Ψηφιακή Επεξεργασία Εικόνας
-Εργασία 3-

—

Image Registration

Α. Ντελόπουλος

Άνοιξη 2023

1 Εισαγωγικά

Στην πρώτη ενότητα της εργασίας θα υλοποιήσετε μια σουίτα από ρουτίνες (Ενότητες 1.1, 1.2 και 1.3) τις οποίες θα πρέπει να χρησιμοποιήσετε για να δώσετε λύση στο τελικό, real life, πρόβλημα (Ενότητα 2). Πιο συγκεκριμένα, θα υλοποιήσετε:

1. Έναν απλό δικό σας τοπικό περιγραφέα (local descriptor)
2. Τον Harris corner detector
3. Αντιστοίχιση σημείων με τη μέθοδο RANSAC

1.1 Ένας Local Descriptor

Η εργαλειοθήκη ξεκινάει με την κατασκευή μιας ρουτίνας που υλοποιεί τον υπολογισμό ένός απλού rotation invariant περιγραφέα της γειτονιάς ενός σημείου $p = [p_1,\ p_2]$. Ο συγκεκριμένος local descriptor κατασκευάζεται σαρώνοντας διαδοχικούς ομόκεντρους κύκλους με κέντρο το p και ακτίνες $\rho = \rho_m : \rho_s : \rho_M$ όπου τόσο η μικρότερη/μεγαλύτερη ακτίνα όσο και το βήμα ρ_s είναι παράμετροι του αλγορίθμου υπολογισμού. Οι κύκλοι σαρώνονται σε $2\pi/N$ σημεία και έτσι για καθέναν από τους κύκλους υπολογίζεται ένα διάνυσμα $x_\rho = [x_{\rho,0}, \ldots, x_{\rho,N-1}]$. Τα στοιχεία του διανύσματος υπολογίζονται από παρεμβολή των στοιχείων της εικόνας με βάση τη θέση στην οποία αντιστοιχούν. Η βασική έκδοση τους περιγραφέα είναι ένα διάνυσμα d με τόσα στοιχεία όσα οι κύκλοι $((\rho_M - \rho_m)/\rho_s)$ που το καθένα έχει τιμή ίση προς το μέσο όρο του αντίστοιχου x_ρ.

Να κατασκευάσετε τη συνάρτηση:

```
function d = myLocalDescriptor(I,p,rhom,rhoM,rhostep,N)
```

η οποία λαμβάνει ως είσοδο μία grayscale εικόνα `I` και τις τιμές των παραπάνω παραμέτρων `rhom,rhoM,rhostep, N` και επιστρέφει το διάνυσμα του περιγραφέα για τη γειτονιά του σημείου `p`. Για ευκολία είναι αποδεκτό η συνάρτηση να επιστρέφει το κενό διάνυσμα αν το `p` βρίσκεται τόσο κοντά στα όρια της εικόνας που ο μεγαλύτερος κύκλος ακτίνας `rhoM` φτάνει έξω από την εικόνα.

Να κατασκευάσετε τη συνάρτηση:

```
function d = myLocalDescriptorUpgrade(I,p,rhom,rhoM,rhostep,N)
```

στην οποία θα αντικαταστήσετε την βασική εκδοχή του περιγραφέα με άλλη δικής σας έμπνευσης που παραμένει rotation invariant αλλά περιέχει πλουσιότερη πληροφορία. Δεν είναι απαραίτητο ο νέος descriptor να έχει το ίδιο μήκος με αυτόν της βασικής έκδοσης.

1.1.1 Παραδοτέα

Να δείξετε:

1. Την βασική έκδοση του περιγραφέα του pixel $p = [100, 100]$ για την αρχική εικόνα πίνακα testimg1.png, του αντίστοιχου pixel μετά την περιστροφή κατά θ_1 και θ_2 όπως παραπάνω. Οι υπολογισμοί να γίνουν με `rhom=5,rhoM=20,rhostep=1, N=8`
2. Την βασική έκδοση του περιγραφέα των pixels $q = [200, 200]$ και $q = [202, 202]$ για την αρχική εικόνα πίνακα testimg1.png με τις ίδιες τιμές παραμέτρων.
3. Να επαναλάβετε τα παραπάνω για τον δικό σας περιγραφέα

1.2 Harris corner detector

Για τον εντοπισμό σημείων ενδιαφέροντος θα υλοποιήσετε τον αλγόριθμο Harris corner detector που πρωτοπροτάθηκε στο το άρθρο [1]. Ακολουθεί η βασική ιδέα. Ας υποθέσουμε ότι $w(x_1, x_2)$ είναι μία διδιάστατη συνάρτηση που έχει μή μηδενικές τιμές κοντά στην αρχή των αξόνων και "πεθαίνει" καθώς το $(x_1,\ x_2)$ απομακρύνεται από το $(0,\ 0)$. Για παράδειγμα:

$$w(x_1, x_2) \ = \ exp\{-\frac{x_1^2 + x_2^2}{2\sigma^2}\} \tag{1}$$

Αν $I(x_1, x_2)$ είναι η φωτεινότητα μιας gray scale εικόνας, τότε η συνάρτηση

$$E(x_1, x_2; p_1, p_2) \ = \ \sum_{u_1,u_2} w(u_1, u_2)\|I(p_1 + u_1 + x_1, p_2 + u_2 + x_2) - I(p_1 + u_1, p_2 + u_2)\|^2 \tag{2}$$

παρουσιάζει την εξής συμπεριφορά:

- Αν το σημείο (p_1, p_2) βρίσκεται σε μία σχετικά ομαλή περιοχή της εικόνας τότε $E(x_1, x_2; p_1, p_2) \approx 0$.
- Αν το (p_1, p_2) βρίσκεται πάνω ή πολύ κοντά σε μία ακμή η οποία - σε σχέση με το μέγεθος της μη μηδενικής περιοχής του παραθύρου $w(x_1, x_2)$ - είναι ευθύγραμμη, τότε $E(x_1, x_2; p_1, p_2) \approx 0$ όταν η ολίσθηση $(x_1,\ x_2)$ είναι (σχεδόν) παράλληλη με την ακμή ενώ θα παίρνει τιμές $>> 0$ όταν η ολίσθηση $(x_1,\ x_2)$ είναι προς την κάθετη στην ακμή κατεύθυνση.
- Αν το (p_1, p_2) βρίσκεται πάνω ή πολύ κοντά σε μία γωνία τότε $E(x_1, x_2; p_1, p_2) >> 0$ για οποιαδήποτε κατεύθυνση της ολίσθησης $(x_1,\ x_2)$.

Συνεπώς η μελέτη της $E(x_1, x_2; p_1, p_2)$ ως συνάρτηση των $(x_1,\ x_2)$ μας επιτρέπει να αποφασίζουμε αν το σημείο $(p_1,\ p_2)$ βρίσκεται σε περιοχή ομαλής φωτεινότητας, σε ακμή ή σε γωνία. Χρησιμοποιώντας ανάπτυγμα Taylor γύρω από το σημείο $(p_1 + u_1, p_2 + u_2)$ (ως προς τις μεταβλητές (x_1, x_2), η $E(x_1, x_2; p_1, p_2)$ γράφεται

$$E(x_1, x_2; p_1, p_2) \approx \sum_{u_1,u_2} w(u_1, u_2)\|x_1 I_1(p_1 + u_1, p_2 + u_2) + x_2 I_2(p_1 + u_1, p_2 + u_2)\|^2 \tag{3}$$

όπου $I_1(x_1, x_2) = \partial I(x_1, x_2)/\partial x_1$, $I_2(x_1, x_2) = \partial I(x_1, x_2)/\partial x_2$ και έχουμε παραλήψει τους όρους υψηλότερης τάξης. Εναλλακτικά,

$$E(x_1, x_2; p_1, p_2) \approx [x_1,\ x_2]\, \mathbf{M}(p_1, p_2) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{4}$$

όπου

$$\mathbf{M}(p_1, p_2) = \sum_{u_1,u_2} w(u_1, u_2)\mathbf{A}(u_1, u_2; p_1, p_2) \tag{5}$$

και

$$\mathbf{A}(u_1, u_2; p_1, p_2) = \begin{bmatrix} I_1(p_1 + u_1, p_2 + u_2)^2 & I_1(p_1 + u_1, p_2 + u_2)I_2(p_1 + u_1, p_2 + u_2) \\ I_1(p_1 + u_1, p_2 + u_2)I_2(p_1 + u_1, p_2 + u_2) & I_2(p_1 + u_1, p_2 + u_2)^2 \end{bmatrix} \tag{6}$$

Πρακτικά οι μερικές παράγωγοι που συμμετέχουν στις παραπάνω εκφράσεις υπολογίζονται με τη χρήση κατάλληλων συνελικτικών μασκών όπως έχουμε εξηγήσει στη θεωρία. Ο υπολογισμός τους μάλιστα γίνεται μια κι έξω για όλη την εικόνα. Οι τρείς εναλλακτικές συμπεριφορές της συνάρτησης $E(x_1, x_2; p_1, p_2)$ που είδαμε παραπάνω αντιστοιχούν σε τρεις αντίστοιχες διαφορετικές εκδοχές για τις ιδιοτιμές του πίνακα $\mathbf{M}(p_1, p_2)$:

- Αν το σημείο (p_1, p_2) βρίσκεται σε μία σχετικά ομαλή περιοχή της εικόνας τότε $\lambda_1 \approx 0$ και $\lambda_1 \approx 0$.
- Αν το (p_1, p_2) βρίσκεται πάνω ή πολύ κοντά σε μία ακμή τότε $\lambda_1 >> \lambda_2 \approx 0$.
- Αν το (p_1, p_2) βρίσκεται πάνω ή πολύ κοντά σε μία γωνία τότε $\lambda_1 | approx \lambda_2 >> 0$.

Συνεπώς η απόφαση για το αν η περιοχή του σημείου (p_1, p_2) είναι ομαλή, βρίσκεται σε ακμή ή σε γωνία μπορεί να ληφθεί από τη μελέτη των δύο ιδιοτιμών του πίνακα $\mathbf{M}(p_1, p_2)$. Για λόγους υπολογιστική πολυπλοκότητας - καθόσον ο υπολογισμός ιδιοτιμών απαιτεί υπολογισμό τετραγωνικών ριζών - χρησιμοποιούμε την παρακάτω μετρική γωνιότητας:

$$R(p_1, p_2) = det(\mathbf{M}(p_1, p_2)) - kTrace(\mathbf{M}(p_1, p_2))^2 \tag{7}$$

όπου $k > 0$ μια οριζόμενη από εμάς παράμετρος. Επειδή

$$R(p_1, p_2) = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \tag{8}$$

η τιμή R θα είναι θετική όταν το σημείο (p_1, p_2) είναι κοντά σε γωνίες, αρνητική στη γειτονιά ακμών και κοντά στο μηδέν σε ομοιόμορφες περιοχές. Κατάλληλα κατώφλια μπορούν να χρησιμοποιηθούν για να επιλεγούν οι πλέον ευδιάκριτες γωνίες.

Κατασκευάστε την συνάρτηση:

```
function c = isCorner(I, p, k, Rthres)
```

η οποία επιστρέφει logical true/false αν η grayscale εικόνα I εμφανίζει γωνία στο pixel $p = [p_1, p_2]$. Η παράμετρος εισόδου k αντιστοιχεί στην παράμετρο k της εξ. 7 και η παράμετρος εισόδου $Rthres$ ορίζει το κατώφλι πάνω από το οποίο η τιμή της $R(p_1, p_2)$ θεωρείται αρκετά θετική ώστε να εντοπίζεται γωνία.

Με χρήση της συνάρτησης αυτής να κατασκευαστεί η συνάρτηση

```
function corners = myDetectHarrisFeatures(I)
```

η οποία υλοποιεί τον αλγόριθμο, όπως περιγράφεται παραπάνω. Η μεταβλητή `I` είναι ένας πίνακας 2 διαστάσεων και περιέχει μία εικόνα σε gray scale, με τιμές πραγματικούς αριθμούς στο διάστημα $[0, 1]$. Η μεταβλητή `corners` είναι ένας πίνακας δύο στηλών, και κάθε του γραμμή αντιστοιχεί στις συντεταγμένες μίας γωνίας. Ο αλγόριθμος είναι ήδη υλοποιημένος στο Image Processing toolbox της MATLAB. Οι παρακάτω εντολές επιδεικνύουν τη χρήση του:

```
I=imread('cameraman.tif');
corners=detectHarrisFeatures(I);
figure
hold on
imshow(I);
plot(corners);
hold off
```

Μπορείτε να χρησιμοποιήσετε τη μεταβλητή `corners.Location` για να επαληθεύσετε την υλοποίησή σας συγκρίνοντας τα δικά σας αποτελέσματα με αυτά της έτοιμης συνάρτησης. Η ορθότητα της υλοποίησης δεν εξαρτάται προφανώς από την σειρά με την οποία παρατίθενται οι γωνίες.

1.3 Matching descriptors

Αφού βρείτε τα ιδιάζοντα σημεία και τους περιγραφείς τους από τις δύο εικόνες, πρέπει στη συνέχεια να αντιστοιχίσετε αυτά τα ανιχνευμένα σημεία μεταξύ των δύο εικόνων. Για να το πετύχετε αυτό, μπορείτε να υπολογίσετε τις Ευκλείδειες αποστάσεις των περιγραφέων για όλα τα ζευγη ανιχνευμένων σημείων και να δημιουργήσετε τον πίνακα αποστάσεων, με διαστάσεις $N_1 \times N_2$, όπου N_i το πλήθος ανιχνευμένων σημείων της $i = 1, 2$ εικόνας. Με βάση αυτόν τον μη συμμετρικό πίνακα αποστάσεων, μπορούμε να δούμε πόσο ”καλά” αντιστοιχίζεται ενα ανιχνευμένο σημείο της 1ης εικόνας με κάποιο ανιχνευμένο σημείο της 2ης εικόνας. Η μέθοδος αυτή περιλαμβάνει έλεγχο και αρα υπολογισμό όλων των δυνατών συνδυασμών ζευγών ανιχνευμένων σημείων και χαρακτηρίζεται ως exhaustive search. Για να περιορίσουμε το πλήθος των πιθανών ζευγών σημείων μπορούμε να εφαρμόσουμε ένα κατώφλι στην απόσταση των περιγραφέων. Συγκεκριμένα θέλουμε να κρατήσουμε ενα ποσοστό, percentageThreshold (πχ 30%), των πιθανών

ζευγών σημείων, οπότε θα εφαρμόσουμε το κατάλληλο κατώφλι που διατηρεί αυτό το επιλεγένο ποσοστό από τα $N_1 \times N_2$ πιθανά ζεύγη σημείων. Παρατηρήστε ότι ένα σημείο της μιας εικόνα μπορεί να αντιστοιχηθεί ”καλά” σε πολλαπλά σημεία της άλλης εικόνας, ενώ κάποιες αντιστοιχίσεις μπορεί να είναι εσφαλμένες (outliers).

Κατασκευάστε την συνάρτηση:

```
function matchingPoints = descriptorMatching(points1, points2,
    percentageThreshold)
```

η οποία επιστρέφει έναν πίνακα $2 \times n$ με τα ζεύγη ανιχνευμένων σημείων που είναι πιο πιθανό να εχουν αντιστοιχηθεί. Δηλαδή ο πίνακας matchingPoints περιλαμβάνει ζεύγη δεικτών στις λίστες points1 και points2.

RANSAC

Για να βρείτε το μετασχηματισμό περιστροφής κατά γωνία θ και μετατόπισης κατά διάνυσμα $\vec{d}$ αρκεί να εχετε 2 σημεία που να αντιστοιχίζονται σωστά στις εικόνες, δηλαδή 2 ζεύγη σημείων. Αν P_1, P_2 είναι οι συντεταγμένες του ίδιου σημείου στις δύο εικόνες τότε υποθέστε ότι

$$P_2 = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} P_1 + \vec{d}$$

Θεωρούμε $H = \{\theta, \vec{d}\}$, το οποίο αναπαριστά το μετασχηματισμό περιστροφής και μετατόπισης ώστε τα κοινά τους σημεία να συμπίπτουν.

Αντί να χρησιμοποιήσετε τα δύο ”καλύτερα” ζεύγη σημείων καλείστε να υλοποιήσετε μια πιο εύρωστη τεχνική που λαμβάνει υπόψη όλα τα ζεύγη σημείων που έχετε βρει να αντιστοιχίζονται ”καλά”, αγνοώντας όσα δε συμφωνούν (outliers). Για να το κάνετε αυτό θα χρειαστεί να υλοποιήσετε τον αλγόριθμο RANdom Sample Consensus (RANSAC). Η εκτίμηση της βέλτιστης λύσης προκύπτει από το μέγεθος του «σετ συμφωνίας» (consensus set), δηλαδή επιλέγεται ως τελικό μοντέλο αυτό στο οποίο συμφωνεί το μεγαλύτερο σετ δεδομένων.

Επαναλαμβανόμενα βήματα RANSAC:

1. Επίλεξε τυχαία δύο ζεύγη σημείων (random minimal set)
2. Υπολόγισε το μετασχηματισμό H που ορίζεται από αυτά τα δύο ζεύγη
3. Υπολόγισε το score ως το άθροισμα των Ευκλείδειων αποστάσεων ανάμεσα στα υπόλοιπα ζεύγη σημείων που προκύπτουν μετά την εφαρμογή του μετασχηματισμού .

Ο αλγόριθμος επαναλαμβάνεται N φορές και έχει ως παράμετρο την ακτίνα r, που χρησιμοποιείται ως κριτήριο κατάταξης του κάθε ζεύγους σημείων σε inlier ή outlier. Στο τέλος των N επαναλήψεων επιλέγεται ο μετασχηματισμός με το βέλτιστο score και γίνεται ο διαχωρισμός των inliers, outliers με βάση την ακτίνα r.

Κατασκευάστε την συνάρτηση:

```
function {H, inlierMatchingPoints, outlierMatchingPoints} = myRANSAC(
    matchingPoints, r, N)
```

η οποία επιστρέφει το βέλτιστο μετασχηματισμό H που προκύπτει από την είσοδο matchingPoints (δηλαδή H.theta $= \theta$ και H.d $= \vec{d}$), καθώς επίσης και τα ζεύγη σημείων που θεωρήθηκαν inliers και outliers. Οι μονοδιάστατοι πίνακες inlierMatchingPoints, outlierMatchingPoints περιλαμβάνουν δείκτες στα στοιχεία του πίνακα matchingPoints και το άθροισμα των μηκών τους είναι ίσο με το μήκος της πίνακα matchingPoints, n.

1.3.1 Παραδοτέα

Να δείξετε:

1. Τα σημεία που εντοπίσατε πάνω στις δύο εικόνες και τη μεταξύ τους αντιστοίχιση. Σημειώστε με γκρι τετράγωνο τα ανιχνευμένα σημεία που αποτελούν outliers στη μεταξύ τους αντιστοίχιση και με διαφορετικά χρώματα, κοινά ανά ζεύγη ανιχνευμένων σημείων, τα σημεία που αντιστοιχίζονται ”καλά” μεταξύ των δύο εικόνων. Αν είναι πολλά αυτά τα ζεύγη σημείων μπορείτε να απεικονίσετε τα inliers και outliers ξεχωριστά.

2 Το πρόβλημα

Οι εικόνες που χρησιμοποιούνται στο εθνικό κτηματολόγιο μπορεί να προέρχονται από διάφορες πηγές, όπως αεροφωτογραφίες, δορυφορικές εικόνες ή επίγειες έρευνες. Οι εικόνες αυτές μπορεί να παρουσιάζουν σημαντικές διαφοροποιήσεις όσον αφορά το μέσο λήψης, τη χωρική ανάλυση και τις συνθήκες λήψης. Κατά συνέπεια, η ευθυγράμμιση και η συνένωση τέτοιων διαφορετικών εικόνων σε μια ενιαία για τη δημιουργία μιας συνεκτικής αναπαράστασης της γης αποτελεί δύσκολο έργο.

Σε αυτή την ενότητα θα χρησιμοποιήσετε τις συναρτήσεις που έχετε φτιάξει σε αυτή την εργασία για να λύσετε το παρακάτω πρόβλημα. Οι εικόνες im1 και im2 τραβήχτηκαν διαδοχικά από μία κάμερα τηλεπισκόπησης που φωτογράφισε μια αστική περιοχή. Στόχος είναι να τις ενώσετε σε μία αφού περιστρέψετε κατάλληλα τη μια από αυτές.

Φτιάξτε τη συνάρτηση `Im=myStitch(im1,im2)` η οποία θα ενώνει την εικόνα im2 στην εικόνα im1 και παρουσιάστε τα αποτελέσματα.

Δοκιμάστε το ίδιο και στο ζεύγος εικόνων imForest1 και imForest2 όπου αντίστοιχα έχουμε δύο αεροφωτογραφίες μιας δασικής έκτασης. Σχολιάστε τις δυσκολίες που προκύπτουν και δώστε μια ερμηνεία.

Σημειώστε ότι οι δύο αεροφωτογραφίες στην αστική περιοχή εχουν την ίδια χωρική ανάλυση (Ground Sampling Distance - GSD) και το ίδιο συμβαίνει και αυτές στις δασικές εικόνες. Σχολιάστε αν θα μπορούσε να λειτουργήσει η μεθοδολογία που αναπτύξατε εαν δεν ίσχυε αυτή η συνθήκη, για παράδειγμα αν οι δύο φωτογραφίες είχαν ληφθεί από διαφορετικό ύψος με τον ίδιο αισθητήρα κάμερας.

Για την υποβολή της εργασίας

Παραδώστε μία αναφορά με τις περιγραφές και τα συμπεράσματα που σας ζητούνται στην εκφώνηση. Η αναφορά θα πρέπει να επιδεικνύει την ορθή λειτουργία του κώδικά σας στις εικόνες που σας δίνονται.

Ο κώδικας θα πρέπει να είναι σχολιασμένος ώστε να είναι κατανοητό τι ακριβώς λειτουργία επιτελεί (σε θεωρητικό επίπεδο, όχι σε επίπεδο κλίσης συναρτήσεων). Επίσης, ο κώδικας θα πρέπει να εκτελείται και να υπολογίζει τα σωστά αποτελέσματα για *οποιαδήποτε* είσοδο πληροί τις υποθέσεις της εκφώνησης, και όχι μόνο για τις εικόνες που σας δίνονται.

Απαραίτητες προϋποθέσεις για την βαθμολόγηση της εργασίας σας είναι ο κώδικας να εκτελείται χωρίς σφάλμα, καθώς και να τηρούνται τα ακόλουθα:

- Υποβάλετε ένα και μόνο αρχείο, τύπου zip.
- Το όνομα του αρχείου πρέπει να είναι `AEM.zip`, όπου `AEM` είναι τα τέσσερα ψηφία του Α.Ε.Μ. του φοιτητή της ομάδας.
- Το προς υποβολή αρχείο πρέπει να περιέχει τα αρχεία κώδικα σε MATLAB ή Python και το αρχείο `report.pdf` το οποίο θα είναι η αναφορά της εργασίας.
- Εαν η εργασία υλοποίηθει σε Python θα πρέπει μαζί με τα αρχεία κώδικα να συμπεριλάβετε και το αρχείο environment.yml με τα πακέτα που έχετε χρησιμοποιήσει.
- Η αναφορά πρέπει να είναι ένα αρχείο τύπου PDF, και να έχει όνομα `report.pdf`.
- Το αρχείο τύπου zip που θα υποβάλετε δεν πρέπει να περιέχει κανέναν φάκελο.
- Μην υποβάλετε τις εικόνες που σας δίνονται για πειραματισμό.
- Μην υποβάλετε αρχεία που δεν χρειάζονται για την λειτουργία του κώδικά σας, ή φακέλους/αρχεία που δημιουργεί το λειτουργικό σας, πχ “Thumbs.db”, “.DS_Store”, “.directory”.
- Για την ονομασία των αρχείων που περιέχονται στο προς υποβολή αρχείο, χρησιμοποιείτε μόνο αγγλικούς χαρακτήρες, και όχι ελληνικούς ή άλλα σύμβολα, πχ “#”, “$”, “%” κλπ.

Αναφορές

[1] C. Harris and M. Stephens, *A Combined Corner and Edge Detector*, Proceedings of the 4th Alvey Vision Conference, August 1988, pp. 147-151