

Ψηφιακή Επεξεργασία Εικόνας -Εργασία 3-

Οπτική αναγνώριση χαρακτήρων

Προεπεξεργασία εικόνας για ανάγνωση χαρακτήρων
Εντοπισμός περιγράμματος
Αναπαράσταση και σύγκριση περιγραμμάτων
Αναγνώριση χαρακτήρων



Μάστορας Ραφαήλ Ευάγγελος

7918

11/6/2017

Περιγραφή της εργασίας

Στην εργασία αυτή μας δόθηκε μια εικόνα κειμένου, την οποία έπρεπε να επεξεργαστούμε κατάλληλα προκειμένου να αναγνωρίσουμε μέσω Matlab το κείμενο που περιέχει. Αρχικά χρειάστηκε να την περιστρέψουμε κατάλληλα. Έπειτα έπρεπε να δημιουργήσουμε περιγραφές για τα γράμματα που μας δόθηκαν, βρίσκοντας τα περιγράμματα τους. Τέλος έπρεπε να χωρίσουμε την εικόνα στα επιμέρους γράμματα που απαρτίζουν το κείμενο και να τα αναγνωρίσουμε.

Προεπεξεργασία εικόνας

Καθώς η εικόνα που μας δόθηκε είχε υποστεί περιστροφή, χρειάστηκε να την αναιρέσουμε. Για να το πετύχω αυτό δημιούργησα την συνάρτηση `fixrotation(x)`. Στην συνάρτηση αυτή χρησιμοποιώ αρχικά ένα Gaussian φίλτρο για να δημιουργήσω θόρυβο και έπειτα ένα ζωνοπερατό φίλτρο για να ενωθούν τα γράμματα μεταξύ τους. Μετά βρίσκω την τον μέγιστο συντελεστή Fourier της εικόνας, αφού πρώτα έχω αφαιρέσει την DC συνιστώσα. Γνωρίζοντας την θέση του μέγιστου συντελεστή Fourier βρίσκω την γωνία θ και την μετατρέπω σε rad. Έπειτα χρησιμοποιώ την γωνία αυτή για να γίνει μια πρώτη περιστροφή της εικόνας. Προκειμένου να βελτιωθεί το αποτέλεσμα ακόμα περισσότερο, αναζητώ σειριακά την βέλτιστη περιστροφή γύρω από την αρχική μου γωνία. Αυτό το κάνω χρησιμοποιώντας την κατακόρυφη προβολή της φωτεινότητας. Συγκεκριμένα ελέγχω σε ποια γωνία έχω περισσότερες λευκές γραμμές και αυτή θα είναι η βέλτιστη.

the battle of the coral sea, fought during 4-8 may 1942, was a major naval battle in the pacific theater of world war ii between the imperial japanese navy (ijn) and naval and air forces from the united states and australia. the battle was the first action in which aircraft carriers engaged each other, as well as the first in which neither side's ships sighted or fired directly upon the other.

in an attempt to strengthen their defensive positioning for their empire in the south pacific, japanese forces decided to invade and occupy port moresby in new guinea and tulagi in the southeastern solomon islands. the plan to accomplish this, called operation mo, involved several major units of japan's combined fleet, including two fleet carriers and a light carrier to provide air cover for the invasion fleets, under the overall command of japanese admiral shigeyoshi inoue. the us learned of the japanese plan through signals intelligence and sent two united states navy carrier task forces and a joint australian-american cruiser force, under the overall command of american admiral frank j. fletcher, to oppose the japanese offensive.

on 3-4 may, japanese forces successfully invaded and occupied tulagi, although several of their supporting warships were surprised and sunk or damaged by aircraft from the us fleet carrier yorktown. now aware of the presence of us carriers in the area, the japanese fleet carriers advanced towards the coral sea with the intention of finding and destroying the allied naval forces. beginning on 7 may, the carrier forces from the two sides exchanged in airstrikes over two consecutive days. the first day, the us sank the japanese light carrier shoho, while the japanese sank a us destroyer and heavily damaged a fleet oiler (which was later scuttled). the next day, the japanese fleet carrier shokaku was heavily damaged, the us fleet carrier lexington was critically damaged (and was scuttled as a result), and the yorktown was damaged. with both sides having suffered heavy losses in aircraft and carriers damaged or sunk, the two fleets disengaged and retired from the battle area. because of the loss of carrier air cover, inoue recalled the port moresby invasion fleet, intending to try again later.

Αρχικά η γωνία βρέθηκε 12.265 και βελτιώθηκε σε 12.015.

Διαχωρισμός γραμμών κειμένου

Στην συνέχεια δημιουργήσα την συνάρτηση `lines=getlines(y)` η οποία λαμβάνει την εικόνα και επιστρέφει έναν cell array που περιέχει τις επιμέρους γραμμές. Για να το πετύχω αυτό χρησιμοποίησα την κατακόρυφη προβολή της φωτεινότητας. Κάθε γραμμή κειμένου προσδιορίζεται βρίσκοντας σε ποια γραμμή της εικόνας η φωτεινότητα γίνεται μεγαλύτερη του μηδενός και έπειτα από μερικές γραμμές ξανά μηδενίζεται. Το διάστημα ανάμεσα στον επιμέρους μηδενισμό της φωτεινότητας προσδιορίζει και την ύπαρξη γραμμής κειμένου.

Έπειτα η συνάρτηση `letters=getletters(lines)` λαμβάνει τις επιμέρους εικόνες των γραμμών του κειμένου και χωρίζει τα γράμματα που βρίσκονται μέσα σε αυτές. Η διαδικασία είναι ίδια με πριν με την διαφορά ότι τώρα χρησιμοποιώ την οριζόντια προβολή φωτεινότητας για να διακρίνω που αρχίζει και που τελειώνει το κάθε γράμμα. Επίσης χρειάστηκε να χρησιμοποιήσω ζωνοπερατό φίλτρο και να διώξω τυχόν 'σκουπίδια' με τον μορφολογικό τελεστή `imopen`.

Οι παραπάνω συναρτήσεις καλούνται μέσω της συνάρτησης `readtext(y)` που υλοποιεί την συνολική διαδικασία αναγνώρισης των γραμμών ανά γραμμή.

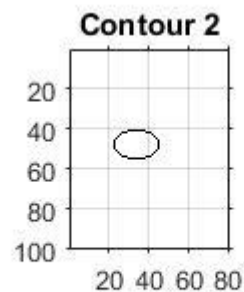
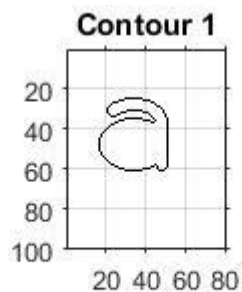
Εντοπισμός περιγραμμάτων

Για τον εντοπισμό των περιγραμμάτων δημιουργήσα την συνάρτηση `getcontour(x)` η οποία λαμβάνει μια εικόνα και εντοπίζει το περίγραμμα του γράμματος που περιέχεται σε αυτή. Ο πίνακας που επιστρέφει είναι οι συντεταγμένες των σημείων που δημιουργούν το περίγραμμα του γράμματος. Αν το γράμμα έχει δύο περιγράμματα τα διαχωρίζει σε διαφορετικό cell. Προκειμένου να βρεθεί το εκάστοτε περίγραμμα κατωφλιώνω αρχικά την αρνητική εικόνα, έπειτα χρησιμοποιώ τον μορφολογικό τελεστή `imdilate` για να της διαστείλω και αφαιρώντας την διασταλμένη εικόνα από την αρχική μένει ένα περίγραμμα. Το περίγραμμα αυτό το λεπταίνω με τον μορφολογικό τελεστή `bwmorph(xtemp,'thin','inf')` και στην συνέχεια αναζητώ σειριακά το περίγραμμα. Μόλις βρεθεί κάποιο σημείο με τιμή φωτεινότητας 1 αναζητώ ωρολογιακά τα σημεία γύρω από αυτό και η διαδικασία συνεχίζεται κάθε φορά στο επόμενο γειτονικό σημείο που εντοπίστηκε πρώτο. Τα σημεία αυτά αποθηκεύονται σε έναν πίνακα και παράλληλα διαγράφονται από την αρχική εικόνα. Έτσι αν υπάρχει δεύτερο περίγραμμα αυτό θα βρίσκεται στον αρχικό πίνακα πλέον μόνο του, καθώς το πρώτο περίγραμμα έχει διαγραφεί από αυτή. Η προηγούμενη διαδικασία επαναλαμβάνεται εφόσον υπάρχει δεύτερο περίγραμμα.

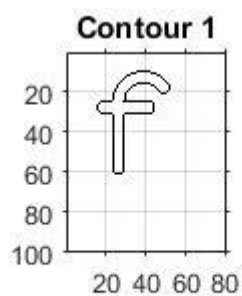
Τα περιγράμματα που ζητήθηκαν εμφανίζονται παρακάτω.

Περιγράμματα:

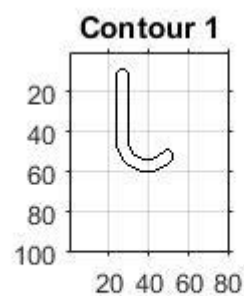
‘a’



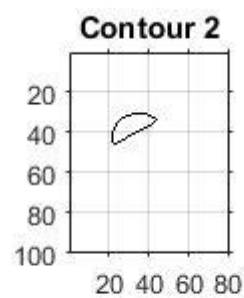
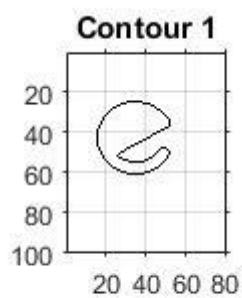
‘f’



‘l’



‘e’



Αναγνώριση κειμένου

Η συνάρτηση `readtext(y)` όπως προανέφερα καλεί τις συναρτήσεις που περιεγράφηκαν παραπάνω. Επίσης δημιουργεί τους περιγραφές των δοθέν γραμμάτων αλλά και των γραμμάτων που βρέθηκαν στην εικόνα. Για την περιγραφή των γραμμάτων χρησιμοποιήθηκε ο τύπος $r[i] = x[i] + jy[i]$. Αφού βρεθούν αυτές οι περιγραφές, για κάθε γράμμα της εικόνας

αναζητείται μέσω της συνάρτησης `getChar(r1Ref,r2Ref,r1,r2)` ο πιο κοντινός περιγραφέας αναφοράς. Καθώς όμως οι περιγραφές των περιγραμμάτων δεν έχουν ίδιο μέγεθος χρειάστηκε αρχικά να κάνω παρεμβολή με την συνάρτηση `interp1`. Έπειτα βρίσκοντας το τετραγωνικό σφάλμα ανάμεσα στην μετασχηματισμό Fourier του περιγραφέα αναφοράς και του περιγραφέα του γράμματος της εικόνας και παίρνοντας το μικρότερο τετραγωνικό σφάλμα, βρίσκω το γράμμα που αντιστοιχεί στον περιγραφέα αναφοράς μέσω της συνάρτησης `char(64+index)` , όπου `index` το νούμερο του βέλτιστου περιγραφέα.

Στο σημείο αυτό να αναφέρω ότι δυστυχώς δεν κατάφερα να αναγνωρίσω σωστά το κείμενο και για αυτό τον λόγο δεν παρατίθεται το αποτέλεσμα. Ωστόσο τα περιγράμματα των γραμμάτων αναφοράς εμφανίζονται σωστά (εκτός των γραμμάτων `k,p,w`) και ο διαχωρισμός των γραμμάτων της εικόνας είναι επιτυχής.