιμή του χρώματος που 'λείπει' να προκύψει από την μέση τιμή των χρωμάτων των γειτόνων που υπάρχουν.

3 Αποτελέσματα

Στη συνέχεια θα παρουσιάσουμε τις εικόνες που προκύπτουν μετά από κάθε επεξεργασία και μετασχηματισμό ενός δείγματος "*RawImage.DNG*". Τα αποτελέσματα αυτά προκύπτουν τρέχοντας το *script demo.m*, το οποίο εμφανίζει την εικόνα κάθε σταδίου της επεξεργασίας μαζί με το ιστόγραμμα του κάθε καναλιού r, g, b. Για την προσομοίωση έχουμε επιλέξει διαστάσεις της τελικής εικόνας $M = 720$, $N = 1080$, το μοτίβο *RGGB* του προτύπου *Bayer* και γι΄ αρχή την μέθοδο παρεμβολής *bilinear*.

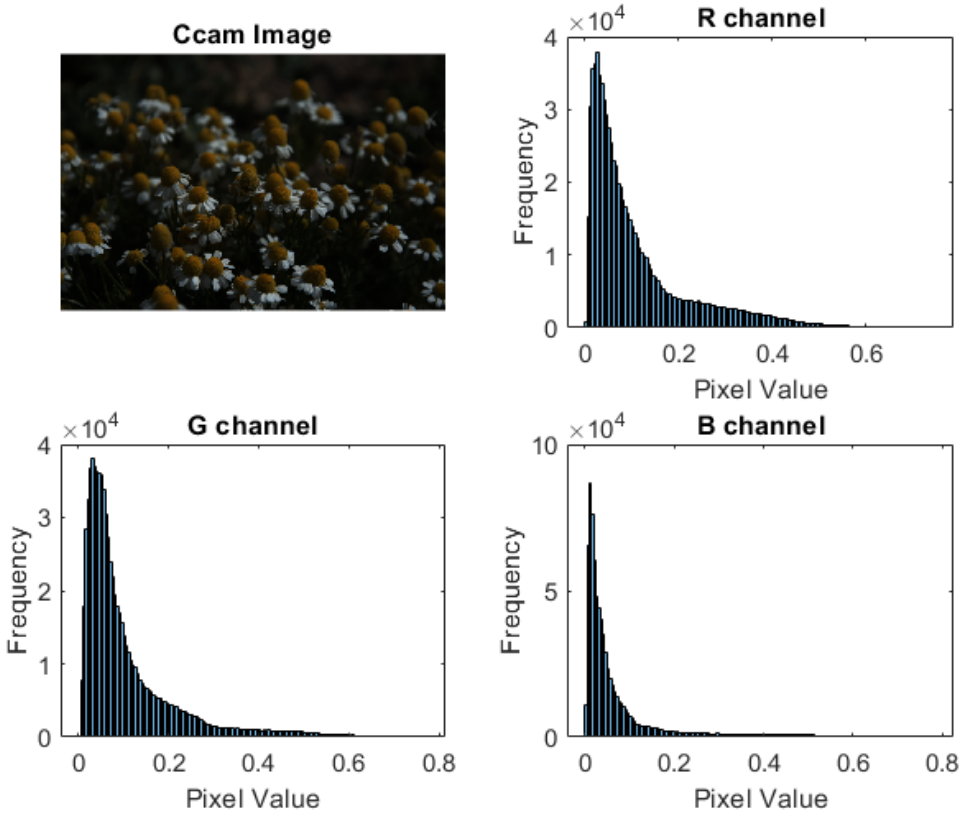
3.1 *Raw* εικόνα

Βλέπουμε, αρχικά, την *raw* εικόνα προς επεξεργασία (μετά την εξαγωγή των metadata). Παρατηρούμε ότι δεν περιέχει κανένα χρώμα, αλλά είναι *grayscale* και πολύ σκοτεινή:

Σχήμα 3.1: *Raw* εικόνα

3.2 *Ccam* εικόνα

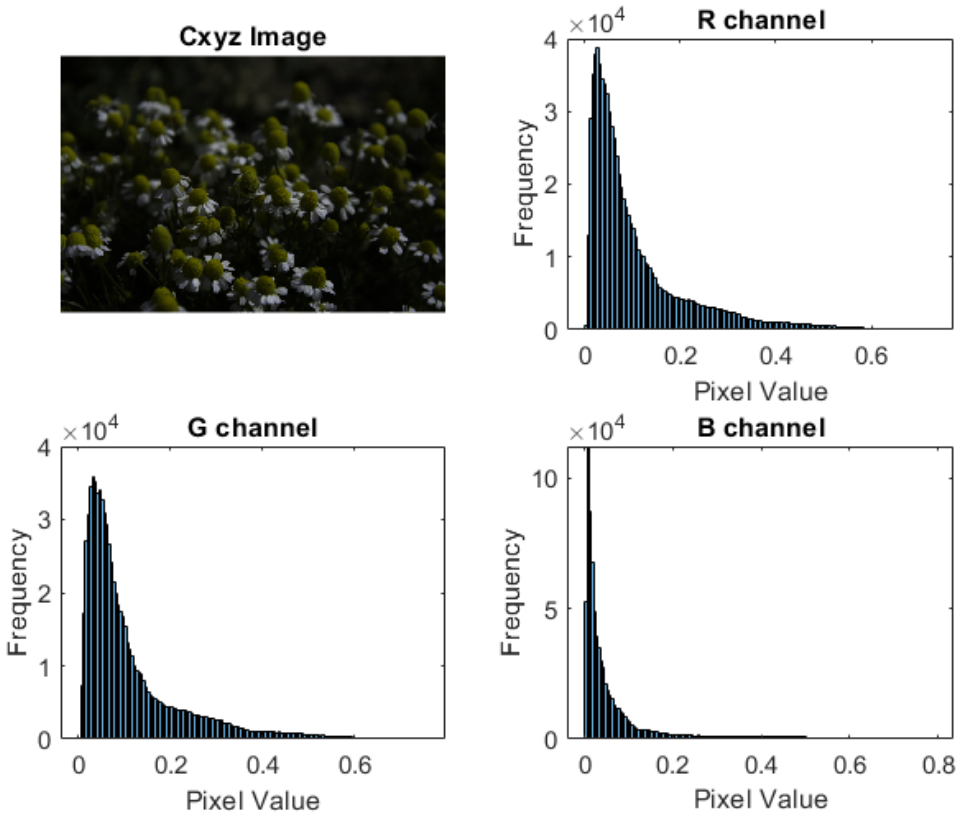
Μετά το *whitebalancing* και την παρεμβολή της *raw* εικόνας σύμφωνα με την μέθοδο *bilinear* προκύπτει η εικόνα *Ccam*. Παρατηρούμε ότι η εικόνα πλέον έχει χρώμα που φαίνεται να πλησιάζει στο επιθυμητό, ωστόσο είναι ακόμη εξαιρετικά σκοτεινή. Αυτό επιβεβαιώνεται και από τα ιστογράμματα για τα τρία κανάλια r, g, b, καθώς βλέπουμε ότι οι τιμές των χρωμάτων είναι συγκεντρωμένες με συντριπτικά μεγαλύτερη συχνότητα κοντά στο 0:



Σχήμα 3.2: *Ccam* εικόνα

3.3 *Cxyz* εικόνα

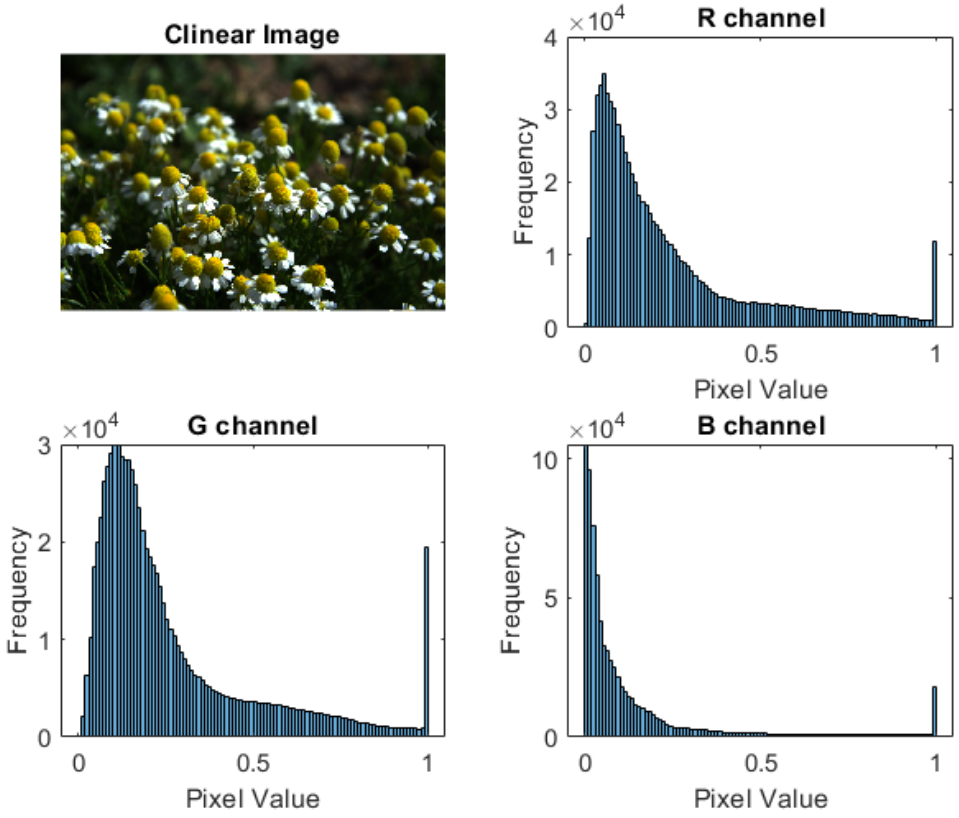
Μετά τον πρώτο γραμμικό μετασχηματισμό προκύπτει η εικόνα *Cxyz*. Παρατηρούμε ότι τα χρώματα φαίνονται λίγο πιο `σωστά', ωστόσο εξακολουθεί η εικόνα να είναι εξαιρετικά σκοτεινή, όπως φαίνεται και από τα ιστογράμματα:



Σχήμα 3.3: *Cxyz* εικόνα

3.4 *Clinear* εικόνα

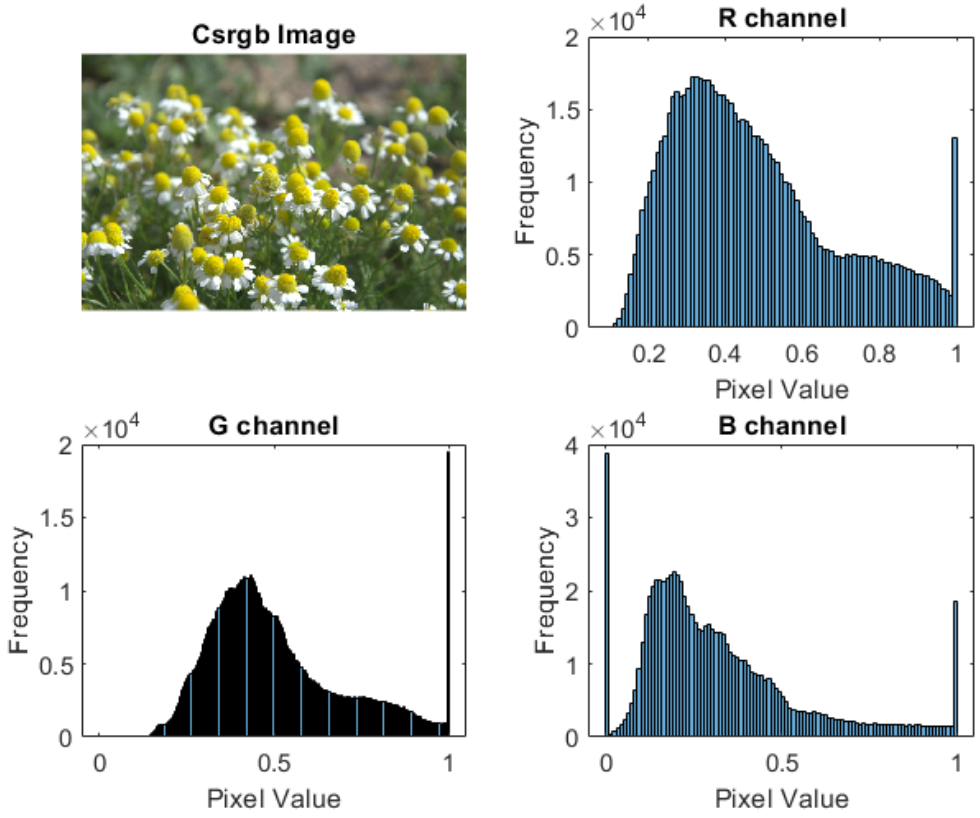
Μετά τον δεύτερο γραμμικό μετασχηματισμό προκύπτει η εικόνα ***Clinear***. Παρατηρούμε ότι επιτέλους τα χρώματα φαίνονται σωστά και το ***contrast*** της εικόνας έχει βελτιωθεί. Εξακολουθεί η εικόνα να είναι λίγο σκοτεινή, όμως βλέπουμε ότι η συχνότητα των τιμών των χρωμάτων έχει μετακινηθεί στα ιστογράμματα προς το κέντρο των τιμών:



Σχήμα 3.4: *Clinear* εικόνα

3.5 *Csrgb* εικόνα

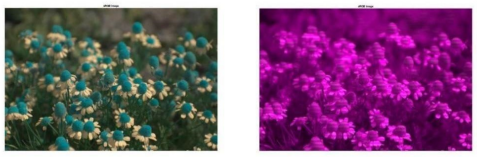
Τέλος, μετά και την διόρθωση της φωτεινότητας και τον μη-γραμμικό μετασχηματισμό της εικόνας βλέπουμε ένα αποτέλεσμα που φαίνεται να προσεγγίζει πολύ καλα την πραγματική εικόνα. Τονίζουμε, ωστόσο, ότι το αποτέλεσμα αυτό είναι αρκετά υποκειμενικό καθώς πειράζοντας τις παραμέτρους της διόρθωσης της φωτεινότητας ή του μη-γραμμικού μετασχηματισμού θα προκύψει ένα λίγο διαφορετικό αποτέλεσμα που θα μπορούσε να είναι πιο επιθυμητό. Δεν γνωρίζουμε το ακριβές χρώμα της φωτογραφίας γι΄ αυτό και επιλέγουμε τους μετασχηματισμούς με βάση την προσωπική μας προτίμηση. Βλέπουμε στα ιστογράμματα των τριών καναλιών της εικόνας πως οι συχνότητες των τιμών έχουν μετακινηθεί κι άλλο προς το κέντρο:



Σχήμα 3.5: *Clinear* εικόνα

3.6 Διαφορετικό *Bayer*

Ενδεικτικά, αναφέρουμε ότι με χρήση διαφορετικού μοτίβου του προτύπου ***Bayer***, όπως αυτό φαίνεται στο σχήμα 1.1, προκύπτουν εικόνες με λανθασμένο χρώμα. Εντοπίσαμε με δοκιμές το σωστό πρότυπο για την συγκεκριμένη φωτογραφία ως το ***RGGB***.



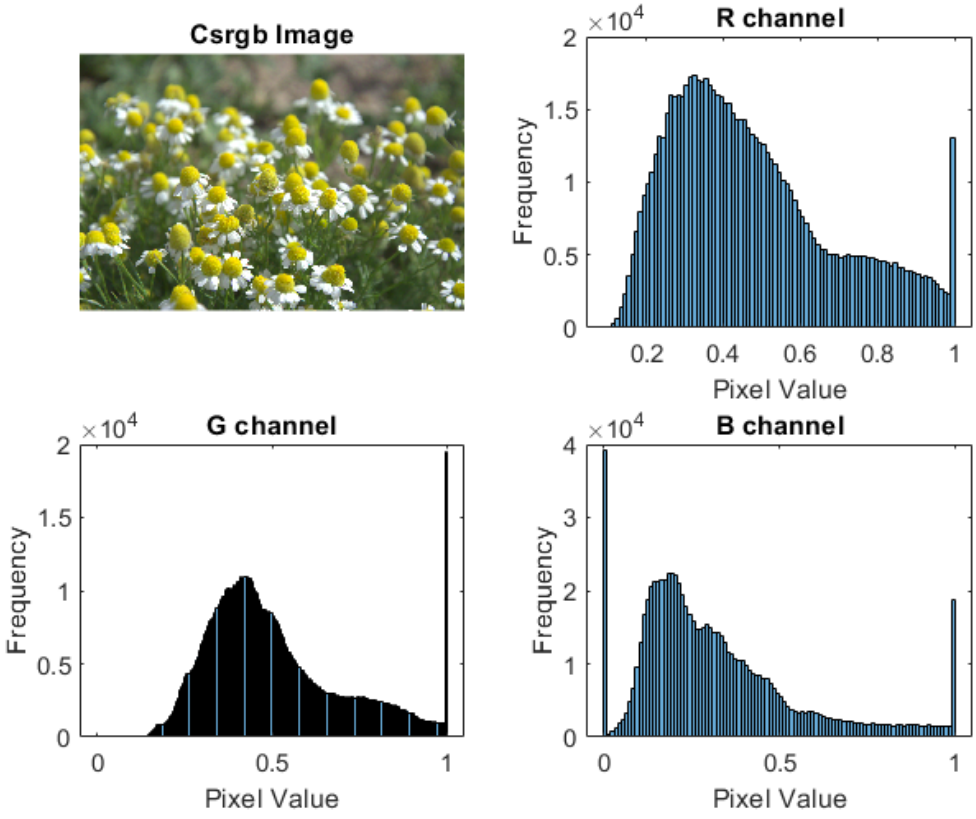
Σχήμα 3.6: *BGGR*, *GBRG*

Σχήμα 3.7: *GRBG*, *RGGB*

3.7 Μέθοδος *nearest*

Όπως αναφέραμε, στα παραπάνω χρησιμοποιήθηκε η μέθοδος ***bilinear*** για την παρουσίαση των εικόνων. Η μέθοδος ***nearest*** παρουσιάζει σχεδόν ακριβώς το ίδιο αποτέλεσμα με απει-

ροελάχιστες διαφορές. Εάν η εικόνα μεγενθυθεί παρατηρούμε ότι με τη μέθοδο ***bilinear*** οι μεταβάσεις απο ***pixel*** σε ***pixel*** είναι κάπως πιο ομαλές. Αυτό μπορεί να επιβεβαιωθεί κι από τα ιστογράμματα των τριών καναλιών τα οποία είναι ελάχιστα πιο ομοιόμορφα κατανεμημένα:



Σχήμα 3.8: Μέθοδος *nearest*

Βιβλιογραφία

[1] Rob Sumner (2014), *Processing RAW Images in MATLAB*, https://rcsumner.net/raw_guide/RAWguide.pdf

[2] *OpenCine.Nearest Neighbor and Bilinear Interpolation*, https://wiki.apertus.org/index.php?title=OpenCine.Nearest_Neighbor_and_Bilinear_Interpolation