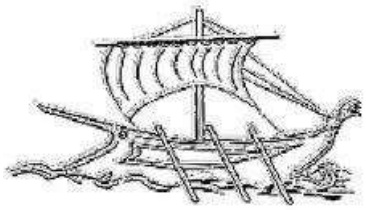
ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΠΕΙΡΑΙΑ



Τ.Ε.Ι ΠΕΙΡΑΙΑ

ΜΕΘΟΔΟΙ ΜΗΧΑΝΙΚΗΣ ΟΡΑΣΗΣ

ΤΣΙΑΜΠΑΣ ΓΕΩΡΓΙΟΣ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ :
ΝΙΚΟΛΑΟΥ ΓΡΗΓΟΡΙΟΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΑΥΤΟΜΑΤΙΣΜΟΥ

ΑΘΗΝΑ

Η πτυχιακή εργασία (Π.Ε.) αποτελεί προιόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα ,όσο και του ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο .

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή .Κάθε τέτοια πράξη αποτελεί προιόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα .Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της πτυχιακής εργασίας ,ο οποίος φέρει και την ευθύνη των συνεπειών ,ποινικών και άλλων αυτής της πράξης .

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα σε περίπτωση που το ίδρυμα του έχει απομείνει πτυχίο ,αυτό ανακαλείται με απόφαση της συνέλευσης του τμήματος .Η συνέλευση του τμήματος με νέα απόφαση της ,μετά από αίτηση του ενδιαφερόμενου ,του αναθέτει εκ νέου την εκπόνηση της Π.Ε. με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή .Η εκπόνηση της εν λόγω πτυχιακής εργασίας πρέπει να ολοκληρωθεί τουλάχιστον εντός ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσης της .Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18 ,παρ.5 του ισχύοντα εσωτερικού κανονισμού .

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία έχει σκοπό να δείξει διαφόρων ειδών επεξεργασίες πάνω στις εικόνες ή και τα frames ενός βίντεο και διάφορες μεθόδους αναλύσεων των εικόνων αυτών ,με σκοπό την αναγνώριση διαφόρων αντικειμένων ,χρωμάτων ,υπολογισμού διαφόρων πληροφοριών ,χαρακτηριστικών ,αφαίρεση και προσθήκη αντικειμένων κ.α. .

Μια εικόνα ισούται με χίλιες λέξεις .Ίσως δεν είναι τυχαία η ρήση αυτή .Λόγω των πληροφοριών που μας δίνει μια εικόνα αλλά και την ευρεία χρήση συσκευών και συστημάτων που σχετίζονται με εικόνες ,η επεξεργασία και η ανάλυση εικόνων έχει επεκταθεί και κρίνεται και όλο πιο απαραίτητη .Οι εφαρμογές που χρειάζεται να επεξεργαστούμε μια εικόνα είναι αμέτρητες .Τόσο στις βιομηχανίες που χρησιμοποιεί κάμερες για την αναγνώριση αντικειμένων ,όσο και για την όραση ενός ρομπότ με σκοπό να αποφεύγει διάφορα εμπόδια και διάφορες άλλες λειτουργίες ,η επεξεργασία και η ανάλυση των εικόνων είναι απαραίτητη .Η επεξεργασία και η ανάλυση μιας εικόνας χρησιμοποιείται και στον κινηματογράφο για προσθήκη εφέ ,βελτιστοποίηση ή και διόρθωση της εικόνας ,ή ακόμα και στην δημιουργία τρισδιάστατων εικόνων κ.α. .Επίσης από κάμερες με σκοπό την καλύτερη δυνατόν φωτογράφηση και βίντεο .Μάλιστα πολλές κάμερες προσπαθούν να ανιχνεύσουν συγκεκριμένα χαρακτηριστικά της εικόνας για να εστιάσουν σε αυτά ,όπως πρόσωπα .Επίσης η χρήση εφαρμογών που επεξεργάζονται και αναλύουν την εικόνα μπορεί να γίνει από οποιονδήποτε τώρα .

Παρακάτω περιγράφεται ένα παράδειγμα της συνάρτησης ShapesDetection στην οποία ανιχνεύονται διάφορα γεωμετρικά σχήματα από διάφορες εικόνες ,αλλά και τυχόν γραμμές τόσο μεμονωμένες αλλά τόσο και σε συστοιχία .

Επίσης στην συνάρτηση αυτή υπολογίζονται διάφορες πληροφορίες για τα γεωμετρικά σχήματα αυτά ,όπως περίμετρος και εμβαδόν αλλά και πόσο απέχουν τα σχήματα που εντοπίστηκαν μεταξύ τους .Ακόμα περιγράφεται και ο τρόπος εύρεσης του χρώματος που μπορεί να έχουν τα γεωμετρικά σχήματα αυτά .

Επίσης στην συνάρτηση αυτή περιγράφεται πως μπορεί να τμηματοποιηθεί μια εικόνα με βάση τα χρώματα ή και την έντασή της .

Επίσης πέρα από συνήθεις λειτουργίες πάνω στις εικόνες ,αναφέρονται και κάποιες μεθόδους ,που μπορεί να μην βγάζουν σωστά αποτελέσματα στη συγκεκριμένη περίπτωση ,αλλά χρησιμοποιούνται γενικά στην επεξεργασία και στην ανάλυση των εικόνων ,.Ο λόγος που περιγράφονται είναι για να διαβάσει ο αναγνώστης και άλλων ειδών επεξεργασίες και αναλύσεις πάνω στις εικόνες που είναι χρήσιμες για άλλων ειδών επεξεργασίες και αναλύσεις .

Η συνάρτηση ShapesDetection εκτελεί διάφορες επεξεργασίες και αναλύσεις πάνω σε μια εικόνα

.Για αυτό το λόγο απαιτείται στην παράμετρο της συνάρτησης να υπάρχει μια εικόνα .Για λόγους ταχύτητας και απλοποίησης όμως ,τις εικόνες τις παίρνουμε εντός της συνάρτησης .

ABSTRACT

This project is intended to show various kinds of processing on the images, or frames of a video, and various methods of analysis of these images, in order to identify various objects, colors, calculating various information, features, removing and adding items etc. . .

An image equals a thousand words. Perhaps this is not a random decision. Because of the information given to us by an image and the widespread use of image-related devices and systems, image processing and analysis has been extended and judged to be all the more necessary . The applications we need to process an image are countless. In industries that use cameras to recognize objects and to see a robot in order to avoid various obstacles and various other functions, Processing and analysis of images is necessary. The processing and analysis of an image is also used in the cinema for adding effects, optimizing or correcting the image, or even creating three-dimensional images, etc. . Also from cameras for the best possible shooting and video. In fact, many cameras try to detect specific image characteristics to focus on them, such as faces. Also, the use of image processing and analyzing applications can be done by anyone now.

Below is described an example of ShapesDetection function wherein various geometric shapes detected by the various figures, and any such individual or in array lines .

Also in this context calculated various information such geometric shapes as perimeter and area and how far the figures identified together .Also described and the method of color finder that can have the same geometries.

Also in this context it is described that can be segmented an image based on color or its intensity.

Also apart from routine operations on the images, described some methods, which may not earn the right results in this case, but are generally used in the processing and analysis of images .The reason is to read the reader and other species processes and analyzes the images that are useful for other types of processing and analysis.

The ShapesDetection function performs various edits and analyzes on an image. For this reason, an image is required in the function parameter. However, for reasons of speed and simplification, the images are taken in the function.

ΣΚΟΠΟΣ ΤΗΣ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Σκοπός της παρούσας πτυχιακής εργασίας είναι να δείξει διάφορες πληροφορίες και μεθόδους που αφορούν την επεξεργασία και την ανάλυση των εικόνων ή και τα frames ενός βίντεο .Οι πληροφορίες και οι μέθοδοι που παρουσιάζει η παρούσα πτυχιακή εργασία δεν αφορούν μόνο στις συγκεκριμένες επεξεργασίες και αναλύσεις των εικόνων ,αλλά και στον προγραμματισμό γενικότερα ,ώστε να μπορεί να καλύψει και άλλων ειδών επεξεργασίες και αναλύσεις των εικόνων .Για αυτό το λόγο αναφέρονται και κάποιες μεθόδους που μπορεί να μην λειτουργούν στο

συγκεκριμένο παράδειγμα ,αλλά είναι χρήσιμες σε άλλες περιπτώσεις .

Γενικές επεξεργασίες των εικόνων δείχνονται στο παρακάτω παράδειγμα .Στο παρακάτω παράδειγμα δείχνεται η μετατροπή μιας έγχρωμης εικόνας σε άχρωμη ,με βάση τα χρώματα που αυτή περιέχει .Έπειτα δείχνεται πως παίρνουμε μια ασπρόμαυρη εικόνα από μια άχρωμη εικόνα ,με βάση την ένταση που έχει κάθε pixel της άχρωμης εικόνας .

Περιγράφεται πως γίνεται να εξισορροπήσουμε την ένταση στην εικόνα ,χρησιμοποιώντας την μέθοδο Otsu .Αυτή η περίπτωση είναι πολλή χρήσιμη στις περιπτώσεις που υπάρχουν παράμετροι που αλλοιώνουν την εικόνα ,όπως σκιές ,χαμηλή αντίθεση κ.α. .Αυτό έχει σαν συνέπεια να αλλοιώνονται και τα αποτελέσματα της επεξεργασίας .Η μέθοδος Otsu ανήκει στην κατηγορία μεθόδων τμηματοποίησης της εικόνας .Άλλη μια μέθοδο που περιέχεται στην παρούσα πτυχιακή εργασία και ανήκει και σε αυτήν την κατηγορία είναι η μέθοδος ομαδοποίησης K-means .Αυτή η μέθοδο τμηματοποιεί τα χρώματα της εικόνας .

Πολλές φορές στην εικόνα υπάρχουν κάποιες κηλίδες ή σκιές στην εικόνα που χρειάζεται να αφαιρεθούν .Αυτές μπορούν να αφαιρεθούν με τις μορφολογικές λειτουργίες της διάβρωσης ή και της διαστολής .οι οποίες αναλύονται παρακάτω .

Επίσης περιέχονται μέθοδοι ανίχνευσης των αντικειμένων που περιέχονται στην εικόνα .Συγκεκριμένα φαίνεται πως ανιχνεύονται τα αντικείμενα της εικόνας ψάχνοντας από 'γειτονιά' σε 'γειτονιά' pixel με ασυνεχή και συνεχή τιμή .Υπάρχουν πολλές και διάφοροι μέθοδοι ,αναλόγως τι θέλουμε να εντοπίσουμε .

Περιγράφεται πως γίνεται να πάρουμε διάφορες πληροφορίες για τα αντικείμενα που ανιχνεύτηκαν και πως μπορούν να χρησιμοποιηθούν εύκολα και απλά .

Επίσης περιγράφεται ένας τρόπος ανίχνευσης γραμμών ,καθώς και διάφορες πληροφορίες για αυτές ,όχι απαραίτητα μεμονωμένων ,χρησιμοποιώντας hough transform .

Τέλος περιγράφονται διάφοροι τρόποι εμφάνισης της εικόνας και των αποτελεσμάτων .

Με τις μεθόδους που περιγράφονται ο αναγνώστης μπορεί να μάθει διάφορους τρόπους ανάλυσης και επεξεργασίας των εικόνων ,για διάφορες λειτουργίες και όχι απαραίτητα ανίχνευσης γεωμετρικών σχημάτων και γραμμών όπως περιγράφονται στο παράδειγμα .

ΚΕΦΑΛΑΙΟ 1 : Μορφολογική επεξεργασία της εικόνας

Εισαγωγή κεφαλαίου

Στο κεφάλαιο αυτό θα αναφερθούν μερικές λειτουργίες που σε πολλές περιπτώσεις πρέπει να γίνουν πριν τη λειτουργία της ανίχνευσης των σχημάτων που υπάρχουν στην εικόνα .

Συγκεκριμένα στο κεφάλαιο αυτό εξηγείται η περίπτωση που εικόνα δεν έχει σταθερή φωτεινότητα σε όλα τα σημεία της .Η φωτεινότητα σε όλα τα σημεία του σχήματος μπορεί να μην είναι ίδια και συγκεκριμένα μπορεί η φωτεινότητα σε κάποια σημεία του σχήματος της εικόνας να είναι σχεδόν ίδια με το φόντο της εικόνας .Αυτό όμως μπορεί να έχει σαν αποτέλεσμα ένα μέρος τους σχήματος να θεωρηθεί ότι αποτελεί μέρος του φόντου της εικόνας .Η συνέπεια του αποτελέσματος αυτού θα είναι κατά την μετατροπή της εικόνας σε δυαδική εικόνα (ασπρόμαυρη) ,ένα μέρος του σχήματος να έχει άλλο χρώμα από το υπόλοιπο και συγκεκριμένα ίδιο χρώμα με το φόντο της εικόνας .Για αυτό ,σε αυτές τις περιπτώσεις πρέπει να γίνει εξισορρόπηση της φωτεινότητας στην εικόνα ,πριν την ανίχνευση του σχήματος της εικόνας .

Μια άλλη περίπτωση που μπορεί να εμποδίζει την αναγνώριση του σχήματος της εικόνας ,είναι τυχόν σταδιακές εναλλαγές του χρώματος ή και της φωτεινότητας που υπάρχουν στην αρχική εικόνα σε διάφορα σημεία της ,κυρίως στην περιφέρεια του σχήματος της εικόνας ,όπως σκιές κ.α. .Αυτές οι προεξοχές είναι απαραίτητο να αφαιρεθούν ,γιατί αλλοιώνουν το πραγματικό σχήμα που δείχνει η δυαδική εικόνα .Δηλαδή στην δυαδική εικόνα απεικονίζεται άλλο σχήμα από αυτό της αρχικής εικόνας .Το να εξισορροπηθεί όμως η φωτεινότητα ,δεν σημαίνει ότι θα εξαλειφθούν αυτές οι προεξοχές .Οι προεξοχές αυτές αποτελούν μέρος της εικόνας και δε γίνεται να αφαιρεθούν με αυτόν τον τρόπο .Για την αφαίρεση των προεξοχών αυτών γίνεται διαφορετική επεξεργασία πάνω στην εικόνα που εξηγείται παρακάτω .

Επίσης σε αυτό το κεφάλαιο εξηγείται ο τρόπος που μπορεί να αυξηθεί η αντίθεση της εικόνας .Σε περιπτώσεις που η αντίθεση της εικόνας είναι χαμηλή ,αυτό μπορεί να έχει σαν αποτέλεσμα τα αντικείμενα του παρασκηνίου ,όπως γεωμετρικά σχήματα , να μην 'ξεχωρίζουν' από το φόντο της εικόνας .Αυτό θα έχει σαν αποτέλεσμα να μην εντοπιστεί το γεωμετρικό σχήμα .Η υπερβολική αύξηση της αντίθεσης από την άλλη μπορεί να έχει και αρνητικά αποτελέσματα .

Επίσης επεξηγούνται κάποιες επιπλέον μορφολογικές λειτουργίες ,χρησιμοποιήσιμες μεν ,αλλά όχι στην συγκεκριμένη περίπτωση .

Οι παραπάνω επεξεργασίες της εικόνας ,μπορεί να δημιουργήσουν κάποιες μικρές 'κουκίδες' στην δυαδική εικόνα που προκύπτει στο τέλος .Για καλύτερα οπτικά αποτελέσματα αλλά και καλύτερη ανάλυση της εικόνας ,είναι προτιμότερο η δυαδική εικόνα να 'καθαριστεί' από τις μικρές αυτές 'κουκίδες' .

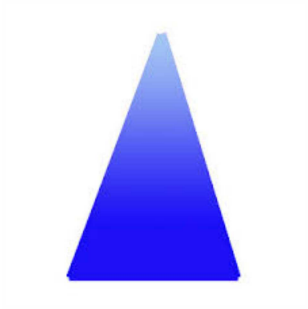
Τα βήματα που κάνουμε σε αυτό το μέρος του προγράμματος είναι τα εξής :

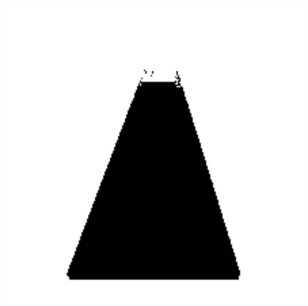
- 1. Παίρνουμε το φόντο της εικόνας ο οποίο αυτό θα περιέχει την πληροφορία που θέλουμε ,δηλαδή την φωτεινότητα της εικόνας στα διάφορα σημεία της και τίποτα άλλο (όπως σχήματα) .
  Άρα πριν ξεκινήσει η διαδικασία να εντοπιστούν κάποια από τα γεωμετρικά σχήματα που

τυχόν μπορεί να υπάρχουν στην εικόνα ,όπως τρίγωνα ,τετράγωνα ,κύκλους και γραμμές ,αναλύεται η περίπτωση που η εικόνα δεν έχει σταθερή φωτεινότητα σε όλα τα σημεία της εικόνας .Διάφορες λεπτομέρειες όπως αλλαγή φωτεινότητας κ.α. ,έχουν ως αποτέλεσμα την αλλοίωση των αποτελεσμάτων .Συγκεκριμένα κατά την μετατροπή της εικόνας σε δυαδική εικόνα (ασπρόμαυρη) ,που γίνεται παρακάτω ,η τιμή χρώματος κάθε pixel της εικόνας συγκρίνεται με ένα όριο .Όταν η φωτεινότητα σε όλα τα σημεία του σχήματος δεν είναι ίδια ,μπορεί αυτό να έχει σαν αποτέλεσμα ένα μέρος τους σχήματος να θεωρηθεί ότι αποτελεί μέρος του φόντου της εικόνας .Η συνέπεια του αποτελέσματος αυτού θα είναι κατά την μετατροπή της εικόνας σε δυαδική εικόνα (ασπρόμαυρη) ,ένα μέρος του σχήματος να έχει άλλο χρώμα από το υπόλοιπο και συγκεκριμένα ίδιο χρώμα με το φόντο της εικόνας .Αυτό έχει σαν αποτέλεσμα στην δυαδική εικόνα που δημιουργείται να μην υπάρχει ολόκληρο το σχήμα που υπάρχει στην έγχρωμη εικόνα .Για αυτό ,σε αυτές τις περιπτώσεις πρέπει να γίνει εξισορρόπηση της φωτεινότητας στην εικόνα ,πριν την ανίχνευση του σχήματος της εικόνας .

- 2. Εξισορροπείται η φωτεινότητα στην εικόνα .
- 3. Αφαιρούνται τυχόν προεξοχές που υπάρχουν στην περιφέρεια του σχήματος της δυαδικής εικόνας (ασπρόμαυρης) .
  Αυτές οι προεξοχές μπορεί να οφείλονται σε τυχόν σταδιακές εναλλαγές του χρώματος ή και της φωτεινότητας που υπάρχουν στην αρχική εικόνα σε διάφορα σημεία της ,κυρίως στην περιφέρεια του σχήματος της εικόνας ,όπως σκιές κ.α. .
  Αυτές οι προεξοχές αλλοιώνουν το πραγματικό σχήμα που δείχνει η δυαδική εικόνα και για αυτό είναι απαραίτητο να αφαιρεθούν .Δηλαδή στην δυαδική εικόνα απεικονίζεται άλλο σχήμα από αυτό της αρχικής εικόνας .
- 4. Επεξηγούνται κάποιες άλλες μέθοδοι μορφολογικής επεξεργασίας της εικόνας .
- 5. Αυξάνεται η αντίθεση της εικόνας .
  Σε περιπτώσεις που η αντίθεση της εικόνας είναι χαμηλή ,αυτό μπορεί να έχει σαν αποτέλεσμα τα αντικείμενα του παρασκηνίου ,όπως γεωμετρικά σχήματα , να μην 'ξεχωρίζουν' από το φόντο της εικόνας .Αυτό θα έχει σαν αποτέλεσμα να μην εντοπιστεί το γεωμετρικό σχήμα της εικόνας .Για αυτό σε αυτές τις περιπτώσεις πρέπει να αυξηθεί η αντίθεση της εικόνας .Η υπερβολική αύξηση της αντίθεσης από την άλλη μπορεί να έχει και αρνητικά αποτελέσματα .
- 6. Για καλύτερα οπτικά αποτελέσματα και καλύτερη ανάλυση της εικόνας ,αφαιρούνται τυχόν μικρές 'κουκίδες' που δημιουργήθηκαν στην δυαδική εικόνα από τις παραπάνω επεξεργασίες .

Για παράδειγμα η παρακάτω εικόνα δείχνει ένα τρίγωνο που το χρώμα του ανοίγει προς τα πάνω .Βλέπουμε ότι στο τέλος ,όταν παίρνουμε τη δυαδική εικόνα ,δεν παίρνουμε το τρίγωνο της έγχρωμης εικόνας .





Κώδικας

```
%%
```

Η συνάρτηση ShapesDetection ,μπορεί να χρησιμοποιηθεί από οποιοδήποτε πρόγραμμα ,απλά καλώντας την χρησιμοποιώντας ως όρισμα μια εικόνα (ShapesDetection(I);) .Για λόγους ταχύτητας και απλοποίησης όμως ,τις εικόνες τις παίρνουμε εντός της συνάρτησης .

```
function ShapesDetection(ImageRead)
```

Παρακάτω παίρνουμε τις εικόνες που θα επεξεργαστούμε .

```
rootFolder = fullfile(toolboxdir('vision'),'visiondata', 'ShapesDetection');
shapesScene = imageSet(fullfile(rootFolder));
ImageRead = read(shapesScene,1);
```

Παρακάτω θα εμφανίσουμε την εικόνα στην οποία θα ανιχνευτούν τυχόν γεωμετρικά σχήματα .

```
figure;
imshow(ImageRead);
title('Original image');
```

%%

Για αρχή ,για να πάρουμε την ίδια εικόνα με σταθερή φωτεινότητα σε όλα τα σημεία ,θα μειώσουμε από την αρχική εικόνα τη φωτεινότητα από κάθε σημείο της εικόνας αναλόγως με την τιμή της φωτεινότητας του ,ώστε να έχουμε την ίδια φωτεινότητα σε όλα τα σημεία της εικόνας .Δηλαδή πρέπει να ρίξουμε πιο πολύ τη φωτεινότητα ενός σημείου της εικόνας που είναι πιο φωτεινό από ένα άλλο σημείο της εικόνας .Η λογική λέει να αφαιρεθούν δύο ίδιες εικόνες ώστε τα σημεία της εικόνας με μεγάλη φωτεινότητα ,να αφαιρεθούν από σημεία με μεγάλη φωτεινότητα και αντίστοιχα για τα σημεία με χαμηλή φωτεινότητα .Δε θα πάμε όμως να αφαιρέσουμε κατευθείαν την αρχική εικόνα με τον εαυτό της ,γιατί όπως είναι λογικό θα πάρουμε μια μαύρη εικόνα (τιμή 0) .Θα πάρουμε πρώτα το φόντο της εικόνας το οποίο αυτό θα περιέχει την πληροφορία που θέλουμε ,δηλαδή την φωτεινότητα της εικόνας στα διάφορα σημεία της και τίποτα άλλο ,όπως σχήματα .Αφαιρώντας έπειτα το φόντο από την αρχική εικόνα θα επιτευχθεί μια σταθερή φωτεινότητα σε όλη την εικόνα ,χωρίς να χαθεί το σχήμα της εικόνας και αυτό γιατί στο φόντο δεν υπάρχει το σχήμα .Άρα με απλά λόγια δεν αφαιρείται σχήμα με σχήμα και άρα αυτό δε χάνεται μετά την αφαίρεση των εικόνων .

Το φόντο μπορούμε να το πάρουμε με τη συνάρτηση imopen .Η συνάρτηση imopen εκτελεί την μορφολογικές επεξεργασίες της διάβρωσης και έπειτα της διαστολής σε μια άχρωμη εικόνα .Με τη μορφολογική επεξεργασία της διάβρωσης μπορούμε να 'θαμπώσουμε' τις έντονες και μόνο τις έντονες περιοχές της εικόνας .Βασικά με τη διάβρωση χάνονται οι λεπτομέρειες της έντονης αυτής περιοχής της εικόνας και αυτή ενσωματώνεται πιο πολύ στο περίγυρο της περιοχής ,δηλαδή το φόντο .Με τη μορφολογική επεξεργασία της διαστολής ,γίνεται διαστολή των έντονων περιοχών της εικόνας (με άσπρο χρώμα) κατακόρυφα .

Εν κατακλείδι ,με τις παραπάνω μορφολογικές επεξεργασίες πετυχαίνουμε να εξαφανίσουμε τα όρια μιας έντονης περιοχής στην εικόνας ,όπως ένα σχήμα και να το ενσωματώσουμε αυτό στο φόντο της εικόνας .Άρα αποτέλεσμα της επεξεργασίας αυτής είναι να πάρουμε το φόντο μιας εικόνας .Συγκεκριμένα χρειαζόμαστε τη φωτεινότητα της εικόνας στα διάφορα σημεία της και τίποτα άλλο ,όπως σχήματα .Αφαιρώντας έπειτα το φόντο από την αρχική εικόνα θα επιτευχθεί μια σταθερή φωτεινότητα σε όλη την αρχική εικόνα ,και χωρίς να χαθεί το σχήμα της αρχικής εικόνας .

Η 'εξαφάνιση' των ορίων μιας έντονης περιοχής και της ενσωμάτωσή της στο φόντο πετυχαίνεται με τις συναρτήσεις imopen και imclose ,που εκτελούν διάφορες μορφολογικές επεξεργασίες .

Η συνάρτηση imopen επεξεργάζεται μορφολογικά τις περιοχές της εικόνας με τον τρόπο που υποδεικνύει το επίπεδο στοιχείων που έχουμε δημιουργήσει με τη συνάρτηση strel .Η συνάρτηση strel δημιουργεί ένα επίπεδο δυαδικών τιμών (όπως ένας πίνακας 2 διαστάσεων) ,όπου οι τιμές καταχωρούνται αναλόγως του είδους της επεξεργασίας που θέλουμε να κάνουμε .Στο παρακάτω παράδειγμα δημιουργείται ένα 'δισκοειδές' επίπεδο .Δηλαδή τα στοιχεία του επιπέδου που βρίσκονται εντός ενός δίσκου με ακτίνα μέχρι 80 στοιχείων και κέντρο το κεντρικό στοιχείο του επιπέδου ,έχουν τιμή 1 .Τα υπόλοιπα έχουν τιμή 0 .Τα στοιχεία με τιμή 1 αντιπροσωπεύουν τα στοιχεία ενός πίνακα (όπως τα pixels μιας εικόνας) που θα υποστούν επεξεργασία από άλλες συναρτήσεις όπως οι imclose και imopen .

```
GRAY = rgb2gray(ImageRead);

figure,
imshow(GRAY),title('Gray Image');
```

```
background = imopen(GRAY,strel('disk',80));

figure;
imshow(background);title('imopen image')
```

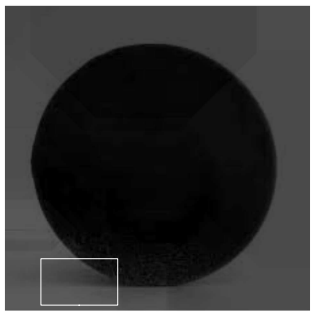
Εάν εκτελέσουμε τη συνάρτηση imclose σε αυτό το σημείο του προγράμματος ,θα πάρουμε μια πιο ομαλή εικόνα από αυτή που δίνουμε στην είσοδο .Επειδή όμως στην είσοδο δίνουμε μόνο το φόντο της εικόνας ,δηλαδή χωρίς καμία λεπτομέρεια ,τότε στην έξοδο θα πάρουμε πάλι μια εικόνα χωρίς λεπτομέρειες και άρα δεν υπάρχει λόγος να εκτελέσουμε τη συνάρτηση imclose .

```
% closeimage = imclose(background,strel('disk',80));
%
```

```
% figure;
% imshow(closeimage);
% title('imclose image')
```

Ύστερα όπως είπαμε ,θα ρίξουμε τη φωτεινότητα της εικόνας στα διάφορα σημεία της εικόνας ,αναλόγως τη φωτεινότητας τους .Αυτό θα το πετύχουμε αφαιρώντας από την αρχική εικόνα το φόντο που υπολογίσαμε (έξοδος συνάρτησης imopen) .Όλη αυτήν την διαδικασία ,δηλαδή την εύρεση του φόντου και την αφαίρεσή του από την αρχική εικόνα την κάνει η συνάρτηση imtophat .Παραπάνω περιέχεται βέβαια η συνάρτηση imopen ,αλλά μόνο για να δούμε το αποτέλεσμα της συνάρτησης αυτής .

```
GRAY2 =  imtophat(GRAY,strel('disk',80));

figure,
imshow(GRAY2),title('steady bright image');
```



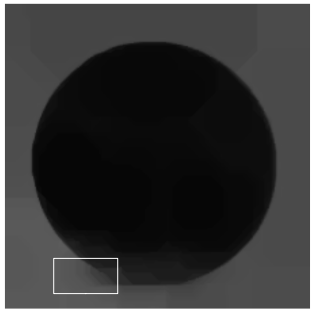
Βλέπουμε στην παραπάνω εικόνα ότι ένα μεγάλο μέρος της σκιάς ,που υπάρχει στην περιφέρεια του σχήματος (εντός ορθογωνίου) από την αρχική εικόνα έχει αφαιρεθεί .Όχι όμως όλη .Παρατηρούμε ότι υπάρχει ακόμα μια σκιά που εξέχει από την περιφέρεια του σχήματος .Αυτές οι προεξοχές μπορεί να οφείλονται σε τυχόν σταδιακές εναλλαγές του χρώματος ή και της φωτεινότητας που υπάρχουν στην αρχική εικόνα σε διάφορα σημεία της .

Η σκιά αυτή στο τέλος (κατά την μετατροπή της εικόνας σε δυαδική) θεωρείται κομμάτι του σχήματος ,με αποτέλεσμα να αλλοιώνεται το πραγματικό σχήμα που δείχνει η εικόνα .Για αυτό είναι απαραίτητο να αφαιρεθεί η σκιά αυτή .Αυτή τη σκιά που εξέχει από τον κύκλο ,μπορεί να αφαιρεθεί ενσωματώνοντάς την στο φόντο της εικόνας που περιβάλλει τη σκιά αυτή .Πιο συγκεκριμένα ,η σκιά μπορεί να καλυφθεί από το φόντο που υπάρχει γύρω από την σκιά αυτή .

Γενικά ,περιοχές και μικρά σημεία μπορούμε να τα αφαιρέσουμε με τη τη συνάρτηση imclose .Η συνάρτηση imclose ,όπως και η συνάρτηση imopen ,εκτελεί τις μορφολογικές επεξεργασίες ,πρώτα της διαστολής και έπειτα της διάβρωσης .Δηλαδή τα έντονα pixels (στη συγκεκριμένη περίπτωση τα pixels του φόντου είναι πιο έντονα από αυτά του κύκλου ,αλλά και της σκιάς) που θα διαστείλουμε και έπειτα θα διαβρώσουμε θα καλύψουνε τη σκιά που θέλουμε να εξαφανίσουμε .Βλέπουμε ότι λόγω της ακτίνας που έχει το επιπέδου δομικών στοιχείων που έχουμε επιλέξει , το σχήμα δεν επηρεάζεται ή τουλάχιστον δεν φαίνεται να επηρεάζεται .Όσο μεγαλώει η ακτίνα τόσο αυτό θα επηρεάζεται .

```
GRAY2 = imclose(GRAY2,strel('disk',17));

figure;
```

```
imshow(GRAY2);title('imclose image')
```



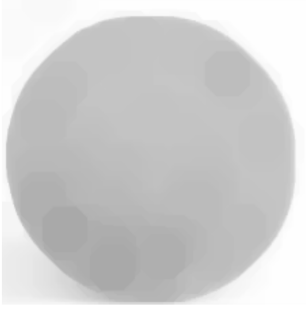
Στην παραπάνω εικόνα φαίνεται αισθητά η αφαίρεση της σκιάς .

Αντί για τις συναρτήσεις imopen και imclose μπορούμε να χρησημοποιήσουμε μια άλλη μέθοδο για την αφαίρεση των σκιών ή και γενικά διάφορων κουκίδων από την εικόνα .Στην άλλη μέθοδο ,η μορφολογική επεξεργασία της διαστολής (imdilate) δεν εφαρμόζεται απευθείας στην εικόνα που έχει υποστεί τη μορφολογική επεξεργασία της διάβρωσης (imerode) αλλά πρώτα δημιουργείται μια εικόνα ,παρόμοια με την αρχική εικόνα αλλά με τονισμένες τις περιοχές που καθορίζει η εικόνα που έχει υποστεί διάβρωση (imreconstruct) .Έπειτα σε αυτήν την εικόνα εφαρμόζεται η μορφολογική επεξεργασία της διαστολής .Η διαφορα της μεθόδου αυτής είναι η εικόνα που δημιουργούμε μοιάζει πιο πολύ στην αρχική εικόνα ,δηλαδή έχει περισσότερες λεπτόμερειες που μοιάζουν στην αρχική εικόνα .Βέβαια κάποιες από τις λεπτομέρειες αυτές μπορεί να είναι τυχόν σκιές που στη συγκεκριμένη περίπτωση θέλουμε να αφαιρέσουμε .

Σε αυτήν την μέθοδο δεν έχουμε αναφέρει τίποτα για εξισορρόπιση της φωτεινότητας στην εικονα γιατί δεν κάνουμε κάτι τέτοιο με αυτή τη μέθοδο .

Παρακάτω θα δοκιμάσουμε τον δεύτερο τρόπο και θα δούμε τα αποτελέσματα .

```
GRAY3 = imerode(GRAY, strel('disk',17));

figure;
imshow(GRAY3);title('imerode image')
```

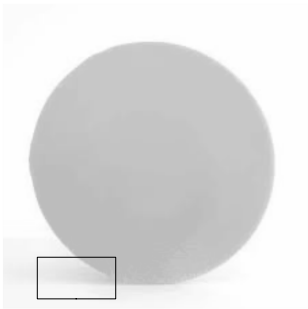


```
GRAY3 = imreconstruct(GRAY3, GRAY);
```

```
figure
imshow(GRAY3), title('Opening-by-reconstruction (GRAY3)')

GRAY3imdilateimage = imdilate(GRAY3, strel('disk',17));
```

Να γίνει παρατήρηση ότι πρέπει πρώτα και μετά ,να υπολογίσουμε το συμπλήρωμα της εικόνας (`imcomplement`) .Το συμπλήρωμα μιας άχρωμης εικόνας προκύπτει από την αφαίρεση των τιμών των δεδομένων της εικόνας (τα pixels) από τη μέγιστη τιμή που μπορούν να πάρουν τα δεδομένα αυτά .Το αποτέλεσμα της αφαίρεσης είναι η νέα τιμή του pixel της εικόνας .Αυτό έχει σαν αποτέλεσμα οι σκούρες περιοχές της εικόνας να γίνονται πιο φωτεινές και το αντίθετο .

```
GRAY3 = imreconstruct(imcomplement(GRAY3imdilateimage), imcomplement(GRAY3));
GRAY3 = imcomplement(GRAY3);

figure
imshow(GRAY3), title('Opening-closing by reconstruction (GRAY3)')
```



Όπως βλέπουμε ,συγκρίνοντας την παραπάνω εικόνα με την πάνω εικόνα στη σελίδα 7 ,ο δεύτερος τρόπος δεν καταφέρνει να αφαιρέσει τη σκιά που δημιουργείται στην περιφέρεια του σχήματος κάτω αριστερά.

Έπειτα θα αυξήσουμε την αντίθεση της εικόνας GRAY2 με τη συνάρτηση imadjust ώστε να γίνει πιο ευκρινές το σχήμα της εικόνας και άρα να αυξηθούν οι πιθανότητες να απομονωθεί το σχήμα της εικόνας από το φόντο της .Η συνάρτηση imadjust μας δίνει την επιλογή να ορίσουμε εμείς την αντίθεση της εικόνας που θα πάρουμε .Παρακάτω όμως αφήνουμε τη συνάρτηση να αυξήσει την

αντίθεση της εικονας με τον προεπιλεγμένο τρόπο .Με αυτό τον τρόπο παίρνουμε μια εικόνα με μεγαλύτερη αντίθεση .

```
GRAY2 = imadjust(GRAY2);

figure;
imshow(GRAY2);title('increasing bright image')
```

Παρακάτω θα μετατρέψουμε την άχρωμη εικόνα (GRAY2) σε δυαδική εικόνα (άσπρο-μαύρο μόνο) με τη συνάρτηση im2bw .Η συνάρτηση im2bw συγκρίνει την φωτεινότητα του κάθε pixel της εικόνας με ένα όριο (threshold) και εάν αυτή είναι μεγαλύτερη τότε το pixel παίρνει τιμή 1 (άσπρο) ,αλλιώς παίρνει τιμή 0 (μαύρο) .Για την κατάλληλη επιλογή του ορίου (threshold) ,ώστε να μην έχουμε μεγάλη διαφορά άσπρου και μαύρου στην εικόνα , χρησιμοποιούμε τη συνάρτηση graythresh .

```
threshold = graythresh(GRAY2);
BW = im2bw(GRAY2, threshold);

figure;
imshow(BW);title('BW image')
```

Οι παραπάνω επεξεργασίες μπορεί να δημιουργήσουν κάποιες μικρές κουκίδες στην εικόνα .Οι κουκίδες αυτές ,πέρα από το οπτικό αποτέλεσμα ,μπορεί να οδηγήσουν σε λάθος ανάλυση της εικόνας .Το πρόβλημα δεν δημιουργείται στις 'χονδρικές' αναλύσεις της εικόνας ,όπως στην αναζήτηση του γεωμετρικού σχήματος της εικονας ,αφού φαίνεται ξεκάθαρα το γεωμετρικό σχήμα της εικόνας ,αλλά στις 'λεπτές' αναλύσεις της εικόνας ,όπως η αναζήτηση pixels με βάση του χρώματος (άσπρο - μαύρο) .

Η αφαίρεση των μικρών κουκίδων πετυχαίνεται με τη συνάρτηση bwareaopen ,όπου θα αφαιρέσουμε τα αντικείμενα της εικόνας που αποτελούνται από λιγότερο από 50 pixels .

```
BW = bwareaopen(BW, 50);

figure;
imshow(BW);title('clear image')
```

Πρώτων να πουμε ότι ο κύκλος που παίρνουμε στο τέλος δεν είναι ακριβώς κύκλος αλλά τείνει αρκετά .Τηρώντας αυστηρές προυποθέσεις για έναν κύκλο (επεξηγούνται παρακατω) το πρόγραμμα δε θα το ανιχνεύσει αυτόν τον κύκλο .Με την επεξεργασία που κάναμε όμως ,μπορούμε με μικρή χαλάρωση των προυποθέσεων ,χωρίς να έχουμε αλλοίωση και των υπόλοιπων αποτελεσμάτων ,να ανιχνεύσουμε τον παραπάνω κύκλο .

Δεύτερον να πούμε δεν μπορούμε να εφαρμόσουμε τις ίδιες μορφολογικές επεξεργασίες σε όλα τα γεωμετρικά σχημάτα ,γιατί σε κάποια θα έχουμε ως αποτέλεσμα την αλλοίωσή τους .Αλλαγές στην φωτεινότητα ή στη αντίθεση που έχει ένα σχήμα της εικόνας με το φόντο ,μπορεί να έχει ως αποτέλεσμα το σχήμα να μην διαφέρει αρκετά ,με αποτέλεσμα το σχήμα της εικόνας να θεωρείται ως ένα κομάτι του φόντου .Για αυτό το λόγο οι παραπάνω μέθοδοι θα εφαρμόζονται μόνο στις εικόνες που απαιτείται .

ΚΕΦΑΛΑΙΟ 2 : Padding

Εισαγωγή κεφαλαίου

Στο κεφάλαιο 2 εξηγούνται η ενέργειες που πρέπει να γίνουν στις περιπτώσεις που οι πλευρές του σχήματος που περιέχει η εικόνα τέμνουν στις πλευρές της εικόνας .

Στις περιπτώσεις που οι πλευρές του σχήματος της εικόνας τέμνουν στις πλευρές της εικόνας ,το σχήμα της εικόνας δε θα εντοπιστεί ,αφού δεν υπάρχει εναλλαγή χρώματος ώστε να γίνει διακριτή η περιφέρεια του σχήματος από το φόντο της εικόνας .Σε αυτήν την περίπτωση πρέπει να μεγαλώσουμε λίγο την εικόνα ,'προσθέτοντας' λίγο φόντο ,δηλαδή γραμμές και στήλες από pixels στις πλευρές της εικόνας ,ώστε να γίνει διακριτό το σχήμα από το φόντο της εικόνας .

Επίσης πρέπει τις γραμμές και τις στήλες από pixels που θα προσθέσουμε (συνάρτηση padarray) να έχουν χρώμα αντίθετο από αυτό της περιφέρειας του σχήματος .Π.χ. εάν είναι άσπρη η περιφέρεια του σχήματος της εικόνας ,πρέπει η εικόνα εξωτερικά να χρωματιστεί μαύρη και το αντίθετο .

Η εικόνα που έχουμε παρακάτω είναι δυαδική (ασπρόμαυρη) .Το άσπρο χρώμα ,σε μια δυαδική εικόνα ,αντιστοιχεί στην τιμή 1 και το μαύρο χρώμα αντιστοιχεί στην τιμή 0 .

Για την εύρεση του χρώματος της περιφέρειας του σχήματος της δυαδικής εικονας πρέπει να λάβουμε όλες τις περιπτώσεις που το σχήμα αυτό μπορεί να καλύπτει την εικόνα στις πλευρές της .Δηλαδή :

Στην περίπτωση ενός τριγώνου ,στη μία περίπτωση αυτό μπορεί να πιάνει την μία πλευρά της εικόνας ,ολόκληρη ή όχι .Στη δεύτερη περίπτωση αυτό μπορεί να πιάνει 2 πλευρές της εικόνας ,ολόκληρες ή όχι .

Στην περίπτωση ενός κύκλου ,το πολύ-πολύ ο κύκλος να ακουμπάει σε 4 σημεία στις πλευρές της εικόνας .

Στην περίπτωση μιας γραμμής ,το πολύ-πολύ η γραμμή να ταυτίζεται με τη μία πλευρά της εικονας .

Στην περίπτωση ενός τετραγώνου έχουμε τις εξής περιπτώσεις .Στην πρώτη περίπτωση αυτό να ακουμπάει τη μία πλευρά της εικόνας ,στη δεύτερη περίπτωση να ακουμπάει τις δύο πλευρές της εκόνας και στην τρίτη περίπτωση να ακουμπάει τρεις πλευρές της εικονας .

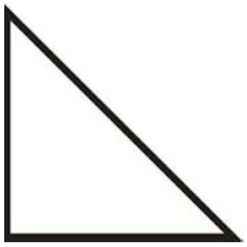
Δηλαδή ,σχεδόν σε όλες τις παραπάνω περιπτώσεις το χρώμα που επικρατεί λιγότερο στις πλευρές της εικόνας είναι αυτό των σχημάτων .Άρα το χρώμα που θα έχουν τα pixels που θα προσθέσουμε στις πλευρές της δυαδικής εικόνας θα έχουν χρώμα ,ίδιο με αυτό που επικρατεί στις πλευρές της δυαδικής εικόνας .

Σε κάποιες περιπτώσεις που δε γίνεται αυτο είναι ,σε ένα τετράγωνο που αυτό μπορεί να ταυτίζεται με το μεγαλύτερο μέρος της εικόνας ,δηλαδή αυτό να τέμνει τις δύο ή και τις τρεις πλευρές της εικόνας ή και στην περίπτωση ενός τριγώνου που πιάνει 2 πλευρές ολόκληρες .Σε αυτές τις

περιπτώσεις βέβαια δε μας νοιάζει γιατί αντίστοιχα έχουμε ένα τετράγωνο ή τρίγωνο στο υπόλοιπο μέρος της εικόνας .Επίσης άλλη μια περίπτωση είναι ένα πολύγωνο μπορεί να επικρατεί στις πλευρές της εικόνας .

Σε αυτό το μέρος του προγράμματος θα κάνουμε τα εξής βήματα .

- 1. Έλεγχο εναλλαγής χρώματος στις πλευρές της εικόνας .
  Άρα για αρχή ελέγχεται εάν η πλευρά του σχήματος της εικονας ταυτίζεται με κάποια πλευρά της δυαδικής εικόνας .Εάν ταυτίζεται ,το χρώμα σε κάποια πλευρά της δυαδικής εικονας (edge1,edge2,edge3,edge4) θα αλλάζει σε κάποιο σημείο ,από μαύρο (τιμή 1) σε άσπρο (τιμή 2) ή και το αντίθετο .Για αυτό το σκοπό ελέγχουμε ένα-ένα τα πλευρικά pixels της δυαδικής εικόνας εάν το χρώμα που έχουν είναι ίδιο (edge1(i-1) ~= edge1(i)) .Εάν δεν είναι τότε σημαίνει ότι οι άκρες του σχήματος που περιέχει η εικόνα ταυτίζονται με τις πλευρές της εικόνας και άρα πρέπει τότε να προσθέσουμε pixels περιφερειακά της δυαδικής εικόνας .Μπορεί βέβαια λόγω ύπαρξης κάποιων κουκίδων στις πλευρές της εικόνας ,λανθασμένα να υπάρχει μια εναλλαγή χρώματος .
- 2. Εάν υπάρχει εναλλαγή χρώματος στις πλευρές της εικόνας θα βρούμε το χρώμα που επικρατεί στις πλευρές της δυαδικής εικόνας (άσπρο ή μαύρο) .
  Το χρώμα που επικρατεί σε κάθε πλευρά της δυαδικής εικόνας θα το βρούμε υπολογίζοντας τη μέση τιμή των pixels αυτών (συνάρτηση mean) .Για αρχή υπολογίζουμε τη μέση τιμή που έχουν τα pixels της κάθε πλευράς της δυαδικής εικόνας ξεχωριστά (meanedge1,meanedge2,meanedge3,meanedge4) και έπειτα όλης της περιφέρειας της εικόνας (meanedge) .Εάν η μέση τιμή όλης της περιφέρειας που θα υπολογίσουμε (συνάρτηση mean) είναι μεγαλύτερη της μέσης τιμής που μπορούν να πάρουν τα pixels μιας δυαδικής εικόνας (0.5) ,τότε σημαίνει ότι το χρώμα που επικρατεί στην περιφέρεια της εικονας είναι άσπρο .Από την άλλη εάν η μέση τιμή όλης της περιφέρειας που θα υπολογίσουμε (συνάρτηση mean) είναι μικρότερη της μέσης τιμής που μπορούν να πάρουν τα pixels μιας δυαδικής εικόνας (0.5) ,τότε σημαίνει ότι το χρώμα που επικρατεί στην περιφέρεια της εικονας είναι μαύρο .
- 3. Θα προσθέσουμε σειρές και στήλες από pixels περιφερειακά της εικόνας ,με χρώμα ίδιο με αυτό που επικρατεί περιφερειακά ης εικόνας .
  Άρα στο τέλος θα προσθέσουμε περιφερειακά της εικόνας pixels με χρώμα ίδιο με αυτό που επικρατεί περισσότερο στην περιφέρεια της εικόνας (όχι αυτού του σχήματος) (padarray(ImageRead,[3 3],_,'both')) .Να επισημάνουμε ότι δεν ισχύει πάντα η περίπτωση να μην επικρατεί το χρώμα του σχήματος .Εάν στο μεγαλύτερο μέρος των πλευρών της εικόνας τέμνουν οι πλευρές του σχήματος ,τότε αλλάζουμε τις ανισότητες παρακάτω .



Κώδικας

```
padindex = false;

edge1 = BW(1,:); % Πάνω πλευρά
edge2 = BW(:,1); % Αριστερή πλευρά
edge3 = BW(end,:); % Κάτω πλευρά
edge4 = BW(:,end); % Δεξιά πλευρά

BWsize = size(BW);
```

Παρακάτω εκτελείται το βήμα 1 που αναφέρεται παραπάνω .

```
if ~padindex
    for i=2:BWsize(2) % Ελέγχουμε όλα τα pixels στην πάνω πλευρά .
        if edge1(i-1) ~= edge1(i)
            padindex = true;
            break
        end
    end
end
if ~padindex
    for i=2:BWsize(1) % Ελέγχουμε όλα τα pixels στην αριστερή πλευρά .
        if edge2(i-1) ~= edge2(i)
            padindex = true;
            break
        end
    end
end
if ~padindex
    for i=2:BWsize(2) % Ελέγχουμε όλα τα pixels στην κάτω πλευρά .
        if edge3(i-1) ~= edge3(i)
            padindex = true;
            break
        end
    end
end
if ~padindex
    for i=2:BWsize(1) % Ελέγχουμε όλα τα pixels στην δεξιά πλευρά .
        if edge4(i-1) ~= edge4(i)
            padindex = true;
            break
        end
    end
end
```

Παρακάτω εκτελείται το βήμα 2 που αναφέρεται παραπάνω .

```
if padindex
    meanedge1 = mean(edge1);
    meanedge2 = mean(edge2);
    meanedge3 = mean(edge3);
    meanedge4 = mean(edge4);
    meanedges = mean((meanedge1 + meanedge2 + meanedge3 + meanedge4));
    if meanedges < 0.5
```

Παρακάτω εκτελείται το βήμα 3 που αναφέρεται παραπάνω .

```
        padImageRead = padarray(ImageRead,[3 3],0,'both');
```

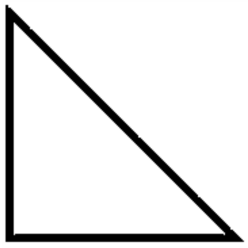
```
        padGRAY = padarray(GRAY,[3 3],0,'both');
        padBW = padarray(BW,[3 3],0,'both');
    end
    if meanedges > 0.5
```

Παρακάτω εκτελείται το βήμα 3 που αναφέρεται παραπάνω .

```
        padImageRead = padarray(ImageRead,[3 3],255,'both');
        padGRAY = padarray(GRAY,[3 3],255,'both');
        padBW = padarray(BW,[3 3],255,'both');
    end
end
```

Παρακάτω αλλάζουμε την μεταβλητή της εικόνας ,ακόμα και εάν δεν προσθέσουμε pixels στην εικόνα .

```
if ~padindex
    padImageRead = ImageRead;
    padGRAY = GRAY;
    padBW = BW;
end

figure,
imshow(padBW),
title('BW image with pad');
```



ΚΕΦΑΛΑΙΟ 3 : Ανάλυση της εικόνας

Εισαγωγή κεφαλαίου

Τα βήματα που κολουθούνται στο παρακάτω κομάτι του προγράμματος είναι τα εξής :

- 1. Αφού αντιστρέψουμε τη δυαδική εικόνα (~ padBW) που έχουμε δημιουργήσει στα παραπάνω κεφάλαια ,θα ανιχνεύσουμε τα σύνορα των αντικειμένων που υπάρχουν στην εικονα με τη συνάρτηση bwboundaries .Τα σύνορα ενός αντικειμένου είναι η περιφέρειά του ,δηλαδή αυτή που οριοθετεί το σχήμα ,με το φόντο της εικόνας .Την εικόνα την αντιστρέφουμε ώστε τα έντονα σημεία της εικόνας (με άσπρο χρώμα) να είναι αυτά του γεωμετρικού σχήματος .Αυτό είναι απαραίτητο για την συνάρτηση regionprops που χρησιμοποιείται αργότερα .
  Η συνάρτηση bwboundaries αριθμεί τα σχήματα που ανιχνεύει ,ώστε να τα ξεχωρίσει .Πιο συγκεκριμένα η συνάρτηση περιέχει τα σημεία που απαρτίζουν το σύνορο κάθε σχήματος .Άρα συγκεκριμένα αριθμεί τα σημεία αυτά ,ώστε να αυτά να ξεχωρίζονται με βάση το σχήμα στο οποίο ανήκουν .
  Η συνάρτηση αυτή επιστρέφει και άλλες παραμέτρους που επεξηγούνται παρακάτω που χρησιμοποιούνται .

- 2. Έπειτα χρησιμοποιείται η συνάρτηση regionprops ώστε να υπολογιστούν κάποιες πληροφορίες για τα σχήματα που ανιχνεύτηκαν από τη συνάρτηση bwboundaries .Αυτές οι πληροφορίες χρειάζονται για να αναγνωριστεί το γεωμετρικό σχήμα των σχημάτων της εικόνας αλλά και για άλλες επεξεργασίες .
  Συγκεκριμένα η συνάρτηση regionprops απομονώνει τα αντικείμενα που έχουν ανιχνευτεί σε ένα τετραγωνικό ή και πολυγωνικό πλαίσιο .Πάνω σε αυτές τα πλαίσια υπολογίζει κάποιες παραμέτρους ,οι οποίες θα χρησιμοποιηθούν για να αναγνωριστεί το γεωμετρικό σχήμα ,αλλά και για να υπολογιστούν και κάποιες άλλοι παράμετροι ,όπως περίμετρος κ.α. .Οι παράμετροι που θα χρησιμοποιηθούν επεξηγούνται παρακάτω .
  Επίσης να γραφτεί ότι η συνάρτηση bwboundaries ανιχνεύει τα σύνορα των σχημάτων της εικόνας .Η συνάρτηση regionprops όμως ,χρησιμοποιώντας ένα μέγεθος 'γειτονιάς' ανιχνεύει και άλλα σημεία του αντικειμένου που συνδέονται μεταξύ τους ,γύρω από τα σύνορα .

- 3. Για την αναγνώριση των σχημάτων στην εικόνα πρέπει να μελετηθεί η περίπτωση που αυτό έχει τρύπα εσωτερικά .Στην περίπτωση που το γεωμετρικό σχήμα που έχει εντοπιστεί δεν είναι 'γεμάτο' εσωτερικά ,τότε τα σημεία (pixels) του σχήματος είναι λιγότερα σε σχέση με όλα τα pixels που βρίσκονται εντός του ορθογώνιου πλαισίου (BoundingBox) ή και του πολυγωνικού πλαισίου(ConvexHull) .Αυτό έχει σαν αποτέλεσμα οι τιμές των μεταβλητών Extent ή και η Solidity (οι οποίες επεξηγούνται παρακάτω) να είναι μικρότερες των αναμενομένων ,ανεξαρτήτως γεωμετρικού σχήματος .Αυτό έχει σαν αποτέλεσμα όλα τα γεωμετρικά σχήματα της εικόνας να αναγνωρίζονται λανθασμένα ,κυρίως ως τρίγωνα .Άρα για αυτό πρέπει να δούμε εάν τα γεωμετρικά σχήματα της εικόνας είναι γεμάτα εσωτερικά .

- 4. Στην περίπτωση που το αντικείμενο έχει τρύπα εσωτερικά θα πρέπει να ακολουθηθεί ένα επιπλέον μονοπάτι ,για να πάρουμε το εσωτερικό τμήμα των γεωμετρικών σχημάτων

ξεχωριστά .Με αυτό το μονοπάτι θα πάρουμε το εσωτερικό τμήμα των γεωμετρικών σχημάτων και πιο συγκεκριμένα ,από το εξωτερικό σύνορο του σχήματος και μέσα .Αυτό δεν σημαίνει ότι δεν μπορούμε να πάρουμε από το εσωτερικό σύνορο του σχήματος και μέσα .
Έχοντας στο τέλος το εσωτερικό τμήμα του γεωμετρικού σχήματος ,οι παράμετροι που θα υπολογίσει η συνάρτηση regionprops για το γεωμετρικό σχήμα αυτό ,θα αντιπροσωπεύουν σωστότερα το σχήμα του πλαισίου .

- 5. Έπειτα θα γίνει μια ανάλυση των ιδιοτήτων που υπολογίζει η συνάρτηση regionprops για κάθε πλαίσιο .Η συνάρτηση regionprops επιστρέφει ένα αντικείμενο (προγραμματιστικά αντικείμενο) όπου για κάθε πλαίσιο που αυτό περιέχει ,έχει υπολογίσει κάποιες πληροφορίες που θα χρησιμοποιηθούν για να αναγνωριστεί το σχήμα (τρίγωνο,ορθογώνιο,κ,τ,λ,) του αντίστοιχου πλαισίου .
Κάποιες πληροφορίες που υπολογίζει η συνάρτηση regionprops είναι το ποσοστό που καταλαμβάνουν τα σημεία του σχήματος μέσα στο πλαίσιο που βρίσκονται .Πληροφορίες χρήσιμες για να αναγνωριστούν τα αντικείμενα της εικόνας .
Η συνάρτηση regionprops υπολογίζει για κάθε πλαίσιο τετραγωνικό (BoundingBox) ή πολυγωνικό (ConvexHull) ,όλα τα pixels του σχήματος (Area) .Επίσης η συνάρτηση υπολογίζει το ποσοστό που καταλαμβάνουν τα pixel του σχήματος (Area) ,μέσα στο πολυγωνικό πλαίσιο (ConvexHull) .Δηλαδή κάνει την πράξη Area/ConvexHull και το αποτέλεσμα το επιστρέφει στην μεταβλητή Solidity .
Επίσης η συνάρτηση υπολογίζει το ποσοστό που καταλαμβάνουν τα pixel του σχήματος (Area) ,μέσα στο τετραγωνικό πλαίσιο (BoundingBox) .Δηλαδή κάνει την πράξη Area/ BoundingBox και το αποτέλεσμα το επιστρέφει στην μεταβλητή Extent .
Κάτι άλλο που υπολογίζει η συνάρτηση είναι η κυκληκότητα που έχει σχηματικά το σχήμα (Eccentricity) .Η συνάρτηση υπολογίζει μια έλλειψη ,όπου περιβάλει το σχήμα .Αναλόγως το σχήμα η έλλειψη μπορεί να είναι ένας κύκλος ή και μια γραμμή .Μια έλλειψη που τείνει να γίνει κύκλος ,η εκκεντικότητα τείνει στο 0 ,ενώ μια έλλειψη που τείνει να γίνει γραμμή ,η εκκεντικότητα τείνει στο 1 .
Επίσης η συνάρρτηση βρίσκει τα 8 ακραία σημεία που έχει η περιοχή .Η λογική λέει ότι μπορούμε να χρησμοποιήσουμε τα ακραία αυτά σημεία για να βρεθεί το σχήμα του αντικειμένου .Για να γίνει αυτό όμως πρέπει οι πλευρές του σχήματος να είναι ευθείες αλλιώς θα βγαίνουν λάθος συμπεράσματα .Παρακάτω είναι ο πίνακας Extrema που υπολογίζεται από τη συνάρτηση regionprops .

```
a = STATS(i).Extrema(1,:); % πάνω-πάνω αριστερά
b = STATS(i).Extrema(7,:); % αριστερά-αριστερά πάνω
c = STATS(i).Extrema(2,:); % πάνω-πάνω δεξιά
d = STATS(i).Extrema(3,:); % δεξιά-δεξιά πάνω
e = STATS(i).Extrema(4,:); % δεξιά-δεξιά κάτω
f = STATS(i).Extrema(5,:); % κάτω-κάτω δεξιά
g = STATS(i).Extrema(6,:); % κάτω-κάτω αριστερά
h = STATS(i).Extrema(8,:); % αριστερά-αριστερά κάτω
```

Παράδειγμα κώδικα χρήσης του πίνακα Extrema .

```
if a == b & c == d & e == f & g == h
   -εντολές-
end
```

- 6. Μετά υπολογίζονται διάφορα χαρακτηριστικά του γεωμετρικού σχήματος αυτού ,όπως περίμετρος ,εμβαδόν κ.α. .

- 7. Ένα ακόμα χαρακτηριστικό είναι το χρώμα κάθε γεωμετρικού σχήματος .Το χρώμα κάθε pixel αποτελεί συνδυασμό τριών χρωμάτων ,του κόκκινου ,του πράσινου και του μπλε .Άρα μελετώντας τις τιμές των τριών αυτών χρωμάτων ,όλων των pixel του γεωμετρικού σχήματος ,μπορούμε να βρούμε το χρώμα του γεωμετρικού σχήματος αυτού .

Κώδικας

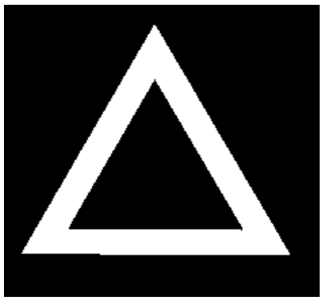
Παρακάτω αντιστρέφουμε τα χρώματα της δυαδική εικόνας (άσπρο-μαύρο) .Για να ανιχνευτούν τα σημεία του σχήματος από τη συνάρτηση regionprops και όχι του φόντου ,πρέπει αυτά να έχουν άσπρο χρώμα .Για αυτό το λόγο αντιστρέφουμε την δυαδική εικόνα παρακάτω .

```
invertedpadBW = ~ padBW;

figure,
imshow(invertedpadBW),
title('Inverted Binary Image (~padBW)');
```

Ξεκινώντας από την αρχή ,παρακάτω απεικονίζονται η αρχική εικόνα και η άχρωμη .Η άχρωμη εικόνα βέβαια δε ξεχωρίζει από την αρχική λόγω του χρώματος της αρχικής εικόνας (ασπρόμαυρη) .

Παρακάτω βλέπουμε τη δυαδική εικόνα ,χωρίς και με pad .Τα επιπλέον pixel λόγω χρώματος δεν φαίνονται .

Παρακάτω βλέπουμε τη δυαδική εικόνα αφού αντιστρέψαμε τα χρώματα .



Αναλόγως το αντικείμενο που προσπαθεί να ανιχνευτεί ,πρέπει να γίνει και η ανάλογη ανίχνευση των χαρακτηριστικών που ξεχωρίζουν στο αντικείμενο αυτό .Ο εντοπισμός απλών σχημάτων ,μπορεί να γίνει από τη διαφοροποίηση που έχει το σχήμα αυτό μπροστά στο φόντο της εικόνας .Οποιαδήποτε σχήμα σε μια εικόνα που διαφέρει από το φόντο της εικόνας γίνεται αντιληπτό .Πιο συγκεκριμένα ,αρκεί η περιφέρεια του σχήματος αυτού να διαφέρει από το φόντο της εικόνας .Επομένως εντοπίζοντας τις περιφέρειες που διαφέρουν στην εικόνα ,γίνεται και ο εντοπισμός των σχημάτων που επιθυμούμε .Οι περιφέρειες εντοπίζονται από τη συνάρτηση bwboundaries .

Η συνάρτηση bwboundaries εντοπίζει τα 'σύνορα' των αντικειμένων που υπάρχουν στην δυαδική εικόνα που δίνεται ως παράμετρος .Να επισυμανθεί ότι η συνάρτηση εντοπίζει τόσο τα εξωτερικά όσο και τα εσωτερικά 'σύνορα' των αντικειμένων ,π.χ. στην περίπτωση που το αντικείμενο έχει τρύπες .Παρακάτω η συνάρτηση επιστρέφει μια παράμετρο L που περιέχει τα σημεία των αντικειμένων αριθμημένα ,αναλόγως το αντικείμενο που ανήκουν .Η συνάρτηση αυτή επιστρέφει και άλλες παραμέτρους που επεξηγούνται παρακάτω που χρησιμοποιούνται .Έπειτα μελετούνται οι πληροφορίες κάθε αντικειμένου ξεχωριστά .

```
[~,L] = bwboundaries(invertedpadBW ,'noholes');
```

Τα σχήματα της εικόνας (ανεξαρτήτου σχήματος) πρέπει να απομονωθούν μεταξύ τους ώστε να αναλυθούν ατομικά .Τα σχήματα της εικόνας ξεχωρίζονται κατά τον εντοπισμό τους ,δηλαδή από τη συνάρτηση bwboundaries ,αλλά απομονώνονται μεταξύ τους από τη συνάρτηση regionprops .Μαζί με την απομόνωση ,η συνάρτηση regionprops υπολογίζει διάφορες ιδιότητες για το σχήμα αυτό ,ώστε να μπορεί να γίνει η ανάλυση αυτού του σχήματος .

Ένα πλεονέκτημα της συνάρτησης regionprops που πρέπει αναφέρουμε ,είναι η ανίχνευση όλου του γεωμετρικού σχήματος και όχι μόνο του συνόρου που έχει ανιχνεύσει η συνάρτηση bwboundaries .Πιο συγκεκριμένα ,η συνάρτηση regionprops ,χρησιμοποιώντας ένα μέγεθος 'γειτονιάς' ανιχνεύει και άλλα σημεία του σχήματος που συνδέονται μεταξύ τους ,γύρω από τα σύνορα που έχει εντοπίσει η συνάρτηση bwboundaries .Με τη λογική ότι τα σχήματα της εικόνας απέχουν αρκετά ,ώστε να μην πέφτουν μέσα στη 'γειτονιά' ,τα σχήματα μεταξύ τους απομονώνονται μεταξύ τους .

```
STATS1 = regionprops(L, 'all');

%%
```

Παρακάτω δηλώνουμε ότι οι μεταβλητές recbboxes ,trianbboxes ,polbboxes και boxes ότι είναι πίνακες .Δεν καταχωρούμε κάποια τιμή σε αυτούς ([ ]) .

```
%
shapescenter = [];
boxes = [];
ids = [];
trianbboxes = [];
recbboxes = [];
polbboxes = [];
circle = [];
%
showinimageindex = [];
%
```

Παρακάτω δημιουργούμε τις δομές lineRegionsPixels και houghtransformline ,καθώς και το διάνυσμα lineids .

```
%

lineRegionsPixels = struct('pixels',{});
lineids = [];
houghtransformline = struct('point1',{} ,'point2',{});

%
```

Παρακάτω αρχικοποιούνται οι παρακάτω μεταβλητές .

```
%

holeindex = false;
STATS1in = 0;
STATS2in = 0;
n = 1;
q = 1;
r = 1;

%

for i = 1 : length(STATS1)

    %
```

Οι παρακάτω μεταβλητές φανερώνουν τις τιμές Eccentricity ,Extent ,Solidity και Area των αντικειμένων που περιέχει το αντικείμενο STATS1 .Χρησιμη πληροφορία στην περίπτωση που θέλουμε να δούμε τα χαρακτηριστικά του σχήματος ,ενός πλαισίου .

```
    %

    ec1 = STATS1(i).Eccentricity
    ex1 = STATS1(i).Extent
    sol1 = STATS1(i).Solidity
    area1 = STATS1(i).Area

    %
```

Παρακάτω βλέπουμε το σχήμα που έχει ανιχνευτεί από τη συνάρτηση bwboundaries .

Η εικόνα που παίρνουμε παρακάτω αποτελεί ένα ορθογώνιο πλαίσιο του αντικειμένου STATS1 .Το σχήμα στο πλαίσιο δεν αποτυπώνει τη θέση του σχήματος πάνω στην αρχική εικόνα ,όπως

συμβαίνει με την εικόνα RegionMask .

```
%
regionimage1 = STATS1(i).Image;
%
figure;
imshow(regionimage1);title('regionimage of ~padBW')
```

Αργότερα θα χρειαστεί να βρεθεί το χρώμα του κάθε αντικειμένου (γεωμετρικού σχήματος .Για να γίνει αυτό ,πρέπει να πάρουμε τα σημεία (pixels) που απεικονίζουν το αντικείμενο αυτό στην αρχική εικόνα .

Για αρχή από το αντικείμενο STATS1 παίρνουμε τις συντεταγμένες των σημείων (pixels) κάθε αντικειμένου ξεχωριστά (STATS1.PixelList) .Να πούμε ότι η συνάρτηση bwboundaries ανιχνεύει τα σύνορα των αντικειμένων της εικόνας .Η συνάρτηση regionprops όμως ,χρησιμοποιώντας ένα μέγεθος 'γειτονιάς' ανιχνεύει και άλλα σημεία του αντικειμένου που συνδέονται μεταξύ τους ,γύρω από τα σύνορα .Τέλος παίρνουμε τις συντεταγμένες των pixels κάθε σχήματος και τα τοποθετούμε το ένα κάτω από το άλλο με την συνάρτηση vertcat ,δημιουργώντας τον πίνακα RegionsPixels1 .Ο πίνακας RegionsPixels1 έχει σε κάθε γραμμή τις συντεταγμένες ενός pixel .Με αυτό τον τρόπο καταφέραμε από την εικόνα να εξάγουμε τα σημεία (pixels) ενός αντικειμένου της εικόνας (RegionsPixels1) .

Έπειτα δημιουργείται μια δυαδική εικόνα ,στο μέγεθος της αρχικής ,η οποία θα εμφανίζει με άσπρο χρώμα μόνο το συγκεκριμένο αντικείμενο του πλαισίου ,με το υπόλοιπο μέρος της εικόνας να είναι μαύρο .

Για αρχή δημιουργούμε μόνο το φόντο της εικόνας (RegionMask) (συνάρτηση false) .Έπειτα θα ζωγραφίσουμε με άσπρο χρώμα τα σημεία της εικονας που αντιστοιχούν στο συγκεκριμένο σχήμα .Τα pixels της αρχικής εικόνας που ανήκουν στο συγκεκριμένο σχήμα τα έχουμε πάρει προηγουμένως από το αντικείμενο STATS1 (RegionsPixels1) .Έπειτα όμως πρέπει να βρούμε τα pixels της δυαδικής εικόνας σε πια pixel αντιστοιχούν πάνω στην αρχική εικόνα .Η αντιστοίχιση γίνεται με τη συνάρηση sub2ind .

Μαζί με την δυαδική εικόνα θα δημιουργήσουμε και έγχρωμη εικόνα ,πάλι στο μέγεθος της αρχικής ,η οποία θα εμφανίζει το συγκεκριμένο αντικείμενο του πλαισίου με το αρχικό του χρώμα ,με το υπόλοιπο μέρος της εικόνας να είναι μαύρο .Μια έγχρωμη εικόνα έχει τρία επίπεδα .Το κάθε επίπεδο αντιστοιχεί σε ένα από τα 3 βασικά χρώματα ,το κόκκινο ,το πράσινο και το μπλε ([R G B]) .Για αυτό το λόγο θα αντιγράψουμε τη δυαδική εικονα RegionMask τρεις φορές ,μία για κάθε επίπεδο ,με χρήση της συνάρτησης repmat .Έπειτα παίρνουμε τα pixels που δεν αντιστοιχούν στο συγκεκριμένο αντικείμενο και τα μηδενίζουμε ώστε να έχουν μαύρο χρώμα .

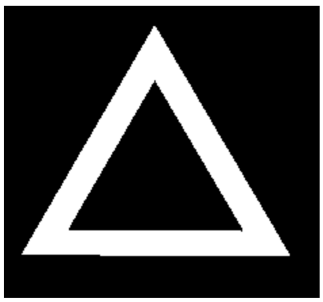
Τέλος εμφανίζουμε την δυαδική εικόνα και την ίδια εικόνα (την δυαδική) με τα χρώματα που έχει στην αρχική .

Όπως παρατηρουμε οι συντεταγμένες που επιστρέφει η συνάρηση regionprops είναι αντιστραμένες ,δηλαδή πρώτα επιστρέφει την y-συντεταγμένη και μετά την x-συντεταγμένη .

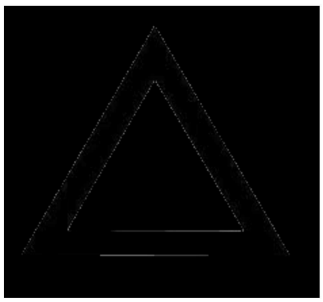
```
%
RegionsPixels1 = vertcat(STATS1(i).PixelList);
%
RegionMask1 = false(size(padGRAY));
ind = sub2ind(size(RegionMask1), RegionsPixels1(:,2),...
    RegionsPixels1(:,1));
RegionMask1(ind) = true;
%
displayImage = padImageRead;
displayImage(~repmat(RegionMask1,1,1,3)) = 0;
%
```

Παρακάτω θα απεικονίσουμε τις 2 εικόνες που δημιουργήσαμε παραπάνω .

```
%
figure;
imshow(RegionMask1); title('RegionMask1')
```



```
figure;
imshow(displayImage); title('RegionImage in the original image')
```



Ένας διαφορετικός τρόπος απεικόνισης του σχήματος ενός πλαισίου ,στην αρχική εικόνα ,είναι με τη συνάρτηση plot .Με τη συνάρτηση plot μπορούμε να τονίσουμε τα σημεία του αντικειμένου που απεικονίζει το πλαίσιο πάνω στην αρχική εικόνα . Για αυτό θα χρειαστούν οι συντεταγμένες των σημείων (x και y) του πλαισίου ,ώστε να καταλάβει η συνάρτηση plot ποια σημεία θα τονίσει πάνω στην αρχική εικόνα .Αυτές περιέχονται στο ατικείμενο που επιστρέφει η συνάρτηση regionprops (STATS1.PixelList) .

Το πρόβλημα όμως είναι ότι η αρχική εικόνα και το πλαίσιο δεν έχουν τις ίδιες διαστάσεις και άρα οι συντεταγμένες των σημείων του σχήματος του πλαισίου δεν είναι ίδιες με αυτού του σχήματος στην αρχική εικονα .

Προσοχή ,οι συντεταγμένες που επιστρέφει το αντικείμενο STATS1 είναι ανάποδα ,δηλαδή πρώτα επιστρέφει την y-συντεταγμένη και μετά την x-συντεταγμένη .

```
%
% figure;
% imshow(padImageRead);
% title('RegionImage of ~padBW with plot') ,hold on;
% plot(RegionsPixels1(:,2),RegionsPixels1(:,1), 'r','LineWidth',3);
%
```

Το μέγεθος του πίνακα RegionsPixels1 παρακάτω το παίρνουμε για να υπολογίσουμε αργότερα το χρώμα (RGB) που έχει κάθε pixel του σχήματος του πλαισίου .Τα pixels του σχήματος του πλαισίου βρίσκονται στον πίνακα RegionsPixels1 .

```
%
RegionsPixels1size = size(RegionsPixels1);
%
```

Υπάρχει όμως μια περίπτωση που περιπλέκει την παραπάνω λειτουργία .Η περίπτωση που το σχήμα δεν είναι γεμάτο εσωτερικά .Οι παράμετροι που υπολογίζει η συνάρτηση regionprops για κάθε πλαίσιο εξαρτώνται από το ποσοστό που καταλαμβάνουν τα σημεία του σχήματος της εικόνας ,μέσα στο πλαίσιο .

Με απλά λόγια ,ένα 'κούφιο' τετράγωνο δεν φανερώνεται από τις παραμέτρους που επιστρέφει η συνάρτηση regionprops ότι είναι τετράγωνο .Για αυτό πρέπει να γεμίσουμε το εσωτερικό τμήμα του σχήματος της εικόνας και μετά να γίνει ανάλυση του σχήματος .Το γέμισμα του σχήματος εσωτερικά προυποθέτει κάποιες επιπλέον επεξεργασίες της εικόνας .

Επεξεργασίες που απαιτούνται είναι η εύρεση της τρύπας που μπορεί να έχει το σχήμα στην εικόνα . Η συνάρτηση bwboundaries πέρα από τις συντεταγμένες των σημείων ενός αντικειμένου που απαρτίζουν το εξωτερικό σύνορο του αντικειμένου ,επιστρέφει και τα σημεία που απαρτίζουν και το εσωτερικό σύνορο του αντικειμένου ,όπως όταν αυτό έχει μια τρύπα εσωτερικά .

Τα κελιά (B(q)) που περιέχουν τις συντεταγμένες των σημείων που απαρτίζουν το εσωτερικό σύνορο είναι μετά από τα κελιά που περιέχουν τις συντεταγμένες των σημείων που απαρτίζουν το εξωτερικό σύνορο .Τον αριθμό των κελιών που απαρτίζουν το εξωτερικό σύνορο δείχνει η μεταβλητή N που παίρνουμε από τη συνάρτηση bwboundaries .Άρα τα κελιά που απαρτίζουν το εσωτερικό σύνορο είναι τα B(i) όπου i > N .

Επίσης απαιτούνται επεξεργασίες γεμίσματος του εσωτερικού του αντικειμένου εάν αυτό έχει τρύπα εσωτερικά .Για να γεμίσουμε το σχήμα εσωτερικά ξεκινάμε ως εξής .Αντί να ανιχνεύσουμε

τα σχήματα της εικόνας ,ανιχνεύουμε το φόντο της εικόνας με τη συνάρτηση regionprops .Με αυτή τη μέθοδο ,το κάθε σχήμα της εικόνας εντός του πλαισίου που το έχει απομονώσει η συνάρτηση regionprops ,έχει μαύρο χρώμα και γεμάτο το εσωτερικό του .Βέβαια όπως είπαμε οι ιδιότητες που υπολογίζει η συνάρτηση regionprops εξαρτώνται από τα άσπρα σημεία του πλαισίου .Άρα αντιστρέφουμε τα χρώματα του πλαισίου αυτού και έπειτα χρησιμοποιούμε ξανά τη συνάρτηση regionprops στο πλαίσιο αυτό .

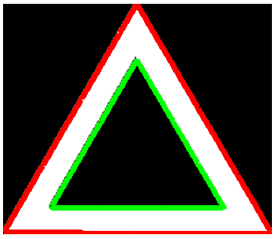
Παρακάτω θα δημιουργήσουμε μια εικόνα με που θα εμφανίζει το εσωτερικό σύνορο του αντικειμένου (RegionMask3) .Η ανάγκη να δημιουργήσουμε αυτήν την εικόνα είναι για να υπολογίσουμε την τρύπα που έχει εσωτερικά το αντικείμενο ,μέσω της συνάρτησης regionprops .Εάν η τρύπα που έχει εσωτερικά το αντικείμενο είναι μεγάλη τότε δημιουργείται το πρόβλημα που αναφέρουμε παραπάνω και σε αυτήν την περίπτωση πρέπει να ακολουθήσουμε και άλλο μονοπάτι πριν την αναγνώριση του σχήματος που περιέχει η εικόνα .

Επίσης η ακτίνα από το κέντρο μέχρι το εσωτερικό σύνορο χρειάζεται να υπολογιστεί για να βρεθεί το εμβαδόν του γεωμετρικού σχήματος που έχει ένα 'κενό' εσωτερικά του .Σε αυτήν την περίπτωση πρέπει να βρεθεί το 'εσωτερικό' εμβαδόν ,έπειτα το 'εξωτερικό' εμβαδόν ,ώστε αφαιρώντας τα ,να βρεθεί το εμβαδόν του γεωμετρικού σχήματος .

```
%
[B3,L3,N3,A3] = bwboundaries(regionimage1);
%
```

Παρακάτω δηλώνουμε τον πίνακα inboundaries .

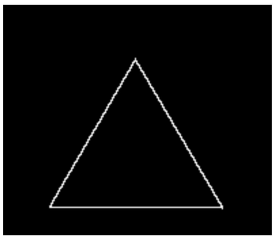
```
%
inboundaries = [];
outboundaries = [];
%
figure;
imshow(regionimage1);
title('out-boundaries and in-boundaries of the object')
,hold on;
%
for m=1:length(B3)
    boundary = B3{m};
    if(m > N3)
        inboundaries = vertcat(inboundaries,boundary);
        plot(boundary(:,2), boundary(:,1), 'g','LineWidth',3);
    else
        outboundaries = vertcat(outboundaries,boundary);
        plot(boundary(:,2), boundary(:,1), 'r','LineWidth',3);
    end
end
```



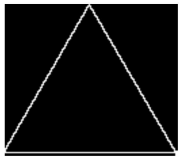
```
if ~isempty(inboundaries)
    RegionMask3 = false(size(regionimage1));
    ind2 = sub2ind(size(RegionMask3), inboundaries(:,1),...
        inboundaries(:,2));
    RegionMask3(ind2) = true;
    %
```

Γίνεται η υπενθύμιση ότι η εικόνα RegionMask3 (τίτλος `in-boundaries of the object)` απεικονίζει το 'εσωτερικό' σύνορο του αντικειμένου του πλαισίου regionimage1 (τίτλος `out-boundaries and in-boundaries of the object)` .Τα σημεία που απεικονίζουν το 'εσωτερικό' σύνορο στην μία εικόνα (με πράσινο χρώμα) και στην άλλη (RegionMask3) έχουν τις ίδιες συντεταγμένες (inboundaries) .

```
%
figure;
imshow(RegionMask3);
title('in-boundaries of the object')
```



```
%
[~,L4] = bwboundaries(RegionMask3 ,'noholes');
STATS4 = regionprops(L4, 'all');
%
regionimage4 = STATS4.Image;
%
figure;
imshow(regionimage4);
title('regionimage of the in-boundaries')
%
```



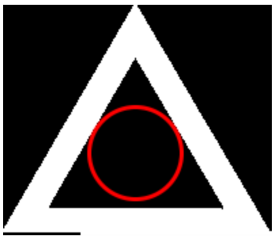
Παρακάτω υπολογίζεται το μέγεθος της τρύπας που έχει εσωτερικά το γεωμετρικό σχήμα .Το μέγεθος της τρύπας υπολογίζεται ως η μικρότερη απόσταση που υπάρχει από το κέντρο της τρύπας εώς την άκρη αυτής ,δηλαδή το εσωτερικό σύνορο του σχήματος .Για να βρούμε την απόσταση αυτή κάνουμε τα εξής βήματα .

Πρώτα βρίσκουμε το κέντρο της τρύπας αυτής (STATS1(i).Centroid) .Λογω ότι τις συντεταγμένες αυτές θα τις χρησιμοποιήσουμε ως δείκτη στον πίνακα onepixelsposition τις στρογγυλοποούμε αργότερα (συνάρτηση round) .Το κέντρο κάθε σχήματος ενός πλαισίου το υπολογίζει η συνάρτηση regionprops (STATS1(i).Centroid) .

Έπειτα θα προσπαθήσουμε να υπολογίσουμε την απόσταση που ξεκινάει από το κέντρο της τρύπας και τελειώνει στο εσωτερικό σύνορο του σχήματος .Το σημείο στο εσωτερικό σύνορο είναι το πιο κοντινό σημείο ως προς το κέντρο με τιμή 1 .Με βάση αυτό το κριτήριο μπορούμε να βρούμε την απόσταση αυτή ως εξής :

Θα βρούμε το πιο κοντινό σημείο ως προς το κέντρο του σχήματος με τιμή 1 .Η συνάρτηση bwdist επιστρέφει έναν πίνακα (distanceImage ή και τον onepixelsposition) που περιέχει για κάθε pixel της εικόνας το πιο κοντινό pixel σε αυτό με τιμή 1 .Το pixel που μας ενδιαφέρει είναι το pixel που αποτελεί το κέντρο του σχήματος ,όπου και το παίρνουμε από τη συνάρτηση regionprops (STATS2.Centroid) .Άρα μέσω του πίνακα distanceImage βρίσκουμε το πιο κοντινό pixel με τιμή 1 ως προς το κέντρο του σχήματος .

```
%
[distanceImage4 ,onepixelsposition4] = bwdist(RegionMask3);
%
for t=1:length(STATS4)
    trregioncenter4 = round(STATS4(t).Centroid);
    %
    traktina4 = distanceImage4(trregioncenter4(2),trregioncenter4(1));
    traktina4matrix(t) = traktina4;
    %
    circle_characteristicts4 = [trregioncenter4 traktina4];
    %
    circle_in_triangle = insertShape(double(regionimage1),'Circle',...
        circle_characteristicts4,'Linewidth',3,'Color','red');
    figure;
    imshow(circle_in_triangle);
    title('distance of the in-boundaries')
```



```
end
 traktina4max = min(traktina4matrix);
 %
```

Το όριο που θέτεται παρακάτω δεν είναι αποκλειστικό .Αυτό πρέπει να μελετηθεί με βάση τα αποτελέσματα που παίρνουμε από τη συνάρτηση regioprops .

```
 %
 if traktina4max > 28
     holeindex = true;
     %
```

Για τους λόγους που αναφέραμε παραπάνω ,παρακάτω θα ακολουθήσουμε ένα επιπλέον μονοπάτι για να πάρουμε το εσωτερικό τμήμα των γεωμετρικών σχημάτων και πιο συγκεκριμένα ,από το εξωτερικό σύνορο του σχήματος και μέσα .Αυτό δεν σημαίνει ότι δεν μπορούμε να πάρουμε από το εσωτερικό σύνορο του σχήματος και μέσα .

Λοιπόν για αρχή θα βρούμε τα 'σύνορα' (συνάρτηση bwboundaries) των σχημάτων που έχει η δυαδική padBW και όχι η ~padBW ,όπως κάνουμε αρχικά .Αυτό το κάνουμε γιατί παίρνοντας έπειτα τα πλαίσια (συνάρτηση regionprops) που περιέχουν τα αντικείμενα με τα σύνορα αυτά (regionimage2 = STATS2.Image) ,παίρνουμε το γεωμετρικό σχήμα του πλαισίου αυτού ,με γεμάτο το εσωτερικό όμως .

Το πρόβλημα μετά όμως είναι ότι η συνάρτηση regionprops έχει πάρει ως σημεία (τα pixels με άσπρο χρώμα) όλα τα σημεία εξωτερικά του γεωμετρικού σχήματος αυτού .Για να λύσουμε και αυτό το πρόβλημα αντιστρέφουμε την δυαδική εικόνα του πλαισίου αυτού και έπειτα ξαναβρίσκουμε τα 'σύνορα' και ξαναυπολογίζουμε τις διάφορες ιδιότητες του αντικειμένου αυτού .Παίρνοντας πάλι την δυαδική εικόνα της περιοχής αυτής βλέπουμε ότι τα σημεία του πλαισίου αυτού είναι όλα τα σημεία εσωτερικά του γεωμετρικού σχήματος .

```
%
[~,L2] = bwboundaries(padBW, 'noholes');
STATS2 = regionprops(L2, 'all');
%
regionimage2a = STATS2.Image;
%
figure;
imshow(regionimage2a);
title('regionimage of BW')
%
```

Ένας διαφορετικός τρόπος απεικόνισης είναι με τη συνάρτηση plot .Με τη συνάρτηση plot μπορούμε να τονίσουμε τα σημεία του αντικειμένου που απεικονίζει η περιοχή πάνω στην αρχική εικόνα .

Για αυτό θα χρειαστούν οι συντεταγμένες των σημείων (x και y) του σχήματος μέσα στο πλαίσιο ,ώστε να καταλάβει η συνάρτηση plot ποια σημεία θα τονίσει στην αρχική εικόνα .Αυτές περιέχονται στο αντικείμενο που επιστρέφει η συνάρτηση regionprops (STATS2.PixelList) .

Το πρόβλημα είναι ότι η αρχική εικόνα και το πλαίσιο δεν έχουν τις ίδιες διαστάσεις και άρα οι συντεταγμένες των σημείων του σχήματος στο πλαίσιο και στην αρχική εικονα δεν είναι ίδιες .

Προσοχή ,οι συντεταγμένες που επιστρέφει το αντικείμενο STATS2 είναι αντεστρεμένες ,δηλαδή πρώτα επιστρέφει την y-συντεταγμένη και μετά την x-συντεταγμένη .

```
            %
            % RegionsPixels2 = vertcat(STATS2.PixelList);
            %
            % RegionsPixels2size = size(RegionsPixels2);
            %
            % figure;
            % imshow(padImageRead);,
            % title('RegionImage of BW with plot') ,hold on;
            %
            % for t=1:RegionsPixels2size(1)
            %    plot(RegionsPixels2(t,2),RegionsPixels2(t,1),
'r','LineWidth',3);
            % end
            %
            [~,L2] = bwboundaries(~regionimage2a, 'noholes');
            STATS2 = regionprops(L2, 'all');
            %
            regionimage2b = STATS2.Image;
            %
            figure; imshow(regionimage2b);
            title('regionimage of the ~(regionimage of BW)')
```

```
    end
end
%
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση συγκεκριμένων γεωμετρικών σχημάτων οπου καταλαμβάνουν στο αντικείμενο STATS1 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS1 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 0;
%
```

Όπως έχει αναφερθεί ,η ανίχνευση του γεωμετρικού σχήματος του αντικειμένου έχει χωριστεί σε δύο τμήματα .Ένα για την περίπτωση που το αντικείμενο είναι ΄πλήρες΄εσωτερικά ,δηλαδή δεν έχει τρύπα και ένα τμήμα για την περίπτωση που έχει .

Αυτά τα τμήματα πρέπει να χωρίζονται μεταξύ τους γιατί δεν είναι ίδιες οι ιδιότητες των αντικειμένων αυτών (τρύπα και χωρίς τρύπα) που υπολογίζονται από τη συνάρτηση regionprops .Άμα δεν χωρίζονται οι περιοχές μεταξύ τους ,αυτό μπορεί να έχει σαν αποτέλεσμα εκτελεστούν και τα δύο τμήματα .Ο διαχωρισμός των δύο τμημάτων αυτών γίνεται με τη μεταβλητή holeindex .Όταν η τιμή της μεταβλητής αυτής είναι 1 (true) τότε σημαίνει ότι το αντικείμενο έχει τρύπα εσωτερικά του και όταν η τιμή της μεταβλητής αυτής είναι 0 (false) τότε σημαίνει ότι το αντικείμενο δεν έχει τρύπα εσωτερικά του .

```
%
if ~holeindex
    if STATS1(i).Area > 1000
```

Παρακάτω ελέγχουμε για τρίγωνα ,στην περίπτωση που αυτά δεν έχουν τρύπα εσωτερικά .Οι προυποθέσεις ανίχνευσης των τριγώνων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές . Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
            if STATS1(i).Eccentricity > 0.11 && STATS1(i).Eccentricity < 0.82
&&...
                    STATS1(i).Extent > 0.33 && STATS1(i).Extent < 0.53 &&...
                    STATS1(i).Solidity > 0.97 && STATS1(i).Solidity <= 1
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα χωρίς τρύπα στο εσωτερικό του .

```
%
STATS1in = 1
%
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,χωρίς τρύπα εσωτερικά ,που έχει αναγνωριστεί ως τρίγωνο .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως τρίγωνο .

```
%
ectr1 = STATS1(i).Eccentricity
extr1 = STATS1(i).Extent
soltr1 = STATS1(i).Solidity
areatr1 = STATS1(i).Area
%
```

Παρακάτω αποθηκεύουμε το κέντρο του συγκεκριμένου γεωμετρικού σχήματος που υπάρχει μέσα στο πλαίσιο ,στον πίνακα shapescenter .Η χρησιμότητα του πίνακα αυτού είναι να μην αναγνωρίσουμε το ίδιο σχήμα δύο ή περισσότερες φορές .

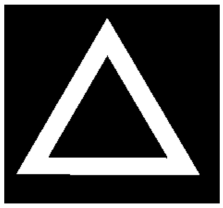
Ο έλεγχος για τυχόν διπλή αναγνώριση του γεωμετρικού σχήματος αυτού γίνεται στο αντικείμενο STATS2 .Το πρόβλημα δεν βρίσκεται στην περίπτωση το ίδιο αντικείμενο (STATS1 ή STATS2) να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα παραπάνω από μία φορά ,αλλά ένα άλλο αντικείμενο που επιστρέφει η συνάρτηση regionprops ,να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα με ένα άλλο αντικείμενο που έχει δώσει η συνάρτηση regionprops .

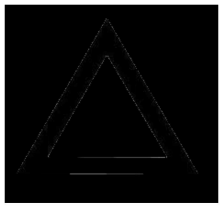
```
%
centertr1 = STATS1(i).Centroid;
shapescenter(n,:) = vertcat(centertr1);
%
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
%
figure;
imshow(regionimage1),title(n)
%
figure;
imshow(RegionMask1); title('RegionMask1')
%
figure;
imshow(displayImage); title('RegionImage in the original image')
%
```

Οι παρακάτω εικόνες ,δεν είναι σε πραγματική κλίμακα .Η πραγματική κλίμακα των παρακάτω εικόνων φαίνεται παραπάνω .





Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις trids για τα τρίγωνα ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η trids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε (στη συγκεκριμένη περίπτωση τρίγωνα) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (trids κ.τ.λ.) μπορεί να διαφέρει .

```
%
boxes(n,:) = STATS1(i).BoundingBox;
ids(n) = n;
%
trianbboxes(n,:) = STATS1(i).BoundingBox;
trperimetros(n) = STATS1(i).Perimeter;
%
```

Παρακάτω υπολογίζεται το εμβαδόν του τριγώνου εντός του πλαισίου αυτού .Ένας τρόπος για να βρούμε το εμβαδόν ενός τριγώνου είναι να πολλαπλασιάσουμε την περίμετρο με την ακτίνα του κύκλου που μπορεί να δημιουργηθεί εντός του τριγώνου .Να πούμε ότι ο κύκλος αυτός πρέπει να τέμνει και στις τρεις πλευρές του τριγώνου .

Για να βρούμε την ακτίνα του κύκλου αυτού κάνουμε τα εξής βήματα :

Πρώτα παίρνουμε το κέντρο του κύκλου αυτού (STATS1(i).Centroid) .Λογω ότι τις συντεταγμένες αυτές θα τις χρησιμοποποιήσουμε ως δείκτη στον πίνακα onepixelsposition1 τις στρογγυλοποούμε (συνάρτηση round) .Ο κύκλος για να ακουμπάει και τις τρεις πλευρές του τριγώνου ,πρέπει το κέντρο του να βρίσκεται στο κέντρο του εμβαδού του τριγώνου και όχι στο κέντρο μιας πλευράς ενός τριγώνου .Το κέντρο το υπολογίζει η συνάρτηση regionprops (STATS1(i).Centroid) .Το πρόβλημα είναι ότι το κέντρο που υπολογίζει δεν είναι το κέντρο του εμβαδού .Άρα σε τρίγωνα που δεν είναι ισοσκελές δεν θα υπολογίζεται το σωστό εμβαδόν .

Έπειτα θα προσπαθήσουμε να υπολογίσουμε την ακτίνα που ξεκινάει από το κέντρο του κύκλου αυτού και τελειώνει στην μία από τις τρεις πλευρές του τριγώνου .Το σημείο αυτό που τέμνει ο κύκλος στην μία από τις τρεις αυτές πλευρές είναι το πιο κοντινό σημείο ως προς το κέντρο του κύκλου με τιμή 1 .Με βάση αυτό το κριτήριο μπορούμε να βρούμε την ακτίνα του κύκλου αυτού με τη βοήθεια του πίνακα distanceImage που επιστρέφει η συνάρτηση bwdist .Ο πίνακας αυτός περιέχει για κάθε pixel της εικόνας ,την απόσταση με το πιο κοντινό pixel σε αυτό με τιμή 1 .Η διαδικασία είναι η ίδια που κάναμε παραπάνω για να βρούμε τη μικρότερη απόσταση απο το κέντρο της τρύπας εώς το εσωτερικό σύνορο του σχήματος .

Παρακάτω όμως θα ακολουθήσουμε έναν άλλο τρόπο ,χρησιμοποιηώντας τον πίνακα onepixelsposition που επιστρέφει η συνάρτηση bwdist .Ο πίνακας onepixelsposition περιέχει για κάθε pixel της εικόνας το πιο κοντινό pixel σε αυτό με τιμή 1 .Το pixel που μας ενδιαφέρει είναι το pixel που αποτελεί το κέντρο του κύκλου ,του οποίου τις συντεταγμένες τις παίρνουμε από το αντικείμενο που επιστρέφει η συνάρτηση regionprops (STATS1.Centroid) .Προσοχή ,οι συντεταγμένες που επιστρέφει είναι αντιστραμμένες ,δηλαδή πρώτα επιστρέφει την y-συντεταγμένη και μετά την x-συντεταγμένη .Άρα μέσω του πίνακα onepixelsposition βρήσκουμε το πιο κοντινό pixel με τιμή 1 ως προς το κέντρο του κύκλου .

Επειδή όμως τα pixel της εικόνας που περιέχει ο πίνακας onepixelsposition αντιπροσωπεύονται με δείκτες ,από τον πίνακα onepixelsposition θα πάρουμε έναν αριθμό και όχι την διεύθυνσή του .Την διεύθυνσή του συγκεκριμένου pixel όμως μπορούμε να την βρούμε με την συνάρτηση ind2sub .

Τέλος την ακτίνα την υπολογίζουμε με την ευκλείδια μέθοδο .

```
%
[distanceImage1 ,onepixelsposition1] = bwdist(~RegionMask1);
%
```

Η συνάρτηση regionprops επιστρέφει τις συντεταγμένες του κέντρου ττου σχήματος που υπάρχει στο πλαίσιο ανάποδα ,δηλαδή πρώτα την y-συντεταγμένη και μετά την x-συντεταγμένη .

Επίσης οι συντεταγμένες που επιστρέφει αναφέρονται στην αρχική εικόνα και όχι του πλαισίου ,δηλαδή στην εικόνα που εντοπίστηκε το αντικείμενο (συνάρτηση bwboundaries) .

```
%
trregioncenter1 = round(STATS1(i).Centroid);
%
```

Αυτός ο τρόπος διαφέρει κυρίως από εδώ και κάτω .

```
%
closest_one_pixel_from_center1 =
onepixelsposition1(trregioncenter1(2),trregioncenter1(1));
regionimage1size = size(RegionMask1);
[x1_crd, y1_crd] = ind2sub(regionimage1size
,closest_one_pixel_from_center1);
%
x1distance = (x1_crd - trregioncenter1(2))^2;
y1distance = (y1_crd - trregioncenter1(1))^2;
%
traktina1a = sqrt(x1distance + y1distance);
tremvadon(n) = trperimetros(n)*traktina1a;
%
```

Η συνάρτηση regionprops υπολογίζει τη διάμετρο του σχήματος ,αλλά όπως θα δούμε παρακάτω ο κύκλος με την ακτίνα αυτήν δεν τέμνει τις πλευρές του τριγώνου .

```
%
traktina1b = STATS1(i).EquivDiameter/2;
%
circle_characteristics1a = [trregioncenter1 traktina1a];
circle_characteristics1b = [trregioncenter1 traktina1b];
%
```

Παρακάτω εμφανίζουμε τον κύκλο που τέμνει τις πλευρές ενός τριγώνου πάνω στην αρχική εικόνα padImageRead .Οι πληροφορίες που επιστρέφει η συνάρτηση regionprops ,όπως οι συντεταγμένες των σημείων του σχήματος ,αναφέρονται στην εικόνα που εντοπίστηκε το σχήμα ,στη συγκεκριμένη περίπτωση είναι η αρχική εικόνα ,και όχι στο πλαίσιο .

```
%
circle_in_triangle1a =
insertShape(padImageRead,'Circle',circle_characteristics1a,'Linewidth',3,...
    'Color','red');
figure;
```

```
                imshow(circle_in_triangle1a); title('circle in the triangle (1a)
in padImageRead')
                %
                circle_in_triangle1b =
insertShape(padImageRead,'Circle',circle_characteristics1b,'Linewidth',3,...
                    'Color','red');
                figure;
                imshow(circle_in_triangle1b); title('circle in the triangle (1b)
in padImageRead')
                %
                trids(n) = n;
                %
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση τρίγωνα) οπου καταλαμβάνουν στο αντικείμενο STATS1 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS1 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
                %
                showinimageindex(i) = 1;
                %
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως τρίγωνο .

```
                %
                % trin = 1
                %
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
                %
                [~,~,P] = size(padImageRead);
                isColorImage = P == 3;
                %
                if isColorImage
```

```
                    q=1;
                    for o = 1:RegionsPixels1size(1)
                        for p = 1:3
                            pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
                            q = q + 1;
                        end
                    end
                    regioncolor(n) = mean(pixelscolor);
                    %
                    if regioncolor(n) < 54
                        trcolortext(n) = {',black ,'};
                    end
                    if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                            (regioncolor(n) > 82 && regioncolor(n) < 86) ||...
                            (regioncolor(n) > 157 && regioncolor(n) < 158)
                        trcolortext(n) = {',blue ,'};
                    end
                    if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                            (regioncolor(n) > 112 && regioncolor(n) < 115)
                        trcolortext(n) = {',red'};
                    end
                    if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                            (regioncolor(n) > 78 && regioncolor(n) < 82)
                        trcolortext(n) = {',green ,'};
                    end
                    if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                            (regioncolor(n) > 139 && regioncolor(n) < 146) ||...
                            (regioncolor(n) > 160 && regioncolor(n) < 161) ||...
                            (regioncolor(n) > 168 && regioncolor(n) < 169)
                        trcolortext(n) = {',purple ,'};
                    end
                    if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                            (regioncolor(n) > 134 && regioncolor(n) < 135)
                        trcolortext(n) = {',orange ,'};
                    end
                    if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                            (regioncolor(n) > 123 && regioncolor(n) < 127) ||...
                            (regioncolor(n) > 129 && regioncolor(n) < 134) ||...
                            (regioncolor(n) > 135 && regioncolor(n) < 137) ||...
                            (regioncolor(n) > 145 && regioncolor(n) < 146)
                        trcolortext(n) = {',light blue ,'};
                    end
                    if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                            (regioncolor(n) > 167 && regioncolor(n) < 168) ||...
                            (regioncolor(n) > 171 && regioncolor(n) < 172)
                        trcolortext(n) = {',yellow ,'};
                    end
                    if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                            (regioncolor(n) > 185 && regioncolor(n) < 186)
                        trcolortext(n) = {',pink ,'};
                    end
                    if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
                            (regioncolor(n) > 154 && regioncolor(n) < 155)
                        trcolortext(n) = {',gray ,'};
                    end
                end
                n = n + 1;
            end
```

Παρακάτω ελέγχουμε για τετράγωνα ,στην περίπτωση που αυτά δεν έχουν τρύπα εσωτερικά .Οι

προυποθέσεις ανίχνευσης των τετραγώνων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές .Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
if STATS1(i).Eccentricity > 0.18 && STATS1(i).Eccentricity < 0.98...
        && STATS1(i).Extent > 0.95 && STATS1(i).Extent <= 1 ...
        && STATS1(i).Solidity > 0.8 && STATS1(i).Solidity <= 1
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα χωρίς τρύπα στο εσωτερικό του .

```
%
STATS1in = 1;
%
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,χωρίς τρύπα εσωτερικά ,που έχει αναγνωριστεί ως τετράγωνο .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως τετράγωνο .

```
%
ecrec1 = STATS1(i).Eccentricity
exrec1 = STATS1(i).Extent
solrec1 = STATS1(i).Solidity
arearec1 = STATS1(i).Area
%
```

Παρακάτω αποθηκεύουμε το κέντρο του συγκεκριμένου γεωμετρικού σχήματος που υπάρχει μέσα σε αυτό το πλαίσιο ,στον πίνακα shapescenter .Η χρησιμότητα του πίνακα αυτού είναι να μην αναγνωρίστεί το ίδιο σχήμα δύο ή περισσότερες φορές .Ο έλεγχος για τυχόν διπλή αναγνώριση του γεωμετρικού σχήματος αυτού γίνεται στις περιπτώσεις που το γεωμετρικό σχήμα έχει τρύπα εσωτερικά .Το πρόβλημα δεν βρίσκεται στην περίπτωση το ίδιο αντικείμενο (STATS1 ή STATS2) να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα παραπάνω από μία φορά ,αλλά ένα άλλο αντικείμενο που επιστρέφει η συνάρτηση regionprops ,να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα με ένα άλλο αντικείμενο που έχει δώσει η συνάρτηση regionprops σε άλλη περίπτωση .

```
%
centerrec1 = STATS1(i).Centroid;
shapescenter(n,:) = vertcat(centerrec1);
%
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
%
figure;
imshow(regionimage1),title(n)
%
figure;
imshow(RegionMask1); title('RegionMask1')
%
figure;
imshow(displayImage); title('RegionImage in the original image')
%
```

Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις recids για τα τετράγωνα ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που

εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η recids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε (στη συγκεκριμένη περίπτωση τετράγωνα) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (recids κ.τ.λ.) μπορεί να διαφέρει .

```
%
boxes(n,:) = STATS1(i).BoundingBox;
ids(n) = n;
%
recperimetros(n) = STATS1(i).Perimeter;
%
```

Παρακάτω θα υπολογιστεί το εμβαδόν του τετραγώνου .Το εμβαδόν του τετραγώνου θα υπολογιστεί με βάση τις διαστασεις του ορθογωνίκού πλαισίου (BoundingBox) που περιέχει το γεωμετρικό σχήμα που είναι ορθογώνιο .Ακολουθείται αυτός ο τρόπος γιατί τόσο το ορθογώνιο πλαίσιο ,όσο και το σχήμα (ορθογώνιο) ,θα έχουν τις ίδιες διαστάσεις .

```
%
recbboxes(n,:) = STATS1(i).BoundingBox;
basi1 = recbboxes(n,3);
ypsos1 = recbboxes(n,4);
recemvadon(n) = ypsos1 * basi1;
%
recids(n) = n;
%
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση τετράγωνα) οπου καταλαμβάνουν στο αντικείμενο STATS1 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS1 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως τετράγωνο .

```
%
% recin = 1
%
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
                %
                [~,~,P] = size(padImageRead);
                isColorImage = P == 3;
                %
                if isColorImage
                    q=1;
                    for o = 1:RegionsPixels1size(1)
                        for p = 1:3
                            pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
                            q = q + 1;
                        end
                    end
                    regioncolor(n) = mean(pixelscolor)
                    %
                    if regioncolor(n) < 54
                        reccolortext(n) = {',black ,'};
                    end
                    if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                            (regioncolor(n) > 82 && regioncolor(n) < 86) ||...
                            (regioncolor(n) > 157 && regioncolor(n) < 158)
                        reccolortext(n) = {',blue ,'};
                    end
                    if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                            (regioncolor(n) > 112 && regioncolor(n) < 115)
                        reccolortext(n) = {',red'};
                    end
                    if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                            (regioncolor(n) > 78 && regioncolor(n) < 82)
                        reccolortext(n) = {',green ,'};
                    end
                    if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                            (regioncolor(n) > 139 && regioncolor(n) < 146) ||...
                            (regioncolor(n) > 160 && regioncolor(n) < 161) ||...
                            (regioncolor(n) > 168 && regioncolor(n) < 169)
                        reccolortext(n) = {',purple ,'};
                    end
                    if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                            (regioncolor(n) > 134 && regioncolor(n) < 135)
                        reccolortext(n) = {',orange ,'};
                    end
                    if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                            (regioncolor(n) > 123 && regioncolor(n) < 127) ||...
                            (regioncolor(n) > 129 && regioncolor(n) < 134) ||...
                            (regioncolor(n) > 135 && regioncolor(n) < 137) ||...
                            (regioncolor(n) > 145 && regioncolor(n) < 146)
                        reccolortext(n) = {',light blue ,'};
                    end
```

```
            if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                    (regioncolor(n) > 167 && regioncolor(n) < 168) ||...
                    (regioncolor(n) > 171 && regioncolor(n) < 172)
                reccolortext(n) = {',yellow ,'};
            end
            if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                    (regioncolor(n) > 185 && regioncolor(n) < 186)
                reccolortext(n) = {',pink ,'};
            end
            if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
                    (regioncolor(n) > 154 && regioncolor(n) < 155)
                reccolortext(n) = {',gray ,'};
            end
        end
        n = n + 1;
    end
```

Παρακάτω ελέγχουμε για πολύγωνα ,στην περίπτωση που αυτά δεν έχουν τρύπα εσωτερικά .Οι προυποθέσεις ανίχνευσης των πολυγώνων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές . Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
if STATS1(i).Eccentricity > 0.04 && STATS1(i).Eccentricity < 0.14...
        && STATS1(i).Extent > 0.7 && STATS1(i).Extent < 0.78...
        && STATS1(i).Solidity > 0.97 && STATS1(i).Solidity < 0.985
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα χωρίς τρύπα στο εσωτερικό του .

```
%
STATS1in = 1;
%
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,χωρίς τρύπα εσωτερικά ,που έχει αναγνωριστεί ως πολύγωνο .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως πολύγωνο .

```
%
ecpol1 = STATS1(i).Eccentricity
expol1 = STATS1(i).Extent
solpol1 = STATS1(i).Solidity
areapol1 = STATS1(i).Area
%
```

Παρακάτω αποθηκεύουμε το κέντρο του συγκεκριμένου γεωμετρικού σχήματος που υπάρχει μέσα σε αυτό το πλαίσιο ,στον πίνακα shapescenter .Η χρησιμότητα του πίνακα αυτού είναι να μην αναγνωρίστεί το ίδιο σχήμα δύο ή περισσότερες φορές .Ο έλεγχος για τυχόν διπλή αναγνώριση του γεωμετρικού σχήματος αυτού γίνεται στις περιπτώσεις που το γεωμετρικό σχήμα έχει τρύπα εσωτερικά .Το πρόβλημα δεν βρίσκεται στην περίπτωση το ίδιο αντικείμενο (STATS1 ή STATS2) να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα παραπάνω από μία φορά ,αλλά ένα άλλο αντικείμενο που επιστρέφει η συνάρτηση regionprops ,να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα με ένα άλλο αντικείμενο που έχει δώσει η συνάρτηση regionprops σε άλλη περίπτωση .

```
%
centerpol1 = STATS1(i).Centroid;
shapescenter(n,:) = vertcat(centerpol1);
%
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
%
figure;
imshow(regionimage1),title(n)
%
figure;
imshow(RegionMask1); title('RegionMask1')
%
figure;
imshow(displayImage); title('RegionImage in the original image')
%
```

Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις polids για τα πολύγωνα ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η polids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε (στη συγκεκριμένη περίπτωση πολύγωνα) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (polids κ.τ.λ.) μπορεί να διαφέρει .

```
%
boxes(n,:) = STATS1(i).BoundingBox;
ids(n) = n;
%
polbboxes(n,:) = STATS1(i).BoundingBox;
polperimetros(n) = STATS1(i).Perimeter;
polids(n) = n;
%
```

Την παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση πολύγωνα) οπου καταλαμβάνουν στο αντικείμενο STATS1 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS1 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως πολύγωνο .

```
%
% polin = 1
%
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές

καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
                %
                [~,~,P] = size(padImageRead);
                isColorImage = P == 3;
                %
                if isColorImage
                    q=1;
                    for o = 1:RegionsPixels1size(1)
                        for p = 1:3
                            pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
                            q = q + 1;
                        end
                    end
                    regioncolor(n) = mean(pixelscolor)
                    %
                    if regioncolor(n) < 54
                        polcolortext(n) = {',black ,'};
                    end
                    if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                            (regioncolor(n) > 82 && regioncolor(n) < 86) ||...
                            (regioncolor(n) > 157 && regioncolor(n) < 158)
                        polcolortext(n) = {',blue ,'};
                    end
                    if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                            (regioncolor(n) > 112 && regioncolor(n) < 115)
                        polcolortext(n) = {',red'};
                    end
                    if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                            (regioncolor(n) > 78 && regioncolor(n) < 82)
                        polcolortext(n) = {',green ,'};
                    end
                    if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                            (regioncolor(n) > 139 && regioncolor(n) < 146) ||...
                            (regioncolor(n) > 160 && regioncolor(n) < 161) ||...
                            (regioncolor(n) > 168 && regioncolor(n) > 169)
                        polcolortext(n) = {',purple ,'};
                    end
                    if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                            (regioncolor(n) > 134 && regioncolor(n) < 135)
                        polcolortext(n) = {',orange ,'};
                    end
```

```
            if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                    (regioncolor(n) > 123 && regioncolor(n) < 127) ||...
                    (regioncolor(n) > 129 && regioncolor(n) < 134) ||...
                    (regioncolor(n) > 135 && regioncolor(n) < 137) ||...
                    (regioncolor(n) > 145 && regioncolor(n) < 146)
                polcolortext(n) = {',light blue ,'};
            end
            if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                    (regioncolor(n) > 167 && regioncolor(n) < 168) ||...
                    (regioncolor(n) > 171 && regioncolor(n) < 172)
                polcolortext(n) = {',yellow ,'};
            end
            if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                    (regioncolor(n) > 185 && regioncolor(n) < 186)
                polcolortext(n) = {',pink ,'};
            end
            if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
                    (regioncolor(n) > 154 && regioncolor(n) < 155)
                polcolortext(n) = {',gray ,'};
            end
        end
        n = n + 1;
    end
```

Παρακάτω ελέγχουμε για κύκλους ,στην περίπτωση που αυτά δεν έχουν τρύπα εσωτερικά .Οι προυποθέσεις ανίχνευσης των κύκλων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές . Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
if STATS1(i).Eccentricity < 0.2 && STATS1(i).Extent > 0.78 &&...
        STATS1(i).Extent < 79 && STATS1(i).Solidity > 0.993 &&...
        STATS1(i).Solidity < 0.996
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα χωρίς τρύπα στο εσωτερικό του .

```
%
STATS1in = 1;
%
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,χωρίς τρύπα εσωτερικά ,που έχει αναγνωριστεί ως κύκλος .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως κύκλος .

```
%
eccir1 = STATS1(i).Eccentricity
excir1 = STATS1(i).Extent
solcir1 = STATS1(i).Solidity
areacir1 = STATS1(i).Area
%
```

Παρακάτω αποθηκεύουμε το κέντρο του συγκεκριμένου γεωμετρικού σχήματος που υπάρχει μέσα σε αυτό το πλαίσιο ,στον πίνακα shapescenter .Η χρησιμότητα του πίνακα αυτού είναι να μην αναγνωρίστεί το ίδιο σχήμα δύο ή περισσότερες φορές .Ο έλεγχος για τυχόν διπλή αναγνώριση του γεωμετρικού σχήματος αυτού γίνεται στις περιπτώσεις που το γεωμετρικό σχήμα έχει τρύπα εσωτερικά .Το πρόβλημα δεν βρίσκεται στην περίπτωση το ίδιο αντικείμενο (STATS1 ή STATS2) να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα παραπάνω από μία φορά ,αλλά ένα άλλο αντικείμενο που επιστρέφει η συνάρτηση regionprops ,να έχει 'αναλύσει' το ίδιο γεωμετρικό σχήμα με ένα άλλο αντικείμενο που έχει δώσει η συνάρτηση regionprops σε άλλη περίπτωση .

```
%
centercir1 = STATS1(i).Centroid;
shapescenter(n,:) = vertcat(centercir1);
%
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
%
figure;
imshow(regionimage1),title(n)
%
figure;
imshow(RegionMask1); title('RegionMask1')
%
figure;
imshow(displayImage); title('RegionImage in the original image')
%
```

Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις cirids για τους κύκλους ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η cirids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε (στη συγκεκριμένη περίπτωση κύκλοι) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (cirids κ.τ.λ.) μπορεί να διαφέρει .

```
%
boxes(n,:) = STATS1(i).BoundingBox;
ids(n) = n;
%
cirperimetros(n) = STATS1(i).Perimeter;
%
```

Παρακάτω υπολογίζεται το εμβαδόν του κύκλου που εντοπίστηκε .Για αρχή παίρνουμε τον κύριο (MajorAxisLength) και τον δευτερεύον (MinorAxisLength) άξονα κάθε έλλειψης που περιβάλει τα σημεία του σχήματος .Μια έλλειψη που τείνει να γίνει κύκλος ,η εκκεντικότητα τείνει στο 0 και τότε ο κύριος και ο δευτερεύον άξονας της έλλειψης τείνουν να έχουν την ίδια τιμή .Τότε οι άξονες της έλλειψης αποτελούν τη διάμετρο ενός κύκλου .Επειδή λόγω θορύβου μπορεί να μην έχουμε ακριβώς κύκλο θα υπολογίσουμε τη μέση τιμή του κύριου και του δευτερεύων άξονα (συνάρτηση mean) ώστε έπειτα να υπολογίσουμε την ακτίνα του κύκλου .

Παρόλο που η μεταβλητή diameters είναι ένα διάνυσμα ,πρέπει να προσθέσουμε μία διάσταση ((i,:)) ,ώστε να έχει τις ίδιες διαστάσεις με τον πίνακα centers ,ώστε έπειτα να τους συγχωνεύσουμε και να δημιουργήσουμε τον πίνακα circle .Ο πίνακας circle θα χρησιμοποιηθεί για να τονιστεί ο κύκλος που περιέχει το πλαίσιο αυτό στην αρχική εικόνα .

```
%
diameters1(n,:) = mean([STATS1(i).MajorAxisLength
STATS1(i).MinorAxisLength],2);
radii1(n) = diameters1(n,:)./2;
```

```
ciremvadon(n) = 3.14*(radii1(n))^2;
%
```

Για να τονιστούν οι κύκλοι που εντοπίστηκαν και αναλύθηκαν από τη συνάρτηση regionprops πάνω στην αρχική εικόνα ,χρειάζονται τα χαρακτηριστικά των κύκλων ,δηλαδή το κέντρο και η ακτίνα του κύκλου .Παρακάτω παίρνουμε τον κύκλο του αντικειμένου STATS1 ,όπου περιέχει επακριβώς το γεωμετρικό σχήμα όπως είναι στην αρχική εικόνα .

Η συνάρτηση regionprops επιστρέφει τις συντεταγμένες του κέντρου του σχήματος ανάποδα ,δηλαδή επιστρέφει πρώτα την y-συντεταγμένη και μετά την x-συντεταγμένη . Επίσης οι συντεταγμένες που επιστρέφει αναφέρονται στην αρχική εικόνα και όχι στο πλαίσιο ,δηλαδή στην εικόνα που εντοπίστηκε το σχήμα (συνάρτηση bwboundaries) .

```
%
centers1(n,:) = STATS1(i).Centroid;
circle = [centers1(n,:) radii1(n)];
%
cirids(n) = n;
%
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση κύκλοι) οπου καταλαμβάνουν στο αντικείμενο STATS1 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS1 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως κύκλος .

```
%
% cirin = 1
%
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
                %
                [~,~,P] = size(padImageRead);
                isColorImage = P == 3;
                %
                if isColorImage
                    q = 1;
                    for o = 1:RegionsPixels1size(1)
                        for p = 1:3
                            pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
                            q = q + 1;
                        end
                    end
                    regioncolor(n) = mean(pixelscolor)
                    %
                    if regioncolor(n) < 54
                        circolortext(n) = {',black ,'};
                    end
                    if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                            (regioncolor(n) > 82 && regioncolor(n) < 86) ||...
                            (regioncolor(n) > 157 && regioncolor(n) < 158)
                        circolortext(n) = {',blue ,'};
                    end
                    if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                            (regioncolor(n) > 112 && regioncolor(n) < 115)
                        circolortext(n) = {',red'};
                    end
                    if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                            (regioncolor(n) > 78 && regioncolor(n) < 82)
                        circolortext(n) = {',green ,'};
                    end
                    if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                            (regioncolor(n) > 139 && regioncolor(n) < 146) ||...
                            (regioncolor(n) > 160 && regioncolor(n) < 161) ||...
                            (regioncolor(n) > 168 && regioncolor(n) < 169)
                        circolortext(n) = {',purple ,'};
                    end
                    if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                            (regioncolor(n) > 134 && regioncolor(n) < 135)
                        circolortext(n) = {',orange ,'};
                    end
                    if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                            (regioncolor(n) > 123 && regioncolor(n) < 127) ||...
                            (regioncolor(n) > 129 && regioncolor(n) < 134) ||...
                            (regioncolor(n) > 135 && regioncolor(n) < 137) ||...
                            (regioncolor(n) > 145 && regioncolor(n) < 146)
                        circolortext(n) = {',light blue ,'};
                    end
                    if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                            (regioncolor(n) > 167 && regioncolor(n) < 168) ||...
                            (regioncolor(n) > 171 && regioncolor(n) < 172)
                        circolortext(n) = {',yellow ,'};
                    end
                    if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                            (regioncolor(n) > 185 && regioncolor(n) < 186)
                        circolortext(n) = {',pink ,'};
                    end
                    if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
```

```
                    (regioncolor(n) > 154 && regioncolor(n) < 155)
                circolortext(n) = {',gray ,'};
            end
        end
        n = n + 1;
    end
end
```

Παρακάτω ελέγχουμε για γραμμές .Οι προυποθέσεις ανίχνευσης των γραμμών (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές . Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .Επειδή στις γραμμές δεν υπάρχει κενό στο εσωτερικό τους ,αλλιώς θα είχαμε ένα άλλο σχήμα , δεν θα γίνεται ανίχνευση για τυχόν κενό στο εωτερικό της γραμμής .

Πέρα από τον εντοπισμό των γραμμών με βάση τα χρακτηριστικά των πλαισίων του αντικειμένου STATS1 ,παρακάτω θα εφαρμοστεί και μια άλλη μέθοδο για τον εντοπισμό των γραμμών .Η μέθοδος εντοπισμού γραμμών με hough transform .Οι γραμμές που μπορούν να εντοπιστούν από την μέθοδο με hough transform μπορεί να είναι περισσότερες εάν εφαρμοστεί ο hough transform σε περισσότερα πλαίσια του αντικειμένου STATS1 ή και στην αρχική εικόνα απευθείας .

Εφαρμόζοντας και τις δύο μεθόδους ,μπορεί να γίνει αντιληπτό ότι η μέθοδος h.t. εντοπίζει τις γραμμές που εντοπίστηκαν στο αντικείμενο STATS1 με βάση τα χαρακτηριστικά των περιοχών τους (σελ. 75-76) .

```
if STATS1(i).Area > 4
    if STATS1(i).Eccentricity > 0.87 && STATS1(i).Extent > 0.5 && ...
            STATS1(i).Solidity > 0.79
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα με τρύπα στο εσωτερικό του .

```
%
STATS1in = 1;
%
```

Προσοχή στις συντεταγμένες που επιστρέφει η συνάρτηση regionprops .Όπως παρατηρουμε οι συντεταγμένες που επιστρέφει η συνάρηση regionprops είναι αντιστραμένες ,δηλαδή πρώτα επιστρέφει την y-συντεταγμένη και μετά την x-συντεταγμένη .

```
%
lineRegionsPixels(r).pixels = RegionsPixels1;
r = r + 1;
%
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση γραμμές) οπου καταλαμβάνουν στο αντικείμενο STATS1 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS1 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως γραμμή .

```
%
% linein = 1
%
```

Παρακάτω θα δείξουμε έναν αποκλειστικό τρόπο εντοπισμού των γραμμών που υπάρχουν στην εικόνα ,χρησιμοποιώντας hough transform .Ο hough transform ξεχωρίζει τις γραμμές ακόμα και όταν αυτές είναι ενωμένες ,εναντιθέση με την συνάρτηση bwboundaries .Δηλαδή σε ένα τετράγωνο η συνάρτηση bwboundaries δε θα ανιχνεύσει τέσσερις ξεχωριστές γραμμές ,όσες δηλαδή και οι πλευρές του .Βέβαια ούτε και με hough transform είναι σίγουρο ότι θα πάρουμε τις τέσσερις πλευρές του τετραγώνου .

Παρακάτω εντοπίζουμε τις γραμμές από το σχήμα που απεικονίζει κάθε πλαίσιο του αντικειμένου STATS1 με τα συγκεκριμένα χαρακτηριστικά ,και ύστερα θα τις απεικονίσουμε με τη συνάρτηση plot .

Αυτό δε σημαίνει ότι δε μπορούμε να εφαρμόσουμε απευθείας τον παρακάτω τρόπο στην αρχική εικόνα ,για τον εντοπισμό των γραμμών που περιέχει η αρχική εικόνα .Απλά σε αυτήν την περίπτωση πρέπει να ξεχωρίσουμε τις γραμμές από την εικόνα πρώτα .Η ανίχνευση των γραμμών σε μια εικονα γίνεται με τη συνάρτηση edge .Δηλαδή ,όπως χρησιμοποιήθηκε η συνάρτηση bwboundaries για τον εντοπισμό των συνόρων των αντικειμένων της εικόνας , θα πρέπει να χρησιμοποιηθεί η συνάρτηση edge για τον εντοπισμό των γραμμών που υπάρχουν στην εικόνα .Άρα πριν από τις παρακάτω εντολές θα χρησιμοποιούσαμε την συνάρτηση edge .

Με τον hough transform είναι δυνατόν να βρεθούν τα άκρα των γραμμών (αρχικο και τελικό σημείο) που εντοπίστηκαν στην εικόνα . Γνωρίζοντας τα άκρα μιας γραμμή μπορούμε να βρούμε και το μήκος μιας γραμμής ή ακόμα κα την εξίσωσή της .

Η συνάρτηση hough δημιουργεί γραμμές που ενώνουν διάφορα σημεία της εικόνας .Τα σημεία αυτά της εικόνας τα μετασχηματίζει σε καμπύλες ,δημιουργώντας το hough επίπεδο (theta,r) .Τις παραμέτρους theta και r τις επιστρέφει η συνάρτηση hough .

```
[H,theta,rho] = hough(RegionMask1);
```

Μέσω του επιπέδου hough ,με βάση το σημείο που 'συναντάται' πιο πολύ ,βρίσκονται τυχόν γραμμές που υπάρχουν στην εικόνα .Αυτή την λειτουργία εκτελεί η συνάρτηση houghpeaks .

```
P = houghpeaks(H,5,'threshold',ceil(0.3*max(H(:))));
```

Η παρακάτω συνάρτηση (houghlines) επιστρέφει εν τέλει τα χαρακτηριστικά των γραμμών που εντοπίστηκαν στην δυαδική εικόνα . Η συνάρτηση houghlines επιστρέφει μια διάταξη από δομές ,όπου η κάθε δομή περιέχει κάποιες πληροφορίες για κάθε γραμμή .Οι πληροφορίες που περιέχει είναι :

point1 : Είναι ένα διάνυσμα 2 στοιχείων που περιέχει τις συντεταγμένες του ενός από τα 2 ακριανά σημεία της γραμμής .
Point2 : Είναι ένα διάνυσμα 2 στοιχείων που περιέχει τις συντεταγμένες του άλλου από τα 2 ακριανά σημεία της γραμμής .
Theta : Περιέχει τη γωνία που υπολογίστηκε από τη συνάρτηση hough .
Rho : Περιέχει το διάνυσμα rho που υπολογίστηκε από τη συνάρτηση hough .

```
                lines =
houghlines(RegionMask1,theta,rho,P,'FillGap',1,'MinLength',7);

                if ~isempty(lines)
                    figure,
                    imshow(padImageRead),title('hough transform') ,hold on
                    max_len = 0;
                    for s = 1:length(lines)

                        % Σχεδιασμός των γραμμών

                        xy = [lines(s).point1; lines(s).point2];
                        plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');

                        % Σχεδιασμός των άκρων των γραμμών .


plot(xy(1,1),xy(1,2),'x','LineWidth',2,'Color','yellow');
                        plot(xy(2,1),xy(2,2),'x','LineWidth',2,'Color','red');

                        % Εύρεση των συντεταγμένων της μεγαλύτερης γραμμής .

                        len = norm(lines(s).point1 - lines(s).point2);
                        if ( len > max_len)
                            max_len = len;
                            xy_long = xy;
                        end
```

Άμα θέλαμε να βρούμε το μήκος κάθε γραμμής τότε κάνουμε τις εξής πράξεις .Χρήσιμος τρόπος σε περιπτώσεις που είναι αναγκαίο να βρεθεί το μήκος μιας γραμμής .

```
                        %
                        % xdistance = (lines(s).point1(1) -
lines(s).point1(2))^2;
                        % ydistance = (lines(s).point2(1) -
lines(s).point2(2))^2;
                        % linelength(t) = sqrt((xdistance+ydistance));
                    end
                    % Εμφάνιση της μεγαλύτερης γραμμής .
                    plot(xy_long(:,1),xy_long(:,2),'LineWidth',2,'Color','red');
                end
            end
        end
    end
```

Στην περίπτωση που το γεωμετρικό σχήμα έχει κενό στο εσωτερικό του ,τότε παραπάνω δημιουργείται ένα δεύτερο σχήμα ,ίδιο με το αρχικό αλλά με γεμάτο το κενό του στο εσωτερικό .

Σε αυτήν την περίπτωση για να αναλυθεί σωστά το σχήμα ,πρέπει να χρησιμοποιήσουμε τις ιδιότητες ((που επιστρέφει η συνάρτηση regionprops) που αναφέρονται στο δεύτερο σχήμα και όχι στο πρώτο .Αυτές οι ιδιότητες χρησιμοποιούνται παρακάτω .

```
    if holeindex
        holeindex = false;
        for j = 1 : length(STATS2)
            sameshape = 0;
            %
```

Οι παρακάτω μεταβλητές φανερώνουν τις τιμές Eccentricity ,Extent ,Solidity και Area των

πλαισίων που περιέχει το αντικείμενο STATS2 .Χρησιμη πληροφορία στην περίπτωση που θέλουμε να δούμε τα χαρακτηριστικά του σχήματος ,ενός πλαισίου .

```
%
ec2 = STATS2(j).Eccentricity
ex2 = STATS2(j).Extent
sol2 = STATS2(j).Solidity
area2 = STATS2(j).Area
%
```

Παρακάτω ελέγχουμε για τρίγωνα ,στην περίπτωση που αυτά έχουν τρύπα εσωτερικά .Οι προυποθέσεις ανίχνευσης των τριγώνων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές .Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
if STATS2(j).Area > 1000
    if STATS2(j).Eccentricity > 0.11 && STATS2(j).Eccentricity <
0.82 &&...
            STATS2(j).Extent > 0.33 && STATS2(j).Extent < 0.53 &&...
            STATS2(j).Solidity > 0.97 && STATS2(j).Solidity <= 1
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,με τρύπα εσωτερικά ,που έχει αναγνωριστεί ως τρίγωνο .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως τρίγωνο .

```
%
ectr2 = STATS2(j).Eccentricity
extr2 = STATS2(j).Extent
soltr2 = STATS2(j).Solidity
areatr2 = STATS2(j).Area
%
```

Παρακάτω ελέγχουμε εάν το γεωμετρικό σχήμα που υπάρχει σε αυτό το πλαίσιο του αντικειμένου STATS2 ,υπάρχει και σε κάποιο πλαίσιο του αντικειμένου STATS1 .Τα γεωμετρικά σχήματα του αντικειμένου STATS1 τα αναγνωρίζουμε από το κέντρο τους ,το οποίο περιέχεται στον πίνακα (shapescenter) . Εάν το γεωμετρικό σχήμα υπάρχει σε κάποιο πλαίσιο του αντικειμένου STATS1 τότε αποφεύγουμε την περαιτέρω ανάλυσή του .

```
%
centertr2 = STATS1(i).Centroid;
currentcenter = centertr2;
shapescentersize = size(shapescenter);
%
for u=1:shapescentersize(1)
    if (currentcenter(1) / shapescenter(u,1) > 0.9) &&...
            (currentcenter(1) / shapescenter(u,1) < 1.1)
        for v=1:shapescentersize(1)
            if (currentcenter(2) / shapescenter(v,2) > 0.9)
&&...
                    (currentcenter(2) / shapescenter(u,2) <
1.1)
                sameshape = 1;
                break
            end
        end
    end
end
if sameshape == 1
    continue
```

```
end
shapescenter(n,:) = vertcat(centertr2);
%
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα με τρύπα στο εσωτερικό του .

```
%
STATS2in = 1;
%
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
%
figure;
imshow(regionimage1),title(n)
%
figure;
imshow(RegionMask1); title('RegionMask1')
%
figure;
imshow(displayImage); title('RegionImage in the original
image')
%
```

Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις trids για τα τρίγωνα ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η trids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντπίστηκε (στη συγκεκριμένη περίπτωση τρίγωνα) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (trids κ.τ.λ.) μπορεί να διαφέρει .

```
%
```

Παρακάτω παρατηρείται επίσης ότι ενώ παίρνουμε τις θέσεις του ορθογώνιου πλαισίου που περιβάλλει το γεωμετρικό σχήμα από το αντικείμενο STATS1 (STATS1(i).BoundingBox) ,δεν γίνεται το ίδιο και με την περίμετρο (STATS2(j).Perimeter) ,την οποία την παίρνουμε από το αντικείμενο STATS2 .Ο λόγος είναι ότι το αντικείμενο STATS1 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στην αρχική εικόνα ,ενώ το αντικείμενο STATS2 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στο πλαίσιο του αντικειμένου STATS1 .Δηλαδή το ίδιο γεωμετρικό σχήμα έχει εντοπιστεί σε άλλες εικόνες (με τη συνάρτηση bwboundaries) και άρα αυτό βρίσκεται σε άλλες συντεταγμένες .Η κλίμακα όμως δεν έχει αλλάξει στο γεωμετρικό σχήμα και για αυτο την περίμετρο μπορούμε να την πάρουμε από και τα δύο αντικείμενα STATS1 και STATS2 .

```
%
boxes(n,:) = STATS1(i).BoundingBox;
ids(n) = n;
%
trianbboxes(n,:) = STATS1(i).BoundingBox;
trperimetros(n) = STATS2(j).Perimeter;
```

%

Παρακάτω υπολογίζεται το εμβαδόν του τριγώνου που υπάρχει εντός του πλαισίου αυτού .Ένας τρόπος για να βρούμε το εμβαδόν ενός τριγώνου είναι να πολλαπλασιάσουμε την περίμετρο με την ακτίνα του κύκλου που μπορεί να δημιουργηθεί εντός το τριγώνου .Να πούμε ότι ο κύκλος αυτός πρέπει να τέμνει και στις τρεις πλευρές του τριγώνου .

Όπως αναφέρθηκε ,στην περίπτωση που το γεωμετρικό σχήμα έχει τρύπα εσωτερικά πρέπει να βρεθεί και το 'εξωτερικό' εμβαδόν (δηλαδή το εμβαδόν του σχήματος που δημιουργείται από το εξωτερικό σύνορό του και μέσα) και το εσωτερικό εμβαδόν (δηλαδή το εμβαδόν του σχήματος που δημιουργείται από το εσωτερικό σύνορό του και μέσα) ,ώστε αφαιρώντας τα να βρούμε το εμβαδόν του γεωμετρικού σχήματος αυτού συμπεριλαμβανομένου και την τρύπα που αυτό έχει .

Την ακτίνα για τον υπολογισμό του 'έσωτερικού' εμβαδού την έχουμε υπολογίσει παραπάνω (traktina4max) .Για την εύρεση της ακτίνας που θα χρειαστεί για τον υπολογισμό του εξωτερικού εμβαδού χρειάζεται να δημιουργήσουμε μια εικόνα με το εξωτερικο σύνορο του γεωμετρικού σχήματος .Για αυτό δε χρειάζεται να γίνει η ίδια λειτουργία που έγινε για την εύρεση του εσωτερικού συνόρου γιατί το εξωτερικο σύνορο του γεωμετρικού σχήματος περιέχεται (και με τις ίδιες διαστάσεις) στο πλαίσιο του αντικειμένου STATS2 .Άρα χρησιμοποιώντας το πλαίσιο που επιστρέφει το αντικείμενο STATS2 ,με τον ίδιο τρόπο θα βρούμε την ακτίνα του κύκλου που τέμνει και τις τρεις πλευρές του τριγώνου που δημιουργείται με το εξωτερικο σύνορο .

Για να βρούμε την ακτίνα του κύκλου αυτού κάνουμε τα εξής βήματα :

Πρώτα παίρνουμε το κέντρο του κύκλου αυτού (STATS1(i).Centroid) .Λογω ότι τις συντεταγμένες αυτές θα τις χρησιμοπποιήσουμε ως δείκτη στον πίνακα onepixelsposition1 τις στρογγυλοποούμε (συνάρτηση round) .Ο κύκλος για να ακουμπάει και τις τρεις πλευρές του τριγώνου ,πρέπει το κέντρο του να βρίσκεται στο κέντρο του εμβαδού του τριγώνου και όχι στο κέντρο μιας πλευράς ενός τριγώνου .Το κέντρο το υπολογίζει η συνάρτηση regionprops (STATS1(i).Centroid) .Το πρόβλημα είναι ότι το κέντρο που υπολογίζει δεν είναι το κέντρο του εμβαδού .Άρα σε τρίγωνα που δεν είναι ισοσκελές δεν θα υπολογίζεται το σωστό εμβαδόν .

Έπειτα θα προσπαθήσουμε να υπολογίσουμε την ακτίνα που ξεκινάει από το κέντρο του κύκλου αυτού και τελειώνει στην μία από τις τρεις πλευρές του τριγώνου .Το σημείο αυτό που τέμνει ο κύκλος στην μία από τις τρεις αυτές πλευρές είναι το πιο κοντινό σημείο ως προς το κέντρο του κύκλου με τιμή 1 .Με βάση αυτό το κριτήριο μπορούμε να βρούμε την ακτίνα του κύκλου αυτού με τη βοήθεια του πίνακα distanceImage που επιστρέφει η συνάρτηση bwdist .Ο πίνακας αυτός περιέχει για κάθε pixel της εικόνας ,την απόσταση με το πιο κοντινό pixel σε αυτό με τιμή 1 .Η διαδικασία είναι η ίδια που κάναμε παραπάνω για να βρούμε τη μικρότερη απόσταση απο το κέντρο της τρύπας εώς το εσωτερικό σύνορο του σχήματος .

Παρακάτω όμως θα ακολουθήσουμε έναν άλλο τρόπο ,χρησιμοποιηώντας τον πίνακα onepixelsposition που επιστρέφει η συνάρτηση bwdist .Ο πίνακας onepixelsposition περιέχει για κάθε pixel της εικόνας το πιο κοντινό pixel σε αυτό με τιμή 1 .Το pixel που μας ενδιαφέρει είναι το pixel που αποτελεί το κέντρο του κύκλου ,του οποίου τις συντεταγμένες τις παίρνουμε από το αντικείμενο που επιστρέφει η συνάρτηση regionprops (STATS1.Centroid) .Προσοχή ,οι συντεταγμένες που επιστρέφει είναι αντιστραμμένες ,δηλαδή πρώτα επιστρέφει την y-συντεταγμένη και μετά την x-συντεταγμένη .Άρα μέσω του πίνακα onepixelsposition βρήσκουμε το πιο κοντινό pixel με τιμή 1 ως προς το κέντρο του κύκλου .

Επειδή όμως τα pixel της εικόνας που περιέχει ο πίνακας onepixelsposition αντιπροσωπεύονται με

δείκτες ,από τον πίνακα onepixelsposition θα πάρουμε έναν αριθμό και όχι την διεύθυνσή του .Την διεύθυνσή του συγκεκριμένου pixel όμως μπορούμε να την βρούμε με την συνάρτηση ind2sub .

Τέλος την ακτίνα την υπολογίζουμε με την ευκλείδια μέθοδο .

```
                    %
                    [distanceImage2 ,onepixelsposition2] =
bwdist(regionimage2a);
                    %
```

Η συνάρτηση regionprops επιστρέφει τις συντεταγμένες του κέντρου ττου σχήματος που υπάρχει στο πλαίσιο ανάποδα ,δηλαδή πρώτα την y-συντεταγμένη και μετά την x-συντεταγμένη .

Επίσης οι συντεταγμένες που επιστρέφει αναφέρονται στην αρχική εικόνα και όχι του πλαισίου ,δηλαδή στην εικόνα που εντοπίστηκε το σχήμα (συνάρτηση bwboundaries) .

```
                    %
                    trregioncenter2 = round(STATS2(j).Centroid);
                    %
```

Αυτός ο τρόπος διαφέρει κυρίως από εδώ και κάτω .

```
                    %
                    closest_one_pixel_from_center2 =
onepixelsposition2(trregioncenter2(2),trregioncenter2(1));
                    regionimage2size = size(~regionimage2a);
                    [x2_crd, y2_crd] = ind2sub(regionimage2size
,closest_one_pixel_from_center2);
                    %
                    x2distance = (x2_crd - trregioncenter2(2))^2;
                    y2distance = (y2_crd - trregioncenter2(1))^2;
                    %
                    trperimetros4(n) = STATS4.Perimeter;
                    tremvadon4(n) = trperimetros4(n)*traktina4max;
                    %
                    traktina2a = sqrt(x2distance + y2distance);
                    trperimetros2(n) = STATS2(j).Perimeter;
                    tremvadon2(n) = trperimetros2(n)*traktina2a;
                    %
```

Στην περίπτωση που το σύνορο του γεωμετρικού σχήματος είναι μια γραμμή ,δηλαδή το εξωτερικό σύνορο και το εσωτερικό σύνορο είναι σχεδόν τέμνον το ένα το άλλο ,τοτε ως εμβαδόν θα υπολογίζεται το εσωτερικό του τριγώνου ,δηλαδή η τρύπα .

```
                    %
                    if traktina4/traktina2a >= 0.1
                        tremvadon(n) = tremvadon2(n) - tremvadon4(n);
                    else
                        tremvadon(n) = tremvadon2(n);
                    end
                    %
```

Η συνάρτηση regionprops υπολογίζει τη διάμετρο της περιοχής ,αλλά όπως θα δούμε παρακάτω ο κύκλος με την ακτίνα αυτήν δεν τέμνει τις πλευρές του τριγώνου .

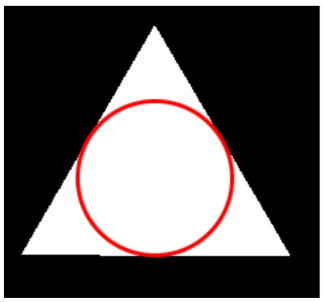
```
                    %
                    traktina2b = STATS2(j).EquivDiameter/2;
```

```
%
circle_characteristics_2a = [trregioncenter2 traktina2a];
circle_characteristics_2b = [trregioncenter2 traktina2b];
%
```

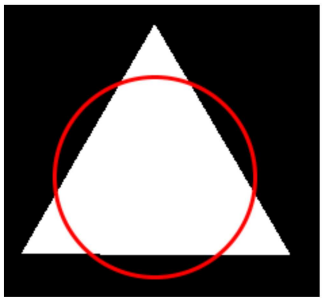
Παρακάτω εμφανίζουμε τον κύκλο που τέμνει τις πλευρές ενός τριγώνου πάνω στην εικόνα ~regionimage2a .Αυτή είναι ίδια με την περιοχή regionimage2b .Η διαφορά τους είναι στις πληροφορίες που επιστρέφει η συνάρτηση regionprops ,όπως το κέντρο του σχήματος που απεικονίζει το πλαίσιο .Αυτός είναι και ο λόγος που χρησιμοποιούμε την εικόνα ~regionimage2a .

Η εικόνα που παίρνει η συνάρτηση insertShape πρέπει να είναι τύπου single ή double κ.α. .

```
%
circle_in_triangle_2a =
insertShape(double(~regionimage2a),'Circle',...
    circle_characteristics_2a,'Linewidth',3,'Color','red');
figure;
imshow(circle_in_triangle_2a); title('circle in the triangle
(2a) in ~regionimage2a')
```



```
%
circle_in_triangle_2b =
insertShape(double(~regionimage2a),'Circle',...
    circle_characteristics_2b,'Linewidth',3,'Color','red');
figure;
imshow(circle_in_triangle_2b); title('circle in the triangle
(2b) in ~regionimage2a')
```



```
%
trids(n) = n;
```

```
%
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση τρίγωνα) οπου καταλαμβάνουν στο αντικείμενο STATS2 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS2 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως τρίγωνο .

```
%
% trin2 = 1
%
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
%
[~,~,P] = size(ImageRead);
isColorImage = P == 3;
%
if isColorImage
    q = 1;
    for o = 1:RegionsPixels1size(1)
        for p = 1:3
        pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
        q = q + 1;
        end
    end
    regioncolor(n) = mean(pixelscolor)
    %
    if regioncolor(n) < 54
```

```
                    trcolortext(n) = {',black ,'};
                end
                if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                        (regioncolor(n) > 82 && regioncolor(n) < 86)
||...
                        (regioncolor(n) > 157 && regioncolor(n) < 158)
                    trcolortext(n) = {',blue ,'};
                end
                if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                        (regioncolor(n) > 112 && regioncolor(n) < 115)
                    trcolortext(n) = {',red'};
                end
                if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                        (regioncolor(n) > 78 && regioncolor(n) < 82)
                    trcolortext(n) = {',green ,'};
                end
                if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                        (regioncolor(n) > 139 && regioncolor(n) < 146)
||...
                        (regioncolor(n) > 160 && regioncolor(n) < 161)
||...
                        (regioncolor(n) > 168 && regioncolor(n) < 169)
                    trcolortext(n) = {',purple ,'};
                end
                if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                        (regioncolor(n) > 134 && regioncolor(n) < 135)
                    trcolortext(n) = {',orange ,'};
                end
                if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                        (regioncolor(n) > 123 && regioncolor(n) < 127)
||...
                        (regioncolor(n) > 129 && regioncolor(n) < 134)
||...
                        (regioncolor(n) > 135 && regioncolor(n) < 137)
||...
                        (regioncolor(n) > 145 && regioncolor(n) < 146)
                    trcolortext(n) = {',light blue ,'};
                end
                if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                        (regioncolor(n) > 167 && regioncolor(n) < 168)
||...
                        (regioncolor(n) > 171 && regioncolor(n) < 172)
                    trcolortext(n) = {',yellow ,'};
                end
                if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                        (regioncolor(n) > 185 && regioncolor(n) < 186)
                    trcolortext(n) = {',pink ,'};
                end
                if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
                        (regioncolor(n) > 154 && regioncolor(n) < 155)
                    trcolortext(n) = {',gray ,'};
                end
            end
            n = n + 1;
        end
```

Παρακάτω ελέγχουμε για τετράγωνα ,στην περίπτωση που αυτά έχουν τρύπα εσωτερικά .Οι προυποθέσεις ανίχνευσης των τετραγώνων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές . Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
        if STATS2(j).Eccentricity > 0.18 && STATS2(j).Eccentricity <
```

```
0.98...
                        && STATS2(j).Extent > 0.95 && STATS2(j).Extent <= 1 ...
                        && STATS2(j).Solidity > 0.8 && STATS2(j).Solidity <= 1
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,με τρύπα εσωτερικά ,που έχει αναγνωριστεί ως τετράγωνο .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως τετράγωνο .

```
                        %
                        ecrec2 = STATS2(j).Eccentricity
                        exrec2 = STATS2(j).Extent
                        solrec2 = STATS2(j).Solidity
                        arearec2 = STATS2(j).Area
                        %
```

Παρακάτω ελέγχουμε εάν το γεωμετρικό σχήμα που υπάρχει σε αυτό το πλαίσιο του αντικειμένου STATS2 ,υπάρχει και σε κάποιο πλαίσιο του αντικειμένου STATS1 .Τα γεωμετρικά σχήματα του αντικειμένου STATS1 τα αναγνωρίζουμε από το κέντρο τους ,το οποίο περιέχεται στον πίνακα (shapescenter) . Εάν το γεωμετρικό σχήμα υπάρχει σε κάποιο πλαίσιο του αντικειμένου STATS1 τότε αποφεύγουμε την περαιτέρω ανάλυσή του .

```
                        %
                        centerrec2 = STATS1(i).Centroid;
                        currentcenter = centerrec2;
                        shapescentersize = size(shapescenter);
                        %
                        for u=1:shapescentersize(1)
                            if (currentcenter(1) / shapescenter(u,1) > 0.9) &&...
                                    (currentcenter(1) / shapescenter(u,1) < 1.1)
                                for v=1:shapescentersize(1)
                                    if (currentcenter(2) / shapescenter(v,2) > 0.9)
&&...
                                            (currentcenter(2) / shapescenter(u,2) <
1.1)
                                        sameshape = 1;
                                        break
                                    end
                                end
                            end
                        end
                        if sameshape == 1
                            continue
                        end
                        shapescenter(n,:) = vertcat(centerrec2);
                        %
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα με τρύπα στο εσωτερικό του .

```
                        %
                        STATS2in = 1;
                        %
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
                        %
                        figure;
                        imshow(regionimage1),title(n)
```

```
                    %
                    figure;
                    imshow(RegionMask1); title('RegionMask1')
                    %
                    figure;
                    imshow(displayImage); title('RegionImage in the original
image')
                    %
```

Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις recids για τα τετράγωνα ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η recids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντπίστηκε (στη συγκεκριμένη περίπτωση τετράγωνα) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (recids κ.τ.λ.) μπορεί να διαφέρει .

```
                    %
```

Παρακάτω παρατηρείται επίσης ότι ενώ παίρνουμε τις θέσεις του ορθογώνιου πλαισίου που περιβάλλει το γεωμετρικό σχήμα από το αντικείμενο STATS1 (STATS1(i).BoundingBox) ,δεν γίνεται το ίδιο και με την περίμετρο (STATS2(j).Perimeter) ,την οποία την παίρνουμε από το αντικείμενο STATS2 .Ο λόγος είναι ότι το αντικείμενο STATS1 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στην αρχική εικόνα ,ενώ το αντικείμενο STATS2 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στο πλαίσιο του αντικειμένου STATS1 .Δηλαδή το ίδιο γεωμετρικό σχήμα έχει εντοπιστεί σε άλλες εικόνες (με τη συνάρτηση bwboundaries) και άρα αυτό βρίσκεται σε άλλες συντεταγμένες .Η κλίμακα όμως δεν έχει αλλάξει στο γεωμετρικό σχήμα και για αυτο την περίμετρο μπορούμε να την πάρουμε από και τα δύο αντικείμενα STATS1 και STATS2 .

```
                    %
                    boxes(n,:) = STATS1(i).BoundingBox;
                    ids(n) = n;
                    %
                    recbboxes(n,:) = STATS1(i).BoundingBox;
                    recperimetros(n) = STATS2(j).Perimeter;
                    %
```

Όπως αναφέρθηκε ,στην περίπτωση που το γεωμετρικό σχήμα έχει τρύπα εσωτερικά πρέπει να βρεθεί και το 'εξωτερικό' εμβαδόν (δηλαδή το εμβαδόν του σχήματος που δημιουργείται από το εξωτερικό σύνορό του και μέσα) και το εσωτερικό εμβαδόν (δηλαδή το εμβαδόν του σχήματος που δημιουργείται από το εσωτερικό σύνορό του και μέσα) ,ώστε αφαιρώντας τα να βρούμε το εμβαδόν του γεωμετρικού σχήματος αυτού συμπεριλαμβανομένου και την τρύπα που αυτό έχει .

Παρακάτω θα υπολογιστεί το εμβαδόν του τετραγώνου .Το εμβαδόν του τετραγώνου θα υπολογιστεί με βάση τις διαστασεις του ορθογωνίκού πλαισίου (BoundingBox) που περιέχει το γεωμετρικό σχήμα που είναι ορθογώνιο .Ακολουθείται αυτός ο τρόπος γιατί τόσο το ορθογώνιο πλαίσιο ,όσο και το σχήμα (ορθογώνιο) ,θα έχουν τις ίδιες διαστάσεις .

Το ορθογώνιο πλαίσιο (BoundingBox) που περιβάλλει το γεωμετρικό σχήμα που δημιουργείται από

το εσωτερικό σύνορο μας το δίνει το αντκείμενο STATS4 .Το άλλο ορθογώνιο πλαίσιο που περιβάλλει το γεωμετρικό σχήμα (BoundingBox) που δημιουργείται από το εξωτερικό σύνορο μας το δίνει το αντκείμενο STATS2 .

```
%
recbboxes4(n,:) = STATS4.BoundingBox;
basi4 = recbboxes(n,3);
ypsos4 = recbboxes(n,4);
recemvadon4(n) = ypsos4 * basi4;
%
recbboxes2(n,:) = STATS2(j).BoundingBox;
basi2 = recbboxes(n,3);
ypsos2 = recbboxes(n,4);
recemvadon2(n) = ypsos2 * basi2;
%
```

Στην περίπτωση που το σύνορο του γεωμετρικού σχήματος είναι μια γραμμή ,δηλαδή το εξωτερικό σύνορο και το εσωτερικό σύνορο είναι σχεδόν τέμνον το ένα το άλλο ,τοτε ως εμβαδόν θα υπολογίζεται το εσωτερικό του τετραγώνου ,δηλαδή η τρύπα .

```
%
if recemvadon4(n)/recemvadon2(n) >= 0.1
    recemvadon(n) = recemvadon2(n) - recemvadon4(n);
else
    recemvadon(n) = recemvadon2(n);
end
%
recids(n) = n;
%
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση τετράγωνα) οπου καταλαμβάνουν στο αντικείμενο STATS2 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS2 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως τετράγωνο .

```
%
% recin2 = 1
%
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
                    %
                    [~,~,P] = size(ImageRead);
                    isColorImage = P == 3;
                    %
                    if isColorImage
                        q = 1;
                        for o = 1:RegionsPixels1size(1)
                            for p = 1:3
                                pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
                                q = q + 1;
                            end
                        end
                        regioncolor(n) = mean(pixelscolor)
                        %
                        if regioncolor(n) < 54
                            reccolortext(n) = {',black ,'};
                        end
                        if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                                (regioncolor(n) > 82 && regioncolor(n) < 86)
||...
                                (regioncolor(n) > 157 && regioncolor(n) < 158)
                            reccolortext(n) = {',blue ,'};
                        end
                        if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                                (regioncolor(n) > 112 && regioncolor(n) < 115)
                            reccolortext(n) = {',red'};
                        end
                        if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                                (regioncolor(n) > 78 && regioncolor(n) < 82)
                            reccolortext(n) = {',green ,'};
                        end
                        if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                                (regioncolor(n) > 139 && regioncolor(n) < 146)
||...
                                (regioncolor(n) > 160 && regioncolor(n) < 161)
||...
                                (regioncolor(n) > 168 && regioncolor(n) < 169)
                            reccolortext(n) = {',purple ,'};
                        end
                        if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                                (regioncolor(n) > 134 && regioncolor(n) < 135)
                            reccolortext(n) = {',orange ,'};
                        end
                        if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                                (regioncolor(n) > 123 && regioncolor(n) < 127)
||...
                                (regioncolor(n) > 129 && regioncolor(n) < 134)
```

```
||...
                                (regioncolor(n) > 135 && regioncolor(n) < 137)
||...
                                (regioncolor(n) > 145 && regioncolor(n) < 146)
                            reccolortext(n) = {',light blue ,'};
                        end
                        if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                                (regioncolor(n) > 167 && regioncolor(n) < 168)
||...
                                (regioncolor(n) > 171 && regioncolor(n) < 172)
                            reccolortext(n) = {',yellow ,'};
                        end
                        if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                                (regioncolor(n) > 185 && regioncolor(n) < 186)
                            reccolortext(n) = {',pink ,'};
                        end
                        if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
                                (regioncolor(n) > 154 && regioncolor(n) < 155)
                            reccolortext(n) = {',gray ,'};
                        end
                    end
                    n = n + 1;
                end
```

Παρακάτω ελέγχουμε για πολύγωνα ,στην περίπτωση που αυτά έχουν τρύπα εσωτερικά .Οι προυποθέσεις ανίχνευσης των πολυγώνων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές . Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
                if (STATS2(j).Eccentricity > 0.04) && (STATS2(j).Eccentricity <
0.14) &&...
                        (STATS2(j).Extent > 0.7) && (STATS2(j).Extent < 0.78)
&&...
                        (STATS2(j).Solidity > 0.97) && (STATS2(j).Solidity <
0.985)
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,με τρύπα εσωτερικά ,που έχει αναγνωριστεί ως τρίγωνο .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως πολύγωνο .

```
                    %
                    ecpol2 = STATS2(j).Eccentricity
                    expol2 = STATS2(j).Extent
                    solpol2 = STATS2(j).Solidity
                    areapol2 = STATS2(j).Area
                    %
                    centerpol2 = STATS1(i).Centroid;
                    currentcenter = centerpol2;
                    shapescentersize = size(shapescenter);
                    %
```

Παρακάτω ελέγχουμε εάν το γεωμετρικό σχήμα που υπάρχει σε αυτό το πλαίσιο του αντικειμένου STATS2 ,υπάρχει και σε κάποιο πλαίσιο του αντικειμένου STATS1 .Τα γεωμετρικά σχήματα του αντικειμένου STATS1 τα αναγνωρίζουμε από το κέντρο τους ,το οποίο περιέχεται στον πίνακα (shapescenter) . Εάν το γεωμετρικό σχήμα υπάρχει σε κάποιο πλαίσιο του αντικειμένου STATS1 τότε αποφεύγουμε την περαιτέρω ανάλυσή του .

```
                    %
                    for u=1:shapescentersize(1)
                        if (currentcenter(1) / shapescenter(u,1) > 0.9) &&...
```

```
                                (currentcenter(1) / shapescenter(u,1) < 1.1)
                            for v=1:shapescentersize(1)
                                if (currentcenter(2) / shapescenter(v,2) > 0.9)
&&...
                                        (currentcenter(2) / shapescenter(u,2) <
1.1)
                                    sameshape = 1;
                                    break
                                end
                            end
                        end
                    end
                    if sameshape == 1
                        continue
                    end
                    shapescenter(n,:) = vertcat(centerpol2);
                    %
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα με τρύπα στο εσωτερικό του .

```
                    %
                    STATS2in = 1;
                    %
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
                    %
                    figure;
                    imshow(regionimage1),title(n)
                    %
                    figure;
                    imshow(RegionMask1); title('RegionMask1')
                    %
                    figure;
                    imshow(displayImage); title('RegionImage in the original
image')
                    %
```

Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις polids για τα πολύγωνα ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η polids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντπίστηκε (στη συγκεκριμένη περίπτωση πολύγωνα) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (polids κ.τ.λ.) μπορεί να διαφέρει .

```
                    %
```

Παρακάτω παρατηρείται επίσης ότι ενώ παίρνουμε τις θέσεις του ορθογώνιου πλαισίου που περιβάλλει το γεωμετρικό σχήμα από το αντικείμενο STATS1 (STATS1(i).BoundingBox) ,δεν γίνεται το ίδιο και με την περίμετρο (STATS2(j).Perimeter) ,την οποία την παίρνουμε από το

αντικείμενο STATS2 .Ο λόγος είναι ότι το αντικείμενο STATS1 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στην αρχική εικόνα ,ενώ το αντικείμενο STATS2 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στο πλαίσιο του αντικειμένου STATS1 .Δηλαδή το ίδιο γεωμετρικό σχήμα έχει εντοπιστεί σε άλλες εικόνες (με τη συνάρτηση bwboundaries) και άρα αυτό βρίσκεται σε άλλες συντεταγμένες .Η κλίμακα όμως δεν έχει αλλάξει στο γεωμετρικό σχήμα και για αυτο την περίμετρο μπορούμε να την πάρουμε από και τα δύο αντικείμενα STATS1 και STATS2 .

```
%
boxes(n,:) = STATS1(i).BoundingBox;
ids(n) = n;
%
polbboxes(n,:) = STATS1(i).BoundingBox;
polperimetros(n) = STATS2(j).Perimeter;
polids(n) = n;
%
```

Την παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση πολύγωνα) όπου καταλαμβάνουν στο αντικείμενο STATS2 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS2 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως πολύγωνο .

```
%
% polin2 = 1
%
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
                    %
                    [~,~,P] = size(ImageRead);
                    isColorImage = P == 3;
                    %
                    if isColorImage
                        q = 1;
                        for o = 1:RegionsPixels1size(1)
                            for p = 1:3
                                pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
                                q = q + 1;
                            end
                        end
                        regioncolor(n) = mean(pixelscolor)
                        %
                        if regioncolor(n) < 54
                            polcolortext(n) = {',black ,'};
                        end
                        if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                                (regioncolor(n) > 82 && regioncolor(n) < 86)
||...
                                (regioncolor(n) > 157 && regioncolor(n) < 158)
                            polcolortext(n) = {',blue ,'};
                        end
                        if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                                (regioncolor(n) > 112 && regioncolor(n) < 115)
                            polcolortext(n) = {',red'};
                        end
                        if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                                (regioncolor(n) > 78 && regioncolor(n) < 82)
                            polcolortext(n) = {',green ,'};
                        end
                        if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                                (regioncolor(n) > 139 && regioncolor(n) < 146)
||...
                                (regioncolor(n) > 160 && regioncolor(n) < 161)
||...
                                (regioncolor(n) > 168 && regioncolor(n) < 169)
                            polcolortext(n) = {',purple ,'};
                        end
                        if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                                (regioncolor(n) > 134 && regioncolor(n) < 135)
                            polcolortext(n) = {',orange ,'};
                        end
                        if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                                (regioncolor(n) > 123 && regioncolor(n) < 127)
||...
                                (regioncolor(n) > 129 && regioncolor(n) < 134)
||...
                                (regioncolor(n) > 135 && regioncolor(n) < 137)
||...
                                (regioncolor(n) > 145 && regioncolor(n) < 146)
                            polcolortext(n) = {',light blue ,'};
                        end
                        if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                                (regioncolor(n) > 167 && regioncolor(n) < 168)
||...
                                (regioncolor(n) > 171 && regioncolor(n) < 172)
                            polcolortext(n) = {',yellow ,'};
                        end
```

```
                    if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                            (regioncolor(n) > 185 && regioncolor(n) < 186)
                        polcolortext(n) = {',pink ,'};
                    end
                    if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
                            (regioncolor(n) > 154 && regioncolor(n) < 155)
                        polcolortext(n) = {',gray ,'};
                    end
                end
                n = n + 1;
            end
```

Παρακάτω ελέγχουμε για κύκλους ,στην περίπτωση που αυτά έχουν τρύπα εσωτερικά .Οι προυποθέσεις ανίχνευσης των κύκλων (Eccentricity ,Extent ,Solidity) δεν είναι αποκληστικές . Αυτές μπορεί να υπολογιστούν καλύτερα με πολλά πειράματα .

```
            if STATS2(j).Eccentricity < 0.2 &&...
                    STATS2(j).Extent > 0.78 && STATS2(j).Extent < 79 &&...
                    STATS2(j).Solidity > 0.993 && STATS2(j).Solidity < 0.996
```

Οι παρακάτω μεταβλητές φανερώνουν τα χαρακτηριστικά ενός γεωμετρικού σχήματος ,με τρύπα εσωτερικά ,που έχει αναγνωριστεί ως τρίγωνο .Χρήσιμη πληροφορία στην περίπτωση που ένα γεωμετρικό σχήμα έχει λανθασμένα αναγνωριστεί ως κύκλος .

```
                %
                ecpcir2 = STATS2(j).Eccentricity
                excir2 = STATS2(j).Extent
                solcir2 = STATS2(j).Solidity
                areacir2 = STATS2(j).Area
                %
```

Παρακάτω ελέγχουμε εάν το γεωμετρικό σχήμα που υπάρχει σε αυτό το πλαίσιο του αντικειμένου STATS2 ,υπάρχει και σε κάποιο πλαίσιο του αντικειμένου STATS1 .Τα γεωμετρικά σχήματα του αντικειμένου STATS1 τα αναγνωρίζουμε από το κέντρο τους ,το οποίο περιέχεται στον πίνακα (shapescenter) . Εάν το γεωμετρικό σχήμα υπάρχει σε κάποιο πλαίσιο του αντικειμένου STATS1 τότε αποφεύγουμε την περαιτέρω ανάλυσή του .

```
                %
                centercir2 = STATS1(i).Centroid;
                currentcenter = centercir2;
                shapescentersize = size(shapescenter);
                %
                for u=1:shapescentersize(1)
                    if (currentcenter(1) / shapescenter(u,1) > 0.9) &&...
                            (currentcenter(1) / shapescenter(u,1) < 1.1)
                        for v=1:shapescentersize(1)
                            if (currentcenter(2) / shapescenter(v,2) > 0.9)
&&...
                                    (currentcenter(2) / shapescenter(u,2) <
1.1)
                                sameshape = 1;
                                break
                            end
                        end
                    end
                end
                if sameshape == 1
                    continue
                end
```

```
shapescenter(n,:) = vertcat(centercir2);
%
```

Η παρακάτω μεταβλητή χρησιμοποιείται ως δείκτης ,ότι εντοπίστηκε ένα γεωμετρικο σχήμα με τρύπα στο εσωτερικό του .

```
%
STATS2in = 1;
%
```

Παρακάτω θα εμφανίσουμε το γεωμετρικό σχήμα που εντοπίστηκε ,μαζί με τον αριθμό αναγνώρισής του .

```
%
figure;
imshow(regionimage1),title(n)
%
figure;
imshow(RegionMask1); title('RegionMask1')
%
figure;
imshow(displayImage); title('RegionImage in the original
image')
%
```

Παρακάτω θα πάρουμε κάποιες πληροφορίες για τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα .Για να ξέρουμε σε ποια αντικείμενα αναφέρονται αυτές οι πληροφορίες ,έχουμε κάποιες μεταβλητές ,τις ids για όλα τα γεωμετρικά σχήματα που εντοπίσαμε στην εικόνα και τις cirids για τους κύκλους ,καθώς και κάποιες αντίστοιχες για τα υπόλοιπα γεωμετρικά σχήματα που εντοπίσαμε .Επίσης άμα παρατηρήσουμε θα δούμε ότι όλες έχουν την ίδια αρίθμηση (k) .Τότε γιατί θέλουμε και τις δύο ;Ο λόγος είναι ότι η αρίθμιση των μεταβλητών που αναφέρονται σε συγκεκριμένα γεωμετρικά σχήματα (όπως η cirids) σταματάνε την αρίθμηση στο αντίστοιχο τελευταίο γεωμετρικό σχήμα που εντπίστηκε (στη συγκεκριμένη περίπτωση κύκλοι) ,ενώ η αρίθμηση της μεταβλητής ids σταματάει στο τελευταίο γεωμετρικό σχήμα που εντοπίστηκε ,οποιοδήποτε και να είναι αυτό .Άρα στο τέλος το μέγεθος της μεταβλητής ids με τις υπόλοιπες (cirids κ.τ.λ.) μπορεί να διαφέρει .

```
%
```

Παρακάτω παρατηρείται επίσης ότι ενώ παίρνουμε τις θέσεις του ορθογώνιου πλαισίου που περιβάλλει το γεωμετρικό σχήμα από το αντικείμενο STATS1 (STATS1(i).BoundingBox) ,δεν γίνεται το ίδιο και με την περίμετρο (STATS2(j).Perimeter) ,την οποία την παίρνουμε από το αντικείμενο STATS2 .Ο λόγος είναι ότι το αντικείμενο STATS1 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στην αρχική εικόνα ,ενώ το αντικείμενο STATS2 έχει 'αίσθηση' της θέσης του γεωμετρικού σχήματος πάνω στο πλαίσιο του αντικειμένου STATS1 .Δηλαδή το ίδιο γεωμετρικό σχήμα έχει εντοπιστεί σε άλλες εικόνες (με τη συνάρτηση bwboundaries) και άρα αυτό βρίσκεται σε άλλες συντεταγμένες .Η κλίμακα όμως δεν έχει αλλάξει στο γεωμετρικό σχήμα και για αυτο την περίμετρο μπορούμε να την πάρουμε από και τα δύο αντικείμενα STATS1 και STATS2 .

```
%
boxes(n,:) = STATS1(i).BoundingBox;
ids(n) = n;
%
cirperimetros(n) = STATS2(j).Perimeter;
%
```

Όπως αναφέρθηκε ,στην περίπτωση που το γεωμετρικό σχήμα έχει τρύπα εσωτερικά πρέπει να βρεθεί και το 'εξωτερικό' εμβαδόν (δηλαδή το εμβαδόν του σχήματος που δημιουργείται από το εξωτερικό σύνορό του και μέσα) και το εσωτερικό εμβαδόν (δηλαδή το εμβαδόν του σχήματος που δημιουργείται από το εσωτερικό σύνορό του και μέσα) ,ώστε αφαιρώντας τα να βρούμε το εμβαδόν του γεωμετρικού σχήματος αυτού συμπεριλαμβανομένου και την τρύπα που αυτό έχει .

Παρακάτω υπολογίζεται το εμβαδόν του κύκλου που εντοπίστηκε .Για αρχή παίρνουμε τον κύριο (MajorAxisLength) και τον δευτερεύον (MinorAxisLength) άξονα κάθε έλλειψης που περιβάλει τα σημεία του σχήματος .Μια έλλειψη που τείνει να γίνει κύκλος ,η εκκεντικότητα τείνει στο 0 και τότε ο κύριος και ο δευτερεύον άξονας της έλλειψης τείνουν να έχουν την ίδια τιμή .Τότε οι άξονες της έλλειψης αποτελούν τη διάμετρο ενός κύκλου .Επειδή λόγω θορύβου μπορεί να μην έχουμε ακριβώς κύκλο θα υπολογίσουμε τη μέση τιμή του κύριου και του δευτερεύων άξονα (συνάρτηση mean) ώστε έπειτα να υπολογίσουμε την ακτίνα του κύκλου .

Τους άξονες της έλλειψης που που περιβάλει το σχήμα που δημιουργείται από το εσωτερικό σύνορο και μέσα ,τους δίνει το αντικείμενο STATS4 .Τους άξονες της έλλειψης που που περιβάλει το σχήμα που δημιουργείται από το εξωτερικό σύνορο και μέσα ,τους δίνει το αντικείμενο STATS2 .

Παρόλο που η μεταβλητή diameters είναι ένα διάνυσμα ,πρέπει να προσθέσουμε μία διάσταση ((i,:)) ,ώστε να έχει τις ίδιες διαστάσεις με τον πίνακα centers ,ώστε έπειτα να τους συγχωνεύσουμε και να δημιουργήσουμε τον πίνακα circle .Ο πίνακας circle θα χρησιμοποιηθεί για να τονιστεί ο κύκλος που περιέχει το πλαίσιο αυτό στην αρχική εικόνα .

```
                    %
                    diameters4(n,:) = mean([STATS4.MajorAxisLength
STATS4.MinorAxisLength],2);
                    radii4(n) = diameters4(n,:)./2;
                    ciremvadon4(n) = 3.14*(radii4(n))^2;
                    %
                    diameters2(n,:) = mean([STATS2(j).MajorAxisLength
STATS2(j).MinorAxisLength],2);
                    radii2(n) = diameters(n,:)./2;
                    ciremvadon2(n) = 3.14*(radii2(n))^2;
                    %
```

Στην περίπτωση που το σύνορο του γεωμετρικού σχήματος είναι μια γραμμή ,δηλαδή το εξωτερικό σύνορο και το εσωτερικό σύνορο σχεδόν τέμνουν το ένα το άλλο ,τοτε ως εμβαδόν θα υπολογίζεται το εσωτερικό του τετραγώνου ,δηλαδή η τρύπα .

```
                    %
                    if radii4(n)/radii2(n) >= 0.1
                        ciremvadon(n) = ciremvadon2(n) - ciremvadon4(n);
                    else
                        ciremvadon(n) = ciremvadon2(n);
                    end
                    %
```

Για να τονιστούν οι κύκλοι που εντοπίστηκαν και αναλύθηκαν από τη συνάρτηση regionprops πάνω στην αρχική εικόνα ,χρειάζονται τα χαρακτηριστικά των κύκλων ,δηλαδή το κέντρο και η ακτίνα του κύκλου .Παρακάτω παίρνουμε τον κύκλο του αντικειμένου STATS2 ,δηλαδή μέχρι το εξωτερικό σύνορο .

Η συνάρτηση regionprops επιστρέφει τις συντεταγμένες του κέντρου του σχήματος ανάποδα ,δηλαδή επιστρέφει πρώτα την y-συντεταγμένη και μετά την x-συντεταγμένη . Επίσης οι

συντεταγμένες που επιστρέφει αναφέρονται στην εικόνα που εντοπίστηκε το αντικείμενο από την συνάρτηση bwboundaries , δηλαδή στην regionimage2a .

```
%
centers2(n,:) = STATS1(i).Centroid;
circle = [centers2(n,:) radii2(n)];
%
cirids(n) = n;
%
```

Η παρακάτω μεταβλητή την χρησιμοποιούμε για να 'υπογραμμίσουμε' τη θέση των συγκεκριμένων γεωμετρικών σχημάτων (στη συγκεκριμένη περίπτωση κύκλοι) οπου καταλαμβάνουν στο αντικείμενο STATS2 .Οι θέσεις του διανύσματος showinimageindex αναφέρονται στις αντίστοιχες θέσεις του αντικειμένου STATS2 .Τη χρήση του διανύσματος showinimageindex θα τη δούμε προς το τέλος ,όπου θα απεικονίσουμε τα γεωμετρικά σχήματα που εντοπίσαμε .

```
%
showinimageindex(i) = 1;
%
```

Τη παρακάτω μεταβλητή την έχουμε για να βλέπουμε εάν το σχήμα στην εικόνα αναγνωρίστηκε ως κύκλο .

```
%
% cirin2 = 1
%
```

Παρακάτω επεξηγήται ο τρόπος υπολογισμού του χρώματος ενός γεωμετρικού σχήματος . Εάν η εικόνα είναι τριών διαστάσεων ,μία για κάθε χρώμα ,κόκκινο ,κίτρινο και μπλε ,τότε παίρνουμε από κάθε pixel του σχήματος (RegionsPixels1) ,τις τιμές για αυτά τα χρώματα .Οι τιμές αυτές καταχωρούνται στο διάνυσμα pixelscolor .

Έπειτα υποθέτοντας ότι σε κάθε σχήμα κυριαρχεί ένα σχεδόν χρώμα , το χρώμα αυτό του σχήματος υπολογίζεται από την μέση τιμή των χρωμάτων κόκκινο ,πράσινο και μπλέ μαζί που πήραμε από κάθε pixel του σχήματος (regioncolor) .Έτσι ξέρουμε συνήθως το χρώμα που επικρατεί σε κάθε περιοχή .

Το χρώμα που μπορεί να έχει κάθε pixel προκύπτει με συνδυασμό των χρωμάτων αυτών .Ένα χρώμα μπορεί να έχει μια μέση τιμή σχεδόν ίδια με ένα άλλο διαφορετικό χρώμα ,λόγω ότι το άρθροισμα των τριών βασικών χρωμάτων είναι σχεδόν ίδιο .Δηλαδή αυτός ο τρόπος δεν έχει ένα μήκος χρώματος και ούτε έχει πάντα σωστά αποτελέσματα .Πρέπει να δείνεται ιδιαίτερη σημασία στην τιμή κάθε χρώματος ,ίδιου ή όχι .Σε περιπτώσεις ειδικά που το γεωμετρικό σχήμα δεν αποτελείται εξολοκλήρου από ένα χρώμα ,το αποτέλεσμα μπορεί να είναι λάθος .

Αυτή μέθοδος υπολογισμού του χρώματος ενός αντικειμένου είναι κυρίως χρήσιμη για αναγνώριση γνωστών χρωμάτων ,που η τιμή τους δηλαδή είναι γνωστή από προηγουμένως .

```
%
[~,~,P] = size(ImageRead);
isColorImage = P == 3;
%
if isColorImage
    q = 1;
    for o = 1:RegionsPixels1size(1)
        for p = 1:3
```

```
                                pixelscolor(q) =
displayImage(RegionsPixels1(o,2),RegionsPixels1(o,1),p);
                                q = q + 1;
                            end
                        end
                        regioncolor(n) = mean(pixelscolor)
                        %
                        if regioncolor(n) < 54
                            circolortext(n) = {',black ,'};
                        end
                        if (regioncolor(n) > 61 && regioncolor(n) < 78) ||...
                                (regioncolor(n) > 82 && regioncolor(n) < 86)
||...
                                (regioncolor(n) > 157 && regioncolor(n) < 158)
                            circolortext(n) = {',blue ,'};
                        end
                        if (regioncolor(n) > 86 && regioncolor(n) < 91) ||...
                                (regioncolor(n) > 112 && regioncolor(n) < 115)
                            circolortext(n) = {',red'};
                        end
                        if (regioncolor(n) > 94 && regioncolor(n) < 96) ||...
                                (regioncolor(n) > 78 && regioncolor(n) < 82)
                            circolortext(n) = {',green ,'};
                        end
                        if (regioncolor(n) > 115 && regioncolor(n) < 116) ||...
                                (regioncolor(n) > 139 && regioncolor(n) < 146)
||...
                                (regioncolor(n) > 160 && regioncolor(n) < 161)
||...
                                (regioncolor(n) > 168 && regioncolor(n) < 169)
                            circolortext(n) = {',purple ,'};
                        end
                        if (regioncolor(n) > 127 && regioncolor(n) < 129) ||...
                                (regioncolor(n) > 134 && regioncolor(n) < 135)
                            circolortext(n) = {',orange ,'};
                        end
                        if (regioncolor(n) > 116 && regioncolor(n) < 121) ||...
                                (regioncolor(n) > 123 && regioncolor(n) < 127)
||...
                                (regioncolor(n) > 129 && regioncolor(n) < 134)
||...
                                (regioncolor(n) > 135 && regioncolor(n) < 137)
||...
                                (regioncolor(n) > 145 && regioncolor(n) < 146)
                            circolortext(n) = {',light blue ,'};
                        end
                        if (regioncolor(n) > 152 && regioncolor(n) < 153) ||...
                                (regioncolor(n) > 167 && regioncolor(n) < 168)
||...
                                (regioncolor(n) > 171 && regioncolor(n) < 172)
                            circolortext(n) = {',yellow ,'};
                        end
                        if (regioncolor(n) > 169 && regioncolor(n) < 171) ||...
                                (regioncolor(n) > 185 && regioncolor(n) < 186)
                            circolortext(n) = {',pink ,'};
                        end
                        if (regioncolor(n) > 121 && regioncolor(n) < 122) ||...
                                (regioncolor(n) > 154 && regioncolor(n) < 155)
                            circolortext(n) = {',gray ,'};
                        end
                    end
```

```
                    n = n + 1;
                end
            end
        end
    end
end
```

ΚΕΦΑΛΑΙΟ 4 : Εμφάνιση αποτελεσμάτων

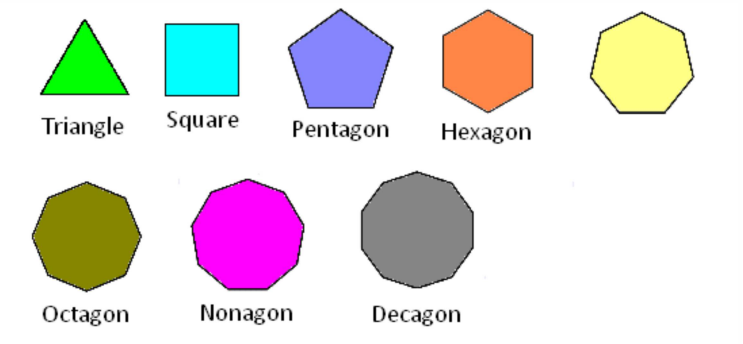
Εισαγωγή κεφαλαίου

Το κεφάλαιο αυτό ασχολείται με την εμφάνιση των αποτελεσμάτων των επεξεργασιών και των αναλύσεων που έγιναν στα παραπάνω κεφάλαια .

Το ένα αποτέλεσμα που εμφανίζεται είναι η απόσταση που έχει κάθε γεωμετρικό σχήμα με τα υπόλοιπα στην εικόνα .Η απόσταση υπολογίζεται με την ευκλείδεια μέθοδο σε αυτό το κεφάλαιο ,αλλά για τον υπολογισμό της χρειαζόμαστε κάποιες μεταβλητές που έχουμε πάρει στο παραπάνω κεφάλαιο .Τα αποτελέσματα αυτά εμφανίζονται στο command window του matlab .

Τα υπόλοιπα αποτελέσματα ,όπως τον αριθμό αναγνώρισης του γεωμετρικού σχήματος ,το χρώμα του και το εμβαδόν του ,γίνονται οπτικά .Τα αποτελέσματα αυτά θα εμφανίζονται πάνω (ή κάτω) από το ορθογώνιο που περιβάλλει το γεωμετρικό σχήμα και στο οποίο αναφέρονται αυτά τα αποτελέσματα .

Για την οπτική λειτουργία του παρόντος κεφαλαίου θα χρησιμοποιηθεί η παρακάτω εικόνα



Κώδικας

```
%%
```

Παρακάτω θα υπολογίζουμε την απόσταση που έχει κάθε γεωμετρικό σχήμα με τα υπόλοιπα (distance) της εικόνας ,με την ευκλείδεια μέθοδο .Με βάση την μέθοδο αυτή ,δυο σημεία (για μας δύο σημεία) σε ένα σύστημα αξόνων δύο διαστάσεων ,που αναγνωρίζονται από τις συντεταγμένες (x1 ,y1 και x2 ,y2 αντίστοιχα) απέχουν μεταξύ τους :
ρίζα((x2-x1)^2 + (y2-y1)) .

Τα x και y σημεία είναι το κέντρο του ορθογώνιου πλαισίου που περιέχει οποιοδήποτε γεωμετρικό

σχήμα της εικόνας .

Άρα για αρχή ,πρέπει να βρούμε το κεντρικό σημείο του κάθε ορθογωνίου αυτού .Το κεντρικό σημείο του ορθογωνίου το βρίσκουμε από τα στοιχεία που μας δίνει η συνάρτηση regionprops για τα ορθογώνια αυτά .Για το ορθογώνιο αυτό η συνάρτηση μας δίνει σε ένα διάνυσμα (BoundingBox) τις συντεταγμένες της κάτω-αριστερής γωνίας (x,y) ,το πλάτος (width) και το ύψος απο αυτήν την γωνία (height) .Άρα οι συντεταγμένες του κεντρικού σημείου του ορθογωνίου είναι:
xk = x + width/2
yk = y + height/2

Έπειτα θα χρησιμοποιήσουμε τις συντεταγμένες του κάθε ορθογωνίου πλαισίου που βρήκαμε για να υπολογίσουμε την απόσταση μεταξύ των σχημάτων της εικόνας (`distance`) .

Το γεωμετρικό σχήμα που έχουμε ως σημείο υπολογισμού κάθε φορα ,είναι το σχήμα u .Εμείς μετράμε την απόσταση που υπάρχει μεταξύ αυτού του γεωμετρικού σχήματος (u) και των υπολοίπων ,όπου τα υπόλοιπα γεωμετρικά σχήματα δηλώνονται από τη μεταβλητή v .Η απόσταση μετριέται μεταξύ των κέντρων των ορθογωνίων πλαισίων που περιβάλουν τα γεωμετρικά σχήματα (BoundingBox) .

```
if ~isempty(boxes)

   xcenter = boxes(:,1) + (boxes(:,3)./2);
   ycenter = boxes(:,2) + (boxes(:,4)./2);

   numobjects = size(boxes);
   for u = 1 : (numobjects(1) - 1)
      v = u+1;
      while v <= numobjects(1)
         x = (xcenter(v) - xcenter(u))^2;
         y = (ycenter(v) - ycenter(u))^2;
         distance(v) = sqrt(x + y);
```

Παρακάτω θα εμφανίζουμε την απόσταση που υπάρχει μεταξύ του γεωμετρικού σχήματος αναφοράς (u) με ένα άλλο γεωμετρικο σχήμα (v) της εικόνας (συνάρτηση disp) .Τα σχήματα u και v αναγνωρίζονται από τον αριθμό τους .

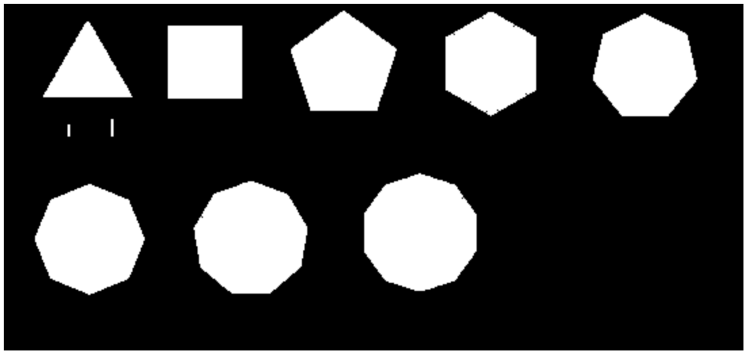
Για να εμφανίσουμε μια πρόταση που περιέχει και αριθμούς και χαρακτήρες ,πρέπει να μετατρέψουμε τους αριθμούς σε τύπου δεδομένων χαρακτήρα (συνάρτηση num2str) .Έπειτα βάζουμε την πρόταση που θέλουμε να εμφανίζεται στο command window σε ένα διάνυσμα (displayvector) .Σε κάθε στοιχείο του διανύσματος υπάρχει μια λέξη .

```
         displayvector = [num2str(ids(u)) ,'->', num2str(ids(v)) ,'='
,num2str(distance(v))];
         disp(displayvector)

         v = v + 1;
      end
   end
end
```

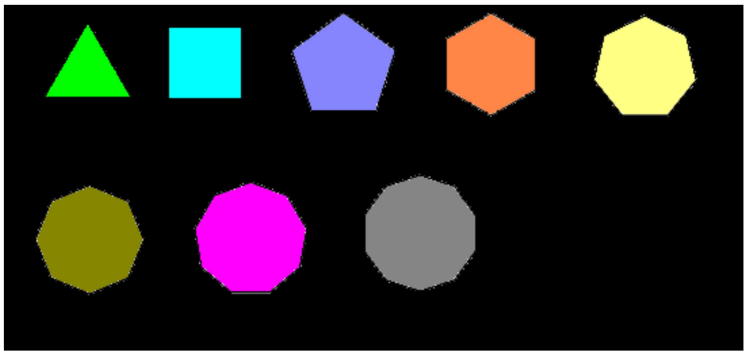
Το παρακάτω σημείο του προγράμματος υποδεικνύει τον τρόπο για να δημιουργηθεί μια εικόνα ,έχρωμη (displayImage) και δυαδική (RegionMask) ,όπου αυτή περιέχει μόνο τα γεωμετρικα σχήματα που θέλουμε .

Τα γεωμετρικά σχήματα αλλά και άλλα αντικείμενα που εντοπίστηκαν στην εικόνα είναι 'αριθμημένα' στα αντικείμενα STATS1 και STATS2 .Η μεταβλητή showinimageindex 'δείχνει' σε όλα τα αντικείμενα (μαζί και των γεωμετρικών σχημάτων) που περιέχονται στα αντικείμενα STATS1 και STATS2 .Σε όποια θέση όμως το διάνυσμα showinimageindex έχει 1 ,αυτό το αντικείμενο θα προσθέσουμε στις εικόνες που θα δημιουργήσουμε .

```
RegionsPixels2 = [];

if STATS1in == 1
    for i=1:length(STATS1)
        if showinimageindex(i) == 1
            RegionsPixels2 = vertcat(RegionsPixels2,STATS1(i).PixelList);
        end
    end
end

if STATS2in == 1
    for j=1:length(STATS2)
        if showinimageindex(j) == 1
            RegionsPixels2 = vertcat(RegionsPixels2,STATS2(j).PixelList);
        end
    end
end

if ~isempty(RegionsPixels2)
    RegionMask = false(size(padGRAY));
    ind = sub2ind(size(RegionMask), RegionsPixels2(:,2),...
                RegionsPixels2(:,1));
    RegionMask(ind) = true;

    displayImage = padImageRead;
    displayImage(~repmat(RegionMask,1,1,3)) = 0;

    figure;
    imshow(RegionMask); title('all shapes in white')
```



```
    figure;
    imshow(displayImage); title('all shapes in color')
```



```
end
```

Παρακάτω δημιουργούνται εικόνες που περιέχουν τα γεωμετρικά σχήματα που εντοπίστηκαν στην αρχική εικόνα ξεχωριστά ,δηλαδή όλα τα τετράγωνα ,τα τρίγωνα κ.ο.κ. . Τα γεωμετρικά σχήματα θα εμφανίζονται εφόσον έχει ανιχνευτεί έστω και ένα είδος γεωμετρικού σχήματος από τα παραπάνω (if ~isempty) .

Τα γεωμετρικά σχήματα θα τονίζονται από ένα ορθογώνιο που θα αναφέρει από πάνω (ή από κάτω του) διάφορες πληροφορίες για το σχήμα αυτό ,όπως χρώμα ,αριθμό αναγνώρισή του κ.α. .Το ορθογώνιο αυτό θα το ζωγραφίσουμε στην εικόνα εμείς ,αλλά πρέπει να έχει τις σωστές διαστάσεις ,ώστε να περικλύει όλο το γεωμετρικό σχήμα .Ένα τέτοιο ορθογώνιο ,με τις σωστές διαστάσεις ,το έχει δημιουργήσει η συνάρτηση regionprops (BoundngBox) και είναι αυτό που οριοθετεί κάθε τετραγωνισμένο πλαίσιο των αντικειμένων STATS1 και STATS2 .Το ίδιο ορθογώνιο χρησιμοποιήσαμε στην αρχή του κεφαλαίου για τον υπολογισμό της απόστασης των γεωμετρικών σχημάτων που υπάρχουν στην εικόνα μεταξύ τους .

Μια συνάρτηση που μπορεί να χρησιμοποιηθεί για να ζωγραφίσει ένα γεωμετρικό σχήμα σε μια εικόνα ,είναι η insertObjectAnnotation .Ένα πλεονέκτημα που έχει αυτή η συνάρτηση είναι ότι μας επιτρέπει να γράψουμε και σχόλια στα γεωμετρικά σχήματα που θα ζωγραφίσει .Τα σχόλια που θα εμφανίσουμε είναι το χρώμα (διάνυσμα ...colortext) .τον αριθμό αναγνώρισης (διάνυσμα ...ids) και το εμβαδόν (διάνυσμα ...emvadon) του γεωμετρικού σχήματος .

```
if ~isempty(trianbboxes)
    ids = int32(trids);
    labels1 = cellstr(int2str(ids'));
```

Για να τρέξουν οι παρακάτω συναρτήσεις (strcat και insertObjectAnnotation) πρεπει οι πίνακες ή τα διανύσματα να έχουν ίδιες γραμμές .Το πρόβλημα είναι ότι το διάνυσμα trcolortext είναι ένα διάνυσμα στηλών ,εναντιθέση με τα διανύσματα ...ids .Κάθε γραμμή του πίνακα ...bboxes ,περιέχει τις συντεταγμένες του ορθογωνίου που περιβάλει το γεωμετρικό σχήμα .Άρα δεν έχουμε πρόβλημα με τον πίνακα αυτό .Άρα για να λύσουμε το πρόβλημα αυτό πρέπει τις στήλες του διανύσματος trcolortext να τις κάνουμε γραμμές .Αυτό κάνουμε παρακάτω .Αντί για την παρακάτω σύνταξη (') ,μπορούμε να χρησιμοποιήσουμε την συνάρτηση transpose .

```
    labels2 = trcolortext.';

    tremvadon = int32(tremvadon);
    labels3 = cellstr(int2str(tremvadon'));
```

Για να τρέξει η παρακάτω εντολή (strcat) πρεπει οι πίνακες labels1 ,labels2 και labels3 να έχουν

ίδιες γραμμές .

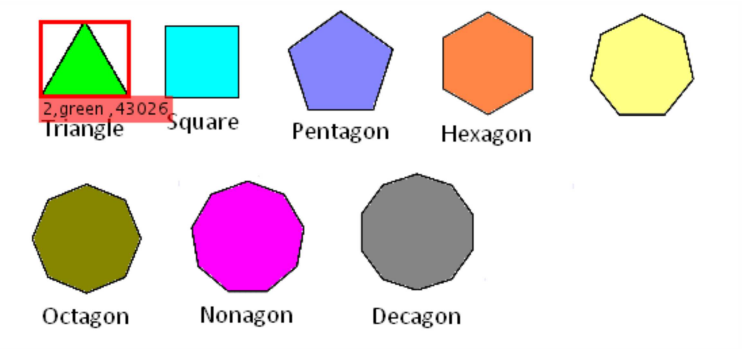
```
    labels = strcat(labels1,labels2,labels3);
```

Αντίστοιχα για να τρέξει η παρακάτω εντολή (insertObjectAnnotation) πρεπει οι πίνακες trianbboxes (POSITION) και labels (LABEL) να έχουν ίδιες γραμμές .

```
    triangles = insertObjectAnnotation(padImageRead,'rectangle',trianbboxes,...
        labels,'Linewidth',3,'Color','red');
```

Παρακάτω φαίνεται μια άλλη συνάρτηση (insertShape) ,που μπορεί να χρησιμοποιηθεί για να ζωγραφιστούν διάφορα γεωμετρικά σχήματα σε ένα σημείο της εικόνας .Η συνάρτηση insertShape δεν δίνει την επιλογή να γραφτούν σχόλια) .

```
    %
    % triangles =
insertShape(padImageRead,'Rectangle',trianbboxes,'Linewidth',3,...
    %     'Color','red');

    figure;
    imshow(triangles); title('trinagles')
```




```
end
if ~isempty(recbboxes)
    ids = int32(recids);
    labels1 = cellstr(int2str(ids'));
```

Για να τρέξουν οι παρακάτω συναρτήσεις (strcat και insertObjectAnnotation) πρεπει οι πίνακες ή τα διανύσματα να έχουν ίδιες γραμμές .Το πρόβλημα είναι ότι το διάνυσμα trcolortext είναι ένα διάνυσμα στηλών ,εναντιθέση με τα διανύσματα ...ids .Κάθε γραμμή του πίνακα ...bboxes ,περιέχει τις συντεταγμένες του ορθογωνίου που περιβάλει το γεωμετρικό σχήμα .Άρα δεν έχουμε πρόβλημα με τον πίνακα αυτό .Άρα για να λύσουμε το πρόβλημα αυτό πρέπει τις στήλες του διανύσματος trcolortext να τις κάνουμε γραμμές .Αυτό κάνουμε παρακάτω .Αντί για την παρακάτω σύνταξη (') ,μπορούμε να χρησιμοποιήσουμε την συνάρτηση transpose .

```
    labels2 = reccolortext.';

    recemvadon = int32(recemvadon);
    labels3 = cellstr(int2str(recemvadon'));
```

Για να τρέξει η παρακάτω εντολή (strcat) πρεπει οι πίνακες labels1 ,labels2 και labels3 να έχουν

ίδιες γραμμές .

```
    labels = strcat(labels1,labels2,labels3);
```

Αντίστοιχα για να τρέξει η παρακάτω εντολή (insertObjectAnnotation) πρεπει οι πίνακες recbboxes (POSITION) και labels (LABEL) να έχουν ίδιες γραμμές .

```
    rectangles = insertObjectAnnotation(padImageRead,'rectangle',recbboxes,...
        labels,'Linewidth',3,'Color','red');
```

Παρακάτω φαίνεται μια άλλη συνάρτηση (insertShape) ,που μπορεί να χρησιμοποιηθεί για να ζωγραφιστούν διάφορα γεωμετρικά σχήματα σε ένα σημείο της εικόνας .Η συνάρτηση insertShape δεν δίνει την επιλογή να γραφτούν σχόλια) .

```
    %
    % rectangles =
insertShape(padImageRead,'Rectangle',recbboxes ,'Linewidth',3,...
    %     'Color','red');

    figure;
    imshow(rectangles); title('rectangles')
```


```
end
if ~isempty(polbboxes)
    ids = int32(polids);
    labels1 = cellstr(int2str(ids'));
```

Για να τρέξουν οι παρακάτω συναρτήσεις (strcat και insertObjectAnnotation) πρεπει οι πίνακες ή τα διανύσματα να έχουν ίδιες γραμμές .Το πρόβλημα είναι ότι το διάνυσμα trcolortext είναι ένα διάνυσμα στηλών ,εναντιθέση με τα διανύσματα ...ids .Κάθε γραμμή του πίνακα ...bboxes ,περιέχει τις συντεταγμένες του ορθογωνίου που περιβάλει το γεωμετρικό σχήμα .Άρα δεν έχουμε πρόβλημα με τον πίνακα αυτό .Άρα για να λύσουμε το πρόβλημα αυτό πρέπει τις στήλες του διανύσματος trcolortext να τις κάνουμε γραμμές .Αυτό κάνουμε παρακάτω .Αντί για την παρακάτω σύνταξη (') ,μπορούμε να χρησιμοποιήσουμε την συνάρτηση transpose .

```
    labels2 = polcolortext.';
```

Για να τρέξει η παρακάτω εντολή (strcat) πρεπει οι πίνακες labels1 και labels2 να έχουν ίδιες γραμμές .

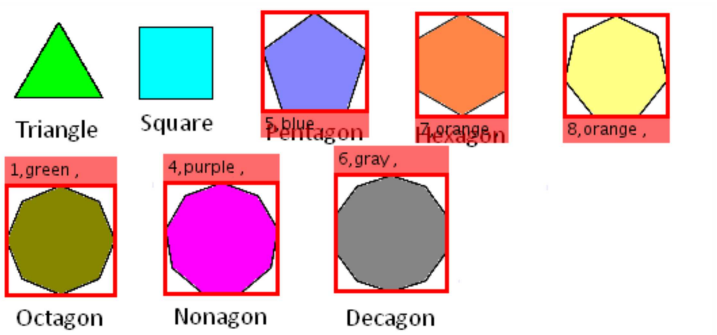
```
    labels = strcat(labels1,labels2);
```

Αντίστοιχα για να τρέξει η παρακάτω εντολή (insertObjectAnnotation) πρεπει οι πίνακες polbboxes (POSITION) και labels (LABEL) να έχουν ίδιες γραμμές .

```
    polygons = insertObjectAnnotation(padImageRead,'rectangle',polbboxes,...
        labels,'Linewidth',3,'Color','red');
```

Παρακάτω φαίνεται μια άλλη συνάρτηση (insertShape) ,που μπορεί να χρησιμοποιηθεί για να ζωγραφιστούν διάφορα γεωμετρικά σχήματα σε ένα σημείο της εικόνας .Η συνάρτηση insertShape δεν δίνει την επιλογή να γραφτούν σχόλια) .

```
    %
    % polygons =
insertShape(padImageRead,'Rectangle',polbboxes,'Linewidth',3,...
    %     'Color','red');

    figure;
    imshow(polygons); title('polygons')
```




```
end
if ~isempty(circle)
    ids = int32(cirids);
    labels1 = cellstr(int2str(ids'));
```

Για να τρέξουν οι παρακάτω συναρτήσεις (strcat και insertObjectAnnotation) πρεπει οι πίνακες ή τα διανύσματα να έχουν ίδιες γραμμές .Το πρόβλημα είναι ότι το διάνυσμα trcolortext είναι ένα διάνυσμα στηλών ,εναντιθέση με τα διανύσματα ...ids .Κάθε γραμμή του πίνακα ...bboxes ,περιέχει τις συντεταγμένες του ορθογωνίου που περιβάλει το γεωμετρικό σχήμα .Άρα δεν έχουμε πρόβλημα με τον πίνακα αυτό .Άρα για να λύσουμε το πρόβλημα αυτό πρέπει τις στήλες του διανύσματος trcolortext να τις κάνουμε γραμμές .Αυτό κάνουμε παρακάτω .Αντί για την παρακάτω σύνταξη (') ,μπορούμε να χρησιμοποιήσουμε την συνάρτηση transpose .

```
    labels2 = circolortext.';

    ciremvadon = int32(ciremvadon);
    labels3 = cellstr(int2str(ciremvadon'));
```

Για να τρέξει η παρακάτω εντολή (strcat) πρεπει οι πίνακες labels1 ,labels2 και labels3 να έχουν ίδιες γραμμές .

```
    labels = strcat(labels1,labels2,labels3);
```

Αντίστοιχα για να τρέξει η παρακάτω εντολή (insertObjectAnnotation) πρεπει οι πίνακες circle (POSITION) και labels (LABEL) να έχουν ίδιες γραμμές .

```
    circles = insertObjectAnnotation(padImageRead,'circle',circle,...
        labels,'Linewidth',3,'Color','red');
```

Παρακάτω φαίνεται μια άλλη συνάρτηση (insertShape) ,που μπορεί να χρησιμοποιηθεί για να ζωγραφιστούν διάφορα γεωμετρικά σχήματα σε ένα σημείο της εικόνας .Η συνάρτηση insertShape δεν δίνει την επιλογή να γραφτούν σχόλια) .

```
    %
    % circles = insertShape(padImageRead,'Circle',circle,'Linewidth',3,...
    %     'Color','red');

    figure;
    imshow(circles); title('circles')
end
```

Παρακάτω εμφανίζονται οι γραμμές που εντοπίστηκαν στην εικόνα ,με χρήση δυο μεθόδων ,μία με βάση τα χαρακτηριστικά της περιοχής των γραμμών αυτών και μία με hough transform .

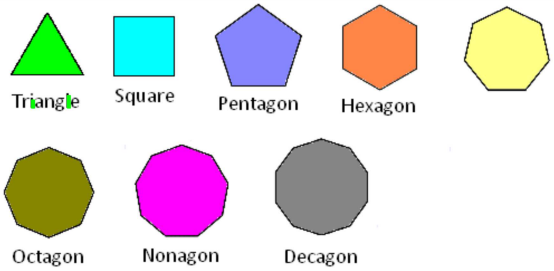
Να επισημανθεί ότι η γραμμές με hough transform εντοπίστηκαν από περιοχές του αντικειμένου STATS1 που υπολογίστηκαν ότι περιέχουν γραμμές .Αυτό γίνεται για να δούμε εάν οι δυο μέθοδοι εμφανίζουν τα ίδια αποτελέσματα σε απλές γραμμές και όχι σε σε σχήματα ,Όπως θα δούμε στις παρακάτω εικόνες εμφανίζουν τα ίδια αποτελέσματα .

Να πούμε επίσης ότι η εικόνα που εμφανίζει τις γραμμές που εντοπίστηκαν με hough transform ,δεν εμφανίζεται εδώ κανονικά ,αλλά μεταφέρθηκε σε αυτό το κεφάλαιο για να συγκριθεί με τις γραμμές που εντοπίστηκαν με βάση τα χαρακτηριστικά της περιοχής τους ,δηλαδή με την άλλη μέθοδο .

```
if ~isempty(lineRegionsPixels)
   figure,
   imshow(padImageRead),title('lines') ,hold on
   for w = 1:length(lineRegionsPixels)
```

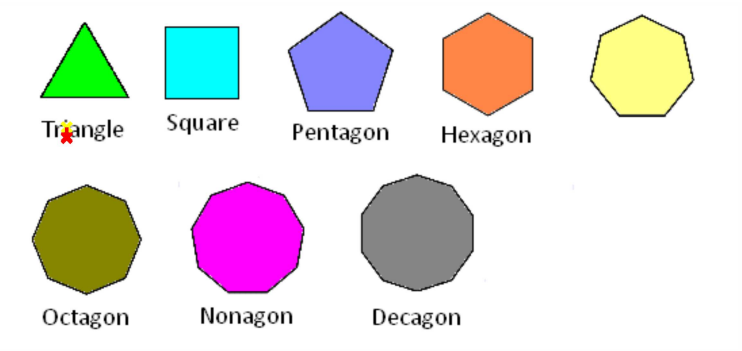
Με τη συνάρτηση plot παρακάτω ,ζωγραφίζουμε τα σημεία κάθε γραμμής που έχει κάθε περιοχή του αντικειμένου STATS1 .Όλα τα σημεία ,όλων των γραμμών (συγκεκριμένα οι συντεταγμένες τους) είναι καταχωρημένες στη δομή lineRegionsPixels .Κάθε αντικείμενο της δομής lineRegionsPixels περιέχει τις συντεταγμένες των γραμμών για μία περιοχή .

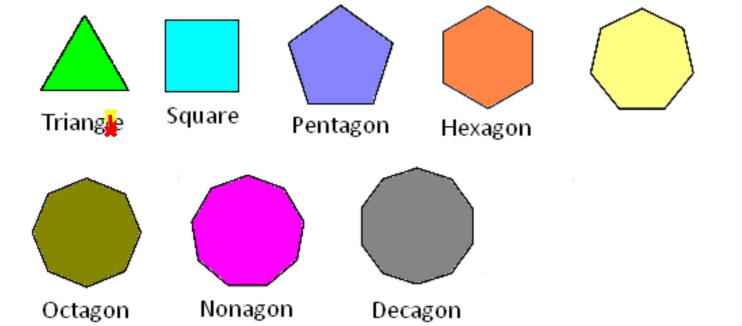
```
      plot(lineRegionsPixels(w).pixels(:,1),lineRegionsPixels(w).pixels(:,2),...
          'LineWidth',2,'Color','green');
```



```
    end
end
```

Παρακάτω φαίνονται οι γραμμές που εντοπίστηκαν με hough transform στις περιοχές των αντικειμένων STATS1 και STATS2 .

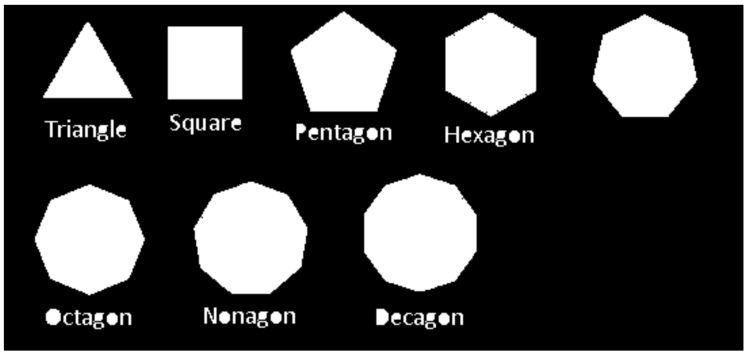


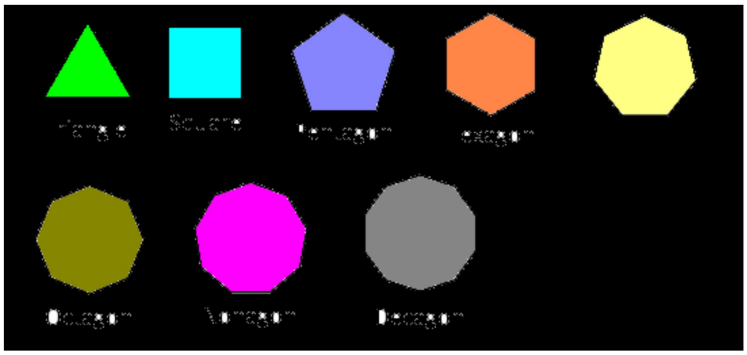


Όπως παρατηρείται ,τα αποτελέσματα ανίχνευσης γραμμών ,πρώτων με hough transform και δεύτερων με βάση τα χαρακτηριστικά των περιοχών που έχει υπολογίσει η συνάρτηση regionprops και που περιέχουν γραμμές δεν είναι ίδια .Καλύτερο αποτέλεσμα έχουμε με hough transform .Ο hough transform ξεχωρίζει τις γραμμές ακόμα και όταν αυτές είναι ενωμένες .

Οι γραμμές που μπορούμε να εντοπιστούν με hough transform μπορεί να είναι περισσότερες εάν εφαρμοστεί ο hough transform σε περισσότερα πλαίσια του αντικειμένου STATS1 ή και σε όλη την εικόνα .

Επίσης μια άλλη παρατήρηση που πρέπει να γίνει είναι οι περιοχές που περιέχονται στα αντικείμενα STATS1 και STATS2 .Παρατηρείται ,από τις εικόνες hough transform ,ότι τα αντικείμενα αυτά περιέχουν περιοχές ,όπου αυτές περιέχουν γράμματα !Για να παρατηρηθεί καλύτερα ,παρακάτω εμφανίζονται δυο εικόνες (μία άχρωμη και μία έχρωμη) που περιέχουν όλες τις περιοχές των αντικειμένων STATS1 και STATS2 .Αυτό απαιτεί μια αλλαγή στο πρόγραμμα .





Εν κατακλείδι ,με βάση τα χαρακτηριστικά που έχει ένα αντικείμενο στην περιοχή του ,μπορούμε να αναγνωριστούν πολλά και διάφορα αντικείμενα ,όπως τα γράμματα .Δηλαδή γενικά ,χρειάζονται κάποιες μέθοδοι επεξεργασίας της εικόνας για να διαγραφούν τυχόν αλλοιώσεις στην εικόνα ,ένας συγκεκριμένος τρόπος ανίχνευσης αντικειμένων (όπως η συνάρτηση bwboundaries) ,και η διαφοροποίησης τους με βάση τα χαρακτηριστικά τους .

ΚΕΦΑΛΑΙΟ 5 : Μέθοδος k-means

Εισαγωγή κεφαλαίου

Στο παρών κεφάλαιο αναλύεται η τεχνική τμηματοποίησης της εικόνας ,με βάση τα χρώματά της .Η τεχνική ομαδοποίησης των χρωμάτων αυτή λέγεται K-means .

Ο τρόπος αυτός ,δεν είναι και τόσο αυτοματοποιημένος ,καθώς πρέπει να ορίσουμε σε πόσα χρώματα θέλουμε να χωρίσουμε την εικόνα ,αλλά είναι πιο σύντομος .Επισης με τον παρακάτω τρόπο χωρίζουμε όλη την εικόνα και όχι μόνο τα πλαίσια που υπάρχουν στο αντικείμενο STATS1 .

Μπορεί να γίνει και μια μίξη των δύο αυτών μεθόδων και να διαχωρίσουμε την εικόνα ,με τη μέθοδο K-means ,σε τόσα χρώματα όσα ανιχνεύτηκαν στο κεφάλαιο 3 συν ένα για το φόντο .

Για την τμηματοποίηση της εικόνας με βάση τα χρώματα που αυτή έχει ,πρέπει να γίνει παρακολούθηση των χρωμάτων (κόκκινο ,κίτρινο και μπλε) που έχει κάθε pixel της εικόνας .Έπειτα αφού αναλύσουμε το χρώμα των pixels της εικόνας ,πρέπει να γίνει μια ομαδοποίηση των pixels με βάση το χρώμα τους .Δηλαδή να έχουμε στο τέλος ξεχωρίσει τα pixels της εικόνας με βάση το χρώμα τους .Αυτή τη διαδικασία γίνεται από τη συνάρτηση kmeans .

Η συνάρτηση kmeans και όπως είναι λογικό ,είναι πιο εύκολο να διαχωρίσει τα χρώματα άμα αυτά αποτελούνται από δύο τιμές και όχι από τρεις .Για αυτό ο λόγο θα μετατρέψουμε την RGB εικόνα σε μια Lab εικόνα .Μια εικόνα RGB αναλύει το χρώμα σε τρεις τιμές (κόκκινο ,κίτρινο και μπλε) .Μετατρέποντας μια εικόνα RGB σε μια εικόνα Lab ,το χρώμα αναλύεται σε δύο τιμές (a και b) .

Τέλος για καλύτερα οπτικά αποτελέσματα θα εμφανίσουμε τόσες εικόνες όσες και τα χρώματα που διαφοροποιήθηκαν και που θα περιέχουν τα pixels με το χρώμα τους .

Κώδικας

```
%%
```

Η συνάρτηση makecform δημιουργεί μια δομή χρωματικού μετασχηματισμού (cform) .Ο χρωματικός μετασχηματισμός της εικόνας γίνεται από τη συνάρτηση applycform .Οι μετασχηματισμοί που μπορούμε να κάνουμε με τη συνάρτηση αυτοί είναι πολλοί .Η μετατροπή που θα κάνουμε εμείς είναι ,η μετατροπή μιας εικόνας RGB ,δηλαδή μια εικόνα οπου τα χρώματά της αποτελούνται από συνδυασμό 3 χρωμάτων ,των κόκκινο ,πράσινο και μπλε σε μια εικόνα L*a*b .

Μια εικόνα L*a*b απότελείται από το πρώτο επίπεδο L όπου αυτό περιέχει την φωτεινότηττα κάθε σημείου της εικόνας .Επίσης αποτελείται από το δεύτερο επίπεδο a όπου αυτό δείχνει το χρώμα κάθε σημείου της εικόνας από τι συνδυασμό κόκκινου-πράσινου χρώματος αποτελείται .Τέλος αποτελείται από το τρίτο επίπεδο b όπου αυτό δείχνει το χρώμα κάθε σημείου της εικόνας από τι συνδυασμό μπλε-κίτρινου χρώματος αποτελείται .

Δηλαδή το χρώμα κάθε σημείου της εικόνας L*a*b αποτελείται από συνδυασμό των επιπέδων a και b ,εναντιθέση με μια RGB εικόνα που το χρώμα κάθε σημείου της εικονας αποτελείται από 3 επίπεδα αλλά επίσης κάθε επίπεδο είναι ένα χρώμα ξεχωριστά και όχι δύο ,όπως την L*a*b εικόνα .

Τη μετατροπή αυτή ,από RGB σε L*a*b εικόνα ,την κάνουμε γιατί η χρωματική διάταξη L*a*b μας επιτρέπει να ξεχωρίσουμε τα χρώματα της εικόνας ,με βάση δύο επιπέδων χρώματος .

```
cform = makecform('srgb2lab');
lab_he = applycform(padImageRead,cform);

figure;
imshow(lab_he); title('original image')

%%
```

Παρακάτω θα ξεχωρίσουμε τα χρώματα της εικόνας με χρήση της συνάρτησης kmeans .

'Αρα για αρχή πρέπει να πάρουμε τις πληροφορίες που έχουμε για τα χρώματα της εικόνας .Τις πληροφορίες αυτές σε μια L*a*b εικόνα υπάρχουν στο επίπεδο a και b ,δηλαδή στο δεύτερο και τρίτο επίπεδο (lab_he(:,:,2:3)) .Για αυτό θα δημιουργήσουμε έναν πίνακα (ab) που θα περιέχει για κάθε pixel της εικόνας (nrows*ncols) τα a και b επίπεδα ,δηλαδή το χρώμα του .

Έπειτα η συνάρτηση kmeans θα ομαδοποιήσει τα pixel σε όσες ομάδες ζητήσουμε (nColors = 3) .Σε ποια ομάδα χρώματος ανήκει κάθε pixel i (1 ,2 ή 3) ,το δείχνει το δάνυσμα cluster_idx ,που παίρνουμε από τη συνάρτηση kmeans .Η τιμή 0 (μαύρο χρώμα) δεν ανήκει σε κάποια ομάδα χρώματος και αντιστοιχεί στο φόντο της εικόνας .Το στοιχείο i του διανύσματος cluster_idx ,αντιστοιχεί στο pixel i του πίνακα ab .Να επισυμάνουμε ότι η κάθε γραμμή του πίνακα ab αντιστοιχεί σε ένα pixel .

```
ab = double(lab_he(:,:,2:3));

nrows = size(ab,1);

ncols = size(ab,2);

ab = reshape(ab,nrows*ncols,2);

nColors = 3;

[cluster_idx] = kmeans(ab,nColors,'distance','sqEuclidean', ...
                                    'Replicates',3);

%%
```

Παρακάτω θα δούμε τις 'περιοχές' χρωμάτων έτσι όπως τις έχει ξεχωρίσει η συνάρτηση kmeans .Πριν εμφανίσουμε την εικόνα με τα pixel ομαδοποιημένα με βάση το χρώμα τους , θα πρέπει να 'δημιουργήσουμε' την εικονα αυτήν .Για αυτό θα χρησιμοποιήσουμε τη συνάρτηση reshape για να δημιουργήσουμε μια εικονα με τις ίδιες διαστάσεις με την αρχική (nrows,ncols) ,και έπειτα θα την εμφανίσουμε .Η εικόνα θα εμφανιστεί ως εξής .Τα pixels με την μικρότερη τιμή ,δηλαδή του φόντου ,θα έχουν μαύρο χρώμα .Τα pixels με την μεγαλύτερη τιμή ,δηλαδή 3 (3 γιατί σε τόσες ομάδες έχουμε επιλέξει να χωριστούν τα χρώματα της εικόνας) ,θα έχουν άσπρο χρώμα .Τα pixels με ενδιάμεση τιμή θα έχουν γκρι χρώμα .Με αυτόν τον τρόπο γίνεται αντιλιπτό τις ομάδες που έχει δημιουργήσει η συνάρτηση kmeans .

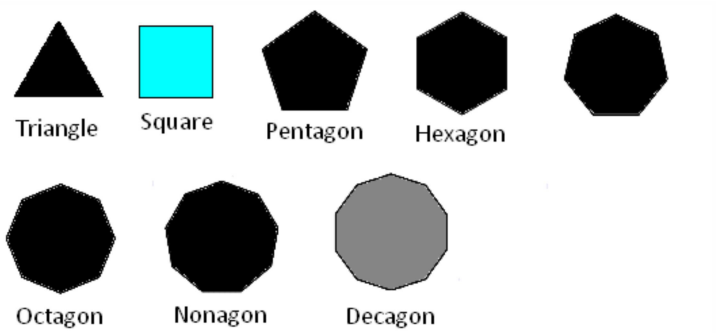
```
pixel_labels = reshape(cluster_idx,nrows,ncols);
```

```
imshow(pixel_labels,[]), title('image labeled by cluster index');
```

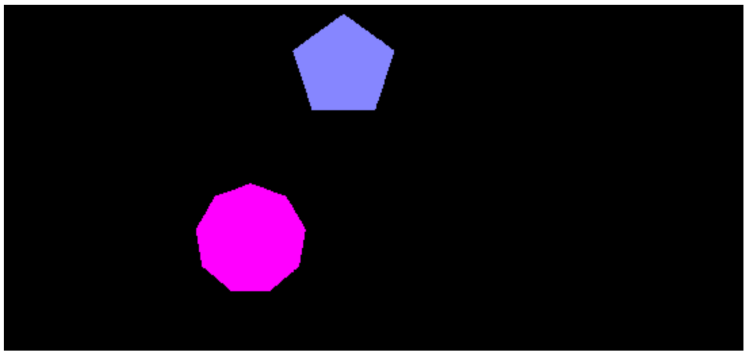
%%

Παρακάτω θα ξεχωρίσουμε τα χρώματα της αρχικής εικόνας (color = he) ,με βάση τις ομάδες που έχει δημιουργήσει η συνάρτηση kmeans .Δηλαδή από την αρχική είκονα ,τα pixels που δεν ανήκουν στην συγκεκριμένη ομάδα (k) θα 'βάφονται' με μαύρο χρώμα ,όπως το φόντο της εικόνας (color(rgb_label ~= k) = 0;) .Η εικόνα που προκύπτει την αποθηκεύουμε σε ένα κελί (segmented_images) και έπειτα θα τις δείξουμε .

```
segmented_images = cell(1,3);
rgb_label = repmat(pixel_labels,[1 1 3]);

for k = 1:nColors
    color = padImageRead;
    color(rgb_label ~= k) = 0;
    segmented_images{k} = color;
end

figure;
imshow(segmented_images{1}), title('objects in cluster 1');
```




```
figure;
imshow(segmented_images{2}), title('objects in cluster 2');
```



```
figure;
imshow(segmented_images{3}), title('objects in cluster 3');
```

Όσες πιο πολλές ομάδες δημιουργηθούν (μεταβλητή nColors) ,τόσο θα μεμονωθούν τα σχήματα που δείχνουν οι παραπάνω εικόνες .

ΑΝΑΦΟΡΕΣ

1. J.R. Parker, '*Algorithms for image Processing and Computer Vision 2nd edition*', Wiley Publication Inc, USA, 2011

2. Rafael C. Gonzales, Richard E. Woods, '*Ψηφιακή Επεξεργασία Εικόνας*', Εκδόσεις Τζιόλα, Θεσσαλονίκη, 2014

3. William K. Pratt, '*Digital Image Processing 3rd Edition*', John Wiley & Sons Inc, USA, 2011

4. http://www.mathworks.com/help/vision/getting-started-with-computer-vision-system-toolbox.html προσπέλαση στις 10/5/2017

5. https://www.mathworks.com/discovery/image-segmentation.html προσπέλαση στις 10/5/2017

6. https://www.mathworks.com/help/images/hough-transform.html#buh9ylp-26 προσπέλαση στις 10/5/2017

7. https://www.mathworks.com/matlabcentral/answers/110855-how-to-detect-the-shape-in-matlab προσπέλαση στις 10/5/2017