Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Πτυχιακή Εργασία

Τίτλος: Καταμέτρηση Αντικειμένων Μέσω Πανοραμικής Φωτογράφισης.

Γαβαλάς Νικόλαος (ΑΜ:1929)

Επιβλέπων καθηγητής : Μαλάμος Αθανάσιος

Επιτροπή Αξιολόγησης : Παχουλάκης Ιωάννης
Μαλάμος Αθανάσιος
Παναγιωτάκης Σπυρίδων

Ημερομηνία παρουσίασης: 19/12/2014

Ευχαριστίες

Ευχαριστώ τον καθηγητή μου κ. Αθανάσιο Μαλάμο που μου εμπιστεύθηκε την εκπόνηση της παρούσας εργασίας και τον συμφοιτητή μου Κώστα Καπετανάκη που με ενθάρρυνε από την πρώτη στιγμή να προχωρήσω και για τις συμβουλές που μου έδωσε. Τέλος να ευχαριστήσω τον Μάρκο Ζάμπογλου για τις προτάσεις του που με οδήγησαν στην λύση ενός σημαντικού προβλήματος στην υλοποίηση της καταμέτρησης.

Abstract

The purpose of the preparation of this thesis is the study of technological solutions that exist and can be used to create an application which receives an image and then isolates the objects from its background , will separate the objects between them and then give a precise count of such items.

Σύνοψη

Ο σκοπός της εκπόνησης αυτής της πτυχιακής είναι η μελέτη των τεχνολογικών λύσεων που υπάρχουν και είναι δυνατόν να χρησιμοποιηθούν στην κατασκευή εφαρμογής η οποία θα δέχεται μία εικόνα και στην συνέχεια θα απομονώνει τα αντικείμενα από το φόντο θα τα ξεχωρίζει μεταξύ τους και θα μας δίνει μια ακριβής καταμέτρηση των αντικειμένων αυτών.

Πίνακας Περιεχομένων

Πίνακας Εικόνων

Λίστα Πινάκων

ΚΕΦΑΛΑΙΟ 1: Εισαγωγή

1.1 Λίγα λόγια για την ψηφιακή εικόνα

Σε έναν ηλεκτρονικό υπολογιστή θα συναντήσουμε δύο είδη εικόνων, τις διανυσματικές εικόνες και τις ψηφιογραφικές εικόνες. Οι διανυσματικές εικόνες δημιουργούνται από γραμμές , κύκλους και διάφορα άλλα σχήματα τα οποία προκύπτουν από μαθηματικές εξισώσεις οι οποίες εξισώσεις μας δίνουν την δυνατότητα να αλλάξουμε τα χαρακτηριστικά της εικόνας όπως χρώμα , μέγεθος κ.α. χωρίς να επηρεαστεί η ποιότητα και η ανάλυση της εικόνας. Από την άλλη πλευρά όμως μια εικόνα ενός αντικειμένου στον πραγματικό κόσμο για να μεταφερθεί σε ηλεκτρονική μορφή στον σκληρό δίσκο του υπολογιστή μας πρέπει να γίνει ψηφιακή, δηλαδή να μετατραπεί σε διακεκριμένο σήμα με την μορφή ψηφιακών πινάκων. Οι πίνακες αυτοί χωρίζουν την εικόνα σε εικονοστοιχεία (pixels) τα οποία είναι οι μικρότερες υποδιαιρέσεις μιας εικόνας, είναι δηλαδή τα άτομα τα οποία απαρτίζουν μια εικόνα. Κάθε εικονοστοιχείο περιέχει τα δεδομένα που απαιτούνται για το χτίσιμο της εικόνας από την οποία προέρχονται. Μια ψηφιακή εικόνα μπορεί να είναι δυαδική (binary) , μονοχρωματική αποχρώσεων του γκρι (gray-scale) ή έγχρωμη (color).

Η δυαδική εικόνα είναι η πιο απλή μορφή εικόνας , απαιτεί το μικρότερο χώρο αποθήκευσης και ελάχιστη υπολογιστική ισχύ για την επεξεργασία της . Έχει μόνο δύο στάθμες φωτεινότητας , το μαύρο που αντιστοιχεί στην τιμή 0 και το άσπρο που αντιστοιχεί στην τιμή 1. Στην gray-level εικόνα οι στάθμες είναι 256 και ξεκινούν από την τιμή 0 το οποίο αντιστοιχεί στο απόλυτο μαύρο και την τιμή 255 που αντιστοιχεί στο απόλυτο λευκό. Η έγχρωμη ψηφιακή εικόνα είναι η πιο ακριβής απεικόνιση της πραγματικότητας , το κάθε εικονοστοιχείο της έγχρωμης εικόνας μπορεί να πάρει 256 διαφορετικές αποχρώσεις οι οποίες είναι συνδυασμός των εξής τριών χρωμάτων, κόκκινο (Red) , πράσινο (Green) , μπλε (Blue).

1.2 Το χρωματικό μοντέλο RGB

Ένα χρωματικό μοντέλο είναι μια τυποποίηση για την σύνδεση των χρωμάτων με μεταβλητές έτσι ώστε να μπορεί να περιγραφεί το κάθε χρώμα με ακρίβεια. Παραδείγματα χρωματικών μοντέλων είναι RGB (κόκκινο, πράσινο, μπλε), CMYK (κυανό, ματζέντα, κίτρινο, μαύρο), EBZ (Απόχρωση, Κορεσμός, Ένταση), YIQ (επίσης γνωστή ως NTSC), και CIE XYZ.Το προσθετικό μοντέλο RGB θεωρεί ως πρωτεύοντα χρώματα τα βασικά χρώματα της ανθρώπινης όρασης, δηλαδή :

- Red (Κόκκινο)
- Green (Πράσινο)
- Blue (Μπλε)

Στο προσθετικό μοντέλο για να δημιουργηθεί ένα χρώμα γίνεται πρόσθεση των τριών πρωτευόντων χρωμάτων μεταξύ τους και στην κατάλληλη αναλογία σύμφωνα με το ποιο χρώμα θέλουμε να δημιουργηθεί.Το χρωματικό μοντέλο RGB χρησιμοποιείται στις συσκευές απεικόνισης όπως οθόνη τηλεόρασης και οθόνη υπολογιστή. Όταν η ένταση και των τριών χρωμάτων έχουν τιμή μηδέν τότε δημιουργείται το μαύρο χρώμα , ενώ όταν η ένταση έχει την μέγιστη τιμή (δηλαδή 255) τότε δημιουργείται το λευκό χρώμα.

1.3 Μορφοποιήσεις (formats) αρχείων εικόνας

Οι μορφοποιήσεις αρχείων εικόνας είναι η προτυποποιημένη διαδικασία που αφορά την συμπίεση της εικόνας αλλα και την αποθήκευση των χρωμάτων.Παρακάτω θα αναφέρουμε και θα αναλύσουμε τους τύπους μορφοποίησης αρχείων εικόνας που θα χρησιμοποιηθούν στην παρούσα εργασία.

1.3.1 JPEG (Joint Photographic Experts Group)

Η πιο διαδεδομένη μορφή κωδικοποίησης που χρησιμοποιείται για συμπίεση εικόνων που προβάλλονται στο διαδίκτυο. Δημιουργήθηκε από την ομάδα εργασίας Joint Photographic Experts του ISO (International Organization for Standardization). Βασίζεται στην αφαίρεση της χρωματικής πληροφορίας η οποία δεν είναι απαραίτητη για το ανθρώπινο μάτι έτσι ώστε να δει την εικόνα που μετατρέπουμε χωρίς ορατή διαφορά με την αρχική. Πλεονέκτημά του ότι διατηρεί όλη την γκάμα των αποχρώσεων του χρωματικού μοντέλου RGB προσφέροντας παράλληλα μικρό μέγεθος αρχείου χρησιμοποιώντας έναν αλγόριθμο απωλεστικής συμπίεσης ο οποίος κρατάει την απαραίτητη για την ποιοτική προβολή της εικόνας πληροφορία και αφαιρεί την υπόλοιπη.

1.3.2 PNG (Portable Network Graphics)

Το πρότυπο αυτό είναι ανοικτό. Σχεδιάστηκε με απώτερο σκοπό να αντικαταστήσει το gif για την παρουσίαση εικόνων στο Web. Χρησιμοποιεί συμπίεση χωρίς απώλεια πληροφορίας (μη απωλεστική συμπίεση) για την μείωση του όγκου των ψηφιακών εικόνων , με αποτέλεσμα να προβάλλουμε εικόνες πραγματικού χρώματος (δηλαδή 16.700.000 αποχρώσεις) με μειονέκτημά του το μεγάλο μέγεθος αρχείου. Το πρότυπο PNG υποστηρίζεται από αρκετές εφαρμογές υπολογιστή και από τις τελευταίες εκδόσεις των περισσότερων Browser. Ιδιαιτερότητά του ότι διατηρεί και την πληροφορία alpha channel.

1.4 Ψηφιακή φωτογραφική μηχανή

Μια ψηφιακή φωτογραφική μηχανή δημιουργεί φωτογραφίες απευθείας σε μορφή ψηφιακού αρχείου εικόνας χωρίς την ανάγκη χρήσης φιλμ και χημικών-μηχανικών μεθόδων που είχαμε συνηθίσει να χρησιμοποιούμε στις αναλογικές φωτογραφικές μηχανές. Αποτελείται από :

- Το κύκλωμα που μετατρέπει την εικόνα από αναλογική σε ψηφιακή (CCD ή CMOS).
- Το μέσο αποθήκευσης που χρησιμοποιείται για την αποθήκευση των ψηφιακών αρχείων (πχ. κάρτα μνήμης).

1.4.1 Αρχή λειτουργίας

Η διαδικασία που ακολουθούν οι περισσότερες σύγχρονες ψηφιακές φωτογραφικές μηχανές βασίζεται σε ένα κύκλωμα CCD ή CMOS που δημιουργεί το ψηφιακό είδωλο της εικόνας που περνάει μέσα από τον φακό της μηχανής :

1.4.2 Ανάλυση

Η ανάλυση μιας ψηφιακής φωτογραφικής μηχανής ορίζεται από το πλήθος των pixels που υπάρχουν στο κύκλωμα που χρησιμοποιεί η μηχανή .Πχ. ανάλυση 1 “Megapixel” δηλαδή ένα εκατομμύριο pixels. Συνηθισμένες τιμές ανάλυσης είναι :

- 640x480 (0.3 Megapixel)
- 1024x768 (0.8 Megapixel)
- 1600x1200 (2.1 Megapixel)
- 2048x1536 (3.2 Megapixel)
- 3200x2400 (7.5 Megapixel)

1.4.3 Τεχνολογία αισθητήρων εικόνας

Για την μετατροπή των εικόνων από αναλογικά σήματα φωτός σε ψηφιακά pixels χρησιμοποιούνται οι παρακάτω τεχνολογίες :

- CCD (Charged Coupling Devices) : Δημιουργούν υψηλής ποιότητας εικόνα με χαμηλό θόρυβο και είναι ευαίσθητοι στο φως αλλά είναι ακριβή τεχνολογία.
- CMOS (Complimentary Metal Oxide Semiconductor): Εχουν μικρότερη κατανάλωση ενέργειας εώς και 100 φορές λιγότερο από έναν αισθητήρα CCD , είναι φθηνή τεχνολογία αλλά υστερεί σε συνθήκες χαμηλού φωτισμού.

1.4.4 Αποθήκευση

Για τις ανάγκες τις αποθήκευσης των αρχείων εικόνων που ψηφιοποιεί μια ψηφιακή φωτογραφική μηχανή χρησιμοποιούνται διάφορες τεχνολογίες μνήμης όπως :

- Ενσωματωμένη μνήμη
- Κάρτες μνήμης τύπου Flash
- Σκληρός δίσκος
- Εγγράψιμα CD και DVD

1.4.5 Μορφοποίηση αρχείων

Οι πλειοψηφία των ψηφιακών φωτογραφικών μηχανών αποθηκεύουν τα αρχεία εικόνων σε μορφοποίηση JPEG λόγω των πλεονεκτημάτων τις απωλεστικής συμπίεσης και την δυνατότητα χρήσης μικρού αποθηκευτικού χώρου. Σε κάποιες μηχανές με μεγάλο αποθηκευτικό χώρο , της τάξης δεκάδων GB , υπάρχει η δυνατότητα χρήσης της ασυμπίεστης μορφής TIFF.

1.5 Κατακόρυφη φωτογράφιση

Κατακόρυφη φωτογραφία είναι η φωτογραφία που τραβάμε από αέρος και κατά το χρονικό διάστημα της λήψης ο άξονας της φωτογραφικής μηχανής είναι κατακόρυφος ή παρουσιάζει απόκλιση από την κατακόρυφο μέχρι 3 μοίρες. Οι λόγοι που την χρησιμοποιούμε είναι :

- Έχει σταθερή κλίμακα
- Δεν υπάρχει αλληλοεπικάλυψη των αντικειμένων που φωτογραφίζουμε
- Η μέτρηση αντικειμένων σε μια κατακόρυφη φωτογραφία είναι πιο εύκολη και πιο ακριβής από μια αντίστοιχη πλάγια φωτογραφία.

ΚΕΦΑΛΑΙΟ 2 : Εισαγωγή στο Matlab

2.1 Τι είναι το Matlab

Το Matlab είναι ένα μαθηματικό λογισμικό πακέτο και το όνομά του προέρχεται από τις λέξεις Matrix και Laboratory. Έχει ενσωματωμένες συναρτήσεις για μια ποικιλία υπολογισμών , αναλύσεων , δημιουργία γραφημάτων και παραγωγή animation. Χρησιμοποιείται σε πολλούς επιστημονικούς τομείς όπως των μαθηματικών της φυσικής της πληροφορικής και πολλών άλλων.Το Matlab υποστηρίζει τα γνωστά λειτουργικά συστήματα (Windows,Linux,Mac). Έχει την δυνατότητα επικοινωνίας με τις γλώσσες προγραμματισμού (Java, C, Fortran, VB net) .Οι συναρτήσεις που διαθέτει χρησιμοποιούνται για υπολογισμούς γραμμικής άλγεβρας, επεξεργασία σήματος , επεξεργασία εικόνας και για την δημιουργία 2-D και 3-D γραφικών. Το λογισμικό αυτό είναι έτσι δομημένο έτσι ώστε κάθε υπολογισμός να μετατρέπεται σε υπολογισμό μεταξύ πινάκων και διανυσμάτων.

2.2 Βασικές εντολές του Matlab

- A=Imread(‘όνομα αρχείου’)

Διαβάζει την εικόνα «όνομα αρχείου».

- Imshow(A) ή Imshow(‘όνομα αρχείου’)

Εμφανίζει την εικόνα που είναι αποθηκευμένη στον πίνακα A.

- Figure

Ανοίγει ένα νέο παράθυρο.

- Subplot

Χρησιμοποιείται για να παρουσιάσουμε σε ένα figure πολλές εικόνες μαζί.

- D=Size(A)

Επιστρέφει το μέγεθος της εικόνας A και αποθηκεύει τον αριθμό στην μεταβλητή D.

- [r c]=size(A)

Επιστρέφει το μέγεθος τη εικόνας A r γραμμές και c στήλες.

- clc

Καθαρίζει το παράθυρο εντολών

- clear all

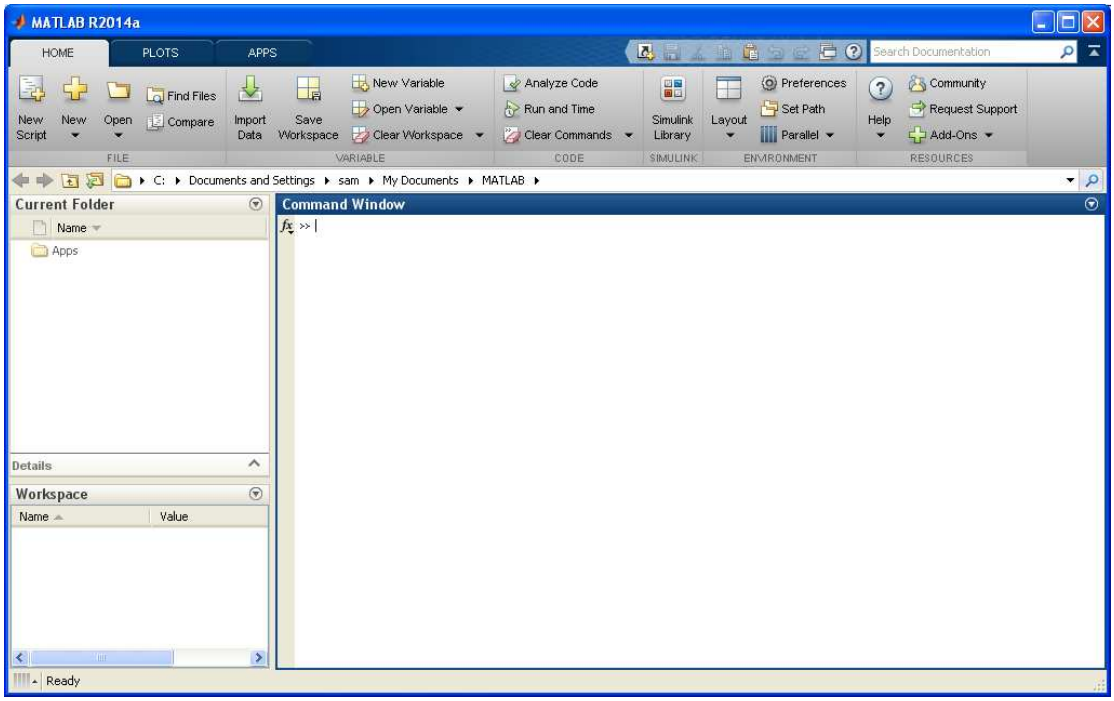
Καθαρίζει όλες τις μεταβλητές και τις συναρτήσεις

- exit

Κάνει έξοδο από το Matlab

2.3 Το περιβάλλον του Matlab

Η επιφάνεια εργασίας του Matlab (Matlab desktop) είναι το παράθυρο που εμφανίζεται κατά την εκκίνηση του προγράμματος .



Εικόνα 1 : Επιφάνεια εργασίας Matlab (Matlab desktop)

Η επιφάνεια εργασίας του Matlab αποτελείται από τα εξής επιμέρους παράθυρα

- Παράθυρο εντολών (Command Window) : Το βασικό παράθυρο στο οποίο πληκτρολογούνται οι εντολές και στο οποίο γίνεται η εισαγωγή των απαραίτητων δεδομένων και η εξαγωγή των αποτελεσμάτων.
- Παράθυρο τρέχοντος καταλόγου (Current Directory) : Εδώ αναγράφεται το σύνολο τον αρχείων που είναι αποθηκευμένα στον τρέχον κατάλογο του συστήματος και από αυτό το παράθυρο έχουμε την δυνατότητα της εκτέλεσης διάφορων επιλογών σχετικών με το αρχείο, όπως διαγραφή αρχείου , μετονομασία κ.α.
- Παράθυρο χώρου εργασίας (Workspace) : Στο παράθυρο αυτό απεικονίζονται όλες οι μεταβλητές οι οποίες εισάγονται και χρησιμοποιούνται στο παράθυρο εντολών.
- Παράθυρο ιστορικού εντολών (Command History) : Εδώ καταγράφονται όλες οι εντολές οι οποίες έχουν εκτελεστεί .

2.4 Ενσωματωμένες συναρτήσεις Matlab για ψηφιακή επεξεργασία εικόνας

Θα αναφέρουμε τις συναρτήσεις και τις εντολές που θα χρησιμοποιηθούν στην παρούσα εργασία για την ψηφιακή επεξεργασία της εικόνας και θα εξηγήσουμε τον τρόπο που λειτουργούν.

2.4.1 Συνάρτηση rgb2gray

Η εντολή rgb2gray μετατρέπει τις τιμές RGB σε τιμές της κλίμακας του γκρι (grayscale) σχηματίζοντας ένα σταθμισμένο άθροισμα των στοιχείων R,G και B.
(0.2989 * R + 0.5870 * G + 0.1140 * B)

Ο ευκολότερος τρόπος στο MATLAB για να μετατρέψουμε μια εικόνα RGB σε gray level εικόνα είναι χρησιμοποιώντας την συνάρτηση rgb2gray του Image Processing Toolbox.Παρακάτω περιγράφεται η σύνταξη της συνάρτησης:

I = rgb2gray (RGB) μετατρέπει την RGB εικόνα με την ασπρόμαυρη εικόνα I.
Η θεωρία πίσω από τη συνάρτηση rgb2gray του MATLAB είναι να μετατραπεί μια εικόνα truecolor από το χρωματικό χώρο RGB στο χρωματικό χώρο YIQ, και στη συνέχεια να λάβει την τιμή του στοιχείου Y ως τιμή της κλίμακας του γκρι. Το επιχείρημα είναι ότι στο χρωματικό χώρο YIQ η φωτεινότητα (Y) αντιπροσωπεύει την κλίμακα του γκρι, ενώ η απόχρωση (I) και ο κορεσμός (Q) αντιπροσωπεύουν την πληροφορία του χρώματος.
Η μήτρα μετασχηματισμού για να μετατραπεί μια εικόνα από RGB σε YIQ είναι το αντίστροφο του ακόλουθου πίνακα

1	0.956	0.621
1	0.272	0.647
1	1.106	1.703

Πίνακας 1 : Πίνακας μετασχηματισμού RGB σε YIQ
Πηγή : Ιστοσελίδα Mathworks του Matlab

Στον παρακάτω κώδικα MATLAB μια τέτοια μήτρα μετασχηματισμού υπολογίζεται και εμφανίζεται:

```
% Υπολογισμός μήτρας μετασχηματισμού.
matrix = [1.0 0.956 0.621; 1.0 -0.272 -0.647; 1.0 -1.106 1.703];
inverse = inv(matrix);

% Εμφάνιση πίνακα μετασχηματισμού.
format long
inverse
inverse =
```

0.298936021293776	0.587043074451121	0.114020904255103
0.595945743070799	-0.274388635745789	-0.321557107325010
0.211497340306828	-0.522910690302974	0.311413349996145

Πίνακας 2 : Μήτρα μετασχηματισμού
Πηγή : Ιστοσελίδα Mathworks του Matlab

Έχοντας πει την παραπάνω εξήγηση, μια έκφραση (με 4 ψηφία ακρίβειας) για να μετατρέψουμε μια εικόνα από το χρωματικό χώρο RGB στο χρωματικό χώρο YIQ είναι η ακόλουθη:

Y		0.2989	0.5870	0.1140	R
I	=	0.5959	-0.2744	-0.3216	x G
Q		0.2115	-0.5229	0.3114	B

Πίνακας 3 : Διαδικασία μετατροπής RGB σε YIQ
Πηγή : Ιστοσελίδα Mathworks του Matlab

Από τα αποτελέσματα που προκύπτουν από την προηγούμενη έκφραση, η τιμή της κλίμακας του γκρι ενός εικονοστοιχείου μπορεί να υπολογιστεί από την αντίστοιχη τιμή RGB ως εξής:

I = 0.2989 * R + 0.5870 * G + 0.1140 * B

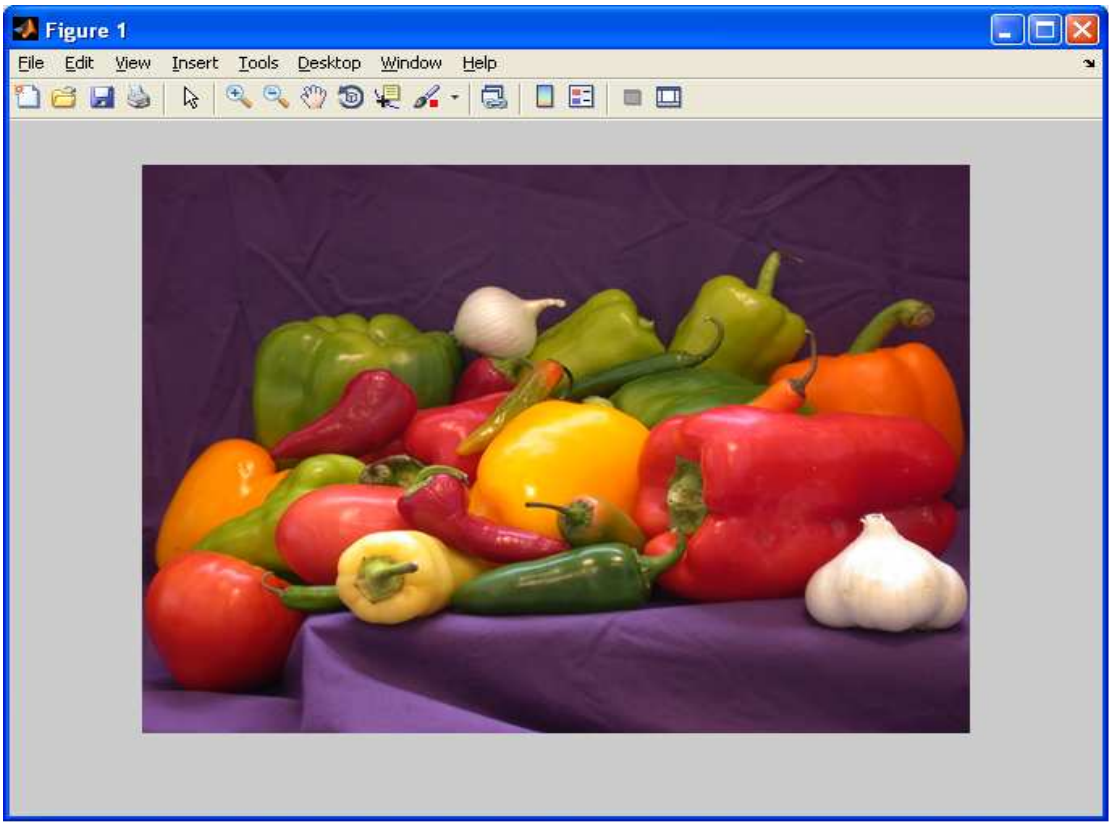
Εάν απαιτούνται περισσότερα από 4 ψηφία ακρίβειας , θα υπολογίσουμε τους συντελεστές που εμφανίζονται στον αντίστροφο πίνακα μετασχηματισμού, όπως απαιτείται. Παράδειγμα :Στο παρακάτω απόσπασμα κώδικα η RGB εικόνα (peppers.png) που περιλαμβάνεται στα αρχεία εγκατάστασης του MATLAB διαβάζεται και εμφανίζεται. Στη συνέχεια μετατρέπεται σε κλίμακα του γκρι χρησιμοποιώντας τη λειτουργία rgb2gray MATLAB και εμφανίζεται η εικόνα που προκύπτει:

```
RGB = imread('peppers.png');
figure; imshow(RGB);
title('Original image (RGB)');
I1 = rgb2gray(RGB);
figure; imshow(I1);
title('Grayscale image (using rgb2gray)');
```

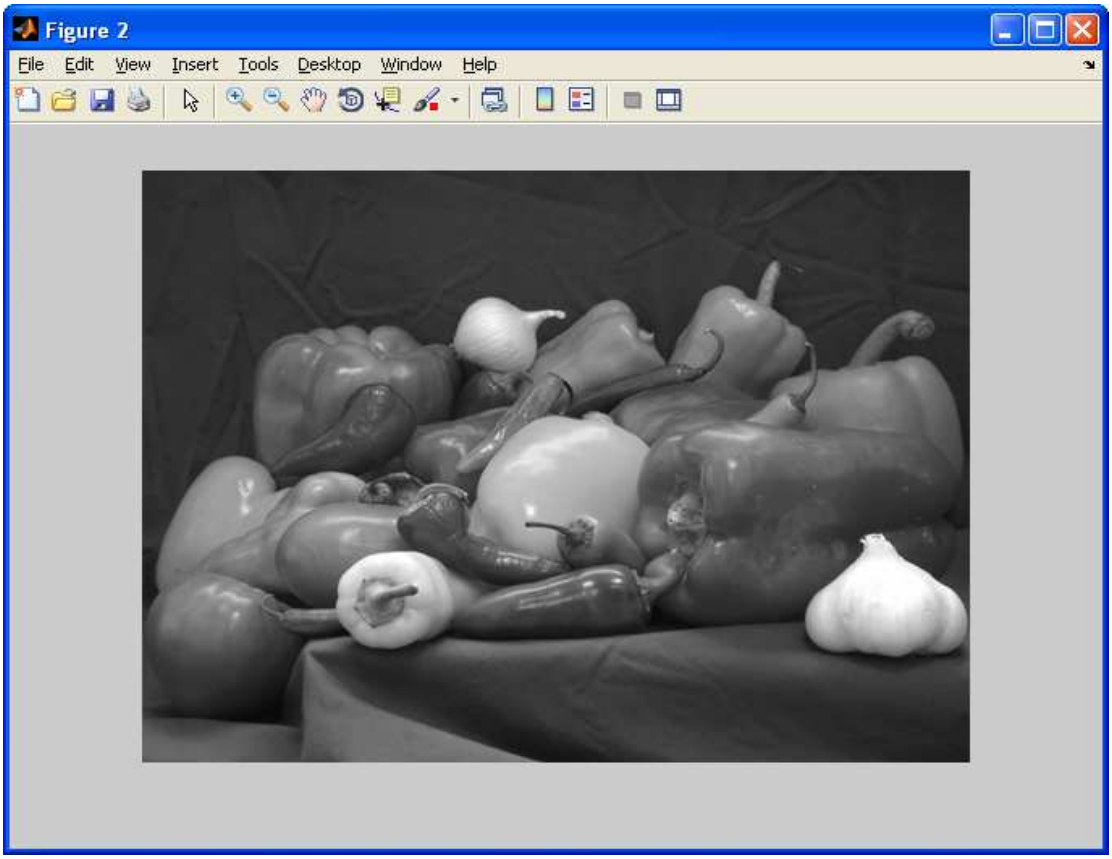
Παρακάτω η ίδια εικόνα RGB έχει μετατραπεί σε graylevel εικόνα χωρίς τη χρήση της συνάρτησης rgb2gray MATLAB.

```
I2 = RGB(:, :, 1) * 0.2989 + RGB(:, :, 2) * 0.5870 + RGB(:, :, 3) * 0.1140;
figure; imshow(I2);
title('Grayscale image (not using rgb2gray)');
```

Παρόλο που και οι δύο graylevel εικόνες I1 και I2 μπορεί να φαίνονται ίδιες σε έναν ανθρώπινο παρατηρητή, αριθμητικά μπορεί να διαφέρουν λίγο λόγω της περικοπής των συντελεστών που συμμετέχουν στον υπολογισμό της εικόνας I2 . Ένας άλλος παράγοντας είναι η αριθμητική προσέγγιση των μη ακέραιων τιμών σε ακέραιες τιμές.



Εικόνα 2 : Αρχική εικόνα
Πηγή : Εικόνα συστήματος Matlab

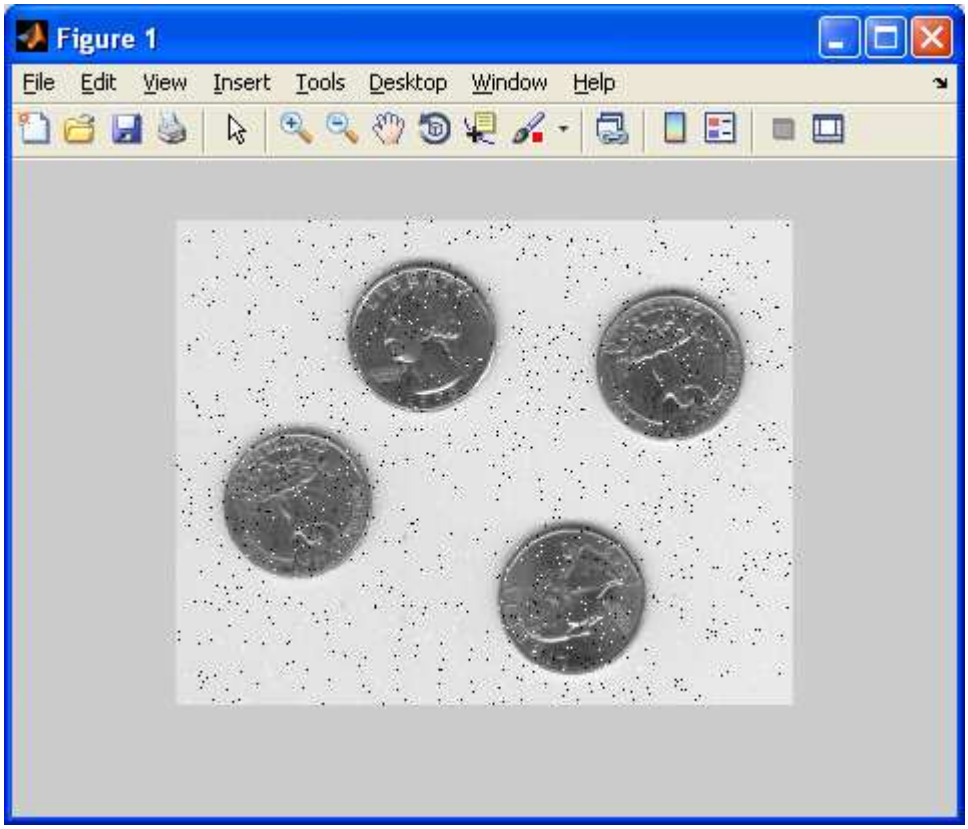


Εικόνα 3 : Μετατροπή αρχικής έγχρωμης εικόνας σε ασπρόμαυρη
Πηγή : Εικόνα συστήματος Matlab

2.4.2 Συνάρτηση medfilt2

Το φιλτράρισμα με ένα φίλτρο μεσαίας τιμής (median filter) είναι μια μη γραμμική τεχνική για τη μείωση του θορύβου και την εξομάλυνση των ακμών σε ψηφιακές εικόνες. Ο παρακάτω κώδικας προσθέτει θόρυβο σε μία εικόνα και στην συνέχεια χρησιμοποιώντας την συνάρτηση medfilt2 τον αφαιρεί.

```
I = imread('eight.tif');
J = imnoise(I,'salt & pepper',0.02);
K = medfilt2(J);
imshow(J), figure, imshow(K)
```



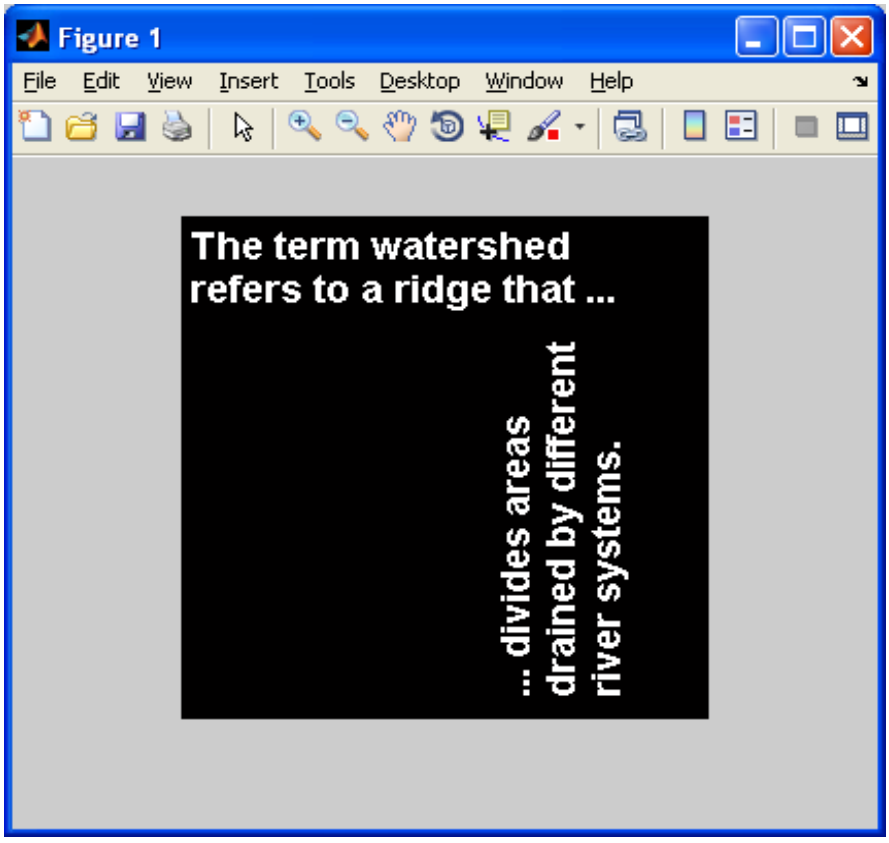
Εικόνα 4: Αρχική εικόνα στην οποία προσθέσαμε θόρυβο
Πηγή : Εικόνα συστήματος Matlab

Εικόνα 5: Χρησιμοποιώντας το φίλτρο μεσαίας τιμής αφαιρέσαμε το θόρυβο
Πηγή : Εικόνα συστήματος Matlab

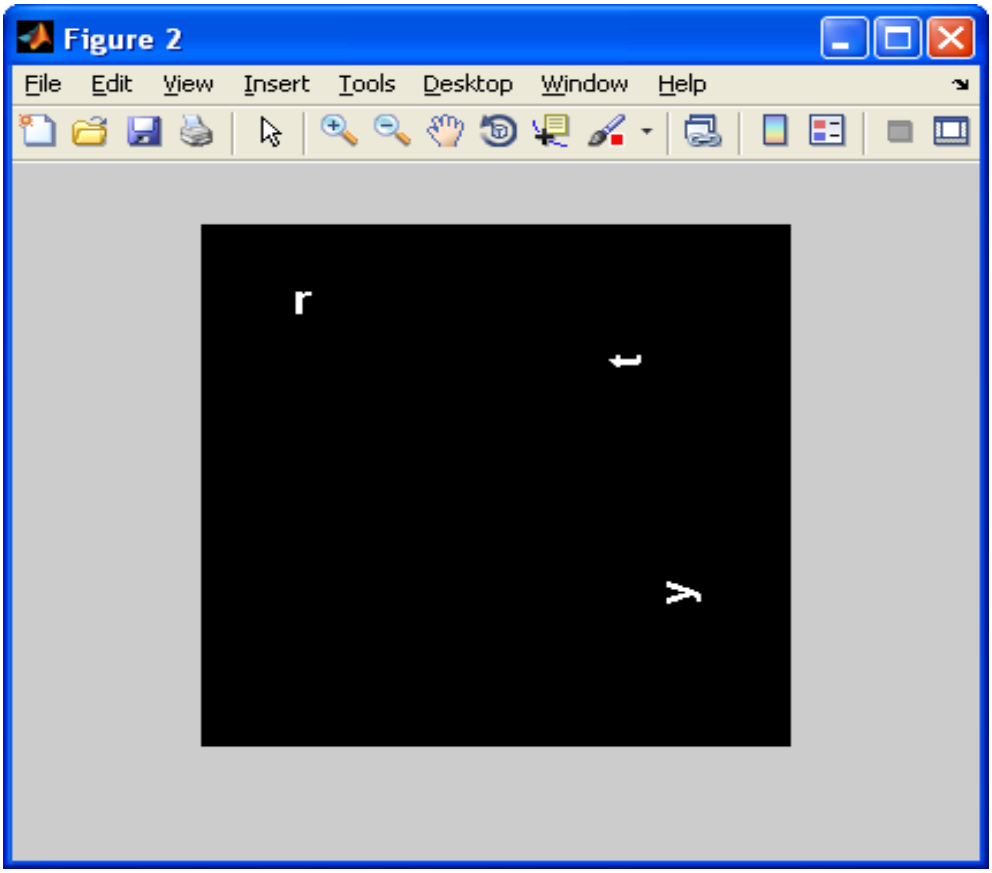
2.4.3 Συνάρτηση bwselect

Εμφανίζει στην οθόνη την binary εικόνα BW και δίνει στον χρήστη την δυνατότητα να επιλέξει με το ποντίκι τις συντεταγμένες (r,c) πατώντας διπλό κλικ πάνω στο σημείο που θέλουμε να επιλέξουμε και έπειτα enter ώστε να δηλώσουμε ότι ολοκληρώθηκε η διαδικασία επιλογής. Το BW2 είναι η binary εικόνα που περιέχει μόνο το σημείο που επιλέξαμε. Το n μπορεί να πάρει τις τιμές (4) ή (8) που δηλώνουν σε ποίο επίπεδο γειτονιάς θα εξετάσει αν υπάρχουν συνδεδεμένα αντικείμενα με το σημείο που επιλέξαμε. Ένα παράδειγμα παρακάτω χωρίς την χρήση του mouse αλλά με την απευθείας δήλωση των συντεταγμένων από τον χρήστη. Συντάσσεται BW2 = bwselect(BW,n).

```
BW1 = imread('text.png');
c = [43 185 212];
r = [38 68 181];
BW2 = bwselect(BW1,c,r,4);
imshow(BW1), figure, imshow(BW2)
```



Εικόνα 6: Αρχική binary εικόνα με κείμενο
Πηγή : Εικόνα συστήματος Matlab

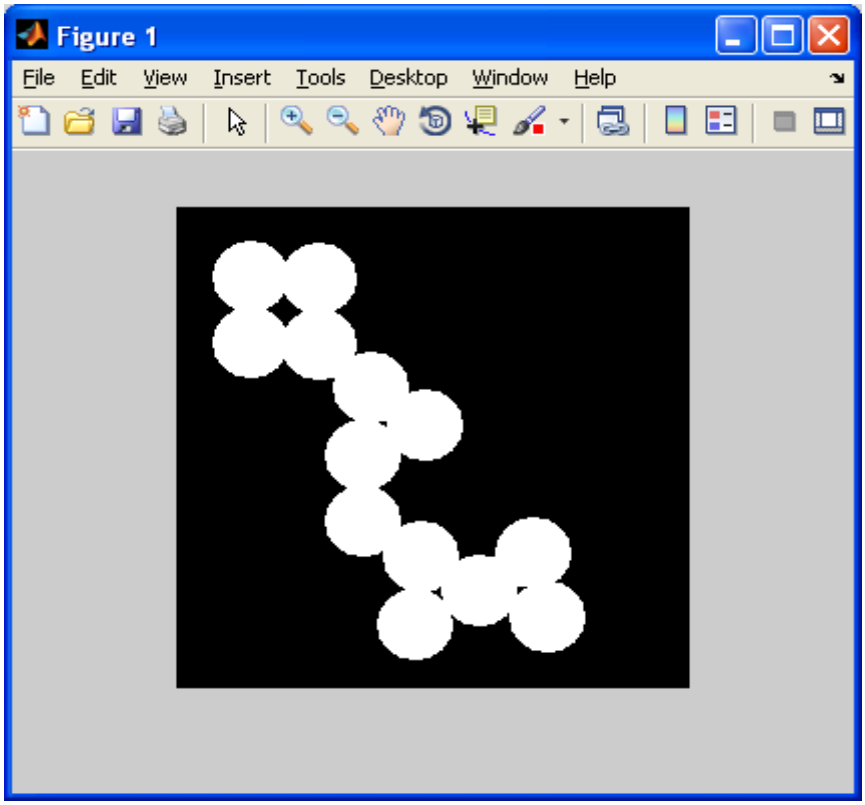


Εικόνα 7: Αφαιρέθηκαν τα γράμματα στις συντεταγμένες που ορίσαμε
Πηγή : Εικόνα συστήματος Matlab

2.4.4 Συνάρτηση bwarea

Υπολογίζει την επιφάνεια σε pixel των αντικειμένων στις binary εικόνες. Δέχεται ως είσοδο αριθμητικές ή λογικές παραστάσεις και επιστρέφει τιμή double.

```
BW = imread('circles.png');
imshow(BW);
bwarea(BW)
ans =

  1.4187e+04
```



Εικόνα 8: Συνδεδεμένοι δίσκοι σαν ένα αντικείμενο
Πηγή : Εικόνα συστήματος Matlab

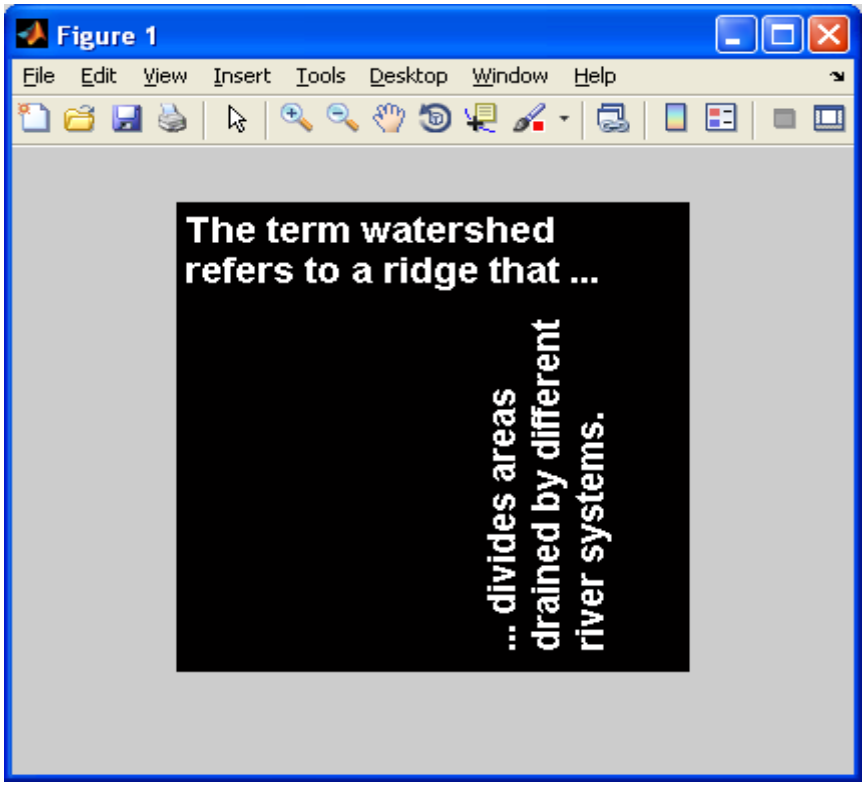
2.4.5 Συνάρτηση bwareaopen

Χρησιμοποιώντας την binary εικόνα BW επισημάνει όλα τα αντικείμενα με μέγεθος μικρότερο από P και δημιουργεί μία νέα binary εικόνα BW2 η οποία δεν περιέχει τα αντικείμενα αυτά. Συντάσσεται BW2 = bwareaopen(BW, P).

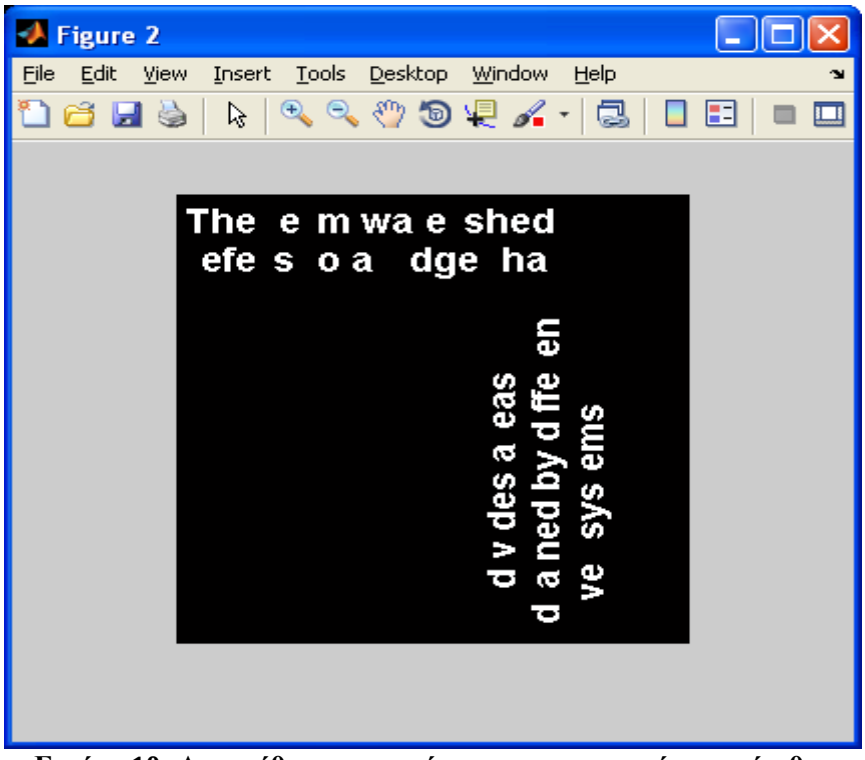
Παράδειγμα: αφαιρούμε όλα τα γράμματα με μέγεθος μικρότερο από 50 pixels.

```
BW = imread('text.png');
BW2 = bwareaopen(BW, 50);
 imshow(BW);
figure, imshow(BW2)
```

Η διαδικασία αυτή χρησιμοποιεί τον παρακάτω αλγόριθμο
Ορίζουμε τα συνδεδεμένα αντικείμενα:
CC = bwconncomp(BW, conn);
Υπολογίζουμε την περιοχή του κάθε αντικειμένου:
S = regionprops(CC, 'Area');
Αφαιρούμε όλα τα μικρά αντικείμενα:
L = labelmatrix(CC);
BW2 = ismember(L, find([S.Area] >= P));



Εικόνα 9: Αρχική binary εικόνα με κείμενο
Πηγή : Εικόνα συστήματος Matlab



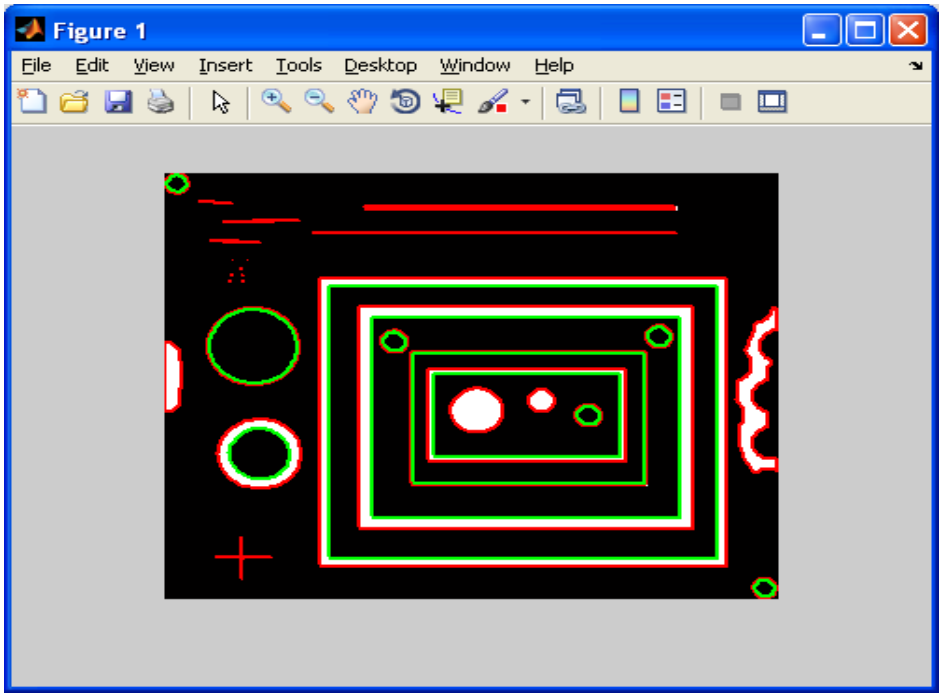
Εικόνα 10: Αφαιρέθηκαν τα γράμματα με το μικρότερο μέγεθος
Πηγή : Εικόνα συστήματος Matlab

2.4.6 Συνάρτηση bwboundaries

Εντοπίζει τα εξωτερικά και τα εσωτερικά περιγράμματα ενός αντικειμένου και επιστρέφει έναν πίνακα με τον αριθμό των αντικειμένων, χρησιμοποιώντας τον αλγόριθμο

εντοπισμού γείτονα του Moore με κριτήριο τερματισμού του Jacob. Παρακάτω ένα παράδειγμα όπου τα περιγράμματα των αντικειμένων έχουν κόκκινο χρώμα και οι τρύπες σε αυτά πράσινο χρώμα.

```
BW = imread('blobs.png');
[B,L,N] = bwboundaries(BW);
figure; imshow(BW); hold on;
for k=1:length(B),
   boundary = B{k};
   if(k > N)
      plot(boundary(:,2),...
         boundary(:,1),'g','LineWidth',2);
   else
      plot(boundary(:,2),...
         boundary(:,1),'r','LineWidth',2);
   end
end
```



Εικόνα 11: Πράσινο χρώμα για το περίγραμμα και κόκκινο για τις τρύπες
Πηγή : Εικόνα συστήματος Matlab

Αλγόριθμος εντοπισμού γειτονιάς Moore : Γειτονιά Moore ενός pixel, P ,είναι ένα σετ από 8 pixels τα οποία μοιράζονται μια κορυφή ή μια ακμή με αυτό το pixel.Το σετ αυτό ονομάζεται P1, P2, P3, P4, P5, P6, P7 και P8 και το βλέπουμε στο παρακάτω γράφημα .

P1	P2	P3
P8	P	P4
P7	P6	P5

Εικόνα 12 : Γειτονιά Moore

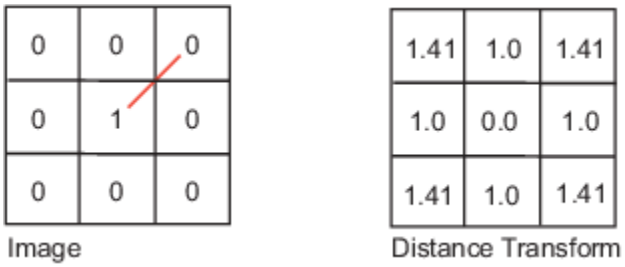
Έχοντας ως ψηφιακό πρότυπο ένα γκρουπ από μαύρα pixels και ως background άσπρα pixels σε ένα πλέγμα, πρέπει να εντοπίσουμε ένα μαύρο pixel το οποίο θα δηλώσουμε ως αρχικό pixel. Ο εντοπισμός του αρχικού pixel θα γίνει με τον εξής τρόπο. Ξεκινώντας από την κάτω αριστερή γωνία του πλέγματος διασχίζουμε κάθε Στήλη του πλέγματος από κάτω προς τα πάνω και κάθε γραμμή από αριστερά προς τα δεξιά μέχρι να βρούμε το πρώτο μαύρο pixel. Η βασική ιδέα είναι κάθε φορά που εντοπίζεις ένα μαύρο pixel (P) πηγαίνεις ένα κελί πίσω στο άσπρο pixel που ήσουν και από εκεί κάνεις έναν κύκλο γύρω από το (P) με την φορά του ρολογιού, δηλαδή δεξιόστροφα. Με λίγα λόγια επισκέπτεσαι κάθε pixel στην Moore γειτονιά του (P) μέχρι να εντοπίσεις το επόμενο μαύρο pixel το οποίο θα γίνει το επόμενο (P).Ο αλγόριθμος τερματίζει όταν επισκεφτεί το αρχικό σημείο για δεύτερη φορά. Τα μαύρα pixel που διασχίζει αποτελούν το περίγραμμα.

Η βασική αδυναμία του αλγόριθμου Moore είναι ο λανθασμένος τερματισμός του όταν έχουμε πολλά αντικείμενα, για αυτό το λόγο θα χρησιμοποιήσουμε ένα διαφορετικό κριτήριο τερματισμού το οποίο προτάθηκε από τον Jacob Eliosoft και από εκεί παίρνει και την ονομασία του ως «Κριτήριο τερματισμού του Jacob". Η ιδέα πάνω σε αυτό είναι ότι ο αλγόριθμος θα σταματήσει όταν επισκεφτεί το αρχικό pixel αλλά με τον ίδιο ακριβώς τρόπο με τον οποίο τον επισκέφτηκε την πρώτη φορά.

2.4.7 Συνάρτηση bwdist

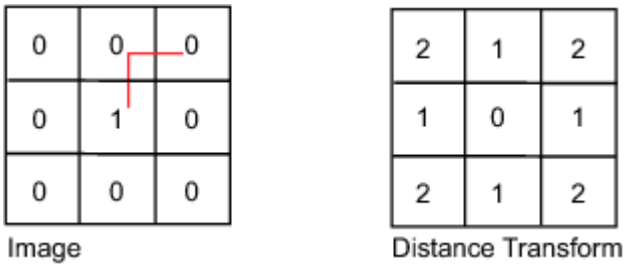
Η συνάρτηση bwdist εφαρμόζεται σε δυαδικές εικόνες και μας παρέχει μία μέτρηση για το βαθμό διαχωρισμού μεταξύ των σημείων της εικόνας στην οποία την εφαρμόζουμε. Αυτό επιτυγχάνεται με τον υπολογισμό της απόστασης μεταξύ κάθε pixel το οποίο έχει τιμή μηδέν (0) και του κοντινότερου μη μηδενικού pixel . Η συνάρτηση υποστηρίζει τα παρακάτω συστήματα μέτρησης.

- Η ευκλείδεια μέθοδος η οποία είναι η ευθεία γραμμή μεταξύ δύο pixel και είναι το default σύστημα μέτρησης το οποίο και θα χρησιμοποιηθεί στην παρούσα εργασία.



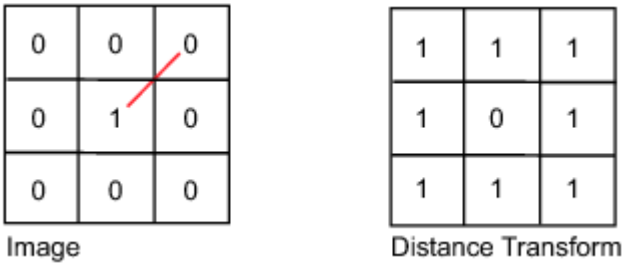
Εικόνα 13 : Euclidean
Πηγή : Ιστοσελίδα Mathworks του Matlab

- Η μέθοδος οικοδομικού τετραγώνου μετράει τα μονοπάτια μεταξύ των pixel σε συνδεδεμένη γειτονία των (4) σημείων. Τα pixel τα οποία ακουμπάνε οι ακμές τους απέχουν μία μονάδα ενώ αυτά που ακουμπάνε μόνο στις γωνίες , διαγώνια δηλαδή , απέχουν δύο μονάδες.



Εικόνα 14 : City-Block
Πηγή : Ιστοσελίδα Mathworks του Matlab

- Η μέθοδος σκάκι μετράει τα μονοπάτια μεταξύ των pixel σε συνδεδεμένη γειτονία των (8) σημείων. Τα pixel τα οποία ακουμπάνε οι ακμές τους και αυτά που ακουμπάνε μόνο διαγώνια απέχουν μία μονάδα.



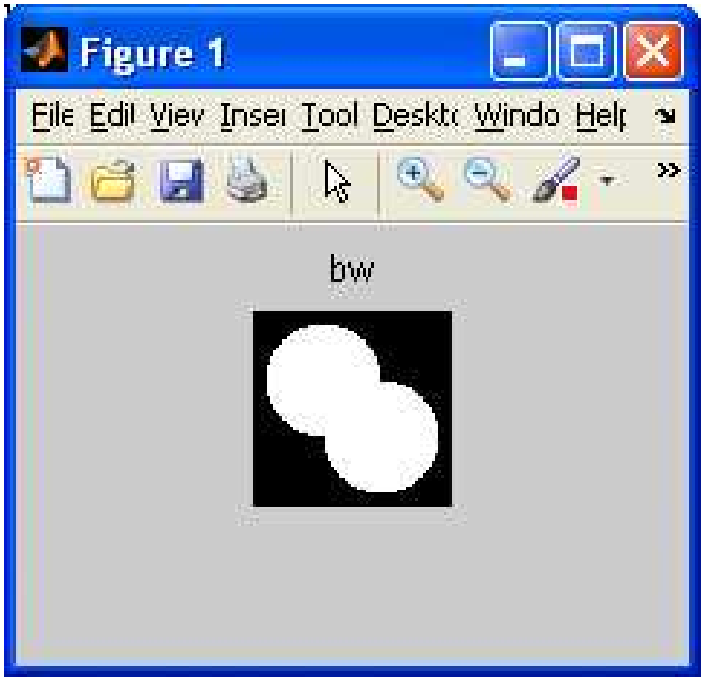
Εικόνα 15 : Chessboard
Πηγή : Ιστοσελίδα Mathworks του Matlab

- Η μέθοδος Quasi-Euclidean μετράει την συνολική ευκλείδεια απόσταση κατά μήκος των οριζόντιων , κάθετων και διαγώνιων γραμμών.

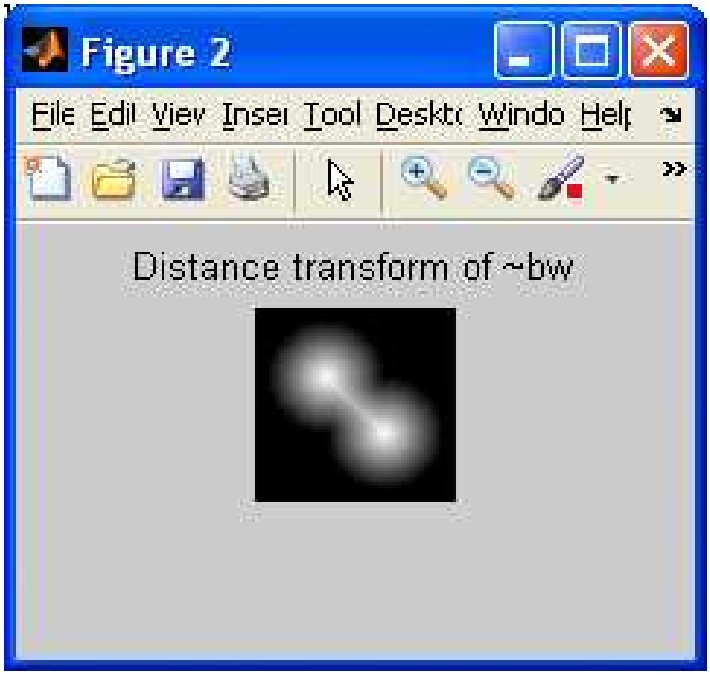
Εικόνα 16 : Quasi-Euclidean
Πηγή : Ιστοσελίδα Mathworks του Matlab

Για να κατανοήσουμε την λειτουργία της συνάρτησης θα δημιουργήσουμε με τον παρακάτω κώδικα μία εικόνα και στην συνέχεια θα καλέσουμε την bwdist για αυτήν την εικόνα και θα προβάλουμε στην οθόνη τον μετασχηματισμό.

```
center1 = -10;
center2 = -center1;
dist = sqrt(2*(2*center1)^2);
radius = dist/2 * 1.4;
lims = [floor(center1-1.2*radius) ceil(center2+1.2*radius)];
[x,y] = meshgrid(lims(1):lims(2));
bw1 = sqrt((x-center1).^2 + (y-center1).^2) <= radius;
bw2 = sqrt((x-center2).^2 + (y-center2).^2) <= radius;
bw = bw1 | bw2;
figure, imshow(bw), title('bw')
D = bwdist(~bw);
figure, imshow(D,[]), title('Distance transform of ~bw')
```



Εικόνα 17 : Δοκιμαστική τεχνητή εικόνα
Πηγή : Εικόνα συστήματος Matlab



Εικόνα 18 : Αποτέλεσμα μετασχηματισμού ευκλείδιας απόστασης
Πηγή : Εικόνα συστήματος Matlab

Στην συνέχεια με τον παρακάτω κώδικα δημιουργούμε μια «εικόνα» πίνακα για να κατανοήσουμε την διεργασία που εκτελείται.

```
bw = zeros(5,5);
bw(2,2) = 1;
bw(4,4) = 1

bw =
```

0	0	0	0	0
0	**1**	0	0	0
0	0	0	0	0
0	0	0	**1**	0
0	0	0	0	0

Πίνακας 4 : Τεχνητή "εικόνα" πίνακας

Υπολογίζουμε την distance transform.
[D,IDX] = bwdist(bw)

D =

1.4142	1.0000	1.4142	2.2361	3.1623
1.0000	**0**	1.0000	2.0000	2.2361
1.4142	1.0000	1.4142	1.0000	1.4142
2.2361	2.0000	1.0000	**0**	1.0000
3.1623	2.2361	1.4142	1.0000	1.4142

Πίνακας 5 : Υπολογισμός ευκλείδειας απόστασης

IDX =

7	7	7	7	7
7	7	7	7	19
7	7	7	19	19
7	7	19	19	19
7	19	19	19	19

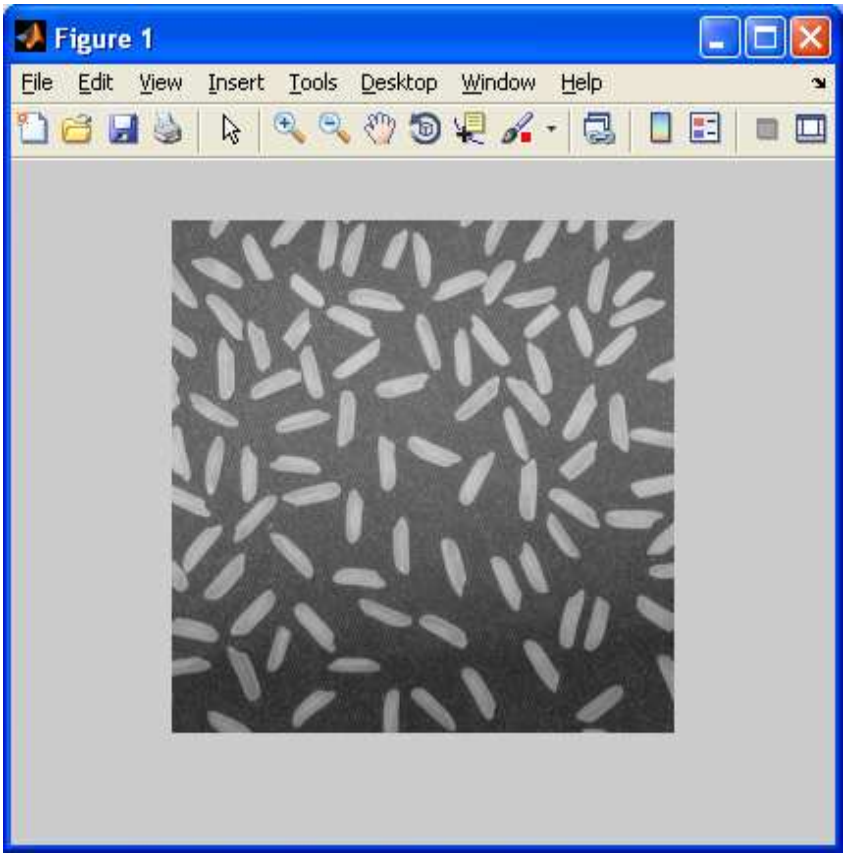
Πίνακας 6 : Κοντινότεροι γείτονες

Στον πίνακα IDX με τους κοντινότερους γείτονες οι τιμές 7 και 19 αντιπροσωπεύουν την θέση των μη μηδενικών στοιχείων χρησιμοποιώντας πίνακα γραμμικής τοποθέτησης. Εάν ένα pixel περιέχει τον αριθμό 7 τότε το κοντινότερο μη μηδενικό γειτονικό στοιχείο σε γραμμική τοποθέτηση είναι 7.

2.4.8 Συνάρτηση imadd

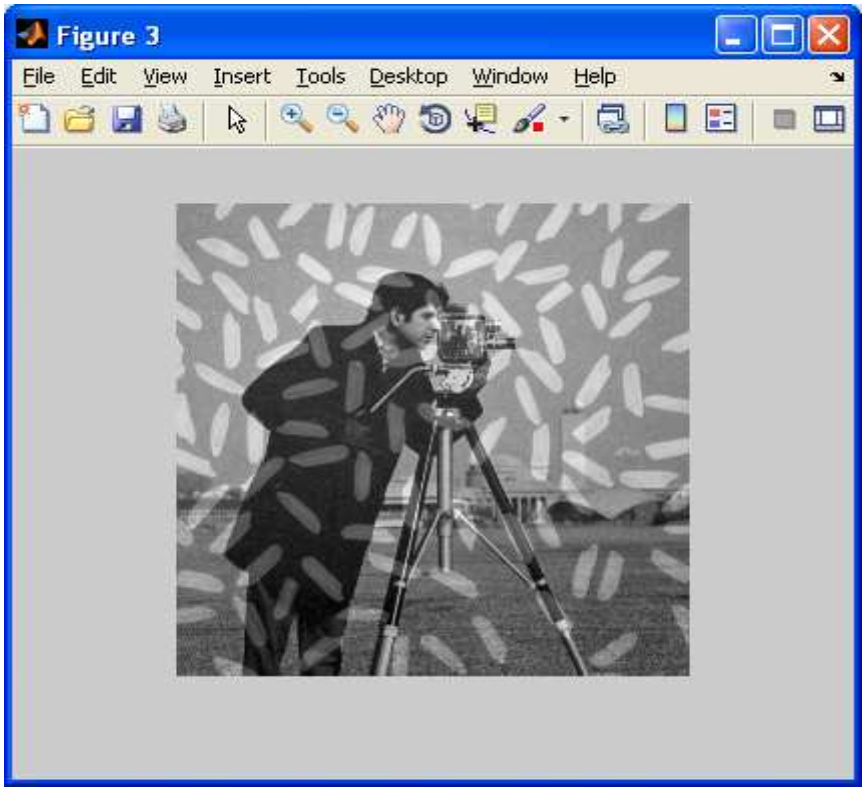
Η συνάρτηση imadd προσθέτει δύο εικόνες ή πίνακες μεταξύ τους , επίσης μπορούμε με αυτή την συνάρτηση να προσθέσουμε μια σταθερά σε μια εικόνα. Παρακάτω θα δούμε τον κώδικα για όλες τις περιπτώσεις με τα αποτελέσματα αυτού.

```
(Πρόσθεση δύο εικόνων.)
I = imread('rice.png');
J = imread('cameraman.tif');
K = imadd(I,J,'uint16');
figure(1);imshow(I);
figure(2);imshow(J);
figure(3);imshow(K,[]);
truesize;
```



Εικόνα 19 : Εικόνα συστήματος rice
Πηγή : Εικόνα συστήματος Matlab

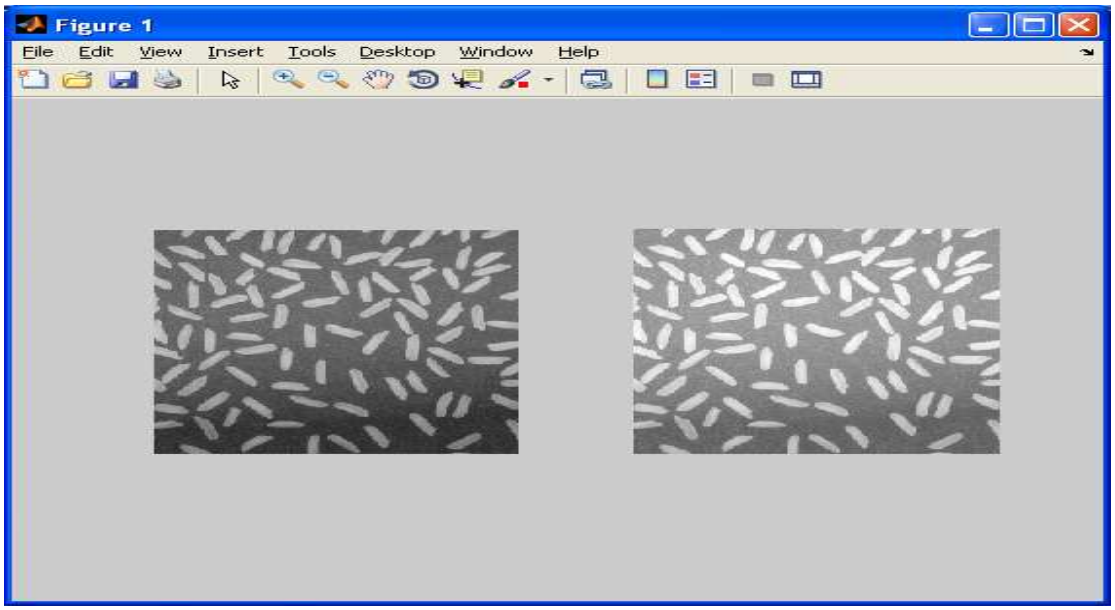
Εικόνα 20 : Εικόνα συστήματος cameraman
Πηγή : Εικόνα συστήματος Matlab



Εικόνα 21 : Αποτέλεσμα πρόσθεσης εικόνας rice με cameraman
Πηγή : Εικόνα συστήματος Matlab

(πρόσθεση μιας σταθεράς σε μια εικόνα)

```
I = imread('rice.png');
J = imadd(I,50);
subplot(1,2,1), imshow(I)
subplot(1,2,2), imshow(J)
```



Εικόνα 22 : Αριστερά βλέπουμε την αρχική εικόνα και δεξιά την ίδια εικόνα μετά την πρόσθεση σε αυτην της σταθεράς (50)
Πηγή : Εικόνα συστήματος Matlab

Προσθήκη δύο πινάκων unsigned ακεραίων 8 bits. Η μέγιστη τιμή που μπορούν να πάρουν είναι 255.

```
X = uint8([ 255 0 75; 44 225 100]);
Y = uint8([ 50 50 50; 50 50 50 ]);
Z = imadd(X,Y)
```

Z =

255	50	125
94	255	150

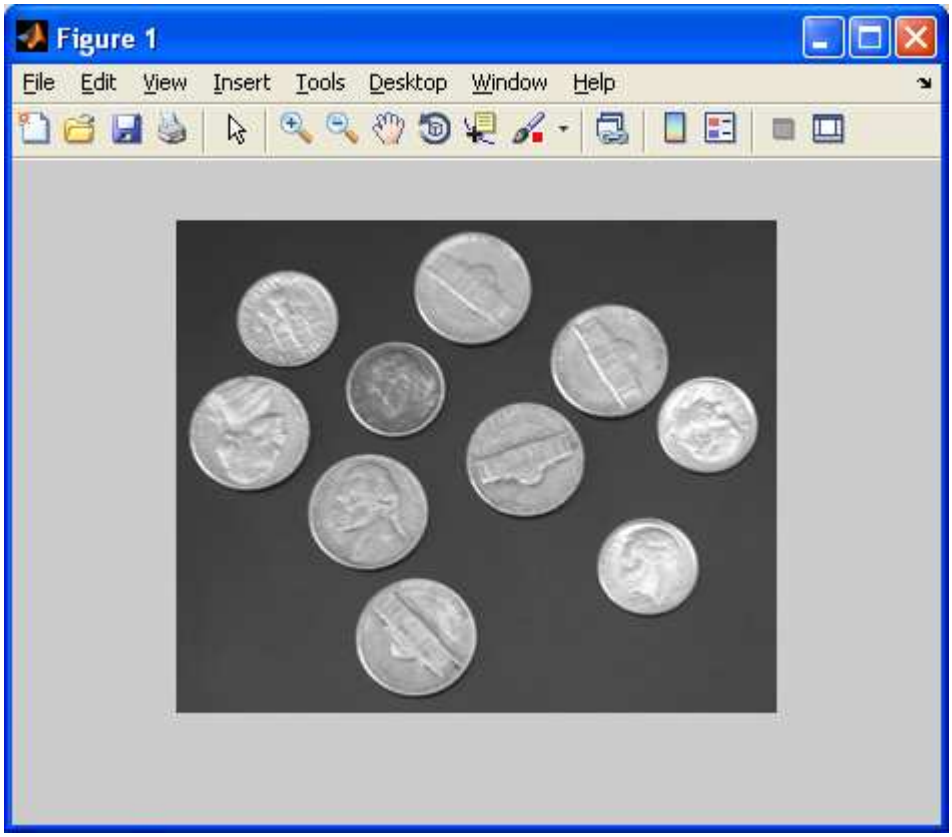
Πίνακας 7 : Αποτέλεσμα πρόσθεσης των πινάκων X,Y

2.4.9 Συνάρτηση graythresh σε συνδυασμό με την συνάρτηση im2bw

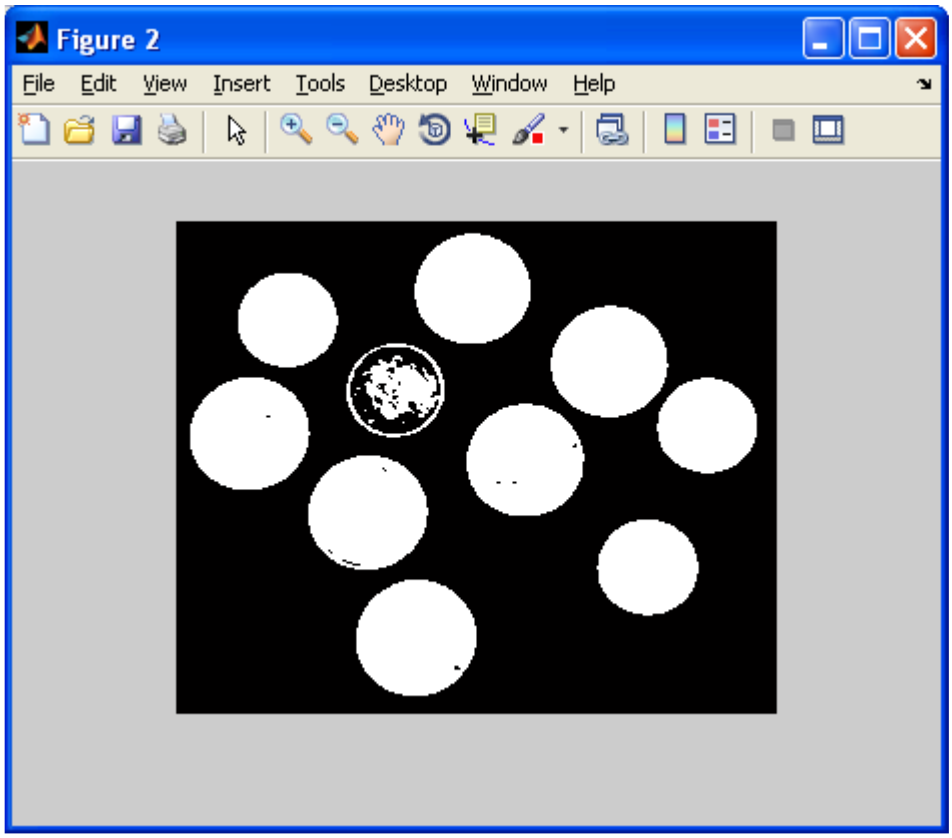
Η συνάρτηση graythresh υπολογίζει ένα γενικό κατώφλι (level) το οποίο μπορεί να χρησιμοποιηθεί για να μετατρέψει μια εικόνα σε binary με την εντολή im2bw. Το (level) είναι μια κανονικοποιημένη τιμή έντασης η οποία κυμαίνεται μεταξύ [0, 1]. Η συνάρτηση graythresh χρησιμοποιεί την μέθοδο Otsu για να επιλέξει το κατώφλι που ελαχιστοποιεί την διακύμανση των μαύρων και άσπρων pixels. Η μέθοδος του Otsu είναι μία από τις καλύτερες τεχνικές εύρεσης κατωφλίου. Το κριτήριο το οποίο χρησιμοποιείται στη μέθοδο του Otsu για τον προσδιορισμό του βέλτιστου κατωφλίου είναι η μεγιστοποίηση της διαχωριστικότητας μεταξύ των σκοτεινών και των φωτεινών περιοχών. Συντάσσεται level = graythresh(I).

Η συνάρτηση im2bw μετατρέπει μια grayscale εικόνα (I) σε binary.Η εικόνα BW που έχουμε ως έξοδο αντικαθιστά στην εικόνα που λάβαμε ως είσοδο κάθε pixel με φωτεινότητα μεγαλύτερη από (level) του δίνει τιμή (1) δηλαδή (άσπρο), ενώ στα υπόλοιπα pixel δίνει την τιμή (0) δηλαδή (μαύρο). Συντάσσεται BW = im2bw(I, level). Ο παρακάτω κώδικας δημιουργεί το γενικό κατώφλι και στην συνέχεια μετατρέπει την graylevel εικόνα σε binary.

```
I = imread('coins.png');
level = graythresh(I);
BW = im2bw(I,level);
figure(1);imshow(I);
figure(2);imshow(BW);
```



Εικόνα 23 : Αρχική graylevel εικόνα coins
Πηγή : Εικόνα συστήματος Matlab



Εικόνα 24 : Binary εικόνα coins
Πηγή : Εικόνα συστήματος Matlab

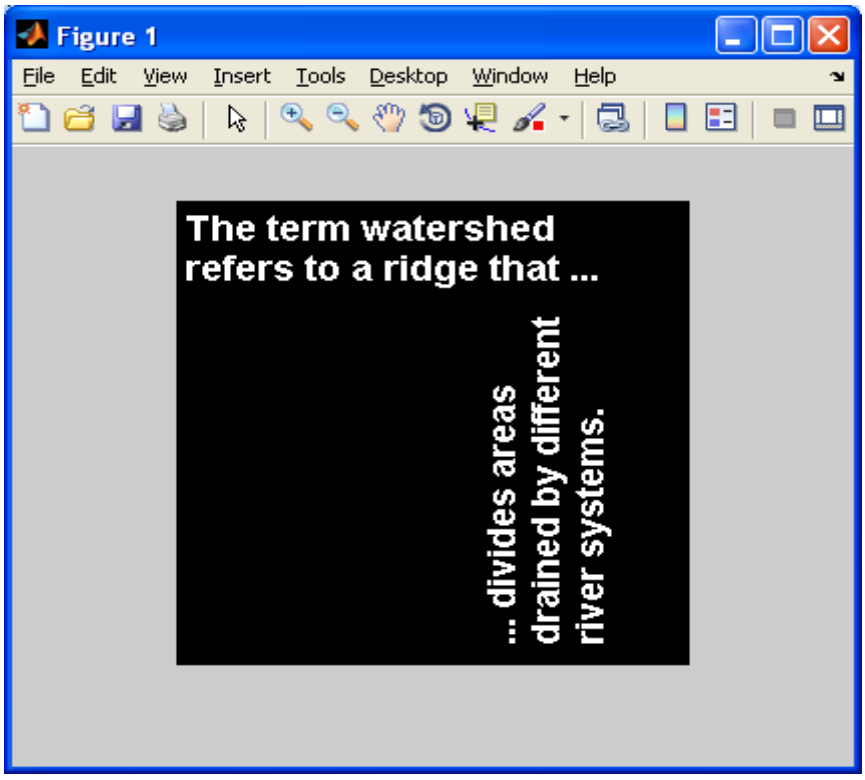
2.4.10 Συνάρτηση bwconncomp

Η συνάρτηση bwconncomp επιστρέφει τα συνδεδεμένα στοιχεία (CC) που βρέθηκαν στην binary εικόνα (BW). Το (CC) είναι μια δομή με 4 πεδία. Συντάσσεται CC = bwconncomp(BW)

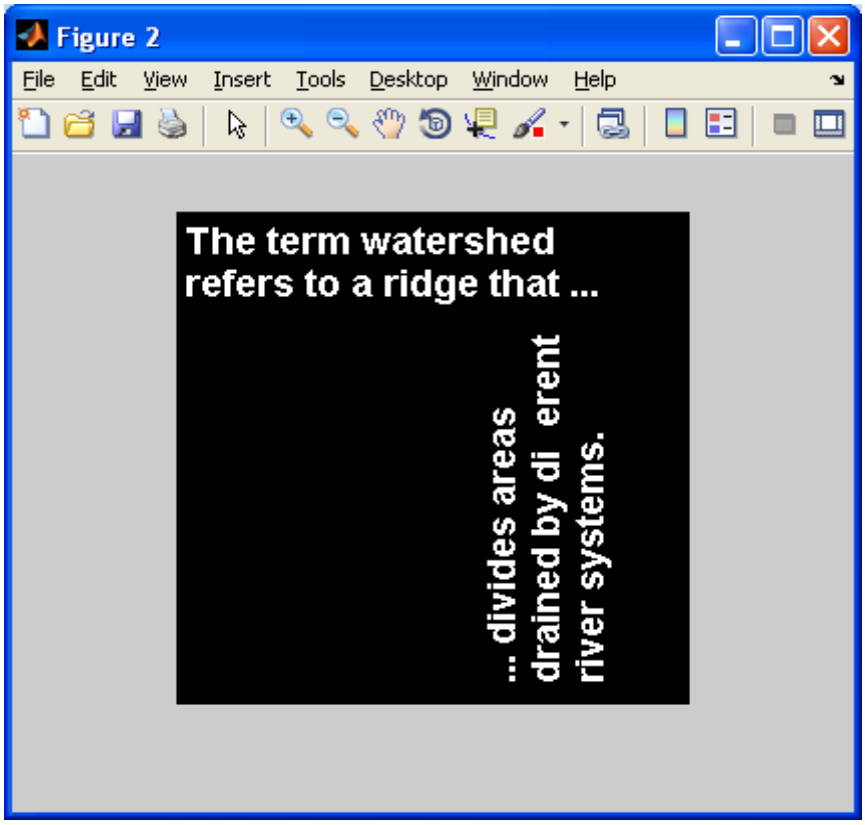
Πεδία	Περιγραφή
Connectivity	Συνδεσιμότητα των αντικειμένων
ImageSize	Μέγεθος της εικόνας
NumObjects	Αριθμός των συνδεδεμένων στοιχείων στην εικόνα BW
PixelIdxList	Πίνακας με μέγεθος όσο και αριθμός των συνδεδεμένων στοιχείων στην εικόνα BW όπου σε κάθε κελί υπ'αριθμόν (k) περιέχει ένα διάνυσμα που είναι γραμμικός δείκτης των pixels στο αντικείμενο υπ'αριθμόν (k)

Παράδειγμα όπου εντοπίζουμε και διαγράφουμε τα μεγαλύτερα γράμματα στην παρακάτω εικόνα.

```
BW = imread('text.png');
imshow(BW);
CC = bwconncomp(BW);
numPixels = cellfun(@numel,CC.PixelIdxList);
[biggest,idx] = max(numPixels);
BW(CC.PixelIdxList{idx}) = 0;
figure, imshow(BW);
```



Εικόνα 25 : Αρχική εικόνα με γράμματα
Πηγή : Εικόνα συστήματος Matlab



Εικόνα 26 : Μετά την αφαίρεση των μεγαλύτερων γραμμάτων
Πηγή : Εικόνα συστήματος Matlab

2.4.11 Συνάρτηση imextendedmin

Η συνάρτηση imextendedmin βρίσκει τα τοπικά ελάχιστα του μετασχηματισμού H-minima.Με τον παρακάτω κώδικα δημιουργούμε μία δοκιμαστική εικόνα με δύο τοπικά ελάχιστα.

```
a = 10*ones(10,10);
a(2:4,2:4) = 7;
a(6:8,6:8) = 2
```

a =

10	10	10	10	10	10	10	10	10	10
10	**7**	**7**	**7**	10	10	10	10	10	10
10	**7**	**7**	**7**	10	10	10	10	10	10
10	**7**	**7**	**7**	10	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10
10	10	10	10	10	**2**	**2**	**2**	10	10
10	10	10	10	10	**2**	**2**	**2**	10	10
10	10	10	10	10	**2**	**2**	**2**	10	10
10	10	10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10

Πίνακας 8 : Τεχνητή εικόνα με δύο τοπικά ελάχιστα

Βρίσκει τα ελάχιστα και τους προσθέτει μια τιμή (2).
b = imhmin(a,2)

b =

10	10	10	10	10	10	10	10	10	10
10	**9**	**9**	**9**	10	10	10	10	10	10
10	**9**	**9**	**9**	10	10	10	10	10	10
10	**9**	**9**	**9**	10	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10
10	10	10	10	10	**4**	**4**	**4**	10	10
10	10	10	10	10	**4**	**4**	**4**	10	10
10	10	10	10	10	**4**	**4**	**4**	10	10
10	10	10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10

Πίνακας 9 : Πρόσθεση μιας σταθεράς στα τοπικά ελάχιστα

Βάζουμε σαν είσοδο στην συνάρτηση imregionalmin την εικόνα b. Η συνάρτηση επιστρέφει μια δυαδική εικόνα στο ίδιο μέγεθος με την b και στην οποία κάθε pixel με τιμή (1) αντιπροσωπεύει τοπικό ελάχιστο ενώ τα υπόλοιπα pixel είναι (0).
B = imregionalmin(b)

B =

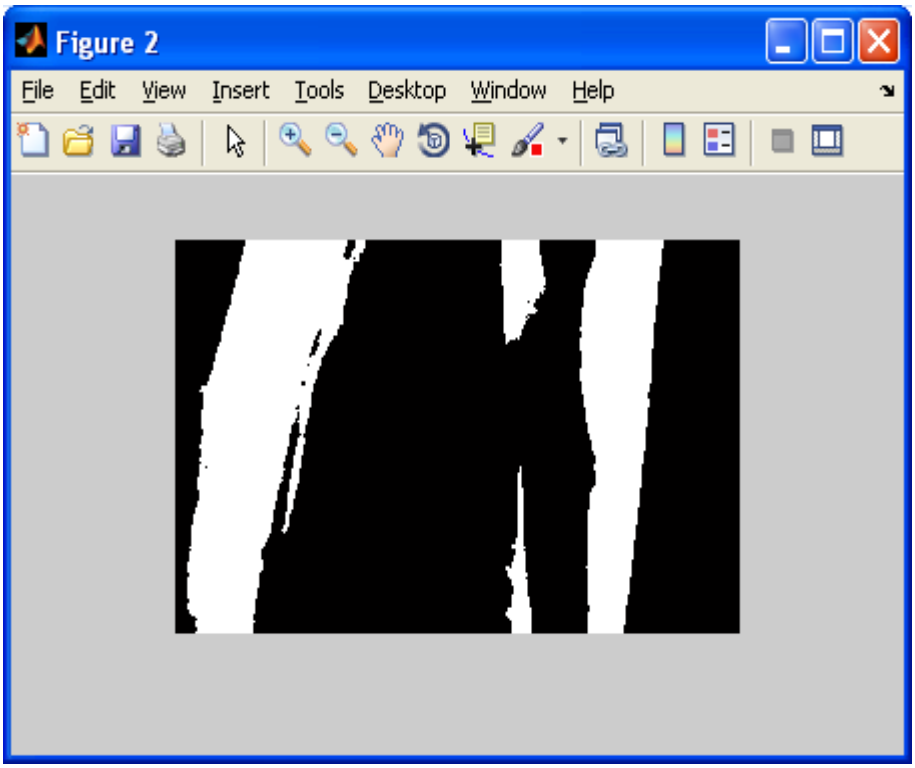
0	0	0	0	0	0	0	0	0	0
0	**1**	**1**	**1**	0	0	0	0	0	0
0	**1**	**1**	**1**	0	0	0	0	0	0
0	**1**	**1**	**1**	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	**1**	**1**	**1**	0	0
0	0	0	0	0	**1**	**1**	**1**	0	0
0	0	0	0	0	**1**	**1**	**1**	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Πίνακας 10 : Μετατροπή πίνακα 9 σε binary

Με τον παρακάτω κώδικα βρίσκουμε τα τοπικά ελάχιστα μιας εικόνας.

```
I = imread('glass.png');
BW = imextendedmin(I,50);
imshow(I), figure, imshow(BW)
```

Εικόνα 27 : Αρχική εικόνα
Πηγή : Εικόνα συστήματος Matlab



Εικόνα 28 : Τοπικά ελάχιστα
Πηγή : Εικόνα συστήματος Matlab

ΚΕΦΑΛΑΙΟ 3: Υλοποίηση εφαρμογής

Για τις ανάγκες της καταμέτρησης αντικειμένων από εικόνα δημιουργήθηκε μια εφαρμογή με το μαθηματικό λογισμικό πακέτο Matlab και για να επιτύχουμε ακριβής αποτελέσματα θα πρέπει ο κώδικας να εκτελεί τα παρακάτω βήματα.

1. Με την εκκίνηση της εφαρμογής ζητάει αρχικά από τον χρήστη να επιλέξει μια εικόνα για επεξεργασία.
2. Μετατρέπει την εικόνα από RGB σε graylevel
3. Περνάει την graylevel εικόνα από ένα φίλτρο μεσαίας τιμής
4. Στην συνέχεια χρησιμοποιώντας το εργαλείο κατωφλίωσης thresh_tool εμφανίζει στην οθόνη του χρήστη την εικόνα που εφαρμόσαμε το median filter κατωφλιωμένη και ζητάει από τον χρήστη χρησιμοποιώντας ένα slice bar με το ποντίκι του να επιλέξει το επιθυμητό επίπεδο κατωφλίωσης. Τέλος όταν ο χρήστης πατήσει το πλήκτρο Done η κατωφλιωμένη εικόνα αποθηκεύεται και περνάμε στο επόμενο βήμα.
5. Σε αυτό το βήμα έχουμε πάλι προτροπή του χρήστη με παράθυρο διαλόγου να κάνει διπλό κλικ με το mouse στην κατωφλιωμένη εικόνα που εμφανίστηκε εκ νέου και να επιλέξει ένα αντικείμενο μέσου μεγέθους το οποίο δεν εφάπτεται με τα υπόλοιπα.
6. Το αντικείμενο το οποίο επιλέξαμε στο βήμα (5) θα αποτελέσει τη βάση της μέτρησης. Ανάλογα με την επιφάνεια που καλύπτουν τα pixel του θα ορίσουμε τι είναι θόρυβος και ποιο αντικείμενο χρήζει διαχωρισμού.
7. Γίνεται αφαίρεση του θορύβου και των μικρών αντικειμένων που δεν ανήκουν στο εύρος μεγέθους που θέλουμε να μετρήσουμε.
8. Δημιουργούνται δύο εικόνες, η μία περιέχει τα μεγάλα αντικείμενα που πρέπει να διαχωριστούν έτσι ώστε να μπορούν να καταμετρηθούν και η άλλη τα αντικείμενα που ανταποκρίνονται στο μέσο μέγεθος του αντικειμένου που επιλέξαμε στο βήμα (5).
9. Στην εικόνα που δεν χρειάζεται περεταίρω διαχωρισμός γίνεται καταμέτρηση των αντικειμένων με βάση τα περιγράμματα αυτών και αποθηκεύουμε τον αριθμό τους για να τον χρησιμοποιήσουμε στην τελική καταμέτρηση.
10. Στην εικόνα με τα μεγάλα αντικείμενα που χρήζουν διαχωρισμού εκτελούνται οι κατάλληλες συναρτήσεις και έχουμε ως έξοδο μια νέα εικόνα με τα διαχωρισμένα αντικείμενα και των αριθμώ αυτών ο οποίος αποθηκεύεται.
11. Στο σημείο αυτό εμφανίζεται μια εικόνα στην οθόνη η οποία περιέχει τα διαχωρισμένα αντικείμενα πάνω στην μάσκα της εικόνας με τα μεγάλα αντικείμενα πριν διαχωριστούν έτσι ώστε να είναι ορατή στο χρήστη η διαδικασία. Στην συνέχεια ζητάει από τον χρήστη να επαλήθευση και να κάνει μικροδιορθώσεις με το mouse στην εικόνα. Όταν ο χρήστης πατήσει αριστερό-κλικ πάνω σε ένα αντικείμενο αυτό προστίθεται στο γενικό σύνολο, ενώ με το δεξί-κλικ αφαιρείται.
12. Τέλος παρουσιάζεται η εικόνα που περιέχει όλα τα αντικείμενα αριθμημένα και αναφέρει το γενικό σύνολο.

3.1 Επεξήγηση κώδικα

Θα αναφέρουμε λεπτομερώς πως λειτουργεί κάθε κομμάτι του κώδικα.

3.1.1 Μέρος πρώτο

Στο πρώτο μέρος του κώδικα γίνεται αρχικά αφαίρεση από την επιφάνεια εργασίας όλων των αντικειμένων , διαγραφεί τυχόν γραφημάτων , καθαρισμός ιστορικού και μεταβλητών έτσι ώστε να μπορούμε κατά την εκκίνηση του προγράμματος να κάνουμε έλεγχο για τυχόν σφάλματα. Δημιουργούμε ένα User Interface για την επιλογή του αρχείου εικόνας από τον χρήστη , και μια συνθήκη στην περίπτωση που δεν επιλέξει αρχείο ο χρήστης να τερματίζεται το πρόγραμμα. Στην συνέχεια η RGB εικόνα που επέλεξε ο χρήστης μετατρέπεται σε gray level και τέλος εφαρμόζεται σε αυτήν ένα φίλτρο μεσαίας τιμής (median filter) για την εξομάλυνση των ακμών και την αφαίρεση του θορύβου.

```
% Αφαιρει τα αντικειμενα απο την επιφανεια εργασιας και κανει
Reset την
% μηχανη MuPAD
clear all;
% διαγραφει τα τρεχον γραφηματα
close all;
% Καθαριζει το παραθυρο command window  απο ολες τις εισοδους
και εξοδους
% που εμφανιζει
clc;
% Εμφανιζει παραθυρο για να επιλεξουμε εικονα
[filename, pathname] = uigetfile('*.jpg', 'Pick an image to
process');
% Αν δεν επιλεξουμε εικονα τερματιζει το προγραμμα
if isequal(filename, 0)
  break;
end
% Διαβαζει την εικονα που επιλεξαμε
I=imread(filename);
% Μετατρεπει την εικονα απο RGB σε gray-level
Gri = rgb2gray(I);
% Περναει την εικονα απο median filter
MeF = medfilt2(Gri);
```

3.1.2 Μέρος δεύτερο

Στο δεύτερο μέρος χρησιμοποιούμε την συνάρτηση thresh_tool η οποία είναι ένα έτοιμο εργαλείο κατωφλίωσης εικόνας με user interface που δίνει την δυνατότητα στον χρήστη να αλλάζει το επίπεδο κατωφλίωσης (threshold level) με την χρήση του mouse και να βλέπει αμέσως στην οθόνη το αποτέλεσμα έτσι ώστε να επιλέξει το βέλτιστο επίπεδο κατωφλίωσης. Την εικόνα που επιστρέφει το thresh_tool την αποθηκεύουμε σε μορφοποίηση PNG γιατί σε αυτό το format εικόνας λειτουργούν οι συναρτήσεις που θα χρησιμοποιήσουμε παρακάτω. Στην συνέχεια εμφανίζουμε στον χρήστη ένα παράθυρο με οδηγίες για τις ενέργειες που πρέπει να εκτελέσει και ορίζουμε και συνθήκη στην περίπτωση που ο χρήστης δεν θέλει να συνεχίσει την διαδικασία και πατήσει το πλήκτρο “cancel” έτσι ώστε να μην συνεχίσει η εκτέλεση των εντολών. Αν ο χρήστης αποφασίσει ότι θέλει να συνεχίσει και πιέσει το πλήκτρο “ok” τότε θα εμφανιστεί στην οθόνη του η κατωφλιωμένη εικόνα που δημιούργησε το thresh_tool και θα του ζητηθεί χρησιμοποιώντας τον κέρσορα του mouse να επιλέξει ένα αντικείμενο μέσου μεγέθους που δεν εφάπτεται με άλλα αντικείμενα και να πιέσει πάνω σε αυτό διπλό κλικ.

```
% Κανει threshold στην εικονα χρησιμοποιωντας συναρτηση
[level,Bin]= thresh_tool(MeF) ;
%αποθηκεύει την τελευταία εικόνα στο directory μας σε μορφή
png
imwrite(Bin,'TestBW.png','png');
% Διαβαζει την εικονα που αποθηκευσαμε
Test = imread('TestBW.png');
% Εμφανιζει την κατοφλιωμενη εικονα και ζηταει απο τον χρηστη
να
% επιλεξει με double-click ενα αντικειμενο
message = sprintf('Choose a middle size sheep.\nby pressing
double-click.');
button = questdlg(message, 'Continue', 'OK', 'Cancel','OK');
% Κανει refresh στην οθονη
drawnow;
% Ελεγχει αν πατησαμε 'cancel' και τοτε τερματιζει το
προγραμμα
if strcmpi(button, 'Cancel')
    return;
end
% Ο χρηστης επιλεγει με double-click  ενα αντικειμενο.
poi = bwselect(Test,8);
```

3.1.3 Μέρος τρίτο

Στο τρίτο μέρος αφού ο χρήστης έχει επιλέξει το επιθυμητό αντικείμενο , αυτό απομονώνεται και μετράται η επιφάνεια του σε pixel. Το αποτέλεσμα της μέτρησης μετατρέπεται σε unsigned integer μεγέθους 16 bits. Για να ορίσουμε πότε ένα αντικείμενο είναι μεγάλο και χρήζει διαχωρισμού θέτουμε το όριο big το οποίο είναι το 120% του αντικειμένου που επιλέξαμε και το όριο small το οποίο είναι το 20% του αντικειμένου που επιλέξαμε. Για να καταλάβουμε καλύτερα θα δώσουμε ένα παράδειγμα : Αν το αντικείμενο που επιλέξαμε ως βάση έχει μέγεθος επιφάνειας 100 pixel τότε ένα αντικείμενο μεγέθους επιφάνειας 120 pixel θεωρείται μεγάλο και ένα αντικείμενο μεγέθους επιφάνειας 20 pixel θεωρείται μικρό. Τέλος τα (big,small) μετατρέπονται σε double έτσι ώστε να μπορούν να μπουν σαν ορίσματα σε συναρτήσεις.

```
% Μετρηση της επιφανειας του αντικειμενου σε pixels.
total = bwarea(poi);
% Μετατροπη της τιμης total σε unsigned integer 16 bit
total2=uint16(total);
% Ορισμος ενος αντικειμενου ως μεγαλο οταν εχει μεγεθος
μεγαλυτερο απο
% το 120% του αντικειμενου που επιλεξαμε
big=total2*1.2;
% Ορισμος ενος αντικειμενου ως μικρο οταν εχει μεγεθος
μικροτερο απο
% το 20% του αντικειμενου που επιλεξαμε
small=total2*0.2;
% Μετατροπη των τιμων (big),(small) σε double για να
μπορεσουμε να τις
% βαλουμε σαν ορισματα σε συναρτησεις
small=double(small);
big=double(big);
```

3.1.4 Μέρος τέταρτο

Στο τέταρτο μέρος δημιουργούνται αντίγραφα της κατωφλιωμένης εικόνας , και αφαιρούνται από την πρώτη εικόνα όλα τα αντικείμενα με μέγεθος μικρότερο από small και όλα τα αντικείμενα με μέγεθος μεγαλύτερο ή ίσο με big έτσι ώστε να μείνουν τα αντικείμενα που μας ενδιαφέρουν να καταμετρήσουμε και δεν χρειάζονται διαχωρισμό. Στην δεύτερη εικόνα αφαιρούμε όλα τα αντικείμενα που έχουν μέγεθος μικρότερο από big έτσι ώστε να μείνουν μόνο τα μεγάλα αντικείμενα που χρήζουν διαχωρισμού.

```
Test2=Test;
Test4=Test;
% Επιλεγουμε να αφαιρεσουμε απο την εικονα τα αντικειμενα με
μεγεθος
% μεγαλυτερο η ισο απο το (big).
CC = bwconncomp(Test2);
% Εκτελει την συναρτηση numel η οποια μας δινει τον αριθμο των
στοιχειων
% σε ενα πινακα, σε καθε κελι του πινακα και επιστρεφει των
% αριθμο των pixels.
numPixels = cellfun(@numel,CC.PixelIdxList);
% Επαναληψη απο i εως τον αριθμο των αντικειμενων
for i=1:CC.NumObjects
% Εαν το αντικειμενο εχει μεγεθος σε pixel μεγαλυτερο η ισο
απο το big τοτε
% μηδενιζουμε τα pixel του αντικειμενου αυτου στην εικονα
Test2
if(numPixels(1,i)>=big)
idx= i;
Test2(CC.PixelIdxList{idx}) = 0;
end
end
% Επιλεγουμε να αφαιρεσουμε απο την εικονα τα αντικειμενα με
μεγεθος
% μικροτερο απο το (small).Συνηθως θορυβος.
Test3 = bwareaopen(Test2, small);
% Επιλεγουμε να αφαιρεσουμε απο την εικονα τα αντικειμενα με
μεγεθος
% μικροτερο απο το (big).
Test4 = bwareaopen(Test4, big);
```

3.1.5 Μέρος πέμπτο

Στο πέμπτο μέρος του κώδικα χρησιμοποιούμε μια συνάρτηση για να μας επιστρέψει τα εξωτερικά μόνο περιγράμματα των αντικειμένων και των δύο εικόνων που δημιουργήσαμε στο τέταρτο μέρος. Η εικόνα με τα μεγάλα αντικείμενα ελέγχεται για το αν περιέχει αντικείμενα για διαχωρισμό ή είναι κενή και στην συνέχεια αν περιέχει αντικείμενα εφαρμόζουμε σε αυτήν την συνάρτηση bwdist που μας δημιουργεί μια νέα εικόνα με την distance transform χρησιμοποιώντας την ευκλείδεια μέθοδο. Στην εικόνα που δημιουργήθηκε εφαρμόζουμε την συνάρτηση imextendedmin για να βρούμε τα τοπικά ελάχιστα και μας επιστρέφει μια νέα εικόνα με την μάσκα των εσωτερικών αντικειμένων στην οποία μετράμε εκ νέου τα περιγράμματα. Τέλος προσθέτουμε την εικόνα που δημιουργήσαμε με την εικόνα των αντικειμένων που δεν χρειάζονται καταμέτρηση έτσι ώστε να έχουμε σε μία εικόνα το σύνολο των αντικειμένων.

```
% Εντοπιζει τα περιγραμματα των αντικειμενων
B  = bwboundaries(Test4,'noholes');
B2 = bwboundaries(Test3, 'noholes');
% Ελεγχει αν υπαρχουν μεγαλα αντικειμενα για διαχωρισμο
if(((length(B))>0))
% Δημιουργει την Distance transform
D = -bwdist(~Test4);
% Δημιουργει την μασκα των εσωτερικων αντικειμενων
mask = imextendedmin(D,3);
else
    mask=Test4;
end
% Εντοπιζει τα περιγραμματα των αντικειμενων
B3 = bwboundaries(mask, 'noholes');
% Δημιουργει μια νεα εικονα με την προσθεση δυο αλλων εικονων.
Z = imadd(Test3,mask);
% Μετατρεπει την νεα εικονα σε δυαδικη.
level2 = graythresh(Z);
Bin2 = im2bw(Z,level2);
```

3.1.6 Μέρος έκτο

Το έκτο μέρος είναι το τελευταίο μέρος του κυρίως κώδικα και ξεκινάει με τον έλεγχο της συνθήκης για το αν χρειάστηκε να γίνει διαχωρισμός των αντικειμένων. Αν είναι αληθής εμφανίζει την εικόνα με τα διαχωρισμένα αντικείμενα πάνω στην μάσκα των μεγάλων αντικειμένων όπως ήταν αρχικά ώστε να είναι ευδιάκριτη η διαδικασία του διαχωρισμού. Στην συνέχεια πάνω σε αυτή την εικόνα ζητάει από τον χρήστη να αφαιρέσει σημεία που έχουν τοποθετηθεί λανθασμένα και να προσθέσει σημεία που δεν έχουν επισημανθεί. Η διαδικασία της προσθαφαίρεσης σημείων γίνεται με την χρήση του mouse , το αριστερό κλικ προσθέτει ένα σημείο στο γενικό σύνολο και τοποθετεί έναν πράσινο σταυρό στο σημείο που επιλέξαμε ενώ το δεξί κλικ αφαιρεί ένα σημείο από το γενικό σύνολο και τοποθετεί ένα κόκκινο (X).

```
if(((length(B))>0))
% Εμφανιζει δυο εικονες σε μορφη blend
figure('Name','Confirm
seperation');imshowpair(Test4,mask,'blend');
%  Διατηρει το γραφημα ετσι ωστε να προστεθουν αλλα γραφηματα
πανω σε αυτο
hold on;
% Δινει την αρχικη τιμη στον μετρητη μας count.
count = 0;
% Δινει την αρχικη τιμη στις μεταβλητες plus,minus.
plus=0;
minus=0;
% Εμφανιζει εναν τιτλο στην εικονα που καθοδηγει τον χρηστη.
title('Press left clik to add a sheep and right click to
remove a false point.To complete the process press enter.',
'FontSize', 10);
% Ξεκιναει το loop οπου ο χρηστης κανει click πανω στην
εικονα.
while count < 1000
    % Εαν ο χρηστης πατησει enter τοτε το (x) επιστρεφει κενο.
    [x,y,button] = ginput(1);
    if isempty(x) %#ok<ALIGN>
```

```
        break;
    % Ελεγχει αν πατηθηκε το αριστερο κουμπι του mouse
    elseif button==1
    % Βαζει ενα πρασινο σταυρο στο σημειο που κανουμε click.
    plot(x, y, 'g+', 'MarkerSize', 12, 'LineWidth', 2);
    % Αυξανει τον μετρητη count και την μεταβλητη plus
    count = count + 1;
    plus=plus+1;
    % Ελεγχει αν πατηθηκε το δεξι κουμπι του mouse
    elseif button==3
    % Βαζει ενα κοκκινο (X) στο σημειο που κανουμε click.
    plot(x, y, 'rx', 'MarkerSize', 12, 'LineWidth', 2);
    % Αυξανει τον μετρητη count και την μεταβλητη minus
    count = count + 1;
    minus=minus+1;
    end

end
end
```

3.1.7 Γραφήματα

Δημιουργούμε ένα γράφημα μέσα στο οποίο θα εμφανιστούν σε διάταξη τριών γραμμών και τριών στηλών (3x3) , εννέα υπογραφήματα με τις εικόνες που δημιουργούνται από την επεξεργασία της αρχικής εικόνας .

```
%FIGURE-----------------------------------------------------------
-----------
% Δημιουργια γραφηματος με ονομα 'image process' πανω στο
οποιο θα
% εμφανιστουν οι παρακατω εικονες σαν υπογραφηματα
figure('Name','Image process ');
% Αρχικη εικονα
subplot(3,3,1);imshow(I);
title('Original image');
% Εικονα που δημιουργηθηκε απο την μετατροπη της αρχικης RGB
σε gray-level
subplot(3,3,2);imshow(Gri);
title('RGB to Gray-level');
% Εικονα gray-level που εχει περασει απο φιλτρο μεσαιας τιμης
subplot(3,3,3);imshow(MeF);
title('Median filter');
% Κατωφλιωμενη εικονα
subplot(3,3,4);imshow(Bin);
title('Thresholded');
% Εικονα που εμφανιζει ολα τα μικρα αντικειμενα συμφωνα με τα
οριο που
% εχουμε ορισει
subplot(3,3,5);imshow(Test2);
title('Small objects with noise');
% εικονα που εμφανιζει ολα τα μικρα αντικειμενα μετα απο την
αφαιρεση του
% θορυβου
subplot(3,3,6);imshow(Test3);
```

```
% Μετρωντας τα περιγραμματα των αντικειμενων δινει τιλο στην
εικονα ποσα
% μικρα αντικειμενα εχουμε
title(strcat('\color{red}Small objects
:',(num2str(length(B2)))));
% Εικονα που εμφανιζει ολα τα μεγαλα αντικειμενα συμφωνα με τα
ορια που
% εχουμε ορισει
subplot(3,3,7);imshow(Test4);
% Μετρωντας τα περιγραμματα των αντικειμενων δινει τιλο στην
εικονα ποσα
% μεγαλα αντικειμενα εχουμε
title(strcat('\color{blue}Big objects
:',(num2str(length(B)))));
% Ελεγχει αν υπαρχουν μεγαλα αντικειμενα για διαχωρισμο
if(((length(B))>0))
% Εικονα που εμφανιζει ολα τα μεγαλα αντικειμενα μετα τον
διαχωρισμο
subplot(3,3,8);imshow(mask);
% Μετρωντας τα περιγραμματα των αντικειμενων δινει τιλο στην
εικονα ποσα
% μεγαλα αντικειμενα εχουμε μετα τον διαχωρισμο
title(strcat('\color{green}Big objects separated
:',(num2str(length(B3)))));
end
% Εικονα που εμφανιζει το συνολο των αντικειμενων αριθμημενα
subplot(3,3,9);imshow(Bin2);
% Μετρωντας τα περιγραμματα των αντικειμενων σε δυο εικονες
% δινει τιλο στην εικονα ποσα ειναι συνολικα τα αντικειμενα
title(strcat('\color{red}Total objects
:',(num2str(length(B2)+length(B3)))));
% Δημιουργει νεο γραφημα με ονομα 'Results'
figure('Name', 'Results');
s = regionprops(Bin2, 'Centroid');
% Εμφανιζει την εικονα με το συνολο των αντικειμενων για
καταμετρηση
imshow(Bin2);
%  Διατηρει το γραφημα ετσι ωστε να προστεθουν αλλα γραφηματα
πανω σε αυτο
hold on
% Ελεγχει αν υπαρχει η μεταβλητη 'count' για να μαθουμε αν ο
χρηστης
% χρησιμοποιησε την επαληθευση καταμετρησης
if (exist('count','var'))
% Μετρωντας τα περιγραμματα των αντικειμενων σε δυο εικονες,
προσθετοντας
% το plus και αφαιροντας το minus δινει τιλο στην εικονα ποσα
ειναι
% συνολικα τα αντικειμενα μετα την επαληθευση του χρηστη
title(strcat('\color{red}Total objects
:',(num2str(length(B2)+length(B3)+plus-minus))));
else
% Μετρωντας τα περιγραμματα των αντικειμενων σε δυο εικονες
% δινει τιλο στην εικονα ποσα ειναι συνολικα τα αντικειμενα
title(strcat('\color{red}Total objects
:',(num2str(length(B2)+length(B3)))));
```

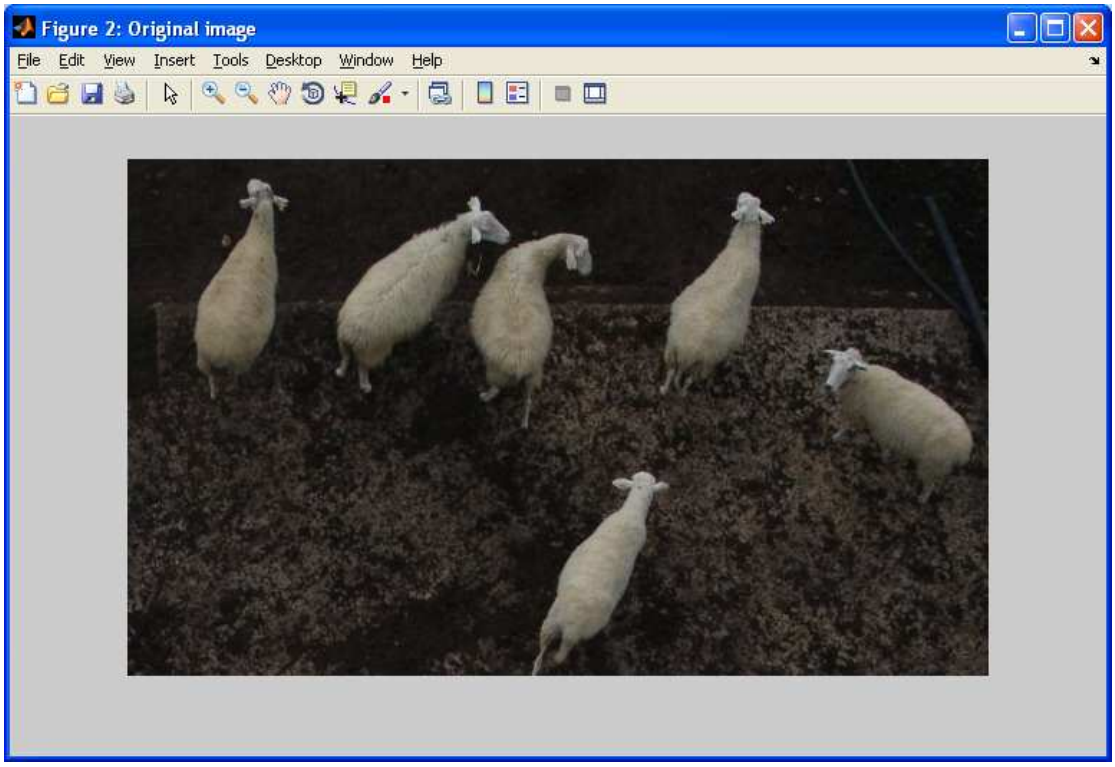
```
end
% Βρογχος επαναληψης οπου σε καθε αντικειμενο δινεται ενας
αριθμος
for k = 1:numel(s)
    c = s(k).Centroid;
    text(c(1), c(2), sprintf('%d', k), ...
        'HorizontalAlignment', 'center', ...
        'VerticalAlignment', 'middle',...
        'color','magenta',...
        'FontWeight', 'bold');

end
% Επαναφερει το hold στην default καταταση,καθαριζει το
τρεχον γραφημα και
% μηδενιζει τα axes
hold off
```

ΚΕΦΑΛΑΙΟ 4: Αποτελέσματα εφαρμογής

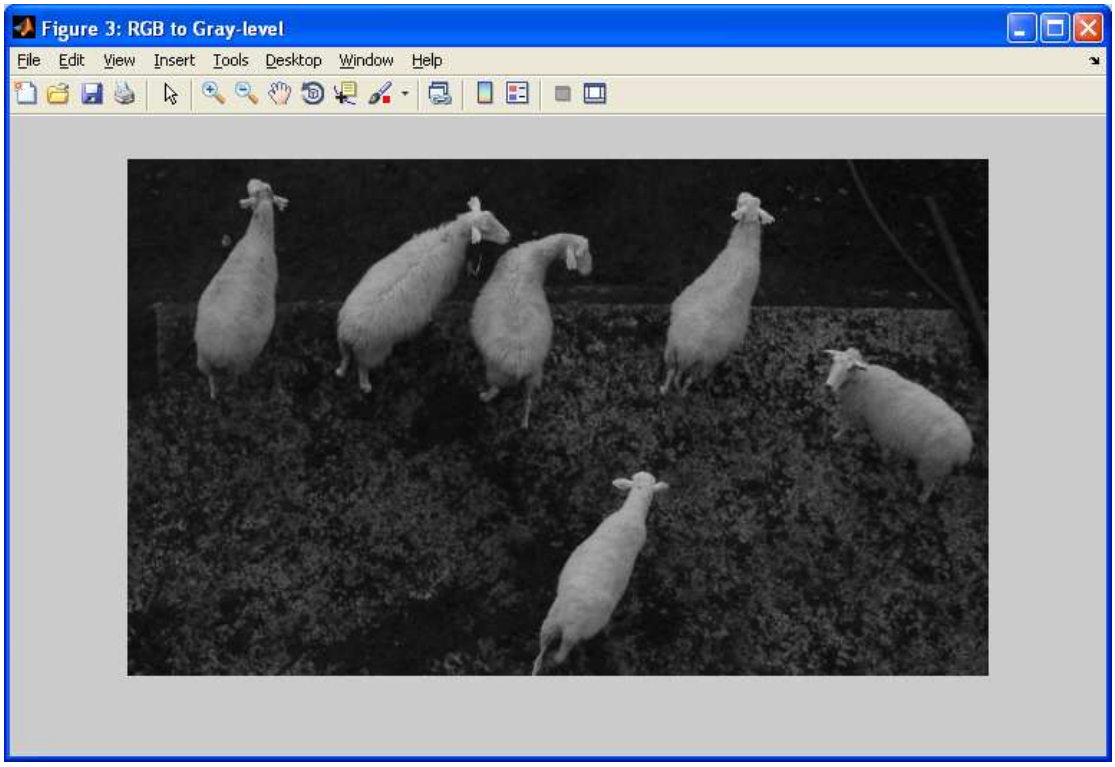
Η δοκιμή της εφαρμογής έγινε με μία σειρά εικόνων που περιέχουν πρόβατα , θα εξετάσουμε πέντε διαφορετικές φωτογραφίες και θα κάνουμε καταμέτρηση των προβάτων που περιέχονται σε αυτές .Θα αναφερόμαστε σε κάθε εικόνα με τον αριθμό των προβάτων που περιέχει και είναι οι εξής :

- 6 πρόβατα
- 14 πρόβατα
- 29 πρόβατα
- 41 πρόβατα
- 60 πρόβατα

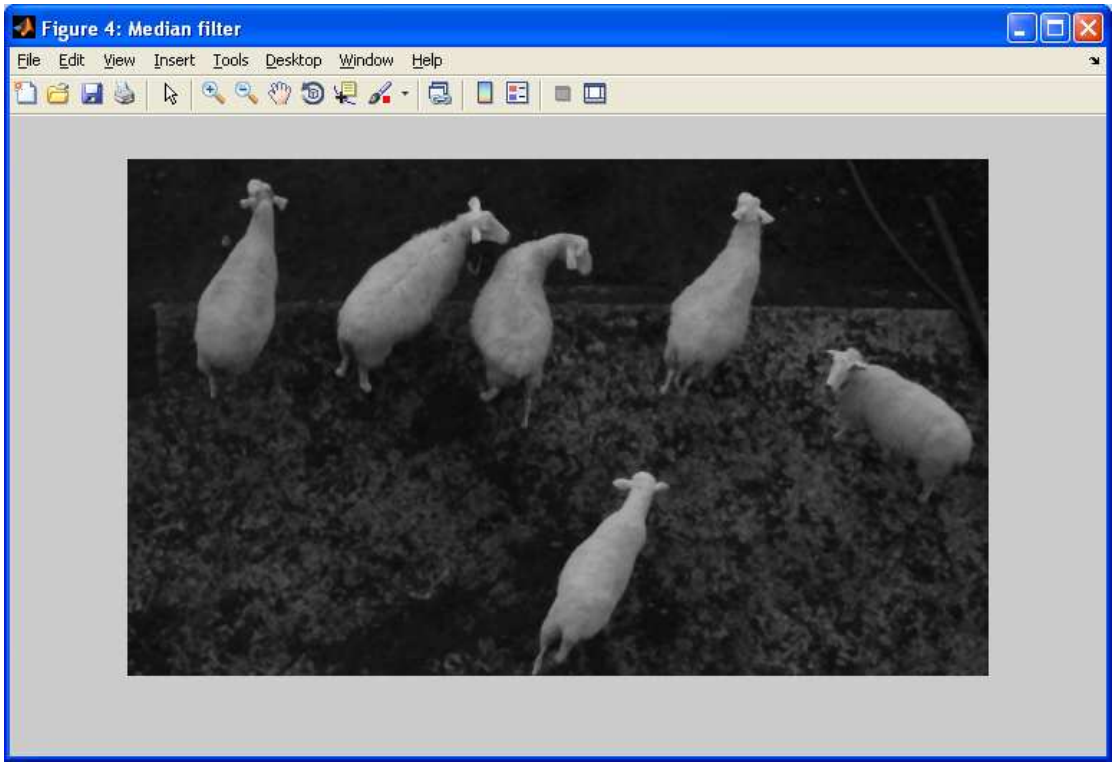
4.1 Δοκιμή με 6 πρόβατα



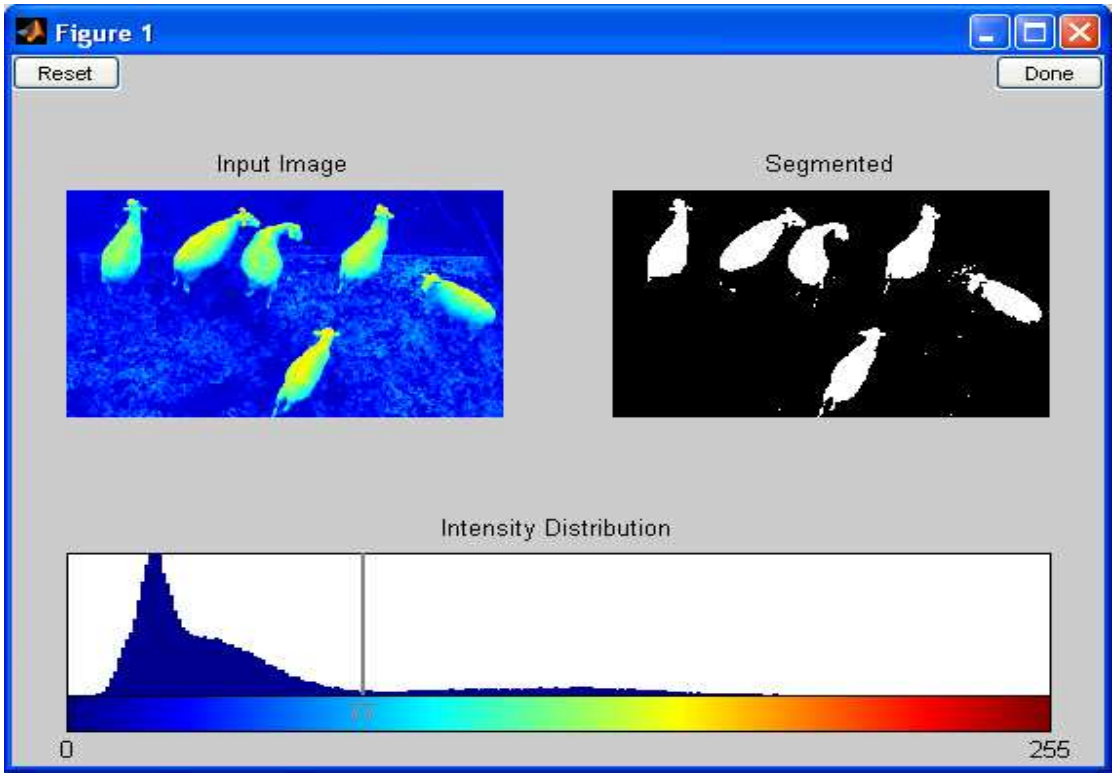
Εικόνα 29: Αρχική εικόνα με 6 πρόβατα



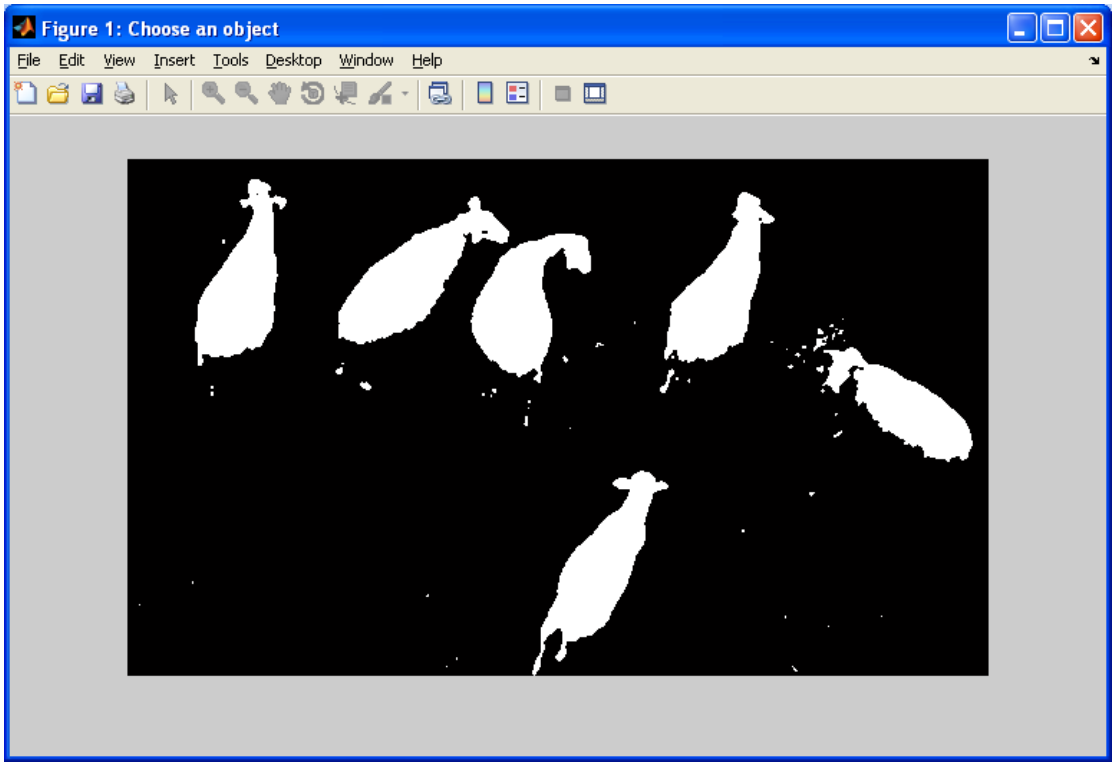
Εικόνα 30: Μετατροπή αρχικής έγχρωμης εικόνας σε ασπρόμαυρη



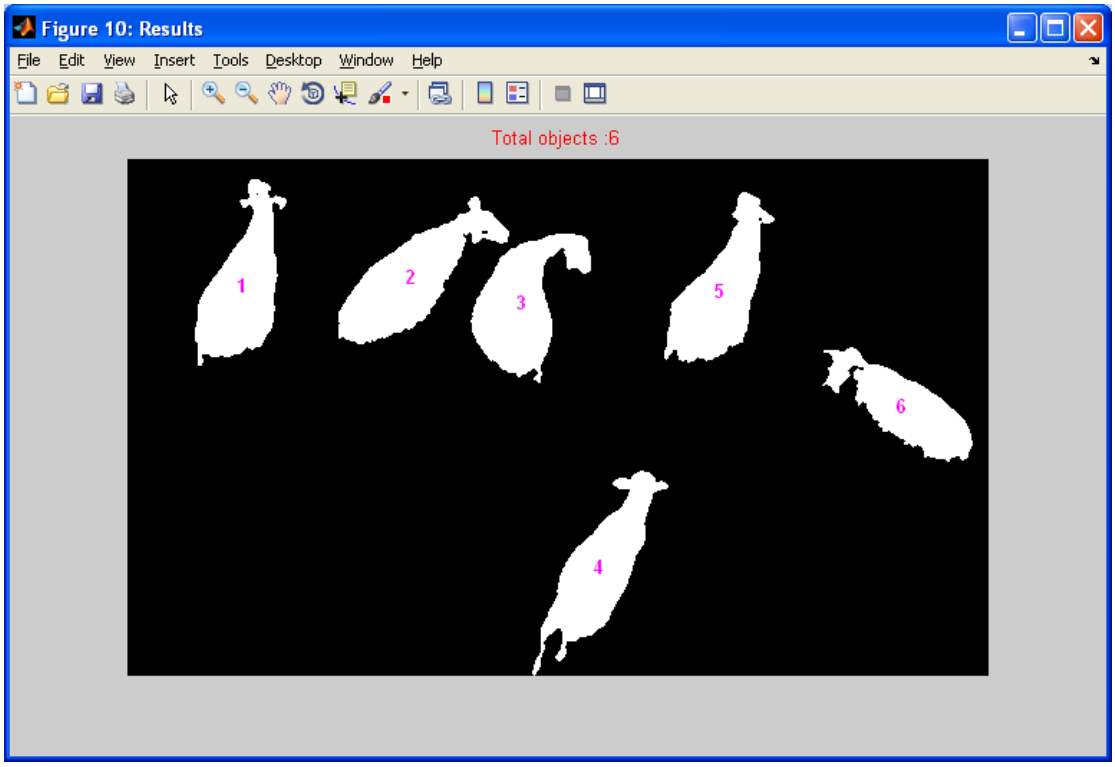
Εικόνα 31: Ασπρόμαυρη εικόνα που έχει περάσει από median filter



Εικόνα 32: Επιλέγουμε το βέλτιστο επίπεδο κατωφλίωσης

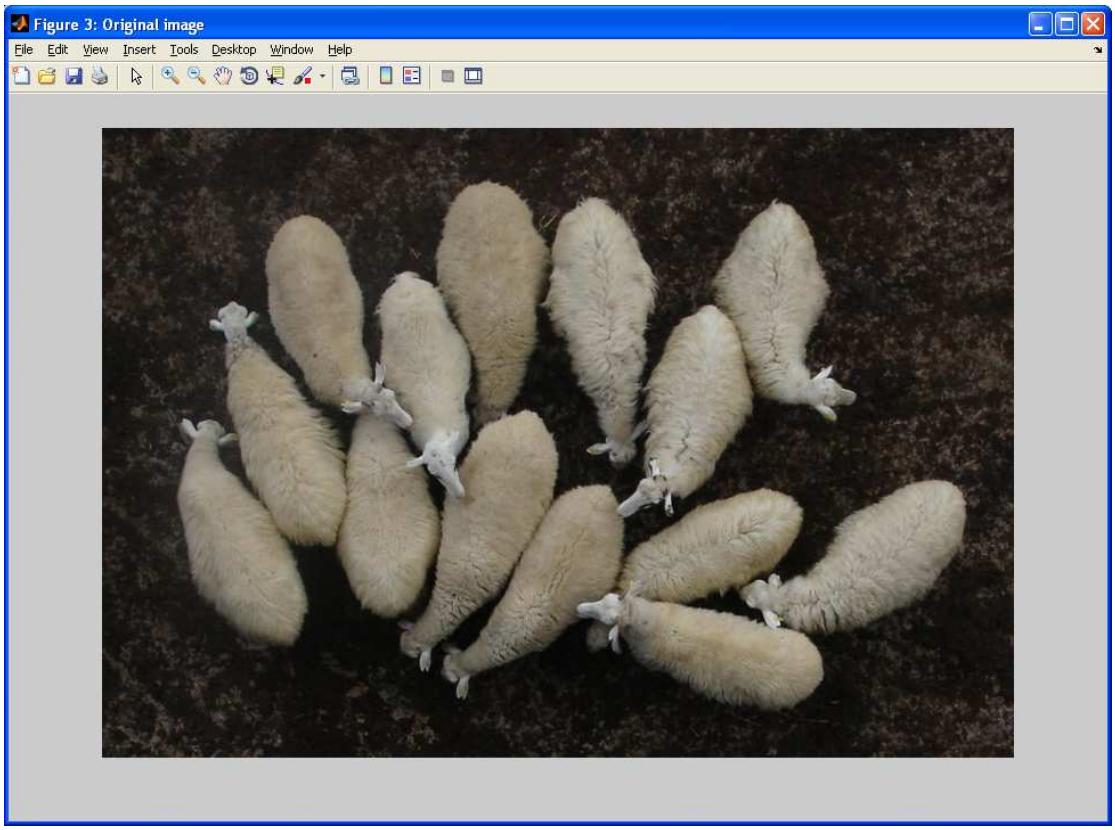


Εικόνα 33: Επιλέγουμε με το mouse κάνοντας διπλό κλικ σε ένα πρόβατο



Εικόνα 34: Τελική εικόνα με το σύνολο των προβάτων αριθμημένα

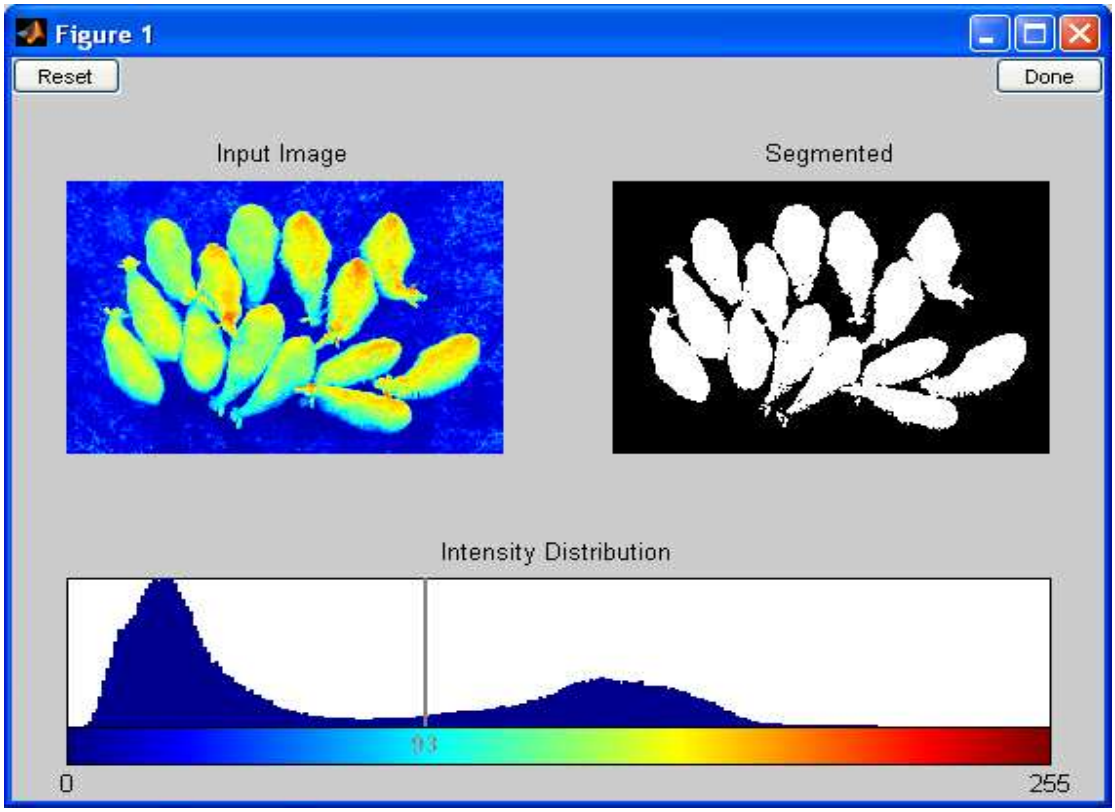
4.2 Δοκιμή με 14 πρόβατα



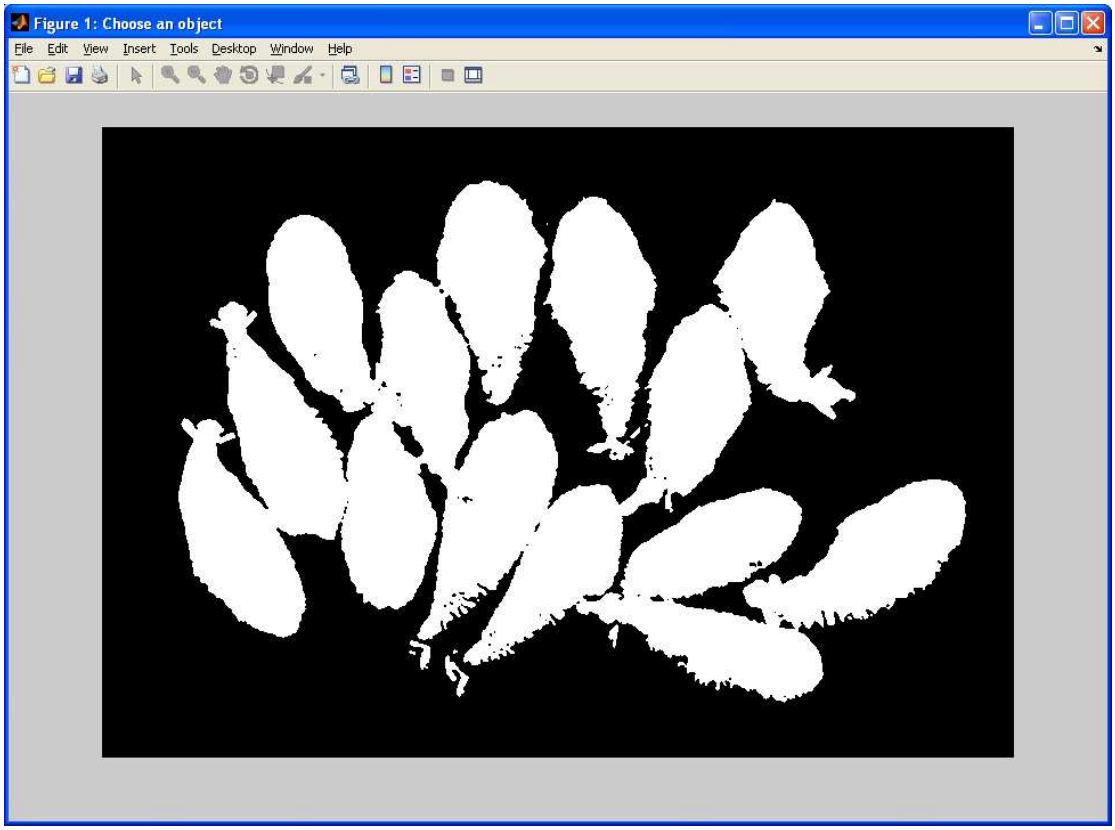
Εικόνα 35: Αρχική εικόνα με 14 πρόβατα

Εικόνα 36: Μετατροπή αρχικής έγχρωμης εικόνας σε ασπρόμαυρη

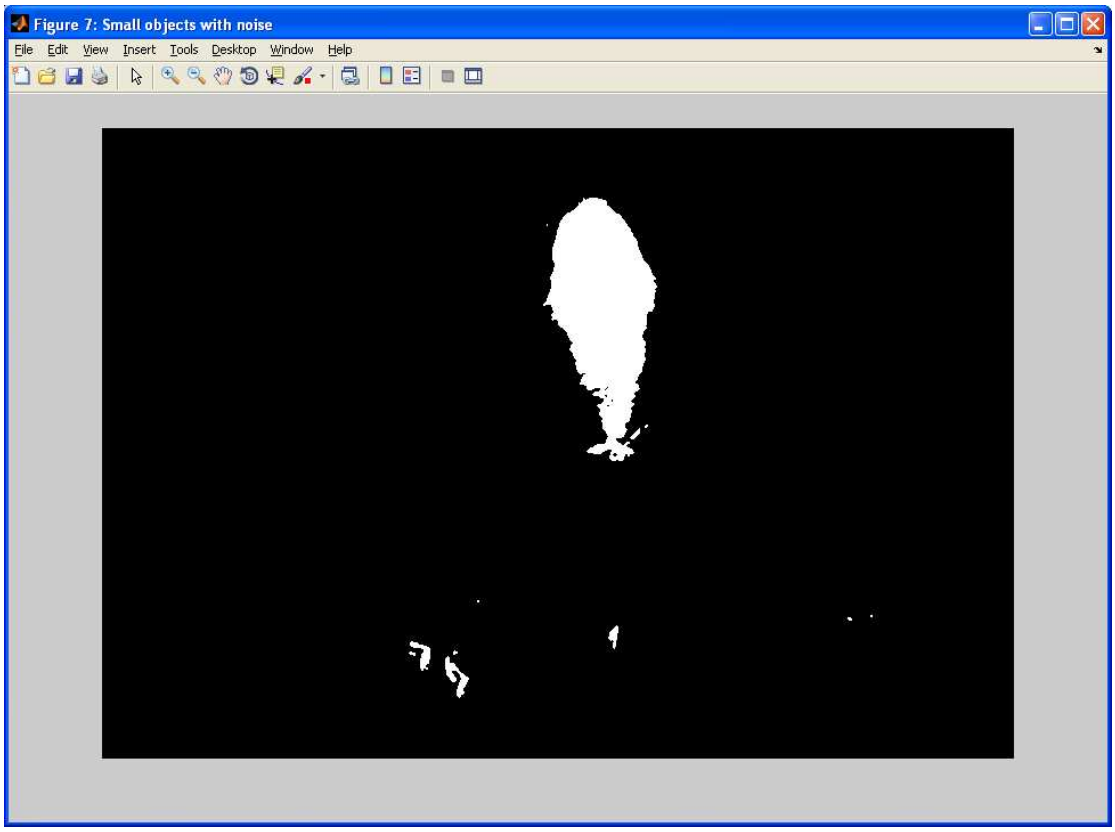
Εικόνα 37: Ασπρόμαυρη εικόνα που έχει περάσει από median filter



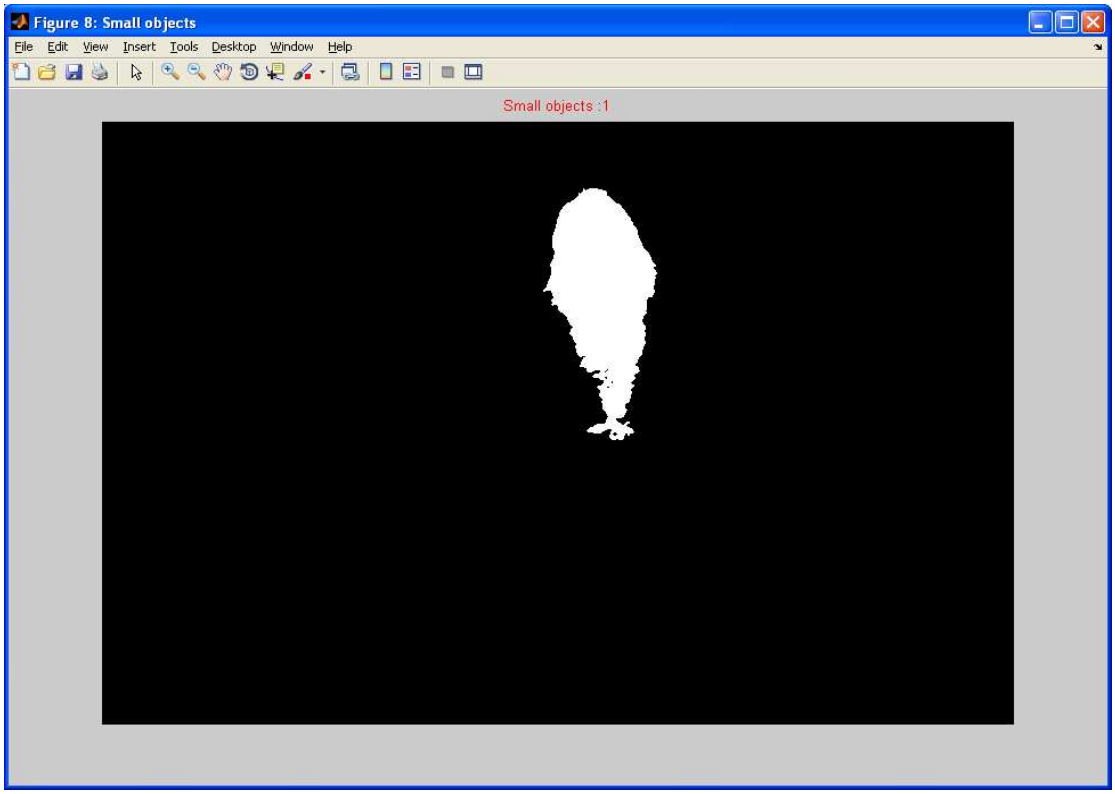
Εικόνα 38: Επιλέγουμε το βέλτιστο επίπεδο κατωφλίωσης



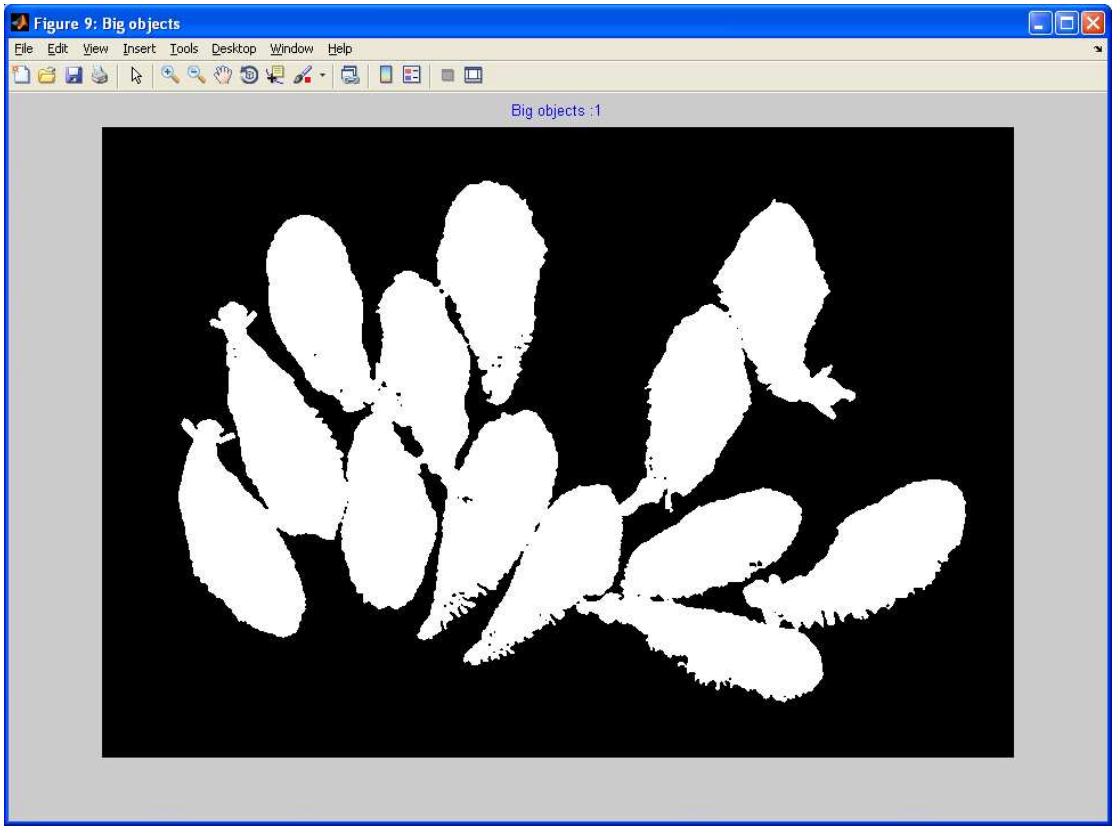
Εικόνα 39: Επιλέγουμε με το mouse κάνοντας διπλό κλικ σε ένα πρόβατο



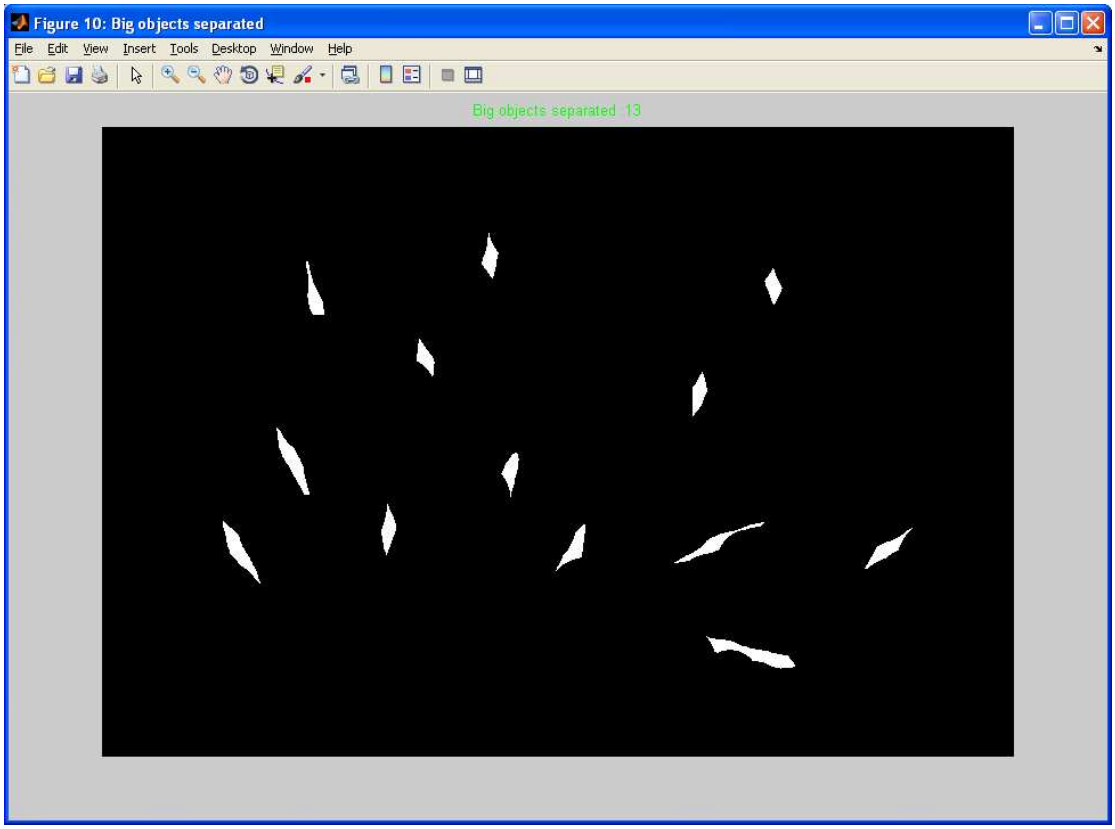
Εικόνα 40: Αντικείμενα μικρού μεγέθους



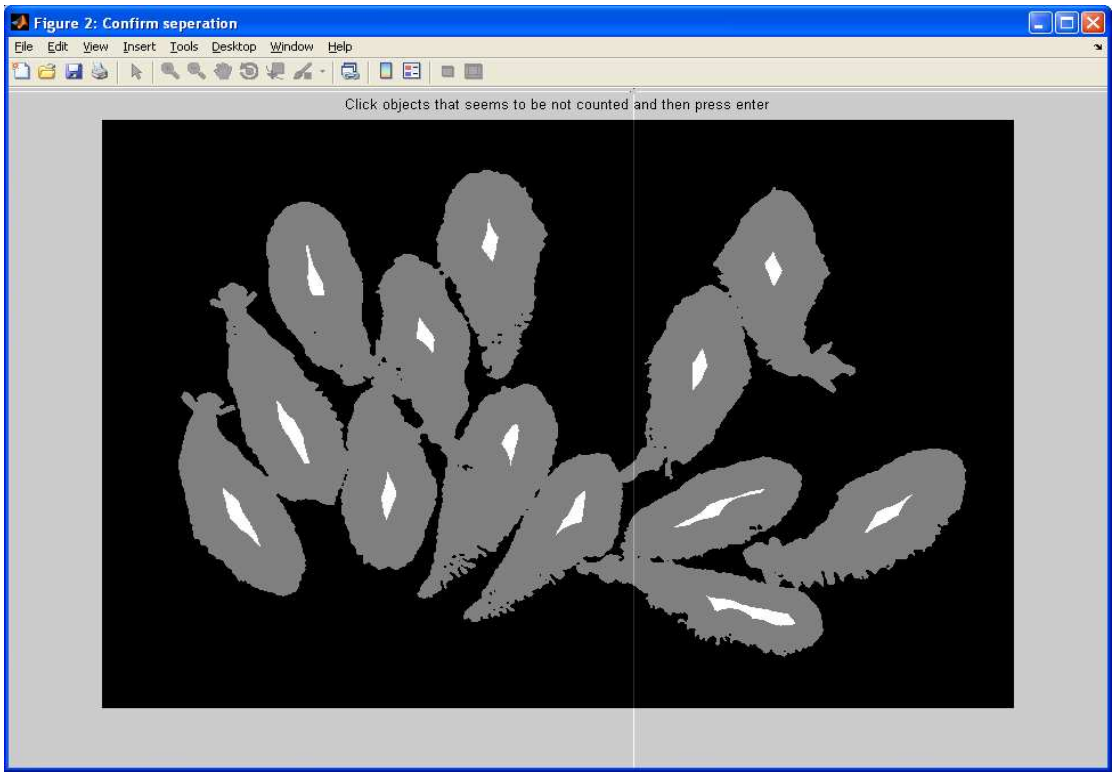
Εικόνα 41: Αντικείμενα μικρού μεγέθους χωρίς τον θόρυβο



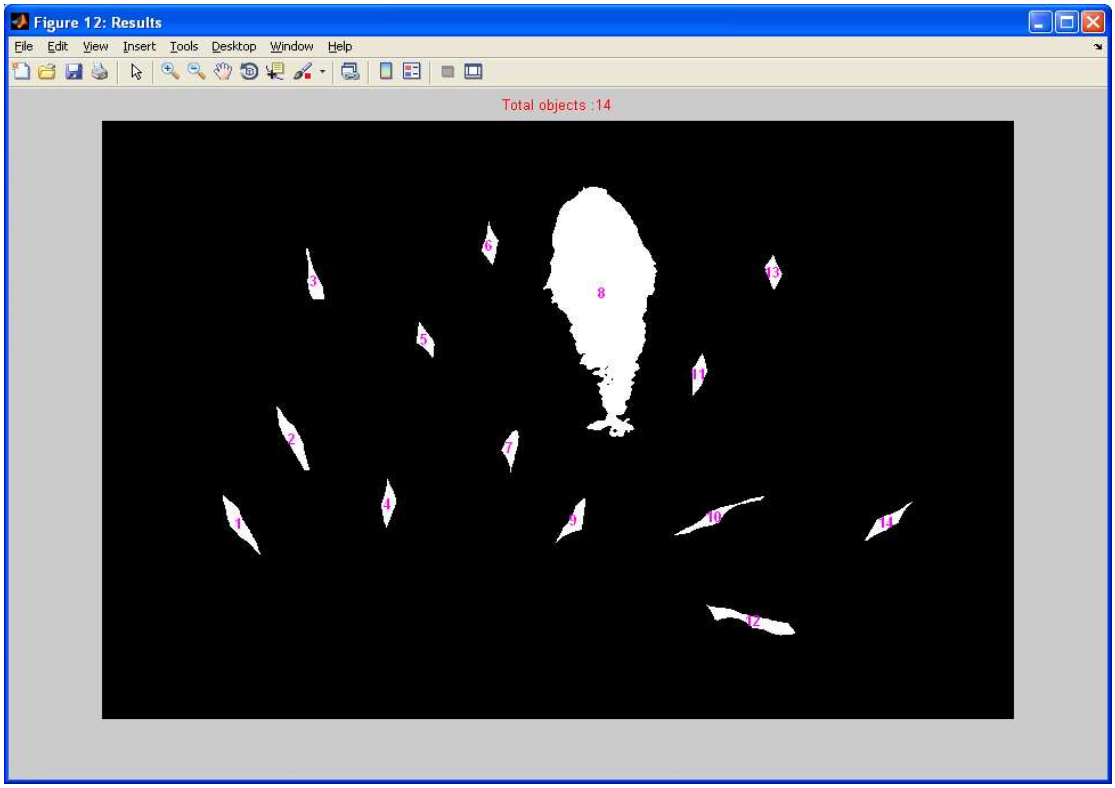
Εικόνα 42: Αντικείμενα μεγάλου μεγέθους



Εικόνα 43: Αντικείμενα μεγάλου μεγέθους διασπασμένα

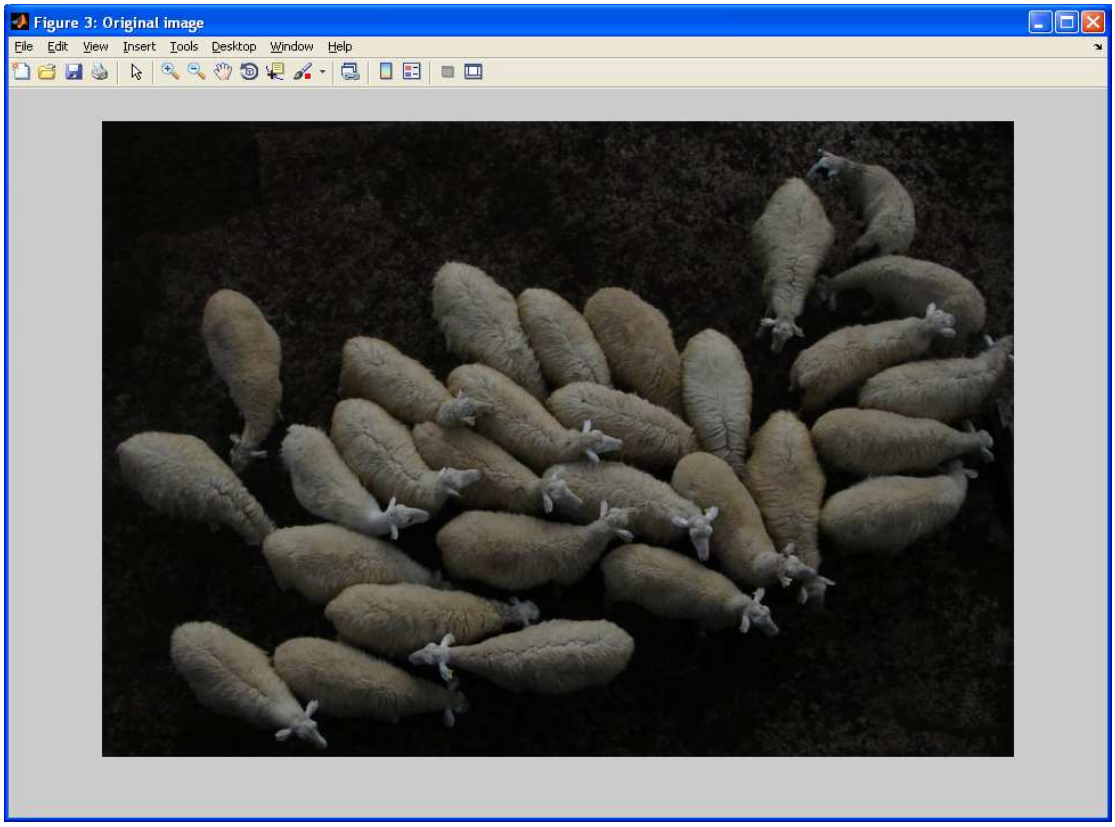


Εικόνα 44: Επαλήθευση από τον χρήστη ότι ο διαχωρισμός των προβάτων είναι έγκυρος

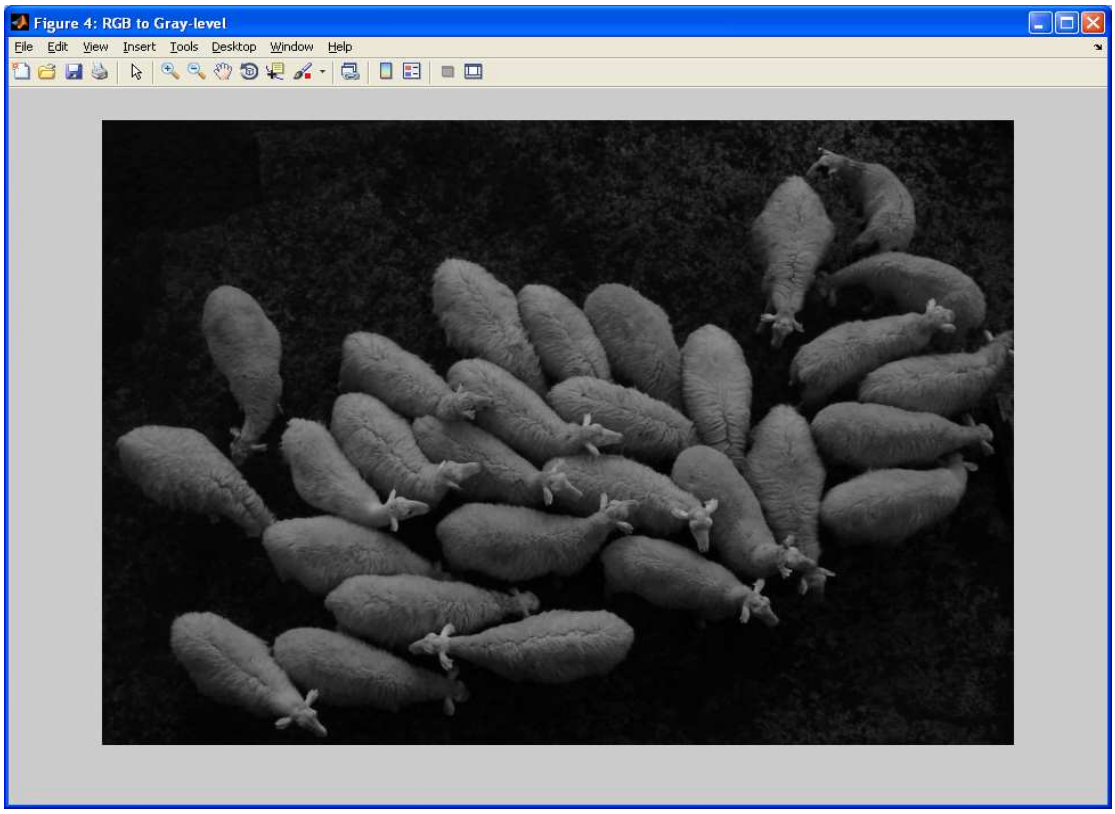


Εικόνα 45: Τελική εικόνα με το σύνολο των προβάτων αριθμημένα

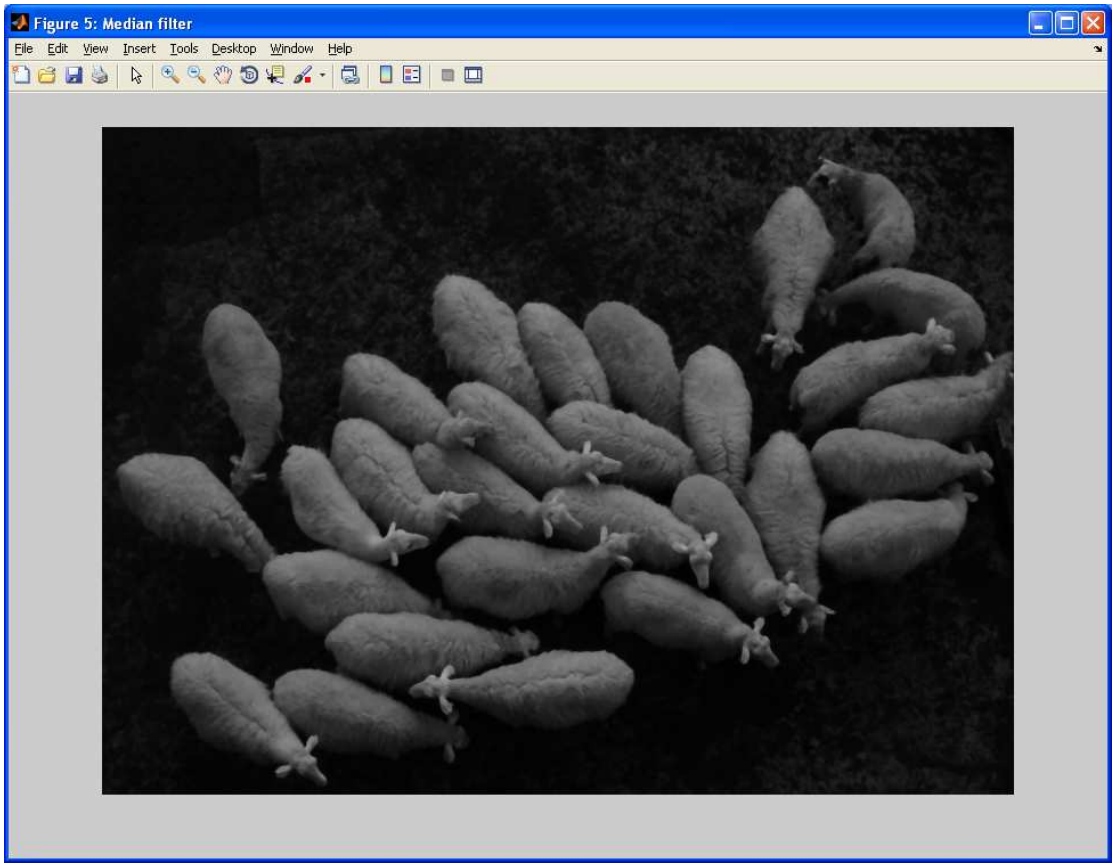
4.3 Δοκιμή με 29 πρόβατα



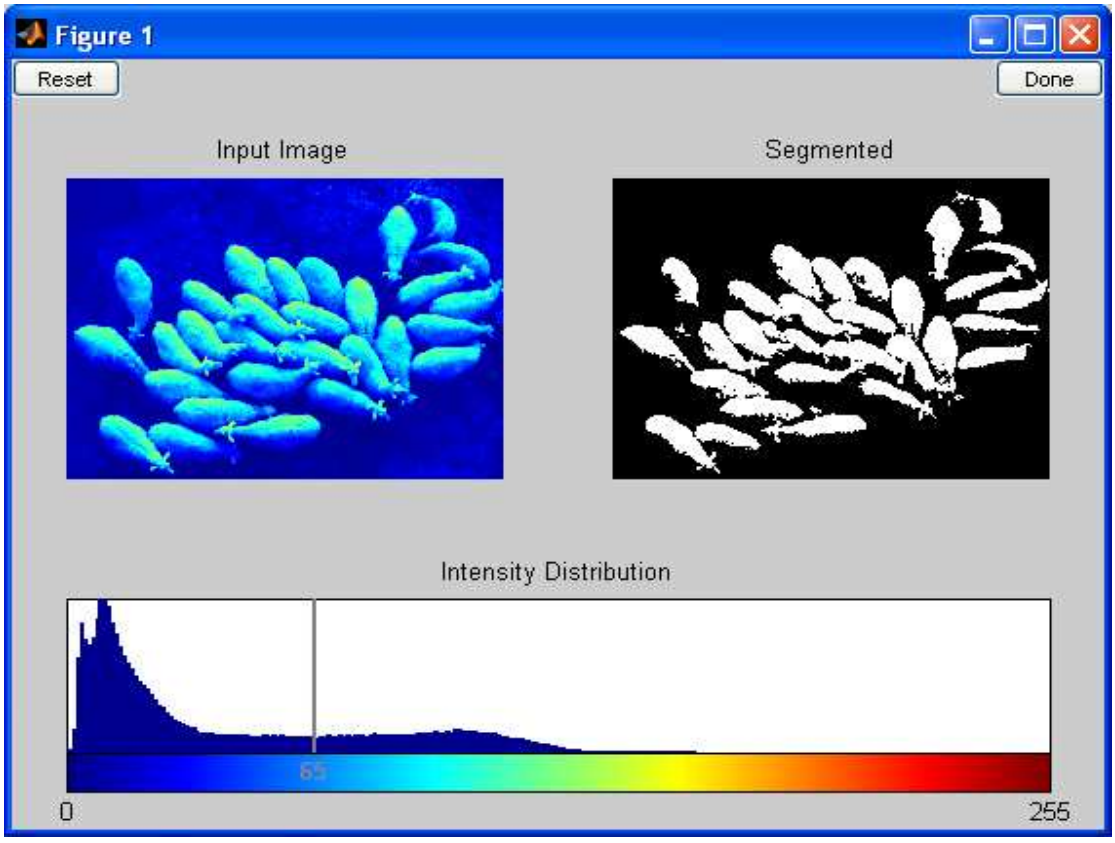
Εικόνα 46: Αρχική εικόνα με 29 πρόβατα



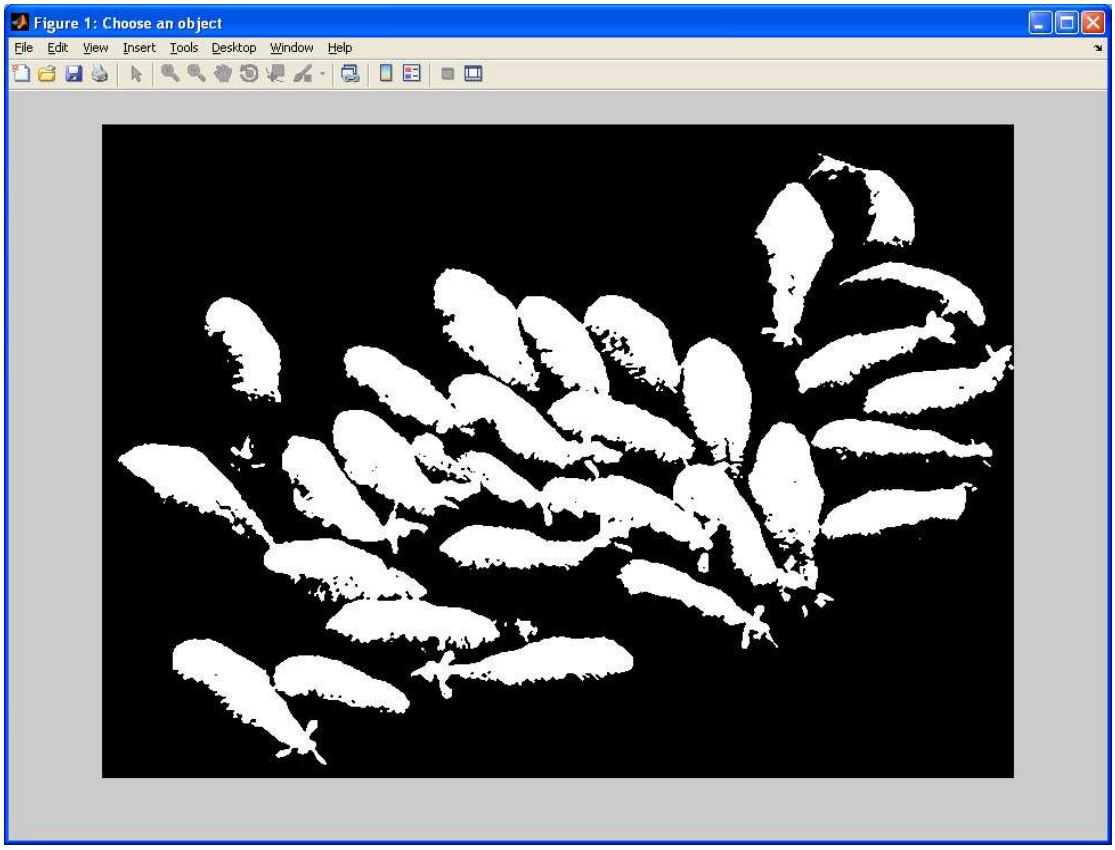
Εικόνα 47: Μετατροπή αρχικής έγχρωμης εικόνας σε ασπρόμαυρη



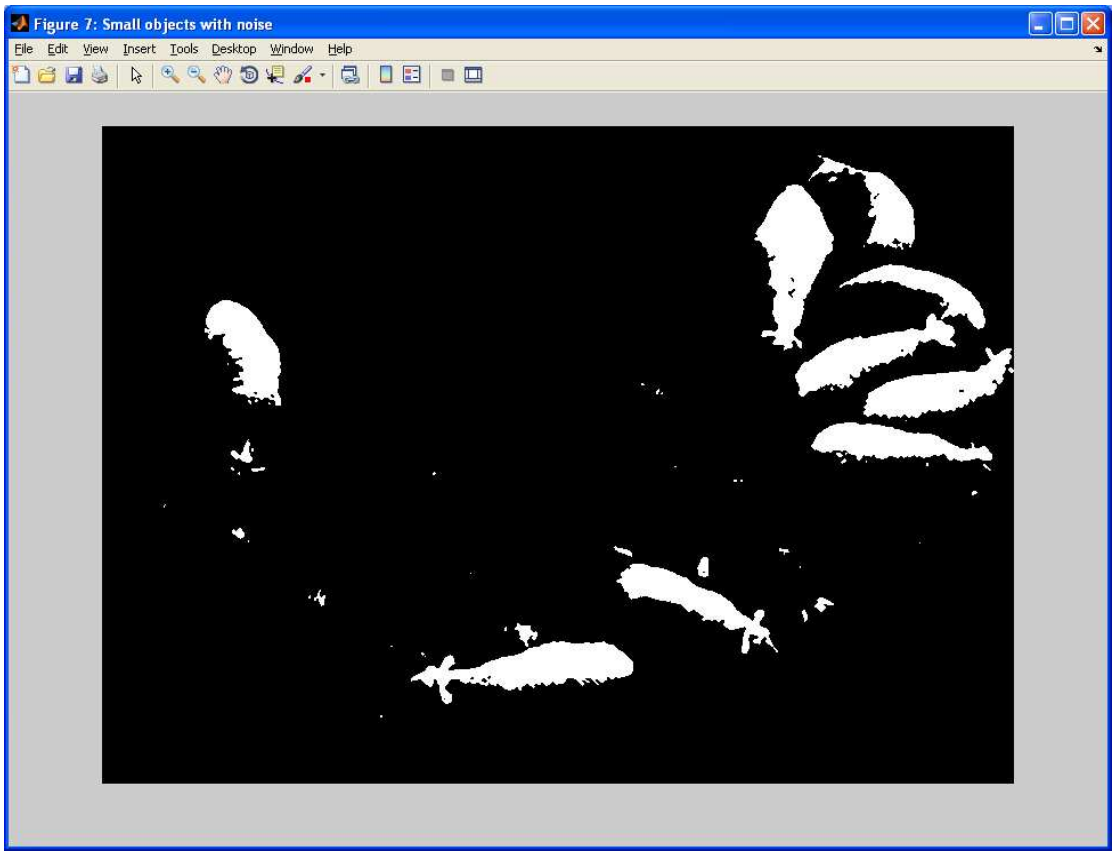
Εικόνα 48: Ασπρόμαυρη εικόνα που έχει περάσει από median filter



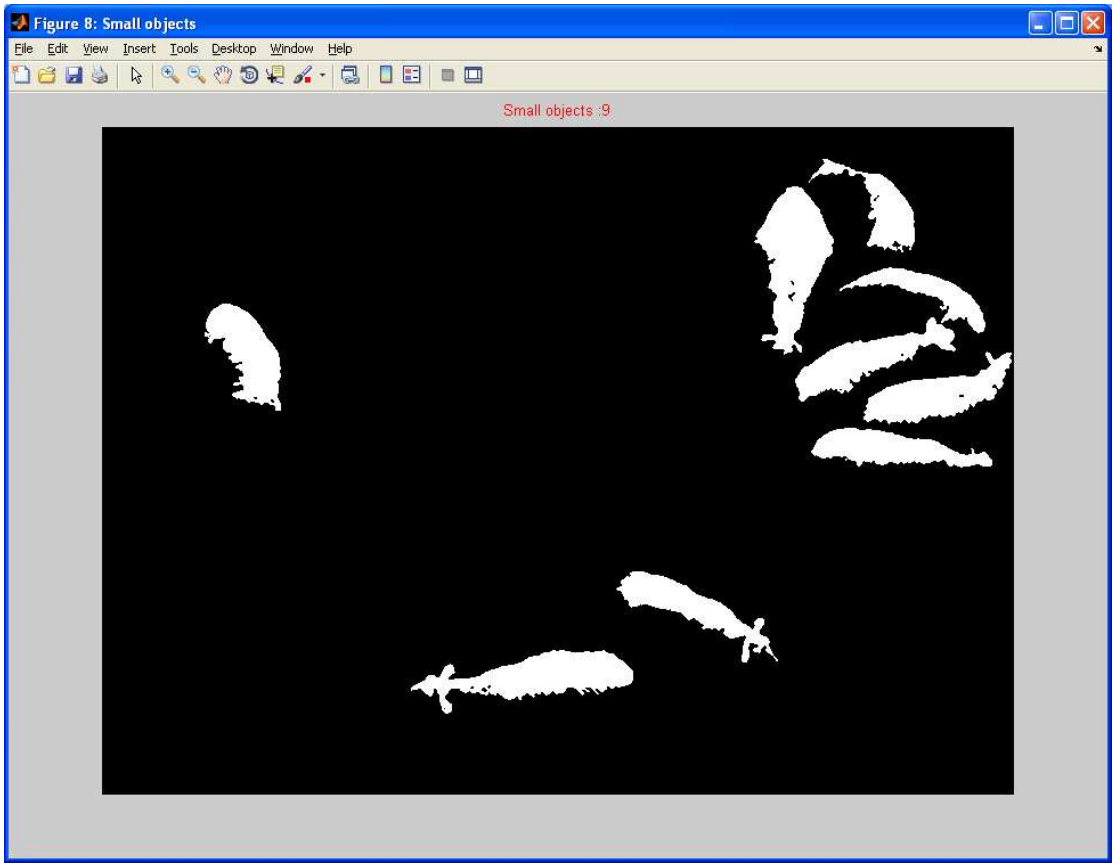
Εικόνα 49: Επιλέγουμε το βέλτιστο επίπεδο κατωφλίωσης



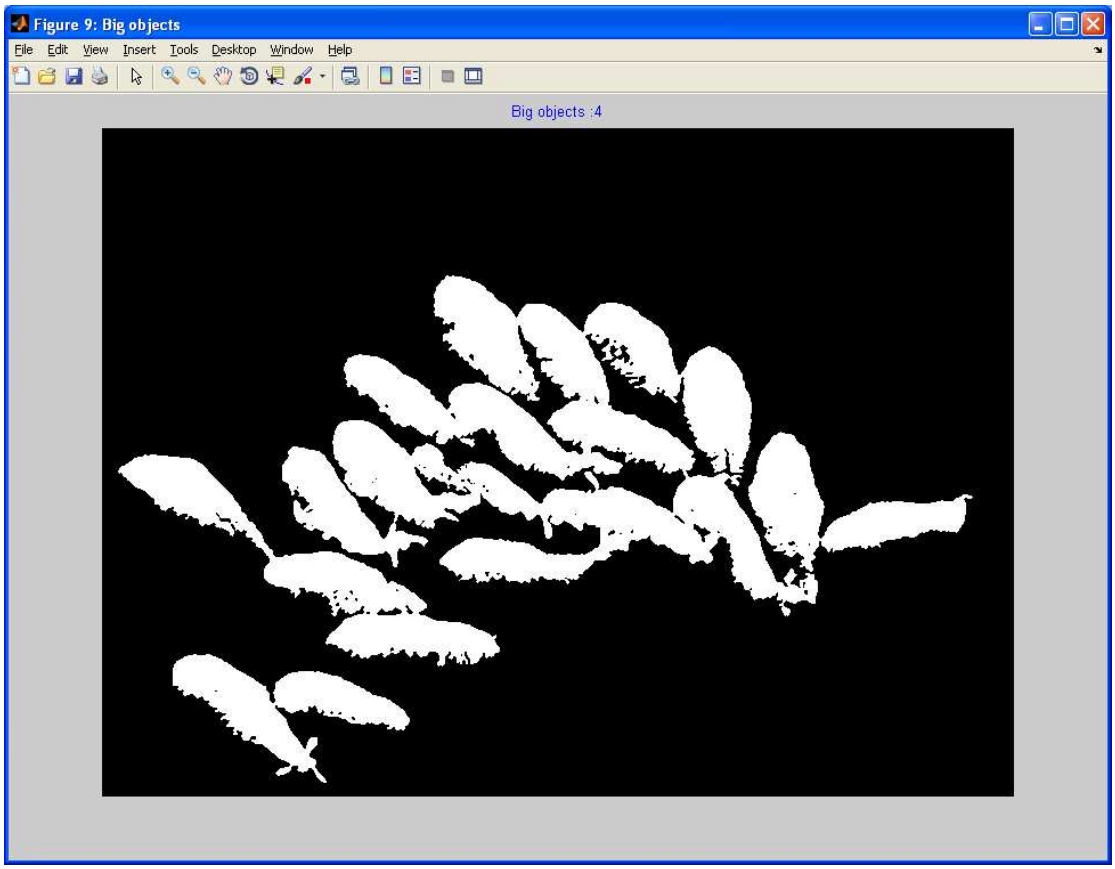
Εικόνα 50: Επιλέγουμε με το mouse κάνοντας διπλό κλικ σε ένα πρόβατο



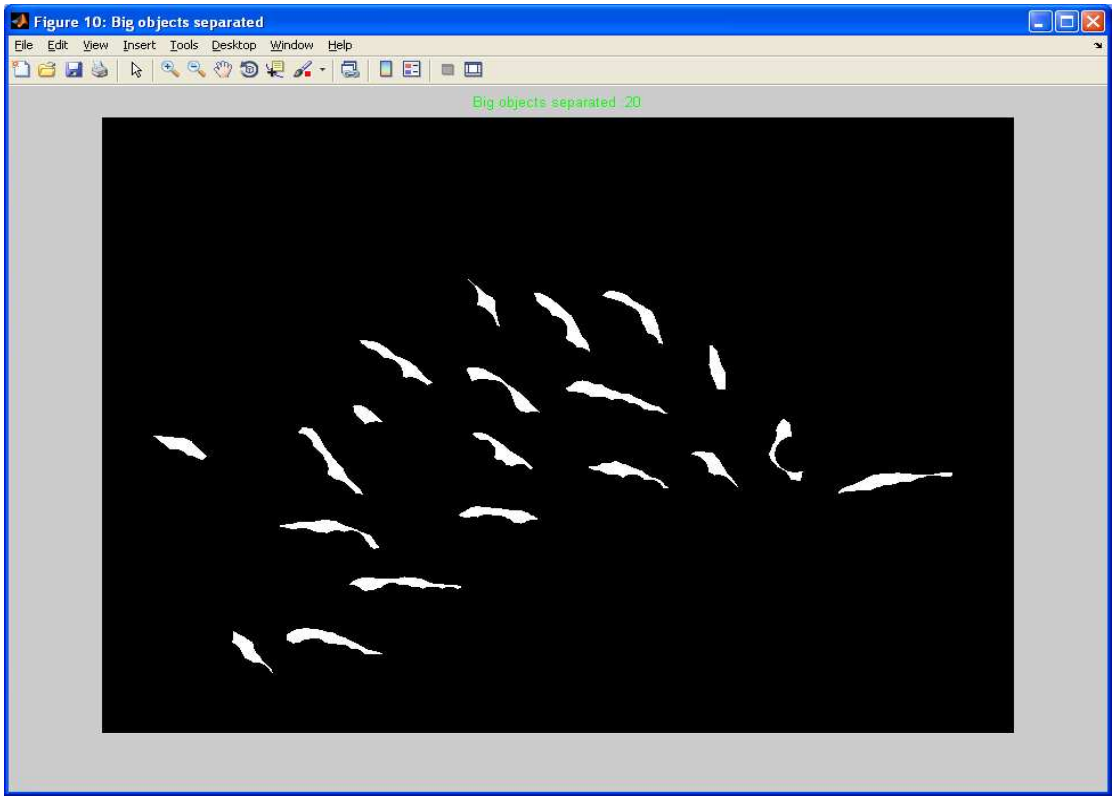
Εικόνα 51: Αντικείμενα μικρού μεγέθους



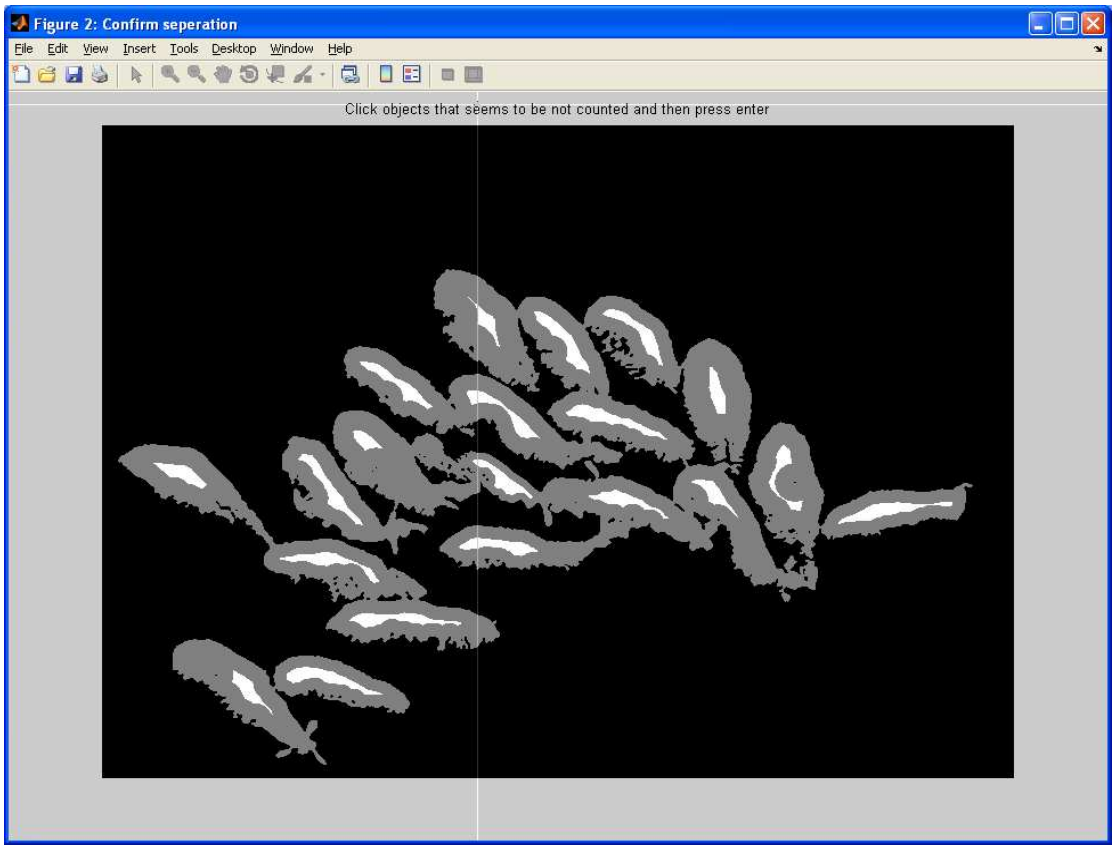
Εικόνα 52: Αντικείμενα μικρού μεγέθους χωρίς τον θόρυβο



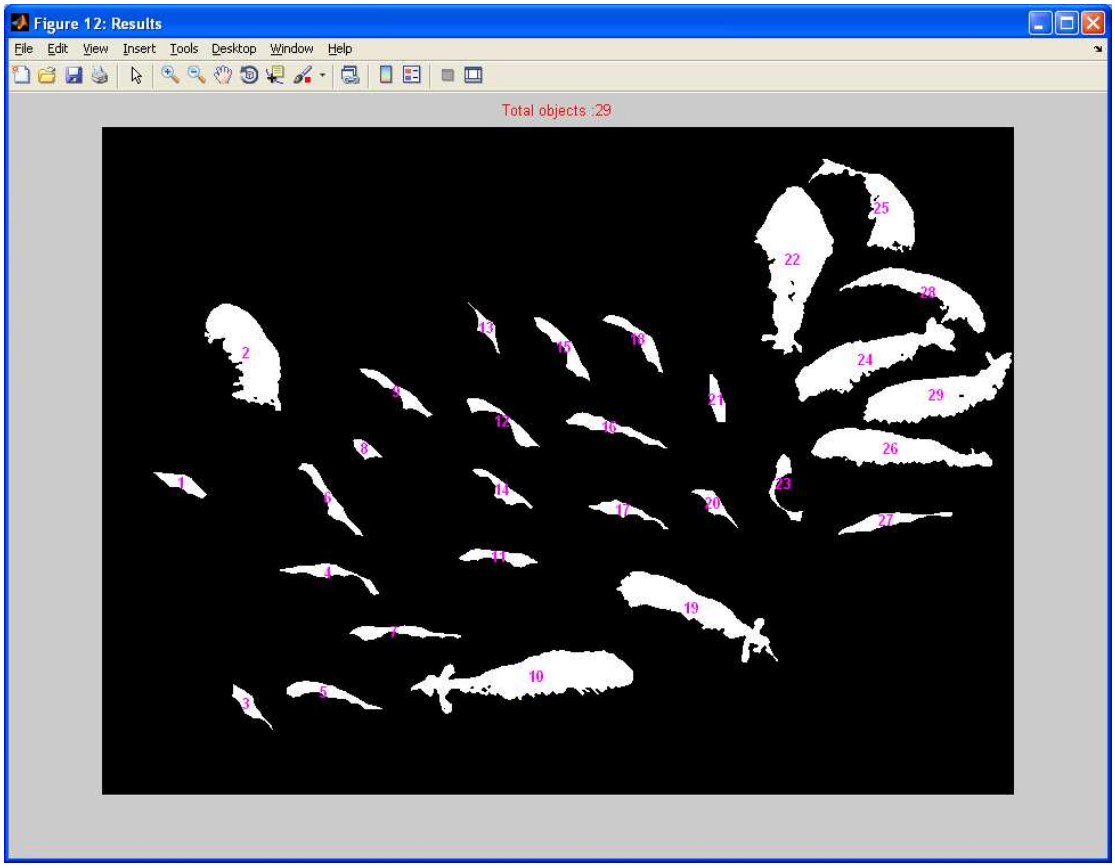
Εικόνα 53: Αντικείμενα μεγάλου μεγέθους



Εικόνα 54: Αντικείμενα μεγάλου μεγέθους διασπασμένα

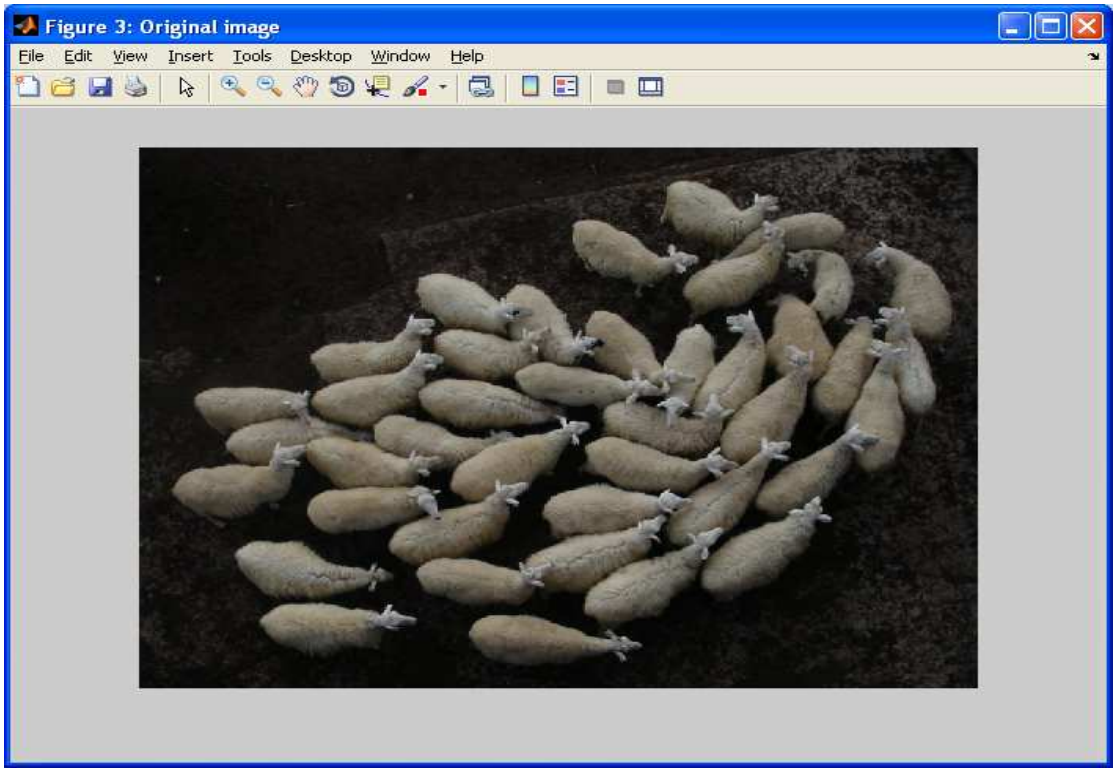


Εικόνα 55: Επαλήθευση από τον χρήστη ότι ο διαχωρισμός των προβάτων είναι έγκυρος

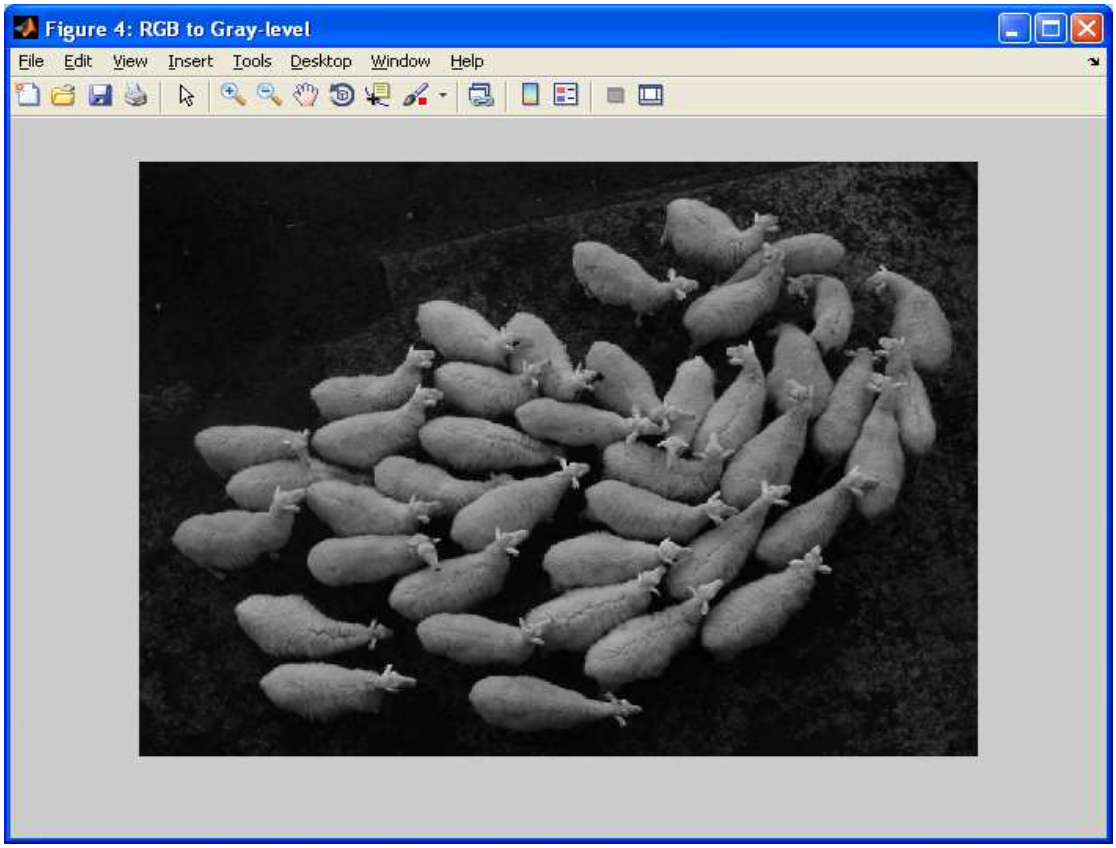


Εικόνα 56: Τελική εικόνα με το σύνολο των προβάτων αριθμημένα

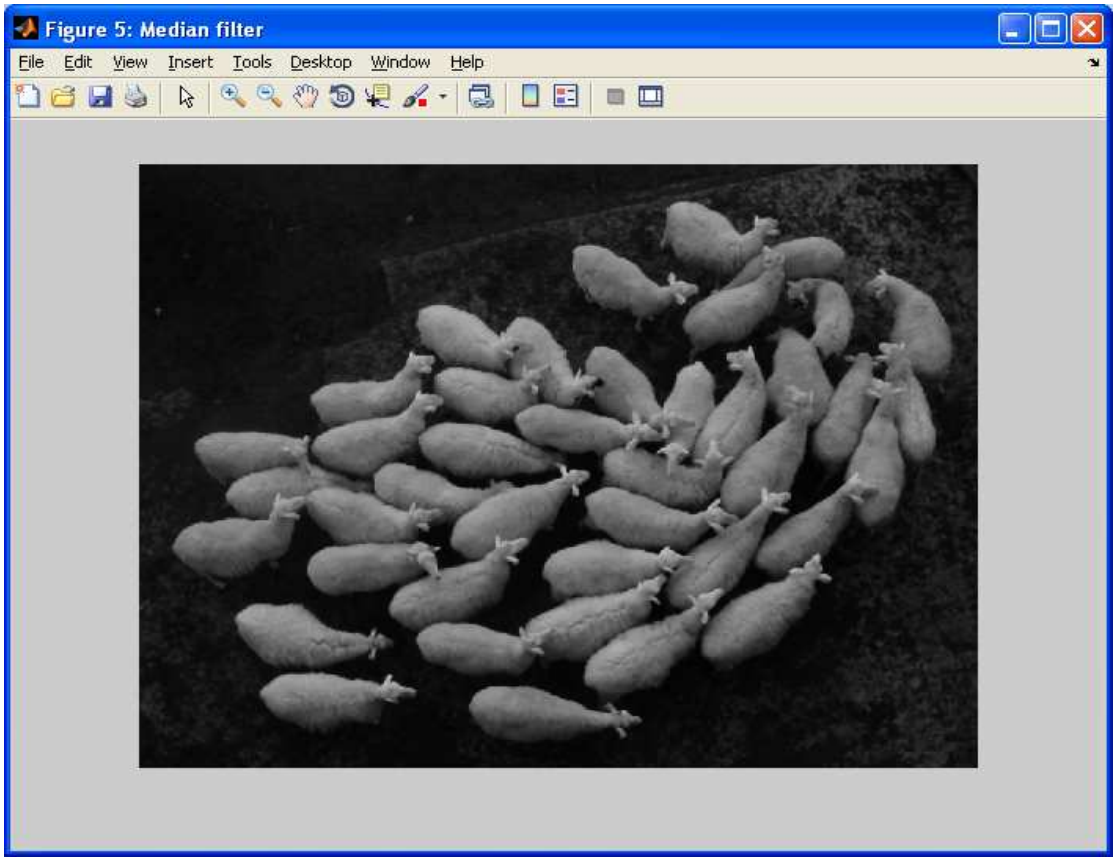
4.4 Δοκιμή με 41 πρόβατα



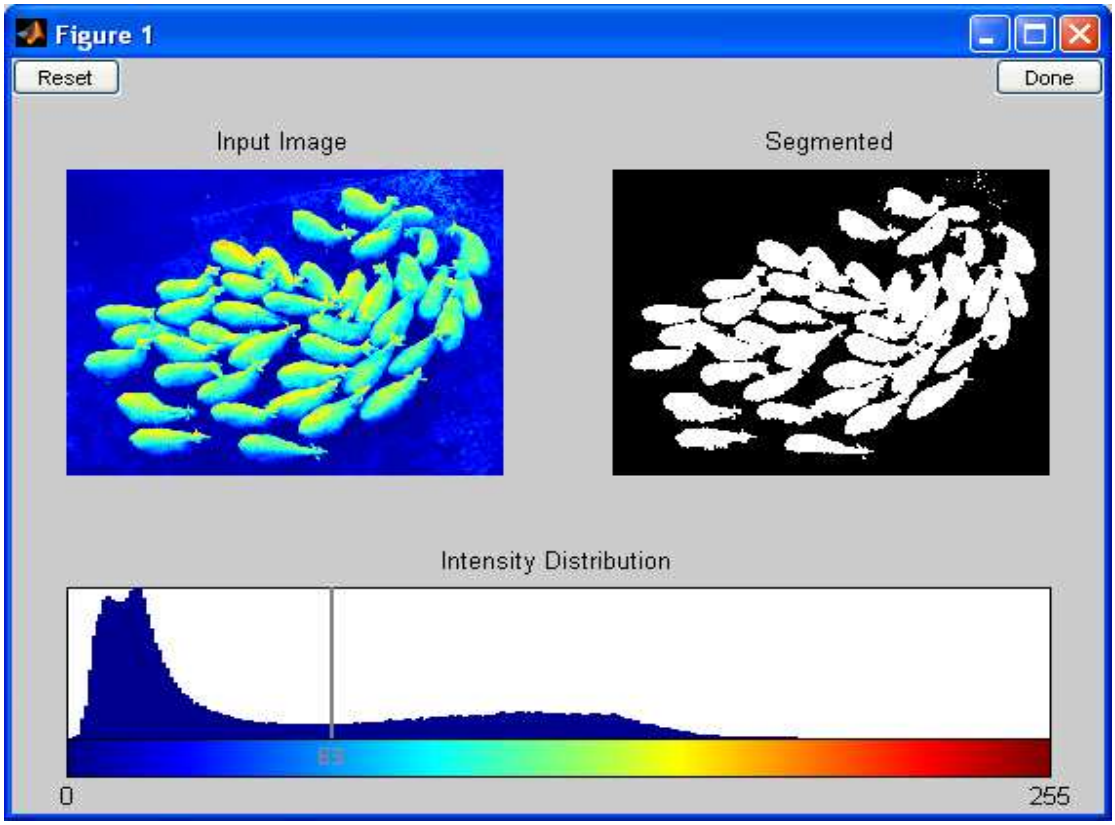
Εικόνα 57: Αρχική εικόνα με 41 πρόβατα



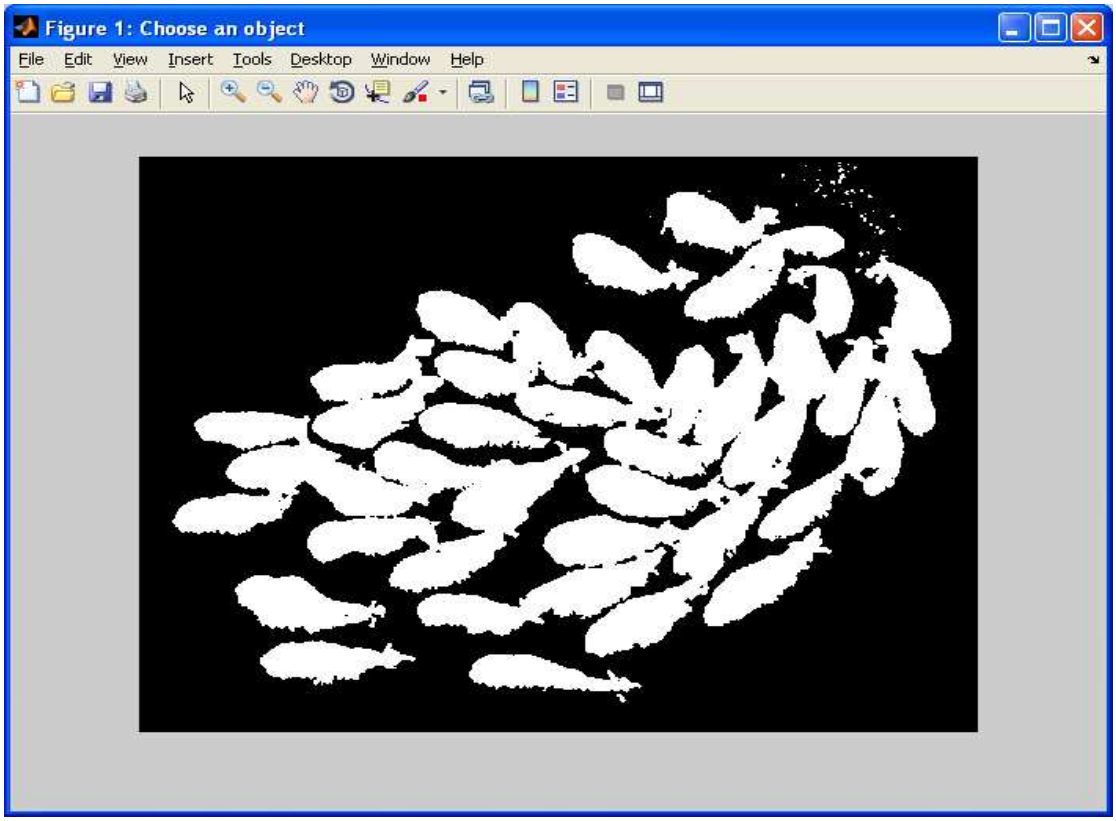
Εικόνα 58: Μετατροπή αρχικής έγχρωμης εικόνας σε ασπρόμαυρη



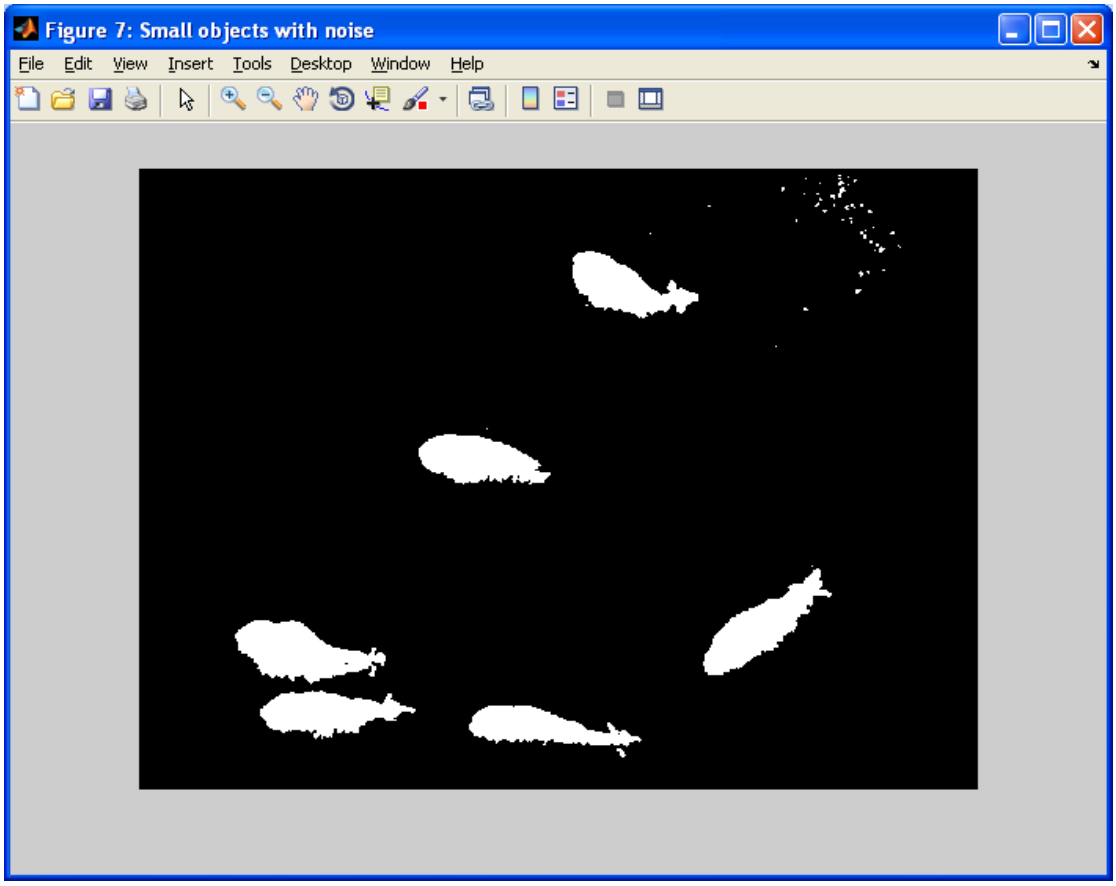
Εικόνα 59: Ασπρόμαυρη εικόνα που έχει περάσει από median filter



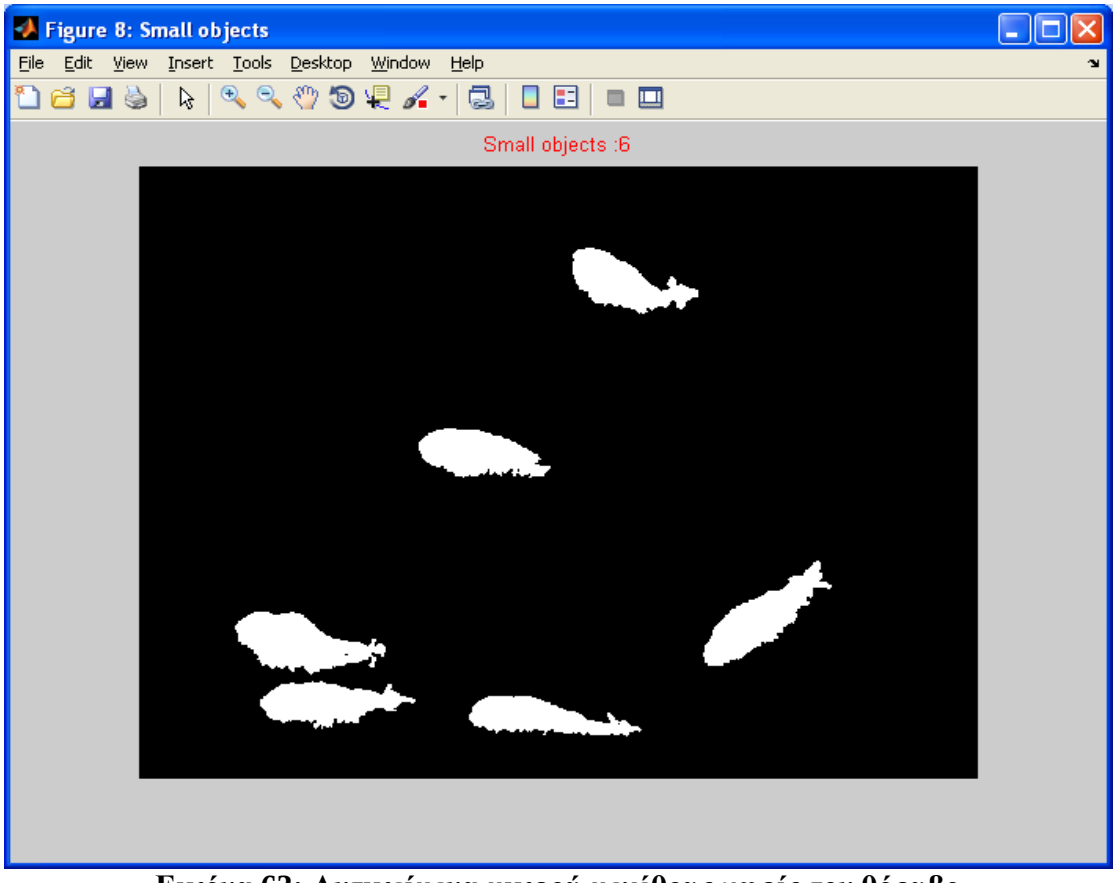
Εικόνα 60: Επιλέγουμε το βέλτιστο επίπεδο κατωφλίωσης



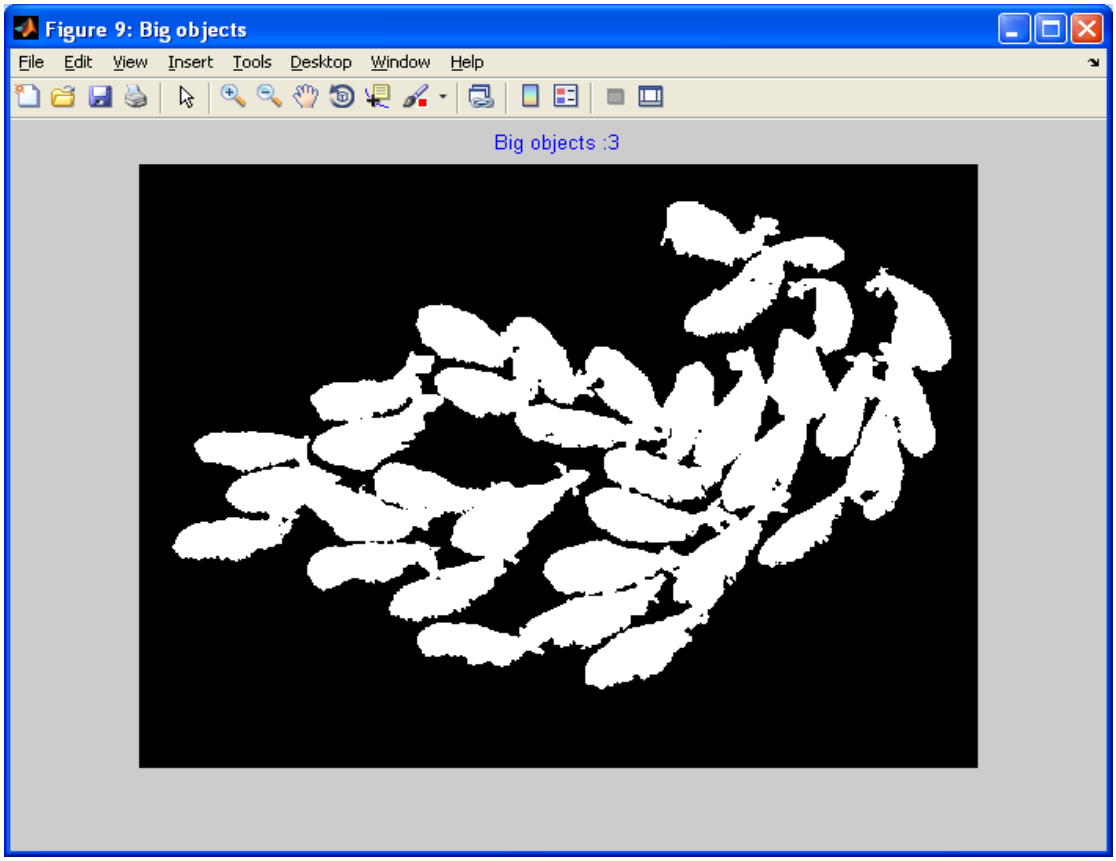
Εικόνα 61: Επιλέγουμε με το mouse κάνοντας διπλό κλικ σε ένα πρόβατο



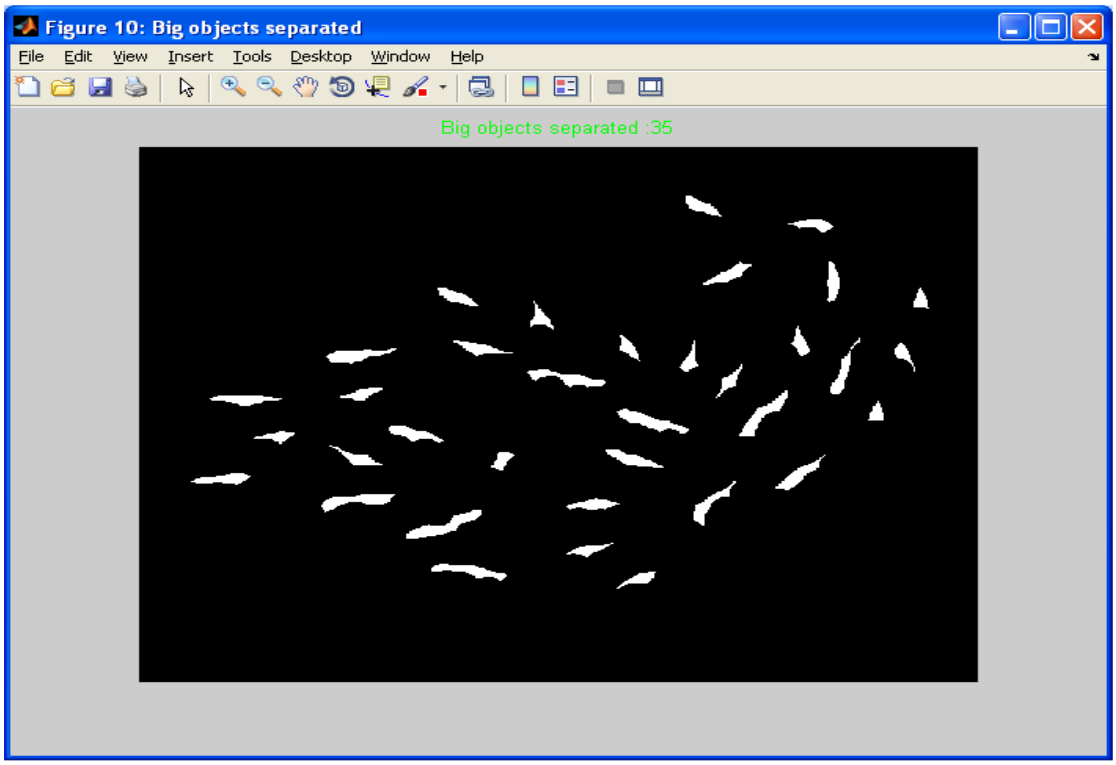
Εικόνα 62: Αντικείμενα μικρού μεγέθους



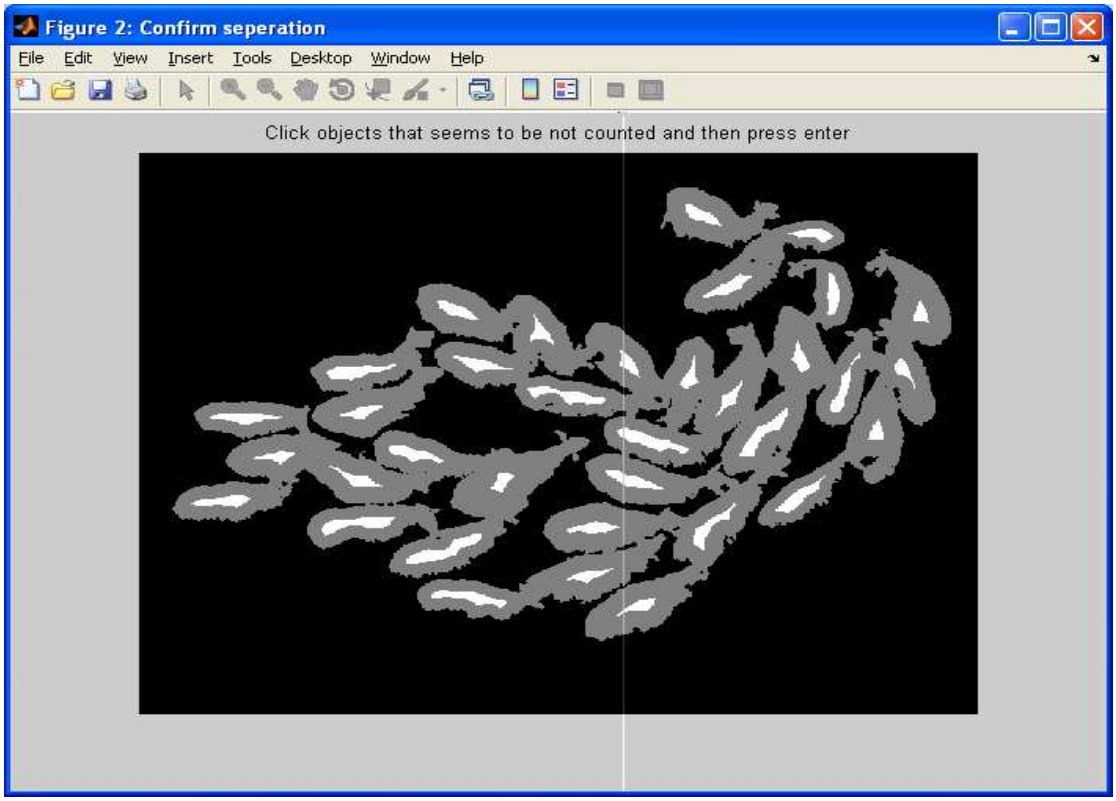
Εικόνα 63: Αντικείμενα μικρού μεγέθους χωρίς τον θόρυβο



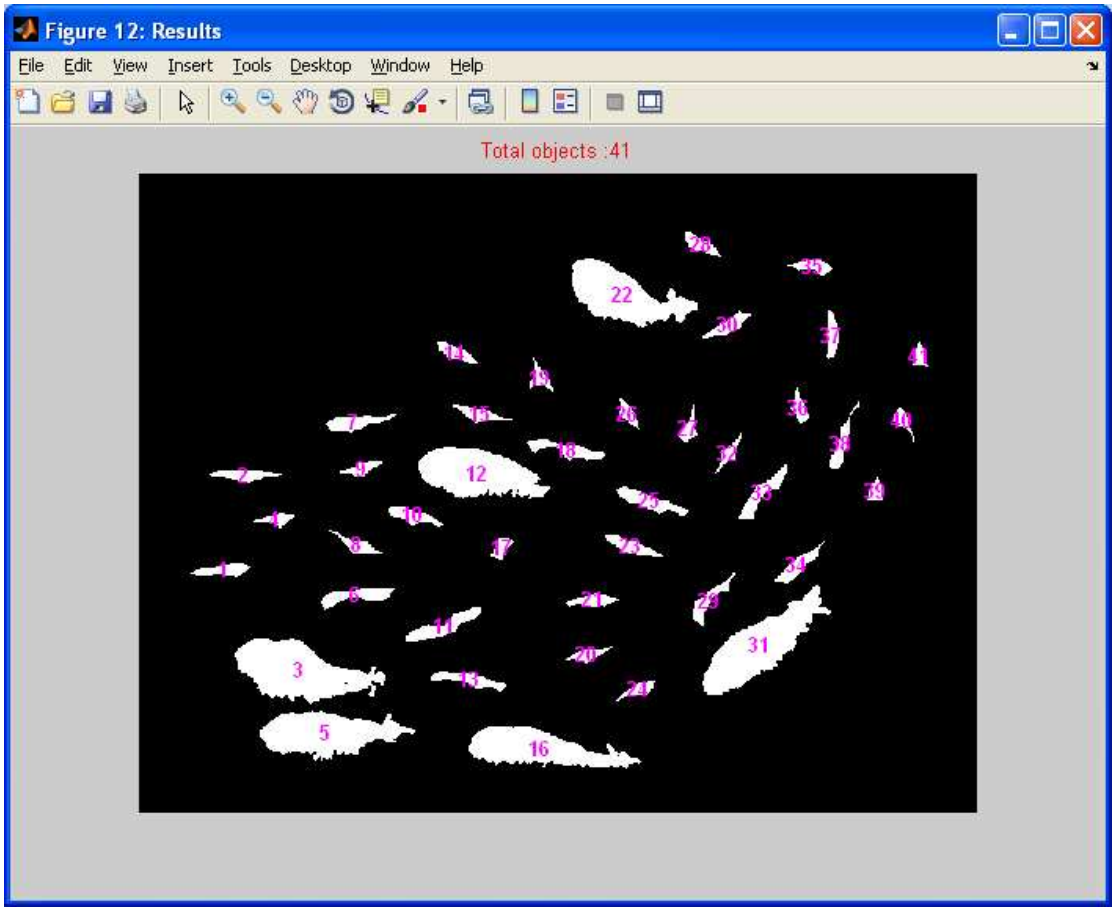
Εικόνα 64: Αντικείμενα μεγάλου μεγέθους



Εικόνα 65: Αντικείμενα μεγάλου μεγέθους διασπασμένα

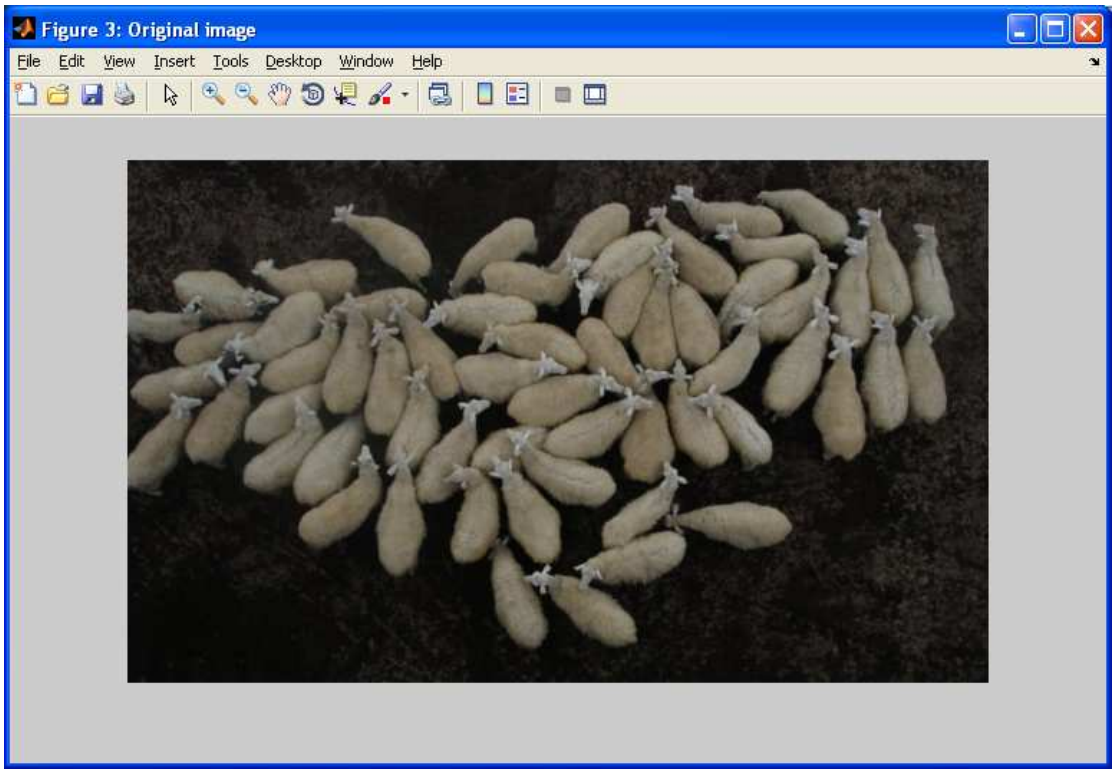


Εικόνα 66: Επαλήθευση από τον χρήστη ότι ο διαχωρισμός των προβάτων είναι έγκυρος

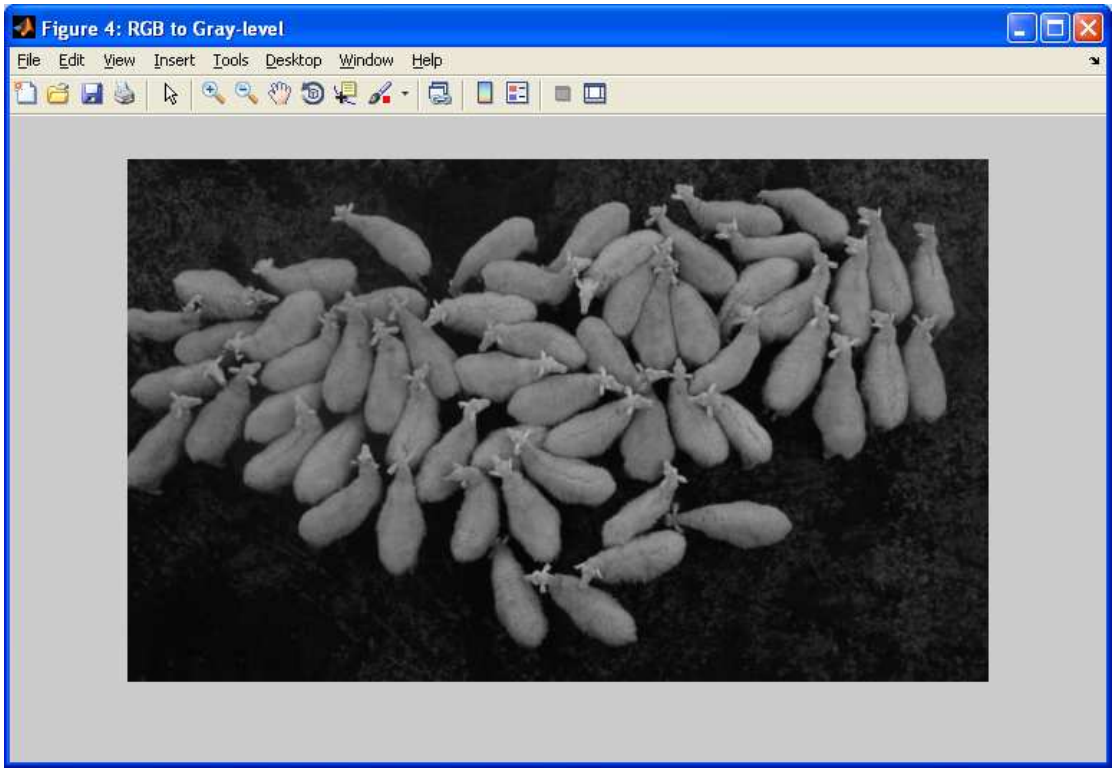


Εικόνα 67: Τελική εικόνα με το σύνολο των προβάτων αριθμημένα

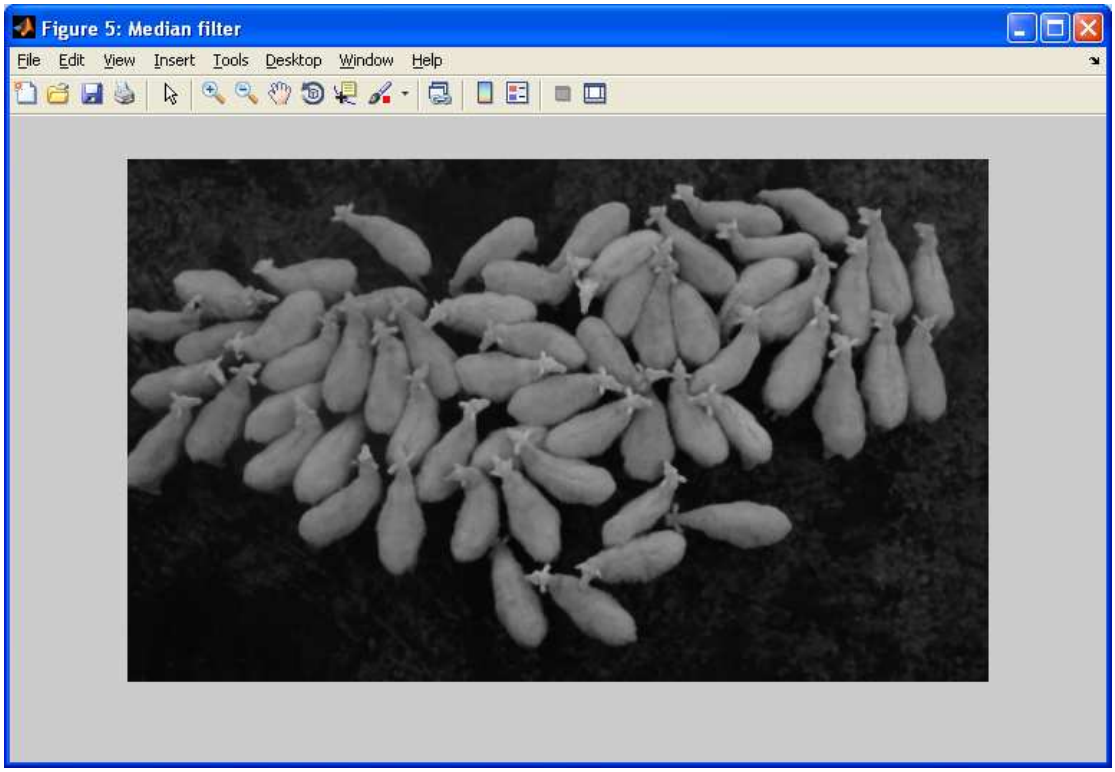
4.5 Δοκιμή με 60 πρόβατα



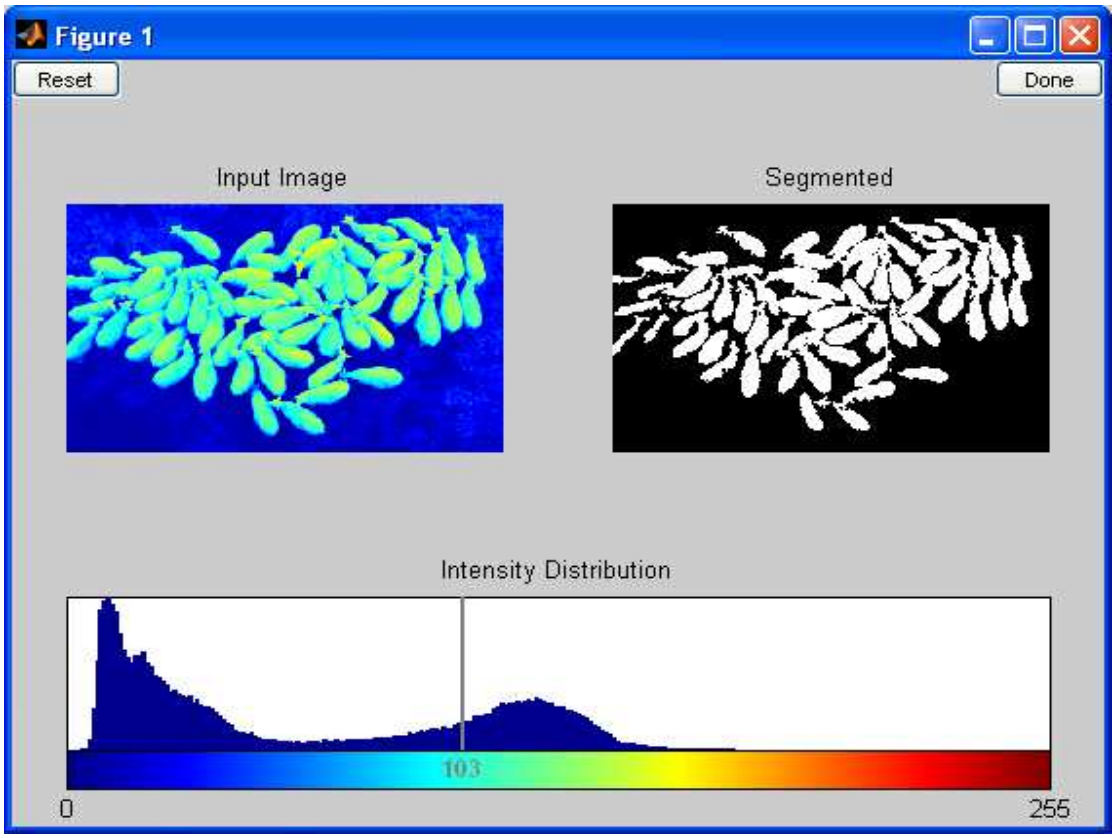
Εικόνα 68: Αρχική εικόνα με 60 πρόβατα



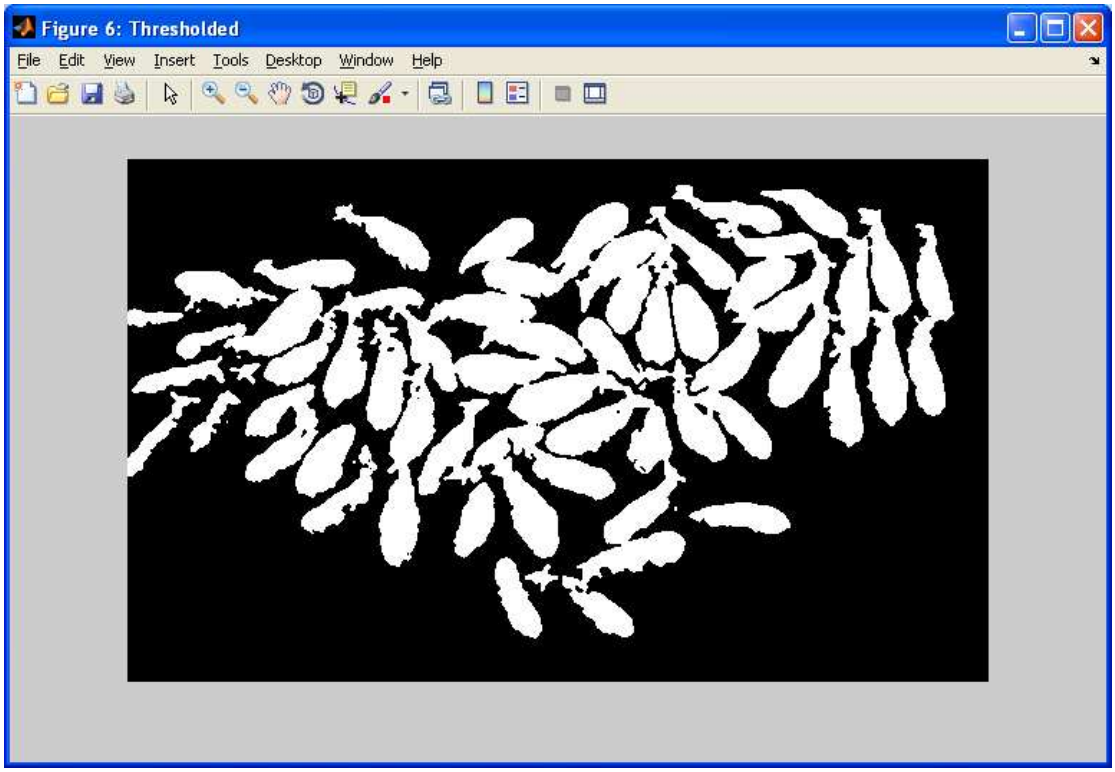
Εικόνα 69: Μετατροπή αρχικής έγχρωμης εικόνας σε ασπρόμαυρη



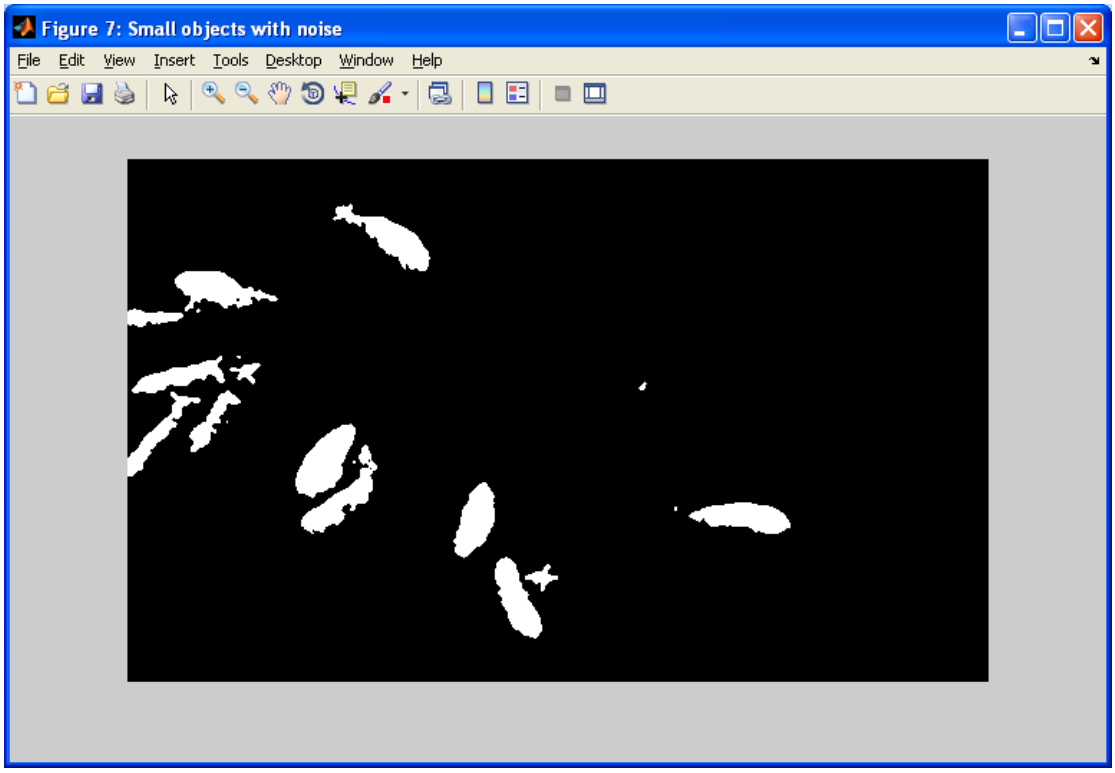
Εικόνα 70: Ασπρόμαυρη εικόνα που έχει περάσει από median filter



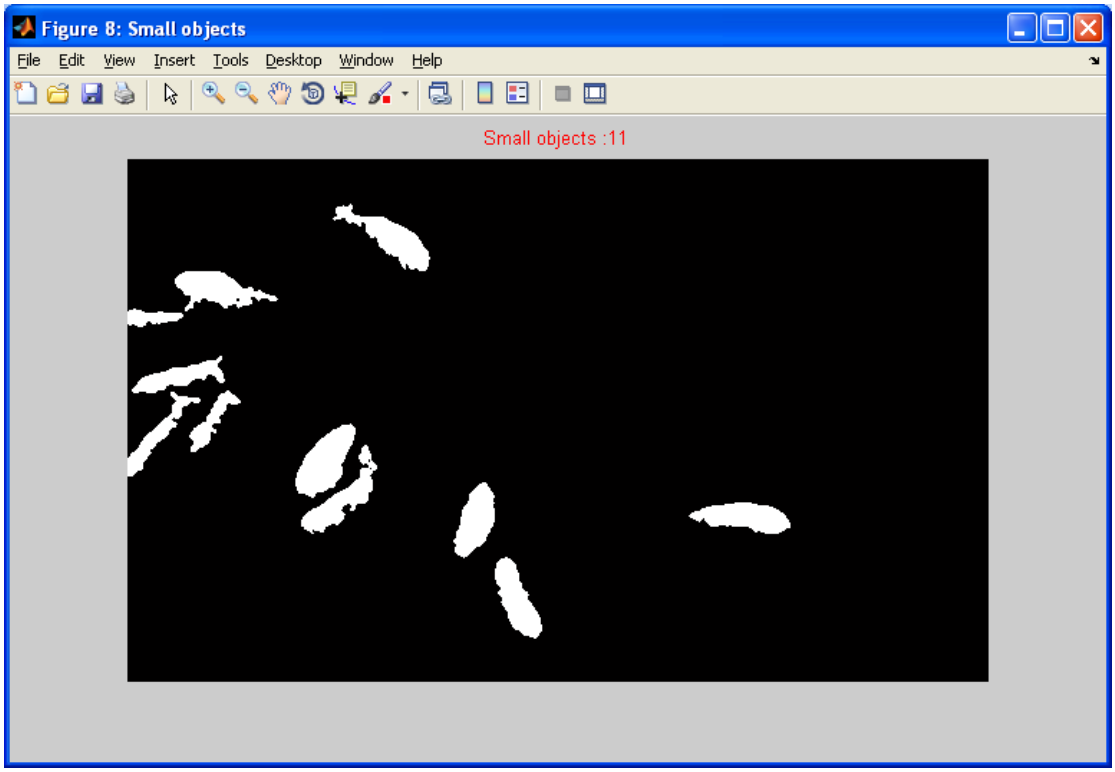
Εικόνα 71: Επιλέγουμε το βέλτιστο επίπεδο κατωφλίωσης



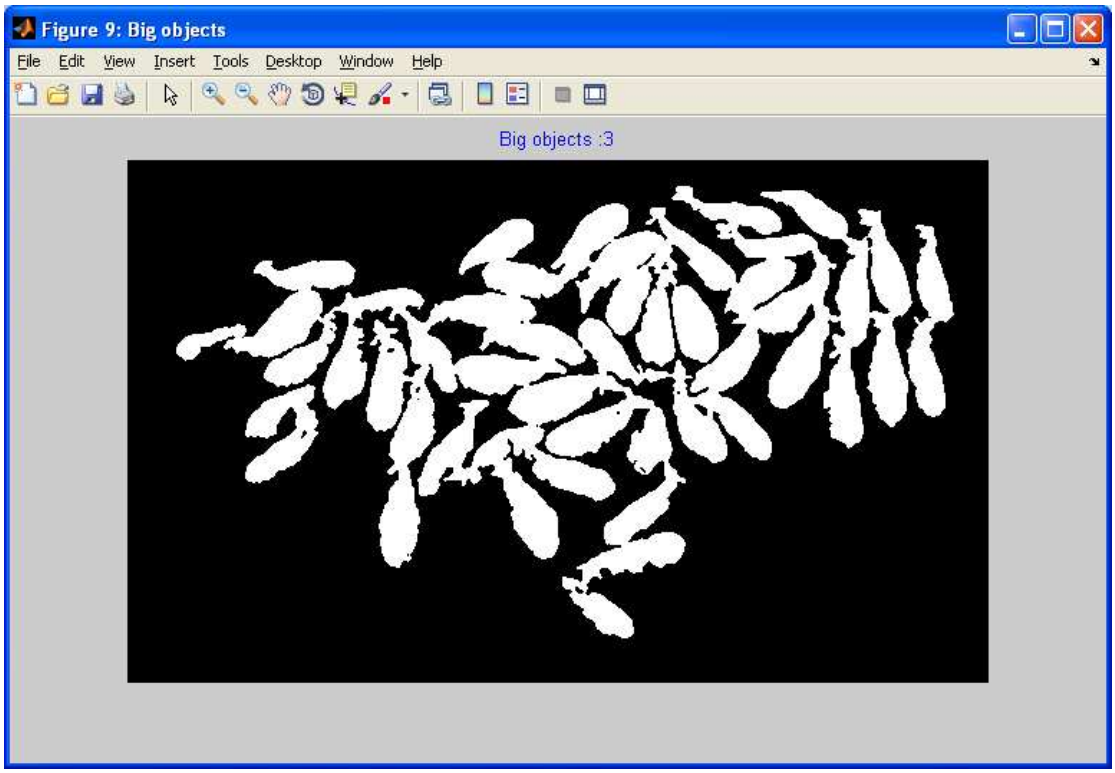
Εικόνα 72: Επιλέγουμε με το mouse κάνοντας διπλό κλικ σε ένα πρόβατο



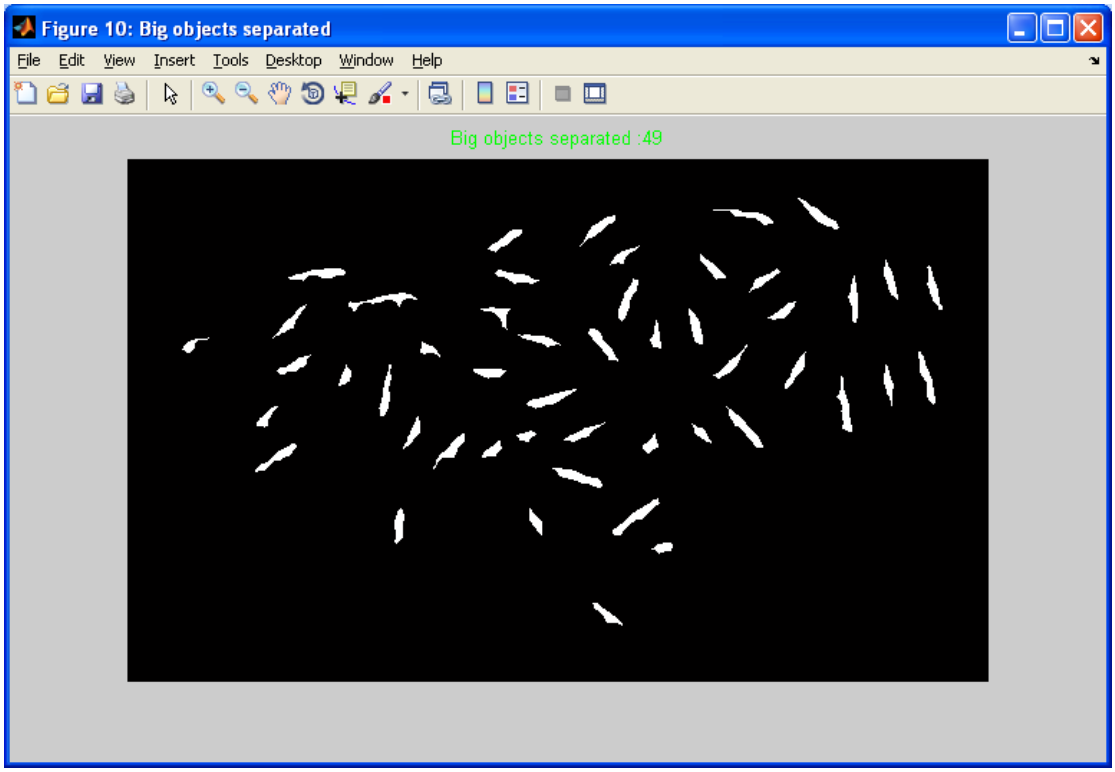
Εικόνα 73: Αντικείμενα μικρού μεγέθους



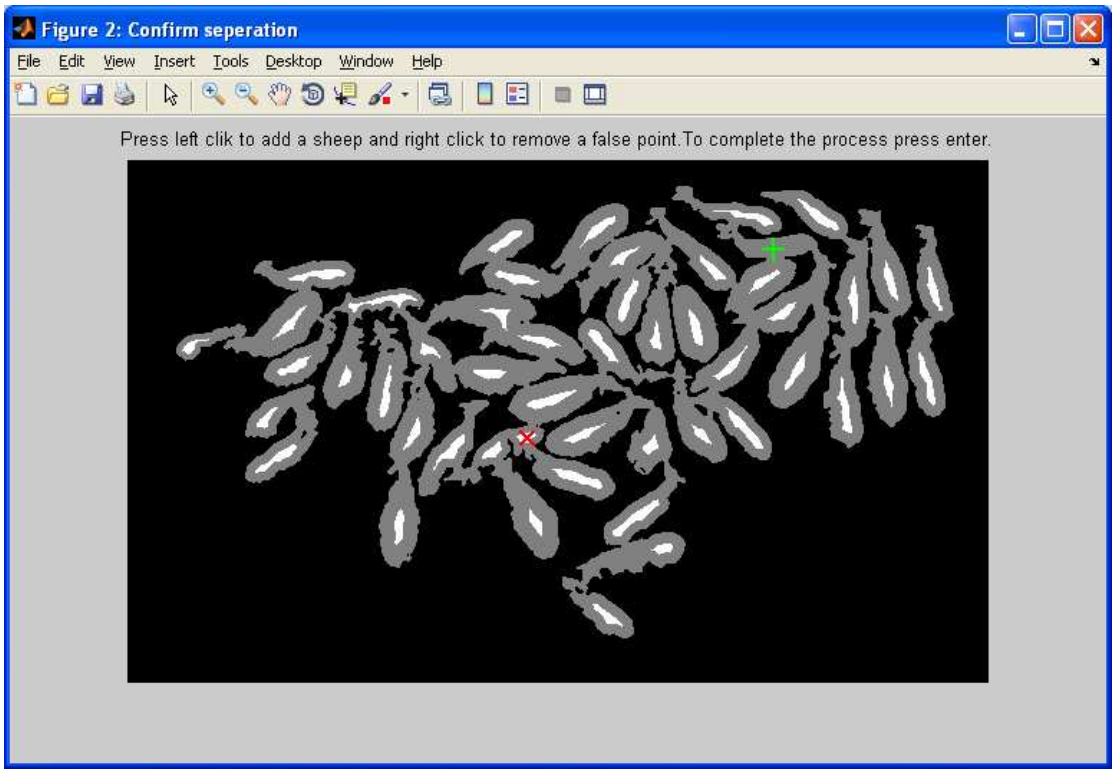
Εικόνα 74: Αντικείμενα μικρού μεγέθους χωρίς τον θόρυβο



Εικόνα 75: Αντικείμενα μεγάλου μεγέθους

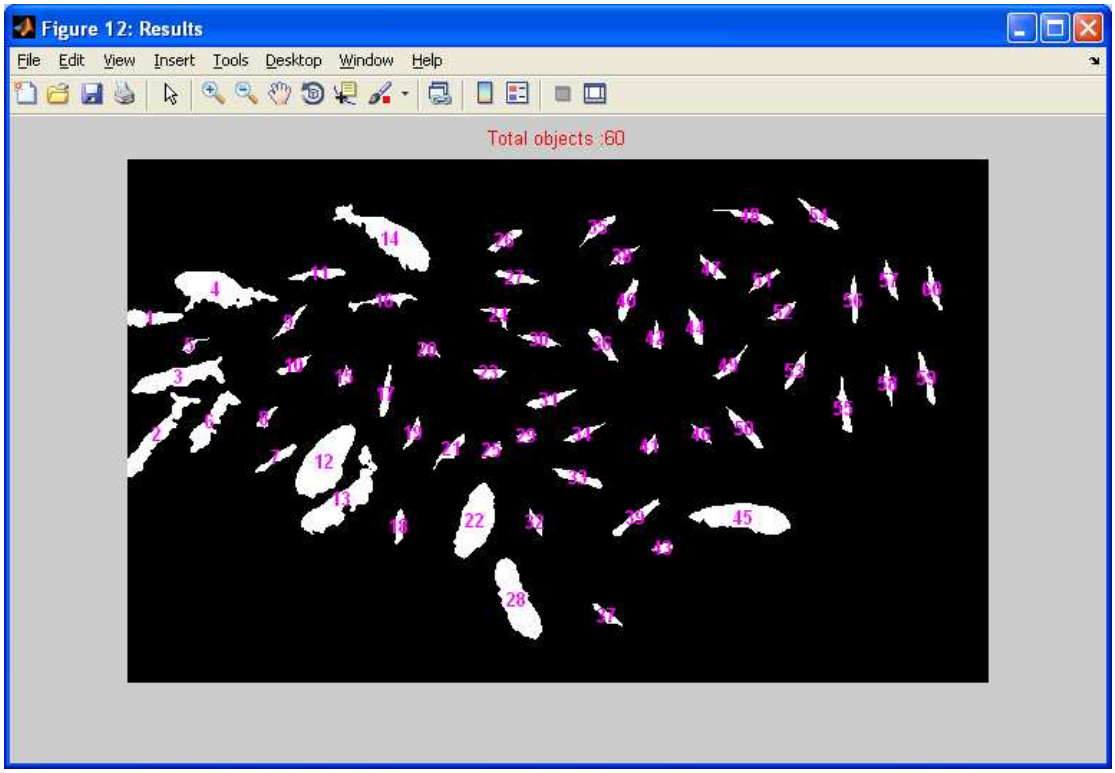


Εικόνα 76: Αντικείμενα μεγάλου μεγέθους διασπασμένα



Εικόνα 77: Επαλήθευση από τον χρήστη ότι ο διαχωρισμός των προβάτων είναι έγκυρος

Στην εικόνα 77 παρατηρούμε ένα κόκκινο (X) σε ένα σημείο που θεωρείται ότι έχει καταμετρηθεί λανθασμένα δύο φορές το ίδιο αντικείμενο και πρέπει να αφαιρεθεί.
Με τον πράσινο σταυρό επισημαίνουμε για να προστεθεί ένα πρόβατο το οποίο δεν καταμετρήθηκε.

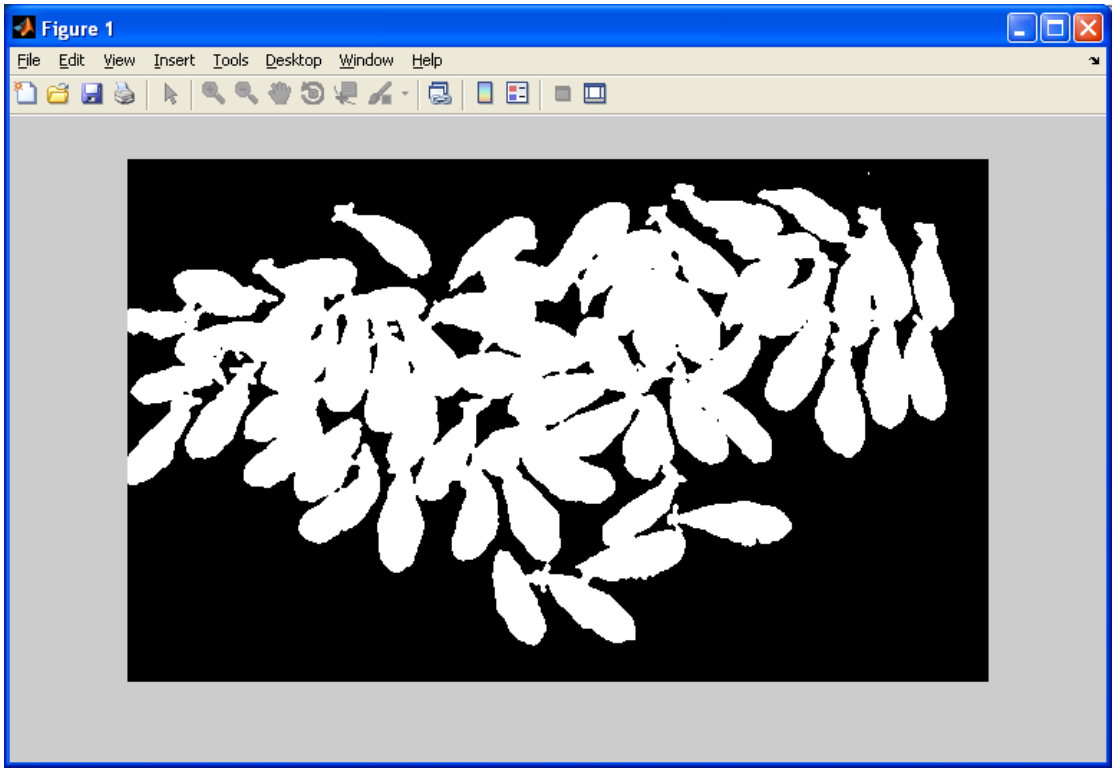


Εικόνα 78: Τελική εικόνα με το σύνολο των προβάτων αριθμημένα

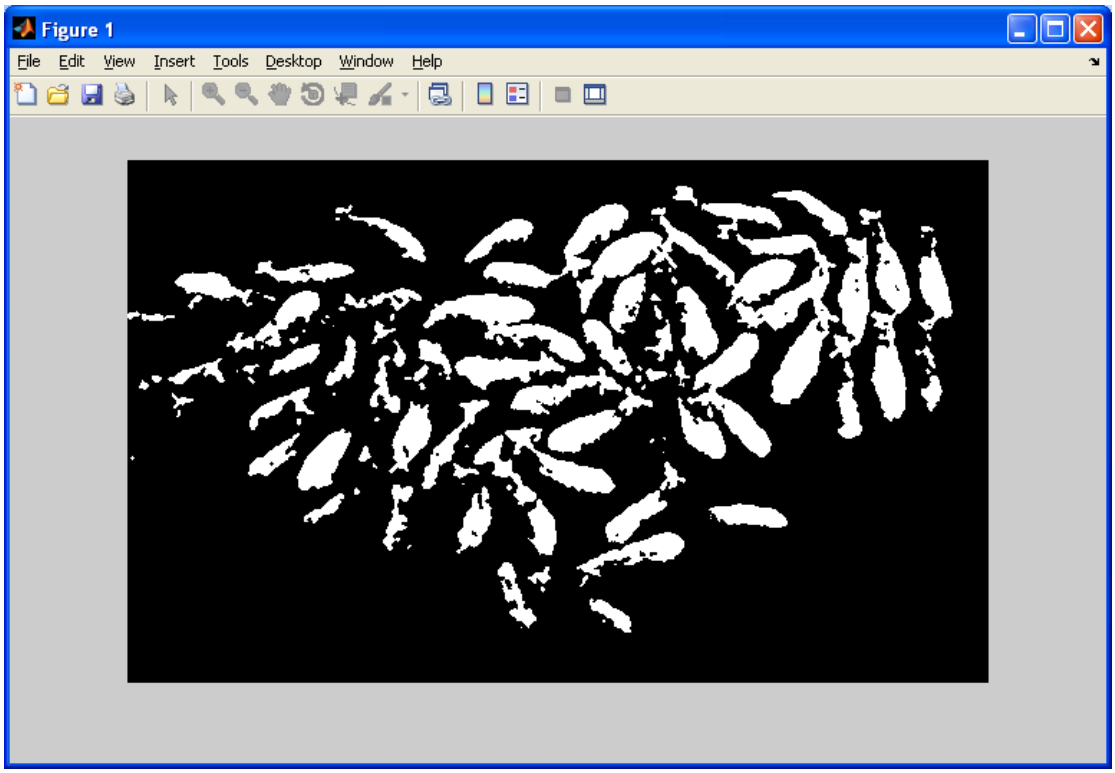
4.6 Προϋποθέσεις λειτουργίας και τυχόν προβλήματα

Στην συγκεκριμένη υλοποίηση ο σκοπός είναι η καταμέτρηση ενός συγκεκριμένου είδους αντικειμένων και το πρόγραμμα έχει παραμέτρους ειδικά για αυτήν την περίπτωση. Κατά την διαδικασία της λήψης της φωτογραφίας ο φωτισμός ήταν κανονικός , τα αντικείμενα που θέλαμε να μετρήσουμε είχαν όλα άσπρο χρώμα και το φόντο ήταν σκουρόχρωμο. Στις εικόνες που φωτογραφίσαμε κάναμε περικοπή ώστε να μην περιλαμβάνονται τα περιμετρικά αντικείμενα που δεν μας αφορούν. Η εφαρμογή που δημιουργήσαμε δεν έχει ανάγκη από εικόνες υψηλής ανάλυσης , ενώ χρησιμοποιήθηκε ψηφιακή φωτογραφική μηχανή ανάλυσης (7 Mpix) έπειτα μειώσαμε το μέγεθος των εικόνων σε ανάλυση 800x600 δηλαδή περίπου (0,5 Mpix) γιατί υπήρχε μεγάλη καθυστέρηση στην επεξεργασία των εικόνων από το Matlab. Για την λήψη των εικόνων χρησιμοποιήσαμε τηλεσκοπικό κοντάρι με βάση για κάμερα προσαρμοσμένη στην άκρη του και οι λήψεις έγιναν από ύψος 12 μέτρων με την μηχανή να βρίσκεται σε κατακόρυφη θέση με τον φακό να βλέπει προς τα κάτω.

Το πρώτο και σημαντικότερο σημείο που πρέπει να προσέξει ο χρήστης και έχει σημασία αν θέλουμε καταμέτρηση με την μικρότερη δυνατή απόκλιση είναι η επιλογή του βέλτιστου επιπέδου κατωφλίωσης (threshold level). Αν είναι πολύ χαμηλό δεν θα είναι οπτικά εμφανής ο διαχωρισμός μεταξύ των προβάτων και στην περίπτωση που ορίσουμε πολύ υψηλό επίπεδο κατωφλίωσης κάποια αντικείμενα μπορεί να εξαλειφθούν από την εικόνα.

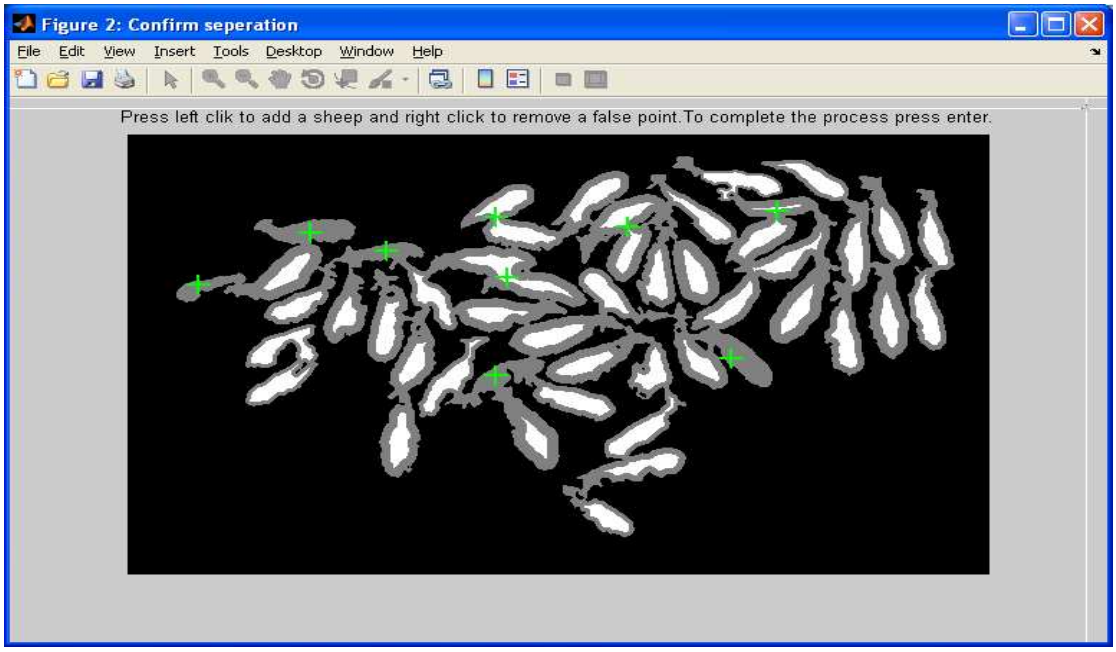


Εικόνα 79: Χαμηλό επίπεδο κατωφλίωσης

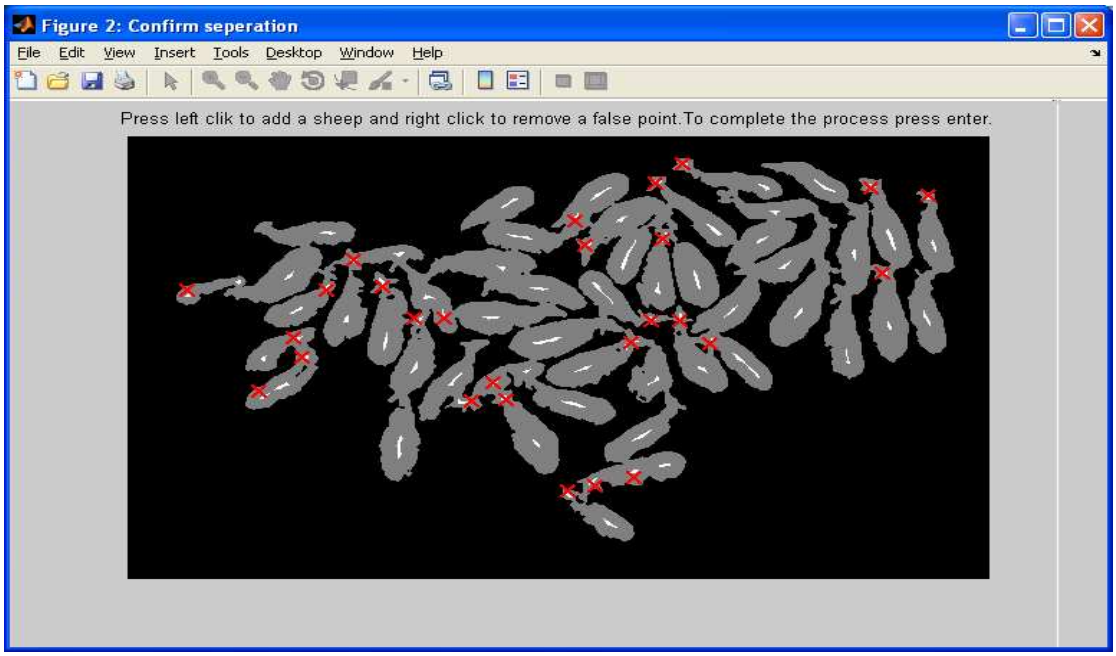


Εικόνα 80: Υψηλό επίπεδο κατωφλίωσης

Δεύτερο σημείο που πρέπει να δώσουμε προσοχή είναι όταν ζητηθεί από τον χρήστη να επιλέξει ένα αντικείμενο μέσου μεγέθους από την εικόνα έτσι ώστε να θέσουμε τα όρια μεγάλου και μικρού αντικειμένου , ο χρήστης πρέπει να παρατηρήσει την εικόνα και να επιλέξει ένα αντικείμενο το οποίο είναι αντιπροσωπευτικό του συνόλου και δεν εφάπτεται με κανένα άλλο αντικείμενο. Τρίτο σημείο που χρειάζεται προσοχή είναι ο ορισμός της τιμής η οποία θα προστίθεται στα τοπικά ελάχιστα , η οποία τιμή αν είναι μεγάλη τα σημεία θα εφάπτονται μεταξύ τους (εικόνα 81) ενώ αν είναι μικρή θα δημιουργηθούνε πολλά ψευδή σημεία.(εικόνα 82). Η τιμή ορίζεται με την κλήση της συνάρτησης imextendedmin(D,3) , η τιμή (3) μας δίνει ικανοποιητικά αποτελέσματα για εικόνες με αντικείμενα του συγκεκριμένου μεγέθους αλλά όταν αλλάζει το μέγεθος της εικόνας ή του αντικειμένου τότε θα πρέπει να πειραματιστούμε με διάφορες τιμές ώστε να επιλέξουμε αυτήν με τη μικρότερη απόκλιση στην μέτρηση.



Εικόνα 81: Τα σημεία εφάπτονται



Εικόνα 82: Δημιουργήθηκαν πολλά ψευδή σημεία

4.7 Συμπεράσματα

Κατά την διάρκεια της παρούσας εργασίας πειραματιστήκαμε με τις δυνατότητες του Matlab στην ψηφιακή επεξεργασία εικόνας , μελετήσαμε αλγορίθμους και μεθόδους για την εξαγωγή δεδομένων από μια εικόνα και συνειδητοποιούμε ότι είναι δυνατό από την στιγμή που η καταμέτρηση αντικειμένων υποστηρίζεται σε επίπεδο συναρτήσεων, τεχνικών αλλά και σε λογισμικό ,υλικό όπως εφαρμόσαμε και αποδείξαμε και μάλιστα με ελάχιστη απόκλιση η οποία μπορεί να γίνει μηδενική με την ελάχιστη επέμβαση του χρήστη. Η χρονοβόρα και κουραστική καταμέτρηση αντικειμένων από ανθρώπινο παράγοντα μπορεί να γίνει παρελθόν με την κατασκευή της κατάλληλης εφαρμογής προσαρμοσμένης στο περιβάλλον που θα χρησιμοποιηθεί.

4.8 Μελλοντικές επεκτάσεις

Βελτιώσεις που μπορούν να γίνουν στην εφαρμογή αφορούν πρώτον τον τρόπο λήψης της εικόνας ο οποίος μπορεί να γίνει με σύγχρονο τηλεκατευθυνόμενο ελικόπτερο (drone) το οποίο φέρει κάμερα με μηχανισμό σταθεροποίησης για καθαρές λήψεις. Επίσης μπορούν να χρησιμοποιηθούν αεροφωτογραφίες παλαιού τύπου από τη βάση του κτηματολογίου για την καταμέτρηση δέντρων σε καλλιέργειες. Όσον αφορά την εφαρμογή της καταμέτρησης θα πρέπει να προσθέσουμε κι άλλες τεχνικές κατάτμησης της εικόνας. Στην παρούσα εργασία και επειδή χρησιμοποιήσαμε εικόνες με άσπρα μόνο πρόβατα η τεχνική που εφαρμόσαμε ήταν η κατωφλίωση με βάση τα επίπεδα του γκρι σε μια εικόνα , αλλά αν θέλουμε την επέκταση της εφαρμογής και την χρήση της σε όλα τα είδη εικόνων τότε θα πρέπει να προσθέσουμε τεχνικές κατάτμησης με βάση το χρώμα και το σχήμα των αντικειμένων. Στην ειδική περίπτωση της καταμέτρησης ζώων η εφαρμογή μπορεί με μικρές τροποποιήσεις να τελειοποιηθεί ώστε να μπορεί να χρησιμοποιηθεί από κάποιον οργανισμό ελέγχου και πληρωμών όπως ο (Ο.Π.Ε.Κ.Ε.Π.Ε) Ελληνικός οργανισμός πληρωμών των κοινοτικών ενισχύσεων. Οι έλεγχοι μέχρι και σήμερα στους κτηνοτρόφους γίνονται με τον απαρχαιωμένο τρόπο της οπτικής καταμέτρησης με το μάτι καθώς τα πρόβατα περνάνε ένα-ένα μπροστά από ομάδα ατόμων .Τα μειονεκτήματα αυτής της μεθόδου είναι :

- Η διαδικασία της μέτρησης προκαλεί στρες και είναι δυσάρεστη και ανθυγιεινή για τα ζώα καθώς στοιβάζονται και αναγκάζονται να περάσουν από στενές πόρτες.
- Δεν υπάρχει κάποιο στοιχείο που να αποδεικνύει ότι έγινε η μέτρηση και ποιος είναι ο ακριβής αριθμός των ζώων.
- Για να γίνει σωστά χρειάζεται το λιγότερο δύο άτομα.
- Είναι χρονοβόρα και κουραστική.

Η μέθοδος που προτείνεται απαιτεί ένα μόνο άτομο το οποίο με την χρήση ενός τηλεκατευθυνόμενου ελικόπτερου θα κάνει λήψεις από αέρος των ζώων που θα βρίσκονται σε περιφραγμένο υπαίθριο χώρο και στην συνέχεια θα περνάει τις εικόνες στην εφαρμογή και θα δέχεται ως έξοδο των αριθμό των ζώων.

Βιβλιογραφία

[1] Lim, Jae S., Two-Dimensional Signal and Image Processing, Englewood Cliffs, NJ, Prentice Hall, 1990, pp. 469-476.
[2] Pratt, William K., Digital Image Processing, New York, John Wiley & Sons, Inc., 1991, p. 634.
[3] Gonzalez, R. C., R. E. Woods, and S. L. Eddins, Digital Image Processing Using MATLAB, New Jersey, Pearson Prentice Hall, 2004.
G. Toussaint, Course Notes: Grids, connectivity and contour tracing (PostScript)
4) T. Pavlidis, Algorithms for Graphics and Image Processing, Computer Science Press, Rockville, Maryland, 1982
5) Mike Alder, Border Tracing (by radial sweep)
6) M. Soss, Proof of correctness of Square Tracing algorithm when both pattern and background are 4-connected
[7] Maurer, Calvin, Rensheng Qi, and Vijay Raghavan, "A Linear Time Algorithm for Computing Exact Euclidean Distance Transforms of Binary Images in Arbitrary Dimensions," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, No. 2, February 2003, pp. 265-270.
[8] Rosenfeld, Azriel and John Pfaltz, "Sequential operations in digital picture processing," Journal of the Association for Computing Machinery, Vol. 13, No. 4, 1966, pp. 471-494.
[9] Paglieroni, David, "Distance Transforms: Properties and Machine Vision Applications," Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing, Vol. 54, No. 1, January 1992, pp. 57-58.
[10] Otsu, N., "A Threshold Selection Method from Gray-Level Histograms," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 9, No. 1, 1979, pp. 62-66.
[11] Soille, P., Morphological Image Analysis: Principles and Applications, Springer-Verlag, 1999, pp. 170-171.

Τον κώδικα για την συνάρτηση thresh_tool που χρησιμοποιήσαμε ως εργαλείο κατωφλίωσης στην εφαρμογή μας τον πήραμε από την επίσημη σελίδα του Matlab στον παρακάτω σύνδεσμο http://www.mathworks.com/matlabcentral/fileexchange/6770-thresholding-tool και δημιουργήθηκε από τον Robert Bemis.