Αναφορά 3ης εργασίας

Ψηφιακή Επεξεργασία Εικόνας

Κωνσταντίνος Λίτσιος, 10047

ΤΗΜΜΥ ΑΠΘ, Ιούνιος 25

Αρχεία κώδικα:

image_to_graph.py: Υλοποιεί τη συνάρτηση image_to_graph, η οποία δέχεται την εικόνα εισόδου και υπολογίζει έναν πίνακα ομοιότητας της φωτεινότητας μεταξύ όλων των pixel της. Η φωτεινότητα ενός pixel υπολογίζεται ως 0.299*red + 0.587*green + 0.114*blue με βάση το πρότυπο ITU-R BT.601 (λαμβάνει υπόψη τη σχετική ευαισθησία του ανθρώπινου ματιού που έχει παρατηρηθεί στα χρώματα κόκκινο πράσινο μπλε), ενώ η ομοιότητα των pixel i, j ως exp(-|φωτεινότητα(i) – φωτεινότητα(j)|)

spectral_clustering.py: Δέχεται έναν πίνακα ομοιότητας και έναν αριθμό k και υπολογίζει αρχικά το Λαπλασιανό πίνακα από τον πίνακα ομοιότητας, έπειτα τα k μικρότερα ιδιοδιανύσματα του Λαπλασιανού, και τέλος την ετικέτα που αντιστοιχεί σε κάθε pixel, όπως υπολογίστηκε από τον αλγόριθμο KMeans για k διαφορετικές ετικέτες. Επιστρέφει τον πίνακα με τις τιμές από τις ετικέτες.

demo1.py: Φορτώνει τον πίνακα ομοιότητας d1a από το δεδομένο αρχείο dip_hw_3.mat και δείχνει τις ετικέτες από τα 12 pixel που τους αναθέτει η συνάρτηση spectral_clustering για k = 2, 3, 4

demo2.py: Φορτώνει τις εικόνες d2a και d2b, υπολογίζει τους πίνακες-γράφους της φωτεινότητας των pixel τους, και έπειτα ομαδοποιεί σε σύνολα (clusters) τα pixel τους για αριθμό συνόλων k = 2, 3, 4. Προβάλλει τις αρχικές και τις επεξεργασμένες εικόνες.

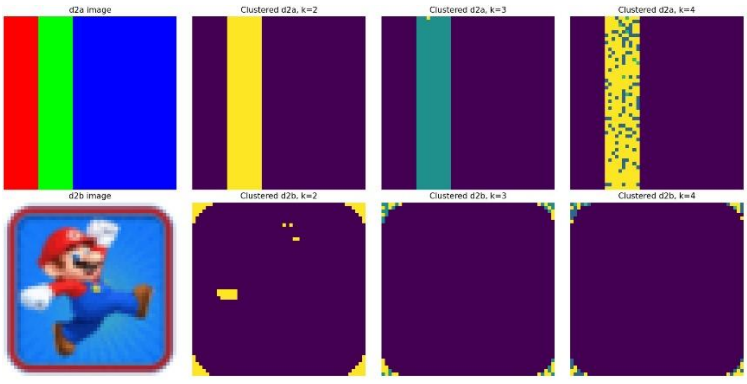
Το 3ο μέρος δεν υλοποιήθηκε

Αποτελέσματα:

demo1:

```
Clustering d1a for k = 2:  [1 1 1 1 0 0 0 0 0 0 0 0]
Clustering d1a for k = 3:  [2 2 2 2 0 0 0 0 1 1 1 1]
Clustering d1a for k = 4:  [2 2 2 2 0 0 3 3 1 1 1 1]
```

demo2:



Παρατηρήσεις: Όπως φαίνεται παραπάνω είναι σε μικρό βαθμό αποτελεσματικός ο αλγόριθμος στο συγκεκριμένο παράδειγμα λόγω τυχαιότητας (στην αρχικοποίηση) του kmeans και της μεγάλης απόστασης της φωτεινότητας των λευκών από τα υπόλοιπα pixel