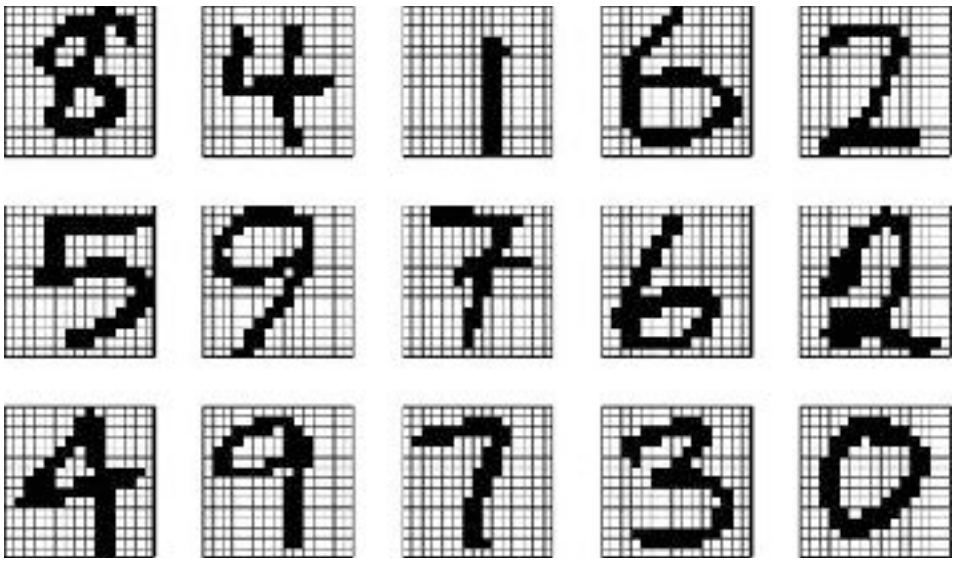
*ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ*
*ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ*

# Προπαρασκευή 1ης Εργαστηριακής Άσκησης

## Οπτική Αναγνώριση Ψηφίων

## Μάθημα : Αναγνώριση Προτύπων



## Ροή Σ

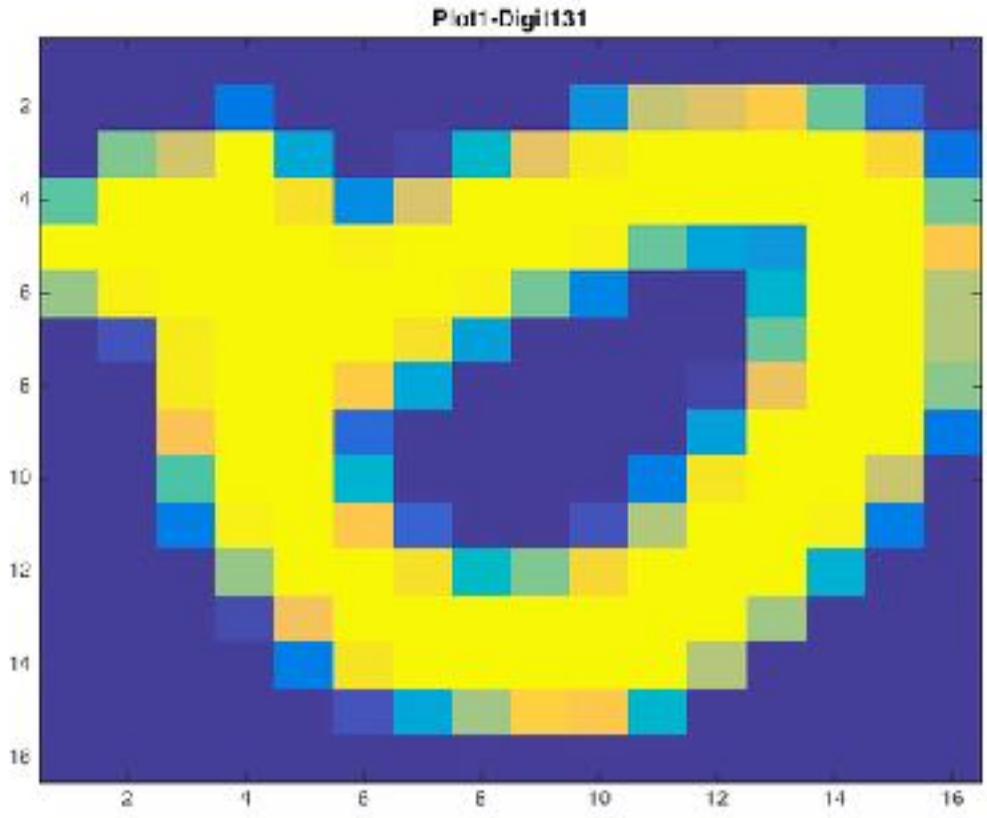
Συνεργάτες :

- Βαβουλιώτης Γεώργιος ( **Α.Μ. : 03112083** )
- Σταυρακάκης Δημήτριος ( **Α.Μ. : 03112017** )

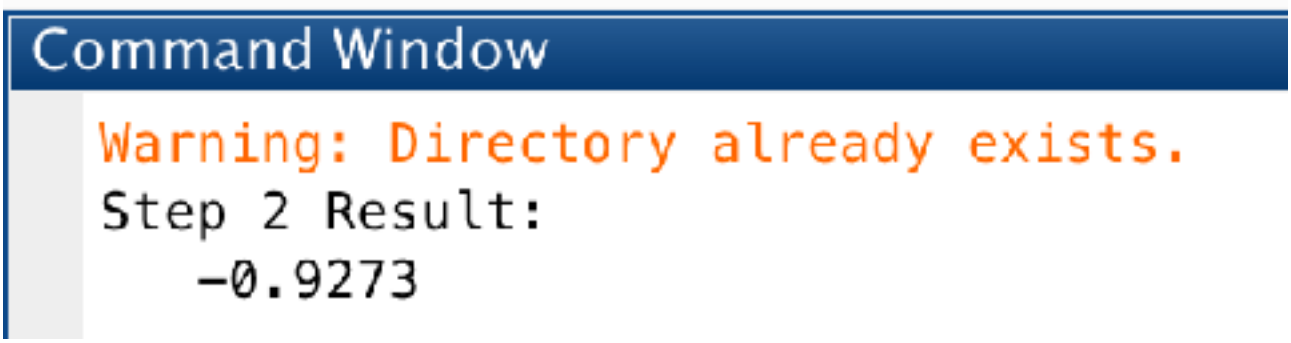
**Σκοπός**: Η άσκηση αυτή έχει ως σκοπό την υλοποίηση ενος συστήματος οπτικής ανγνώρισης ψηφίων. Για την υλοποίηση και έλεγχο των αποτελεσμάτων που δίνει, μας δίνονται δεδομένα απο την US Postal Service τα οποία διακρίνονται σε train και test αντίστοιχα, για την εκπαίδευση και έλεγχο του συστήματος μας. Η πορεία την οποία ακολουθήσαμε για να υλοποιήσουμε τα ζητούμενα φαίνεται παρακάτω :

# Εκτέλεση Άσκησης

**Βήμα 1** : Αρχικά εισάγουμε στο Matlab τα train δεδομένα με χρήση της συνάρτησης importdata(). Τα δεδομένα αποθηκεύονται σε ενα πίνακα διαστάσεων 7291X257. Πηγαίνοντας στη γραμμή 131 του πίνακα αυτού και αγνοώντας το στοιχείο (131,1) το οποίο λέει ποιό ψηφίο είναι, παίρνω τα χαρακτηριστικά του ψηφίου αυτού(δηλαδή την υπόλοιπη γραμμή) και με χρήση της συνάρτησης reshape() τα οργανώνω σε ενα 16X16 πίνακα. Στη συνέχεια αυτό που κάνω είναι να εμφανίζω και να αποθηκεύω σε κατάλληλο directory το ψηφίο το οποίο εμφανίστηκε με χρήση της imagesc(). Η παραπάνω διαδικασία είναι πιο εύκολα κατανοητή αν δείτε το script που σας παραθέτω. Η σχεδίαση του 131ου ψηφίου με τη βοήθεια του Matlab φαίνεται παρακάτω:



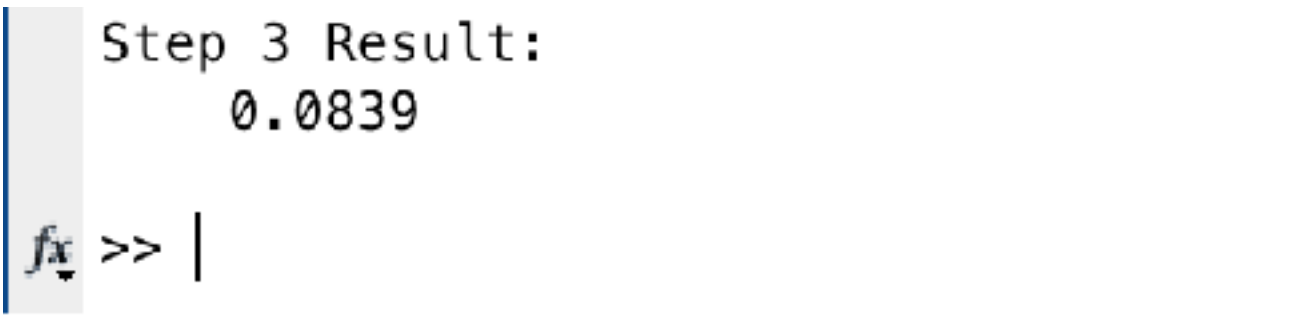
**Βήμα 2** : Στο δεύτερο βήμα για τον υπολογισμό της μέσης τιμής των χαρακτηριστικών του pixel (10,10) για το ψηφίο 0 με βάση τα train δεδομένα ακολουθήσαμε την εξης συλλογιστική : Βρίσκω απο τα train data όλα τα μηδενικά και για καθένα απο αυτά κρατάω τα χαρακτηριστικά του, τα οργανώνω σε ενα 16X16 πίνακα όπως πριν και αθροίζω τη τιμή που είναι στο pixel (10,10) σε ενα μετρητή. Επίσης διατηρώ και το πλήθος των μηδενικών σε ενα άλλο μετρητή. Αφού εξετάσω όλα train δεδομένα μπορώ να βρω τη μέση τιμή των χαρακτηριστικών του pixel (10,10) παίρνοντας το λόγο των δυο παραπάνω μετρητών.
Το αποτέλεσμα το οποίο πήρα όπως αυτό φαίνεται στο Matlab παρουσιάζεται παρακάτω :



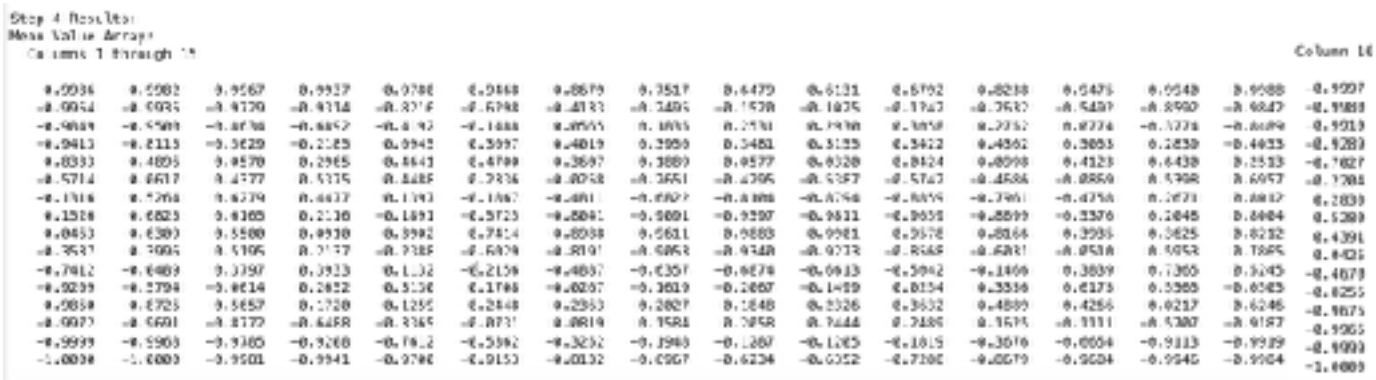

**Βήμα 3** : Στο τρίτο βήμα αυτό που έχω να κάνω είναι να βρω τη διασπορά των χαρακτηριστικών του pixel (10,10) για το ψηφίο 0 με βάση τα train δεδομένα. Αυτό που κάναμε είναι να χρησιμοποιήσουμε την σχέση που συνδέει τη μέση τιμή και τη διασπορά, η οποία φαίνεται στη συνέχεια :

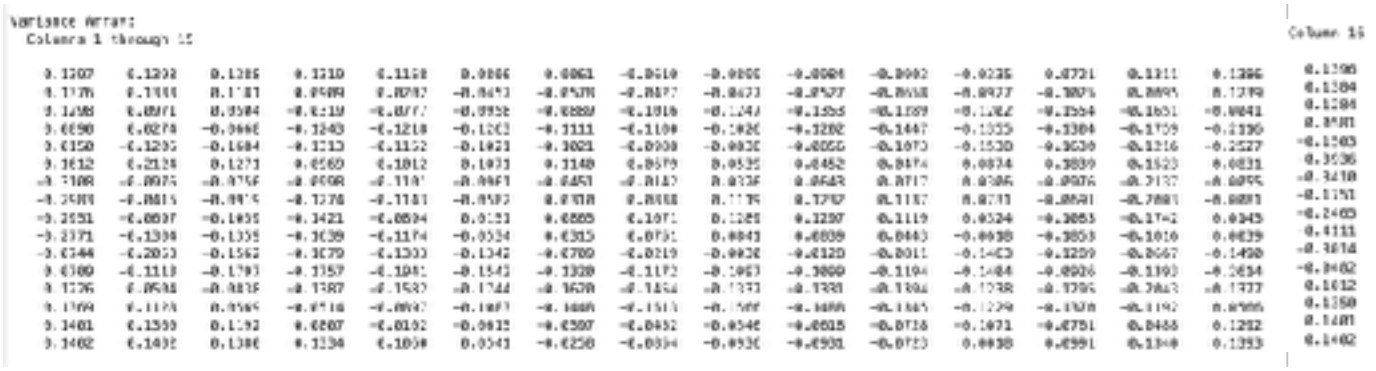
$$\mathbf{Var}(X) = \mathbf{E}[X^2] - (\mathbf{E}[X])^2$$

Ουσιαστικά αυτό που χρησιμοποιήσαμε είναι ενας ακόμα μετρητής ο οποίος κρατούσε το άθροισμα των τετραγώνων των τιμών που έχουν τα pixel (10,10) όλων των μηδενικών. Διαιρώντας το άθροισμα αυτό με το πλήθος των μηδενικών των train data παίρνουμε τον πρώτο όρο της παραπάνω εξίσωσης. Ο δεύτερος όρος της παραπάνω εξίσωσης είναι έτοιμος απο το Βήμα 2 και το μόνο που έχουμε να κάνουμε είναι να αφαιρέσουμε τις δυο αυτές τιμές. Όπως είναι προφανές τα βήματα 2,3 μπορούνα να γίνουν μαζί(στο script που σας παραδίδουμε θα δείτε οτι υπολογίζονται στην ίδια for loop). Το αποτέλεσμα το οποίο πήρα όπως αυτό φαίνεται στο Matlab παρουσιάζεται παρακάτω :

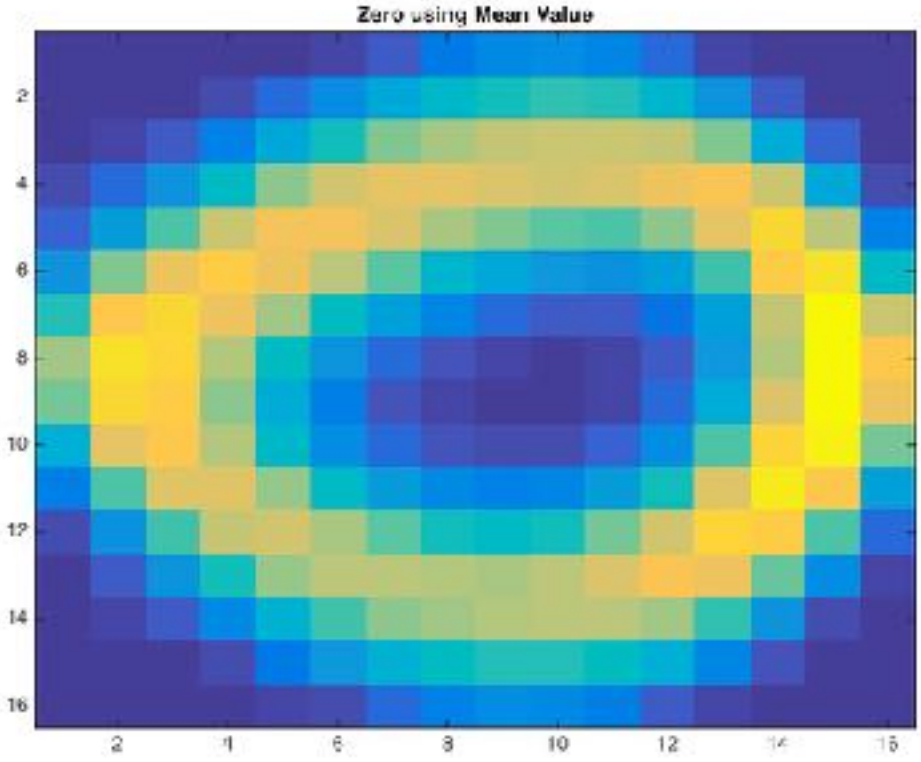


**Βήμα 4** : Στο βήμα αυτό το μόνο που έχουμε να κάνουμε είναι να υλοποιήσομε τη διαδικασία των βημάτων 2 και 3 για όλα τα pixel του ψηφίου 0 και όχι μόνο για το (10,10). Ουσιαστικά θα πρέπει να βρούμε τη μέση τιμή και τη διασπορά για καθένα απο τα pixel του ψηφίου 0, άρα χρησιμοποιούμε δυο πίνακες 16Χ16 αντι για δυο απλούς αθροιστές των οποίων το κάθε κελί είναι το άθροισμα των χαρακτηριστικών και το άθροισμα των τετραγώνων των χαρακτηριστικών αντίστοιχα, για το καθένα pixel. Η υπόλοιπη διαδικασία είναι όμοια με αυτή που εξηγήσαμε παραπάνω. Στη συνέχεια δημιουργούμε δυο πίνακες 16Χ16 οι οποίοι είναι αυτοί που θα κρατούν την μέση τιμή και την διασπορά για κάθε pixel. Για να βρω τη μέση τιμή και τη διασπορά χρησιμοποιώ τους τύπους που ανέφερα στα βήματα 2,3 και κρίνω σκόπιμο να μην τους αναφέρω ξανά. Το αποτέλεσμα το οποίο πήρα όπως αυτό φαίνεται στο Matlab παρουσιάζεται παρακάτω :

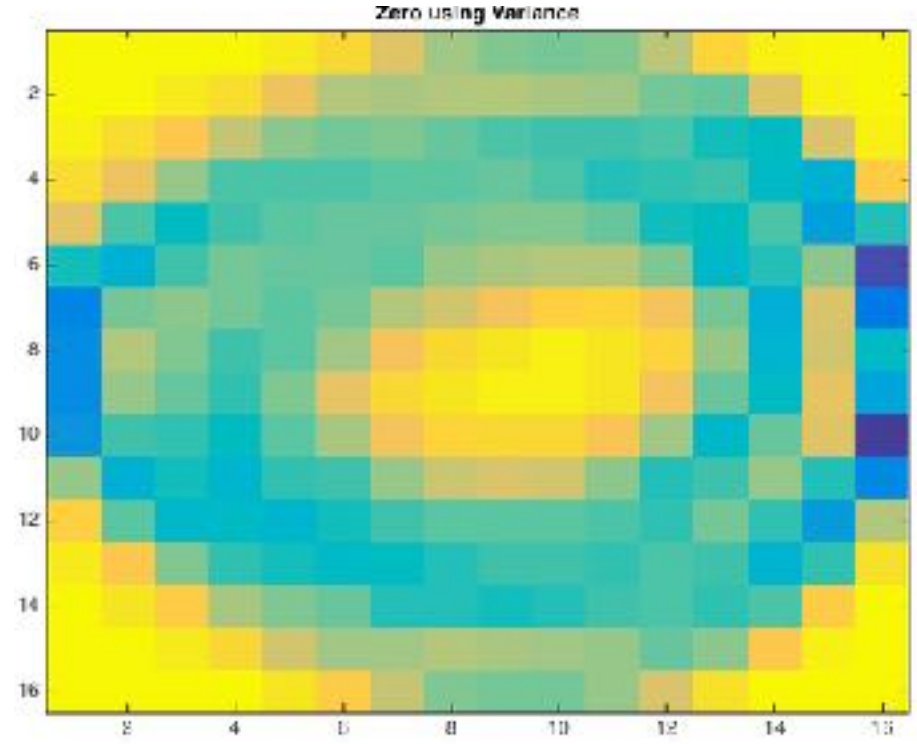


**Βήμα 5** : Με βάση τις τιμές της μέσης τιμής τις οποίες υπολογίσαμε στο Βήμα 4, το αποτέλεσμα το οποίο παίρνουμε στο σχεδιασμό του μηδενός με χρήση αυτών είναι το παρακάτω :



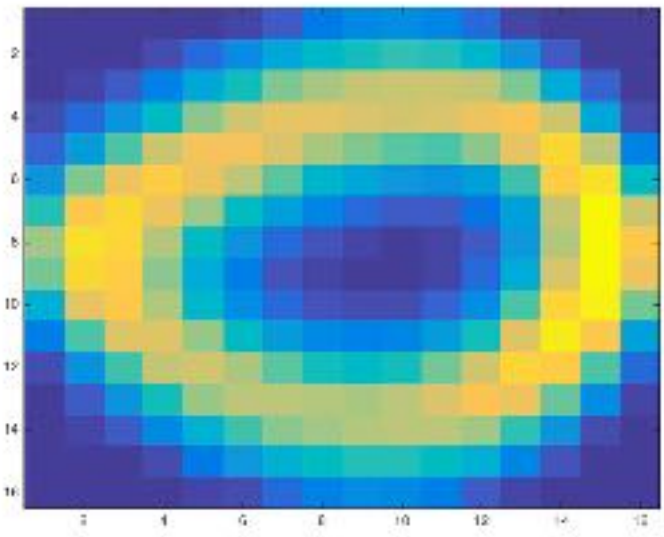

**Βήμα 6** : Με βάση τις τιμές της διασποράς τις οποίες υπολογίσαμε στο Βήμα 4, το αποτέλεσμα το οποίο παίρνουμε στο σχεδιασμό του μηδενός με χρήση αυτών είναι το παρακάτω :

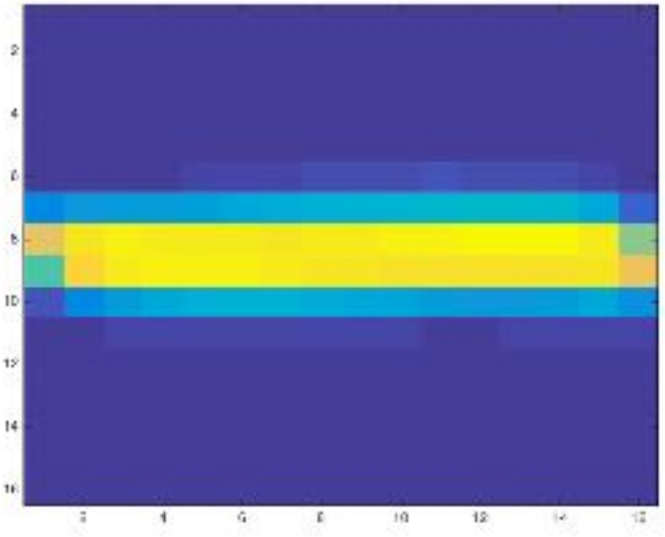


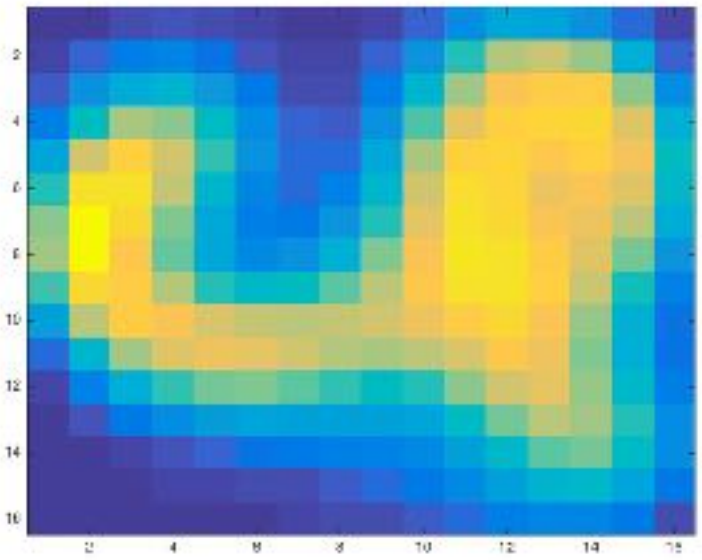
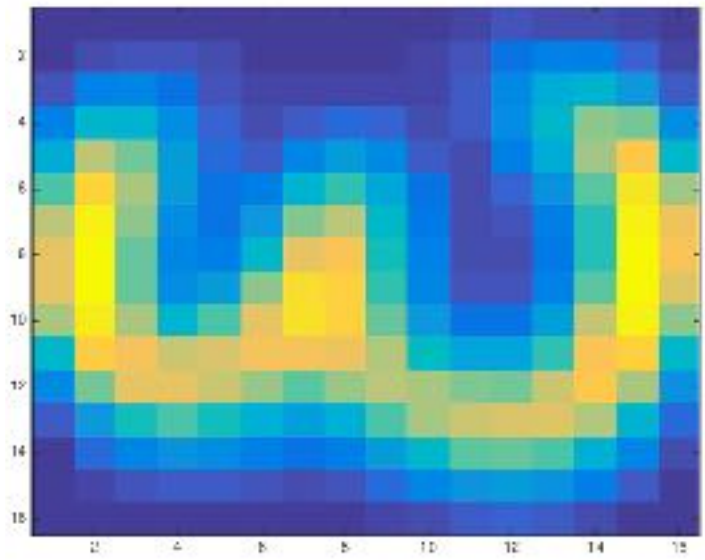
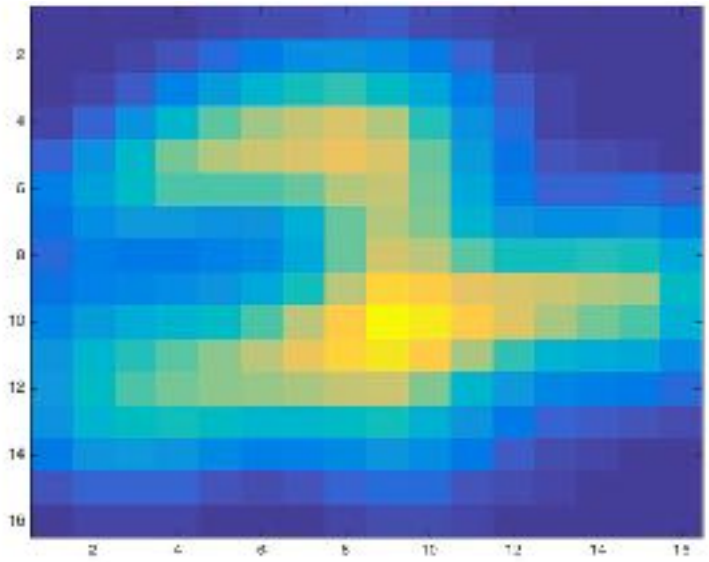
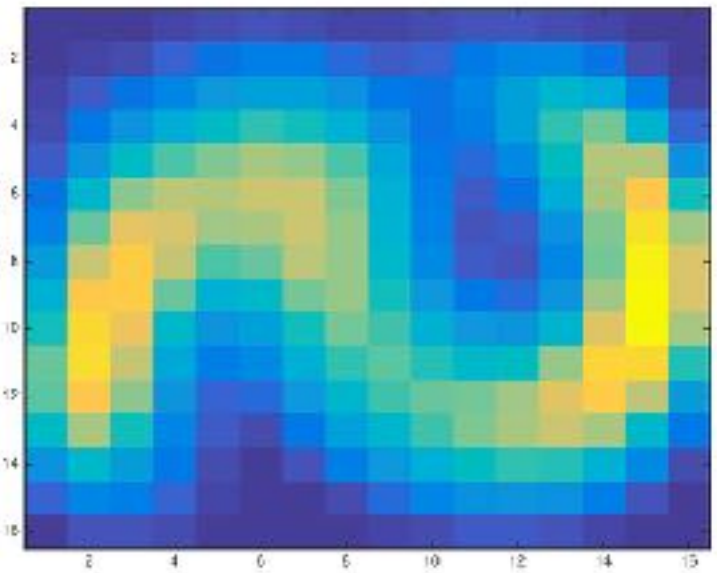
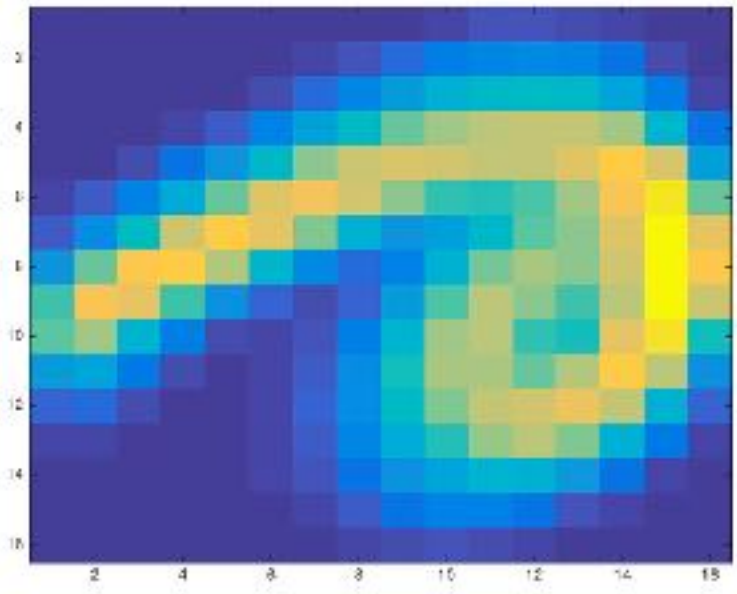
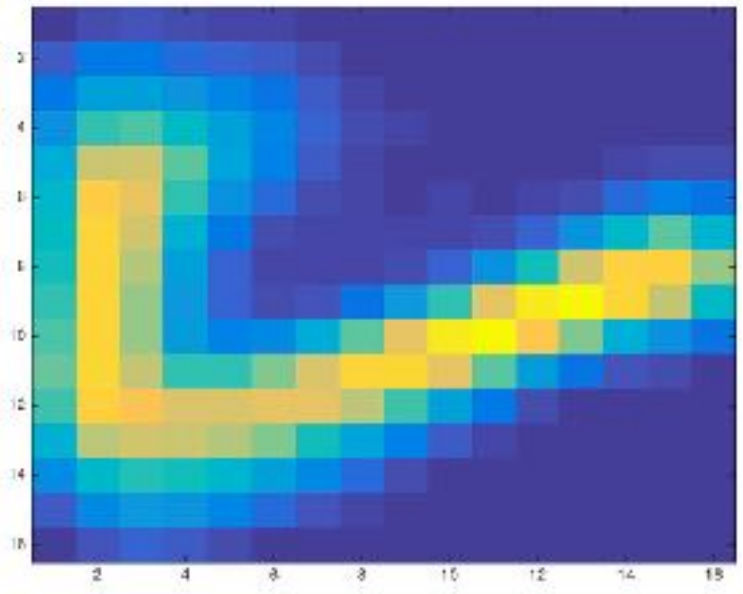
## Σύγκριση Αποτελεσμάτων Βημάτων 5,6

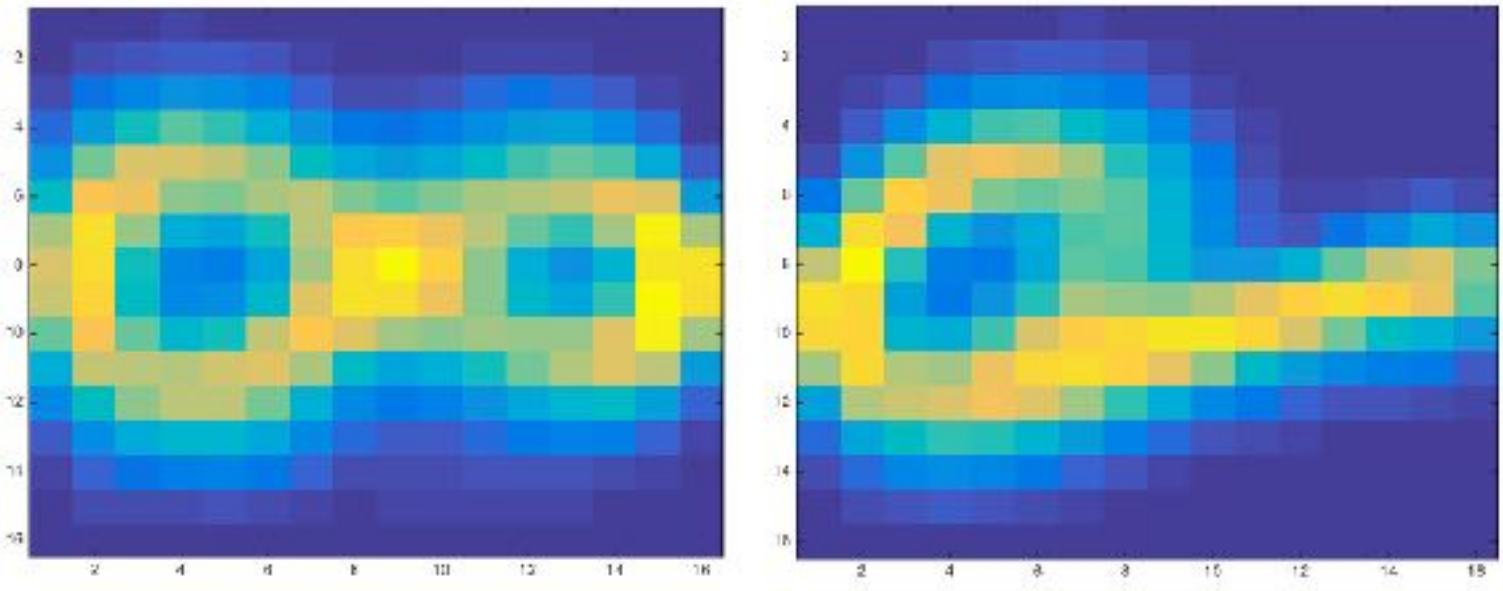
Απο τα παραπάνω γραφήματα βλέπουμε οτι όταν χρησιμοποιούμε τη μέση τιμή στη σχεδίαση του μηδενός το αποτέλεσμα το οποίο παίρνουμε είναι πολύ καλύτερο απο οτι αν παίρναμε τη διασπορά. Αυτό είναι λογικό διότι

**Βήμα 7** : Στο πρώτο μισό του βήματος αυτού έχουμε να υπολογίσουμε τις τιμές για τη μέση τιμή και τη διασπορά των χαρακτηριστικών για καθένα απο τα ψηφία, κάνοντας χρήση των train δεδομένων. Ουσιαστικά στο βήμα αυτό θέλουμε να επεκτείνουμε αυτό που κάναμε Βήμα 4 για όλα τα digits και οχι μόνο για το 0. Συγκεκριμένα θα πρέπει να βρούμε τη μέση τιμή και τη διασπορά για καθένα απο τα pixel κάθε ψηφίου, άρα χρησιμοποιούμε δυο πίνακες 16X16X10 αντι για δυο 2d αθροιστές των οποίων το κάθε κελί είναι το άθροισμα των χαρακτηριστικών και το άθροισμα των τετραγώνων των χαρακτηριστικών αντίστοιχα, για το καθένα pixel του καθενός ψηφίου. Η υπόλοιπη διαδικασία είναι όμοια με αυτή που εξηγήσαμε στα παραπάνω βήματα.
Στο δεύτερο μέρος του βήματος αυτού έχουμε να σχεδιάσουμε τα digits κάνοντας χρήση των αποτελέσματων που βρήκα παραπάνω. Συγκεκριμένα δημιουργούμε δυο πίνακες 16X16X10 οι οποίοι είναι αυτοί που θα κρατούν την μέση τιμή και την διασπορά για κάθε pixel του κάθε ψηφίου. Για να βρω τη μέση τιμή και τη διασπορά χρησιμοποιώ τους τύπους που ανέφερα στα βήματα 2,3 και κρίνω σκόπιμο να μην τους αναφέρω ξανά. Τα αποτελέσματα τα οποία πήραμε φαίνονται παρακάτω :





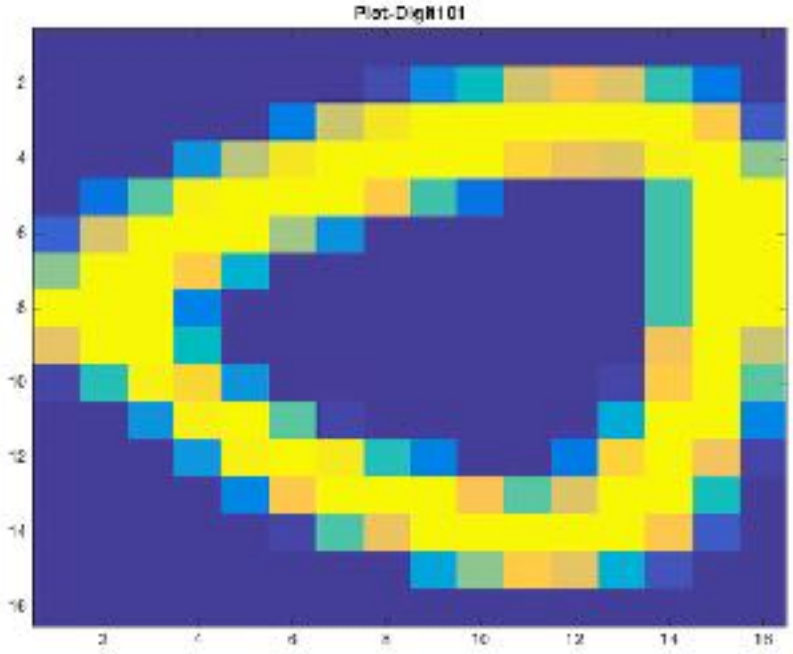


Όπως φαίνεται απο τα παραπάνω γραφηματα ο σχεδιασμός των ψηφίων έγινε με σωστό τρόπο και το αποτέλεσματ το οποίο προέκυψε είναι πολύ ικανοποιητικό.

**Βήμα 8** : Στο βήμα αυτό αρχικά εισάγουμε στο Matlab τα test δεδομένα με χρήση της συνάρτησης importdata(), την οποία αναφέραμε και παραπάνω. Στη συνέχεια παίρνουμε το 101ο digit των test data και χρησιμοποιώντας σα μετρική την Ευκλείδια απόσταση προσπαθήσαμε να το κατατάξουμε σε μια απο τις 10 κατηγορίες με βάση τα αποτελέσματα που πήραμε στο Βήμα 7α. Συγκεκριμένα αφου χρησιμοποιήσαμε τη συνάρτηση reshape() για να οραγανώσουμε τα χαρακτηριστικά του 101ου ψηφίου σε ενα 16X16 πίνακα για να υπολογίσουμε την ευκλείδια απόσταση χρησιμοποιούμε ενα πίνακα 16X16X10 για να κρατήσουμε τις ευκλείδιες αποστάσεις για κάθε pixel κάθε ψηφίου και ενα πίνακα 10X1 ο οποίος περιέχει το άθροισμα των ευκλείδιων αποστάσεων για κάθε pixel κάθε ψηφίου. Βρίσκωντας σε ποιά θέση του πίνακα αυτού έχουμε το ελάχιστο άθροισμα βρίσκουμε στην ουσία με ποιό ψηφίο κάνει match ο ταξινομητής μας το 101ο ψηφίο των test data. Το αποτέλεσμα το οποίο πήραμε, όπως αυτό φαίνεται στο Matlab φαίνεται παρακάτω:

```
Step 8 --> Digit 101 of test data result =
     0
```

Όπως μπορείτε να δείτε το 101ο ψηφίο των test data είναι το 0, το οποίο σημαίνει οτι ο ταξινομητής μας έκανε σωστό classification. Για πληρότητα σας παραθέτω και το 101ο ψηφίο των test data:




**Βήμα 9** : Στο πρώτο μέρος του βήματος αυτού θα πρέπει να ταξινομήσουμε όλα τα ψηφία των test data σε μια απο τις 10 κατηγορίες με χρήση της Ευκλείδιας απόστασης, ουστικά να επεκτείνουμε το ερώτημα 8. Η ιδέα που ακολουθήσαμε εδω είναι η εξής: Τα test data στο Matlab είναι ενας πίνακας 2007X257, άρα η απάντηση στο ερώτημα αυτό θα είναι ενα πίνακας διαστάσεων 2007X1 ο οποίος σε κάθε θέση του θα έχει το αποτέλεσμα του ταξινομητή μας για το αντίστοιχο digit των test data. Για να γεμίσουμε το πίνακα αυτό τρέχουμε μια for-loop 2007 φορές και για κάθε ψηφίο παίρνουμε τα χαρακτηριστικά του, τα οργανώνουμε σε ενα πίνακα 16X16, δημιουργούμε το πίνακα ευκλείδιων αποστάσεων όπως στο ερώτημα 8 και ακολουθούμε την ίδια διαδικασία για να πάρουμε το αποτέλεσμα του ταξινομητή μας για κάθε ψηφίο(ο πίνακας res είναι ο πίνακας με αυτά τα αποτελέσματα, όπως μπορείτε να δείτε και στο script που σας παραδίδουμε στο zip αρχείο).

Στο δεύτερο μέρος του βήματος αυτού ουστικά μετράμε τα σωστά αποτελέσματα του ταξινομητής μας για τα 2007 test data. Το ποσοστό επιτυχίας το οποίο πήραμε με χρήση του Matlab φαίνεται παρακάτω:

```
Step 9 : Success Rate =
   81.4150
```

Για πληρότητα υπολογίσαμε και το ποσοστό επιτυχίας για καθένα digit ξεχωριστά για να ελέγξουμε την ορθότητα των αποτελεσμάτων του ταξινομητής μας. Τα αποτελέσματα τα οποία πήραμε σας τα παραθέτω παρακάτω :

```
success ratio of 0
   82.7298

success ratio of 1
   98.1061

success ratio of 2
   73.2323

success ratio of 3
   78.9157

success ratio of 4
    75

success ratio of 5
   76.8750
success ratio of 5
   84.1176

success ratio of 7
   79.5918

success ratio of 8
   77.1084

success ratio of 9
   79.6610
```

Απο τα παραπάνω αποτελέσματα βλέπουμε οτι ο ταξινομητής μας δίνει αρκετά καλά αποτελέσματα και κάνει αρκετά καλό classifiation ψηφίων.

Παραπάνω λεπτομέριες για το τρόπο υλοποίησης κάθε βήματος θα βρείτε μέσα στο script που σας επισυνάπτουμε στο zip αρχείο αφού ο κώδικας έχει πολλά κατανοητά σχόλια.