Εργασία 1

Ψηφιακή Επεξεργασία Εικόνας 2018

Μελεζιάδης Ιωάννης

ΑΕΜ 8760

1. Συνέλιξη εικόνας με εικόνα

myConvSpat()

Σημείωση: Για την καλύτερη κατανόηση της λειτουργείας του αλγορίθμου υπάρχουν σχόλια μέσα στον κώδικα. Έτσι ο αναγνώστης καλείται να έχει ανοιγμένο το αρχείο *myConvSpat.m* συγχρόνως με το παρών έγγραφο ώστε να βοηθηθεί. Επιπλέον έχουν αφεθεί σκόπιμα σε σχόλια εντολές που χρησιμοποιήθηκαν στο debugging του κώδικα.

Επεξήγηση: Για το χωρικό φιλτράρισμα δυο εικόνων χρησιμοποιώ γέμισμα του πίνακα της πρώτης εικόνας με μηδενικά, στη συνέχεια κάνω τη συνέλιξη και στο τέλος κάνω crop στην τελική εικόνα ώστε η imOut να έχει το ίδιο μέγεθος με την imX. Χρησιμοποιώ την εντολή *paddingx=size(imY,1)-1;* ακολουθώντας τη θεωρία από το βιβλίο(του Gonzalez) που αναφέρει ότι θέλω τουλάχιστον N-1 γραμμές πάνω και κάτω του imX για να επιτευχθεί το γέμισμα με μηδενικά (N=γραμμές του πίνακα imY). Στην συνέχεια κάνω αναστροφή του πίνακα imY για να κάνω συνέλιξη και όχι συσχέτιση. Ακολουθεί ένα διπλό for loop στο οποίο πραγματοποιώ συνέλιξη για κάθε στοιχείο που δεν έχει δημιουργηθεί από γέμισμα. Προσέχω ώστε το μέγεθος των πινάκων, που πολλαπλασιάζω τα στοιχεία τους, να είναι το ίδιο και στη συνέχεια τους μετατρέπω σε διανύσματα για να κάνω εσωτερικό γινόμενο. Τέλος κάνω το crop και αποθηκεύω το αποτέλεσμα στον imOut.

Το αποτέλεσμα της εντολής :

```
A=myConvSpat(im1_gray,im2_gray); imshow(A);
```

(Το αποτέλεσμα είναι μόνο η άσπρη εικόνα και όχι το γκρι περίγραμμα, το πρόσθεσα για να φαίνεται). Έχει διαστάσεις 251x221.

myConvFreq()

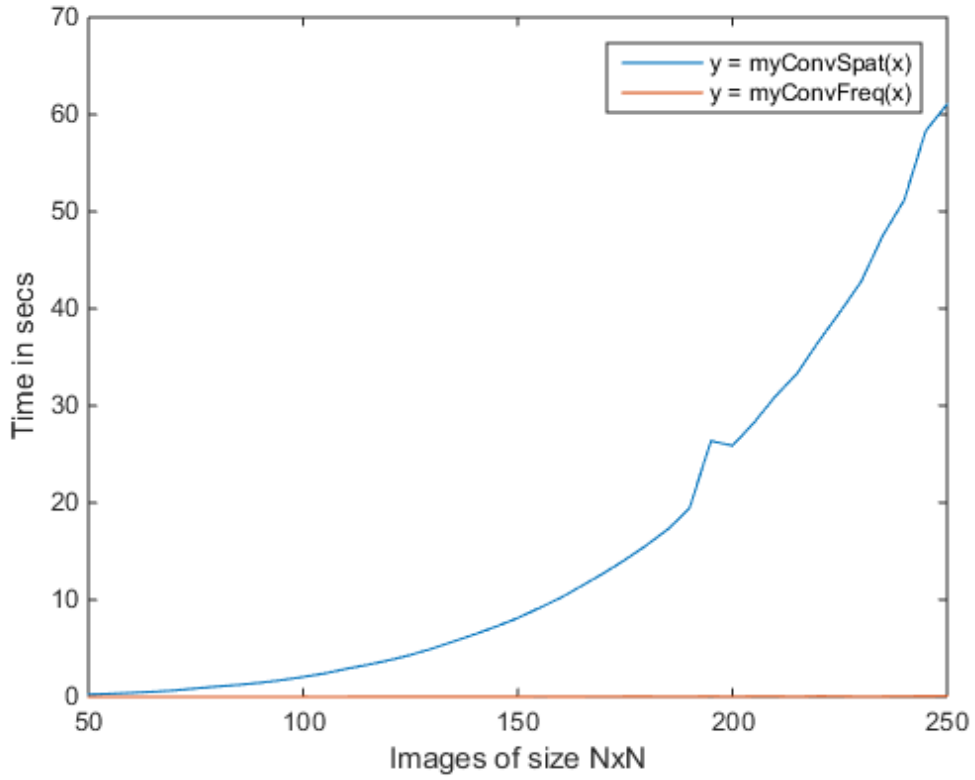
Επεξήγηση: Αρχικά υπολογίζω το διδιάστατο Μ/Σ Φουριέ των εικόνων imX και imY που έχω κάνει zero padding στους πίνακες. Αυτό επιτυγχάνεται με τη χρήση της fft2 της matlab. Στη συνέχεια πολλαπλασιάζω στοιχείο προς στοιχείο τους δυο πίνακες και στο αποτέλεσμα χρησιμοποιώ ifft2 (αντίστροφο Μ/Σ Φουριέ). Το αποτέλεσμα του ifft2 χρησιμοποιείται από την real() του matlab για να παραχθεί ο τελικός πίνακας μετά από συνέλιξη των δυο εικόνων στο πεδίο της συχνότητας.

Το αποτέλεσμα της εντολής:

```
B=myConvFreq(im1_gray,im2_gray); imshow(B);
```

Είναι το ίδιο με το αποτέλεσμα παραπάνω.

Μετρώντας τον χρόνο σε δευτερόλεπτα για συνέλιξη εικόνων μεγέθους 50x50 έως 250x250 παρατήρησα ότι η συνέλιξη στη συχνότητα είναι κατά πολύ γρηγορότερη από τη συνέλιξη στο χώρο, κάτι το οποίο φαίνεται και από το γράφημα που προέκυψε ως αποτέλεσμα του *demo1.m* :



2. Συνέλιξη εικόνας με μάσκα

myFilter()

Σημείωση: Υπάρχουν και εδώ σχόλια στον κώδικα οπότε ο αναγνώστης καλείται να ανοίξει το αρχείο *myFilter*

Επεξήγηση: Το αρχείο υλοποιεί φιλτράρισμα μιας εικόνας με μια μάσκα χρησιμοποιώντας την *myConvSpat*.Υποστηρίζει τρεις λειτουργίες και τυπώνει μήνυμα σφάλματος για μία άλλη οι οποίες παρουσιάζονται παρακάτω:

1. Εικόνα 3-κανάλια -> μάσκα 1-κανάλι

2. Εικόνα 3-κανάλια -> μάσκα 3-κανάλια

3. Εικόνα 1-κανάλι -> μάσκα 1-κανάλι

4. Εικόνα 1-κανάλι -> μάσκα 3-κανάλια Δεν υφίσταται θεωρητικά και απορρίπτεται απο τον αλγόριθμο.

Για την $1^{η}$ περίπτωση σπάω την εικόνα σε τρεις διδιάστατους πίνακες που αντιπροσωπεύουν τα κανάλια RGB και εφαρμόζω στο καθένα το φιλτράρισμα με τη μάσκα που δόθηκε.Τέλος συνενώνω τα τρία κανάλια ώστε να εμφανιστεί ξανά η RGB εικόνα.

Για τη $2^{η}$ περίπτωση σπάω και την εικόνα και τη μάσκα στα 3 επιμέρους κανάλια τους και κάνω συνέλιξη αυτα που αντιστοιχούν μεταξύ τους. Τέλος τα συνενώνω και βγαίνει πάλι η εικόνα RGB.

Για τη $3^{η}$ περίπτωση απλά κάνω τη συνέλιξη κατευθείαν αφού πρόκειται για grayscale εικόνα με μονοκάναλη μάσκα.

Για τη $4^{η}$ περίπτωση εμφανίζω κατάλληλο μήνυμα σφάλματος.

Τα αποτελέσματα του demo2.m είναι τα εξής μαζί με τις διαστάσεις τους για τις εντολές 1 και 3 (οι εντολές 2 και 4 έβγαλαν error καθώς δεν ορίζεται η συνέλιξη)

Αποτέλεσμα εντολής 1(101x121x3 double)

Αποτέλεσμα εντολής 3(151x101x3 double)

3 Motion blur

myMotBlur()

Σημείωση: Υπάρχουν και εδώ σχόλια στον κώδικα οπότε ο αναγνώστης καλείται να ανοίξει το αρχείο *myMotBlur.m*

Επεξήγηση: Το πρόγραμμα λειτουργεί με τον περιορισμό ότι το exposure_time πρέπει να ανήκει στο διάστημα (0,1]. Στην αρχή κάνω έλεγχο για το αν πληρείται η

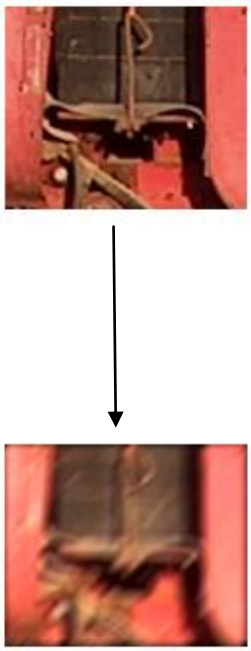
προυπόθεση και μετά συνεχίζω υπολογίζοντας τις μοίρες μέσω του διανύσματος *mot_vec.*Αυτό επιτυγχάνεται χρησιμοποιώνας τις συναρτήσεις *atan2()* και *radtodeg()* της matlab. Το υπόλοιπο κομμάτι του κώδικα επιλέγει το είδος μάσκας που θα δημιουργηθεί ανάλογα με το πόσες μοίρες δόθηκαν σαν κατεύθυνση. Σε κάθε περίπτωση ανάλογα με την τιμή του *exposure_time* κατηγοριοποιώ το blurring σε γρήγορο ή αργό. Συγκεκριμένα αν *exposure_time>0.5* τότε το θεωρώ γρήγορο αλλιώς αργό. Οι τιμές που δόθηκαν στις μάσκες βγήκαν με πολλές δοκιμές και παρατήρηση των αλλαγών.

Ακολουθούν δυο ενδεικτικά αποτελέσματα της συνάρτησης *myMotBlur()* :

Εντολή 1:

```
A = myMotBlur(im1_rgb,[1 1],0.6); imshow( A );
```

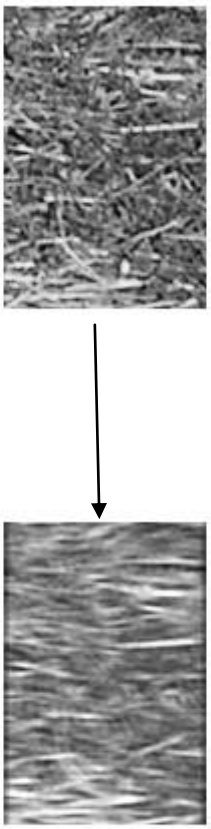
Οι τιμές του διανύσματος [1 1] αντιστοιχούν σε 45 μοίρες και η τιμή 0.6 αντιστοιχεί σε γρήγορο blurring :



Εντολή 2:

```
B = myMotBlur(im2_gray,[1 0],0.6); imshow(B);
```

Οι τιμές του διανύσματος [1 0] αντιστοιχούν σε 90 μοίρες και η τιμή 0.6 αντιστοιχεί σε γρήγορο blurring :



Σημείωση: Οι διαστάσεις των εικόνων μετά το blurring είναι οι ίδιες με τις αρχικές εικόνες.