Documentation

1. Στόχος

Στόχος της εργασίας είναι ο **αυτόματος χρωματισμός** μιας ασπρόμαυρης εικόνας με βάση ένα **σύνολο έγχρωμων εικόνων εκπαίδευσης**, ακολουθώντας τα παρακάτω βήματα:

1. Αναπαράσταση Εικόνας στον Χρωματικό Χώρο Lab
2. Διακριτοποίηση του Χρωματικού Χώρου Lab με βάση ένα σύνολο συναφών εικόνων εκπαίδευσης.
3. Κατάτμηση Εικόνας σε Superpixels σύμφωνα με τον αλγόριθμο SLIC.
4. Εξαγωγή Χαρακτηριστικών Υφής (SURF Features & Gabor Features) ανά Super Pixel.
5. Εκμάθηση Τοπικών Μοντέλων Πρόγνωσης Χρώματος με Χρήση Ταξινομητών SVM
6. Εκτίμηση Χρωματικού Περιεχομένου Ασπρόμαυρης Εικόνας με Χρήση Αλγορίθμων Κοπής Γραφημάτων.

2. Εκτέλεση

i. Αναπαράσταση στον χρωματικό χώρο Lab

Script : import_image.py

Βιβλιοθήκες: scikit-image, numpy

Λειτουργία:

- Διαβάζει τις έγχρωμες εικόνες εκπαίδευσης από data/train_*.jpg.
- Τις κανονικοποιεί σε [0,1].
- Τις μετατρέπει σε Lab με color.rgb2lab(...).
- Αποθηκεύει ξεχωριστά τα κανάλια L, a, b σε λίστες για να τα δουν τα επόμενα βήματα.
- Εκτυπώνει εύρος τιμών για κάθε κανάλι

Screenshot:

```
PS C:\Users\punis\Desktop\py_img> python .\import_image.py
Φορτώθηκε data/train_1.jpg με σχήμα (371, 508, 3)
    Εύρος τιμών L: 0.00 έως 100.00
    Εύρος τιμών a: -86.18 έως 76.07
    Εύρος τιμών b: -73.63 έως 91.76
Φορτώθηκε data/train_2.jpg με σχήμα (2576, 3865, 3)
    Εύρος τιμών L: 0.00 έως 99.93
    Εύρος τιμών a: -40.56 έως 52.09
    Εύρος τιμών b: -82.67 έως 75.78
Φορτώθηκε data/train_3.jpg με σχήμα (669, 1000, 3)
    Εύρος τιμών L: 0.00 έως 100.00
    Εύρος τιμών a: -59.19 έως 80.32
    Εύρος τιμών b: -73.07 έως 93.65
Φορτώθηκε data/train_4.jpg με σχήμα (386, 686, 3)
    Εύρος τιμών L: 0.00 έως 100.00
    Εύρος τιμών a: -67.60 έως 80.44
    Εύρος τιμών b: -66.02 έως 91.48
Φορτώθηκε data/train_5.jpg με σχήμα (702, 735, 3)
    Εύρος τιμών L: 0.77 έως 98.15
    Εύρος τιμών a: -54.86 έως 78.94
    Εύρος τιμών b: -17.19 έως 92.85
Φορτώθηκε data/train_6.jpg με σχήμα (300, 300, 3)
    Εύρος τιμών L: 1.98 έως 92.42
    Εύρος τιμών a: -44.49 έως 43.36
    Εύρος τιμών b: 2.94 έως 65.25
Φορτώθηκε data/train_7.jpg με σχήμα (408, 612, 3)
    Εύρος τιμών L: 0.20 έως 96.05
    Εύρος τιμών a: -44.85 έως 77.17
    Εύρος τιμών b: -35.19 έως 92.03
Φορτώθηκε data/train_8.jpg με σχήμα (1024, 683, 3)
    Εύρος τιμών L: 0.00 έως 99.95
    Εύρος τιμών a: -54.82 έως 63.50
    Εύρος τιμών b: -38.04 έως 88.60
Συνολικές εικόνες εκπαίδευσης: 8
```

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

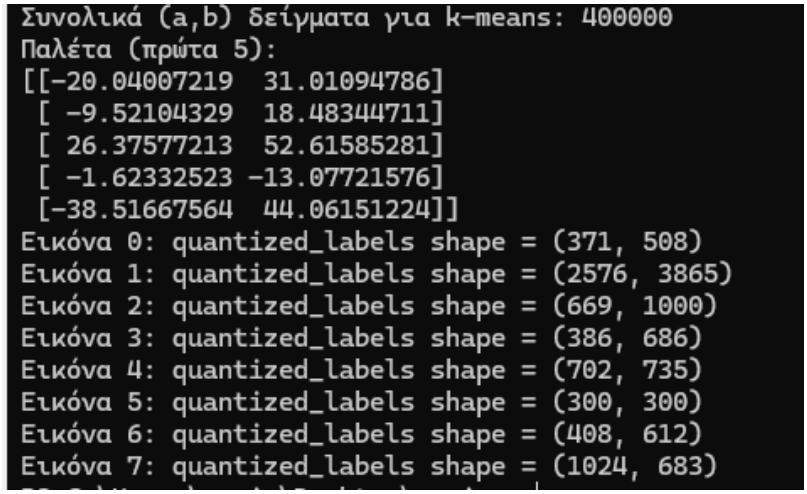
Script: lab.py

Βιβλιοθήκες: joblib, numpy, sklearn

Λειτουργία:

- Συγκεντρώνει όλα τα (a,b) από τις training εικόνες.
- Κάνει MiniBatchKMeans με 64 κλάσεις (N_COLOR_CLASSES = 64).
- Αποθηκεύει το μοντέλο σε artifacts/kmeans_palette.joblib.
- Για κάθε training εικόνα εξάγει έναν χάρτη «σε ποια κλάση χρώματος ανήκει κάθε pixel» και τα γράφει στο artifacts/quantized_labels.npz.

Screenshot:



iii. Κατάτμηση Εικόνας σε Superpixels σύμφωνα με τον αλγόριθμο SLIC

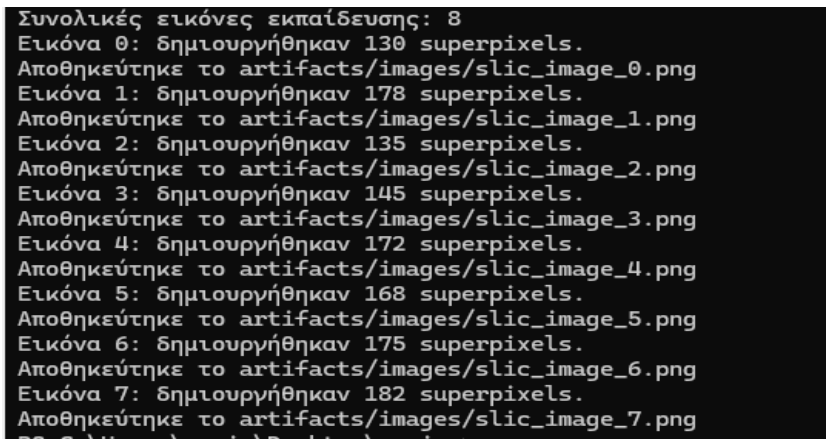
Script: slic.py

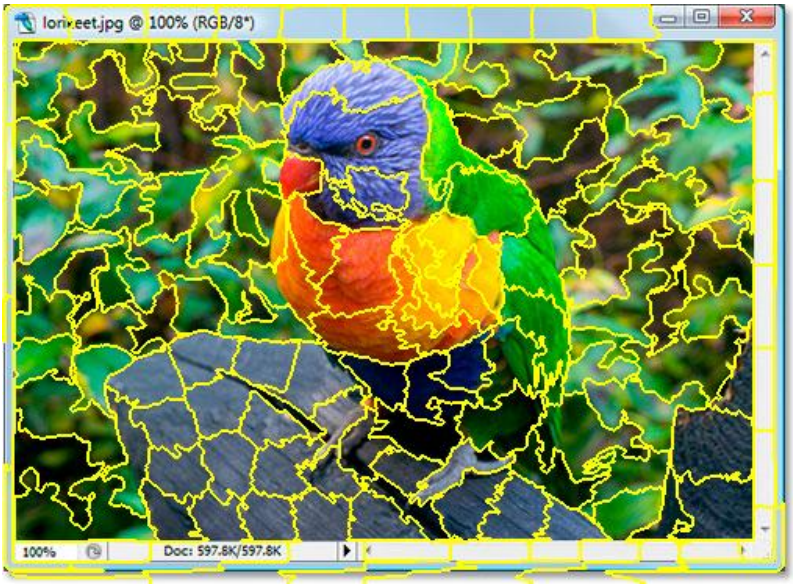
Βιβλιοθήκες: matplotlib,numpy, sklearn, skimage

Λειτουργία:

- Για κάθε training εικόνα τρέχει skimage.segmentation.slic(...) με 200 segments.
- Αποθηκεύει όλα τα segments σε 1 αρχείο artifacts/slic_segments.npz.
- Εξάγει και εικόνες με τα όρια (artifacts/images/slic_image_i.png).

Screenshot





iv. Εξαγωγή Χαρακτηριστικών Υφής (SURF Features & Gabor Features) ανά Super Pixel.

Script: gabor_surf.py

Βιβλιοθήκες: scipy,numpy, sklearn, skimage, cv2

Λειτουργία:

- Φορτώνει: training Labs, SLIC segments, και τους quantized χάρτες (ώστε να ξέρει ποια κλάση χρώματος αντιστοιχεί σε ΚΑΘΕ superpixel).
- Για κάθε superpixel υπολογίζει:
  - μέση και τυπική απόκλιση του L,
  - μέσο όρο των αποκρίσεων από 8 Gabor filters (4 γωνίες × 2 συχνότητες × 2 (real, imag)),
  - τοπικό descriptor: πρώτα προσπαθεί SURF και, αν δεν υπάρχει, κάνει fallback σε ORB,
  - τοποθεσία (x,y).
- Δημιουργεί ένα μεγάλο X_all και ένα y_all και τα αποθηκεύει στο artifacts/training_features.npz.

Screenshot:

```
Συνολικές εικόνες εκπαίδευσης: 8
Συνολικά (a,b) δείγματα για k-means: 400000
Παλέτα (πρώτα 5):
[[-16.7326785   41.67982981]
 [ -0.34277214  16.21318101]
 [ 36.31725724  67.65136393]
 [ -6.02604059  -6.81077174]
 [-45.4400091   73.42672726]]
Εικόνα 0: quantized_labels shape = (371, 508)
Εικόνα 1: quantized_labels shape = (2576, 3865)
Εικόνα 2: quantized_labels shape = (669, 1000)
Εικόνα 3: quantized_labels shape = (386, 686)
Εικόνα 4: quantized_labels shape = (702, 735)
Εικόνα 5: quantized_labels shape = (300, 300)
Εικόνα 6: quantized_labels shape = (408, 612)
Εικόνα 7: quantized_labels shape = (1024, 683)
Θα χρησιμοποιηθεί τοπικός descriptor: SURF
Feature matrix X shape: (1285, 84)
Label vector y shape: (1285,)
PS C:\Users\punis\Desktop\py_img>
```

v. Εκμάθηση μοντέλου SVM

Script: svm_train.py

Βιβλιοθήκες: joblib,numpy, sklearn

Λειτουργία:

- Φορτώνει τα features που έφτιαξε το προηγούμενο βήμα.
- Δημιουργεί pipeline StandardScaler → SVC(rbf) με probability=True και class_weight="balanced".
- Εκπαιδεύει και αποθηκεύει στο artifacts/svm_colorizer.joblib.
- Εκτυπώνει και ένα μικρό δείγμα πιθανότητων.

Screenshot:

```
PS C:\Users\punis\Desktop\py_img> python .\svm_train.py
Εκκίνηση...
X_train shape = (1285, 84)
y_train shape = (1285,)
Παλέτα με 64 κλάσεις.
Το SVM εκπαιδεύτηκε.
Probabilities sample shape: (5, 64)
Sample 0:
  Class 0: 0.0339
  Class 1: 0.0123
  Class 2: 0.0111
  Class 3: 0.0266
  Class 4: 0.0064
  Class 5: 0.0075
  Class 6: 0.0268
  Class 7: 0.0029
  Class 8: 0.0095
  Class 9: 0.0140
  Class 10: 0.0044
  Class 11: 0.0197
  Class 12: 0.0027
  Class 13: 0.0009
  Class 14: 0.0086
  Class 15: 0.0077
  Class 16: 0.0329
  Class 17: 0.0129
  Class 18: 0.0012
  Class 19: 0.0109
  Class 20: 0.0116
  Class 21: 0.0233
  Class 22: 0.0020
  Class 23: 0.0084
  Class 24: 0.0312
  Class 25: 0.0337
  Class 26: 0.0274
  Class 27: 0.0041
  Class 28: 0.0028
  Class 29: 0.0156
  Class 30: 0.0114
  Class 31: 0.0042
```

vi. Εκτίμηση Χρωματικού Περιεχομένου Ασπρόμαυρης Εικόνας με Χρήση Αλγορίθμων Κοπής Γραφημάτων.

Script: predict_color.py

Βιβλιοθήκες: joblib,numpy, matplotlib, gco, skimage

Λειτουργία:

- Διαβάζει μια ασπρόμαυρη εικόνα, τη μετατρέπει στον χρωματικό χώρο Lab και στην συνέχεια την κατατέμνει μέσω του αλγόριθμου SLIC
- Εξετάζει κάθε μία από αυτές τις περιοχές (τα superpixels) ξεχωριστά. Περνάει τα features από το SVM και υπολογίζει τις πιθανότητες για το τι «βλέπει» σε κάθε περιοχή.
- Δημιουργεί έναν χάρτη (RAG) και δείχνει ποιες περιοχές είναι δίπλα-δίπλα και καθορίζει πόσο «στενά» σχετίζονται μεταξύ τους.
- Μέσω gco.cut_general_graph βελτιστοποιεί, χρησιμοποιώντας τις παραπάνω πιθανότητές και σχέσεις γειτνίασης, την απόφαση.
- Ξαναχτίζει την εικόνα σε Lab και την σώζει ως colored_result.png.

Screenshot:

```
θα χρησιμοποιηθεί τοπικός descriptor: SURF
Feature matrix X shape: (1285, 84)
Label vector y shape: (1285,)
--- Script Χρωματισμού (Πρόβλεψη) ---
Test εικόνα: 513x607
Superpixels test: 180
X_test shape = (180, 84)
Προβλέψεις (με γέμισμα): (180, 64)
Edges for graph cut: (506, 2)
Αποθήκευση εικόνας μόνο με SVM...
C:\Users\punis\Desktop\py_img\predict_color.py:106: UserWarning: Conversion from CIE-LAB, via XYZ to sRGB color space resulted in 9145 negative Z values that have been clipped to zero
  svm_rgb = color.lab2rgb(svm_lab)
Εκτέλεση Graph Cut...
Graph cut ολοκληρώθηκε.
C:\Users\punis\Desktop\py_img\predict_color.py:133: UserWarning: Conversion from CIE-LAB, via XYZ to sRGB color space resulted in 6393 negative Z values that have been clipped to zero
  final_rgb = color.lab2rgb(final_lab)
```

Εικόνα που χρησιμοποιήθηκε

1η και 2η προσπάθεια.

Debug

3. Αποτελέσματα και Συμπεράσματα

Όπως φαίνεται και στις εικόνες τα 2 τελικά αποτελέσματα (διαφορετικές ρυθμίσεις το καθένα) δεν χρωματίστηκαν σωστά! Αντιθέτως είναι σχεδόν ομοιόμορφά με ένα χρώμα.

Θεωρώ πως αυτό συνέβη για τους παρακάτω λόγους:

1. Το σύστημα δεν είχε αρκετά παραδείγματα για να μάθει. Τα 1285 δείγματα (superpixels) ήταν πολύ λίγα για να μπορέσει να ξεχωρίσει 64 διαφορετικές κατηγορίες.
2. Η απόκλιση μεταξύ των εικόνων εκπαίδευσης και της εικόνας δοκιμής.
3. Ίσως το graph cut να ομαλοποίησε κάθε λεπτομέρεια γιατί στο debug του svm βλέπουμε ότι δεν υπάρχει τόσο μεγάλη ομοιομορφία
4. Ίσως κάποιο χρώμα να υπερκάλυψε τα υπόλοιπα, αυτό σχετίζεται με το ότι τα παραδείγματα που πέρασα είχαν μεγάλη απόκλιση στα pixel
5. Βλέπουμε το warning *negative Z values clipped to zero,* αυτό να μας ενημερώνει ότι κάποιο ab κόπηκε με αποτέλεσμα να έχουμε μεγαλύτερη μείωση στις χρωματικές διαφορές.

Νέες ρυθμίσεις:

- SUPERPIXELS από 200 σε 550 => περισσότερα superpixels για μεγαλύτερη λεπτομέρεια
- LAMBDA_SMOOTH από 20 σε 4 => πιο αδύναμη εξομάλυνση
- EDGE_WEIGHT_SCALE από 200 σε 40 => λιγότερο «επιθετικά» edges
- N_COLOR_CLASSES από 64 σε 16 => λιγότερες κλάσεις
- Νέες εικόνες και αλλαγή εικόνας παραδείγματος => Βγήξαν εκτός οι φωτογραφίες 3,4,7 λόγω πολλών χρωμάτων, μεγάλων L , πολλών παρεμφερή δειγμάτων.

Αποτελέσματα:

Φαίνετια πως με τα μικρά tweaks στις ρυθμίσες και στα δεδομένα μας ο αλγόριθμός πάει λίγο πιο σωστά!

Παράδειγμα/τελικό αποτέσμα/svm model

Η διαδικασία υλοποίησης καθώς και οι προηγούμενες ρυθμίσεις φαίνονται στο https://github.com/mixaslfc/py_img