ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

# ΕΡΓΑΣΙΑ ΑΝΑΛΥΣΗ ΕΙΚΟΝΑΣ

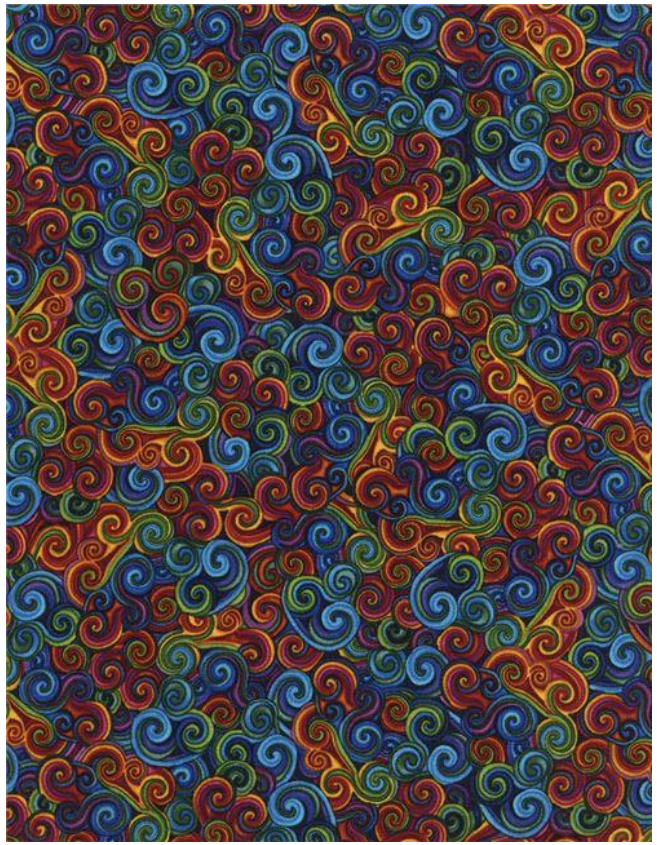
ΜΑΡΟΠΟΥΛΟΣ ΠΑΡΑΣΚΕΥΑΣ Π15086

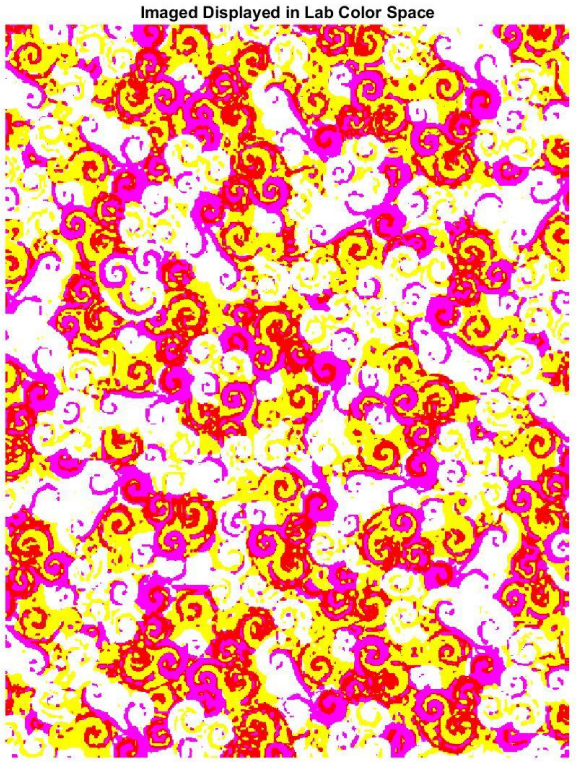
Ακαδημαίκο έτος 2018-2019

# 1. Αναπαράσταση Εικόνας στον Χρωματικό Χώρο Lab

- Ο χρωματικός χώρος Lab είναι ένα σύστημα χρωμάτων 3 αξόνων με διάσταση L για ελαφρότητα και a και b για τις διαστάσεις χρώματος. Η εργασία με το εργαστηριακό χώρο χρώματος περιλαμβάνει όλα τα χρώματα στο φάσμα, καθώς και τα χρώματα εκτός της ανθρώπινης αντίληψης.
- Ο χώρος χρώματος Lab είναι το πιο ακριβές μέσο αναπαραγωγής του χρώματος και είναι ανεξάρτητο από τη συσκευή. Αυτή η ακρίβεια και η φορητότητα το καθιστούν κατάλληλο σε διάφορες βιομηχανίες όπως η εκτύπωση, η αυτοκινητοβιομηχανία, τα κλωστοϋφαντουργικά προϊόντα και τα πλαστικά.
- Αν και ο χώρος χρώματος Lab είναι η πιο ακριβής αναπαράσταση του χρώματος, δεν είναι ο συνηθέστερα χρησιμοποιούμενος. Το χρώμα του εργαστηρίου μετατρέπεται συνήθως σε λιγότερο ακριβείς χρωματικούς χώρους, όπως τα RGB και CYMK, επειδή οι οθόνες υπολογιστών και οι εκτυπωτές χρησιμοποιούν τρία ή τέσσερα χρώματα για να αναπαραστήσουν τις εικόνες.

# Αναπαράσταση Εικόνας στον Χρωματικό Χώρο Lab.





```
= imread("fabric.jpg");
Lab_fabric = rgb2lab(I);

imshow(Lab_fabric)
title("Imaged Displayed in Lab Color Space")
```

# 2.Διακριτοποίηση του Χρωματικού Χώρου Lab με βάση ένα σύνολο συναφών εικόνων εκπαίδευσης.

**Βήμα 1: Αποκτηση εικόνας**

```
fabric = imread("fabric.jpg");
lab_fabric = rgb2lab(fabric);
```

**Βήμα 2: Υπολογίστε τα χρώματα δείγματος σε L*a*b* Χρωματικό χώρο για κάθε περιοχή**

```
load regioncoordinates;
nColors = 6;
sample_regions = false([size(fabric,1) size(fabric,2) nColors]);
for count = 1:nColors
  sample_regions(:,:,count) = roipoly(fabric,region_coordinates(:,1,count), ...
                                      region_coordinates(:,2,count));
end
lab_fabric = rgb2lab(fabric);
a = lab_fabric(:,:,2);
b = lab_fabric(:,:,3);
color_markers = zeros([nColors, 2]);
for count = 1:nColors
  color_markers(count,1) = mean2(a(sample_regions(:,:,count)));
  color_markers(count,2) = mean2(b(sample_regions(:,:,count)));
end
fprintf('[%0.3f,%0.3f] \n',color_markers(2,1),color_markers(2,2));
```

**Βήμα 3: Ταξινόμηση κάθε εικονοστοιχείου χρησιμοποιώντας τον κανόνα πλησιέστερου γείτονα**

--Κάθε δείκτης χρώματος έχει τώρα μια τιμή 'a *' και 'b *'. Μπορείτε να ταξινομήσετε κάθε εικονοστοιχείο στην εικόνα lab_fabric υπολογίζοντας την ευκλείδεια απόσταση μεταξύ αυτού του εικονοστοιχείου και κάθε δείκτη χρώματος. Η μικρότερη απόσταση θα σας πει ότι το εικονοστοιχείο ταιριάζει περισσότερο με αυτό το δείκτη χρώματος. Για παράδειγμα, εάν η απόσταση μεταξύ ενός εικονοστοιχείου και του κόκκινου δείκτη χρώματος είναι η μικρότερη, τότε το εικονοστοιχείο θα επισημαίνεται ως ένα κόκκινο εικονοστοιχείο. Δημιουργώ έναν πίνακα που περιέχει τις έγχρωμες ετικέτες σας, δηλ. 0 = φόντο, 1 = κόκκινο, 2 = πράσινο, 3 = μοβ, 4 = ματζέντα και 5 = κιτρινο

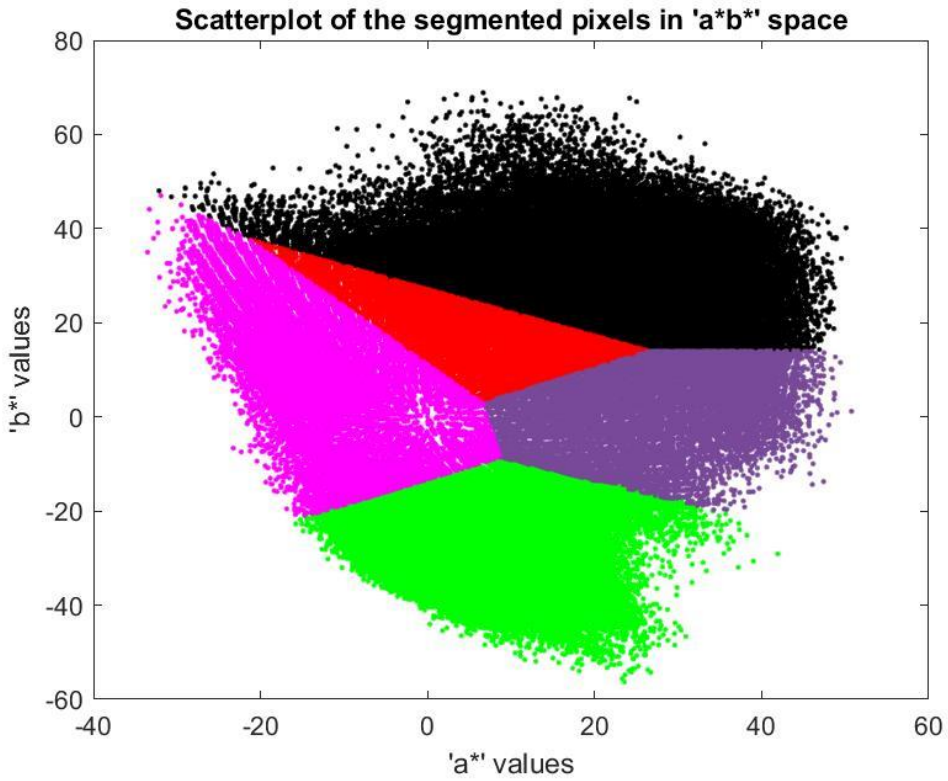
```
color_labels = 0:nColors-1;


a = double(a);
b = double(b);
distance = zeros([size(a), nColors]);


for count = 1:nColors
  distance(:,:,count) = ( (a - color_markers(count,1)).^2 + ...
                        (b - color_markers(count,2)).^2 ).^0.5;
end


[~,label] = min(distance,[],3);
label = color_labels(label);
clear distance;
```

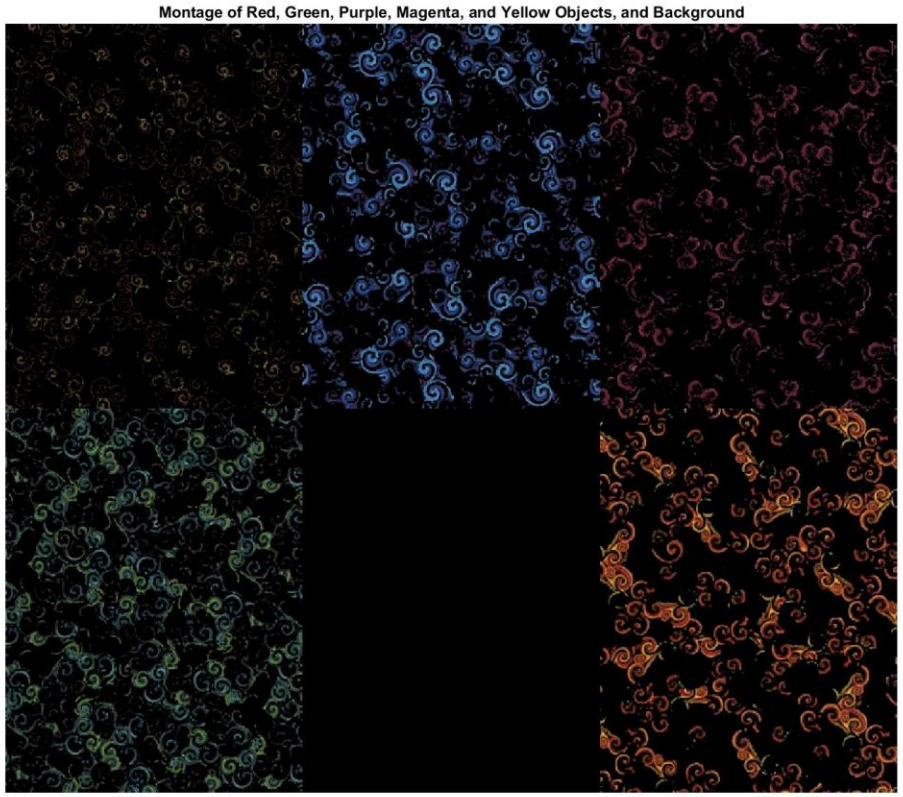
## Βήμα 4: Εμφάνιση αποτελεσμάτων της πλησιέστερης ταξινόμησης γειτόνων

```
rgb_label = repmat(label,[1 1 3]);
segmented_images = zeros([size(fabric), nColors],'uint8');

for count = 1:nColors
  color = fabric;
  color(rgb_label ~= color_labels(count)) = 0;
  segmented_images(:,:,:,count) = color;
end

montage({segmented_images(:,:,:,2),segmented_images(:,:,:,3) ...
    segmented_images(:,:,:,4),segmented_images(:,:,:,5) ...
    segmented_images(:,:,:,6),segmented_images(:,:,:,1)});
title("Montage of Red, Green, Purple, Magenta, and Yellow Objects, and Background")
```

## Βήμα 5: Εμφανίστε τις τιμές 'a *' και 'b *' των ετικετών χρωμάτων

```
purple = [119/255 73/255 152/255];
plot_labels = {'k', 'r', 'g', purple, 'm', 'y'};

figure
for count = 1:nColors
  plot(a(label==count-1),b(label==count-1),'.','MarkerEdgeColor', ...
      plot_labels{count}, 'MarkerFaceColor', plot_labels{count});
  hold on;
end

title('Scatterplot of the segmented pixels in ''a*b*'' space');
xlabel('''a*'' values');
ylabel('''b*'' values');
```


Scatterplot of the segmented pixels in 'a*b*' space
'b*' values
'a*' values
80
60
40
20
0
-20
-40
-60
-40
-20
0
20
40
60

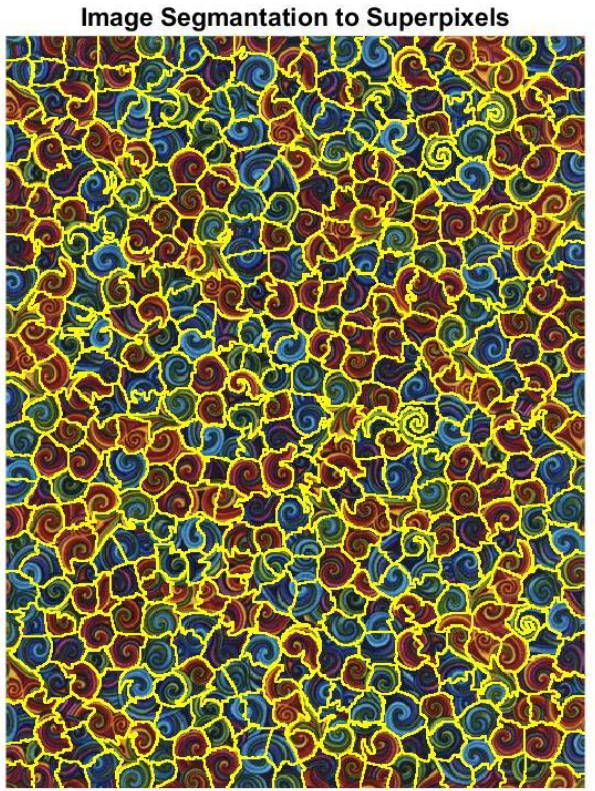

Montage of Red, Green, Purple, Magenta, and Yellow Objects, and Background

# 3. Κατάτμηση Εικόνας σε Superpixels σύμφωνα με τον αλγόριθμο SLIC

Θα χρησιμοποιήσουμε την συναρτηση superpixels η οποια χρησιμοποιεί τον αλγοριθμο SLIC(simple linear iterative clastering ή αλγόριθμο απλής γραμμικής επαναληπτικής ομαδοποίησης).Ο οποίος ομαδοποιει τα pixel σε περιοχες με παρομοιες τιμές.Η χρήση αυτών των περιοχών στις λειτουργίες επεξεργασίας εικόνας, όπως η κατάτμηση, μπορεί να μειώσει την πολυπλοκότητα αυτών των λειτουργιών.

[L, NumLabels] = superpixels (A, N) υπολογίζει τα superpixels της κλίμακας του γκρι 2-D ή της εικόνας RGB A. Το N ορίζει τον αριθμό των superpixel που θέλετε να δημιουργήσετε. Η συνάρτηση επιστρέφει το L, μια μήτρα ετικέτας τύπου double και τα NumLabels, τον πραγματικό αριθμό των superpixel που υπολογίστηκαν.

```
fabric = imread("fabric.jpg");
[L,N]  = superpixels(fabric, 500);

figure

BW = boundarymask(L);
imshow(imoverlay(fabric,BW,'yellow'),'InitialMa
gnification',67)
title("Image Segmantation to Superpixels")
```

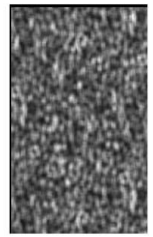
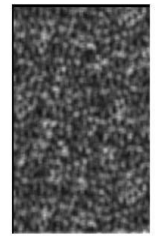
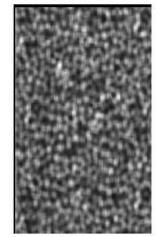
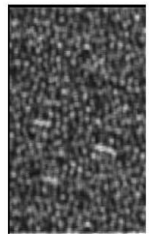
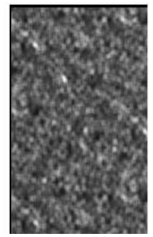
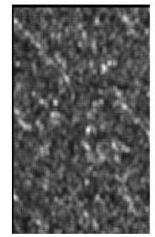
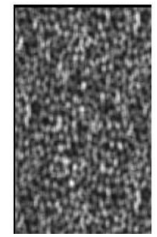
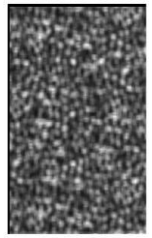
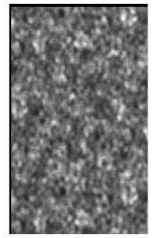
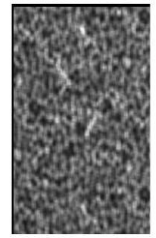
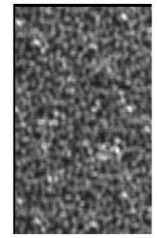
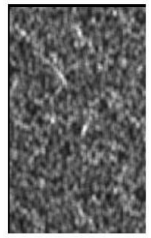
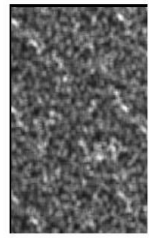
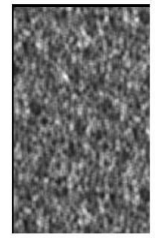
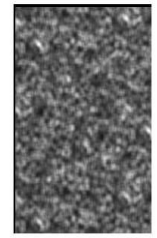


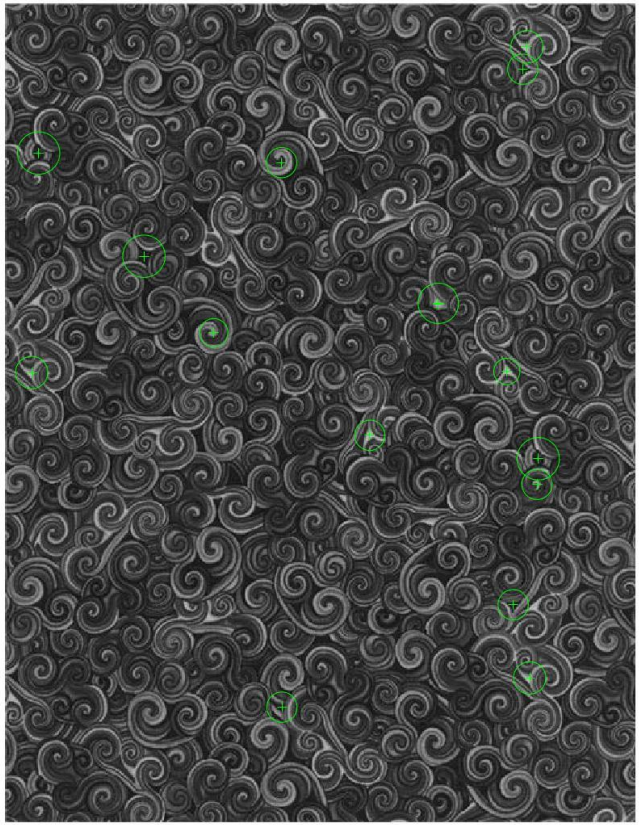
# 4. Εξαγωγή Χαρακτηριστικών Υφής .

Για να υπολογίσουμε την ενέργεια της υφής ακολουθούμε τον αλγόριθμο του Laws και θα χρησιμοποιήσοουμε ένα σύνολο συνελεκτικών μασκών. Η ενέργεια της υφής θα αναπαριστάνεται με ένα διάνυσμα αριθμών για κάθε pixel. Το διάνυσμα L5 είναι για το κεντραρισμένο τοπικό μέσο, το E5 για την ανίχνευση ακμών, το S5 για την ανίχνευση κηλίδων και το R5 για την ανίχνευση κυματισμών. Η μεταβλητή premim παίρνει τις τιμές της εικόνας Im. Στην συνέχεια εκτελείται μια επανάληψη, για το j από 1 έως το πλάτος της εικόνας και με βήμα το πλάτος του παραθύρου. Γίνεται κάθε φορά έλεγχος για το τρέχων παράθυρο έτσι ώστε να μην υπερβεί το πλάτος της εικόνας. Στην nested επανάληψη για i= 1 εώς το ύψος της εικόνας. Αν το παράθυρο είναι μέσα στην εικόνα, τότε σε κάθε επανάληψη αφαιρείται από το κάθε σημείο που βρίσκεται το παράθυρο, ο τοπικός μέσος.

Το τελικό αποτέλεσμα είναι ο πίνακας με διαστάσεις όσο το άθροισμα της κάθε διάστασης κάθε διανύσματος μείον 1 (Xa + Xb -1, Ya +Yb -1 ).

Η συνάρτηση Ek επιστρέφει κάθε φορά τον χάρτη ενέργειας υφής.



points = detectSURFFeatures(I)

επιστρέφει ένα αντικείμενο SURFPoints, σημεία που περιέχουν πληροφορίες σχετικά με τις ιδιότητες SURF που ανιχνεύονται στην εικόνα εισόδου 2-D της κλίμακας του γκρι I. Η λειτουργία εντοπισμούSURFFeatures εφαρμόζει τον αλγόριθμο σφιχτού χαρακτηριστικού (SURF) για την εύρεση χαρακτηριστικών 'σταγώνων'

# 5.Εκμάθηση Τοπικών Μοντέλων Πρόγνωσης Χρώματος με Χρήση Ταξινομητών SVM

Αρχικη εικονα

Ορισμος ROI

Εφαρμογη αλγοριθμου κοπης

Δημιουργια masked image

# 6. Εκτίμηση Χρωματικού Περιεχομένου Ασπρόμαυρης Εικόνας με Χρήση Αλγορίθμων Κοπής Γραφημάτων.