ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
Δ.Π.Μ.Σ. «ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΙΣΜΟΥ»
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΤΩΝ ΚΑΤΕΡΓΑΣΙΩΝ

Μεταπτυχιακή Εργασία

Θέμα

Αυτόματη ταυτοποίηση και προσδιορισμός θέσης μηχανουργικών τεμαχίων με τη βοήθεια μηχανικής όρασης

Παύλος Α. Παρασκευάς

Επιβλέπων: Γεώργιος-Χριστόφορος. Βοσνιάκος,
Καθηγητής Ε.Μ.Π

Αθήνα, Απρίλιος 2018

Περίληψη

Η παρούσα μεταπτυχιακή εργασία πραγματεύεται τη μελέτη και την ανάλυση των μεθόδων αναγνώρισης και εύρεσης αντικειμένων σε εικόνα μέσω μεθόδων μηχανικής όρασης.

Απώτερος σκοπός της εργασίας είναι η επιλογή και υλοποίηση της κατάλληλης μεθόδου για την αναγνώριση μηχανουργικών αντικειμένων τοποθετημένα σε ένα επίπεδο εργασίας. Τα μηχανουργικά αντικείμενα που εξετάζονται δεν είναι «επίπεδα», καθώς όλες οι πλευρές είναι συγκρίσιμες μεταξύ τους. Στο επίπεδο εργασίας η πλευρά τοποθέτησης των αντικειμένων είναι τυχαία. Μετά την επιτυχή αναγνώριση πρέπει να εξάγονται οι συντεταγμένες και ο προσανατολισμός του εκάστοτε αντικειμένου με σκοπό την παραλαβή του από έναν ρομποτικό βραχίονα για περαιτέρω επεξεργασία.

Αρχικά παρουσιάζονται οι μέθοδοι αναγνώρισης που θα χρησιμοποιηθούν και δίνεται ένα μαθηματικό και θεωρητικό υπόβαθρο της καθεμιάς. Με αυτόν τον τρόπο γίνονται σαφή τα πλεονεκτήματα και μειονεκτήματα κάθε μεθόδου, καθώς επίσης οι δυνατότητες και περιορισμοί τους. Συγκεκριμένα, αναλύονται μέθοδοι αναγνώρισης μέσω γεωμετρικών μετασχηματισμών, μέθοδοι αναγνώρισης μοτίβου καθώς και ανιχνευτές χαρακτηριστικών.

Εν συνεχεία γίνεται μια εφαρμογή των μεθόδων δυαδικής επεξεργασίας εικόνας για την περίπτωση που εξετάζεται, με σκοπό την απομόνωση ενός αντικειμένου από ένα πλήθος αντικειμένων σε μια εικόνα και την εξαγωγή σημαντικών μεγεθών. Η επεξεργασία έγινε με τη χρήση του λογισμικού Matlab. Η διαδικασία της απομόνωσης ενός αντικειμένου είναι σημαντική καθώς θα επιτρέψει τη σύγκριση του απομονωμένου αντικειμένου της εικόνας με ένα αντικείμενο γνωστό και καθορισμένο εξαρχής που βρίσκεται σε μια βάση δεδομένων.

Εξετάζονται δύο διατάξεις λήψης εικόνας, η πρώτη μέσω κάμερας και η δεύτερη μέσω σκάνερ. Στην πρώτη τα αντικείμενα τοποθετούνται σε ένα μονόχρωμο επίπεδο εργασίας υψηλής χρωματικής αντίθεσης, ενώ στη δεύτερη οι εικόνες λαμβάνονται απευθείας από το επίπεδο σάρωσης. Ακολουθεί η υλοποίηση των μεθόδων, που προαναφέρθηκαν, στο λογισμικό Matlab, και η εξαγωγή αποτελεσμάτων σχετικά με την αξιοπιστία και την ταχύτητα εκτέλεσης του προγράμματος. Τέλος, αφού γίνει μια αξιολόγηση των αποτελεσμάτων επιλέγεται η μέθοδος που θα χρησιμοποιηθεί.

Abstract

The present postgraduate thesis deals with the study and analysis of object detection and matching in image through machine vision techniques.

The utter purpose of the current thesis is the choice and implementation of the appropriate method for the recognition of mechanical objects that are placed on a plane surface. The objects used are not considered as «flat», as every side is comparable to the others. The side that the object is placed on the surface is random. After the successful detection and recognition of the object, its coordinates and orientation should be exported by the system, in order to be received by a robot arm for further processing.

At first, there is a presentation of the recognition methods that are going to be used. A theoretical and mathematical background is given for each one. In this way, the advantages and disadvantages as well as the capabilities and restrictions of each method become clearer. In particular the methods that are studied are methods of matching through geometrical transformations, template matching methods, as well as feature detectors.

Furthermore, an object in the image is isolated among plenty of others through binary image processing. The processing was performed using the Matlab software. The isolation process is significant, because it will allow the comparison of the isolated object with another one which is defined in advance stored in a database.

Two ways of image capturing are presented, one using a webcam and a second one using a document scanner. At the former the objects are placed on a single-colored, high contrast plane surface while at the latter the images are taken directly by the scanning surface. Subsequently the Matlab implementations of the aforementioned object recognition methods are presented. For each method there is an evaluation of the reliability, robustness and execution time based on the corresponding results. Finally, the fitting technique for our case is selected.

Ευχαριστίες

Η παρούσα μεταπτυχιακή εργασία εκπονήθηκε στο Εθνικό Μετσόβιο Πολυτεχνείο, στη Σχολή Μηχανολόγων Μηχανικών και συγκεκριμένα στο Εργαστήριο Τεχνολογίας Κατεργασιών υπό την επίβλεψη του καθηγητή κ. Βοσνιάκου, Θα ήθελα να τον ευχαριστήσω θερμά για τη βοήθειά του καθ' όλη την πορεία εκπόνησης της εργασίας και πρωτίστως την κατανόησή του το χρονικό διάστημα που έπρεπε να διακόψω τις σπουδές μου.

Ιδιαίτερες ευχαριστίες θα ήθελα να δώσω στον υποψήφιο διδάκτορα Γιώργο Παπαζέτη καθώς και στα υπόλοιπα μέλη του εργαστηρίου για την απλόχερη υποστήριξη που προσέφεραν, κυρίως κατά την παραμονή μου στους χώρους του εργαστηρίου.

Τέλος, θα ήθελα να ευχαριστήσω θερμά τους φίλους μου Γιώργο και Γιάννη, καθώς χωρίς την βοήθειά τους το τελευταίο διάστημα, θα ήταν πολύ δύσκολη η ολοκλήρωση της εργασίας.

Περιεχόμενα

ΚΕΦΑΛΑΙΟ 3

ΕΥΡΕΣΗ ΑΝΤΙΚΕΙΜΕΝΟΥ ΣΕ ΕΙΚΟΝΑ ΚΑΙ ΛΗΨΗ ΣΗΜΑΝΤΙΚΩΝ ΜΕΓΕΘΩΝ

ΚΕΦΑΛΑΙΟ 4

ΕΠΙΛΟΓΗ ΜΕΘΟΔΟΥ ΑΝΑΓΝΩΡΙΣΗΣ ΑΝΤΙΚΕΙΜΕΝΩΝ - ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

Η μηχανική όραση αναφέρεται στην προσπάθεια αναπαράστασης της λειτουργίας της ανθρώπινης όρασης σε ένα μηχανικό - υπολογιστικό περιβάλλον. Πιο συγκεκριμένα, αναφέρεται στην αυτοματοποιημένη ανάλυση ψηφιακών εικόνων, οι οποίες αναπαριστούν σε δύο διαστάσεις ένα τρισδιάστατο περιβάλλον. Ο όρος μηχανική όραση περικλείει διάφορες χρήσιμες τεχνικές οι οποίες ακόμα και ξεχωριστά είναι αρκετά σημαντικές, όπως επεξεργασία εικόνας, αναγνώριση προτύπων, αλλά κυρίως τεχνικές που αφορούν την περιγραφή σχημάτων και όγκων για γεωμετρικές μοντελοποιήσεις.

1.1 Βιομηχανική μηχανική όραση

Οι αυτοματισμοί στη βιομηχανία εκτελούν απλές επαναλαμβανόμενες εργασίες, οι οποίες περιλαμβάνουν σχετικά λίγα αντικείμενα, τα οποία είναι γνωστά και ορισμένα εκ των προτέρων. Το βιομηχανικό περιβάλλον που λαμβάνουν χώρα οι εργασίες αυτές είναι δομημένο έτσι ώστε να τις απλουστεύει όσο το δυνατόν περισσότερο. [1]

Δεν υπάρχει κανένα βιομηχανικό σύστημα μηχανικής όρασης, το οποίο να λειτουργεί αξιόπιστα για οποιαδήποτε εφαρμογή. Για το σχεδιασμό του συστήματος απαιτείται η εκ των προτέρων γνώση των απαιτήσεων της εκάστοτε εφαρμογής. Τα περισσότερα βιομηχανικά συστήματα μηχανικής όρασης χρησιμοποιούν γνωστά και καθορισμένα εκ των προτέρων αντικείμενα. Το φόντο της εικόνας προς επεξεργασία συνήθως είναι υψηλής χρωματικής αντίθεσης με αυτό των αντικειμένων ώστε να ξεχωρίζουν. Ένας κρίσιμος παράγοντας είναι ο φωτισμός, ο οποίος αναλόγως την εφαρμογή είναι διαφορετικός, τόσο ως προς την τοποθέτηση των φωτιστικών σημείων όσο και ως προς το είδος του (πχ υπέρυθρος φωτισμός). Τέλος, οι περισσότερες εφαρμογές στη βιομηχανία απομονώνουν το αντικείμενο που πρόκειται να αναλυθεί. [2]

Η αξιοπιστία, η αποδοτικότητα αλλά και η ευρωστία ενός συστήματος μηχανικής όρασης βασίζονται στην εξασφάλιση και τη γνώση των παραπάνω συνθηκών.

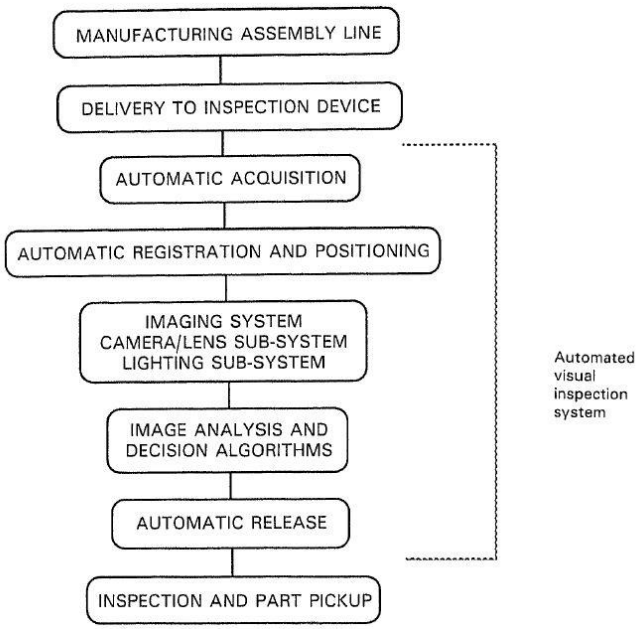
1.2 Εφαρμογές στη βιομηχανία

Στη βιομηχανία οι κύριες εφαρμογές της μηχανικής όρασης είναι οι εξής:

- Διαστασιολόγηση αντικειμένων: Μετρώνται ορισμένες διαστάσεις των αντικειμένων στην εικόνα για να διαπιστωθεί το μέγεθος των αποκλίσεων από τις επιθυμητές.
- Διαχωρισμός αντικειμένων: Ανάλογα την εφαρμογή αναγνωρίζονται αντικείμενα με βάση το χρώμα, το μέγεθος, το σχήμα, την υφή. Το σύστημα διαχωρισμού συγκρίνει τα αντικείμενα απομακρύνει τα ελαττωματικά προϊόντα ή ξένα σώματα. Ο οπτικός διαχωρισμός βρίσκει μεγάλη εφαρμογή στην βιομηχανία τροφίμων, στην φαρμακευτική βιομηχανία, στην βιομηχανία καπνού καθώς και στην ανακύκλωση απορριμμάτων.
- Έλεγχος ποιότητας: Τα αντικείμενα συγκρίνονται με το επιθυμητό για να διαπιστωθεί εάν είναι ελαττωματικά. Για παράδειγμα ελέγχεται αν υπάρχουν φθορές στην επιφάνειά τους, αν υπάρχουν πρόσθετα κομμάτια που δεν έπρεπε να υπάρχουν ή αν λείπει κάποιο κομμάτι.

1.3 Τυπική αρχιτεκτονική συστήματος μηχανικής όρασης

Ένα τυπικό σύστημα μηχανικής όρασης περιλαμβάνει τα εξής επιμέρους συστήματα όπως φαίνονται στο επόμενο διάγραμμα:



Σχήμα 1.1: Επιμέρους συστήματα ενός βιομηχανικού συστήματος μηχανικής όρασης.[1]

Στο Σχήμα 1.1 παρατηρούμε πως ένα σύστημα μηχανικής όρασης περιλαμβάνει το σημείο όπου στοιβάζονται αρχικά τα αντικείμενα, ένα μηχανισμό τροφοδοσίας των αντικειμένων στο σύστημα, ένα μηχανισμό που να παραλαμβάνει το αντικείμενο και να το τοποθετεί, το σύστημα λήψης της εικόνας και ανάλυσής της και τέλος τους μηχανισμούς απελευθέρωσης του αντικειμένου.

Όσον αφορά το σύστημα λήψης εικόνας αυτό περιλαμβάνει τα εξής 4 υποσυστήματα:

- Σύστημα σχηματισμού εικόνας (Image formation system): Παράγει μια αναλογική αναπαράσταση της σκηνής, συνήθως σε μορφή βίντεο. Περιλαμβάνει τις συνθήκες φωτισμού και τα αισθητήρια όργανα (πχ κάμερα).
- Σύστημα λήψης και (προ)-επεξεργασίας εικόνας: Είναι υπεύθυνο για τη λήψη της ψηφιακής εικόνας και την επεξεργασία της για να διευκολύνει την εξαγωγή των απαιτούμενων πληροφοριών.
- Σύστημα ανάλυσης εικόνας: Από την επεξεργασμένη εικόνα του προηγούμενου υποσυστήματος εξάγονται όλες οι απαιτούμενες πληροφορίες ανάλογα με την εφαρμογή.
- Σύστημα επεξήγησης εικόνας: Σε αυτό το υποσύστημα λαμβάνονται οι αποφάσεις, ώστε να συνεχιστεί η διαδικασία.

Ένα τυπικό παράδειγμα φαίνεται στο σχήμα 1.2 το οποίο παρουσιάζει μια μηχανή διαχωρισμού αντικειμένων.

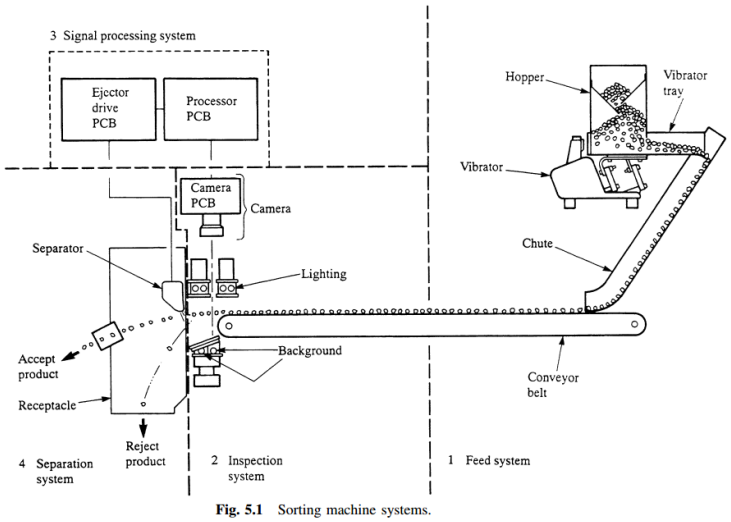


Fig. 5.1 Sorting machine systems.

Σχήμα 1.2: Επιμέρους συστήματα ενός συστήματος οπτικού διαχωρισμού αντικειμένων.

1.4 Διατύπωση του προβλήματος

Όπως αναφέρθηκε, για την δημιουργία ενός συστήματος μηχανικής όρασης απαιτείται ο εκ των προτέρων σαφής προσδιορισμός των απαιτήσεών του καθώς και ορισμένων παραμέτρων.

Στην περίπτωσή μας, γνωρίζουμε το είδος των μηχανουργικών αντικειμένων που θα εξετάζει το σύστημα. Πρόκειται για αντικείμενα, ίδιου χρώματος, των οποίων οι τρεις διαστάσεις είναι συγκρίσιμες μεταξύ τους. Απώτερος σκοπός είναι ο σχεδιασμός ενός συστήματος, που θα εντοπίζει τη θέση και τον προσανατολισμό ενός αντικειμένου ή ίδιων αντικειμένων ανάμεσα στα υπόλοιπα. Στην παρούσα εργασία, δεν θα μελετηθεί ο σχεδιασμός όλων των απαιτούμενων υποσυστημάτων, παρά μόνο ότι αφορά το υποσύστημα λήψης και ανάλυσης της εικόνας. Είναι σαφές, πως ο τρόπος τοποθέτησης και παραλαβής των αντικειμένων δεν μπορεί να είναι τυχαίος, αφού οι παραδοχές που έγιναν κατά την επεξεργασία και την ανάλυση των εικόνων, θέτουν περιορισμούς στο σχεδιασμό και στον τρόπο λειτουργίας των συστημάτων αυτών.

Στην πορεία θα εξετάσουμε διάφορα συστήματα λήψης εικόνας καθώς και διαφορετικούς τρόπους ανάλυσης των εικόνων για την εξαγωγή των απαιτούμενων αποτελεσμάτων.

1.5 Δομή της εργασίας

Στο **κεφάλαιο 2** αναλύονται ορισμένες από τις μεθόδους αναγνώρισης και εύρεσης αντικειμένων σε εικόνα μέσω ψηφιακής επεξεργασίας. Όλες οι μέθοδοι που παρουσιάζονται, θα εξεταστούν σε επόμενο κεφάλαιο για την επίλυση του προβλήματος. Για κάθε μέθοδο που αναφέρεται, ακολουθεί μια συνοπτική ανάλυση του θεωρητικού και μαθηματικού υπόβαθρού της.

Στο **κεφάλαιο 3**, αρχικά θα γίνει μια ανάλυση των τρόπων προβολής ενός αντικειμένου σε εικόνα. Στη συνέχεια περιγράφονται τεχνικές απομόνωσης ενός αντικειμένου στην εικόνα με σκοπό την εξαγωγή σημαντικών μεγεθών, όπως η θέση και ο προσανατολισμός του. Η επεξεργασία των εικόνων γίνεται στο λογισμικό Matlab.

Στο **κεφάλαιο 4**, προτείνονται δύο διατάξεις λήψης εικόνας και παρατίθενται τα αποτελέσματα των μεθόδων που χρησιμοποιήθηκαν. Για κάθε μέθοδο γίνεται μια ποιοτική αξιολόγηση των αποτελεσμάτων, τόσο της επιτυχίας αντιστοίχισης όσο και

του χρόνου εκτέλεσης. Επιλέγεται η καλύτερη μέθοδος αναγνώρισης με κριτήριο την αξιοπιστία, ευρωστία και την ταχύτητα του συστήματος. Με αυτήν εξάγονται οι πραγματικές συντεταγμένες των αντικειμένων της εικόνας, οι οποίες θα χρησιμοποιηθούν από έναν ρομποτικό βραχίονα για τη συλλογή τους.

Στο **κεφάλαιο 5** γίνεται μια ανασκόπηση της εργασίας και εξάγονται ορισμένα συμπεράσματα. Τίθενται, επίσης, προβληματισμοί και στόχοι για μελλοντική εργασία.

Στο **παράρτημα,** τέλος, παρατίθεται ο πηγαίος κώδικας για τη μέθοδο που επιλέχθηκε.

ΚΕΦΑΛΑΙΟ 2

ΜΕΘΟΔΟΙ ΑΝΑΓΝΩΡΙΣΗΣ ΚΑΙ ΕΥΡΕΣΗΣ ΑΝΤΙΚΕΙΜΕΝΩΝ ΜΕΣΩ ΨΗΦΙΑΚΗΣ ΕΠΕΞΕΡΓΑΣΙΑΣ ΕΙΚΟΝΑΣ

Η αναγνώριση αντικειμένων είναι η διαδικασία εύρεσης ενός συγκεκριμένου αντικείμενου σε μια ψηφιακή εικόνα ή βίντεο. Οι άνθρωποι μπορούν να αναγνωρίσουν πολλαπλά αντικείμενα σε μια εικόνα χωρίς μεγάλη προσπάθεια, ακόμη κι αν αυτά απεικονίζονται σε διαφορετικό μέγεθος, θέση, προσανατολισμό ή υπό διαφορετική οπτική γωνία. Η διαδικασία αυτή είναι πολύ πιο απαιτητική όταν πρέπει να εκτελεστεί από ένα τεχνητό σύστημα όρασης. Στο κεφάλαιο αυτό θα μελετηθούν ορισμένες από τις μεθόδους αναγνώρισης αντικειμένων, που έχουν αναπτυχθεί. [3]

2.1 Αναγνώριση αντικειμένων μέσω γεωμετρικών μετασχηματισμών

Όπως θα αναλύσουμε εκτενέστερα στο επόμενο κεφάλαιο, σε μια εικόνα αναπαρίσταται μια δισδιάστατη προβολή του τρισδιάστατου κόσμου. Κάθε αντικείμενο στην εικόνα αποτελείται από ένα συνδυασμό συντεταγμένων x,y που καθορίζουν το σχήμα του. Η λογική λοιπόν αυτής της μεθόδου είναι αρκετά απλή. Αν έχουμε το σχήμα ενός αντικειμένου που θέλουμε να αναγνωρίσουμε και τα σχήματα των αντικειμένων που απεικονίζονται στην εικόνα, τότε μέσω διαφόρων γεωμετρικών μετασχηματισμών προσπαθούμε να τα «ταιριάξουμε».

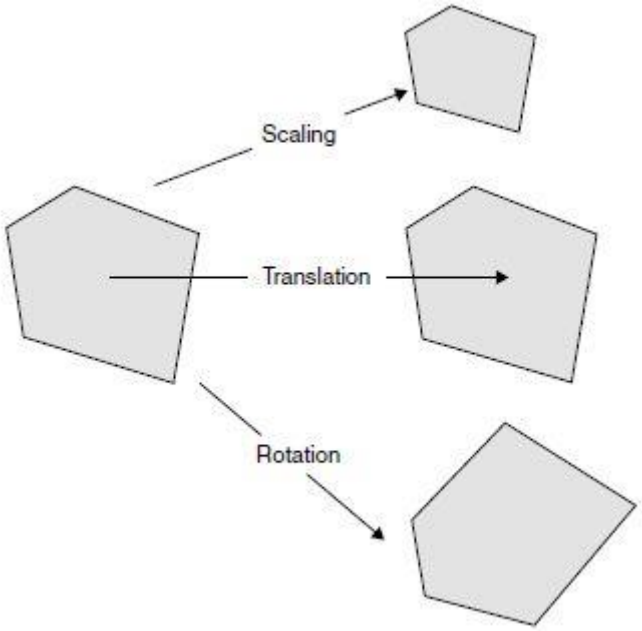
2.1.1 Σχήμα

Η πιο απλή έννοια του σχήματος υπονοεί την ύπαρξη ορίων ή περιγράμματος. Υπάρχουν αντικείμενα που το περίγραμμά τους μόνο μπορεί να αποτελέσει μια ικανοποιητική μαθηματική περιγραφή του σχήματός τους. Υπάρχουν, όμως, πολλές περιπτώσεις που το περίγραμμα δεν αρκεί. Μια τέτοια ενδεικτική περίπτωση είναι το ανθρώπινο πρόσωπο. Αν θέλουμε να διαχωρίσουμε δυο πρόσωπα δεν αρκεί το περίγραμμα του κεφαλιού, αλλά και εσωτερικά περιγράμματα που προκύπτουν από τα μάτια μύτη κτλ. Για την περιγραφή ενός σχήματος απαιτείται ο ορισμός πεπερασμένων σημείων N. Ένα δισδιάστατο σχήμα περιγράφεται από ένα πίνακα της μορφής:

$$\boldsymbol{x} = \begin{bmatrix} x1 & y1 \\ \vdots & \vdots \\ xN & yN \end{bmatrix} \tag{2.1}$$

2.1.2 Μετασχηματισμοί που διατηρούν το σχήμα των αντικειμένων

Η *μεταφορά* ενός αντικειμένου σε μια διαφορετική θέση, η *περιστροφή* του ή η *αλλαγή κλίμακας* είναι διαδικασίες που αλλάζουν τον πίνακα συντεταγμένων του σχήματος, αλλά το σχήμα του αντικειμένου παραμένει το ίδιο. Το γεγονός αυτό απεικονίζεται στο σχήμα 2.1.



Σχήμα 2.1: Το σχήμα του αντικειμένου παραμένει αμετάβλητο από τους γραμμικούς μετασχηματισμούς της αλλαγής κλίμακας, μεταφοράς, περιστροφής.

Μεταφορά: Όπως προαναφέραμε το σχήμα ενός αντικειμένου μπορεί να περιγραφεί από έναν πίνακα της μορφής:

$$S = \begin{bmatrix} x1 & \cdots & xN \\ y1 & \cdots & yN \end{bmatrix} \tag{2.2}$$

Ο μετασχηματισμός της μεταφοράς γίνεται πολύ συχνά, αφού πολλές φορές πρέπει να μετακινήσουμε τα σχήματα ως προς ένα σημείο αναφοράς, για παράδειγμα την αρχή των αξόνων. Ο μετασχηματισμός αυτός επιτυγχάνεται με πρόσθεση της απόστασης μετακίνησης σε κάθε συντεταγμένη. Δηλαδή:

$$S' = S + d \tag{2.3}$$

Περιστροφή/Αλλαγή κλίμακας : Η περιστροφή ή η αλλαγή κλίμακας του σχήματος επιτυγχάνεται με τον πολλαπλασιασμό του πίνακα του σχήματος με έναν πίνακα μετασχηματισμού T.

$$S' = TS \tag{2.4}$$

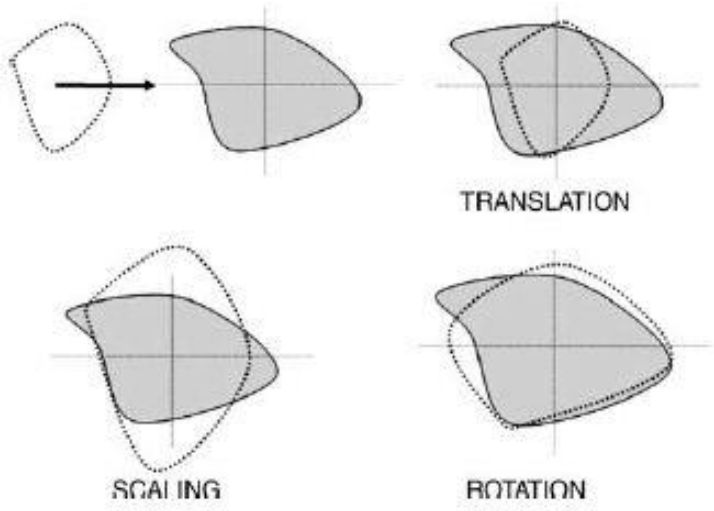
Ο πίνακας μετασχηματισμού Τ για την κάθε περίπτωση είναι:

$$Tsc = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix}, \qquad T\theta = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \tag{2.5}$$

Όπου Tsc ο πίνακας μετασχηματισμού για την αλλαγή κλίμακας με s το ποσοστό αλλαγής και Τθ ο πίνακας μετασχηματισμού για την περιστροφή κατά γωνία θ.[4]

2.1.3 Μέθοδος του Προκρούστη (Procrustes alignment method)

Ενώ είναι δυνατόν ο χρήστης να κάνει όλους τους απαραίτητους γεωμετρικούς μετασχηματισμούς χειροκίνητα, με σκοπό να ταιριάξει δύο σχήματα, πολλές φορές προτιμάται αυτή η διαδικασία να γίνεται αυτόματα. Αυτός είναι ο σκοπός αυτής της μεθόδου και απεικονίζεται στο σχήμα 2.2. Έστω ότι υπάρχουν δύο σχήματα, Α και Β. Μέσω της μεθόδου αυτής, στο σχήμα Α γίνονται οι τρεις γραμμικοί μετασχηματισμοί (μεταφορά, αλλαγή κλίμακας περιστροφή) με σκοπό την ελαχιστοποίηση της ευκλείδειας απόστασης των *αντίστοιχων* σημείων τους.



Σχήμα 2.2: Ταίριασμα αντικειμένων με τη μέθοδο του Προκρούστη.

Έστω λοιπόν ότι:

$$A = \begin{bmatrix} x1 & \cdots & xN \\ y1 & \cdots & yN \end{bmatrix} \quad \text{και} \quad B = \begin{bmatrix} x1' & \cdots & xN' \\ y1' & \cdots & yN' \end{bmatrix} \tag{2.6}$$

i. Αρχικά, πρέπει τα σημεία του Α να μετακινηθούν ως προς συντεταγμένες του Β ώστε να ελαχιστοποιηθεί η απόσταση μεταξύ των αντίστοιχων σημείων τους. Με άλλα λόγια:

$$\vec{x} = \vec{x} + t \tag{2.7}$$

Πρέπει να ελαχιστοποιηθεί η συνάρτηση κόστους:

$$Q = \sum[\vec{x} + t - \vec{x}']^T\,[\vec{x} + t - \vec{x}'] \tag{2.8}$$

Λύνοντας ως προς t, εύκολα δείχνεται πως πρέπει:

$$t = \langle \bar{B} \rangle - \langle \bar{A} \rangle \tag{2.9}$$

Δηλαδή, η διαφορά μεταξύ των κέντρων μάζας μεταξύ των δύο σχημάτων. Συνήθως αναφέρουμε τις συντεταγμένες του B ως προς ένα σημείο αναφοράς, όπως για παράδειγμα την αρχή των αξόνων (0,0). Σε αυτή την περίπτωση, το κέντρο μάζας του σχήματος B θα είναι στο μηδέν, συνεπώς ο μετασχηματισμός θα είναι:

$$t = -\langle \bar{A} \rangle \tag{2.10}$$

ii. Στη συνέχεια πρέπει τα δύο σχήματα να μεταφραστούν ως προς μία ενιαία κλίμακα ώστε να είναι δυνατή η σύγκρισή τους.

$$A' = SA \tag{2.11}$$

Όπου S, ο πίνακας μετασχηματισμού:

$$S = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix} \tag{2.12}$$

Με το s να προκύπτει από:

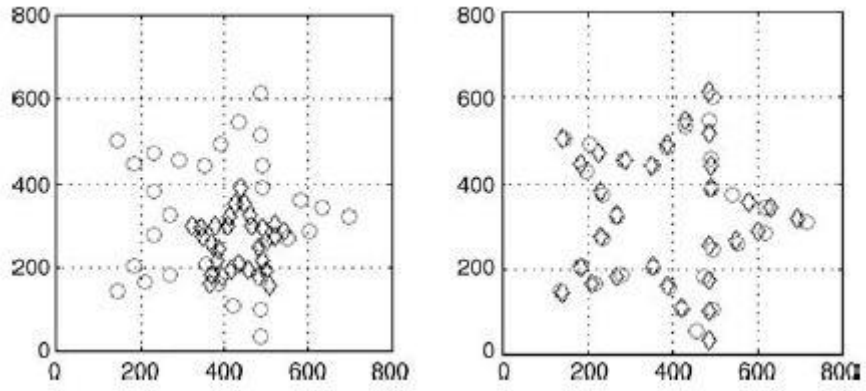
$$s = \frac{\sum \vec{x}'^T \vec{x}}{\sum \vec{x}\vec{x}'^T} \tag{2.13}$$

iii. Το τελευταίο βήμα της διαδικασίας είναι η περιστροφή του σχήματος A, με σκοπό πάντα η ευκλείδεια απόσταση των αντίστοιχων σημείων να ελαχιστοποιείται. Έστω ένας πίνακας σφάλματος E=B-RA (R πίνακας περιστροφής του A). Πρέπει να ελαχιστοποιηθεί το κόστος Q:

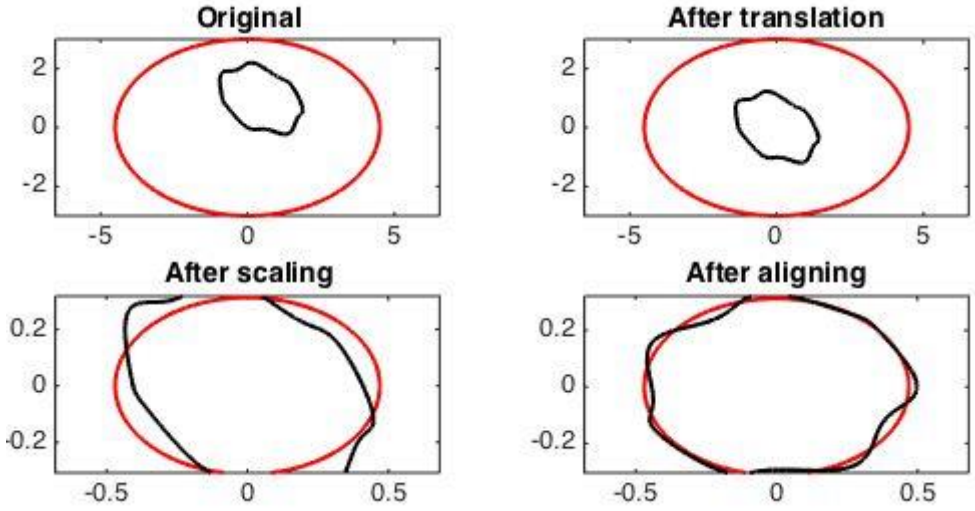
$$Q = trace(E^T E)$$

Αυτό αποδεικνύεται ότι συμβαίνει όταν $R=VU^T$, όπου U,V τα ιδιοδιανύσματα που προκύπτουν από την SVD (singular value decomposition) του πίνακα AB^T.[5]

Εφαρμογές της μεθόδου παρουσιάζονται στα δύο επόμενα σχήματα.



Σχήμα 2.3: Εφαρμογή της μεθόδου του Προκρούστη [4].



Σχήμα 2.4: Εφαρμογή της μεθόδου του Προκρούστη [6].

2.1.4 Ανασκόπηση των μεθόδων αναγνώρισης αντικειμένων μέσω γεωμετρικών μετασχηματισμών

Σε αυτή την παράγραφο μελετήθηκαν μέθοδοι αναγνώρισης αντικειμένων που έχουν ως βάση το γεωμετρικό μετασχηματισμό του περιγράμματος/σχήματος τους. Οι μέθοδοι αυτοί αποτελούν μια πολύ απλή υλοποίηση με ορθά αποτελέσματα, αρκεί να έχουμε δυο περιγράμματα αντικειμένων που επιθυμούμε να συγκρίνουμε. Υπάρχουν όμως και περιορισμοί. Όπως αναφέρθηκε τα περιγράμματα των δυο σχημάτων πρέπει να έχουν τον ίδιο αριθμό σημείων. Επίσης, είναι απαραίτητο οι συντεταγμένες των δύο σχημάτων να είναι αντιστοιχισμένες μεταξύ τους.

Ο πρώτος περιορισμός είναι εύκολο να ξεπεραστεί, με κάποια δειγματοληψία, ο δεύτερος, όμως, είναι πιο περίπλοκος. Ένα σύστημα το οποίο βρίσκει τα σημεία του

περιγράμματος μέσω μιας αυτοματοποιημένης διαδικασίας είναι δύσκολο να τα βρίσκει με μια συγκεκριμένη σειρά. Συνεπώς, πρέπει για την υλοποίηση τέτοιων μεθόδων να υπάρχουν αλγόριθμοι αντιστοίχισης των σημείων του περιγράμματος κάποιοι από τους οποίους θα αναφερθούν στη συνέχεια.

Είναι σαφές, τέλος, πως η χρησιμοποίηση τέτοιων μεθόδων απαιτεί υπολογιστική ισχύ και χρόνο, καθώς όλα τα αντικείμενα πρέπει να συγκριθούν ένα προς ένα με κάθε ένα από τα αντικείμενα ενδιαφέροντος.

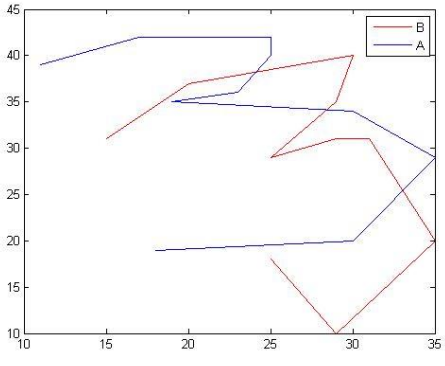
2.1.5 Ουγγρικός αλγόριθμος (Hungarian Algorithm)

Όπως αναφέρθηκε, είναι αναγκαίο με τη χρησιμοποίηση των παραπάνω μεθόδων να υπάρχει ένας αλγόριθμος αντιστοίχισης των σημείων των σχημάτων προς σύγκριση. Ένας τέτοιος είναι ο ουγγρικός αλγόριθμος. Πρόκειται για έναν αλγόριθμο που λύνει προβλήματα ανάθεσης ελαχιστοποιώντας μια συνάρτηση κόστους.([7],[8]) Στην περίπτωσή μας, η συνάρτηση κόστους, θα μπορούσε να είναι η ευκλείδεια απόσταση μεταξύ των σημείων, αλλά τις περισσότερες φορές χρησιμοποιείται μια συνάρτηση κόστους η οποία προκύπτει από τη διαδικασία του «συναφούς σχήματος» (shape context) που θα αναλυθεί παρακάτω. [9]

2.1.6 "Συναφές σχήμα" (Shape Context)

Όπως αναλύσαμε προηγουμένως, το σχήμα ενός αντικειμένου δίνεται από έναν αριθμό συντεταγμένων. Σκοπός της μεθόδου είναι, δεδομένων δυο σχημάτων, να βρίσκονται οι αντιστοιχίσεις των συντεταγμένων τους [10]. Για παράδειγμα, τα παρακάτω σημεία λήφθηκαν από δυο χειρόγραφους αριθμούς, που παρουσιάζονται στο σχήμα 2.5.

$$A = \begin{bmatrix} 11 & 39 \\ 17 & 42 \\ 25 & 42 \\ 25 & 40 \\ 23 & 36 \\ 19 & 35 \\ 30 & 34 \\ 35 & 29 \\ 30 & 20 \\ 18 & 19 \end{bmatrix} \quad B = \begin{bmatrix} 15 & 31 \\ 20 & 37 \\ 30 & 40 \\ 29 & 35 \\ 25 & 29 \\ 29 & 31 \\ 31 & 31 \\ 35 & 20 \\ 29 & 10 \\ 25 & 18 \end{bmatrix}$$



Σχήμα 2.5: Σημεία που σχηματίζουν 2 χειρόγραφους αριθμούς.

Τα σημεία αυτά θα μπορούσαν να είχαν ληφθεί με διαφορετική σειρά. Για παράδειγμα ο πίνακας Α να ήταν γραμμένος ως εξής:

$$A' = \begin{bmatrix} 11 & 39 \\ 25 & 42 \\ 25 & 40 \\ 23 & 36 \\ 30 & 34 \\ 35 & 29 \\ 18 & 19 \\ 17 & 42 \\ 30 & 20 \\ 19 & 35 \end{bmatrix}$$

Τα σημεία στο χώρο είναι τα ίδια, η φορά διαγραφής τους, όμως, διαφορετική. Γνωρίζοντας ότι η επιθυμητή φορά διαγραφής είναι αυτή του πίνακα Β, μέσω της μεθόδου αυτής θα προσπαθήσουμε να αντιστοιχίσουμε τα σημεία του πίνακα Α΄ όσο το δυνατόν καλύτερα με αυτά του Β, ώστε να αναπροσαρμοστεί κατάλληλα η φορά διαγραφής.

Αρχικά θεωρούμε τα διανύσματα όλων των άλλων σημείων του σχήματος ως προς το πρώτο. Όλα αυτά τα διανύσματα εκφράζουν το σχήμα ως προς αυτό το σημείο. Όσο περισσότερα σημεία τόσο καλύτερη αναπαράσταση του σχήματος. Για κάθε σημείο αi του πίνακα Α υπολογίζουμε ένα ιστόγραμμα hi των σχετικών συντεταγμένων των υπόλοιπων σημείων.

$$hi(k) = \#\{q \neq ai) \sqsubseteq bin(k)\} \tag{2.14}$$

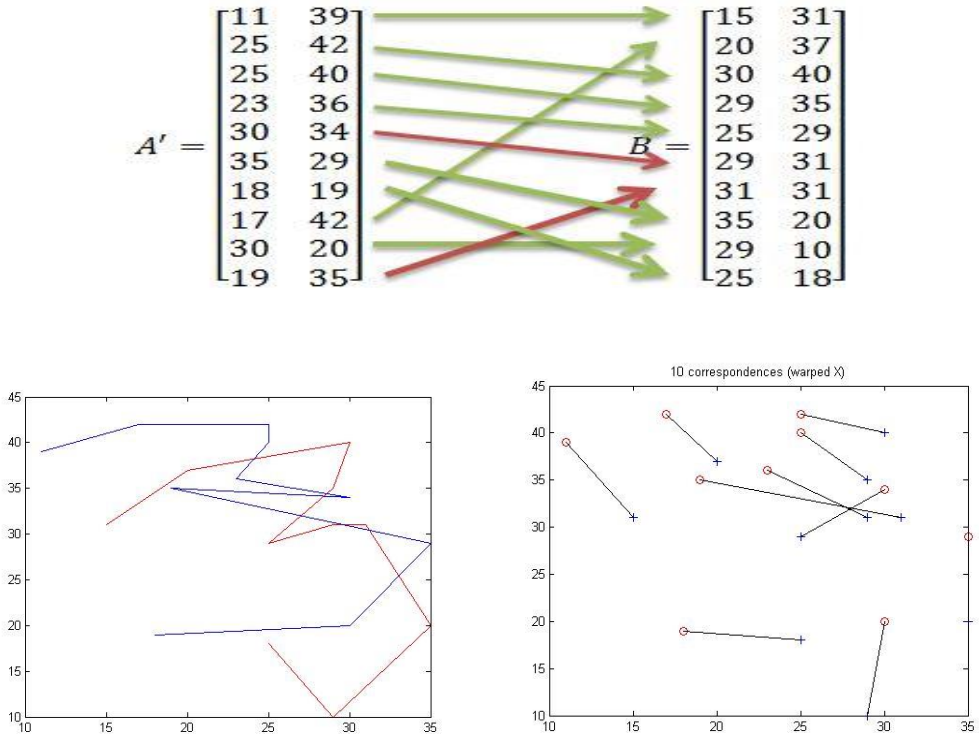
Αυτό το ιστόγραμμα ορίζεται ως το «συναφές σχήμα» (shape context) του σημείου αi. Το ιστόγραμμα πρέπει είναι πιο ευαίσθητο σε κοντινά σημεία, για το λόγο αυτό χρησιμοποιούνται λογαριθμικές πολικές συντεταγμένες. Το κόστος της συνάρτησης αντιστοίχισης ενός σημείου του Α με ένα σημείο του Β, χρησιμοποιώντας την κατανομή χ^2 δίνεται από:

$$Cij = \frac{1}{2}\sum_{k}\frac{[hi(k) - hj(k)]^2}{[hi(k) + hj(k)]} \tag{2.15}$$

Όπου, hi και hj τα ιστογράμματα για τα σημεία αi και bj αντίστοιχα.[11]

Στη συνέχεια μέσω του ουγγρικού αλγόριθμου ελαχιστοποιείται το παραπάνω κόστος αντιστοιχίζοντας έτσι τα σημεία του Α με του Β. Η αντιστοίχιση που έγινε στην περίπτωσή μας φαίνεται στο ακόλουθο σχήμα (σχήμα 2.6). Βλέπουμε πως ένα

σημείο δεν αντιστοιχήθηκε σωστά, αλλά αυτό έχει να κάνει με το γεγονός πως υπήρχαν μόνο 10 σημεία συνολικά.



Σχήμα 2.6: Τελική αντιστοίχιση σημείων με τη μέθοδο «συναφούς» σχήματος.

2.2 Μέθοδος αναγνώρισης μοτίβου σε εικόνα (Template matching/Pattern Recognition)

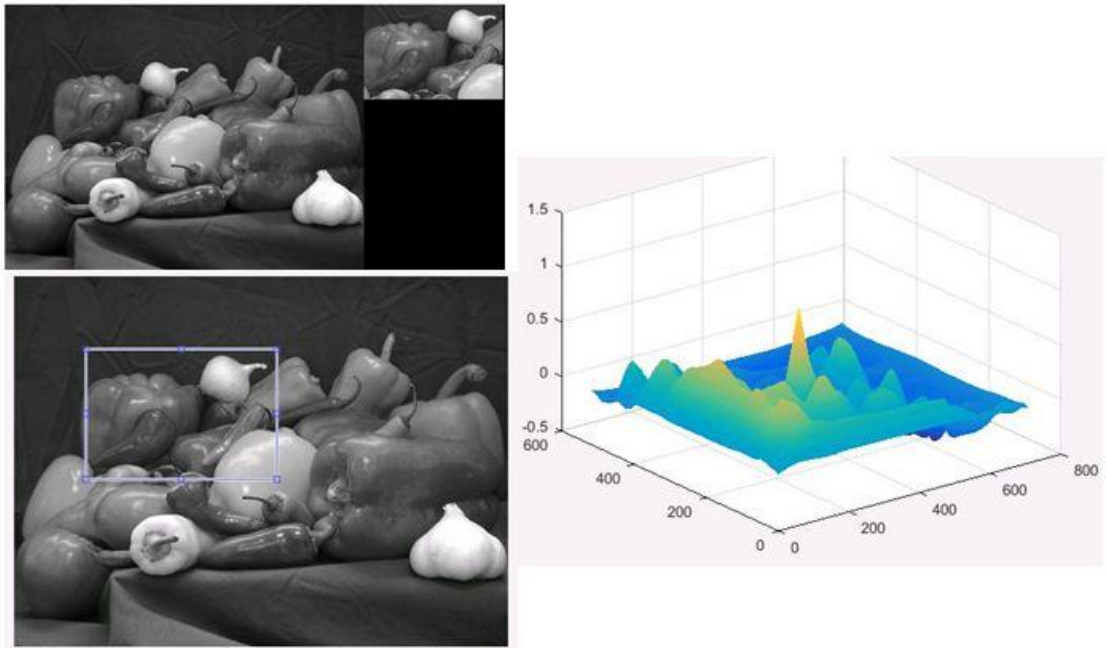
Με τη μέθοδο αυτή μπορεί να βρεθεί ένα υποσύνολο της εικόνας μέσα σε αυτή. Τέτοιες τεχνικές βρίσκουν εφαρμογή στην ανίχνευση οχημάτων, σε ιατρικές εφαρμογές στη ρομποτική όπως και στη βιομηχανία. Το κρίσιμο σημείο είναι η υιοθέτηση ενός μεγέθους για την ποσοτικοποίηση της αντιστοίχισης. Συνοπτικά, η υλοποίηση της μεθόδου περιλαμβάνει τη μετακίνηση της εικόνας του αντικειμένου που αναζητείται σε όλη την επιφάνεια της εικόνας που πραγματοποιείται η αναζήτηση.

2.2.1 Γενικά

Για την υλοποίηση της μεθόδου απαιτείται η ύπαρξη δύο εικόνων.

- Η εικόνα αναζήτησης
- Η εικόνα του αντικειμένου που αναζητείται, η οποία πρέπει να είναι ένα υποσύνολο της εικόνας αναζήτησης.

Στη συνέχεια η εικόνα του αντικειμένου μετακινείται σε κάθε πιθανό σημείο της επιφάνειας της εικόνας αναζήτησης, ώστε να βρεθεί η θέση της. Η αντιστοίχιση που επιτυγχάνεται κάθε φορά στην εικόνα αναζήτησης ποσοτικοποιείται μέσω κατάλληλων μεγεθών που θα αναλυθούν στη συνέχεια. Τέλος, η θέση που έχει επιτευχθεί η καλύτερη αντιστοίχιση δίνεται ως η θέση του αντικειμένου στην εικόνα αναζήτησης [12]. Στο ακόλουθο σχήμα φαίνεται η παραπάνω διαδικασία για δυο ενδεικτικές εικόνες.



Σχήμα 2.7: Εφαρμογή της μεθόδου αναγνώρισης μοτίβου για δυο τυπικές εικόνες (Πάνω αριστερά - Εικόνα αναζήτησης και εικόνα αντικειμένου), (Δεξιά - Βαθμός αντιστοίχισης στις διάφορες θέσεις), (Κάτω αριστερά - Αποτέλεσμα αντιστοίχισης).

Ήδη γίνονται αντιληπτοί κάποιοι από τους περιορισμούς της μεθόδου. Το αντικείμενο που αναζητείται, πρέπει να υπάρχει στην εικόνα αναζήτησης, ειδάλλως η διαδικασία θα επιστρέψει την αμέσως επόμενη καλύτερη αντιστοίχιση. Επίσης, το αντικείμενο πρέπει να παρουσιάζεται αυτούσιο στην εικόνα αναζήτησης. Σε κάθε άλλη περίπτωση απαιτούνται επιπρόσθετες διαδικασίες για την τελική αντιστοίχιση. [13] Τέλος, η εικόνα του αντικειμένου πρέπει να συγκριθεί με όλη την εικόνα, τουλάχιστον μια φορά, γεγονός που αυξάνει τον υπολογιστικό φόρτο.[14]

2.2.2 Τρόποι ποσοτικοποίησης της αντιστοίχισης [15]

1. Ετεροσυσχέτιση

Μια ψηφιακή εικόνα είναι ουσιαστικά ένα διακριτό σήμα. Μια ασπρόμαυρη εικόνα για παράδειγμα είναι ένα διακριτό σήμα το οποίο μπορεί να πάρει τιμές όσες κι οι αποχρώσεις του γκρι (π.χ. 256). Μια ειδική περίπτωση είναι η δυαδική εικόνα που πρόκειται για ψηφιακό σήμα με τιμές 0 και 1. Συνεπώς, μπορούμε να χρησιμοποιήσουμε τεχνικές σύγκρισης διακριτών σημάτων σε εικόνες. ([16],[17])

Στην επεξεργασία σήματος, η ετεροσυσχέτιση είναι ένα μέτρο της ομοιότητας δύο σημάτων. Για διακριτά σήματα δίνεται από τον τύπο:

$$Corr_{x,y} = \sum_{n=0}^{N-1} x[n]y[n] \tag{2.16}$$

Αν έχουμε διακριτά σήματα δύο μεταβλητών i,j όπως είναι οι εικόνες η παραπάνω εξίσωση μετατρέπεται στην:

$$Corr_{x,y} = \sum_{\iota}\sum_{j} x[i,j]y[i,j] \tag{2.17}$$

Η ετεροσυσχέτιση δύο διακριτών σημάτων μας δίνει ικανοποιητικά αποτελέσματα όταν τα σήματα που συγκρίνουμε είναι συγκρίσιμης ενέργειας. Σε διαφορετική περίπτωση τα αποτελέσματα δεν είναι συγκρίσιμα. Για το λόγο αυτό, κανονικοποιούμε την συνάρτηση της ετεροσυσχέτισης με έναν παράγοντα που δίνει την ενέργεια των σημάτων. Οι τιμές της κανονικοποιημένης ετεροσυσχέτισης πλέον είναι στο διάστημα [0,1]. Όσο μεγαλύτερες τιμές τόσο μεγαλύτερη ομοιότητα μεταξύ των δύο σημάτων.

$$Norm\_Corr_{x,y} = \frac{\sum_{\iota}\sum_{j} x[i,j]y[i,j]}{\sqrt{\sum_{i}\sum_{j} x^2[i,j]\sum_{i}\sum_{j} y^2[i,j]}} \tag{2.18}$$

2. Άθροισμα απόλυτων διαφορών (Sum of Absolute Differences)

Με αυτή τη μέθοδο υπολογίζονται οι διαφορές στις απόλυτες τιμές των δυο εικόνων. Αν η διαφορά είναι 0 τότε σημαίνει ότι το αντικείμενο έχει βρεθεί στην εικόνα αναζήτησης.

$$SAD(i,j) = \sum_{i}\sum_{j} |x[i,j] - y[i,j]| \tag{2.19}$$

Για να χρησιμοποιηθεί η μέθοδος αυτή πρέπει οι δύο εικόνες να έχουν το ίδιο μέγεθος.

3. Άθροισμα τετραγωνικών διαφορών (Sum of Squared Differences)

Όπως και στην προηγούμενη μέθοδο, υπολογίζονται οι διαφορές στις τιμές σε κάθε συντεταγμένη των δύο εικόνων. Βρίσκοντας τη θέση του ελάχιστου της συνάρτησης, υπολογίζεται η θέση του αντικειμένου στην εικόνα αναζήτησης.

$$SSD(i,j) = \sum_{i}\sum_{j} (x[i,j] - y[i,j])^2 \tag{2.20}$$

4. Γενικευμένος Μετασχηματισμός Hough

Ο μετασχηματισμός Hough αρχικά αναπτύχθηκε για την ανίχνευση ορισμένων σχημάτων, όπως ευθείες, κύκλους, ελλείψεις σε μια εικόνα. Σε αυτές τις περιπτώσεις είναι γνωστή η συνάρτηση του σχήματος και αναζητείται ο προσανατολισμός και η θέση του στην εικόνα. Ο γενικευμένος μετασχηματισμός Hough είναι μια παραλλαγή αυτού, που μπορεί να χρησιμοποιηθεί στην ανίχνευση αφηρημένων σχημάτων, των οποίων δεν υπάρχει γνωστή συνάρτηση.[18]

Κατά τη μέθοδο αυτή χρησιμοποιείται η πληροφορία στις ακμές για να καθοριστεί μια αντιστοίχιση του προσανατολισμού ενός σημείου μιας ακμής σε σχέση με ένα σημείο αναφοράς στο αντικείμενο (π.χ. κέντρο μάζας). Στην περίπτωση μιας δυαδικής εικόνας, κάθε σημείο ακμής μπορεί να είναι σημείο του επιθυμητού μοτίβου συνεπώς δημιουργείται ένας γεωμετρικός τόπος στον χώρο Hough. Τα σημεία του χώρου Hough που έχει το μεγαλύτερο αριθμό σημείων δείχνει τα σημεία που βρίσκεται το επιθυμητό μοτίβο στην εικόνα αναζήτησης.

Συγκεκριμένα η διαδικασία εύρεση μοτίβου σε εικόνα χρησιμοποιώντας το γενικευμένο μετασχηματισμό Hough ([19],[20]) έχει ως εξής:

Στην εικόνα αντικειμένου:

α) Επιλογή ενός σημείου αναφοράς Z0 για το σχήμα.

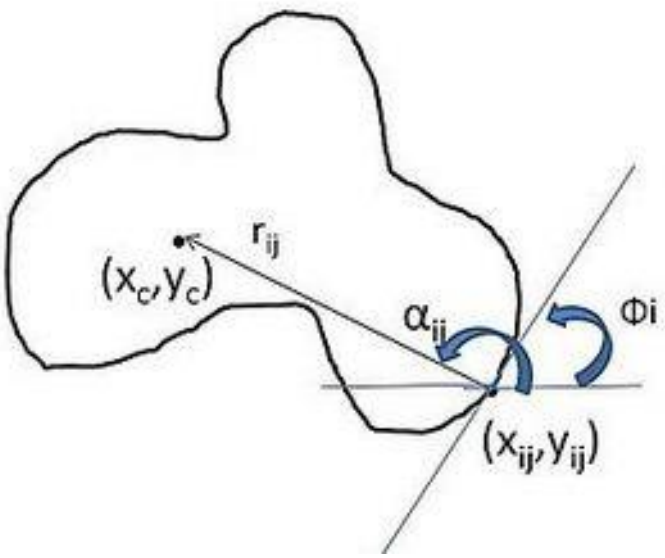
β) Για κάθε σημείο στο περίγραμμα υπολογίζεται ο προσανατολισμός της βάθμωσης Φi και η απόσταση των συντεταγμένων από το σημείο αναφοράς r_{ij}=Zij-Z0.

γ) Αποθήκευση των rij σαν συνάρτηση των προσανατολισμων Φ σε έναν πίνακα R. (Κάθε προσανατολισμός θα περιέχει αρκετές αποστάσεις rij.)

Στην εικόνα αναζήτησης:

Για κάθε σημείο ακμής x στην εικόνα βρίσκεται ο προσανατολισμός της βάθμωσης Φ και σε έναν πίνακα Α αυξάνονται κατά x+r όλα τα αντίστοιχα σημεία. Κατά αυτόν τον τρόπο δίνεται κάθε πιθανή θέση του σημείου αναφοράς στην εικόνα αναζήτησης. Το μέγιστο στον πίνακα Α σημαίνει παρουσία του μοτίβου στην εικόνα αναζήτησης.

i	ϕ_i	R_{ϕ_i}
1	0	(r_{11}, α_{11}) $(r_{12}, \alpha_{12})\dots$ (r_{1n}, α_{1n})
2	$\Delta\phi$	(r_{21}, α_{21}) $(r_{22}, \alpha_{22})\dots$ (r_{2m}, α_{2m})
3	$2\Delta\phi$	(r_{31}, α_{31}) $(r_{32}, \alpha_{32})\dots$ (r_{3k}, α_{3k})
...	...	...



Σχήμα 2.8: Γενικευμένος μετασχηματισμός Hough - Εύρεση πίνακα R

Ένα από τα βασικά πλεονεκτήματα του μετασχηματισμού Hough είναι ότι λειτουργεί ακόμα και με μη συνεχόμενες ακμές, καθώς επίσης ότι έχει μεγάλη ανοχή στο θόρυβο.

Οι παραπάνω είναι οι πιο συνήθεις μέθοδοι που χρησιμοποιούνται για την υλοποίηση της αναγνώρισης μοτίβου σε εικόνα. Στη βιβλιογραφία αναφέρονται και άλλες, όπως για παράδειγμα ορισμένες που βασίζονται σε τεχνικές βέλτιστης διαδρομής, απόστασης επεξεργασίας και ευκλείδειας απόστασης που δε θα αναλυθούν περαιτέρω στα πλαίσια της εργασίας.([21], [22])

2.3 Ανιχνευτές Χαρακτηριστικών (Feature Detectors)

Μέσω των μεθόδων αυτών γίνεται αφαίρεση πληροφορίας από την εικόνα, και για κάθε σημείο της παίρνονται αποφάσεις αν υπάρχει ένα χαρακτηριστικό συγκεκριμένου τύπου στο συγκεκριμένο σημείο.

2.3.1 Χαρακτηριστικά εικόνας

Δεν υπάρχει κάποιος συγκεκριμένος ορισμός του τι αποτελεί χαρακτηριστικό (feature) και η ακριβής ερμηνεία εξαρτάται από το εκάστοτε πρόβλημα ή την εφαρμογή. Δεδομένου αυτού, ένα χαρακτηριστικό είναι ένα σημείο ενδιαφέροντος μιας εικόνας. Τα χαρακτηριστικά χρησιμοποιούνται ως βάση για την ανάπτυξη πολλών αλγορίθμων τεχνητής όρασης, συνεπώς, όσο καλύτερος ο ανιχνευτής, που χρησιμοποιείται για την εύρεση τους τόσο καλύτερος είναι ο αλγόριθμος. Η κύρια

ιδιότητα που πρέπει να χαρακτηρίζει έναν τέτοιο ανιχνευτή είναι η επαναληψιμότητα.[23] Ορισμένοι τύποι χαρακτηριστικών προς ανίχνευση είναι οι εξής:

- Ακμές
- Γωνίες
- Περιοχές (blobs)

2.3.2 Ανιχνευτής Γωνιών Harris (Harris Corner Detector)

Οι γωνίες σε μια εικόνα είναι περιοχές με μεγάλες αλλαγές έντασης (απόχρωσης) προς όλες τις κατευθύνσεις. Ο ανιχνευτής αυτός αναζητά λοιπόν που η διαφορά έντασης μεγιστοποιείται [24]. Η διαφορά της έντασης για μια μετακίνηση (u,v) προς όλες τις κατευθύνσεις δίνεται από την συνάρτηση:

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \qquad (2.21)$$

Όπου: w(x,y), η συνάρτηση του παραθύρου που εξετάζεται.
I(x+u,y+v), η ένταση που προέκυψε από τη μετακίνηση.
I(x,y), η ένταση.

Η συνάρτηση του παραθύρου είναι είτε ένα ορθογωνικό παράθυρο, είτε ένα γκαουσιανό παράθυρο. Όπως αναφέρθηκε, για την ανίχνευση γωνίας πρέπει η συνάρτηση E να είναι μέγιστη. Αναλύοντας με ανάπτυγμα Taylor την E προκύπτει:

$$E(u,v) \approx [u,v] M \begin{bmatrix} u \\ v \end{bmatrix} \qquad (2.22)$$

Όπου $$M = \sum_{x,y} w(x,y) \begin{bmatrix} IxIx & IxIy \\ IxIy & IyIy \end{bmatrix} \qquad (2.23)$$

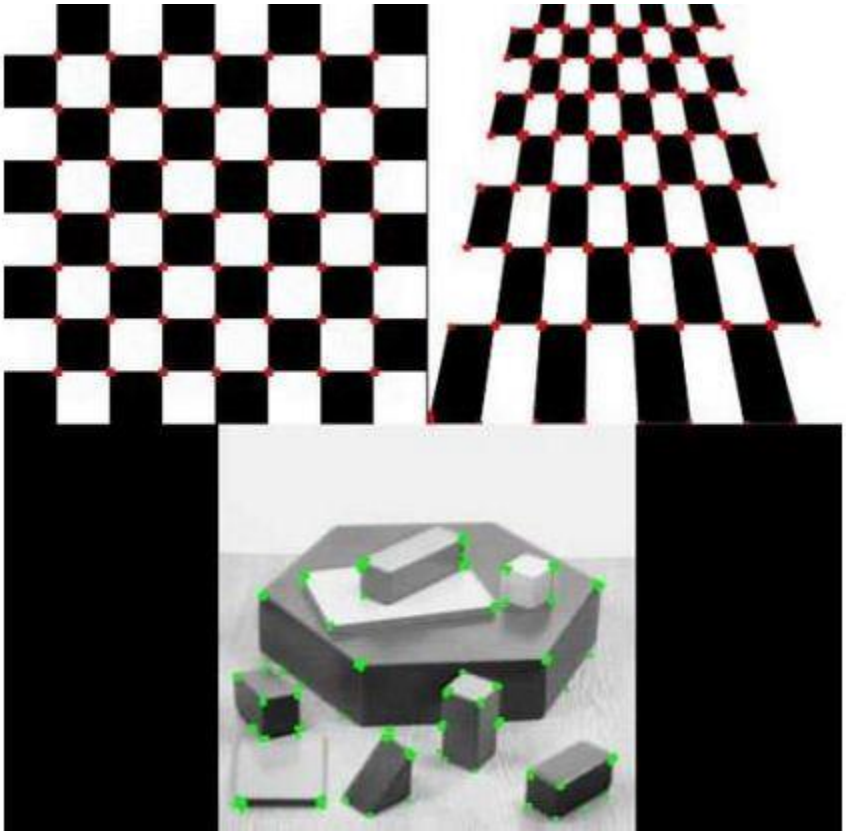
Με Ix,Iy τις παραγώγους κατά την x και y διεύθυνση της έντασης, αντίστοιχα.

Για να βρεθεί αν το παράθυρο που εξετάζεται είναι γωνία ελέγχεται μια τιμή R που προκύπτει από:

$$R = \det(M) - k(trace(M))^2 \qquad (2.24)$$

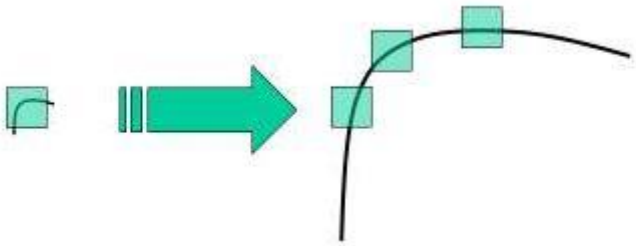
Αν:
$$\begin{cases} R\ μικρό, τότε\ η\ περιοχή\ είναι\ ευθεία \\ R < 0, τότε\ η\ περιοχή\ είναι\ ακμή \\ R\ μεγάλο, τοτέ\ είναι\ γωνία \end{cases}$$

Στο ακόλουθο σχήμα παρουσιάζονται ορισμένες ενδεικτικές εφαρμογές του ανιχνευτή αυτού [25].



Σχήμα 2.8: Εφαρμογή του ανιχνευτή γωνιών Harris σε τρεις ενδεικτικές εικόνες.

Είναι σαφές, πως τα αποτελέσματα αυτού του ανιχνευτή παραμένουν αμετάβλητα ως προς την περιστροφή. Δε συμβαίνει το ίδιο όμως και με την αλλαγή κλίμακας. Μια περιστροφή της εικόνας δε θα άλλαζε τις γωνίες της, μια αλλαγή κλίμακας όμως, θα τις επηρέαζε, όπως φαίνεται στο παρακάτω σχήμα.



Σχήμα 2.9: Για το ίδιο παράθυρο η ίδια καμπύλη δεν ανιχνεύεται ως γωνία με τον ανιχνευτή Harris

2.3.3 SIFT (Scale Invariant Feature Transform) [26]

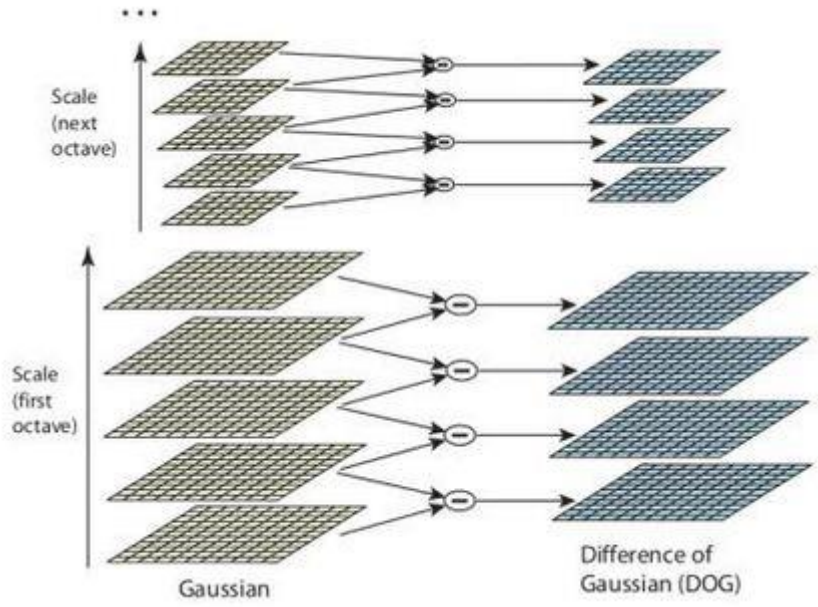
Από το σχήμα 2.9 γίνεται αντιληπτό, πως χρησιμοποιώντας το ίδιο παράθυρο, δεν είναι δυνατό να ανιχνεύσουμε σημεία ενδιαφέροντος σε εικόνες υπό διαφορετική κλίμακα. Για το λόγο αυτό στη μέθοδο αυτή χρησιμοποιείται κλιμακωτό φιλτράρισμα (scale-space filtering). Υπολογίζεται η Λαπλασιανή της εικόνας, η οποία έχει

φιλτραριστεί με ένα φίλτρο Gauss αρχικά, για να περιοριστεί η ευαισθησία στο θόρυβο.

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{\frac{-(x^2+y^2)}{2\sigma^2}} \qquad (2.25)$$

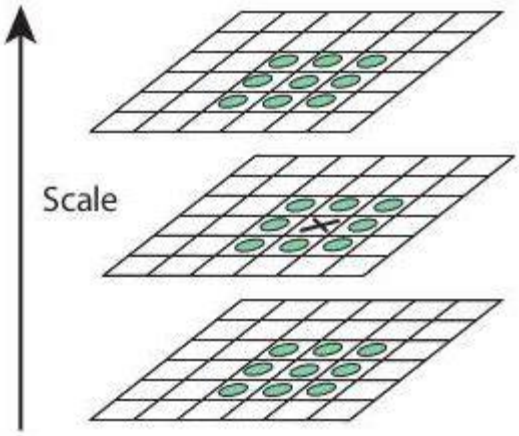
Η διαδικασία αυτή επαναλαμβάνεται για διαφορετικές τιμές του σ. Αυτό έχει ως αποτέλεσμα η συνάρτηση LoG να λειτουργεί ως ανιχνευτής περιοχών σημείων ενδιαφέροντος (blob detector). Η παράμετρος σ λειτουργεί ουσιαστικά ως μεταβλητή κλίμακας.

Για τη μείωση του υπολογιστικού φόρτου ο αλγόριθμος δεν χρησιμοποιεί τη LoG αλλά τη διαφορά Γκαουσιανών (Difference of Gaussians), που είναι μια προσέγγιση της LoG. Η DoG, λοιπόν είναι η διαφορά που προκύπτει από τη χρήση στην εικόνα γκαουσιανών φίλτρων με δύο διαφορετικές τιμές του σ. Η διαδικασία αυτή επαναλαμβάνεται για διαφορετικές *οκτάβες* στην γκαουσιανή πυραμίδα της εικόνας.([27],[28]) Τα παραπάνω απεικονίζονται καλύτερα στο επόμενο σχήμα.



Σχήμα 2.10: Εύρεση της διαφοράς Γκαουσιανών για διαφορετικές οκτάβες της γκαουσιανής πυραμίδας.

Εφόσον οι διαφορές έχουν βρεθεί, αναζητούνται στην εικόνα τοπικά μέγιστα στις διαφορετικές κλίμακες. Για παράδειγμα, ένα pixel σε μια εικόνα συγκρίνεται με τα 8 γειτονικά του, καθώς επίσης με τα 9 pixels της επόμενης κλίμακας και τα 9 των προηγούμενων. Αν είναι τοπικό μέγιστο τότε είναι εν δυνάμει σημείο ενδιαφέροντος.

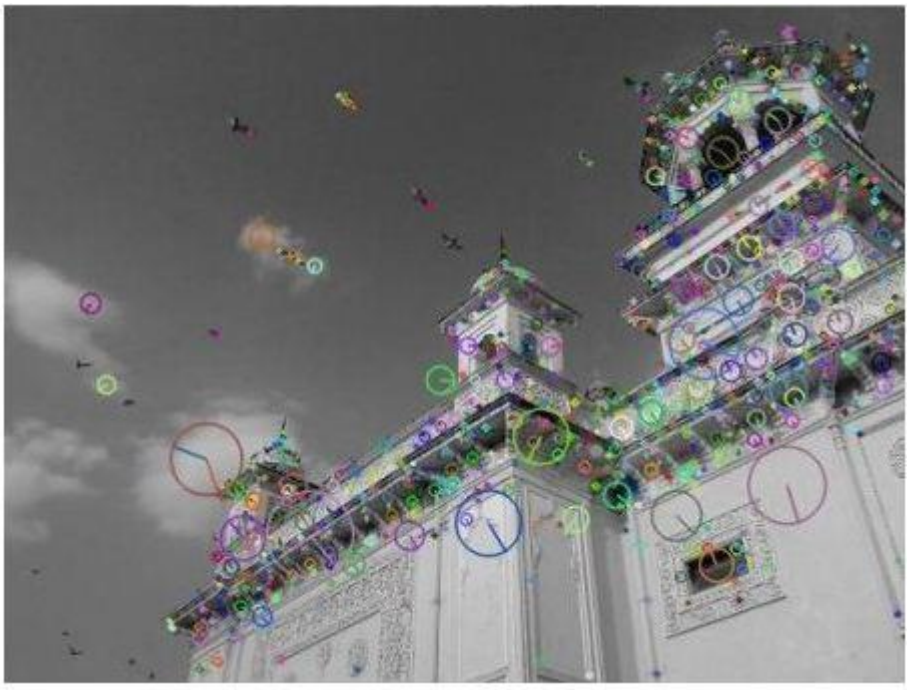


Σχήμα 2.11: Εύρεση τοπικών μεγίστων.

Στη συνέχεια, πρέπει τα εν δυνάμει σημεία ενδιαφέροντος να επεξεργαστούν και να απορριφθούν ορισμένα, ώστε να παραμείνουν μόνο τα πιο ισχυρά. Από το διαχωρισμό απορρίπτονται σημεία χαμηλής αντίθεσης και σημεία ακμών.

Σε κάθε εναπομείναν σημείο γίνεται μια ανάθεση προσανατολισμού, ώστε τα αποτελέσματα να μην επηρεάζονται από την περιστροφή.

Παρακάτω απεικονίζονται τα σημεία ενδιαφέροντος που βρέθηκαν με τη μέθοδο SIFT καθώς και ο προσανατολισμός τους.



Σχήμα 2.12: Ενδεικτική εφαρμογή της μεθόδου SIFT (Εύρεση σημείων ενδιαφέροντος και του προσανατολισμού τους).[29]

2.3.4 SURF (Speeded Up Robust Features)

Η μέθοδος SIFT αν και παρουσιάζει αξιόπιστα αποτελέσματα είναι σχετικά αργή. Μια γρηγορότερη μέθοδος στον εντοπισμό των σημείων ενδιαφέροντος είναι η SURF. Πλέον η LoG δεν προσεγγίζεται από τη διαφορά γκαουσιανών (DoG), αλλά μέσω φιλτραρίσματος τύπου «κουτιού» (box blur). Η προσέγγιση αυτή αυξάνει την ταχύτητα των υπολογισμών, καθώς η συνέλιξη της εικόνας με τέτοια φίλτρα μπορεί να απλουστευτεί κατά πολύ μέσω κάποιων χειρισμών (πχ integral images). Ο καθορισμός του προσανατολισμού και η περιγραφή του σημείου γίνεται με τη χρήση του ίδιου πίνακα (Hessian matrix). Σύμφωνα με τη βιβλιογραφία, η μέθοδος αυτή είναι μέχρι και 3 φορές γρηγορότερη από τη SIFT, παρέχοντας παράλληλα συγκρίσιμα αποτελέσματα. [30]

ΚΕΦΑΛΑΙΟ 3

ΕΥΡΕΣΗ ΑΝΤΙΚΕΙΜΕΝΟΥ ΣΕ ΕΙΚΟΝΑ ΚΑΙ ΛΗΨΗ ΣΗΜΑΝΤΙΚΩΝ ΜΕΓΕΘΩΝ

3.1 Λήψη εικόνας

Το πρώτο και από τα σημαντικότερα στάδια της τεχνητής όρασης (computer vision), αφορά τη λήψη της εικόνας προς επεξεργασία ή αλλιώς την αναπαράσταση του τρισδιάστατου κόσμου σε μια δισδιάστατη προβολή. Ο τρόπος με τον οποίο γίνεται αυτή η αναπαράσταση, είναι κρίσιμος, καθώς επηρεάζει την ικανότητα επεξεργασίας και ανάλυσης ψηφιακών εικόνων.

3.1.1 Τρισδιάστατη προβολή

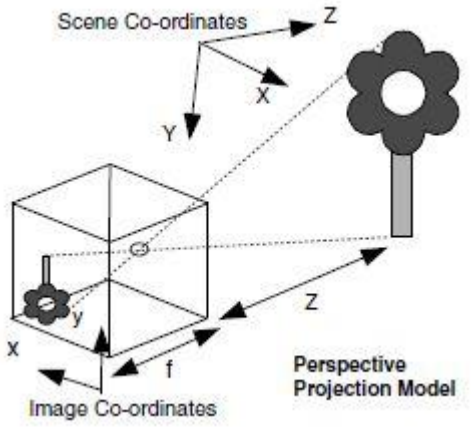
Ένα οποιοδήποτε μέσο λήψης εικόνας, όπως για παράδειγμα, μια κάμερα, μετατρέπει τις συντεταγμένες του τρισδιάστατου χώρου (X,Y,Z) σε συντεταγμένες δύο διαστάσεων (x,y) στο επίπεδο της εικόνας. Δύο τέτοιοι τρόποι αναπαράστασης είναι η *ορθογραφική (παράλληλη)* και η *προοπτική προβολή*.

Προοπτική προβολή: Σε αυτόν τον τρόπο προβολής το μέγεθος των αντικειμένων στην εικόνα εξαρτάται από την απόστασή τους από το μέσο λήψης. Συνεπώς, αντικείμενα που βρίσκονται μακρύτερα φαίνονται μικρότερα στην εικόνα. Γενικά η προοπτική προβολή μπορεί να περιγραφεί από τις παρακάτω εξισώσεις:

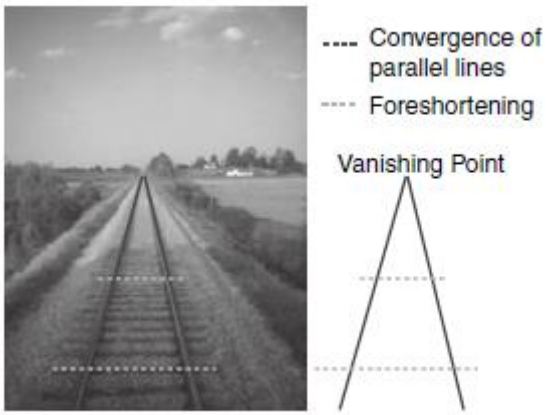
$$x = f\frac{X}{Z} \text{ και } y = f\frac{Y}{Z} \tag{3.1}$$

Ένα οποιοδήποτε σημείο με συντεταγμένες X,Y,Z (Z το βάθος) αναπαρίσταται στη θέση χ,y στην εικόνα που καθορίζεται από το μήκος εστίασης f (focal length) του φακού.

Χαρακτηριστικό, επίσης, αυτού του τρόπου προβολής είναι ότι παράλληλες γραμμές στον τρισδιάστατο χώρο τείνουν να τέμνονται στην εικόνα. Αυτό είναι γνωστό και ως προοπτική παραμόρφωση.



Σχήμα 3.1: Προοπτική προβολή αντικειμένου.



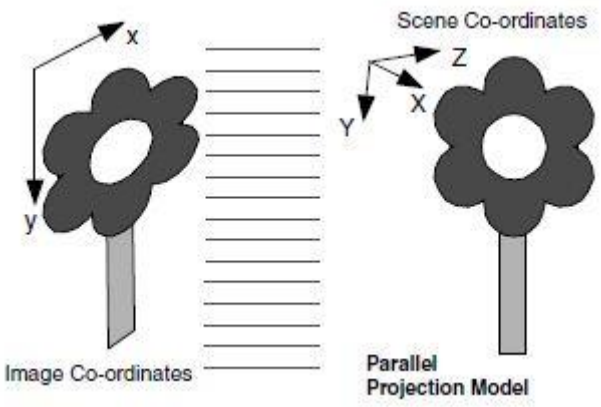
Σχήμα 3.2: Επιδράσεις προοπτικής προβολής στη γεωμετρία.

Τέλος, πρέπει να σημειωθεί, πως η προοπτική προβολή δεν είναι εύκολα αναστρέψιμη από μία και μόνο εικόνα. Για παράδειγμα, έχοντας ως δεδομένες τις συντεταγμένες (x,y) στην εικόνα και το μήκος εστίασης f του φακού, δεν είναι δυνατό να ανακτήσουμε τις συντεταγμένες X,Y,Z του αντικειμένου στο χώρο. Αν ωστόσο είναι γνωστές και οι συντεταγμένες X,Y στο χώρο (μετρημένες σε σχέση με τη θέση της κάμερας), τότε μπορεί να εκτιμηθεί το βάθος Z που βρίσκεται το αντικείμενο.

Ορθογραφική ή παράλληλη προβολή: Αυτός ο τρόπος προβολής επιτυγχάνεται με ειδικά μέσα λήψης εικόνας, όπως για παράδειγμα ένα σκάνερ. Μπορεί να περιγραφεί απλά από τις εξισώσεις:

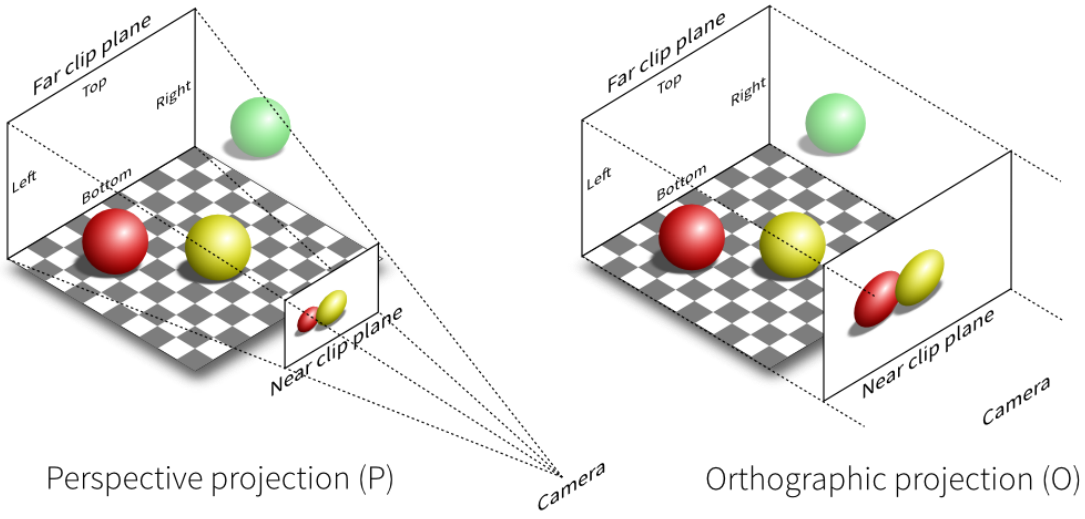
$$x = X \text{ και } y = Y \tag{3.2}$$

Όπως, παρατηρείται και από τις εξισώσεις (3.2), το επίπεδο που βρίσκεται το αντικείμενο και το επίπεδο προβολής στην εικόνας είναι παράλληλα. Αυτό παρουσιάζεται καλύτερα και στο σχήμα 3.3, παρακάτω.



Σχήμα 3.3: Παράλληλη προβολή αντικειμένου.

Σε αυτόν τον τύπο προβολής, όλη η πληροφορία που έχει να κάνει με το βάθος/ύψος έχει πλέον χαθεί. Συνεπώς, το μέγεθος του αντικειμένου δεν έχει σχέση με την απόστασή του από το μέσο λήψης. Είναι εύκολα αντιληπτό, πως κατ' αυτόν τον τρόπο προβολής η ανάκτηση της θέσης του αντικειμένου είναι αρκετά ευκολότερη, με δεδομένο ότι γνωρίζουμε το βάθος τοποθέτησής του Z. Τέλος, στο σχήμα 2.4 παρουσιάζεται ένα παράδειγμα προοπτικής προβολής σε αντιδιαστολή με ένα παράδειγμα παράλληλης προβολής. [1]

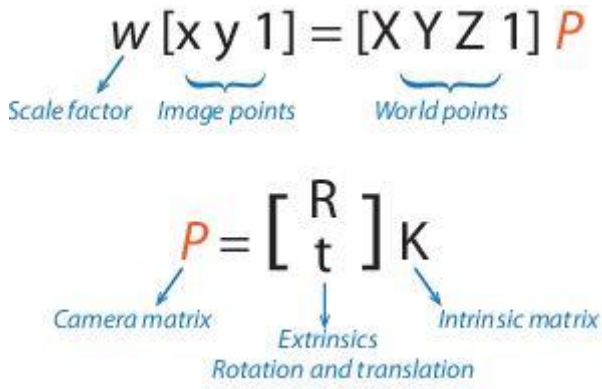


Σχήμα 3.4: Προοπτική και παράλληλη προβολή αντικειμένου.

3.1.2 Ανάκτηση πραγματικών συντεταγμένων

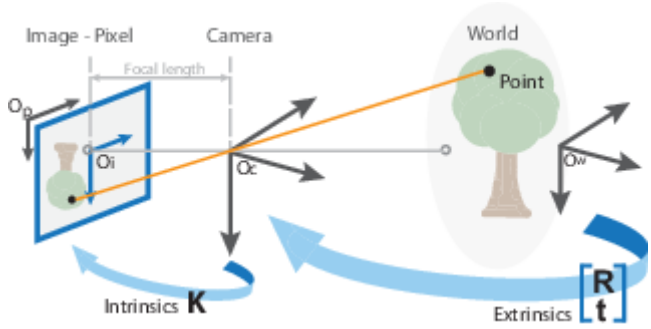
Όπως αναφέρθηκε, η ανάκτηση των χωρικών συντεταγμένων είναι άμεσα συνδεδεμένη με τον τρόπο προβολής, που έχει χρησιμοποιηθεί. Στην περίπτωση της ορθογραφικής προβολής, τα πράγματα είναι απλά, αφού το μόνο που χρειάζεται είναι το βάθος, για το οποίο η πληροφορία χάνεται εντελώς κατά την προβολή και σε ορισμένες περιπτώσεις ένας όρος αναλογίας ανάμεσα στις συντεταγμένες x,y της εικόνας και X,Y του χώρου. Όσον αφορά την προοπτική προβολή, χρησιμοποιείται πλέον η τεχνική του καλιμπραρίσματος της κάμερας (camera calibration).

Κάθε σημείο του τρισδιάστατου χώρου, προβάλλεται στο δισδιάστατο επίπεδο της εικόνας μέσω ενός μετασχηματισμού που επιβάλλει η κάμερα, όπως φαίνεται στις παρακάτω εξισώσεις στο σχήμα 3.5:



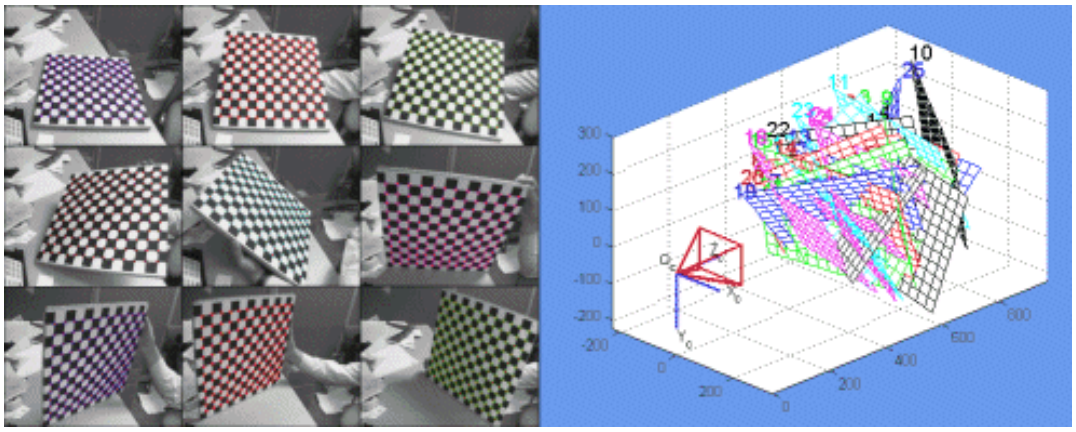
Σχήμα 3.5: Μετασχηματισμός συντεταγμένων χώρου σε συντεταγμένες στο επίπεδο της εικόνας (προοπτική προβολή).

Η διαδικασία του καλιμπραρίσματος υπολογίζει το μετασχηματισμό που επιβάλλει η κάμερα (πίνακας P), χρησιμοποιώντας της εξωγενείς (extrinsic) και ενδογενείς (intrinsic) παραμέτρους. Οι εξωγενείς παράμετροι αφορούν τη θέση της κάμερας στο χώρο, ενώ οι ενδογενείς το οπτικό κέντρο και το μήκος εστίασης. Μέσω των εξωγενών παραμέτρων ένα σημείο του χώρου μεταφράζεται σε συντεταγμένες της κάμερας, και εν συνεχεία σε συντεταγμένες στο επίπεδο της εικόνας, μέσω των ενδογενών παραμέτρων. Αυτό φαίνεται στο σχήμα 3.6.



Σχήμα 3.6: Ενδογενείς και εξωγενείς παράμετροι κάμερας.

Για το καλιμπράρισμα της κάμερας, απαιτείται η χρησιμοποίηση ενός μοτίβου καλιμπραρίσματος. Συνήθης τακτική είναι η επιλογή της «ασύμμετρης σκακιέρας» ως μοτίβο, η μια πλευρά της, δηλαδή, να αποτελείται από ζυγό αριθμό ασπρόμαυρων τετραγώνων και η άλλη από μονό. [31] Πριν τη βαθμονόμηση, το μοτίβο φωτογραφίζεται από την κάμερα υπό διάφορες οπτικές γωνίες, μετριέται το μέγεθος των τετραγώνων της σκακιέρας, με όσο το δυνατόν μεγαλύτερη ακρίβεια και εισάγονται στο πρόγραμμα. [32]



. Σχήμα 3.7: Καλιμπράρισμα κάμερας κατά την προοπτική προβολή για ανάκτηση των πραγματικών συντεταγμένων.

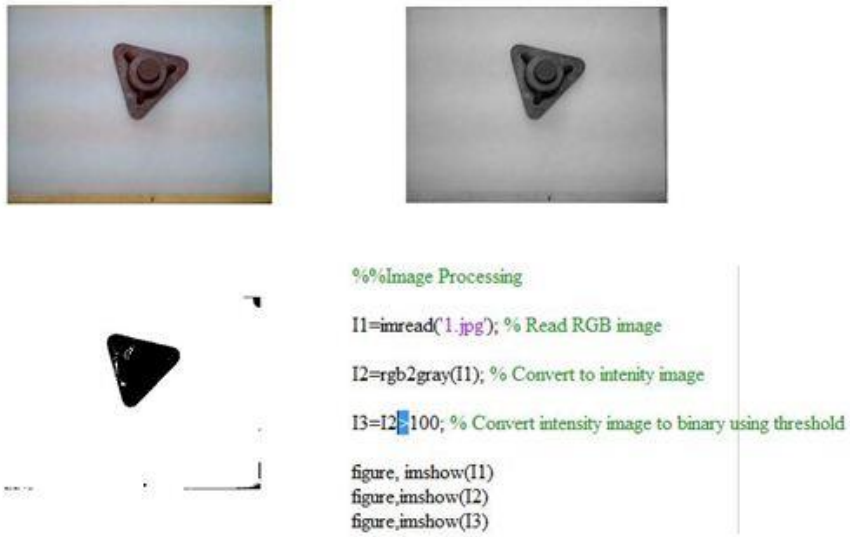
Η διαδικασία του καλιμπραρίσματος μας εξάγει τον πίνακα P, με τις ενδογενείς και εξωγενείς παραμέτρους της κάμερας, τον οποίο μπορούμε να χρησιμοποιούμε για να αντιστοιχίσουμε κάθε σημείο της εικόνας με ένα σημείο στο χώρο. [33]

3.2 Δυαδική Επεξεργασία Εικόνας

Οι εικόνες που λαμβάνουμε, συνήθως, είναι είτε έγχρωμες (RGB images) είτε ασπρόμαυρες (grayscale/intensity images). Κάθε έγχρωμη εικόνα μπορεί να μετατραπεί σε ασπρόμαυρη χρησιμοποιώντας διαφορετικές τιμές έντασης του γκρίζου χρώματος για κάθε απόχρωση. Ένας συνήθης αριθμός τιμών έντασης είναι 256, ενώ δεν είναι σπάνιο να χρησιμοποιηθούν 32, 64, 128 ακόμα και 512 αποχρώσεις για ορισμένες εφαρμογές. Όσο περισσότερες αποχρώσεις του γκρι τόσο καλύτερη αναπαράσταση της σκηνής, αλλά με κόστος το χώρο αποθήκευσης, και το χρόνο επεξεργασίας.

Εύκολα γίνεται αντιληπτό πως μια ασπρόμαυρη εικόνα μπορεί να μετατραπεί σε δυαδική εικόνα 2, δηλαδή, αποχρώσεων του γκρι (άσπρο – μαύρο). Με αυτόν τον τρόπο μπορούν να μειωθούν κατά πολύ οι χρόνοι επεξεργασίας της εικόνας. Για παράδειγμα μια εικόνα 256 αποχρώσεων έχει 8 φορές τις απαιτήσεις μνήμης μιας δυαδικής εικόνας.[34]

Στο παρακάτω παράδειγμα γίνεται η μετατροπή μιας έγχρωμης εικόνας ενός τεμαχίου, πρώτα σε ασπρόμαυρη 256 αποχρώσεων και τέλος σε δυαδική. Παρατίθενται ο κώδικας που χρησιμοποιήθηκε στο λογισμικό της Mathworks, Matlab και οι αντίστοιχες εικόνες.

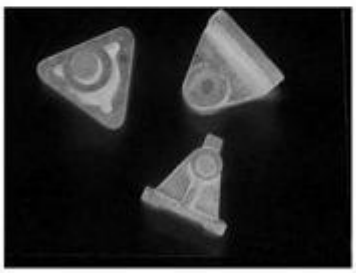


Σχήμα 3.8: Έγχρωμη εικόνα (Πάνω αριστερά). Ασπρόμαυρη 256 αποχρώσεων (Πάνω δεξιά), Δυαδική εικόνα (Κάτω αριστερά).

3.2.1 Απομόνωση - αναγνώριση αντικειμένου σε εικόνα

Συστήματα «δυαδικής όρασης» είναι χρήσιμα όταν η σιλουέτα ενός αντικειμένου περιέχει ικανή πληροφορία για την αναγνώριση του αντικειμένου και όπου το περιβάλλον μπορεί να ελεγχθεί επαρκώς. Για να ανακτηθεί μια καλή σιλουέτα του αντικειμένου, πρέπει, αρχικά, να απομονωθεί από το παρασκήνιο. Για να επιτευχθεί αυτό χρησιμοποιήθηκαν κάποιες βασικές τεχνικές επεξεργασίας εικόνας, που θα αναλύσουμε παρακάτω.

Επιθυμούμε να επεξεργαστούμε μόνο την περιοχή την οποία βρίσκονται τοποθετημένα τα αντικείμενα, ενώ στην εικόνα πιθανώς να έχουν συμπεριληφθεί και περιοχές έξω από αυτή. Αυτό συμβαίνει για παράδειγμα στο σχήμα 3.9. Παρατηρούμε, πως στην εικόνα προβάλλεται και μέρος του περιγράμματος της περιοχής τοποθέτησης. Για να μη μας επηρεάσει στη διαδικασία εξαγωγής της σιλουέτας του αντικειμένου, αρχικά, εισάγουμε μια εικόνα του φόντου και ύστερα την αφαιρούμε από την εικόνα προς επεξεργασία. Τα αποτελέσματα και οι εντολές που χρησιμοποιούνται παρουσιάζονται στο σχήμα 3.9.



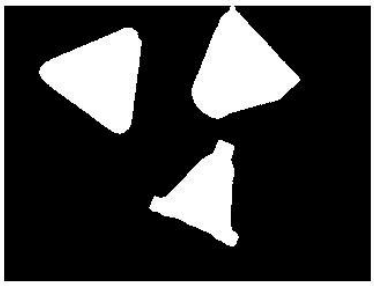
```
%% SUBSTRACT BACKGROUND

i1=imread('10.jpg');     %Read image
figure,imshow(i1)
i1=rgb2gray(i1);         % Convert image to grayscale
bg=imread('bg.jpg');     % Read background image
figure,imshow(bg)
bg=rgb2gray(bg);         % Convert image to grayscale
sub=imabsdiff(i1,bg);    % Substract background from image
figure,imshow(sub)
```

. Σχήμα 3.9: Εικόνα προς επεξεργασία (Πάνω αριστερά), Εικόνα παρασκηνίου (Πάνω δεξιά), Αποτέλεσμα από την αφαίρεση παρασκηνίου (Κάτω αριστερά).

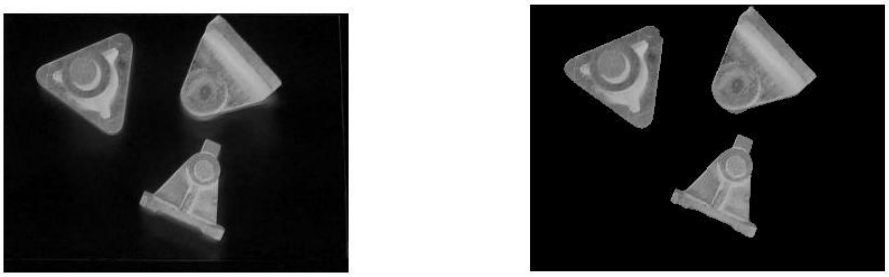
Αν η περιοχή τοποθέτησης, λοιπόν, δεν έχει ατέλειες, η εικόνα που προκύπτει μετά τη χρήση της εντολής *imabsdiff* θα περιέχει τα αντικείμενα που θέλουμε να απομονώσουμε, σε ένα ενιαίο φόντο. Είναι σημαντικό, η εικόνα του παρασκηνίου να

ληφθεί υπό τις ίδιες συνθήκες φωτισμού και θέσης της κάμερας, με την εικόνα προς επεξεργασία.

Εξαιτίας του φωτισμού, δημιουργούνται φαινόμενα σκίασης τα οποία είναι εμφανή μετά την αφαίρεση του παρασκηνίου. Για να τα περιορίσουμε κατά το δυνατόν χρησιμοποιούμε τη «μάσκα» των αντικειμένων. Η «μάσκα» είναι μια δυαδική εικόνα η οποία περιέχει μόνο το χώρο που περιλαμβάνουν τα αντικείμενα. Αφού τη δημιουργήσουμε, τη χρησιμοποιούμε για να αφήσουμε στην εικόνα προς επεξεργασία μόνο τα αντικείμενα που μας ενδιαφέρουν. Για παράδειγμα, για την εικόνα που προέκυψε στο σχήμα 3.9, χρησιμοποιούμε τη μάσκα που φαίνεται στο σχήμα 3.10. Στο σχήμα 3.11 εφαρμόζουμε τη μάσκα και βλέπουμε πλέον πως τα φαινόμενα σκίασης έχουν περιοριστεί κατά πολύ.



Σχήμα 3.10: Δημιουργία «μάσκας» αντικειμένων.



Σχήμα 3.11: Εικόνα προς επεξεργασία, πριν και μετά τη χρήση «μάσκας».

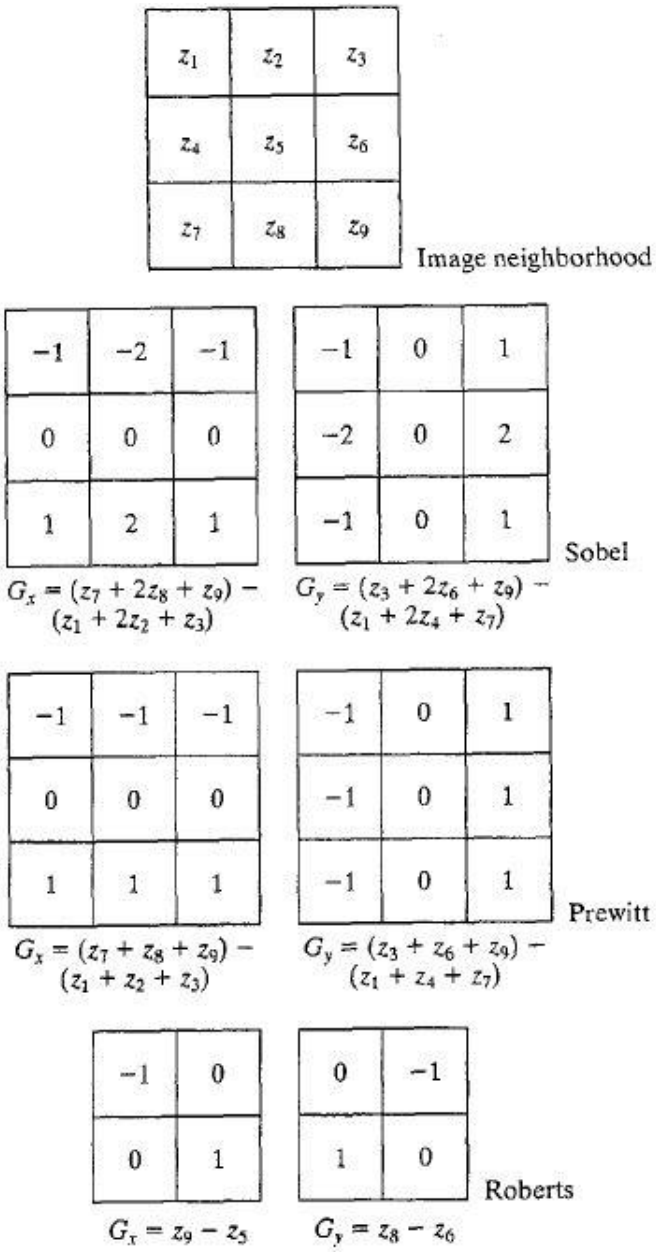
Εξαγωγή σιλουέτας αντικειμένων

Πλέον τα αντικείμενα μπορούν εύκολα να απομονωθούν από το παρασκήνιο, καθώς στα σημεία που υπάρχουν αντικείμενα η τιμή της απόχρωσής του γκρι διαφέρει σημαντικά από τα υπόλοιπα. Το λογισμίκο Matlab δίνει τη δυνατότητα χρησιμοποίησης αρκετών διαφορετικών ανιχνευτών ακμών (edge detectors). Κάποιοι απ' αυτούς είναι οι Sobel, Prewitt, Roberts, LoG, Zero crossings, Canny. Ο παρακάτω πίνακας εξηγεί συνοπτικά τις βασικές ιδιότητές τους.

Ανιχνευτής Ακμών	**Βασικές Ιδιότητες**
Sobel	Βρίσκει τις Gx και Gy συνιστώσες της βάθμωσης της γειτονιάς της εικόνας χρησιμοποιώντας ένα φίλτρο Sobel.
Prewitt	Βρίσκει τις Gx και Gy συνιστώσες τις βάθμωσης της γειτονιάς της εικόνας χρησιμοποιώντας ένα φίλτρο Prewitt.
Roberts	Βρίσκει τις Gx και Gy συνιστώσες τις βάθμωσης της γειτονιάς της εικόνας χρησιμοποιώντας ένα φίλτρο Roberts.
Laplacian of Gaussian	Βρίσκει τις ακμές στην εικόνα αναζητώντας μηδενικά σημεία εφαρμόζοντας ένα γκαουσιανό φίλτρο.
Zero Crossings	Βρίσκει τις ακμές στην εικόνα αναζητώντας μηδενικά σημεία εφαρμόζοντας ένα φίλτρο ορισμένο από το χρήστη.
Canny [35]	Βρίσκει τις ακμές ψάχνοντας τοπικά μέγιστα της βάθμωσης, η οποία υπολογίζεται χρησιμοποιώντας ένα γκαουσιανό φίλτρο. Η μέθοδος χρησιμοποιεί δύο όρια για να βρεί ισχυρές και μη ακμές

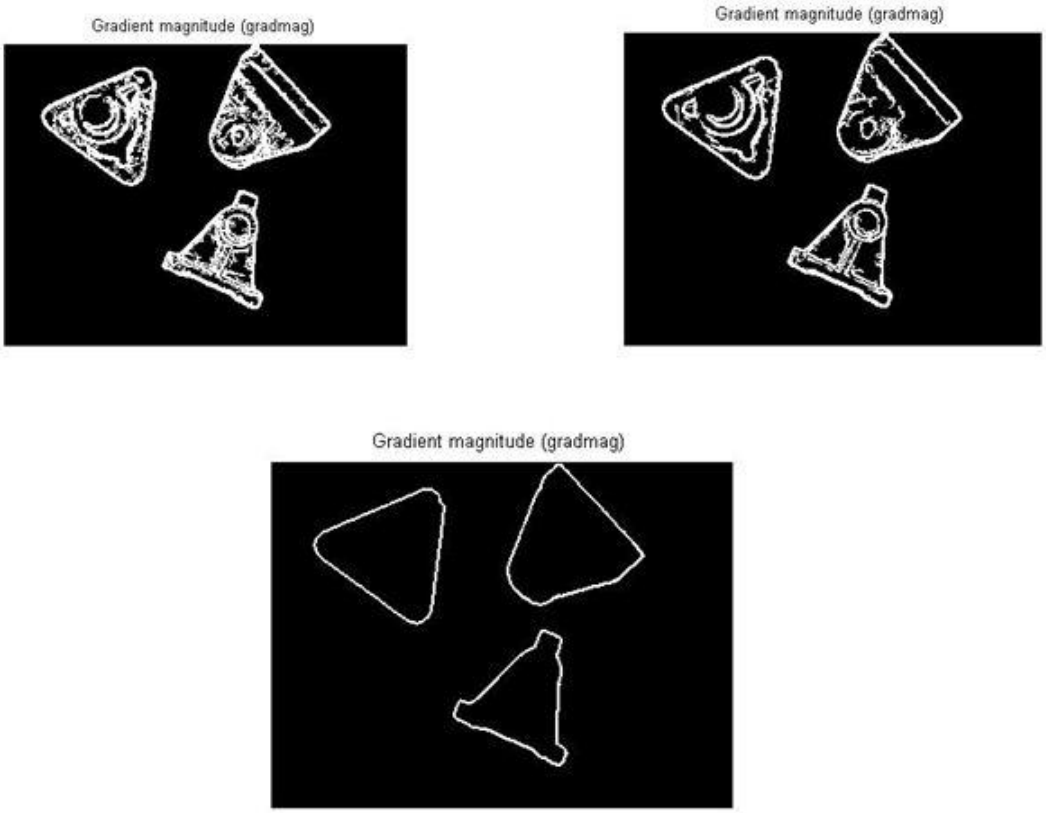
Πίνακας 3.1: Αναφορά των βασικών ιδιοτήτων για κάθε μέθοδο ανίχνευσης ακμών.[1]

Στο επόμενο σχήμα είναι συγκεντρωμένα τα φίλτρα Sobel, Prewitt, Roberts και απεικονίζεται ο τρόπος με τον οποίο υπολογίζονται οι συνιστώσες της βάθμωσης σε κάθε περίπτωση.



Σχήμα 3.12: Φίλτρα Sobel, Prewitt, Roberts στον υπολογισμό των συνιστωσών της βάθμωσης.

Στην παρούσα εργασία επιλέχθηκε η χρησιμοποίηση του ανιχνευτή ακμών Sobel. Στο σχήμα 3.13 φαίνεται μια ενδεικτική εφαρμογή του στην εικόνα, στην οποία έχει εφαρμοστεί η μάσκα (σχήμα 3.11).



Σχήμα 3.13: Ανίχνευση ακμών με χρήση του ανιχνευτή Sobel, μεταβάλλοντας την τιμή κατωφλίου (threshold).

Παρατηρούμε, πως όσο αυξάνουμε την τιμή κατωφλίου τόσο περισσότερες λεπτομέρειες χάνονται από την εικόνα. Τα αντικείμενα, που μελετώνται, μπορούν να αναγνωριστούν μόνο από το περίγραμμά τους, συνεπώς για μείωση του υπολογιστικού φόρτου, θέτουμε μια μεγάλη τιμή κατωφλίου.

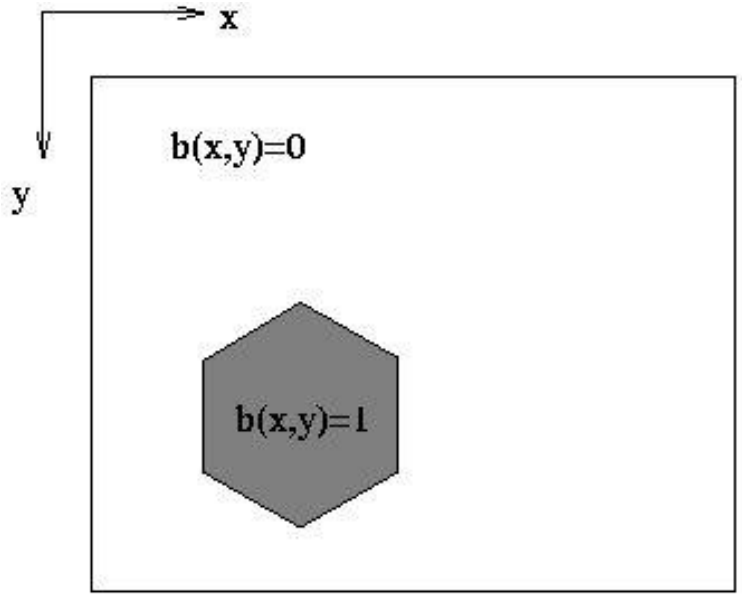
3.2.2 Ανάκτηση σημαντικών μεγεθών

Εφόσον έχουμε μια δυαδική εικόνα, δίνεται η δυνατότητα μέσω του λογισμικού να εξάγουμε σημαντικά μεγέθη, χρήσιμα για περαιτέρω υπολογισμούς. Αρχικά μπορούμε να εξάγουμε το περίγραμμα του κάθε αντικειμένου στην εικόνα μέσω της εντολής *bwboundaries* και εν συνεχεία να εμφανίσουμε μεγέθη όπως τα κέντρα των αντικειμένων (εκφρασμένα σε συντεταγμένες στο επίπεδο της εικόνας), η επιφάνειά τους, ο προσανατολισμός τους κ.α. μέσω της εντολής *regionprops*. Είναι σημαντικό σε αυτό το σημείο να επεξηγηθεί ο τρόπος υπολογισμού των παραπάνω μεγεθών, ώστε να αντιληφθούμε τις δυνατότητες και τους περιορισμούς, χρησιμοποιώντας την εν λόγω εντολή.

- Επιφάνεια

Σε μια δυαδική εικόνα η συνάρτηση που περιγράφει ένα αντικείμενο έχει την παρακάτω μορφή:

$$b(x,y) = \begin{cases} 1, & \textit{για } x,y \textit{ εντός του αντικειμένου} \\ 0, & \textit{για } x,y \textit{ εκτός του αντικειμένου} \end{cases} \quad (3.3)$$



Σχήμα 3.14: Δυαδική εικόνα

Η επιφάνεια που καταλαμβάνει το αντικείμενο στην εικόνα υπολογίζεται χρησιμοποιώντας τη μηδενική ροπή του αντικειμένου, δηλαδη:

$$A = \iint b(x,y)dxdy \quad (3.4)$$

- Κέντρο μάζας

Η θέση ενός αντικειμένου παίζει σημαντικό ρόλο σε πολλές εφαρμογές. Υπάρχουν πολλοί τρόποι να εκτιμηθεί η θέση ενός αντικειμένου, όπως να περιβληθεί από ένα ορθογώνιο ή μέσω του κέντρου μάζας. Η εντολή regionprops μας δίνει και τις δυο δυνατότητες. Συγκεκριμένα, το κέντρο μάζας υπολογίζεται χρησιμοποιώντας τις πρώτες ροπές του αντικειμένου, δηλαδή:

$$\overline{x} = \frac{\iint xb(x,y)dxdy}{A} \text{ και } \overline{y} = \frac{\iint yb(x,y)dxdy}{A} \quad (3.5)$$

Όπως γίνεται αντιληπτό, λόγω της προοπτικής προβολής το κέντρο μάζας και η επιφάνεια που καταλαμβάνει το αντικείμενο θα περιέχουν κάποιο σφάλμα σε σχέση με την πραγματικότητα. [36]

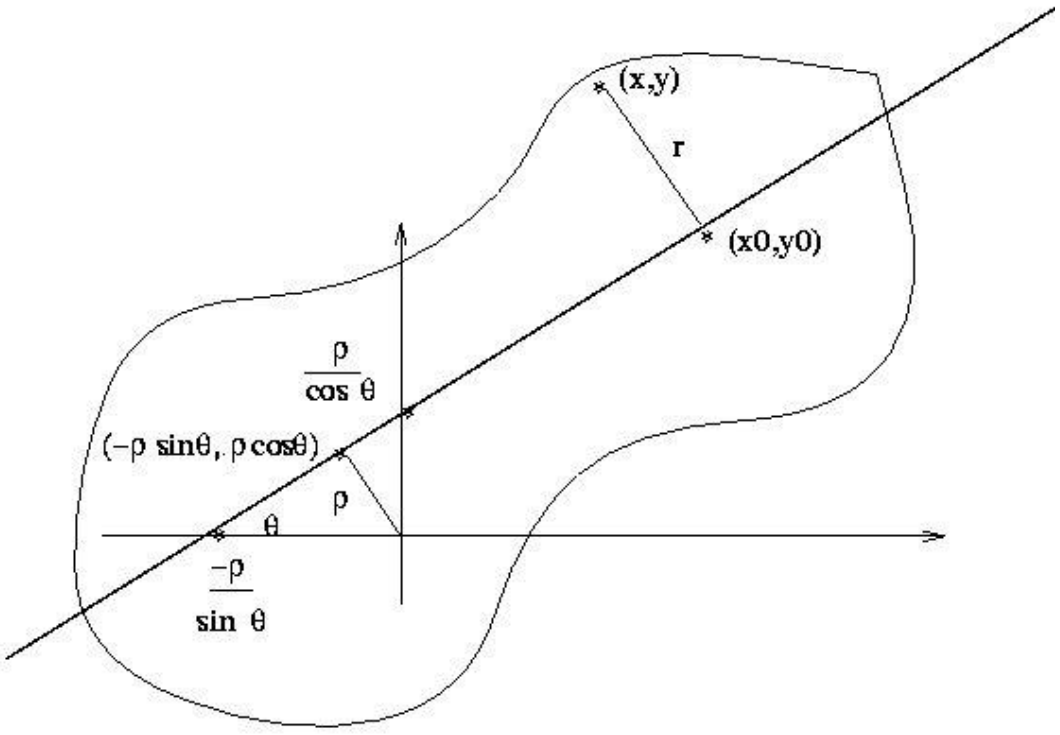
- Προσανατολισμός

Ο υπολογισμός του προσανατολισμού ενός αντικειμένου είναι πιο περίπλοκος από τον υπολογισμό της θέσης του. Για κάποια σχήματα ο προσανατολισμός δεν είναι μοναδικός. Για παράδειγμα, τέτοια σχήματα είναι ο κύκλος (άπειροι προσανατολισμοί), ένα ισόπλευρο τρίγωνο (3 προσανατολισμοί) και γενικά οποιοδήποτε σχήμα συμμετρικό ως προς την περιστροφή. Ο υπολογισμός του προσανατολισμού μέσω της εντολής *regionprops* είναι δυνατός μόνο αν το σχήμα δεν είναι συμμετρικό ως προς την περιστροφή. [37]

Συγκεκριμένα, ο προσανατολισμός υπολογίζεται μέσω του άξονα της ελάχιστης αδράνειας. Πρέπει δηλαδή να βρεθεί μια ευθεία, για την οποία ελαχιστοποιείται το:

$$I = \iint r^2\, b(x,y)dxdy \qquad (3.6)$$

Όπου r, κάθετη απόσταση του σημείου από την ευθεία που ψάχνουμε (σχήμα 3.15).



Σχήμα 3.15: Εύρεση προσανατολισμού

Αν η ευθεία αναπαρασταθεί σε πολικές συντεταγμένες, δηλαδή:

$$xsin\theta - ycos\theta + \rho = 0 \qquad (3.7)$$

τότε $$(xsin\theta - ycos\theta + \rho)^2 = r^2 \qquad (3.8)$$

Αν αντικαταστήσουμε το αποτέλεσμα της εξίσωσης 3.8 στην εξίσωση 3.6, παραγωγίσουμε ως προς ρ και θέσουμε το αποτέλεσμα ίσο με μηδέν έχουμε:

$$\iint 2(xsin\theta - ycos\theta + \rho)b(x,y)dxdy = 0 \qquad (3.9)$$

Διαιρώντας με την επιφάνεια Α που υπολογίσαμε στην εξίσωση 3.4 καταλήγουμε ότι :

$$A(\bar{x}sin\theta - \bar{y}cos\theta + \rho) = 0 \qquad (3.10)$$

Συνεπώς, η ευθεία του προσανατολισμού περνά από το κέντρο μάζας. Για να βρούμε λοιπόν τον προσανατολισμό πρέπει η γωνία θ να ελαχιστοποιεί το I. Αν αντικαταστήσουμε στην εξίσωση 3.6 τις μεταβλητές x,y με $\acute{x} = x - \bar{x}$ και $\acute{y} = y - \bar{y}$, το πρόβλημα ελαχιστοποίησης καταλήγει στην εξίσωση:

$$I = \alpha sin^2\theta - bsin\theta cos\theta + ccos^2\theta \qquad (3.11)$$

Όπου : $\alpha = \iint \acute{x}^2\, b(x,y)dxdy$, $b = 2\iint x'y'\, b(x,y)dxdy$ *και* $c = \iint \acute{y}^2\, b(x,y)dxdy$

Χρησιμοποιώντας τριγωνομετρικές ιδιότητες η σχέση 3.11 μπορεί να ξαναγραφεί ως εξής:

$$I = \frac{1}{2}(c + \alpha) - \frac{1}{2}(a - c)cos2\theta - \frac{1}{2}bsin2\theta \qquad (3.12)$$

Παραγωγίζοντας ως προς τη γωνία θ, και θέτοντας το αποτέλεσμα ίσο με 0, έχουμε:

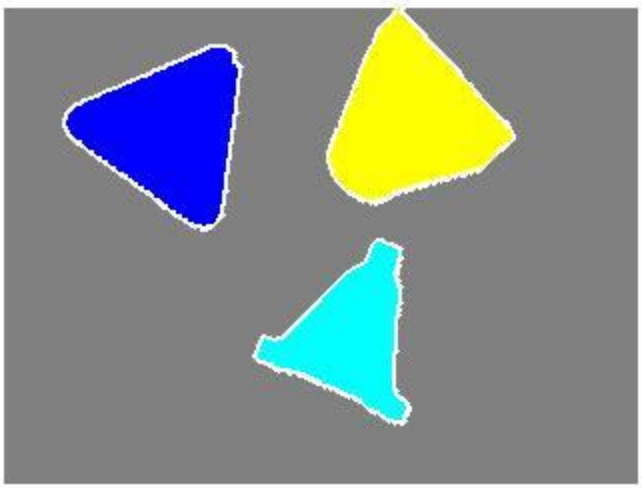
$$(a - c)sin2\theta - bcos2\theta = 0 \qquad (3.13)$$

Αυτό σημαίνει ότι, αν b≠0 και α≠c:

$$tan2\theta = \frac{b}{a-c} \qquad (3.14)$$

Στην περίπτωση που b=0 και α=c το I παραμένει ανεπηρέαστο από τον άξονα προσανατολισμού, που σημαίνει πως το αντικείμενο είναι συμμετρικό ως προς την περιστροφή.

Κάποια ενδεικτικά αποτελέσματα που δίνονται από τις εντολές *regionprops* και *bwboundaries* φαίνονται στα ακόλουθα σχήματα καθώς και τον επόμενο πίνακα.



Σχήμα 3.16: Εύρεση ορίων και αντικειμένων με χρήση της εντολής bwboundaries.

Σχήμα 3.17: Εκτίμηση κέντρων μάζας αντικειμένων με χρήση της εντολής regionprops.

Α/Α αντικειμένου	**Επιφάνεια (pixel x pixel)**	**Περίμετρος (pixel)**	**Προσανατολισμός (μοίρες)**
1	506	305	-23
2	523	297	69
3	521	306	-82

Πίνακας 3.2: Πίνακας με σημαντικά μεγέθη που ανακτήθηκαν μέσω της εντολής regionprops.

ΚΕΦΑΛΑΙΟ 4

ΕΠΙΛΟΓΗ ΜΕΘΟΔΟΥ ΑΝΑΓΝΩΡΙΣΗΣ ΑΝΤΙΚΕΙΜΕΝΩΝ - ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ

Στο κεφάλαιο 2 παρουσιάστηκαν ορισμένες από τις μεθόδους και τεχνικές, που χρησιμοποιούνται στην αναγνώριση αντικειμένων μέσω επεξεργασίας εικόνας. Σε αυτό το κεφάλαιο αφού οριστεί και εξεταστεί το πρόβλημα προς επίλυση, θα παρουσιαστούν αποτελέσματα των μεθόδων αυτών και θα γίνει η τελική επιλογή, η οποία θα οδηγεί σε όσο το δυνατόν πιο εύρωστη και αποτελεσματική επίλυση του προβλήματος.

4.1 Εξέταση του προβλήματος

Το πρόβλημα που εξετάζεται είναι ο εντοπισμός της θέσης μηχανουργικών αντικειμένων πάνω σε οριζόντια επιφάνεια με χρήση ψηφιακής επεξεργασίας εικόνων, που έχουν ληφθεί από τα αναφερθέντα μηχανουργικά αντικείμενα. Στην παρούσα διπλωματική εξετάζονται μοντέλα που προορίζονται για τη δημιουργία καλουπιών χύτευσης, καθώς παρουσιάζουν τις ιδιότητες που επιθυμούμε, ώστε το πρόβλημα να μπορεί να γενικευτεί και σε άλλα μηχανουργικά αντικείμενα.

Συγκεκριμένα, δεν είναι «επίπεδα» κομμάτια, που σημαίνει πως οι τρεις διαστάσεις τους είναι συγκρίσιμες μεταξύ τους. [38] Τα συγκεκριμένα αντικείμενα δεν μπορούν να διαχωριστούν με βάση κάποια άλλη ιδιότητα τους, αφού είναι κομμάτια συγκρίσιμου χρώματος, βάρους και μεγέθους.

Το πρόβλημα περιλαμβάνει την αναγνώριση των παραπάνω αντικειμένων υπό τυχαία τοποθέτηση επάνω στην επιφάνεια. Αυτό σημαίνει πως, επειδή τα αντικείμενα δεν είναι επίπεδα, μπορούν να τοποθετηθούν υπό οποιαδήποτε βάση τους επάνω στην επιφάνεια. Μετά την επιτυχή αναγνώριση του εκάστοτε αντικειμένου θα εξάγονται οι συντεταγμένες της θέσης του στην επιφάνεια και του προσανατολισμού του. [39] Σκοπός είναι η παραλαβή του αναγνωρισθέντος αντικειμένου από ρομποτικό βραχίονα για περαιτέρω επεξεργασία. ([40], [41])

Για την επίλυση θα εξεταστούν δύο διατάξεις. Η πρώτη διάταξη είναι παρόμοια με αυτή που αναπτύχθηκε κατά την εκπόνηση της μεταπτυχιακής εργασίας του Ν. Πεισμάνη. [42] Η δεύτερη πρόκειται για μια διάταξη στην οποία η εικόνα

αναζήτησης θα λαμβάνεται μέσω scanner. Οι δύο διατάξεις θα αναλυθούν εκτενέστερα στη συνέχεια.

Θεωρούμε ότι τα αντικείμενα που βρίσκονται στην οριζόντια επιφάνεια είναι όλα γνωστά εκ των προτέρων και ενταγμένα σε μια βάση δεδομένων. Φυσικά, τα αντικείμενα της βάσης δεδομένων δεν είναι αναγκαίο να βρίσκονται όλα στην εικόνα αναζήτησης. Επίσης ένα αντικείμενο της βάσης δεδομένων μπορεί να εμφανίζεται παραπάνω από μια φορές στην εικόνα αναζήτησης. Τέλος, η επεξεργασία γίνεται σε μεμονωμένες φωτογραφίες και όχι σε ροή εικόνας (βίντεο), που σημαίνει πως σε κάθε λήψη τα αντικείμενα είναι ακινητοποιημένα και η οποιαδήποτε μετακίνησή τους συμβαίνει ανάμεσα στις λήψεις.

4.2 Αρχιτεκτονική επίλυσης

Το μοντέλο επίλυσης υλοποιήθηκε στο λογισμικό της Mathworks, Matlab. Ένα γενικό διάγραμμα ροής για την εξαγωγή αποτελεσμάτων είναι το εξής:

- *Εισαγωγή εικόνων στο σύστημα*

Η εισαγωγή εικόνων στο σύστημα περιλαμβάνει δύο ειδών εικόνες. Οι πρώτες έχουν να κάνουν με εικόνες που συνιστούν μια βάση δεδομένων από τα αντικείμενα προς αναζήτηση. Αυτές αφορούν φωτογραφίες αντικειμένων στην οριζόντια επιφάνεια υπό τις πιθανές θέσεις/βάσεις τοποθέτησής τους. Η βάση δεδομένων μπορεί να δημιουργείται σε οποιαδήποτε στιγμή (offline) και περιλαμβάνει εκτός από την εισαγωγή των εικόνων, το προϊόν της επεξεργασίας τους (επόμενο βήμα).

Το δεύτερο είδος εικόνων έχει να κάνει με την εικόνα αναζήτησης. Την εικόνα, δηλαδή, όπου βρίσκονται τοποθετημένα αρκετά από τα αντικείμενα προς αναζήτηση. Οι εικόνες αυτές λαμβάνονται και επεξεργάζονται offline. Η επεξεργασία τους έχει να κάνει με τις συνθήκες φωτισμού και τον τρόπο λήψης τους.

- *Επεξεργασία εικόνων (όπως αναλύθηκε στο προηγούμενο κεφάλαιο)*
- *Χρήση μεθόδων αναγνώρισης αντικειμένου*
- *Επεξεργασία αποτελεσμάτων*

Οι αντιστοιχίσεις μεταξύ των αντικειμένων που δίνονται από την εκάστοτε μέθοδο επεξεργάζονται ώστε να αποφευχθούν τυχόν λάθη, πριν την τελική εξαγωγής τους.

- *Εξαγωγή αποτελεσμάτων*

Η εξαγωγή των αποτελεσμάτων περιλαμβάνει την εξαγωγή των πραγματικών συντεταγμένων της θέσης και προσανατολισμού ως προς ένα σύστημα αναφοράς. Αρχικά η εξαγωγή γίνεται στην οθόνη και στη συνέχεια, σε ένα αρχείο που είναι δυνατό να αξιοποιηθεί από ένα σύστημα οδήγησης ρομποτικού βραχίονα.

4.3 Διάταξη λήψης εικόνων μέσω ψηφιακής κάμερας

Η διάταξη αυτή περιγράφεται αναλυτικά στην μεταπτυχιακή διπλωματική εργασία του Ν. Πεισμάνη.[42]Η οριζόντια επιφάνεια στην οποία τοποθετούνται τα αντικείμενα, αποτελείται από ξύλινο ορθογώνιο κουτί με την άνω πλευρά του καλυμμένη με επιφάνεια γυαλιού, διαστάσεων 420 x 150 mm (Σχήμα 4.1). Η ψηφιακή κάμερα στηρίζεται από ορθοστάτη, σε οριζόντια θέση παράλληλα στο επίπεδο της γυάλινης επιφάνειας και σε ύψος 350 mm πάνω από το κέντρο της επιφάνειας.

Σχήμα 4.1: Διάταξη λήψης εικόνων μέσω κάμερας.

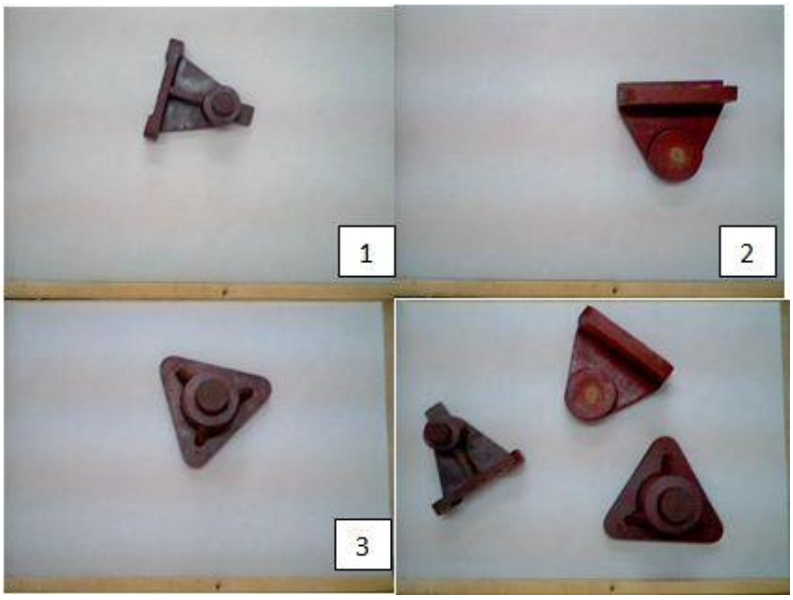
Στη συνέχεια παρουσιάζονται τα αποτελέσματα ορισμένων από τις μεθόδους που χρησιμοποιήθηκαν για την επίλυση του προβλήματος.

4.3.1 Αναγνώριση αντικειμένων μέσω γεωμετρικών μετασχηματισμών

Η λογική αυτής της μεθόδου παρουσιάστηκε εκτενώς στο κεφάλαιο 2. Ένα από τα βασικά πλεονεκτήματά της είναι ότι δεν απαιτούνται αυστηρά ορισμένες συνθήκες φωτισμού κατά τη λήψη των εικόνων. Επίσης, δεν απαιτείται υψηλής ανάλυσης ψηφιακή κάμερα. Η ύπαρξη μιας σχετικά λείας, μονόχρωμης επιφάνειας τοποθέτησης, με χρωματική αντίθεση από τα αντικείμενα προς αναζήτηση, είναι αρκετή για την εξαγωγή ικανοποιητικών αποτελεσμάτων.

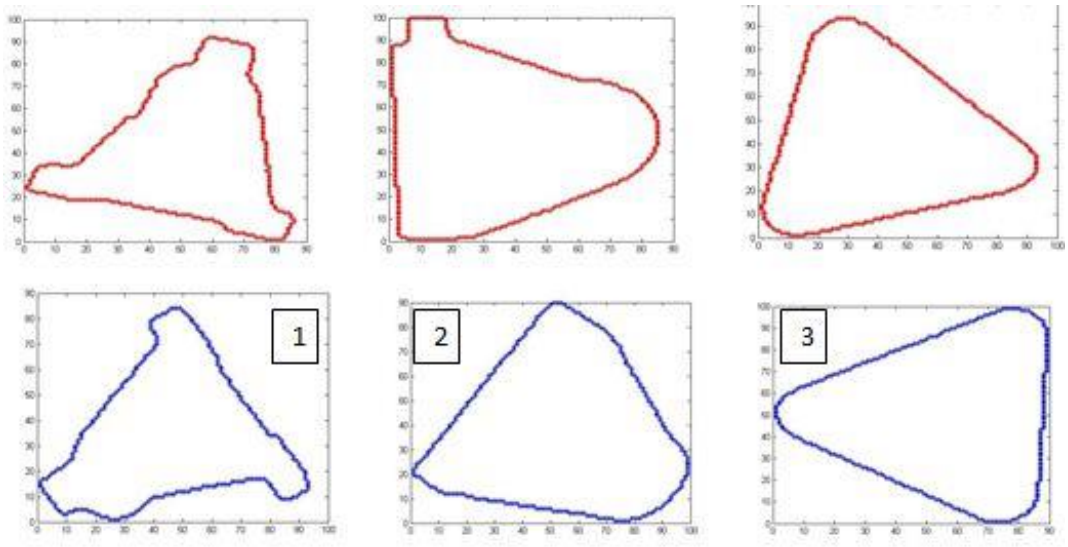
Φυσικά η μέθοδος αυτή, είναι επιρρεπής στις αποκρύψεις ορισμένων χαρακτηριστικών (occlusion) των αντικειμένων, εξαιτίας της προοπτικής προβολής. Συνεπώς, η μέθοδος είναι ιδανική για αναγνώριση «επίπεδων» κομματιών, ενώ στη δική μας περίπτωση θα υπάρχουν κάποιες ανοχές ως προς το επικείμενο σφάλμα, το οποίο επιβάλλουν τυχόν αποκρύψεις χαρακτηριστικών, λόγω ύψους των αντικειμένων. Αναφέρθηκε τέλος ότι ο χρόνος επεξεργασίας και εξαγωγής αποτελεσμάτων είναι σχετικά μεγάλος, καθώς το κάθε αντικείμενο της βάσης δεδομένων, πρέπει να συγκριθεί με κάθε αντικείμενο της εικόνας. Παρακάτω παρουσιάζεται βήμα-βήμα η υλοποίηση της μεθόδου.

Αρχικά εισάγονται οι εικόνες των αντικειμένων που συνιστούν τη βάση δεδομένων, και η εικόνα αναζήτησης στο σύστημα, με σκοπό την προεπεξεργασία τους. Ενδεικτικά στο επόμενο σχήμα εμφανίζονται οι εικόνες που χρησιμοποιήθηκαν ως είσοδοι στο σύστημα.



Σχήμα 4.2: Εικόνες που δίνονται ως είσοδοι στο σύστημα. Τα αντικείμενα είναι αριθμημένα με τη σειρά καταχώρησής τους στη βάση δεδομένων. Η εικόνα κάτω αριστερά είναι η εικόνα αναζήτησης.

Η προεπεξεργασία των εικόνων περιλαμβάνει τις μεθόδους που αναλύθηκαν στο προηγούμενο κεφάλαιο. Μετά την επεξεργασία τους διατηρείται το περίγραμμα των αντικειμένων της βάσης δεδομένων σε δυαδική εικόνα. Η επεξεργασία της εικόνας αναζήτησης περιλαμβάνει δύο στάδια. Κατά το πρώτο βρίσκονται τα περιγράμματα των αντικειμένων στην εικόνα, η θέση και ο προσανατολισμός τους, ενώ κατά το δεύτερο απομονώνεται το κάθε αντικείμενο, ώστε στη συνέχεια να συγκριθεί με όλα τα αντικείμενα που είναι καταχωρημένα στη βάση δεδομένων. Μετά την προεπεξεργασία λοιπόν των εικόνων λαμβάνουμε τα εξής αποτελέσματα:



Σχήμα 4.3: Με κόκκινο παρουσιάζονται τα σχήματα των εικόνων που είναι καταχωρημένα στη βάση δεδομένων, ενώ με μπλέ τα σχήματα που απομονώθηκαν στην εικόνα αναζήτησης. Η αρίθμηση των δεύτερων αφορά τη σειρά προσπέλασής τους από το λογισμικό.

Από το στάδιο αυτό ανακτήσαμε ορισμένα μεγέθη που είναι χρήσιμα για τους υπολογισμούς στη συνέχεια. Αυτά είναι ο προσανατολισμός των σχημάτων της βάσης δεδομένων αλλά και αυτών της εικόνας αναζήτησης καθώς και η θέση των δεύτερων στην εικόνα. Ο προσανατολισμός και η θέση βρέθηκαν μέσω της εντολής regionprops. Στο σημείο αυτό κρίνεται απαραίτητο να υπενθυμιστεί πως ο προσανατολισμός που δίνεται από την εντολή αυτή ενδεχομένως να είναι λάθος αν το σχήμα μας είναι συμμετρικό ως προς την περιστροφή. Αυτό συμβαίνει τουλάχιστον για ένα αντικείμενο στην περίπτωσή μας, το περίγραμμα του οποίου είναι ισόπλευρο τρίγωνο (αντικείμενο #3). Συνεπώς, η εντολή σ αυτήν την περίπτωση μπορεί να δώσει λάθος προσανατολισμό μεταξύ αυτών των δυο αντίστοιχων σχημάτων. Επίσης εξαιτίας του υπολογισμού του τόξου εφαπτομένης από το λογισμικό ο προσανατολισμός δίνεται από -90^0 έως 90^0. Ενδεχομένως, λοιπόν, στον

προσανατολισμό μεταξύ δυο αντίστοιχων σχημάτων να υπάρχει διαφορά 180^0. Στον πίνακα 4.1 φαίνονται οι προσανατολισμοί για τα σχήματα των αντικειμένων της βάσης δεδομένων, ενώ στον πίνακα 4.2 οι θέσεις και οι προσανατολισμοί των αντικειμένων στην εικόνα αναζήτησης.

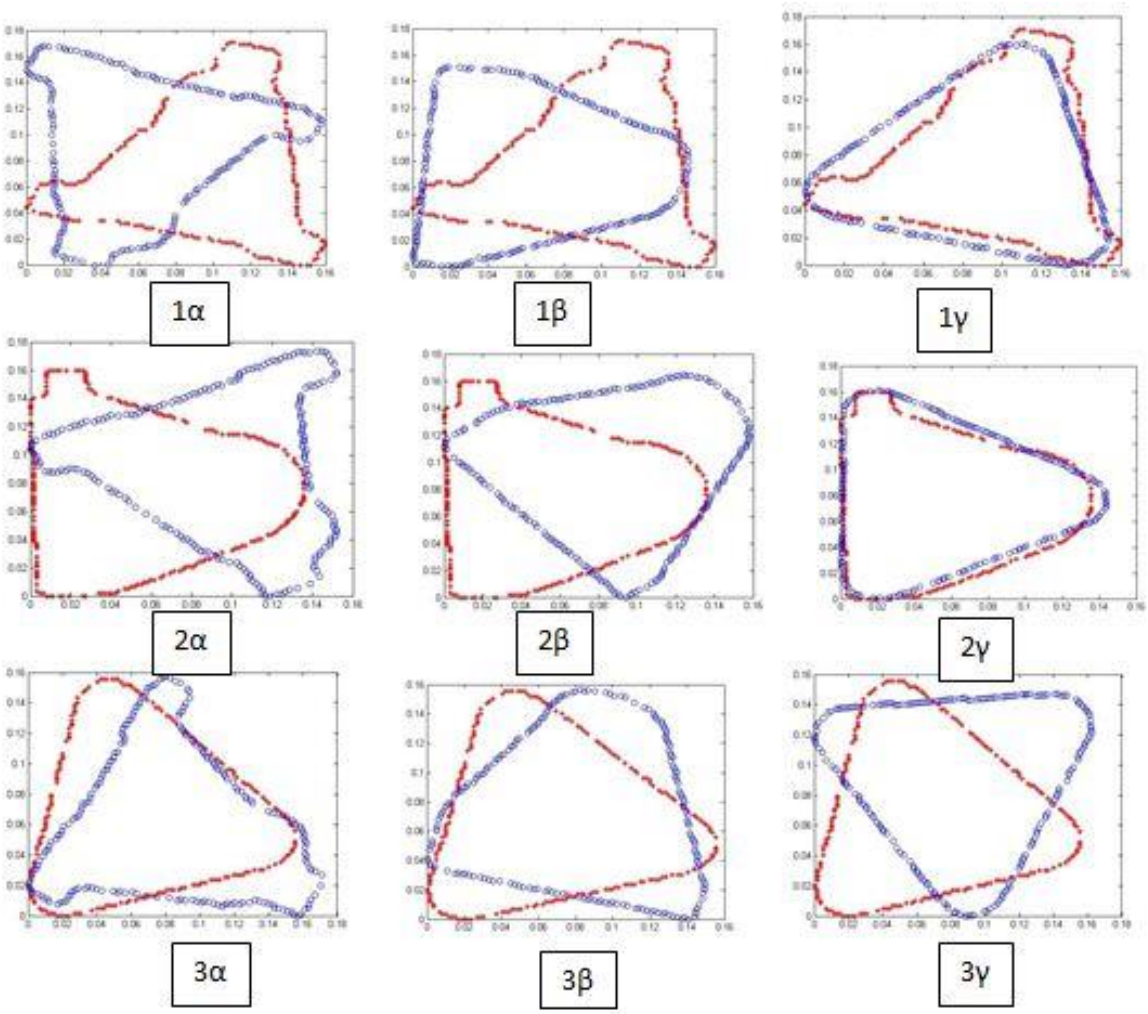
Αντικείμενο Βάσης Δεδομένων	***Προσανατολισμός σε ^{0} (μοίρες)***
Αντικείμενο #1	-26
Αντικείμενο #2	+18
Αντικείμενο #3	+53

Πίνακας 4.1: Προσανατολισμός αντικειμένων βάσης δεδομένων

Αντικείμενο Εικόνας Αναζήτησης	***Προσανατολισμός σε ^{0} (μοίρες)***	***Θέση Κέντρου Μάζας (συντεταγμένες εικόνας)***
Αντικείμενο #1	-64	(55, 131)
Αντικείμενο #2	-58	(175, 58)
Αντικείμενο #3	-43	(219, 166)

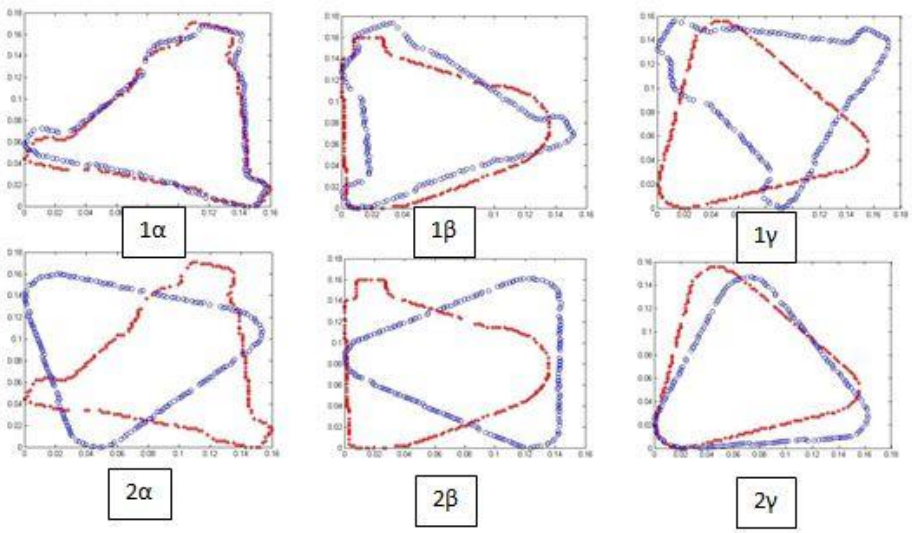
Πίνακας 4.2: Προσανατολισμός και θέση αντικειμένων στην εικόνα αναζήτησης

Εφόσον έχουμε τα παραπάνω δεδομένα, εφαρμόζουμε γεωμετρικούς μετασχηματισμούς, όπως αναλύθηκε στο κεφάλαιο 2, σε κάθε σχήμα αντικειμένου, που είναι τοποθετημένο στην εικόνα αναζήτησης, ώστε να συγκριθεί με κάθε σχήμα αντικείμενου της βάσης δεδομένων. Εφόσον όλα τα αντικείμενα έχουν συγκριθεί γίνεται έλεγχος για ποια αντικείμενα υπάρχει αντιστοίχιση και βρίσκεται η θέση τους στην εικόνα αναζήτησης. Για το παράδειγμα που εξετάζεται δίνονται τα αντίστοιχα αποτελέσματα στα ακόλουθα σχήματα.



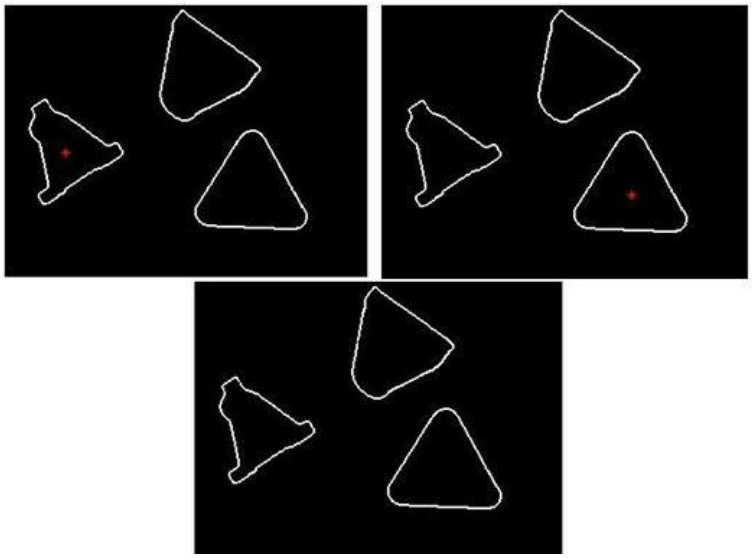
Σχήμα 4.4: Με κόκκινο παρουσιάζονται τα σχήματα των εικόνων που είναι καταχωρημένα στη βάση δεδομένων, ενώ με μπλε τα σχήματα που απομονώθηκαν στην εικόνα αναζήτησης. Κάθε σχήμα της βάσης δεδομένων συγκρίνεται με κάθε σχήμα της εικόνας αναζήτησης (9 συγκρίσεις συνολικά).

Φαίνεται πως η μόνη πιθανή αντιστοίχιση είναι μεταξύ του αντικειμένου 2 της βάσης δεδομένων και του αντικειμένου 3 της εικόνας αναζήτησης. Είναι προφανές πως μια τέτοια αντιστοίχιση πρέπει να αποφευχθεί, καθώς τα δύο αντικείμενα στην πραγματικότητα είναι διαφορετικά. Ένα άλλο που μπορούμε να παρατηρήσουμε είναι πως ο προσανατολισμός όπως εξήχθη από την εντολή regionprops σε όλες τις περιπτώσεις πραγματικής αντιστοιχίας είναι λανθασμένος. Αυτό όπως αναφέρθηκε μπορεί να οφείλεται στον τρόπο υπολογισμού του τόξου εφαπτομένης ή στη φύση των αντικειμένων που χρησιμοποιούνται (συμμετρικά ως προς την περιστροφή). Για το λόγο αυτό εισάγουμε ένα επιπλέον βήμα στον αλγόριθμο. Για όποια αντικείμενα δεν υπήρξε αντιστοιχία να επαναλάβει τη διαδικασία με 180^0 επιπλέον στον προσανατολισμό.



Σχήμα 4.5: Για όσα σχήματα δεν υπήρξε αντιστοιχία, επαναλαμβάνουμε τη διαδικασία με προσανατολισμό 180^0 αυτή τη φορά. Παρατηρούμε ότι όντως στην περίπτωση 1α υπήρχε λάθος στον προσανατολισμό που εξήχθη από την εντολή regionprops.

Μετά από αυτό το βήμα ο αλγόριθμος αντιστοιχίζει τα αντικείμενα και εξάγει τις συντεταγμένες τους στην εικόνα αναζήτησης.



Σχήμα 4.6: Αντιστοίχιση αναγνωρισθέντων αντικειμένων στην εικόνα αναζήτησης.

Από το σχήμα 4.6 παρατηρούμε πως το αντικείμενο 1 της βάσης δεδομένων έχει αντιστοιχηθεί με το ομόλογό του στην εικόνα αναζήτησης. Το αντικείμενο 2 έχει αντιστοιχηθεί με το αντικείμενο 3 στην εικόνα αναζήτησης λανθασμένα. Ενώ το αντικείμενο 3 της βάσης δεδομένων δεν αντιστοιχήθηκε με κάποιο. αντιστοιχήθηκε με κάποιο της εικόνας αναζήτησης.

Η λανθασμένη αντιστοίχιση του δεύτερου αντικειμένου δεν αποτελεί κρίσιμο πρόβλημα καθώς μικραίνοντας λίγο τις ανοχές και ορίζοντας επιπλέον συνθήκες

αντιστοίχισης μπορεί να επιλυθεί. Το μεγαλύτερο πρόβλημα όπως διαφαίνεται είναι ο καθορισμός του προσανατολισμού των αντικειμένων, καθώς το μοντέλο που χρησιμοποιεί η εντολή regionprops δεν μπορεί να χρησιμοποιηθεί για τα αντικείμενά μας, καθώς αρκετά από αυτά είναι συμμετρικά ως προς την περιστροφή (2 σε αυτό το παράδειγμα). Τέλος, ακόμα κι αν υπήρχε καλύτερος τρόπος υπολογισμού του προσανατολισμού, είναι εμφανής η επίδραση της προοπτικής προβολής στο περίγραμμα του αντικειμένου 2. Τα χαρακτηριστικά του αλλοιώνονται κατά πολύ, συνεπώς μειώνοντας τις ανοχές για την αποφυγή λανθασμένης αντιστοίχισης είναι πολύ πιθανόν ακόμα και υπό τον ίδιο προσανατολισμό η αντιστοίχιση του αντικειμένου #2 της βάσης δεδομένων με το ομόλογό του στην εικόνα αναζήτησης να απορριφθεί.

Όσον αφορά τον χρόνο εκτέλεσης του προγράμματος για το συγκεκριμένο μέγεθος και αριθμό εικόνων, κυμαίνεται μεταξύ 40-50sec συμπεριλαμβάνοντας, όμως, και το κτίσιμο της βάσης δεδομένων.

Λαμβάνοντας υπόψη όλα τα παραπάνω, κρίθηκε πως η παραπάνω μέθοδος για το συγκεκριμένο πρόβλημα που εξετάζεται δεν μπορεί να οδηγήσει σε αξιόλογα αποτελέσματα γεγονός που οδήγησε σε αναζήτηση και εφαρμογή άλλων τεχνικών.

4.3.2 Αναγνώριση αντικειμένων μέσω μεθόδων αναγνώρισης μοτίβου

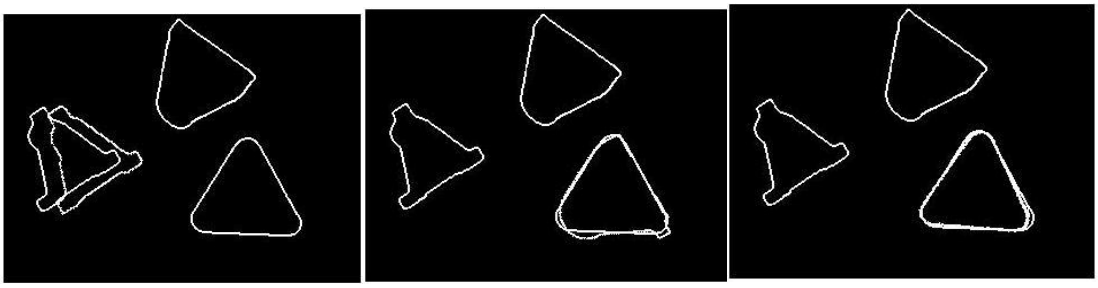
Ο μη αξιόπιστος καθορισμός του προσανατολισμού για τα αντικείμενα που εξετάζονται, οδήγησε στην αναζήτηση μεθόδων αναγνώρισης κατά τις οποίες δεν απαιτείται ο εκ των προτέρων ορισμός του.

Οι μέθοδοι αναγνώρισης μοτίβου όπως περιγράψαμε δεν πληρούν αυτή τη συνθήκη, αφού αναφέρθηκε ότι τα αποτελέσματα είναι άμεσα εξαρτώμενα από τον προσανατολισμό των αντικειμένων. Έχουν, όμως το πλεονέκτημα ότι είναι αρκετές φορές γρηγορότεροι από τις μεθόδους γεωμετρικών μετασχηματισμών. Συνεπώς, αντί να καθορίζουμε κάθε φορά τη διαφορά των προσανατολισμών μεταξύ των τεμαχίων της βάσης δεδομένων και της εικόνας, απλά μπορούμε να περιστρέφουμε την εικόνα αναζήτησης με ένα σταθερό ρυθμό για κάθε εικόνα της βάσης, μέχρι να πετύχουμε το καλύτερο μέτρο αντιστοίχισης για κάθε εικόνα. Εφόσον όμως, δεν είναι υποχρεωτικό κάθε εικόνα της βάσης να βρίσκεται στην εικόνα αναζήτησης, αφού ολοκληρωθεί η πρώτη αντιστοίχιση ορίζουμε κάποιες συνθήκες προκειμένου να βελτιώσουμε τα τελικά αποτελέσματα.

Το πρώτο στάδιο υλοποίησης περιλαμβάνει την είσοδο και προεπεξεργασία των εικόνων όπως ακριβώς και προηγουμένως. Κατά συνέπεια, αναμένουμε τα προβλήματα, που δημιουργεί η προοπτική προβολή, να παρουσιαστούν ξανά. Το δεύτερο στάδιο διαφοροποιείται ανάλογα με το μέτρο της αντιστοίχισης που χρησιμοποιούμε. Στην παρούσα εργασία μελετήθηκε η αναγνώριση μοτίβου μέσω της κανονικοποιημένης ετεροσυσχέτισης και του γενικευμένου μετασχηματισμού Hough.

4.3.2.1 Κανονικοποιημένη Ετεροσυσχέτιση

Χρησιμοποιώντας τις εικόνες του σχήματος 4.2 ως εισόδους, τα αποτελέσματα που λαμβάνονται παρουσιάζονται στα ακόλουθα σχήματα.



Σχήμα 4.7: Πρώτη αντιστοίχιση αντικειμένων μετρώντας την κανονικοποιημένη ετεροσυσχέτιση.

Στο σχήμα 4.7 παρατηρούμε πως το αντικείμενο 1 της βάσης δεδομένων έχει αντιστοιχηθεί σωστά με το αντικείμενο 1 της εικόνας, όπως και το αντικείμενο 3 της βάσης δεδομένων με το αντικείμενο 3 της εικόνας αναζήτησης. Για το αντικείμενο 2, λόγω της επίδρασης της προοπτικής προβολής, η καλύτερη αντιστοίχησή του στην εικόνα αναζήτησης είναι πλέον με το αντικείμενο 3. Όπως προαναφέρθηκε, μετά από αυτή την πρώτη αντιστοίχιση ορίζονται κάποιες συνθήκες πριν την εξαγωγή των τελικών αποτελεσμάτων. Με αυτές πρέπει να καταφέρουμε το αντικείμενο 3 να αντιστοιχηθεί με το ομόλογό του, ενώ το αντικείμενο 2 να μην αντιστοιχηθεί λανθασμένα με κάποιο.

Ελέγχοντας τις συνθήκες λαμβάνουμε τον εξής πίνακα αντιστοίχισης από το πρόγραμμα:

	Αντιστοιχεί με αντικείμενο εικόνας:		
Αντικείμενο # 1 (Βάσης Δεδομένων)	1	0	0
Αντικείμενο # 2 (Βάσης Δεδομένων)	0	0	0
Αντικείμενο # 3 (Βάσης Δεδομένων)	0	0	3

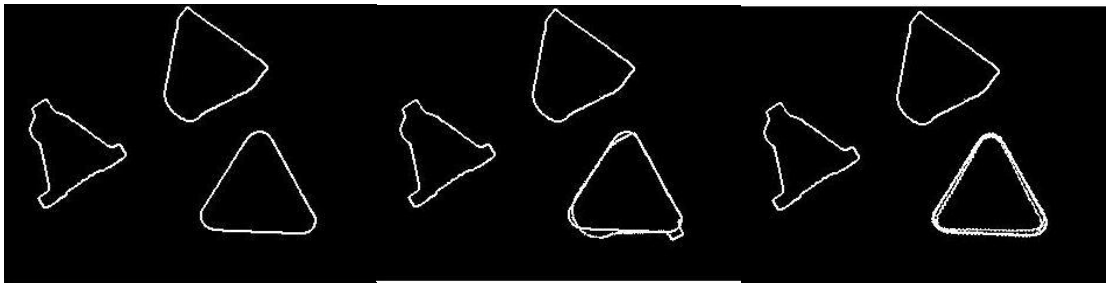
Πίνακας 4.3: Πίνακας αντιστοίχισης που εξάγεται από το πρόγραμμα.

Ο παραπάνω πίνακας έχει γραμμές όσες και τα αντικείμενα της βάσης δεδομένων και στήλες όσες τα αντικείμενα της εικόνας. Τα σημειωμένα του κελιά δεν μπορούν να είναι παραπάνω από τα αντικείμενα που βρίσκονται στην εικόνα, ενώ η κάθε γραμμή μπορεί να έχει παραπάνω από μία εκχωρήσεις. Αν τα σημειωμένα κελιά είναι λιγότερα από τα αντικείμενα αναζήτησης, όπως συμβαίνει τώρα, σημαίνει πως κάποια από τα αντικείμενα της εικόνας δεν έχουν βρει αντιστοίχιση.

Ο χρόνος εκτέλεσης του προγράμματος κυμαίνεται μεταξύ 18-25 sec, συμπεριλαμβανομένου ξανά του καθορισμού της βάσης δεδομένων. Μειώνοντας στο μισό το χρόνο εκτέλεσης έχουμε έναν αρκετά πιο αξιόπιστο αλγόριθμο για την περίπτωση μας, ο οποίος για να εφαρμοστεί έχει πολύ μικρές απαιτήσεις όσον αφορά το φωτισμό και τις απαιτήσεις σε ανάλυση. Ενδεικτικά η ψηφιακή κάμερα που χρησιμοποιήθηκε για την εξαγωγή αυτών των αποτελεσμάτων έχει ικανότητα ανάλυσης 240x320 pixel, ενώ ο φωτισμός των αντικειμένων προέρχεται από το διάχυτο φωτισμό του δωματίου.

4.3.2.2 Γενικευμένος μετασχηματισμός Hough

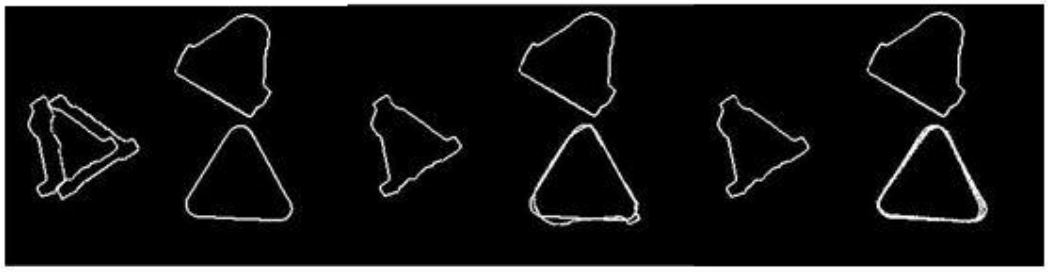
Παρόμοια αποτελέσματα έχει να επιδείξει και η υλοποίηση της μεθόδου με το γενικευμένο μετασχηματισμό Hough, όπως φαίνεται και στο σχήμα 4.8, όμως εκ των δύο υλοποιήσεων προτιμήθηκε αυτή της κανονικοποιημένης ετεροσυσχέτισης. Αυτό συνέβη λόγω του ότι το μέτρο της αντιστοίχισης εκφράζεται σε ένα μέγεθος που λαμβάνει τιμές στο διάστημα 0-1, δίνοντας τη δυνατότητα ευκολότερου χειρισμού των αποτελεσμάτων.



Σχήμα 4.8: Πρώτη αντιστοίχιση αντικειμένων με χρήση του γενικευμένου μετασχηματισμού Hough.

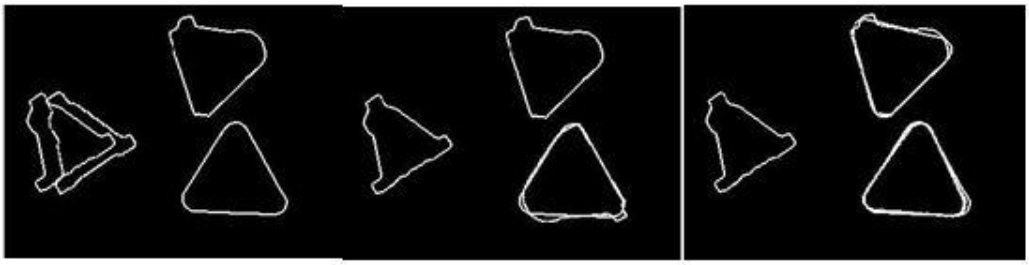
Τέλος, οι χρόνοι υλοποίησης των δύο μοντέλων είναι παρόμοιοι με τη διαφορά πως η αξιοπιστία των αποτελεσμάτων της υλοποίησης βάση της κανονικοποιημένης ετεροσυσχέτισης είναι σχετικά μεγαλύτερη.

Στα παρακάτω σχήματα παρουσιάζονται αντιστοιχίσεις που προέκυψαν σε διαφορετικές περιπτώσεις με τη μέθοδο της κανονικοποιημένης ετεροσυσχέτισης.



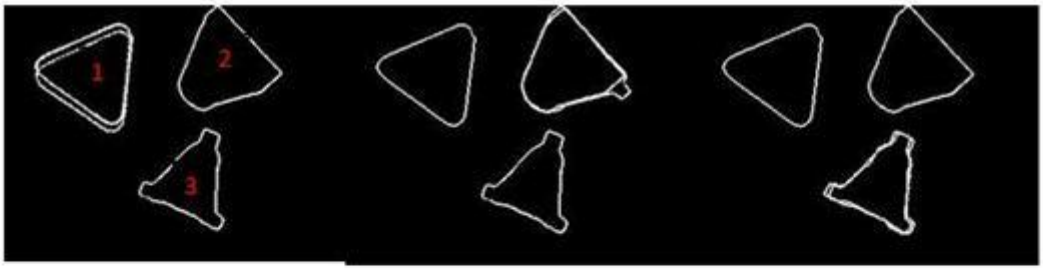
	Αντιστοιχεί με αντικείμενο εικόνας:		
Αντικείμενο # 1 (Βάσης Δεδομένων)	1	0	0
Αντικείμενο # 2 (Βάσης Δεδομένων)	0	0	0
Αντικείμενο # 3 (Βάσης Δεδομένων)	0	0	3

Σχήμα 4.9: Πρώτη και τελική αντιστοίχιση αντικειμένων μετρώντας την κανονικοποιημένη ετεροσυσχέτιση.



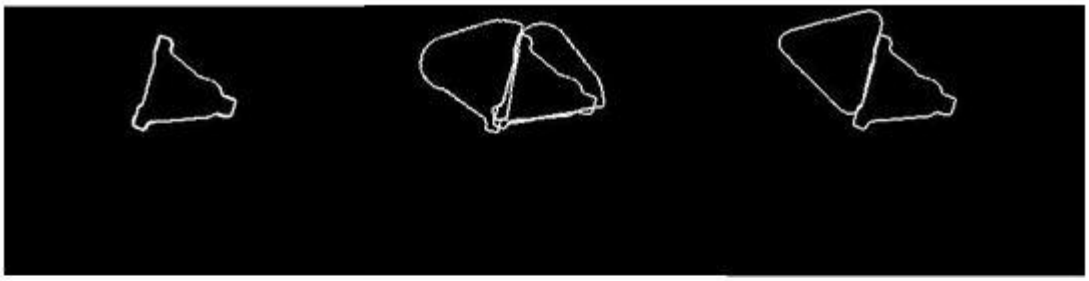
	Αντιστοιχεί με αντικείμενο εικόνας:		
Αντικείμενο # 1 (Βάσης Δεδομένων)	1	0	0
Αντικείμενο # 2 (Βάσης Δεδομένων)	0	0	0
Αντικείμενο # 3 (Βάσης Δεδομένων)	0	0	3

Σχήμα 4.10: Πρώτη και τελική αντιστοίχιση αντικειμένων μετρώντας την κανονικοποιημένη ετεροσυσχέτιση.



	Αντιστοιχεί με αντικείμενο εικόνας:		
Αντικείμενο # 1 (Βάσης Δεδομένων)	0	0	3
Αντικείμενο # 2 (Βάσης Δεδομένων)	0	2	0
Αντικείμενο # 3 (Βάσης Δεδομένων)	1	0	0

Σχήμα 4.11: Πρώτη και τελική αντιστοίχιση αντικειμένων μετρώντας την κανονικοποιημένη ετεροσυσχέτιση.



	Αντιστοιχεί με αντικείμενο εικόνας:
Αντικείμενο # 1 (Βάσης Δεδομένων)	1
Αντικείμενο # 2 (Βάσης Δεδομένων)	0
Αντικείμενο # 3 (Βάσης Δεδομένων)	0

Σχήμα 4.12: Πρώτη και τελική αντιστοίχιση αντικειμένων μετρώντας την κανονικοποιημένη ετεροσυσχέτιση.

Στα σχήματα 4.9 και 4.10 αλλάζουμε τον προσανατολισμό ορισμένων αντικειμένων στην εικόνα αναζήτησης για να δούμε πως ανταποκρίνεται η μέθοδος. Παρατηρούμε πως πάλι η παραμόρφωση του περιγράμματος εξαιτίας της προοπτικής προβολής οδηγεί στο μη εντοπισμό του αντικειμένου #2 της βάσης δεδομένων. Στο σχήμα 4.10 αλλάζουμε την θέση τοποθέτησης των κομματιών στην εικόνα αναζήτησης. Πλέον η σειρά προσπέλασης των αντικειμένων είναι αυτή που έχει σημειωθεί με κόκκινους αριθμούς. Παρατηρούμε πως το πρόγραμμα κατάφερε να αντιστοιχίσει και τα 3 αντικείμενα σωστά σε αυτήν την περίπτωση. Τέλος, στο σχήμα 4.12 εισάγουμε μια εικόνα της βάσης δεδομένων σαν εικόνα αναζήτησης, για να ελεγχθεί η απόκριση του προγράμματος όταν στην εικόνα αναζήτησης βρίσκονται λιγότερα αντικείμενα από αυτά που είναι καταχωρημένα στη βάση. Όπως αναμενόταν για κάθε αντικείμενο της βάσης υπάρχει μια βέλτιστη πρώτη αντιστοίχιση. Τελικά λαμβάνουμε την καλύτερη από αυτές που είναι για το πρώτο αντικείμενο της βάσης δεδομένων, όπως ήταν αναμενόμενο.

Συμπεραίνουμε, πως η μέθοδος αυτή παράγει αξιόλογα αποτελέσματα σε σχετικά μικρούς χρόνους εκτέλεσης, ειδικά για «επίπεδα» κομμάτια ή κομμάτια για τα οποία η παραμόρφωση εξαιτίας της προοπτικής προβολής είναι σχετικά μικρή.

4.3.3 Αναγνώριση αντικειμένων μέσω ανιχνευτών χαρακτηριστικών σημείων

Αναζητώντας μεθόδους αναγνώρισης που θα μπορούσαν να ξεπεράσουν το πρόβλημα της παραμόρφωσης του περιγράμματος, εξαιτίας της προοπτικής προβολής, οδηγηθήκαμε στην εξέταση της επίλυσης του προβλήματος μέσω της χρήσης ανιχνευτών χαρακτηριστικών σημείων. Όπως αναφέρθηκε και στο κεφάλαιο 2, οι ανιχνευτές δε χρησιμοποιούν το περίγραμμα του αντικειμένου, αλλά την ίδια την εικόνα για την αναγνώριση αντικειμένων σε αυτήν. Είναι αντιληπτό πως πλέον η ανάλυση της εικόνας έχει βαρύνουσα σημασία, όπως και ο φωτισμός των αντικειμένων. Είδαμε πως τα αποτελέσματα των μεθόδων είναι αμετάβλητα ως προς την περιστροφή. Αν τα αντικείμενα ήταν επίπεδα, όπως για παράδειγμα ένα βιβλίο, ο εντοπισμός του υπό διαφορετικούς προσανατολισμούς, λοιπόν, δε θα αποτελούσε πρόβλημα για τη μέθοδο αυτή. Για μη επίπεδα κομμάτια, αλλάζοντας τη θέση ή τον προσανατολισμό του αντικειμένου, αλλάζει η οπτική γωνία ως προς την κάμερα, γεγονός που κάνει τον εντοπισμό του αντικειμένου αρκετά δύσκολο εγχείρημα.

Αρχικά, θέλοντας να δείξουμε πως η ανάλυση της κάμερας επηρεάζει τα αποτελέσματα εκτελούμε το ακόλουθο πείραμα:

Στην αρχή δίνουμε ως είσοδο στον ανιχνευτή SURF μια εικόνα που χρησιμοποιούσαμε στις προηγούμενες περιπτώσεις, ανάλυσης 320x240pixel, στη συνέχεια λαμβάνουμε μια εικόνα με 5-πλάσια ανάλυση και στη συνέχεια μια εικόνα με 10-πλάσια ανάλυση. Παραθέτουμε τα αποτελέσματα στον παρακάτω πίνακα:

Ανάλυση Εικόνας (pixel x pixel)	**Χαρακτηριστικά σημεία**
240x320	28
1200x1600	35
2448x3264	50

Πίνακας 4.4: Αύξηση χαρακτηριστικών σημείων ανάλογα με την ανάλυση της εικόνας.

Παρατηρείται μια αύξηση των χαρακτηριστικών σημείων, αλλά δεν είναι ραγδαία. Το γεγονός αυτό σχετίζεται με τη φύση των αντικειμένων που εξετάζονται. Αν τα αντικείμενά μας περιείχαν ορισμένες αντιθέσεις τότε θα υπήρχαν πολύ περισσότερα χαρακτηριστικά σημεία. Οι εικόνες που χρησιμοποιήθηκαν παρατίθενται στο σχήμα 4.13.

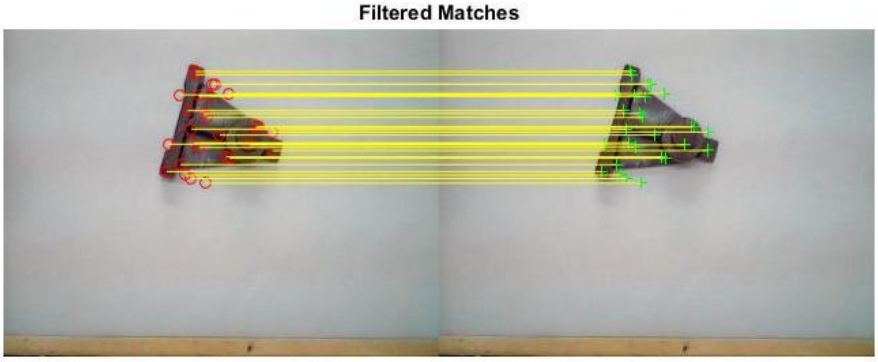
Σχήμα 4.13: Εικόνες που χρησιμοποιήθηκαν για να ελεγχθεί πόσο επηρεάζει η ανάλυση την εύρεση των χαρακτηριστικών σημείων (Παρουσιάζονται με αύξουσα σειρά ως προς την ανάλυση).

Αν είχαμε ένα βιβλίο στη θέση της τρίτης εικόνας τα χαρακτηριστικά σημεία που θα βρισκόταν είναι παραπάνω από 900. Για το βιβλίο του σχήματος 4.14 βρέθηκαν 938 σημεία (100 από τα οποία είναι σημειωμένα στο σχήμα). Όσα περισσότερα σημεία βρεθούν, τόσο μεγαλύτερες πιθανότητες η αλλαγή της οπτικής γωνίας να μην επηρεάσει το τελικό αποτέλεσμα. Συμπεραίνουμε πως τα υπό εξέταση αντικείμενα, δύσκολα θα μπορέσουν να αναγνωριστούν με τη χρήση των συγκεκριμένων μεθόδων.

Σχήμα 4.14: Εύρεση χαρακτηριστικών σημείων σε ένα τυπικό εξώφυλλο βιβλίου και σημείωση των 100 πρώτων σε αυτό.

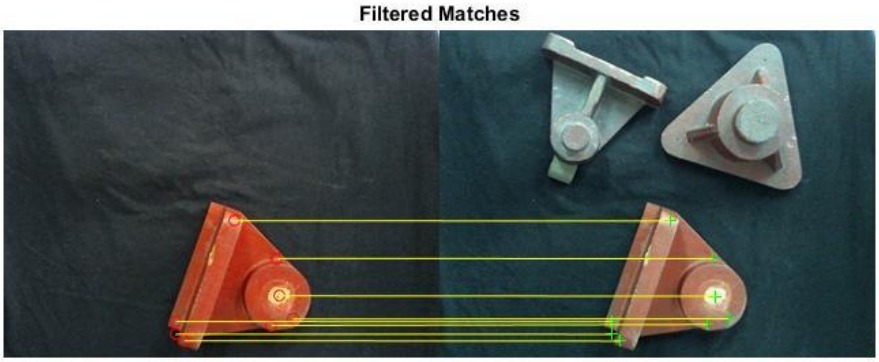
4.3.3.1 Χρήση αλγόριθμου SURF

Στην αρχή γίνεται σύγκριση δύο ίδιων εικόνων και παρουσιάζονται τα αποτελέσματα. Όπως αναμενόταν όλα τα χαρακτηριστικά σημεία (συνολικά 28) που βρέθηκαν σε κάθε εικόνα αντιστοιχήθηκαν μεταξύ τους.



Σχήμα 4.15: Αναγνώριση αντικειμένου μέσω του αλγόριθμου SURF.

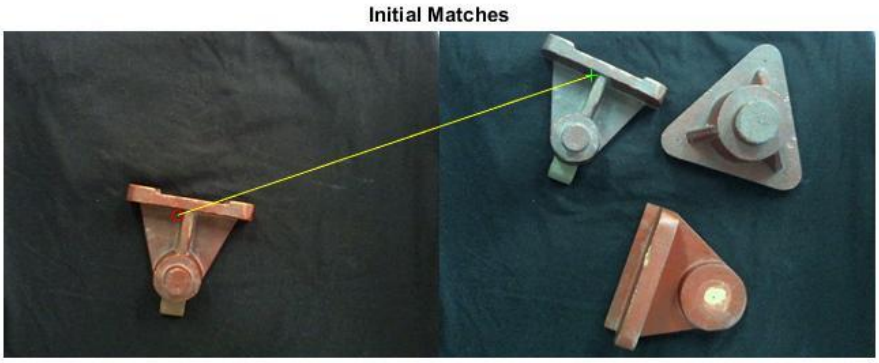
Στη συνέχεια, αφήνοντας τη θέση του αντικειμένου ακέραια και τον προσανατολισμό του, προσθέτουμε στην εικόνα αναζήτησης περισσότερα αντικείμενα. Στο σχήμα 4.16 παρουσιάζουμε τα αποτελέσματα.



Σχήμα 4.16: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SURF.

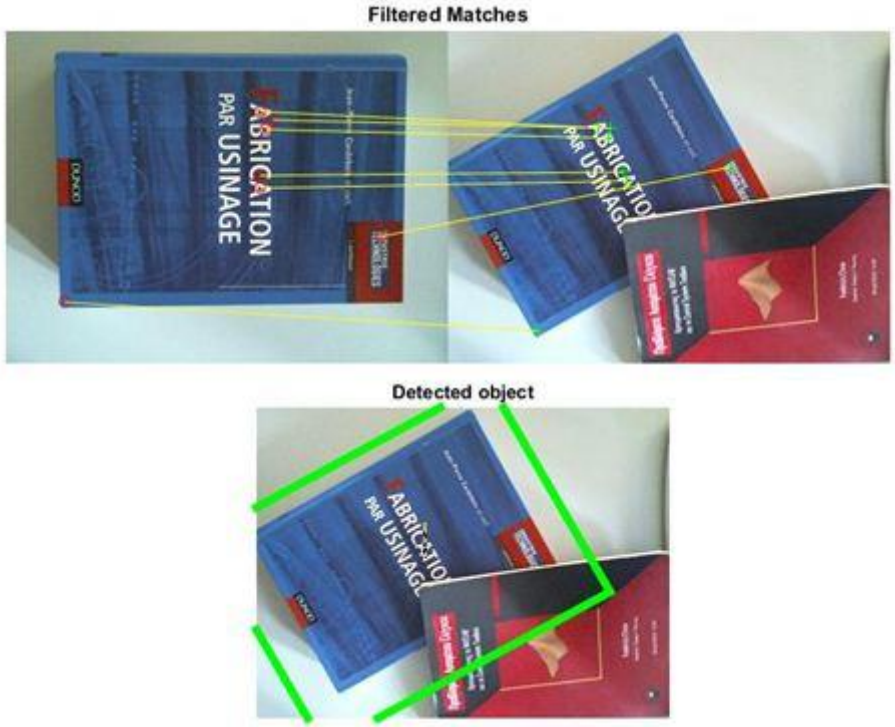
Στην εικόνα του αντικειμένου της βάσης δεδομένων βρέθηκαν 11 σημεία, ενώ στην εικόνα αναζήτησης 54 συνολικά σημεία. Ανάμεσα στις δύο εικόνες βρέθηκαν 8 αντιστοιχίσεις. Βλέπουμε πως η αντιστοίχιση των σημείων έγινε σωστά (κάθε γραμμή αντιστοίχησης παράλληλα με τις άλλες) και το αντικείμενο αναγνωρίστηκε στη σκηνή, παρόλο που υπήρχαν άλλα αντικείμενα.

Το επόμενο βήμα είναι να ελέγξουμε τα αποτελέσματα της διαδικασίας όταν η θέση του αντικειμένου είναι τυχαία καθώς και ο προσανατολισμός του.



Σχήμα 4.16: Αναγνώριση αντικειμένου ανάμεσα σε άλλα (υπό τυχαίο προσανατολισμό) μέσω του αλγόριθμου SURF.

Στο σχήμα 4.16 παρατηρούμε πως πλέον μεταβάλλοντας τον προσανατολισμό και τη θέση του αντικειμένου στην εικόνα αναζήτησης οι αντιστοιχίσεις μεταξύ των δυο εικόνων μειώνονται ραγδαία. Στη συγκεκριμένη περίπτωση από τα αρχικά 28 σημεία της εικόνας του αντικειμένου και τα 55 σημεία της εικόνας αναζήτησης αντιστοιχήθηκε μόνο το 1. Αν η θέση ή ο προσανατολισμός του αντικειμένου διέφερε ακόμα περισσότερο στις δύο εικόνες δε θα υπήρχε καμία αντιστοίχιση. Ενδιαφέρον έχει να ελέγξουμε τι γίνεται στην περίπτωση που είχαμε αρκετά μεγαλύτερο αριθμό σημείων, όπως στην περίπτωση του εξώφυλλου του βιβλίου.



Σχήμα 4.17: Αναγνώριση αντικειμένου ανάμεσα σε άλλα (υπό τυχαίο προσανατολισμό) μέσω του αλγόριθμου SURF.

Στο σχήμα 4.17 φαίνονται οι αντιστοιχήσεις που έγιναν για το εξώφυλλο ενός βιβλίο όταν στην εικόνα αναζήτησης είναι υπό άλλον προσανατολισμό και μάλιστα καλύπτεται ένα μέρος του από άλλο βιβλίο. Από τα 100 σημεία της εικόνας (για ανάλυση 240x320 pixel) του αντικειμένων και 117 της εικόνας αναζήτησης αντιστοιχήθηκαν μόνο τα 7. Υπήρχε δηλαδή κι εδώ ραγδαία μείωση των αντιστοιχισμένων σημείων αλλά ο αριθμός είναι ικανοποιητικός για να αναγνωριστεί το αντικείμενο.

Συνεπώς οι μέθοδοι αυτοί λειτουργούν αρκετά καλά, αλλά όχι στην περίπτωση των αντικειμένων που εξετάζονται. Οι ελάχιστες λεπτομέρειες στην επιφάνεια των κομματιών οδηγεί στη μη εύρεση ικανού αριθμού χαρακτηριστικών σημείων, ώστε να γίνει η αντιστοίχιση υπό τυχαίο προσανατολισμό και θέση των αντικειμένων. Η αύξηση της ανάλυσης της εικόνας αυξάνει τον αριθμό των σημείων αλλά όχι σε ικανοποιητικό βαθμό. Είδαμε, πως μια εικόνα ενός τυπικού εξωφύλλου αυξάνοντας την ανάλυση 10 φορές, εμφανίζει 10 φορές περισσότερα σημεία, ενώ για τα αντικείμενα, που εξετάζονται, μια τέτοια αύξηση στην ανάλυση, αποφέρει μόνο διπλασιασμό των χαρακτηριστικών σημείων. Τέλος, παρόμοια ήταν τα αποτελέσματα, τα οποία προήλθαν από χρησιμοποίηση του αλγορίθμου SIFT και δεν κρίνεται σκόπιμο να παρουσιαστούν.

4.4 Διάταξη λήψης εικόνων μέσω scanner

Στην προηγούμενη ενότητα έγιναν εμφανή τα προβλήματα που προκύπτουν εξαιτίας της προοπτικής προβολής, όταν τα αντικείμενα που εξετάζονται δεν είναι «επίπεδα». Ένας τρόπος να αποφευχθεί αυτό είναι η εικόνες που λαμβάνουμε να είναι μια ορθογραφική προβολή του τρισδιάστατου κόσμου. Ο πιο απλός τρόπος να γίνει αυτό είναι να λαμβάνονται οι εικόνες μέσω ενός scanner. Η επιφάνεια τοποθέτησης πλέον είναι η επιφάνεια σάρωσης του μηχανήματος. Η διαδικασία για την κάθε μέθοδο ακολουθεί τα ίδια βήματα με προηγουμένως.

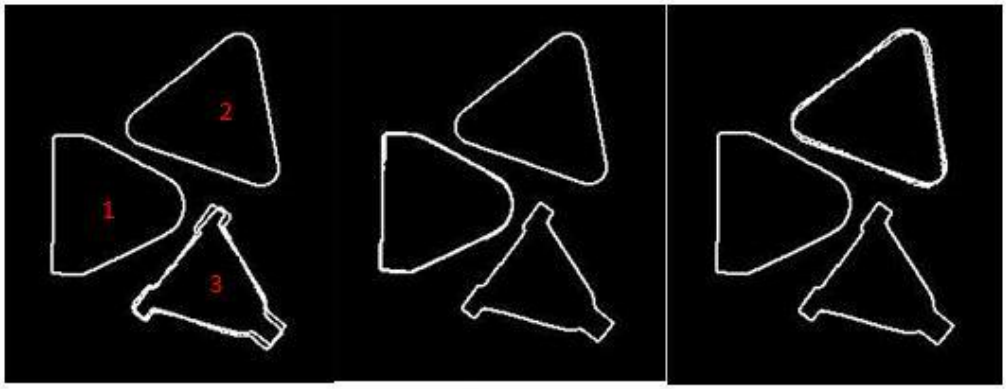
Παρακάτω θα παρουσιάσουμε αποτελέσματα που αφορούν τις μεθόδους αναγνώρισης μοτίβου καθώς και τις μεθόδους που χρησιμοποιούν ανιχνευτές χαρακτηριστικών σημείων.

4.4.1 Αναγνώριση αντικειμένων μέσω μεθόδων αναγνώρισης μοτίβου

Στο πρόγραμμα, πλέον, εισάγονται εικόνες που έχουν ληφθεί από scanner. Αυτές είναι παρόμοιες με αυτές του σχήματος 4.18.

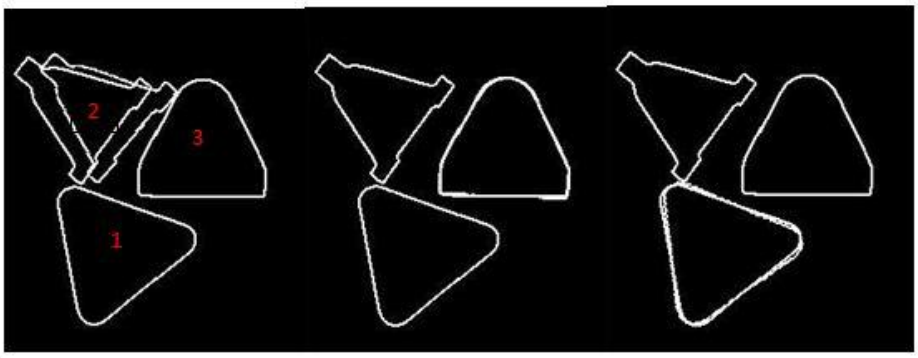
Σχήμα 4.18: Εικόνες που έχουν ληφθεί μέσω scanner και εισάγονται στο πρόγραμμα.

Εύκολα παρατηρούμε πως πλέον έχει χαθεί κάθε πληροφορία του ύψους των αντικειμένων, ενώ η εικόνα αποτελεί μια κάτοψη της συγκεκριμένης πλευράς τους. Πλέον κάθε αλλαγή στη θέση ή τη φορά τοποθέτησης δε θα μεταβάλλει το περίγραμμα του αντικειμένου. Στα σχήματα 4.19 -4.23 δίνονται αποτελέσματα για ορισμένες περιπτώσεις.



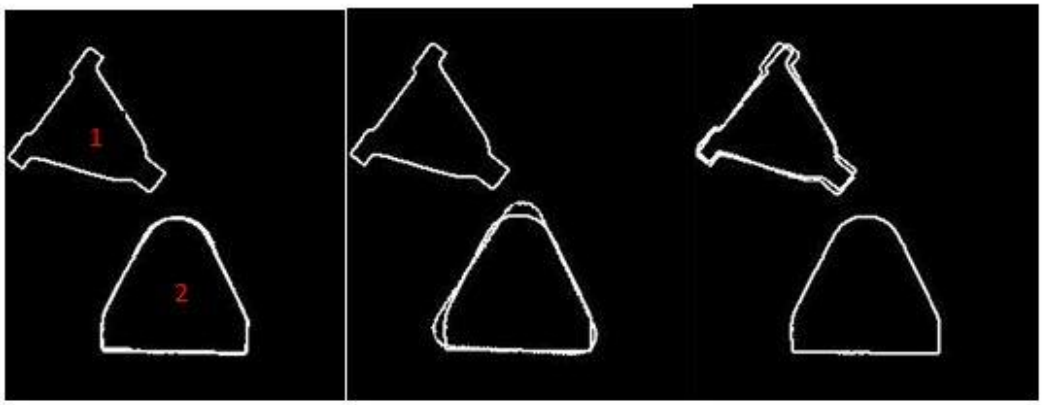
	Αντιστοιχεί με αντικείμενο εικόνας:		
Αντικείμενο #1 (Βάσης Δεδομένων)	3	0	0
Αντικείμενο #2 (Βάσης Δεδομένων)	0	1	0
Αντικείμενο #3 (Βάσης Δεδομένων)	0	0	2

Σχήμα 4.19: Πρώτη και τελική αντιστοίχιση χρησιμοποιώντας λήψη εικόνων μέσω scanner.



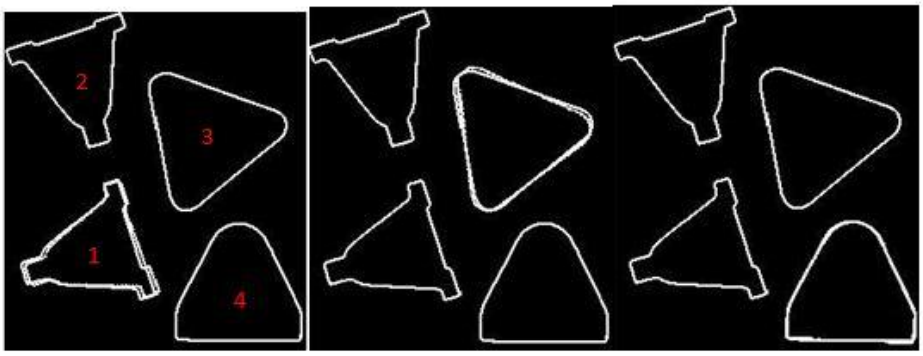
	Αντιστοιχεί με αντικείμενο εικόνας:		
Αντικείμενο #1 (Βάσης Δεδομένων)	2	0	0
Αντικείμενο #2 (Βάσης Δεδομένων)	0	3	0
Αντικείμενο #3 (Βάσης Δεδομένων)	0	0	1

Σχήμα 4.20: Πρώτη και τελική αντιστοίχιση χρησιμοποιώντας λήψη εικόνων μέσω scanner.



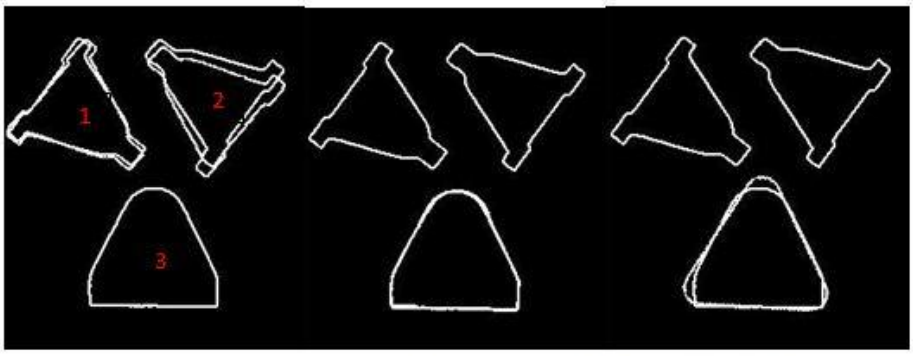
	Αντιστοιχεί με αντικείμενο εικόνας:	
Αντικείμενο #1 (Βάσης Δεδομένων)	1	0
Αντικείμενο #2 (Βάσης Δεδομένων)	0	2

Σχήμα 4.21: Πρώτη και τελική αντιστοίχιση χρησιμοποιώντας λήψη εικόνων μέσω scanner.



	Αντιστοιχεί με αντικείμενο εικόνας:			
Αντικείμενο #1 (Βάσης Δεδομένων)	1	2	0	0
Αντικείμενο #2 (Βάσης Δεδομένων)	0	0	0	4
Αντικείμενο #3 (Βάσης Δεδομένων)	0	0	3	0

Σχήμα 4.22: Πρώτη και τελική αντιστοίχιση χρησιμοποιώντας λήψη εικόνων μέσω scanner.



	Αντιστοιχεί με αντικείμενο εικόνας:		
Αντικείμενο #1 (Βάσης Δεδομένων)	1	2	0
Αντικείμενο #2 (Βάσης Δεδομένων)	0	3	0
Αντικείμενο #3 (Βάσης Δεδομένων)	0	0	0

Σχήμα 4.23: Πρώτη και τελική αντιστοίχιση χρησιμοποιώντας λήψη εικόνων μέσω scanner.

Από τα παραπάνω σχήματα συμπεραίνουμε, πως το πρόγραμμα αντιστοιχίζει σωστά τα αντικείμενα σε οποιοδήποτε συνδυασμό, αριθμού, θέσης και προσανατολισμού τους στην εικόνα. Ο χρόνος εκτέλεσης, εντωμεταξύ, παρέμεινε στα ίδια επίπεδα, δηλαδή 20-25 sec.

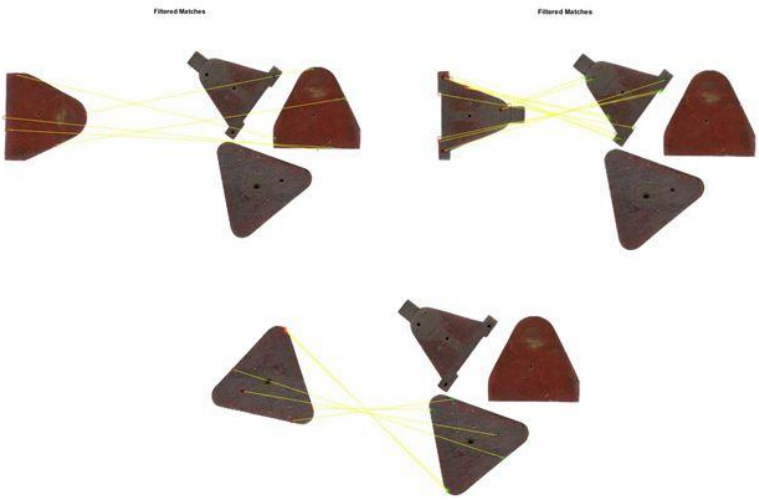
Η παράλληλη προβολή των αντικειμένων λειτούργησε για την περίπτωσή μας, αλλά δεν πρέπει να λησμονηθούν οι περιορισμοί που επιβάλλει. Η κάτοψη για παράδειγμα ενός αντικειμένου μπορεί να είναι η ίδια με ένα άλλο, όμως το ύψος του τελείως διαφορετικό. Στο πρόβλημα που εξετάζεται, τα κομμάτια θεωρούμε ότι μπορούν να προσδιοριστούν πλήρως από την κάτοψη τους.

Τέλος, η χρήση Scanner, δημιουργεί ένα κλειστό περιβάλλον, όσον αφορά τις συνθήκες φωτισμού των αντικειμένων κατά τη λήψη των εικόνων, αποτρέποντας τις αρνητικές συνέπειες της σκίασης. Κατά συνέπεια, η προεπεξεργασία αυτών των εικόνων είναι αρκετά πιο εύκολη, ενώ παράλληλα υπάρχει μεγάλη επαναληψιμότητα στις εικόνες που προκύπτουν, από το στάδιο αυτό.

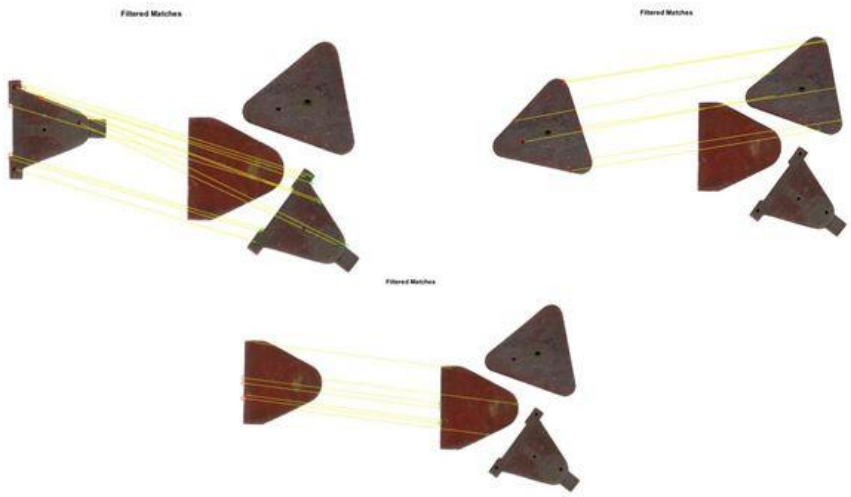
4.4.2 Αναγνώριση αντικειμένων μέσω ανιχνευτών χαρακτηριστικών σημείων

Εφόσον δεν υπάρχει μεταβολή, πλέον, στο φωτισμό των αντικειμένων, ή στην οπτική γωνία λήψης της εικόνας, οι εν λόγω μέθοδοι θα παρουσιάζουν με τη σειρά τους και αυτές ορθότερα αποτελέσματα. Στη συνέχεια παρατίθενται ορισμένα από αυτά ενδεικτικά.

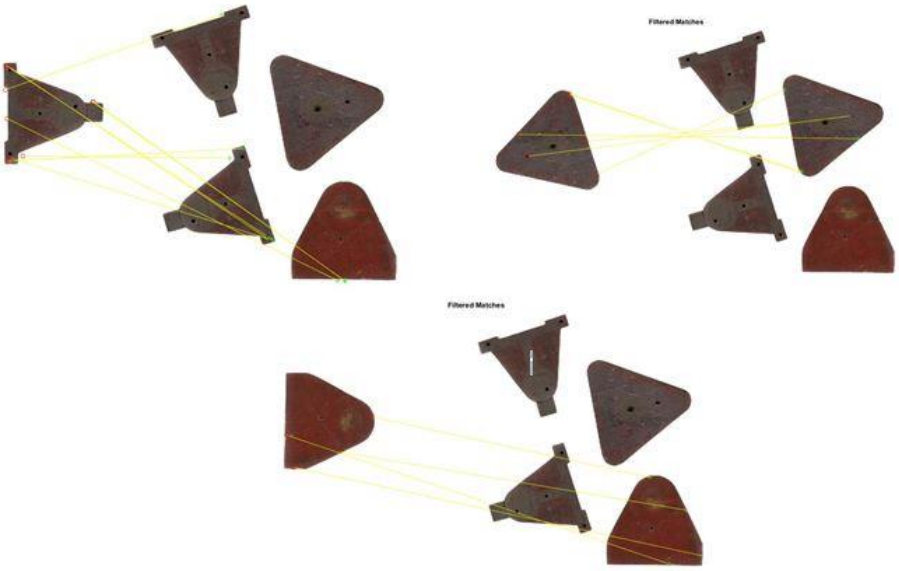
4.4.2.1 Χρήση αλγόριθμου SURF



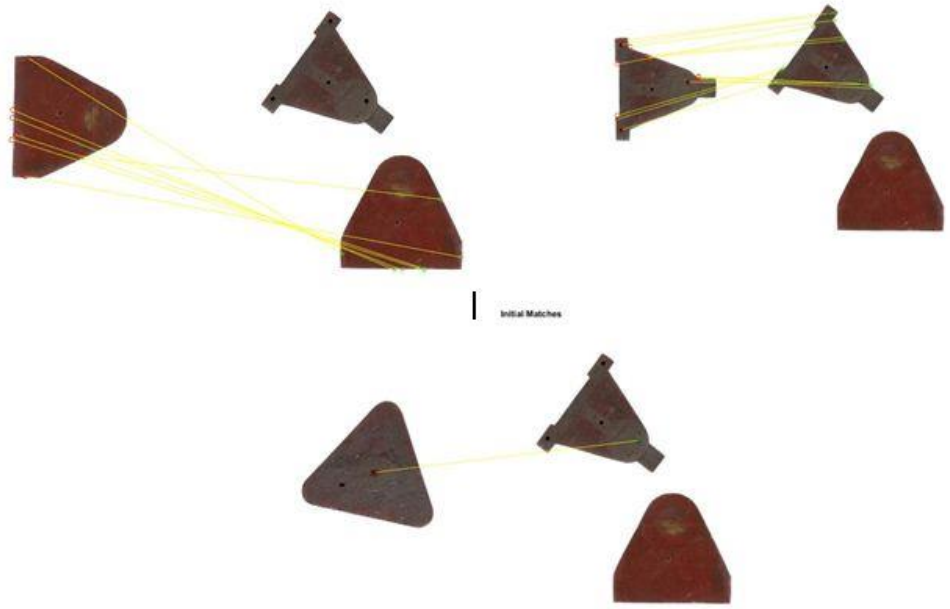
Σχήμα 4.24: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SURF.



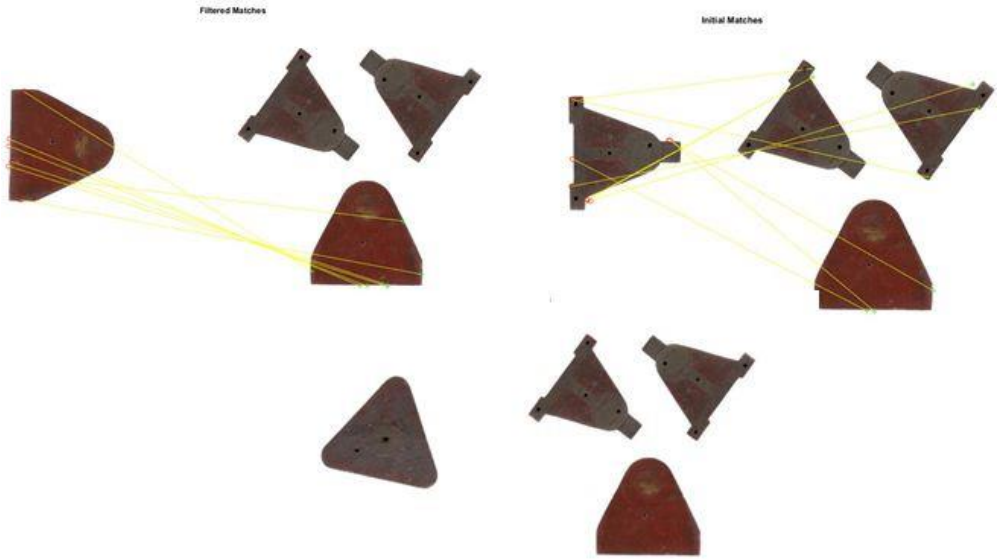
Σχήμα 4.25: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SURF.



Σχήμα 4.26: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SURF.



Σχήμα 4.27: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SURF.

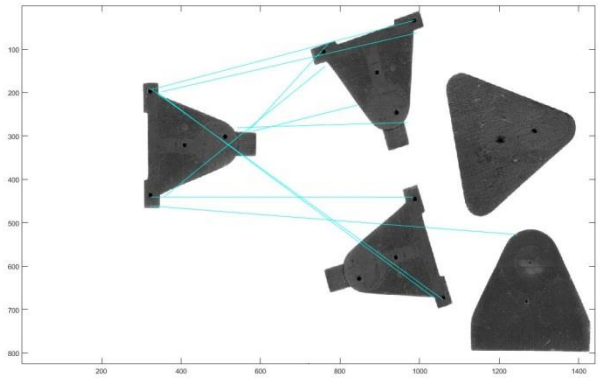


Σχήμα 4.28: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SURF.

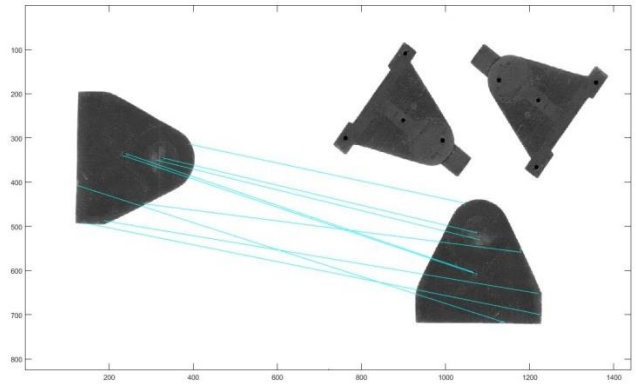
Η αναγνώριση της θέσης και του προσανατολισμού των αντικειμένων στην πλειονότητα των περιπτώσεων γίνεται σωστά. Ο αλγόριθμος παρουσιάζει λανθασμένες εκτιμήσεις, όταν το αντικείμενο εμφανίζεται παραπάνω από μια φορές στην εικόνα αναζήτησης. Συνεπώς, με περαιτέρω επεξεργασία του αλγορίθμου είναι δυνατή η σωστή αναγνώριση των αντικειμένων.

4.4.2.2 Χρήση αλγόριθμου SIFT

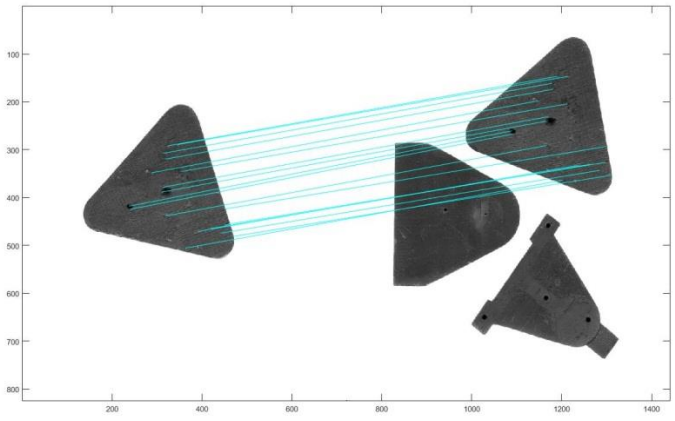
Παρόμοια αποτελέσματα παρουσιάζονται χρησιμοποιώντας τον αλγόριθμο SIFT αντί του SURF. Για λόγους πληρότητας παρουσιάζονται στα επόμενα σχήματα ορισμένα αποτελέσματα.



Σχήμα 4.29: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SIFT.



Σχήμα 4.30: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SIFT.



Σχήμα 4.31: Αναγνώριση αντικειμένου ανάμεσα σε άλλα μέσω του αλγόριθμου SIFT.

4.5 Αναγνώριση αντικειμένων υπό τυχαία πλευρά τοποθέτησης στην εικόνα αναζήτησης

Παραπάνω αναφερθήκαμε στους τρόπους αντιστοίχισης των αντικειμένων θεωρώντας τον τρόπο τοποθέτησής τους δεδομένο. Όλα τα αντικείμενα τοποθετούνταν με την επίπεδη πλευρά τους. Σε ένα πραγματικό σύστημα όμως αυτό είναι αρκετά δύσκολο να επιτευχθεί, όταν τα αντικείμενα μπορούν να στηριχθούν εξίσου σε όλες τις πλευρές τους. Συνεπώς, κρίθηκε σκόπιμο να μετατραπεί ο αλγόριθμος, ώστε να λαμβάνει υπόψη και αυτή την παράμετρο.

4.5.1 Δημιουργία της βάσης δεδομένων

Πλέον κάθε γνωστό αντικείμενο στο χρήστη διαθέτει ξεχωριστό φάκελο. Εκεί καταχωρείται ο ελάχιστος αριθμός εικόνων, που μπορούν να περιγράψουν το αντικείμενο τοποθετημένο υπό κάθε πλευρά του. Είναι σημαντικό να αναφερθεί πως στη διάταξη λήψης εικόνων με κάμερα, ένα αντικείμενο μπορεί να απαιτείται να διαθέτει παραπάνω από μία εικόνες όντας τοποθετημένο σε κάποια πλευρά του, μετατοπισμένο όμως ως προς την κάμερα, εξαιτίας της προοπτικής προβολής.

Είναι κατανοητό ότι, λόγω του μεγαλύτερου αριθμού εικόνων που πρέπει πλέον να διαχειριστεί ο αλγόριθμος, οι χρόνοι εκτέλεσης αυξάνονται αισθητά. Για να αποφευχθούν οι μεγάλοι χρόνοι εκτέλεσης, ο αλγόριθμος δίνει τη δυνατότητα στο χρήστη να επιλέξει ποιο αντικείμενο της βάσης δεδομένων αναζητά στην εικόνα.

4.5.2 Διάταξη λήψης εικόνων μέσω κάμερας

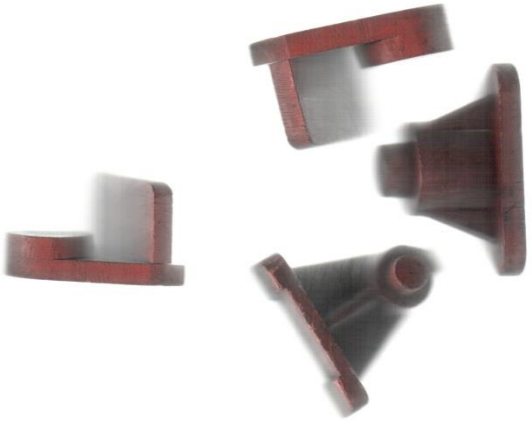
Σε αυτή την περίπτωση τα αποτελέσματα δεν ήταν αξιόλογα. Η βάση δεδομένων έπρεπε να είναι πάρα πολύ μεγάλη. Κάθε πλευρά του αντικειμένου πρέπει να φωτογραφηθεί πολλές φορές σε πολλά σημεία της τράπεζας και σε διάφορες γωνίες ως προς την κάμερα, λόγω της προοπτικής προβολής. Ακόμη όμως κι αν συμβεί αυτό τα αποτελέσματα αντιστοίχισης δε θα είναι τα βέλτιστα καθώς και οι χρόνοι εκτέλεσης θα είναι πάρα πολύ μεγάλοι (εξαιτίας των πολλών εικόνων στη βάση δεδομένων). Χαρακτηριστικό παράδειγμα των παραπάνω αποτελούν οι δύο επόμενες εικόνες. Πρόκειται για την πλάγια όψη του ίδιου αντικειμένου σε δυο διαφορετικές θέσεις στο επίπεδο εργασίας.

Σχήμα 4.32: Η πλάγια όψη του αντικειμένου αναπαρίσταται στην εικόνα πολύ διαφορετικά λόγω της προοπτικής προβολής.

Όπως είναι κατανοητό, το πρόγραμμα δεν πρόκειται να αντιστοιχίσει σωστά το αντικείμενο σε αυτή την περίπτωση. Συνεπώς, απαιτείται ένα πλήθος εικόνων για κάθε πλευρά ενός αντικειμένου.

4.5.3 Αποτελέσματα χρησιμοποιώντας τη διάταξη λήψης εικόνων μέσω scanner

Στη διάταξη αύτη τα πράγματα είναι απλούστερα, καθώς η προβολή της πλευράς του αντικειμένου θα είναι η ίδια σε κάθε θέση στο επίπεδο σάρωσης όπως επίσης και για κάθε προσανατολισμό του αντικειμένου. Χαρακτηριστικό παράδειγμα αποτελούν οι δύο επόμενες εικόνες, όπου το αντικείμενο αν και βρίσκεται σε διαφορετική θέση και προσανατολισμό προβάλλεται με τον ίδιο τρόπο στο επίπεδο σάρωσης.



Σχήμα 4.33: Η προβολή της πλάγιας όψης του αντικειμένου δεν επηρεάζεται από τη θέση του στο επίπεδο σάρωσης.

Το γεγονός αυτό διευκολύνει τη δημιουργία της βάσης δεδομένων, αφού αύτη θα περιλαμβάνει τον ελάχιστο αριθμό εικόνων. Στη γενική περίπτωση απαιτείται μια εικόνα για κάθε πλευρά του αντικειμένου.

Με την καταχώρηση, όμως, πολλαπλών εικόνων για κάθε αντικείμενο στη βάση δεδομένων δημιουργείται το πρόβλημα του χρόνου εκτέλεσης. Συγκεκριμένα, ενώ

προηγουμένως το πρόγραμμα αναζητούσε στην εικόνα τρία αντικείμενα, πλέον αναζητεί την κάθε πλευρά των τριών αντικειμένων στην εικόνα (στην περίπτωσή μας 12 πλευρές συνολικά). Ο απαιτούμενος χρόνος εκτέλεσης μιας καθολικής αναζήτησης, δηλαδή, είναι τουλάχιστον τετραπλάσιος. Στην πράξη όμως είναι αρκετά μεγαλύτερος. Με τις ρυθμίσεις των προηγούμενων προσομοιώσεων από 30 sec η εκτέλεση έφτασε στα επίπεδα των 500 sec. Για να επανέλθει ο χρόνος εκτέλεσης σε λογικά επίπεδα έπρεπε να τροποποιηθούν ανάλογα οι ακόλουθες παράμετροι:

- Μέγεθος εικόνας (από 240x275 μειώθηκε στο 175x126 περίπου στο μισό)

Το ιδανικό είναι η επεξεργασία να γίνεται με όσο το δυνατόν μικρότερες εικόνες. Δεν μπορεί να μειωθεί όμως πάρα πολύ καθώς υπάρχει ο κίνδυνος αλλοίωσης των χαρακτηριστικών των αντικειμένων (π.χ ενοποίηση δυο σχημάτων αν το μεταξύ τους κενό δεν είναι αρκετά μεγάλο)

- Δειγματοληψία σημείων περιγράμματος

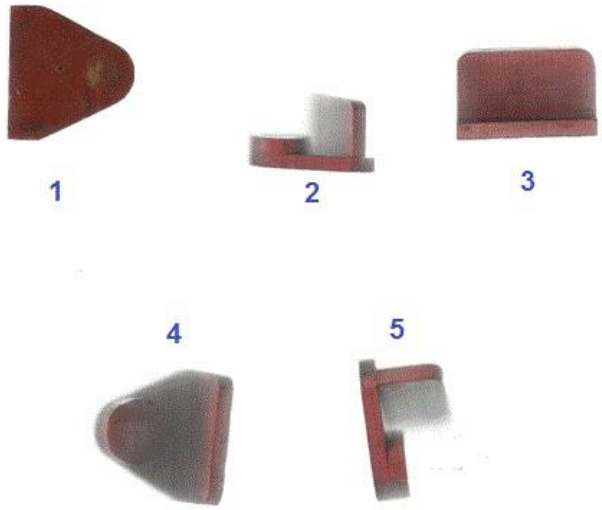
Μετά την προεπεξεργασία των εικόνων προκύπτει το περίγραμμά τους. Για την τελική απόφαση αντιστοίχισης γίνεται η σύγκριση σημείο προς σημείο των δύο σχημάτων (της εικόνας αναζήτησης και της εικόνας από τη βάση δεδομένων) Επειδή τα σημεία του περιγράμματος είναι πάρα πολλά (~1000) γίνεται μια δειγματοληψία για την ταχύτερη εκτέλεση του προγράμματος.

Από την άλλη πλευρά, όσο περισσότερα σημεία τόσο ασφαλέστερα αποτελέσματα. Μετά από δοκιμές τα δειγματοληπτούμενα σημεία από 100 μειώθηκαν σε 60, καθώς από τη μία μειώνουν το χρόνο εκτέλεσης, από την άλλη προσφέρουν επαναληψιμότητα στα τελικά αποτελέσματα.

- Γωνία περιστροφής

Μία άλλη παράμετρος που μπορούσε να τροποποιηθεί είναι η γωνία με την οποία περιστρέφεται κάθε φορά η εικόνα του αντικειμένου της βάσης δεδομένων, πριν πραγματοποιηθεί η συνέλιξή της με την εικόνα αναζήτησης. Μεγαλύτερη γωνία περιστροφής σημαίνει μικρότερος αριθμός συνολικών επαναλήψεων. Αυτή η παράμετρος όμως αποφασίστηκε να μείνει ως έχει (10 μοίρες) καθώς συμβάλλει λίγο στον τελικό χρόνο προσομοίωσης.

Στη συνέχεια παρουσιάζεται η βάση δεδομένων που δημιουργήθηκε για τα εξεταζόμενα αντικείμενα κατά τη λήψη εικόνων μέσω scanner.



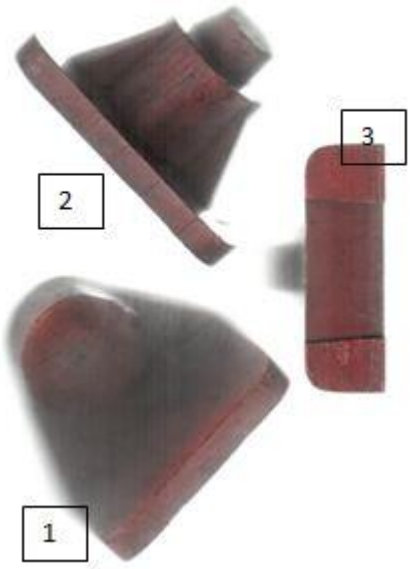
Σχήμα 4.34: Σάρωση των πλευρών του πρώτου αντικειμένου και σειρά καταχώρησής τους στη βάση δεδομένων.

Σχήμα 4.35: Σάρωση των πλευρών του δεύτερου αντικειμένου και σειρά καταχώρησής τους στη βάση δεδομένων.

Σχήμα 4.36: Σάρωση των πλευρών του τρίτου αντικειμένου και σειρά καταχώρησής τους στη βάση δεδομένων.

Όπως παρατηρούμε, ο αριθμός των εικόνων σε κάθε περίπτωση δεν είναι ο ίδιος, καθώς εξαρτάται από τη μορφή κάθε αντικειμένου. Ειδικότερα, για το πρώτο αντικείμενο απαιτείται η σάρωση και των δύο πλάγιων όψεων καθώς η μία είναι ο καθρέπτης της άλλης.

Παρακάτω παρουσιάζονται τα αποτελέσματα του αλγορίθμου για ορισμένες περιπτώσεις.



Σχήμα 4.37: Σάρωση τριών αντικειμένων υπό τυχαία πλευρά τοποθέτησης – Οι αριθμοί δείχνουν τη σειρά προσπέλασής τους από το πρόγραμμα.

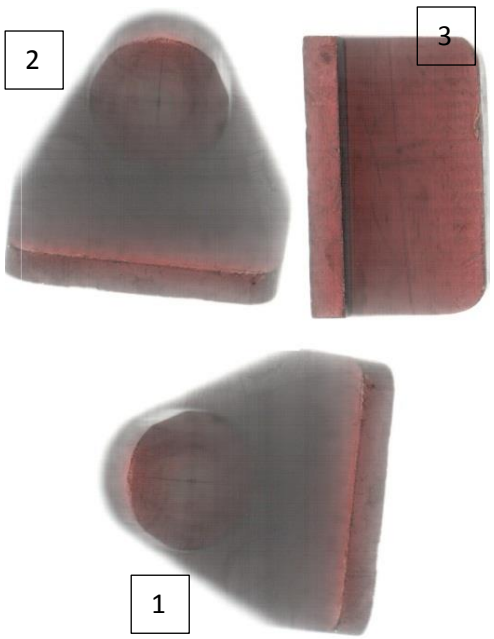
Στην παραπάνω εικόνα αναζήτησης είναι τοποθετημένα και τα τρία αντικείμενα που εξετάζονται, κάθε ένα σε τυχαία πλευρά και προσανατολισμό. Τα αποτελέσματα συγκεντρώνονται στον παρακάτω πίνακα:

	1	2	3
1	4	0	0
2	0	0	2
3	0	3	0

Σχήμα 4.38: Πίνακας αντιστοίχησης αντικειμένων.

Οι γραμμές είναι ο αριθμός του φακέλου στη βάση δεδομένων, οι στήλες είναι το αντικείμενο στην εικόνα αναζήτησης. Τα νούμερα στα κελιά είναι ο αριθμός της εικόνας στο φάκελο της βάσης δεδομένων. Έτσι, η 4^{η} εικόνα του αντικειμένου 1 της βάσης δεδομένων αντιστοιχίζεται με το αντικείμενο 1 της εικόνας αναζήτησης.

Επόμενο βήμα είναι η εξέταση των αποτελεσμάτων όταν στην εικόνα σάρωσης παρουσιάζεται το ίδιο αντικείμενο παραπάνω από μια φορές. Στο επόμενο σχήμα είναι τοποθετημένο το ίδιο αντικείμενο δύο φορές τοποθετημένο με την ίδια πλευρά και μια ακόμα φορά τοποθετημένο σε άλλη πλευρά.



Σχήμα 4.39: Το ίδιο αντικείμενο εμφανίζεται στην εικόνα δύο φορές υπό διαφορετικό προσανατολισμό και μια τοποθετημένο υπό άλλη βάση στήριξης.

Τα αποτελέσματα παρουσιάζονται συνοπτικά στον πίνακα του σχήματος 4.38.

	1	2	3
1	4	4	3
2	0	0	0
3	0	0	0

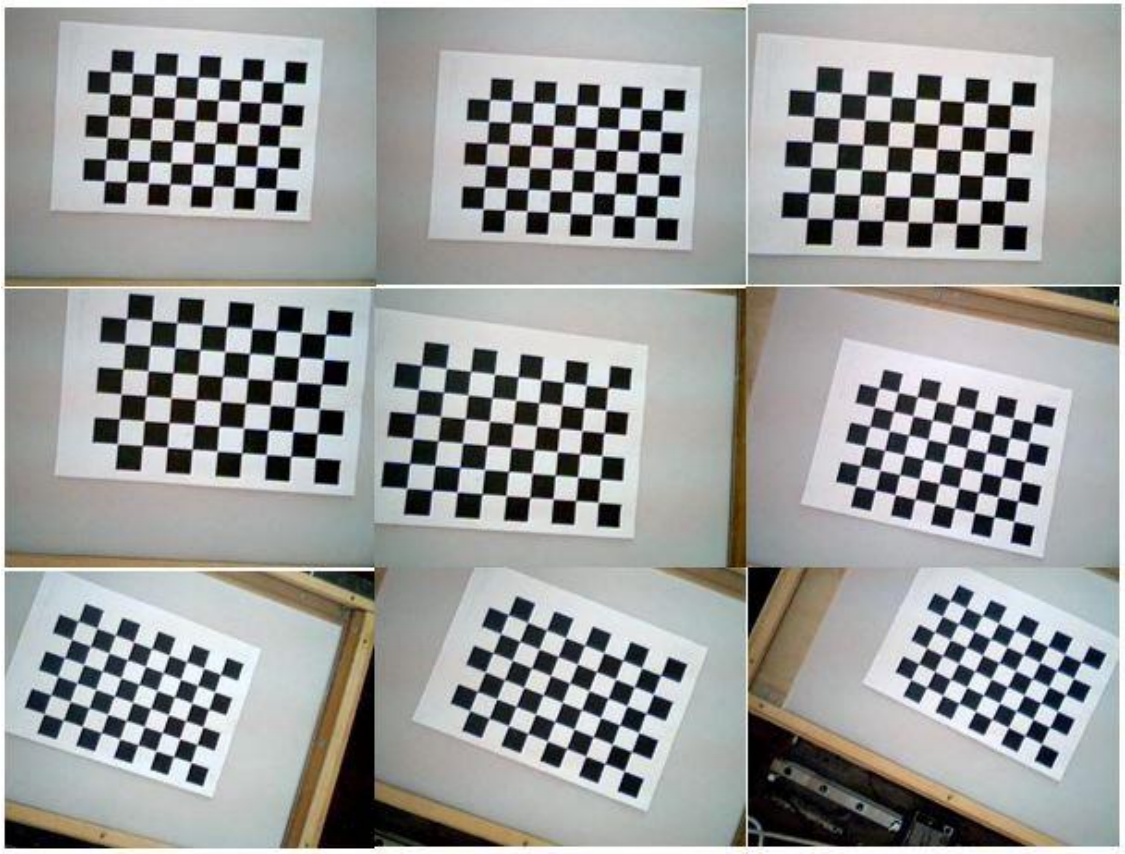
Σχήμα 4.40: Πίνακας αντιστοίχησης αντικειμένων.

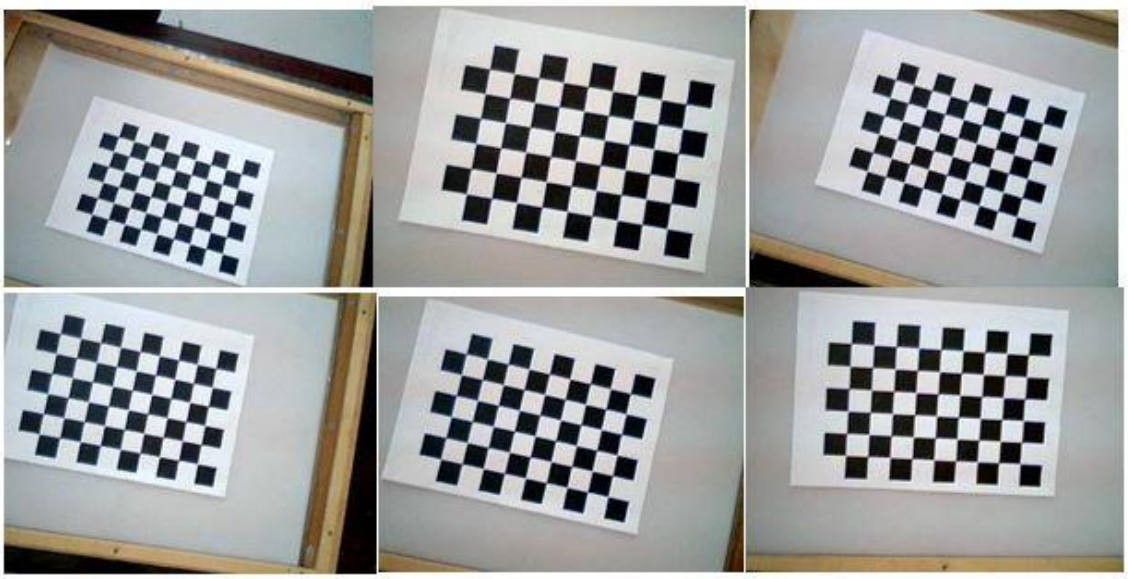
4.6 Αντιστοίχιση συντεταγμένων εικόνας με πραγματικές συντεταγμένες – Εξαγωγή αποτελεσμάτων

Η αντιστοίχιση των συντεταγμένων εικόνας με τις πραγματικές συντεταγμένες γίνεται με διαφορετικό τρόπο ανάλογα τη διάταξη που χρησιμοποιείται. Στην πρώτη περίπτωση, που οι εικόνες λαμβάνονται μέσω κάμερας, η αντιστοίχιση θα γίνει με τη μέθοδο του καλιμπραρίσματος της κάμερας. Στη δεύτερη διάταξη η αντιστοίχιση των συντεταγμένων θα γίνει μετρώντας την επιφάνεια σάρωσης.

4.6.1 Διάταξη κάμερας

Η διαδικασία του καλιμπραρίσματος απαιτεί την λήψη αρκετών εικόνων ενός μοτίβου καλιμπραρίσματος από την κάμερα της διάταξης. Το μοτίβο φωτογραφίζεται κοντά στην περιοχή όπου προορίζεται η τοποθέτηση των αντικειμένων, υπό διαφορετικούς προσανατολισμούς και οπτικές γωνίες. Το μοτίβο που επιλέγεται συνήθως είναι αυτό της ασύμμετρης σκακιέρας. Συνήθως, απαιτούνται 10 με 20 εικόνες του μοτίβου για το καλιμπράρισμα. Οι εικόνες που χρησιμοποιήθηκαν για το καλιμπράρισμα στην περίπτωσή μας παρουσιάζονται στα επόμενα σχήματα.



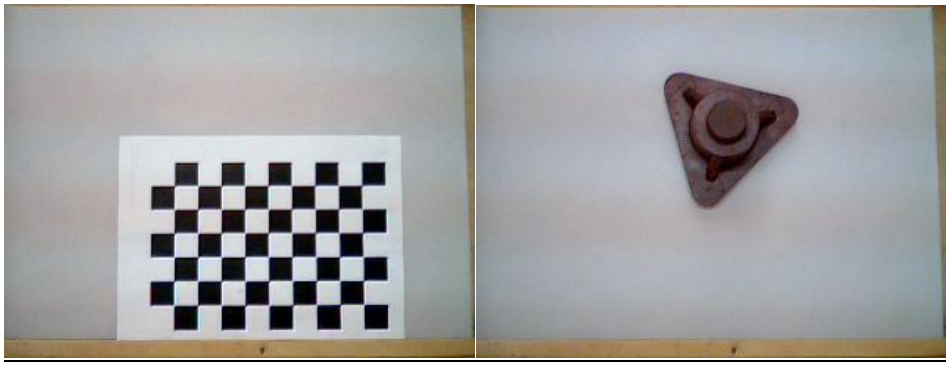


Σχήμα 4.41: Εικόνες μοτίβου καλιμπραρίσματος.

Αφού εισαχθούν οι εικόνες του μοτίβου, για την ολοκλήρωση της διαδικασίας του καλιμπραρίσματος, απαιτείται η εισαγωγή στο πρόγραμμα μιας τελικής φωτογραφίας η οποία να περιέχει το μοτίβο και η σχετική θέση της κάμερας με το επίπεδο θα είναι αυτή με την οποία θα ληφθούν οι εικόνες των αντικειμένων στην πορεία.

4.6.1.1 Αντιστοίχιση συντεταγμένων εικόνας με πραγματικές συντεταγμένες

Όπως αναλύθηκε στο προηγούμενο κεφάλαιο, η διαδικασία του καλιμπραρίσματος εξάγει έναν πίνακα με τις ενδογενείς και εξωγενείς παραμέτρους της κάμερας, τον οποίο μπορούμε να χρησιμοποιούμε για να αντιστοιχίσουμε κάθε σημείο της εικόνας με ένα σημείο στο χώρο.



Σχήμα 4.42: Εικόνα καλιμπραρίσματος και εικόνα αναζήτησης (η λήψη γίνεται με την κάμερα τοποθετημένη στο ίδιο σημείο).

Πριν ληφθεί η εικόνα του αντικειμένου πρέπει να ληφθεί η εικόνα της ασύμμετρης σκακιέρας με την κάμερα τοποθετημένη στο ίδιο σημείο. Το μέγεθος των

τετραγώνων στην περίπτωσή μας είναι 25mm. Ό κώδικας που χρησιμοποιήσαμε στο λογισμικό Matlab, είναι ο εξής:

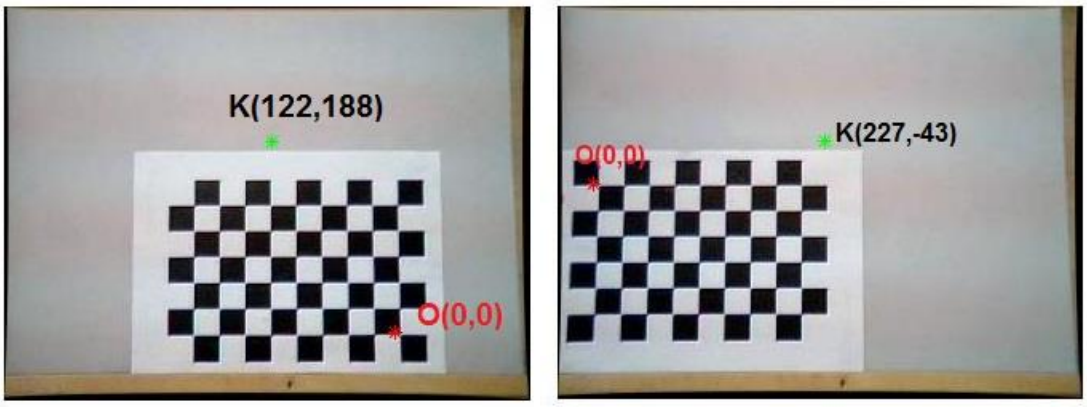
```
clear
close all
clc
% Read Images of the pattern
numImages = 16;
files = cell(1, numImages);
for i = 1:numImages
    files{i} = fullfile('C:\','Users','This','Desktop','review' ...
        ,'clbr','calibration', sprintf('image0%d.jpg', i));
end

% Detect the checkerboard corners in the images.
[imagePoints, boardSize] = detectCheckerboardPoints(files);

% Generate the world coordinates of the checkerboard corners in the
% pattern-centric coordinate system, with the upper-left corner at (0,0).
squareSize = 25; % in millimeters
worldPoints = generateCheckerboardPoints(boardSize, squareSize);

% Calibrate the camera.
cameraParams = estimateCameraParameters(imagePoints, worldPoints);

% Evaluate calibration accuracy.
figure; showReprojectionErrors(cameraParams);
title('Reprojection Errors');


im=imread('demo.jpg');
[im, newOrigin] = undistortImage(im, cameraParams, 'OutputView', 'full');

% Detect the checkerboard.
[imagePoints, boardSize] = detectCheckerboardPoints(im);

% Compute rotation and translation of the camera.
[R, t] = extrinsics(imagePoints, worldPoints, cameraParams);

```

Σχήμα 4.43: Κώδικας για την υλοποίηση της διαδικασίας του καλιμπραρίσματος της κάμερας

Στο τέλος έχει υπολογιστεί ο πίνακας R, μέσω του οποίου θα μεταφραστεί οποιοδήποτε σημείο της εικόνας σε σημείο του πραγματικού χώρου.

Σχήμα 4.44: Εύρεση του κέντρου του αντικειμένου στην εικόνα αναζήτησης.

Το κέντρο του αντικειμένου βρίσκεται σε συντεταγμένες εικόνας στο σημείο (167.7, 84.9). Αν το μεταφράσουμε σε πραγματικές συντεταγμένες θα είναι το σημείο (122, 188). Οι τιμές δηλώνουν πλέον οριζόντια και κάθετη απόσταση (σε mm) από το σημείο αναφοράς (με πραγματικές συντεταγμένες (0,0). Αν το μοτίβο φωτογραφιζόταν σε διαφορετική θέση το σημείο αναφοράς θα ήταν διαφορετικό και συνεπώς το κέντρο του παραπάνω αντικειμένου θα είχε διαφορετικές πραγματικές συντεταγμένες. Αυτά αποτυπώνονται σαφέστερα στα ακόλουθα σχήματα.



Σχήμα 4.45: Η διαφορετική θέση του μοτίβου έχει ως αποτέλεσμα την αλλαγή του συστήματος αναφοράς.

Στα δυο σχήματα παρατηρούμε πως, δεν έχει αλλάξει μόνο το σημείο αναφοράς, αλλά και η θετική κατεύθυνση των αξόνων x και y. Παρατηρούμε επίσης πως το σημείο αναφοράς δεν είναι ένα τυχαίο σημείο του μοτίβου αλλά είναι το ίδιο, ασχέτως αν έχει αλλάξει η θέση η ο προσανατολισμός του. Κατά συνέπεια, γνωρίζοντάς το εκ των προτέρων, μπορούμε να ορίσουμε ποιο θα είναι το σημείο αναφοράς στην εικόνα αλλά και τη θετική κατεύθυνση στους οριζόντιους και κάθετους άξονες. Η μεγάλη πλευρά του μοτίβου είναι πάντα ο άξονας x ενώ η μικρή πλευρά ο άξονας y.

4.6.1.2 Εξαγωγή αποτελεσμάτων

Αφού ολοκληρωθεί η διαδικασία της αντιστοίχισης των αντικειμένων στην εικόνα με αυτά της βάσης δεδομένων, τα τελικά αποτελέσματα εξάγονται σε έναν πίνακα. Το σύνολο των γραμμών του πίνακα αντιπροσωπεύουν τον αριθμό των τελικών αντιστοιχίσεων. Υπάρχουν πέντε στήλες με αποτελέσματα. Στην πρώτη αποθηκεύεται ο αριθμός του φακέλου, της βάσης δεδομένων του αντικειμένου που βρέθηκε. Στη δεύτερη αποθηκεύεται ποια πλευρά του αντικειμένου φαίνεται στην εικόνα (με ποια εικόνα του εκάστοτε φακέλου έχει αντιστοιχηθεί το αντικείμενο). Στηω τρίτη και

τέταρτη στήλη αποθηκεύονται οι πραγματικές συντεταγμένες (x,y) αντίστοιχα (ως προς το σημείο αναφοράς), ενώ στην τελευταία αποθηκεύεται ο προσανατολισμός σε μοίρες. Παρακάτω παρουσιάζεται ο πίνακας αποτελεσμάτων που εξήχθη στο Excel, έχοντας ως εικόνα αναζήτησης το σχήμα (4.46) και ως εικόνα καλιμπραρίσματος αυτή που παρουσιάστηκε στο σχήμα(4.42).

Σχήμα 4.46: Εικόνα αναζήτησης.

Folder_of_Object	Matching_Side	Position_X_mm	Position_Y_mm	Orientation_degrees
1	1	295	114	110
2	1	105	228	10
3	1	39	60	290

Πίνακας 4.5: Πίνακας αποτελεσμάτων που εξάγεται από το πρόγραμμα.

4.6.2 Διάταξη scanner

Η εξαγωγή των πραγματικών συντεταγμένων σε εικόνες που έχουν ληφθεί υπό αυτή τη διάταξη γίνεται πιο απλά. Όπως αναφέρθηκε, τα αντικείμενα στην εικόνα προβάλλονται παράλληλα με το επίπεδο σάρωσης, συνεπώς για την εύρεση των συντεταγμένων απαιτείται μόνο ένας συντελεστής αναλογίας μεταξύ των πραγματικών συντεταγμένων και αυτών στην εικόνα. Μετρώντας, λοιπόν, με όσο το δυνατόν μεγαλύτερη ακρίβεια την επιφάνειας σάρωσης και έχοντας την ανάλυση της εικόνας σε pixel x pixel μπορούμε να βρούμε πόσα pixel αντιστοιχούν σε 1 mm τόσο στη x όσο και στην y κατεύθυνση.

4.6.2.1 Αντιστοίχιση συντεταγμένων εικόνας με πραγματικές συντεταγμένες

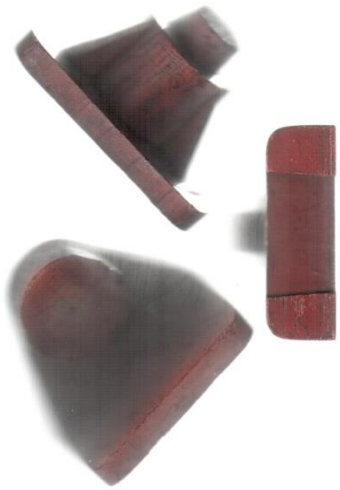
Επιλέχθηκε να γίνει σάρωση μιας επιφάνειας αντίστοιχου μεγέθους ενός φύλλου Α4 από το σαρωτή. Το μέγεθος της εικόνας είναι (1755x1276) ενώ αυτό του φύλλου Α4 είναι 297x210.

Σχήμα 4.47: Εύρεση του κέντρου του αντικειμένου στην εικόνα.

Το κέντρο του αντικειμένου σε συντεταγμένες εικόνας βρίσκεται στο σημείο (723.2, 1076.4). Σε πραγματικές συντεταγμένες το κέντρο του βρίσκεται στο σημείο (168, 128). Τα δύο συστήματα συντεταγμένων έχουν κοινό σημείο (0,0) – η επάνω και αριστερά γωνία της εικόνας, έτσι οι πραγματικές συντεταγμένες εκφράζουν την οριζόντια (μετατόπιση προς τα δεξιά) και κάθετη απόσταση (μετατόπιση προς τα κάτω) σε mm από το σημείο αυτό.

4.6.2.2 Εξαγωγή αποτελεσμάτων

Χρησιμοποιώντας τη βάση δεδομένων που αναπτύχθηκε στην προηγούμενη ενότητα και την εικόνα αναζήτησης του σχήματος (4.48), εξήχθη ο πίνακας αποτελεσμάτων.



Σχήμα 4.48: Εικόνα αναζήτησης.

Folder_of_Object	Matching_Side	Position_X_mm	Position_Y_mm	Orientation_degrees
1	4	82	203	40
3	3	96	65	50
2	2	181	139	190

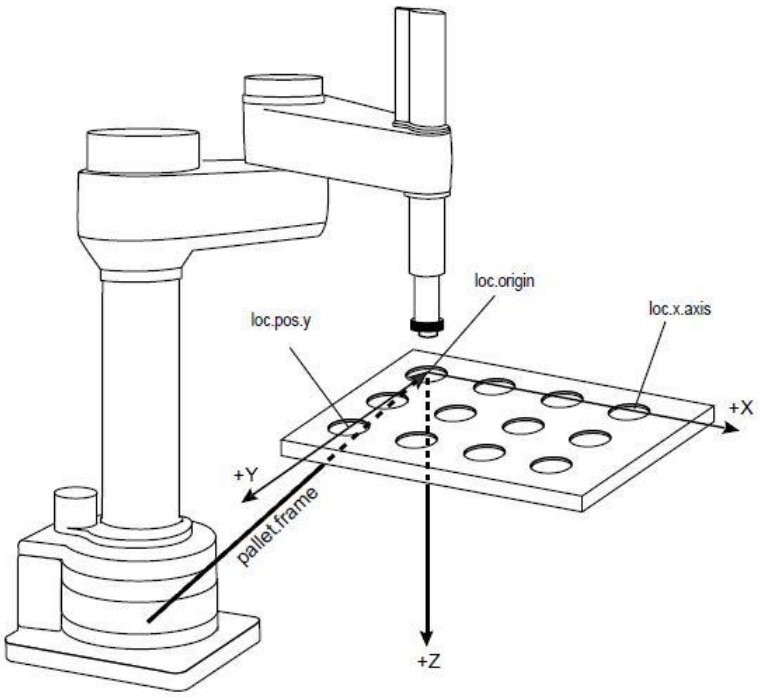
Πίνακας 4.6: Πίνακας αποτελεσμάτων που εξάγεται από το πρόγραμμα.

Τα αποτελέσματα του πίνακα ερμηνεύονται ως εξής: «Το αντικείμενο με πραγματικές συντεταγμένες (82,203), αντιστοιχήθηκε με την $4^{η}$ πλευρά (σύμφωνα με τη σειρά καταχώρησης στο φάκελο) του πρώτου αντικειμένου της βάσης δεδομένων, υπό προσανατολισμό 40 μοιρών (με αναφορά την εικόνα της βάσης δεδομένων).»

Παρατηρούμε, πως ο τελικός πίνακας αποτελεσμάτων και στις δύο διατάξεις δεν παρέχει την Z κατεύθυνση. Η συντεταγμένες στη κατεύθυνση αυτή εξαρτώνται από το ύψος του αντικειμένου, συνεπώς η πληροφορία είναι άμεσα συνδεδεμένη με την πλευρά τοποθέτησης [39]. Για κάθε πλευρά του αντικειμένου, λοιπόν, που βρίσκεται στη βάση δεδομένων, με βάση τα γεωμετρικά χαρακτηριστικά του αντικειμένου, μπορεί να οριστεί μια κατάλληλη Z συνιστώσα.

4.7 Σύστημα συντεταγμένων ρομποτικού βραχίονα

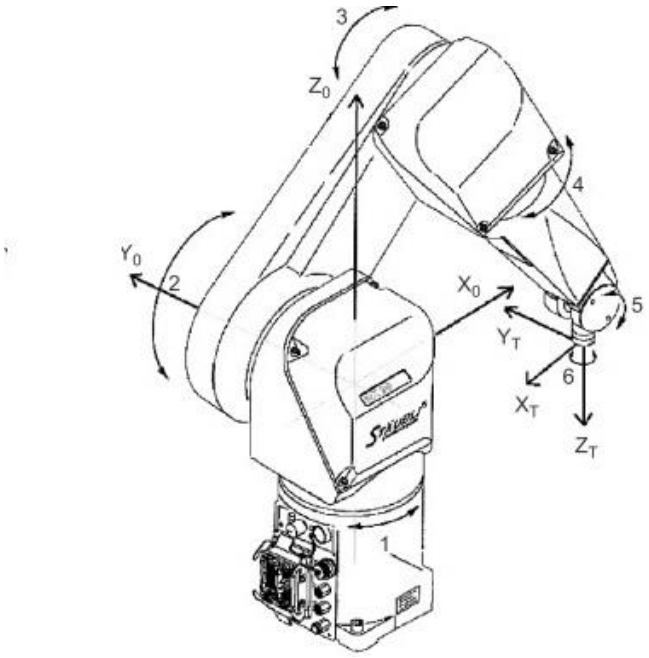
Σκοπός είναι η χρησιμοποίηση των πληροφοριών του εξαχθέντος πίνακα για την παραλαβή του αντικειμένου από έναν ρομποτικό βραχίονα. Αυτό απαιτεί τον προσδιορισμό ενός συστήματος συντεταγμένων ως αναφορά για τον ρομποτικό βραχίονα. Ο προσδιορισμός θα επιτευχθεί με μια τεχνική εκμάθησης. Συγκεκριμένα, ο ρομποτικός βραχίονας θα τοποθετηθεί σε τρία σημεία, τα οποία θα αποθηκευτούν και θα ορίσουν το σύστημα συντεταγμένων. Πρώτα πρέπει να ληφθεί το σημείο αναφοράς μετά ένα σημείο στον άξονα x για να προσδιοριστεί η θετική κατεύθυνσή του, και τέλος σε ένα σημείο στα θετικά του άξονα y. [43]



Σχήμα 4.49: Προσδιορισμός συστήματος συντεταγμένων ρομποτικού βραχίονα.

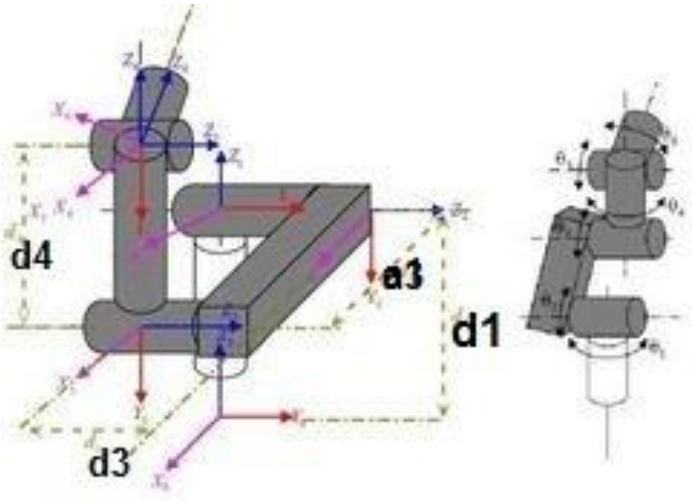
Στην περίπτωση της διάταξης λήψης εικόνων με κάμερα τα σημεία αυτά καθορίζονται από τη θέση του μοτίβου καλιμπραρίσματος. Το σημείο αναφοράς του συστήματος συντεταγμένων θα οριστεί σε ένα ύψος Z0 πάνω από το σημείο αναφοράς του μοτίβου καλιμπραρίσματος (σχήμα 4.45). Στην περίπτωση, τέλος της διάταξης λήψης εικόνων μέσω σκάνερ το επίπεδο συντεταγμένων θα είναι παράλληλο με αυτό του επιπέδου σάρωσης και πάλι μετατοπισμένο κατά ένα ύψος Z0.

Ο ρομποτικός βραχίονας που πρόκειται να χρησιμοποιηθεί είναι ο Staubli RX - 90L. [44] Όπως παρατηρούμε στο επόμενο σχήμα είναι ένας βραχίονας 6 περιστροφικών αρθρώσεων.



Σχήμα 4.50: Κίνηση αρθρώσεων του ρομποτικού βραχίονα Staubli RX-90L.

Στη συνέχεια χρησιμοποιώντας τη μεθοδολογία Denavit – Hartenberg θα αναπτυχθούν οι πίνακες μετασχηματισμού των συντεταγμένων του «καρπού» του ρομποτικού βραχίονα ως προς το παγκόσμιο σύστημα. Οι Denavit-Hartenberg πρότειναν έναν συστηματικό τρόπο τοποθέτησης των καρτεσιανών συστημάτων σε μία κινηματική αλυσίδα, με στόχο την τυποποίηση και απλοποίηση των αναγκαίων μετασχηματισμών. [45] Στο επόμενο σχήμα απεικονίζεται ο ορισμός των συστημάτων συντεταγμένων, που έγινε σε κάθε άρθρωση σύμφωνα με τη μεθοδολογία D-H.



Σχήμα 4.51: Ορισμός συστημάτων συντεταγμένων στις αρθρώσεις του ρομποτικού βραχίονα Staubli RX-90L σύμφωνα με τη μεθοδολογία D-H.

Για το βραχίονα της παρούσας εργασίας, οι παράμετροι Denavit – Hartenberg έχουν την εξής μορφή:

	ti	***ai***	***di***	***θι***
1	0	0	0	θ1
2	-90	0	0	θ2
3	0	a3	d3	θ3
4	90	0	d4	θ4
5	-90	0	0	θ5
6	90	0	0	θ6

Πίνακας 4.7: Πίνακας παραμέτρων της μεθόδου D-H για το ρομποτικό βραχίονα Staubli RX-90L.

Γνωρίζοντας τις παραπάνω παραμέτρους μπορούμε να αναπτύξουμε τους πίνακες μετασχηματισμού του συστήματος συντεταγμένων κάθε άρθρωσης σε σχέση με την προηγούμενη. Για να απλουστευθούν οι υπολογισμοί θεωρούμε ότι το παγκόσμιο σύστημα συντεταγμένων συμπίπτει με το σύστημα της βάσης συνεπώς d1=0.

$$T_i^{i-1} = \begin{bmatrix} cos\theta i & -sin\theta\iota \cdot costi & sin\theta\iota \cdot sinti & aicos\theta i \\ sin\theta i & cos\theta i \cdot costi & -cos\theta i \cdot sinti & aisin\theta i \\ 0 & sinti & costi & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.1)$$

Έτσι λοιπόν αντικαθιστώντας τις τιμές των παραμέτρων από τον πίνακα 4.50 προκύπτουν οι εξής πίνακες μετασχηματισμών κάθε συνδέσμου:

$$T_1^0 = \begin{bmatrix} cos\theta 1 & -sin\theta 1 & 0 & 0 \\ sin\theta 1 & cos\theta 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^1 = \begin{bmatrix} cos\theta 2 & 0 & -sin\theta 2 & 0 \\ sin\theta 2 & 0 & cos\theta 2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} cos\theta 3 & -sin\theta 3 & 0 & a3 \cdot cos\theta 3 \\ sin\theta 3 & cos\theta 3 & 0 & a3 \cdot sin\theta 3 \\ 0 & 0 & 1 & d3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_4^3 = \begin{bmatrix} cos\theta 4 & 0 & sin\theta 4 & 0 \\ sin\theta 4 & 0 & -cos\theta 4 & 0 \\ 0 & 1 & 0 & d4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_5^4 = \begin{bmatrix} cos\theta 5 & 0 & -sin\theta 5 & 0 \\ sin\theta 5 & 0 & cos\theta 5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_6^5 = \begin{bmatrix} cos\theta 6 & 0 & sin\theta 6 & 0 \\ sin\theta 6 & 0 & -cos\theta 6 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ο ομογενής μετασχηματισμός του άκρου ως προς τη βάση προκύπτει ως εξής:

$$T_6^0 = T_1^0 \cdot T_2^1 \cdot T_3^2 \cdot T_4^3 \cdot T_5^4 \cdot T_6^5 = \begin{bmatrix} r11 & r12 & r13 & pwx \\ r21 & r22 & r23 & pwy \\ r31 & r32 & r33 & pwz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Χρησιμοποιώντας τον παραπάνω μετασχηματισμό είναι δυνατόν να μεταφραστεί οποιοδήποτε σημείο του συστήματος συντεταγμένων του εργαλείου του ρομπότ ως προς το παγκόσμιο σύστημα συντεταγμένων, συναρτήσει της περιστροφής της κάθε άρθρωσης.

Έχοντας τη θέση και τον προσανατολισμό του άκρου του ρομποτικού βραχίονα είναι δυνατόν να προσδιοριστεί η περιστροφή των αρθρώσεων μέσω της επίλυσης των εξισώσεων της αντίστροφης κινηματικής.([46], [47])

ΚΕΦΑΛΑΙΟ 5

ΣΥΜΠΕΡΑΣΜΑΤΑ – ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

5.1 Ανασκόπηση της εργασίας και συμπεράσματα

Στα πλαίσια της εργασίας αναλύθηκαν ορισμένοι τρόποι εύρεσης και αντιστοίχισης μηχανουργικών αντικειμένων, μέσα από ένα πλήθος παρόμοιων σε εικόνα. Βασικός στόχος ήταν η αντιστοίχιση των αντικειμένων της εικόνας αναζήτησης με αυτά της βάσης δεδομένων και η εξαγωγή της θέσης και του προσανατολισμού τους.

Η αναζήτηση της βέλτιστης μεθόδου για τις συγκεκριμένες ανάγκες του προβλήματος, άρχισε σταδιακά με βάση την πολυπλοκότητα της κάθε μεθόδου. Αρχικά, παρατηρήθηκε πως τα αντικείμενα που εξετάστηκαν, είχαν διακριτά χαρακτηριστικά ως προς το σχήμα. Συνεπώς, μελετήθηκαν μέθοδοι που είχαν να κάνουν με γεωμετρικούς μετασχηματισμούς και σύγκριση των σχημάτων τον αντικειμένων. Τα δύο μειονεκτήματα αυτής της τεχνικής ήταν ο χρόνος εκτέλεσης και ο δύσκολος τρόπος προσδιορισμού του προσανατολισμού των αντικειμένων, καθώς αυτά είναι στη γενική περίπτωση συμμετρικά ως προς την περιστροφή. Κατά συνέπεια η αδυναμία προσδιορισμού του προσανατολισμού έκανε αδύνατη την εφαρμογή των γεωμετρικών μετασχηματισμών (περιστροφή) και εν τέλει τη σύγκριση.

Συνεχίζοντας με παρόμοιο σκεπτικό επίλυσης μελετήθηκαν μέθοδοι αναγνώρισης μοτίβου. Η διαφορά πλέον είναι ότι δεν είναι απαραίτητο να προσδιοριστεί το σχήμα των αντικειμένων παρά μόνο να απομονωθεί η σιλουέτα τους. Στη συνέχεια αυτή αναζητείται στην εικόνα αναζήτησης μέσω της διαδοχικής συνέλιξης των δύο εικόνων. Η διαδικασία αυτή εκτελείται ταχύτατα αλλά τα αποτελέσματα δεν είναι αξιόπιστα αν το αντικείμενο είναι υπό διαφορετικό προσανατολισμό στις δύο εικόνες. Για να επιλυθεί αυτό το πρόβλημα, προστέθηκε ένα στάδιο όπου η εικόνα περιστρέφεται κατά βήματα και αποθηκεύεται το βέλτιστο αποτέλεσμα και η αντίστοιχη γωνία. Αν και προστέθηκαν πολλαπλά βήματα για μια εικόνα ο χρόνος εκτέλεσης παρέμεινε μικρότερος της προηγούμενης μεθόδου και τα αποτελέσματα αρκετά αξιόπιστα. Οι όποιες λανθασμένες αντιστοιχίσεις, ή η μη εύρεση ορισμένων αντικειμένων είχαν να κάνουν κυρίως με την προοπτική προβολή.

Η προοπτική προβολή αποτελούσε πρόβλημα και για τους δύο παραπάνω τρόπους, καθώς τα αντικείμενα δεν ήταν «επίπεδα», συνεπώς ανάλογα με τη θέση τους στο επίπεδο εργασίας ήταν δυνατή η απόκρυψη μεγάλου μέρους πληροφορίας. Η αναζήτηση μεθόδων που μπορούν να ανταποκριθούν σε απόκρυψη πληροφορίας (occlusions) οδήγησε στη δοκιμή ανιχνευτών χαρακτηριστικών (feature detectors). Η λογική είναι ότι το αντικείμενο, πλέον, δε χαρακτηρίζεται από τη γεωμετρία του αλλά από τα χαρακτηριστικά του. Για παράδειγμα, ένα αντικείμενο φέρει ορισμένες γωνίες ή κάποιο κυλινδρικό τμήμα τα οποία το διαχωρίζουν από τα υπόλοιπα. Στην περίπτωση των αντικειμένων που εξετάζονται, όμως, δεν ήταν δυνατή η εύρεση ικανοποιητικού αριθμού σημείων για την εύρωστη και αξιόπιστη αντιστοίχιση.

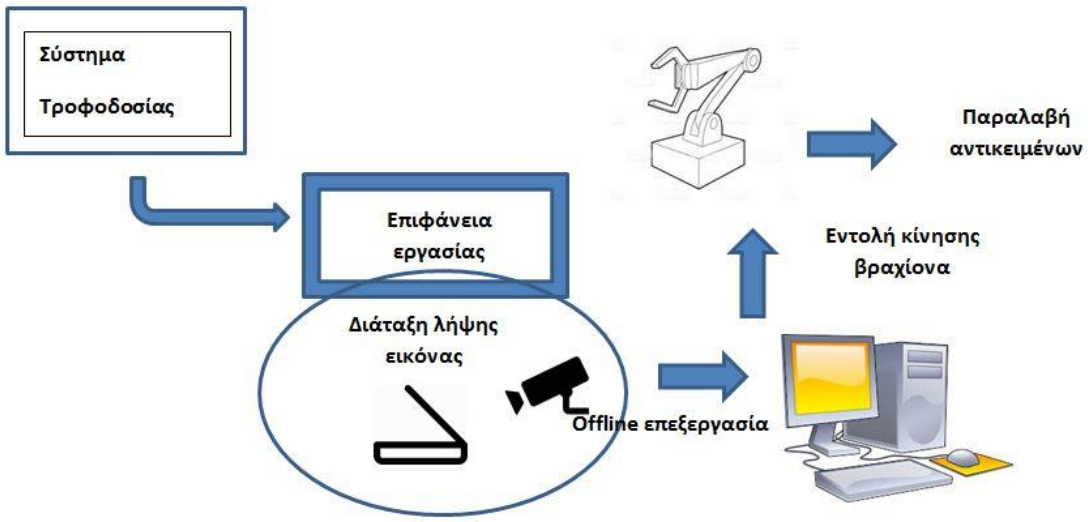
Το πρόβλημα που δημιουργούσε στην εξαγωγή αποτελεσμάτων η προοπτική προβολή οδήγησε στην αναζήτηση μιας διαφορετικής διάταξης λήψης εικόνων. Σε συνδυασμό με την υπόθεση ότι οι εικόνες των αντικειμένων στο επίπεδο εργασίας δε θα λαμβάνονται μεμονωμένα, προτάθηκε η χρήση σκάνερ. Με την παράλληλη προβολή, τα αποτελέσματα των αντιστοιχίσεων ήταν πλέον αξιόπιστα. Με αυτή τη διάταξη λήψης εικόνων πραγματοποιήθηκε και ο στόχος που είχε τεθεί τα αντικείμενα να μπορούν να τοποθετηθούν υπό τυχαία πλευρά στο επίπεδο. Είναι σαφές όμως, πως η χρήση της παράλληλης προβολής οδηγεί σε απόκρυψη μεγάλου μέρους πληροφορίας κάθε φορά. Για παράδειγμα αν δύο αντικείμενα έχουν μία ίδια πλευρά αλλά η υπόλοιπη γεωμετρία τους είναι εντελώς διαφορετική και τοποθετηθούν υπό αυτήν την πλευρά θα υπάρξει λανθασμένη αντιστοίχιση. Συνεπώς, πρέπει να υπάρξει μέριμνα για τέτοιου είδους αντικείμενα. Αν παραδείγματος χάρη, βρεθούν δύο τέτοια αντικείμενα το σύστημα δεν πρέπει να λάβει απόφαση αν δεν «δει» τις υπόλοιπες πλευρές. Τέλος, η χρήση σκάνερ διευκολύνει το στάδιο προεπεξεργασίας των εικόνων καθώς δημιουργεί ένα κλειστό σύστημα, όσον αφορά τις συνθήκες φωτισμού.

Κατά την δημιουργία της βάσης δεδομένων οι διάφορες πλευρές καταχωρήθηκαν ως διαφορετικές σαρώσεις των εκάστοτε πλευρών του αντικειμένου. Σε αντικείμενα παρόμοιων γεωμετρικών χαρακτηριστικών, όμως, με αυτά που χρησιμοποιήθηκαν, η βάση δεδομένων θα μπορούσε να δημιουργηθεί από τις πλάγιες όψεις και τις κατόψεις των αντικειμένων, καθώς η στήριξή των «μη επίπεδων» πλευρών στο επίπεδο σάρωσης, εισάγει λίγη παραμόρφωση στην τελική εικόνα. Φυσικά κάτι τέτοιο, θα ήταν αδύνατο με αντικείμενα με πολύ περίεργη γεωμετρία.

Τα αντικείμενα πλέον μπορούν να τοποθετούνται υπό τυχαία πλευρά στην επιφάνεια, αλλά όχι σε εντελώς τυχαία θέση. Δυο αντικείμενα δεν μπορούν να βρίσκονται πάρα πολύ κοντά μεταξύ τους ή το ένα τοποθετημένο επάνω στο άλλο. Αυτό θα έχει ως αποτέλεσμα να εκληφθεί από το πρόγραμμα σαν ένα αντικείμενο, του οποίου το περίγραμμα δεν θα μπορεί να ταυτοποιηθεί με αυτά της βάσης δεδομένων. Συνεπώς, η μέθοδος που αναπτύχθηκε εισάγει ορισμένους περιορισμούς στους μηχανισμούς τοποθέτησης των αντικειμένων στην επιφάνεια εργασίας.

5.2 Μελλοντική εργασία

Είναι φανερό πως η χρησιμοποίηση κάθε μεθόδου, επιφέρει περιορισμούς στο σχεδιασμό του συνολικού συστήματος. Είναι σημαντικό να δοθεί η δομή ενός συστήματος με βάση τη μέθοδο που επιλέχθηκε, ώστε να υπάρξουν προτάσεις για μελλοντικούς στόχους ή βελτιώσεις. Τα υποσυστήματα δίνονται διαγραμματικά στο επόμενο σχήμα.



Σχήμα 5.1: Διαγραμματική αναπαράσταση των διαφόρων υποσυστημάτων.

Ένα σύστημα με βάση τα παρόντα δεδομένα, λοιπόν, θα περιείχε ένα σύστημα τροφοδοσίας, το οποίο θα τοποθετούσε τα αντικείμενα στο επίπεδο εργασίας, υπό τυχαία πλευρά αλλά σε κάποια απόσταση μεταξύ τους. Αφού τοποθετηθούν τα αντικείμενα το σύστημα λήψης εικόνων (κάμερα ή σκάνερ) λαμβάνει την εικόνα αναζήτησης. Αυτή επεξεργάζεται από το πρόγραμμα (offline) και εξάγει τα αποτελέσματα τα οποία χρησιμοποιεί ένας ρομποτικός βραχίονας για να παραλάβει τα αντικείμενα που ενδιαφέρουν το χρήστη για περεταίρω επεξεργασία.

Κατά συνέπεια οι προτάσεις για μελλοντική εργασία μπορούν να χωριστούν σε δύο σκέλη. Το πρώτο αφορά προτάσεις για ένα σύστημα αυτής της λογικής, που

χρησιμοποιεί παρόμοιες τεχνικές αναγνώρισης των αντικειμένων, ενώ το δεύτερο προτείνει αλλαγές στις μεθόδους αναγνώρισης και κατ' επέκταση στη δομή του συστήματος.

Όσον αφορά ένα σύστημα με την παρούσα λογική, μπορεί να τροποποιηθεί η διάταξη λήψης εικόνων. Σημαντικό πρόβλημα με τη χρήση της κάμερας ήταν η προοπτική προβολή και η απόκρυψη πληροφορίας ως απόρροια. Ανάλογα με το μέγεθος της επιφάνειας εργασίας θα μπορούσαν να χρησιμοποιηθούν παραπάνω από μια κάμερες και να λειτουργούν συνεργατικά. Η κάθε κάμερα να ελέγχει για τα πιο κοντινά στο πεδίο της αντικείμενα και να αγνοεί τα υπόλοιπα, τα οποία θα ελέγχονται από τις εικόνες των άλλων καμερών. Αυτό προϋποθέτει έναν χωρισμό εξαρχής του επιπέδου εργασίας κατά τέτοιο τρόπο ώστε τα όρια των περιοχών ελέγχου της κάθε κάμερας να είναι επικαλυπτόμενα, ώστε να μην υπάρχουν αντικείμενα σε θέσεις που να μην μπορούν να αναγνωριστούν σε καμία εικόνα. Μια τέτοια διάταξη θα έλυνε το πρόβλημα της δημιουργίας της βάσης δεδομένων στην περίπτωση που η πλευρά τοποθέτησης των αντικειμένων είναι τυχαία. Αντί για ένα πλήθος συσκευών λήψεως θα μπορούσε να ήταν μετακινούμενη κάμερα σε διακριτά προσδιορισμένα εξ αρχής σημεία. [48]

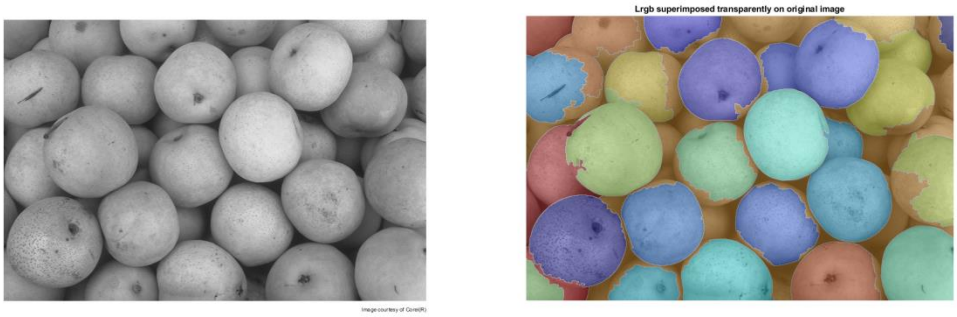
Όπως είδαμε η χρησιμοποίηση σκάνερ ως συσκευή λήψης έχει σαν αποτέλεσμα τη δημιουργία ενός κλειστού συστήματος, όσον αφορά τις συνθήκες φωτισμού. Αν η συσκευή λήψης είναι κάμερα θα πρέπει να δημιουργηθούν αυτές η συνθήκες, ώστε να μην επηρεάζεται το στάδιο της επεξεργασίας των εικόνων που λαμβάνονται. Αυτό μπορεί να επιτευχθεί με ένα αυτοματοποιημένο σύστημα φωτισμού το οποίο θα ελέγχεται από αισθητήρες τοποθετημένους σε διάφορα σημεία της επιφάνειας.

Αναφέρθηκε πως το σύστημα τροφοδοσίας και τοποθέτησης δεν πρέπει να τοποθετεί τα αντικείμενα πολύ κοντά μεταξύ τους, ή το ένα πάνω στο άλλο, καθώς η συγκεκριμένη μέθοδος αναγνώρισης θα αποτύχει. Ένας τρόπος για να επιλυθεί αυτό το πρόβλημα είναι να γίνεται ένας αρχικός έλεγχος για το πόσα αντικείμενα απαντώνται στην επιφάνεια. Αν ο αριθμός είναι μικρότερος από τον πραγματικό το σύστημα τοποθέτησης, ή ένα άλλο υποσύστημα θα επανατοποθετεί τα αντικείμενα.

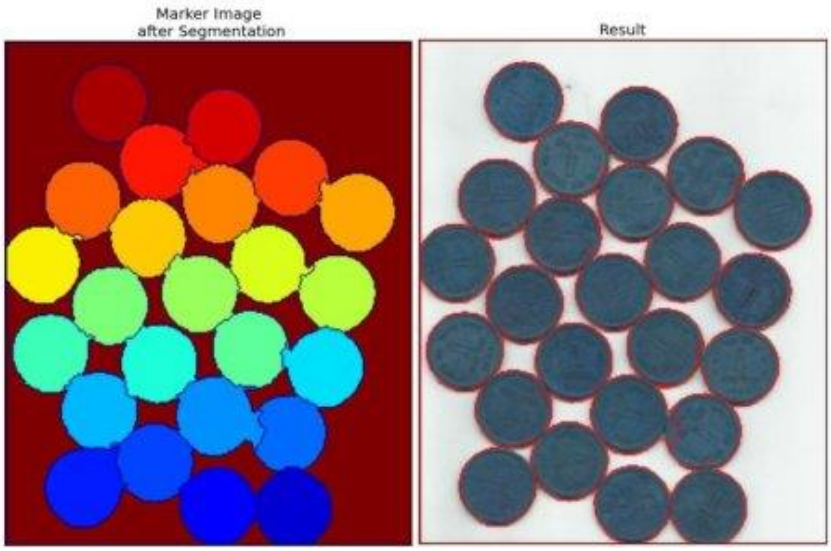
Απαραίτητη είναι η ύπαρξη ενός συστήματος διεπαφής με τον χρήστη. Η εύκολη επιλογή των αναγκών από το χρήστη θα μειώσει, όπως είναι κατανοητό τον υπολογιστικό φόρτο και θα επιταχύνει το χρόνο εκτέλεσης.

Ένα σύστημα τροφοδοσίας απαλλαγμένο από τον περιορισμό της τοποθέτησης των αντικειμένων σε κάποια απόσταση μεταξύ τους, απαιτεί την αλλαγή της μεθόδου

αναγνώρισης. Στη βιβλιογραφία υπάρχουν τεχνικές όπου διαχωρίζονται αντικείμενα που βρίσκονται πολύ κοντά μεταξύ τους, ή ακόμα διακρίνονται στοιβαγμένα αντικείμενα. [49] Η εφαρμογή αυτών των τεχνικών, όπως είναι κατανοητό, θα ήταν αδύνατη με μια διάταξη λήψης σκάνερ.



Σχήμα 5.2: Διαχωρισμός στοιβαγμένων αντικειμένων σε εικόνα.



Σχήμα 5.3: Εύρεση αντικειμένων που βρίσκονται σε μικρή απόσταση μεταξύ τους.

Ένας απώτερος στόχος είναι οι περιορισμοί στους μηχανισμούς τοποθέτησης των αντικειμένων να μην υπάρχουν. Να χρησιμοποιούταν μια μέθοδος με την οποία το πρόγραμμα θα μπορούσε να αντιστοιχίσει τυχαίως στοιβαγμένα αντικείμενα. [50] Ιδανικά τα αντικείμενα θα βρισκόταν στοιβαγμένα σε ένα «βαρέλι». Ο τομέας της μηχανικής όρασης είναι ταχέως αναπτυσσόμενος. Τα τελευταία χρόνια αναπτύσσονται συνεχώς νέες μέθοδοι για την επίλυση διαφορετικών προβλημάτων. Οι τεχνικές της μάθησης και τα νευρωνικά δίκτυα έχουν πλέον δώσει νέα δυναμική σε αυτόν τον τομέα. Μέσω αυτών των μεθόδων μπορεί να καταχωρηθεί ένας πολύ μεγάλος όγκος εικόνων για το κάθε αντικείμενο με σκοπό την εκμάθηση του προγράμματος ώστε να το αναγνωρίζει ακόμα και ανάμεσα σε πλήθος άλλων. Τέτοιες

τεχνικές δεν έχουν χρησιμοποιηθεί ακόμα ευρέως σε τέτοιου είδους αντικείμενα, αλλά χρησιμοποιούνται για την κατηγοριοποίηση διαφόρων αντικειμένων με βάση τα χαρακτηριστικά τους. [51] Ενδιαφέρον αποτελεί πως σε ένα τέτοιο σύστημα η αναγνώριση των αντικειμένων είναι δυνατόν να γίνει online, ακόμα και με τη χρήση ροής εικόνας – βίντεο.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] D. Vernon, *Machine Vision*. Prentice Hall International (UK) Ltd, 1991.

[2] J. D. L. E.N. Malamas, E.G.M. Petrakis, M. Zervakis, L. Petit, "A survey on industrial vision systems,applications and tools, Image and Vision Computing," *Image Vis. Comput.*, vol. 21, no. 2, pp. 171–188, 2003.

[3] a Ashbrook and N. Thacker, "Tutorial : Algorithms For 2-Dimensional Object Recognition .," *Biomed. Eng. (NY).*, no. 1996, pp. 1995–1998, 1998.

[4] M. B. Stegmann and D. D. Gomez, "A brief introduction to statistical shape analysis," *Informatics Math. Model.*, no. March, pp. 1–15, 2002.

[5] C. Solomon and T. Breckon, *Fundamentals of Digital Image Processing*. Wiley-Blackwell, 2011.

[6] A. Townsend, "Procrustes shape analysis," 2011. [Online]. Available: https://www.mathworks.com/examples/matlab/community/22656-procrustes-shape-analysis.

[7] Harvard, "The Assignment Problem and the Hungarian Method," *Introduction to Linear Algebra and Multivariable Calculus*. 2005.

[8] F. HungarianAlgorithm.com, "Hungarian algorithm," 2010. [Online]. Available: http://www.hungarianalgorithm.com/index.php.

[9] C. Rougier, J. Meunier, A. St-arnaud, J. Rousseau, and P. Shape, "Procrustes Shape Analysis for Fall Detection To cite this version : Procrustes Shape Analysis for Fall Detection," 2008.

[10] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," *Adv. Neural Inf. Process. Syst.*, vol. 546, pp. 831–837, 2000.

[11] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, 2002.

[12] F. OpenCV, "Template Matching - OpenCV Tutorials." [Online]. Available: https://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html.

[13] A. M. Alkababji, “Improving the Reliability of Object Recognition Based On Template Matching,” pp. 81–88, 2015.

[14] D. Vernon, “Applied Computer Vision Lecture 14 Object Recognition Template matching : normalized cross-correlation , chamfer matching.” pp. 1–57.

[15] U. of Center for Visualization & Virtual EnvironmentsKentucky, “Object Recognition and Template Matching.” pp. 1–23.

[16] K. Ahuja and P. Tuli, “Object Recognition by Template Matching Using Correlations and Phase Angle Method,” *Int. J. Adv. Res. ...*, vol. 2, no. 3, pp. 1368–1373, 2013.

[17] S. Gomathi and T. Santhanam, “Template Matching in Human Body Parts Recognition using Correlation,” vol. 10, no. 1, pp. 77–90, 2017.

[18] D. H. Ballard, “Generalizing the HT to detect arbitrary shapes,” *Pattern Recognit.*, vol. 13, no. 2, pp. 111–122, 1981.

[19] K. R. Sloan and D. H. Ballard, “Experience with the Generalized Hough Transform,” *Proc. 5th Int. Jt. Conf. Pattern Recognition, Miami Beach, Florida, USA*, pp. 174–179, 1980.

[20] F. OpenCV, “OPENCV IMPLEMENTATION OF GENERALIZED HOUGH TRANSFORM.” [Online]. Available: http://www.itriacasa.it/generalized-hough-transform/default.html.

[21] B. Vidhyapeeth Rajasthan Neelam Sharma and A. Professor Banasthali Vidhyapeeth Rajasthan, “An Overview of Various Template Matching Methodologies in Image Processing,” *Int. J. Comput. Appl.*, vol. 153, no. 10, pp. 975–8887, 2016.

[22] E. J. Kundukulam and A. Sudharson, “Implementing and Optimizing Template Matching Techniques for Home Automation,” *Indian J. Sci. Technol.*, vol. 8, no. August, pp. 74–78, 2015.

[23] F. Wikipedia, “Feature detection (computer vision).” [Online]. Available: https://en.wikipedia.org/wiki/Feature_detection_(computer_vision).

[24] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,”

Procedings Alvey Vis. Conf. 1988, p. 23.1-23.6, 1988.

[25] F. Mathworks, “Harris Features.” [Online]. Available: https://www.mathworks.com/help/vision/ref/detectharrisfeatures.html.

[26] D. G. Lowe, “Distinctive image features from scale invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, pp. 91–11020042, 2004.

[27] J. Clemons, “Sift : Scale Invariant Feature Transform,” *Transform*, vol. 2. pp. 91–110, 2007.

[28] F. OpenCV, “Introduction to SIFT (Scale-Invariant Feature Transform).” [Online]. Available: http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_sift_intro/py_sift_intro.html.

[29] D. Lowe, “Demo Software: SIFT Keypoint Detector.” .

[30] F. OpenCV, “Introduction to SURF (Speeded-Up Robust Features).” [Online]. Available: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html.

[31] Jean-Yves Bouguet, “Camera Calibration Toolbox for Matlab.” [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/.

[32] G. T. Laureano, M. Stela, V. De Paiva, and A. Soares, “Topological Detection of Chessboard Pattern for Camera Calibration,” *Proc. Int. Conf. Image Process. Comput. Vision, Pattern Recognit.*, 2013.

[33] F. Mathworks, “Camera Calibration.” [Online]. Available: https://www.mathworks.com/help/vision/ug/camera-calibration.html.

[34] R. Jain, R. Kasturi, and B. G. Schunck, “Machine Vision.” p. 549, 1995.

[35] J. Canny, “A Computational Approach to Edge Detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[36] R. Owens, “Analysis of Binary Images.” [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT2/node3.html.

[37] N. Foster, “Determining Object Orientation from a Single Image Using Multiple Information Sources,” 1984.

[38] S. Patel, P. Trivedi, V. Gandhi, and G. I. Prajapati, “2D Basic Shape Detection Using Region Properties,” vol. 2, no. 5, pp. 1147–1153, 2013.

[39] P. Tsarouchi, S. A. Matthaiakis, G. Michalos, S. Makris, and G. Chryssolouris, “A method for detection of randomly placed objects for robotic handling,” *CIRP J. Manuf. Sci. Technol.*, vol. 14, pp. 20–27, 2016.

[40] B. Iscimen, H. Atasoy, Y. Kutlu, S. Yildirim, and E. Yildirim, “Smart robot arm motion using computer vision,” *Elektron. ir Elektrotechnika*, vol. 21, no. 6, pp. 3–7, 2015.

[41] R. Kumar, S. Lai, S. Kumar, and P. Chand, “Object detection and recognition for a pick and place Robot,” 2014.

[42] Ν. Πεισμάνης, “Εντοπισμός θέσης κυλινδρικών τεμαχίων σε οριζόντια επιφάνεια μέσω ψηφιακής επεξεργασίας εικόνας,” Εθνικό Μετσόβιο Πολυτεχνείο, 2015.

[43] Adept Technology Inc, *V + Language User ’s Guide*. 1997.

[44] Staubli, “Arm - RX series 160 family Characteristics.” 2012.

[45] M. W. Spong, S. Hutchinson, and M. Vidyasagar, “FORWARD KINEMATICS: THE DENAVIT-HARTENBERG CONVENTION BT - Robot Dynamics and Control,” *Robot Dyn. Control*, pp. 1–32, 2003.

[46] Ε. Λεβαδιανός, “Διερεύνηση Μεθοδολογίας και Εργαλείων Κατασκευής Μοντέλων Εικονικής Πραγματικότητας για Ρομποτικά Κύτταρα Κατεργασιών,” Εθνικό Μετσόβιο Πολυτεχνείο, 2013.

[47] Σ. Μίχας, “Προγραμματισμός Τροχιάς και Τηλεπαρακολούθηση Λειτουργίας Ρομποτικού Βραχίονα σε Περιβάλλον Εικονικής Πραγματικότητας με χρήση Αισθητήρων Ηλεκτρονικών Συσκευών Ευρείας Κατανάλωσης,” Εθνικό Μετσόβιο Πολυτεχνείο, 2015.

[48] A. Coates and A. Y. Ng, “Multi-camera object detection for robotics,” *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 412–419, 2010.

[49] F. Open CV, “Image Segmentation with Watershed Algorithm.” [Online]. Available: http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_watershed/py_wat

ershed.html.

[50] B. R. Shuskov, “Detection and Pose Determination of a Part for Bin Picking,” Czech Technical University in Prague, 2017.

[51] A. Nehemiah and V. Leung, “Deep Learning for Computer Vision with MATLAB,” *Mathworks*. [Online]. Available: https://www.mathworks.com/company/newsletters/articles/deep-learning-for-computer-vision-with-matlab.html.

ΠΑΡΑΡΤΗΜΑ

(Κώδικας του συνολικού προγράμματος στο λογισμικό Matlab)

Main.m

```
clear
clc
close all;
tic

%% Preprocessing

Original = imread('21.jpg');
Original=imresize(Original,1/10); %Διάβασμα της εικόνας αναζήτησης
backg=imread('bg.jpg');
backg=imresize(backg,1/10); %Διάβασμα της εικόνας παρασκηνίου
backg=rgb2gray(backg);

%% main

% Καλείται η υπορουτίνα preprocessing που είναι υπεύθυνη για την
% επεξεργασία των εικόνων της βάσης δεδομένων καθώς και της εικόνας
% αναζήτησης
[Is,foo,dbImg ] = preprocessing( Original,backg );

%Δήλωση του path των φακέλων της βάσης δεδομένων
sdirectory = 'C:\Users\This\Desktop\review\6_template_matching
newsides\normalized corr\database';
p = regexp(genpath(sdirectory),['[^;]*'],'match');

%Δίνεται επιλογή στο χρήστη αν θα ψάξει όλα τα αντικείμενα της βάσης
δεδομένων
prompt = 'Do you want to compare every object of the database? Y/N ';
str = input(prompt,'s');
while (str~='Y')&&(str~='N')
  prompt = 'Type Y for yes or N for no';
  str = input(prompt,'s');
end


if (str=='Y') %Σε περίπτωση θετικής απάντησης

%Το πρόγραμμα θα εκτελεστεί ανατρέχοντας σε όλους τους φακέλους της
%βάσης δεδομένων (length(p))

for o=2:length(p)

subdirectory=p{1,o};
dbImg = dir([subdirectory '/*.jpg']);
for kk=1:length(dbImg)
foo.(['error' num2str(o-1) num2str(kk)])=[];
foo.(['MatchedObject' num2str(o-1) num2str(kk)])=[];
end

%length(dbImg) μας δίνει τον αριθμό των εικόνων στο φάκελο της βάσης
δεδομένων

for kk=1:length(dbImg)
    BestScore=-100000;

%Itm ορίζεται ως το αντικείμενο που θα προσπαθήσει το πρόγραμμα να
```

```
%βρει στην εικόνα αναζήτησης
Itm=foo.(['CropBw' num2str(o-1) num2str(kk)]);


Sitm=size(Itm);
Itm_dilation=floor(sqrt(Sitm(1)*Sitm(2))/80);

ItmAng=[];
Ybest=[];% mark best location y
Xbest=[];% mark best location x
BestScore1=[];

St=size(Itm);
Ss=size(Is);

Ang=0;
while (Ang<360) % Η εικόνα του αντικειμένου περιστρέφεται ανα 10
μοίρες κάθε φορά

 disp([num2str((Ang)/3.6) '% Scanned']);
 Itr=imrotate(Itm,-Ang);

%Καλείται η υποροutίνα template_match η οποία συνελίσσει την
περιστραμμένη εικόνα Itm με την εικόνα αναζήτησης Is με τη μέθοδο της
κανονικοποιημένης ετεροσυσχέτισης και μας δίνει το 'score' – κατά
πόσο δηλαδή βρέθηκε το αντικείμενο στην εικόνα

[score, y, x]=Template_match(Is,Itr,Itm_dilation);

%Τα καλύτερα score αποθηκεύονται καθώς και οι γωνίες για τις οποίες
επιτεύχθηκαν

    if score > 0.85*BestScore
           BestScore=score; % remember best score
           BestScore1=[BestScore1,BestScore];
           Ybest=[Ybest,y];% mark best location y
           Xbest=[Xbest,x];% mark best location x
           ItmAng=[ItmAng,Ang];
           %Bscale=scale;
     end;

     Ang=Ang+10;
end;

%Εύρεση των καλύτερων αντιστοιχίσεων

score=max(BestScore1);
index=find(BestScore1> (0.85*score));
Ang1=[];
Xbest1=[];
Ybest1=[];


for i=1:size(index,2)
    Ybest1=[Ybest1,Ybest(index(i))];
    Xbest1=[Xbest1,Xbest(index(i))];
    Ang1=[Ang1,ItmAng(index(i))];
end
```

```
Ismarked=Is;
for i=1:size(index,2)
%if BestScore>-100000% Display best match
Itr=imrotate(Itm,-Ang1(i));

   % [yy,xx] =find(Itr);
   % Ismarked=set2(Ismarked,[yy,xx],255,Ybest1(i),Xbest1(i));%Mark
best match on image
% Iborders=logical(zeros(size(Is)));
%Iborders=set2(Iborders,[yy,xx],1,Ybest1(i),Xbest1(i));
C=regionprops(Itr,'Centroid');
Ybest1(i)=Ybest1(i)+C(1).Centroid(2);
Xbest1(i)=Xbest1(i)+C(1).Centroid(1);
%end;

%Καλείται η υπορουτίνα shape comparison η οποία αξιολογεί τις
καλύτερες αντιστοιχίσεις που βρέθηκαν προηγουμένως.

[er,reg,object] = shape_comparison(Is,Itr,Xbest1(i),Ybest1(i) );

foo.(['error' num2str(o-1) num2str(kk)])=[foo.(['error' num2str(o-1)
num2str(kk)]),er];
foo.(['MatchedObject' num2str(o-1)
num2str(kk)])=[foo.(['MatchedObject' num2str(o-1)
num2str(kk)]),object];
end


foo.(['Angle' num2str(o-1) num2str(kk)])=Ang1;
%figure,imshow(Ismarked);
end



%% display-----------------------------------------------------
%Δημιουργία των πινάκων αντιστοίχισης και εξαγωγή αποτελεσμάτων

if o==2
displayMatch=zeros((length(p)-1),numel(reg));
dispAngles=360*ones((length(p)-1),numel(reg));
dispError=ones((length(p)-1),numel(reg));
end

%Αρχικά για κάθε αντικείμενο στην εικόνα αναζήτησης βρίσκεται η
καλύτερη αντιστοίχιση (το λάθος 'error' να είναι το ελάχιστο)

for j=1:length(dbImg)
for kk=1:numel(reg)

  ind=find(foo.(['MatchedObject' num2str(o-1) num2str(j)])==kk);

    if size(ind,2)>1
      A=[];
     for i=1:size(ind,2)
         A=[A,foo.(['error' num2str(o-1) num2str(j)])(ind(i))];
     end
      for i=1:size(ind,2)
```

```
     if foo.(['error' num2str(o-1) num2str(j)])(ind(i))> min(A)
        foo.(['error' num2str(o-1) num2str(j)])(ind(i))=1;
        foo.(['MatchedObject' num2str(o-1) num2str(j)])(ind(i))=0;
        foo.(['Angle' num2str(o-1) num2str(j)])(ind(i))=360;
     end
       end
    end
end
end

Match=zeros(length(dbImg),numel(reg));
Error=ones(length(dbImg),numel(reg));
Angles=360*ones(length(dbImg),numel(reg));

for j=1:length(dbImg)
for kk=1:numel(reg)
ind=find(foo.(['MatchedObject' num2str(o-1) num2str(j)])==kk);
  if ~isempty(ind)
    Match(j,kk)=foo.(['MatchedObject' num2str(o-1) num2str(j)])(ind);
    Error(j,kk)=foo.(['error' num2str(o-1) num2str(j)])(ind);
    Angles(j,kk)= foo.(['Angle' num2str(o-1) num2str(j)])(ind);
  end
end
end

for kk=1:numel(reg)

[y x]=find(Match==kk);
if size(y,1)>1
    A=[];
    for i=1:size(y,1)
      A=[A,Error(y(i),x(i))];
    end
    for i=1:size(y,1)
        if Error(y(i),x(i))>min(A)
            Error(y(i),x(i))=1;
            Match(y(i),x(i))=0;
            Angles(y(i),x(i))=360;
        end
    end
end
end

%Αν το λάθος 'error' μεγαλύτερο από μια τιμή κατωφλίου απορρίπτεται η
αντιστοίχιση
for j=1:length(dbImg)
for kk=1:numel(reg)

    if Error(j,kk)>0.027
        Error(j,kk)=1;
        Match(j,kk)=0;
        Angles(j,kk)=360;
    end
end
end

for i=1:numel(reg)
  [mr mc]=find(Match==i);
  if ~isempty(mr)
displayMatch(o-1,i)=mr;
```

```
dispAngles(o-1,i)=Angles(mr,mc);
dispError(o-1,i)=Error(mr,mc);
  end
end
end

%find min error of each dispError column (each object should appear
once...
% in the results)
mEC=min(dispError,[],1);%minErrorofColumn

for j=1:numel(reg)
    for kk=1:(length(p)-1)

        if dispError(kk,j)>mEC(1,j)

            dispError(kk,j)=1;
            displayMatch(kk,j)=0;
            dispAngles(kk,j)=360;

    end
end

end

%----------αν η απάντηση του χρήστη είναι αρνητική-------------------

else

% ο χρήστης ορίζει πιο φάκελο της βάσης δεδομένων
    prompt = 'Which object do you want to compare, Type the number of
the folder';
    o = input(prompt);
    while (o>(length(p)-1))
        prompt = 'Type a valid number of folder';
    o = input(prompt);
    end
    o=o+1;
%η διαδικασία πλέον είναι η ίδια μόνο που τώρα εκτελείται μόνο για
τον αριθμό των εικόνων του φακέλου που επιλέχθηκε

subdirectory=p{1,o};
dbImg = dir([subdirectory '/*.jpg']);
for kk=1:length(dbImg)
foo.(['error' num2str(o-1) num2str(kk)])=[];
foo.(['MatchedObject' num2str(o-1) num2str(kk)])=[];
end

%length(dbImg) μας δίνει τον αριθμό των εικόνων στο φάκελο της βάσης
δεδομένων

for kk=1:length(dbImg)
    BestScore=-100000;

%Itm ορίζεται ως το αντικείμενο που θα προσπαθήσει το πρόγραμμα να
%βρει στην εικόνα αναζήτησης
Itm=foo.(['CropBw' num2str(o-1) num2str(kk)]);

Sitm=size(Itm);
```

```
Itm_dilation=floor(sqrt(Sitm(1)*Sitm(2))/80);

ItmAng=[];
Ybest=[];% mark best location y
Xbest=[];% mark best location x
BestScore1=[];

St=size(Itm);
Ss=size(Is);

Ang=0;
while (Ang<360) % Η εικόνα του αντικειμένου περιστρέφεται ανα 10
μοίρες κάθε φορά

 disp([num2str((Ang)/3.6) '% Scanned']);
 Itr=imrotate(Itm,-Ang);

%Καλείται η υπορουτίνα template_match η οποία συνελίσσει την
περιστραμμένη εικόνα Itm με την εικόνα αναζήτησης Is με τη μέθοδο της
κανονικοποιημένης ετεροσυσχέτισης και μας δίνει το 'score' – κατά
πόσο δηλαδή βρέθηκε το αντικείμενη στην εικόνα

[score, y, x]=Template_match(Is,Itr,Itm_dilation);

%Τα καλύτερα score αποθηκεύονται καθώς και οι γωνίες για τις οποίες
επιτεύχθηκαν

    if score > 0.85*BestScore
           BestScore=score; % remember best score
           BestScore1=[BestScore1,BestScore];
           Ybest=[Ybest,y];% mark best location y
           Xbest=[Xbest,x];% mark best location x
           ItmAng=[ItmAng,Ang];
           %Bscale=scale;
     end;

     Ang=Ang+10;
end;

%Εύρεση των καλύτερων αντιστοιχίσεων

score=max(BestScore1);
index=find(BestScore1> (0.85*score));
Ang1=[];
Xbest1=[];
Ybest1=[];

for i=1:size(index,2)
    Ybest1=[Ybest1,Ybest(index(i))];
    Xbest1=[Xbest1,Xbest(index(i))];
    Ang1=[Ang1,ItmAng(index(i))];
end

Ismarked=Is;
for i=1:size(index,2)
%if BestScore>-100000% Display best match
Itr=imrotate(Itm,-Ang1(i));

   % [yy,xx] =find(Itr);
```

```
   % Ismarked=set2(Ismarked,[yy,xx],255,Ybest1(i),Xbest1(i));%Mark
best match on image
% Iborders=logical(zeros(size(Is)));
%Iborders=set2(Iborders,[yy,xx],1,Ybest1(i),Xbest1(i));
C=regionprops(Itr,'Centroid');
Ybest1(i)=Ybest1(i)+C(1).Centroid(2);
Xbest1(i)=Xbest1(i)+C(1).Centroid(1);
%end;

%Καλείται η υπορουτίνα shape comparison η οποία αξιολογεί τις
καλύτερες αντιστοιχίσεις που βρέθηκαν προηγουμένως.

[er,reg,object] = shape_comparison(Is,Itr,Xbest1(i),Ybest1(i) );

foo.(['error' num2str(o-1) num2str(kk)])=[foo.(['error' num2str(o-1)
num2str(kk)]),er];
foo.(['MatchedObject' num2str(o-1)
num2str(kk)])=[foo.(['MatchedObject' num2str(o-1)
num2str(kk)]),object];
end


foo.(['Angle' num2str(o-1) num2str(kk)])=Ang1;
%figure,imshow(Ismarked);
end

%% display-------------------------------------------------------
%Δημιουργία των πινάκων αντιστοίχισης και εξαγωγή αποτελεσμάτων

if o==2
displayMatch=zeros((length(p)-1),numel(reg));
dispAngles=360*ones((length(p)-1),numel(reg));
dispError=ones((length(p)-1),numel(reg));
end

for j=1:length(dbImg)
for kk=1:numel(reg)

  ind=find(foo.(['MatchedObject' num2str(o-1) num2str(j)])==kk);

    if size(ind,2)>1
      A=[];
     for i=1:size(ind,2)
         A=[A,foo.(['error' num2str(o-1) num2str(j)])(ind(i))];
     end
      for i=1:size(ind,2)
     if foo.(['error' num2str(o-1) num2str(j)])(ind(i))> min(A)
        foo.(['error' num2str(o-1) num2str(j)])(ind(i))=1;
        foo.(['MatchedObject' num2str(o-1) num2str(j)])(ind(i))=0;
        foo.(['Angle' num2str(o-1) num2str(j)])(ind(i))=360;
     end
       end
    end
end
end

Match=zeros(length(dbImg),numel(reg));
Error=ones(length(dbImg),numel(reg));
Angles=360*ones(length(dbImg),numel(reg));
```

```
for j=1:length(dbImg)
for kk=1:numel(reg)



ind=find(foo.(['MatchedObject' num2str(o-1) num2str(j)])==kk);
 if ~isempty(ind)
Match(j,kk)=foo.(['MatchedObject' num2str(o-1) num2str(j)])(ind);
Error(j,kk)=foo.(['error' num2str(o-1) num2str(j)])(ind);
Angles(j,kk)= foo.(['Angle' num2str(o-1) num2str(j)])(ind);
 end


end
end

for kk=1:numel(reg)

[y x]=find(Match==kk);
if size(y,1)>1
    A=[];
    for i=1:size(y,1)
      A=[A,Error(y(i),x(i))];
    end
    for i=1:size(y,1)
        if Error(y(i),x(i))>min(A)
            Error(y(i),x(i))=1;
            Match(y(i),x(i))=0;
            Angles(y(i),x(i))=360;
        end
    end
end
end

for j=1:length(dbImg)
for kk=1:numel(reg)

    if Error(j,kk)>0.027
        Error(j,kk)=1;
        Match(j,kk)=0;
        Angles(j,kk)=360;
    end
end
end

for i=1:numel(reg)
  [mr mc]=find(Match==i);
  if ~isempty(mr)
displayMatch(o-1,i)=mr;
dispAngles(o-1,i)=Angles(mr,mc);
dispError(o-1,i)=Error(mr,mc);
  end
end


%find min error of each dispError column (each object should appear
once...
% in the results)
```

```
mEC=min(dispError,[],1);%minErrorofColumn

for j=1:numel(reg)
    for kk=1:(length(p)-1)

        if dispError(kk,j)>mEC(1,j)

            dispError(kk,j)=1;
            displayMatch(kk,j)=0;
            dispAngles(kk,j)=360;

    end
end

end
end

%% creation of the output matrix

% Find the centers of the objects in original image (image
coordinates)
 C=regionprops(Is,'Centroid');
% translate into world coordinates (Διαταξη λήψης εικόνων μέσω
σκάνερ)
Center=zeros(numel(reg),2);
    for j=1:numel(reg)
        Center(j,:)=C(j).Centroid(:,:);
    end

   world= zeros(numel(reg),2);


        for j=1:numel(reg)
            world(j,1)=10*Center(j,1)*210/1276;
            world(j,2)=10*Center(j,2)*297/1755;
        end


%Output Matrix
%rows (number of matches in image)
% columns (folder_of_object-matchingside - positionX -position Y -
angle)
[valuerow,valuecol]=find(displayMatch);
Rowsize=size(valuerow,1);
OUTPUT=zeros(Rowsize,5);

%first column -> Number of folder/ object in db/ rows of dispMatch
for i=1:Rowsize
    OUTPUT(i,1)=valuerow(i);
end
%second column -> Matching Side /Values of displayMatch
for i=1:Rowsize
    OUTPUT(i,2)=displayMatch(valuerow(i),valuecol(i));
end
%fifth column orientation -> Values of dispAngles
for i=1:Rowsize
    OUTPUT(i,5)=dispAngles(valuerow(i),valuecol(i));
end
% third column position x
```

```
for i=1:Rowsize
    OUTPUT(i,3)=world(i,1);
end
% fourth column position y
for i=1:Rowsize
    OUTPUT(i,4)=world(i,2);
end

% Edit row Names and column Names - extract Table to file
rowNames=[];
for i=1:Rowsize
    rowNames=[rowNames,{['MATCH_' num2str(i)]}];
end
colNames =
{'Folder_of_Object','Matching_Side','Position_X_mm','Position_Y_mm','
Orientation_degrees'};
sTable =
array2table(OUTPUT,'RowNames',rowNames,'VariableNames',colNames);
writetable(sTable,'OUTPUTtable.xls');
```

Preprocessing.m

```
function [ Is,foo,dbImg ] = preprocessing( Original,backg )
%Σε αυτήν την υπορουτίνα γίνεται η προεπεξεργασία των εικόνων

%% Ανάκτηση εικόνων από την βάση δεδομένων
sdirectory = 'C:\Users\This\Desktop\review\6_template_matching
newsides\normalized corr\database';
p = regexp(genpath(sdirectory),['[^;]*'],'match');


for i=2:length(p)
subdirectory=p{1,i};
dbImg = dir([subdirectory '/*.jpg']);
for k = 1:length(dbImg)
filename = [subdirectory '/' dbImg(k).name];
img = imread(filename);
img = rgb2gray(img);                    %Convert to grayscale
img=imresize(img,1/10);
grayImage=imabsdiff(img,backg);
%imshow(grayImage)

% Δημιουργία Μάσκας
mask = grayImage < 80;
mask=~mask;
%figure,imshow(mask);

se = strel('disk', 5, 0);
mask = imclose(mask, se);
mask = imfill(mask, 'holes');

%figure,imshow(mask);

% Mask the image
maskedImage = grayImage;
maskedImage(~mask) = 0;

%figure,imshow(maskedImage);

% Εφαρμογή φίλτρου Sobel για ανίχνευση περιγράμματος
hy = fspecial('sobel');
hx = hy';
Iy = imfilter(double(maskedImage), hy, 'replicate');
Ix = imfilter(double(maskedImage), hx, 'replicate');
gradmag = sqrt(Ix.^2 + Iy.^2);
foo.(['bw' num2str(i-1) num2str(k)])=gradmag>220;
foo.(['bw' num2str(i-1) num2str(k)])=bwareaopen(foo.(['bw' num2str(i-
1) num2str(k)]),50);
%figure, imshow(foo.(['bw' num2str(i-1) num2str(k)])),
title('Gradient magnitude (gradmag)')



foo.(['BoundB'   num2str(i-1) num2str(k)]) = regionprops(foo.(['bw'
num2str(i-1) num2str(k)]),...
   grayImage,{'BoundingBox'});

foo.(['CropBw' num2str(i-1) num2str(k)]) = imcrop(foo.(['bw'
num2str(i-1) num2str(k)]),...
```

```
    [foo.(['BoundB' num2str(i-1) num2str(k) ])(1).BoundingBox(1),...
    foo.(['BoundB' num2str(i-1) num2str(k) ])(1).BoundingBox(2),...
    foo.(['BoundB' num2str(i-1) num2str(k)])(1).BoundingBox(3),...
    foo.(['BoundB' num2str(i-1) num2str(k) ])(1).BoundingBox(4)]);
%figure, imshow(foo.(['CropBw' num2str(i-1) num2str(k)]))
end
end

%% Preprocessing Original Image

Original = rgb2gray(Original);
grayImage=imabsdiff(Original,backg);

mask = grayImage < 80;
mask=~mask;
%imshow(mask);

% Erode to shrink the mask.
se = strel('disk', 5, 0);
mask = imclose(mask, se);
mask = imfill(mask, 'holes');

%imshow(mask);

% Mask the image
maskedImage = grayImage;
maskedImage(~mask) = 0;

%imshow(maskedImage);

Iy = imfilter(double(maskedImage), hy, 'replicate');
Ix = imfilter(double(maskedImage), hx, 'replicate');
gradmag = sqrt(Ix.^2 + Iy.^2);
Is=gradmag>250;
Is=bwareaopen(Is,50);
%figure, imshow(Is)
end
```

Template_match.m

```
function [score,  y1, x1 ]=Template_match(Is,Itm,Itm_dilation )


%% Prepare template

It=double(Itm);

for f=1:1:Itm_dilation
    It=dilate(It);%DILATE Template
end

Im=double(It);

%% Prepare image

   Iedg=Is;
   Iedg=double(Iedg);
   Ir=Im;

%% Search for template in the image (Με τη μέθοδο κανονικοποιημένης
ετεροσυσχέτισης

Itr=normxcorr2(Ir,Iedg);

% find the location first 10 best matches and put them in the x,y
array
mx=max(max(Itr));
[y1,x1]=find(Itr==mx,1);

ss=size(Ir);

% normalize the location of the cordinate to so it will point on the
edge of the image

   score=Itr(y1,x1);
   y1=round(y1-ss(1));
   x1=round(x1-ss(2));

end

%  dilate binary image bw
function bw2=dilate(bw)
bw2=imdilate(bw,strel('square',3));%dilate image
end
```

Shape_comparison.m

```
function [ er, C,object] = shape_comparison(Is,Itr,Xbest1,Ybest1 )

C=regionprops(Is,'Centroid');% Is is input1
numObj=numel(C);


    center=[];
    for i=1:numObj
D=sqrt((Xbest1-C(i).Centroid(1))^2 + (Ybest1-
C(i).Centroid(2))^2);%Xbest,Ybest inputs 2,3
center=[center,D];
    end
%find minimum distance
minDis=min(center);
object=find(center==minDis,1)


 [ob,ol] = bwboundaries(Is,'noholes');


% we have now 1 cropped image and the object from db (Itr) %
% Itr is input 4
[B,L] = bwboundaries(Itr,'noholes');


sxhma=[];
for k = 1:length(B)
   boundary = B{k};
  sxhma=[sxhma;boundary];
end

%% Compare Objects

% Make matrices of the shapes of the objects

er=[];
sxhmaOR=[];

boundary2 =ob{object};
sxhmaOR=[sxhmaOR;boundary2];

% Sample shape of the image in db (ώστε οι δυο πίνακες να έχουν το
ίδιο μέγεθος)

[n m]=size(sxhma);

C1=sxhma;

[ny my]=size(sxhmaOR);

C2=sxhmaOR;

if n>ny
```

```
yp=mod(n,ny);

sr=(n-yp)/ny;
if sr>1
C1=downsample(sxhma,sr);
end

nr=size(C1,1);

yg=nr-ny;

kk=0;
while kk<yg
    temp=randperm(nr);
    random=temp(1);
   C1(random,:)=[];
    kk=kk+1;
    nr=nr-1;
end

elseif n<ny

    yp=mod(ny,n);

sr=(ny-yp)/n;
if sr>1
C2=downsample(sxhmaOR,sr);
end

nr=size(C2,1);

yg=nr-n;

kk=0;
while kk<yg
    temp=randperm(nr);
    random=temp(1);
   C2(random,:)=[];
    kk=kk+1;
    nr=nr-1;
end

else

   nr= size(C1,1);


end


% Translate shapes 1 & 2

muX = mean(C1,1);
muY = mean(C2,1);
X0 = C1 - repmat(muX, nr, 1);
Y0 = C2 - repmat(muY, nr, 1);
```

```
 % move to 0,0
M1 = min(X0);
M2 = min(Y0);


X0=X0-repmat(M1,nr,1);
Y0=Y0-repmat(M2,nr,1);

%plot of boundary
%figure,plot(X0(:,1), X0(:,2),'r*', Y0(:,1), Y0(:,2),'bo')


%επιλέγονται τυχαία 60 σημεία για να γίνει η σύγκριση
 X1=X0;
mege8os=size(X1,1);

ypoloipo=mege8os-60;
if ypoloipo>0

kk=0;
while kk<ypoloipo
    temp=randperm(mege8os);
    random=temp(1);
    X1(random,:)=[];
    kk=kk+1;
    mege8os=mege8os-1;
end
end


%Compare ( με βάση την απόσταση σημείο προς σημείο)
 Distance=[];
for jj=1:nr
    d=[];
for ii=1:mege8os

   d1= sqrt((Y0(jj,1)-X1(ii,1))^2+ (Y0(jj,2)-X1(ii,2))^2);
  d=[d,d1];

end
Distance=[Distance;d];
end

if mege8os<nr
X2=zeros(nr,(nr-mege8os));
Distance=[Distance,X2];
end

% Καλείται η υπορουτίνα hungarian για να εκτελέστει ο ουγγρικός
αλγόριθμος, που θα βρει τα αντιστοιχα σημεία των δύο πινάκων ώστε να
συγκριθούν

 c=hungarian(Distance);

Z=[];
 for ii=1:mege8os
```

```
     Z=[Z;Y0(c(ii),:)];
 end

 % Compare Z and X1 matrixes

error=[];
for ii=1:mege8os

   dis=sqrt((X1(ii,1)-Z(ii,1))^2+(X1(ii,2)-Z(ii,2))^2);
   error=[error;dis];
end

er=[er,mean(error)];

maxdis=max(X0);
maxd= sqrt((maxdis(1,1))^2 + (maxdis(1,2))^2);

er=er/maxd;

end
```

Hungarian.m

```
function [C,T]=hungarian(A)
%HUNGARIAN Solve the Assignment problem using the Hungarian method.
%
%[C,T]=hungarian(A)
%A - a square cost matrix.
%C - the optimal assignment.
%T - the cost of the optimal assignment.

% Adapted from the FORTRAN IV code in Carpaneto and Toth, "Algorithm
548:
% Solution of the assignment problem [H]", ACM Transactions on
% Mathematical Software, 6(1):104-111, 1980.

% v1.0  96-06-14. Niclas Borlin, niclas@cs.umu.se.
%                 Department of Computing Science, Umeå University,
%                 Sweden.
%                 All standard disclaimers apply.

% A substantial effort was put into this code. If you use it for a
% publication or otherwise, please include an acknowledgement or at
least
% notify me by email. /Niclas



[m,n]=size(A);

if (m~=n)
    error('HUNGARIAN: Cost matrix must be square!');
end

% Save original cost matrix.
orig=A;

% Reduce matrix.
A=hminired(A);

% Do an initial assignment.
[A,C,U]=hminiass(A);

% Repeat while we have unassigned rows.
while (U(n+1))
    % Start with no path, no unchecked zeros, and no unexplored rows.
    LR=zeros(1,n);
    LC=zeros(1,n);
    CH=zeros(1,n);
    RH=[zeros(1,n) -1];

    % No labelled columns.
    SLC=[];

    % Start path in first unassigned row.
    r=U(n+1);
    % Mark row with end-of-path label.
    LR(r)=-1;
```

```
    % Insert row first in labelled row set.
    SLR=r;

    % Repeat until we manage to find an assignable zero.
    while (1)
        % If there are free zeros in row r
        if (A(r,n+1)~=0)
            % ...get column of first free zero.
            l=-A(r,n+1);

            % If there are more free zeros in row r and row r in not
            % yet marked as unexplored..
            if (A(r,l)~=0 && RH(r)==0)
                % Insert row r first in unexplored list.
                RH(r)=RH(n+1);
                RH(n+1)=r;

                % Mark in which column the next unexplored zero in
this row
                % is.
                CH(r)=-A(r,l);
            end
        else
            % If all rows are explored..
            if (RH(n+1)<=0)
                % Reduce matrix.
                [A,CH,RH]=hmreduce(A,CH,RH,LC,LR,SLC,SLR);
            end

            % Re-start with first unexplored row.
            r=RH(n+1);
            % Get column of next free zero in row r.
            l=CH(r);
            % Advance "column of next free zero".
            CH(r)=-A(r,l);
            % If this zero is last in the list..
            if (A(r,l)==0)
                % ...remove row r from unexplored list.
                RH(n+1)=RH(r);
                RH(r)=0;
            end
        end

        % While the column l is labelled, i.e. in path.
        while (LC(l)~=0)
            % If row r is explored..
            if (RH(r)==0)
                % If all rows are explored..
                if (RH(n+1)<=0)
                    % Reduce cost matrix.
                    [A,CH,RH]=hmreduce(A,CH,RH,LC,LR,SLC,SLR);
                end

                % Re-start with first unexplored row.
                r=RH(n+1);
            end

            % Get column of next free zero in row r.
            l=CH(r);
```

```
            % Advance "column of next free zero".
            CH(r)=-A(r,l);

            % If this zero is last in list..
            if(A(r,l)==0)
                % ...remove row r from unexplored list.
                RH(n+1)=RH(r);
                RH(r)=0;
            end
        end

        % If the column found is unassigned..
        if (C(l)==0)
            % Flip all zeros along the path in LR,LC.
            [A,C,U]=hmflip(A,C,LC,LR,U,l,r);
            % ...and exit to continue with next unassigned row.
            break;
        else
            % ...else add zero to path.

            % Label column l with row r.
            LC(l)=r;

            % Add l to the set of labelled columns.
            SLC=[SLC l];

            % Continue with the row assigned to column l.
            r=C(l);

            % Label row r with column l.
            LR(r)=l;

            % Add r to the set of labelled rows.
            SLR=[SLR r];
        end
    end
end

% Calculate the total cost.
T=sum(orig(logical(sparse(C,1:size(orig,2),1))));


function A=hminired(A)
%HMINIRED Initial reduction of cost matrix for the Hungarian method.
%
%B=assredin(A)
%A - the unreduced cost matris.
%B - the reduced cost matrix with linked zeros in each row.

% v1.0  96-06-13. Niclas Borlin, niclas@cs.umu.se.

[m,n]=size(A);


% Subtract row-minimum values from each row.
rowMin=min(A')';
A=A-rowMin(:,ones(1,n));
```

```
% Subtract column-minimum values from each column.
colMin=min(A);
A=A-colMin(ones(n,1),:);

% Get positions of all zeros.
[i,j]=find(A==0);

% Extend A to give room for row zero list header column.
A(1,n+1)=0;
for k=1:n
    % Get all column in this row.
    cols=j(k==i)';
    % Insert pointers in matrix.
    A(k,[n+1 cols])=[-cols 0];
end


function [A,C,U]=hminiass(A)
%HMINIASS Initial assignment of the Hungarian method.
%
%[B,C,U]=hminiass(A)
%A - the reduced cost matrix.
%B - the reduced cost matrix, with assigned zeros removed from lists.
%C - a vector. C(J)=I means row I is assigned to column J,
%              i.e. there is an assigned zero in position I,J.
%U - a vector with a linked list of unassigned rows.

% v1.0  96-06-14. Niclas Borlin, niclas@cs.umu.se.

[n,np1]=size(A);

% Initalize return vectors.
C=zeros(1,n);
U=zeros(1,n+1);

% Initialize last/next zero "pointers".
LZ=zeros(1,n);
NZ=zeros(1,n);

for i=1:n
    % Set j to first unassigned zero in row i.
    lj=n+1;
    j=-A(i,lj);

    % Repeat until we have no more zeros (j==0) or we find a zero
    % in an unassigned column (c(j)==0).

    while (C(j)~=0)
        % Advance lj and j in zero list.
        lj=j;
        j=-A(i,lj);

        % Stop if we hit end of list.
        if (j==0)
            break;
        end
    end

    if (j~=0)
```

```
        % We found a zero in an unassigned column.

        % Assign row i to column j.
        C(j)=i;

        % Remove A(i,j) from unassigned zero list.
        A(i,lj)=A(i,j);

        % Update next/last unassigned zero pointers.
        NZ(i)=-A(i,j);
        LZ(i)=lj;

        % Indicate A(i,j) is an assigned zero.
        A(i,j)=0;
    else
        % We found no zero in an unassigned column.

        % Check all zeros in this row.

        lj=n+1;
        j=-A(i,lj);

        % Check all zeros in this row for a suitable zero in another
row.
        while (j~=0)
            % Check the in the row assigned to this column.
            r=C(j);

            % Pick up last/next pointers.
            lm=LZ(r);
            m=NZ(r);

            % Check all unchecked zeros in free list of this row.
            while (m~=0)
                % Stop if we find an unassigned column.
                if (C(m)==0)
                    break;
                end

                % Advance one step in list.
                lm=m;
                m=-A(r,lm);
            end

            if (m==0)
                % We failed on row r. Continue with next zero on row
i.
                lj=j;
                j=-A(i,lj);
            else
                % We found a zero in an unassigned column.

                % Replace zero at (r,m) in unassigned list with zero
at (r,j)
                A(r,lm)=-j;
                A(r,j)=A(r,m);

                % Update last/next pointers in row r.
                NZ(r)=-A(r,m);
```

```
                LZ(r)=j;

                % Mark A(r,m) as an assigned zero in the matrix . . .
                A(r,m)=0;

                % ...and in the assignment vector.
                C(m)=r;

                % Remove A(i,j) from unassigned list.
                A(i,lj)=A(i,j);

                % Update last/next pointers in row r.
                NZ(i)=-A(i,j);
                LZ(i)=lj;

                % Mark A(r,m) as an assigned zero in the matrix . . .
                A(i,j)=0;

                % ...and in the assignment vector.
                C(j)=i;

                % Stop search.
                break;
            end
        end
    end
end

% Create vector with list of unassigned rows.

% Mark all rows have assignment.
r=zeros(1,n);
rows=C(C~=0);
r(rows)=rows;
empty=find(r==0);

% Create vector with linked list of unassigned rows.
U=zeros(1,n+1);
U([n+1 empty])=[empty 0];


function [A,C,U]=hmflip(A,C,LC,LR,U,l,r)
%HMFLIP Flip assignment state of all zeros along a path.
%
%[A,C,U]=hmflip(A,C,LC,LR,U,l,r)
%Input:
%A   - the cost matrix.
%C   - the assignment vector.
%LC  - the column label vector.
%LR  - the row label vector.
%U   - the
%r,l - position of last zero in path.
%Output:
%A   - updated cost matrix.
%C   - updated assignment vector.
%U   - updated unassigned row list vector.

% v1.0  96-06-14. Niclas Borlin, niclas@cs.umu.se.
```

```
n=size(A,1);

while (1)
    % Move assignment in column l to row r.
    C(l)=r;

    % Find zero to be removed from zero list..

    % Find zero before this.
    m=find(A(r,:)==-l);

    % Link past this zero.
    A(r,m)=A(r,l);

    A(r,l)=0;

    % If this was the first zero of the path..
    if (LR(r)<0)
        ...remove row from unassigned row list and return.
        U(n+1)=U(r);
        U(r)=0;
        return;
    else

        % Move back in this row along the path and get column of next
zero.
        l=LR(r);

        % Insert zero at (r,l) first in zero list.
        A(r,l)=A(r,n+1);
        A(r,n+1)=-l;

        % Continue back along the column to get row of next zero in
path.
        r=LC(l);
    end
end


function [A,CH,RH]=hmreduce(A,CH,RH,LC,LR,SLC,SLR)
%HMREDUCE Reduce parts of cost matrix in the Hungerian method.
%
%[A,CH,RH]=hmreduce(A,CH,RH,LC,LR,SLC,SLR)
%Input:
%A   - Cost matrix.
%CH  - vector of column of 'next zeros' in each row.
%RH  - vector with list of unexplored rows.
%LC  - column labels.
%RC  - row labels.
%SLC - set of column labels.
%SLR - set of row labels.
%
%Output:
%A   - Reduced cost matrix.
%CH  - Updated vector of 'next zeros' in each row.
%RH  - Updated vector of unexplored rows.

% v1.0  96-06-14. Niclas Borlin, niclas@cs.umu.se.
```

```
n=size(A,1);

% Find which rows are covered, i.e. unlabelled.
coveredRows=LR==0;

% Find which columns are covered, i.e. labelled.
coveredCols=LC~=0;

r=find(~coveredRows);
c=find(~coveredCols);

% Get minimum of uncovered elements.
m=min(min(A(r,c)));

% Subtract minimum from all uncovered elements.
A(r,c)=A(r,c)-m;

% Check all uncovered columns..
for j=c
    % ...and uncovered rows in path order..
    for i=SLR
        % If this is a (new) zero..
        if (A(i,j)==0)
            % If the row is not in unexplored list..
            if (RH(i)==0)
                % ...insert it first in unexplored list.
                RH(i)=RH(n+1);
                RH(n+1)=i;
                % Mark this zero as "next free" in this row.
                CH(i)=j;
            end
            % Find last unassigned zero on row I.
            row=A(i,:);
            colsInList=-row(row<0);
            if (length(colsInList)==0)
                % No zeros in the list.
                l=n+1;
            else
                l=colsInList(row(colsInList)==0);
            end
            % Append this zero to end of list.
            A(i,l)=-j;
        end
    end
end

% Add minimum to all doubly covered elements.
r=find(coveredRows);
c=find(coveredCols);

% Take care of the zeros we will remove.
[i,j]=find(A(r,c)<=0);

i=r(i);
j=c(j);

for k=1:length(i)
    % Find zero before this in this row.
```

```
    lj=find(A(i(k),:)==-j(k));
    % Link past it.
    A(i(k),lj)=A(i(k),j(k));
    % Mark it as assigned.
    A(i(k),j(k))=0;
end

A(r,c)=A(r,c)+m;
```

Set2.m

```
function [m2] = set2( m,k,v,y ,x )
% set value of  v in coordinates k of image m  with initial position
x and y,
if nargin<4
    y=0;
    x=0;
end;
if x<0
    x=0;
end;
if y<0
    y=0;
end;
[a,b]=size(k);
m2=m;
for f=1:1:a
    for jj=1:size(y)
    m2(k(f,1)+y(jj),k(f,2)+x(jj))=v;
    end
%m(6,6)=7;
end
end
```