Εργασία στο μάθημα
Συστήματα πολυμέσων και εικονικής πραγματικότητας
Κωδικοποιητής/αποκωδικοποιητής MPEG-1

Μπεκιάρης Θεοφάνης ΑΕΜ:8200

Σχετικά με του κώδικες της εργασίας

Στην εργασία συμπεριλαμβάνονται 3 φάκελοι (Step1,2,3) με τα αντίστοιχα αρχεία της κάθε ενότητας. Επιπλέον υπάρχει ο φάκελος ‘images’ στον οποίο τοποθετούμε τις εικόνες – frame. Να σημειωθεί ότι τα ονόματα των εικόνων πρέπει να είναι της μορφής πχ coastguard2 και όχι coastguard02,δηλαδή χωρίς μηδενικά,μόνο όνομα και τρέχον αριθμός frame. Στις ενότητες 2 και 3 που χρειάζονται οι συναρτήσεις της ενότητας 1,δημιουργώ εσωτερικά των συναρτήσεων encodeMPEG, decodeMPEG και encodeMPEG3,decodeMPEG3 διαδρομές μέσω της συνάρτησης addpath(genpath('../Step1')) ώστε να καλούνται κατευθείαν οι συναρτήσεις μέσα από τον φάκελο Step1.Το ίδιο κάνω και για τις εικόνες. Επομένως κάθε αρχείο βρίσκεται στον φάκελο που του αντιστοιχεί και δεν επαναλαμβάνονται.
Μπαίνοντας στον φάκελο Step2 και Step3 φαίνονται οι συναρτήσεις κωδικοποίησης των εικόνων. Ενώ στον φάκελο decode που περιέχεται εσωτερικά αυτών βρίσκονται οι συναρτήσεις αποκωδικοποίησης για την κάθε ενότητα. Για την εκτέλεση των προγραμμάτων ας πάρουμε για παράδειγμα της ενότητα 2,μπορούμε να καλέσουμε κατευθείαν μέσα από τον φάκελο Step2 την συνάρτηση encodeMPEG με τα αντίστοιχα ορίσματα η οποία θα παράξει την ζητούμε δομή με όνομα Stream.mat. Στην συνέχεια πηγαίνοντας στον φάκελο decode και καλώντας την decodeMPEG, η συνάρτηση θα αποκωδικοποιήσει τη προηγούμενη δομή και θα δημιουργήσει έναν φάκελο με όνομα saveImages στον οποίο θα αποθηκεύσει τις παραγόμενες εικόνες .Με απλά λόγια για την εκτέλεση των προγραμμάτων αρκεί η εκτέλεση των συναρτήσεων encodeMPEG, decodeMPEG με τα επιθυμητά ορίσματα. Φυσικά μέσα σε κάθε φάκελο Step1,2,3 υπάρχει το αντίστοιχο αρχείο demo που επιδεικνύει τον τρόπο εκτέλεσης τους.

Ενότητα 1 (MPEG Library)

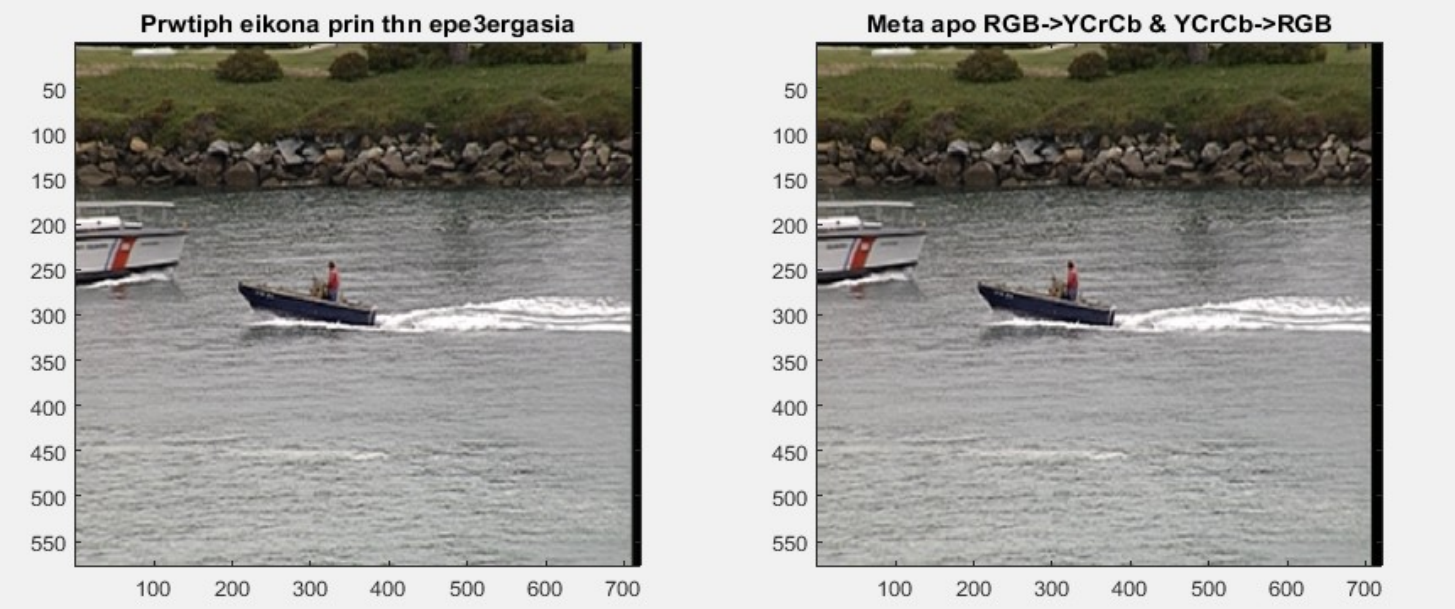
Περιγραφή:
Σκοπός της ενότητας είναι η δημιουργία μιας βιβλιοθήκης συναρτήσεων που θα χρησιμοποιηθούν αργότερα για την κατασκευή του κωδικοποιητή .

Συναρτήσεις : Σχετικά με τα ορίσματα και τις εξόδους των συναρτήσεων είναι ακριβώς όπως ζητούνται στην εργασία,επομένως παραλείπεται η περιγραφή τους στην αναφορά, θα αναφερθούν μόνο όπου αυτό είναι απαραίτητο. Στην συνέχεια παραθέτω συνοπτική περιγραφή των συναρτήσεων που κατασκευάστηκαν.

ccir2ycrcb():Η συνάρτηση παίρνει ως είσοδο έναν 576x720x3 πίνακα που αντιστοιχεί στις RGB χρωματικές συνιστώσες και μετά την μετατροπή τους σε χρωματικό σύστημα YCrCb και μετατροπή σε CCIR 601 format 4:2:2 χρησιμοποιώ την διαδικασία με την εφαρμογή των φίλτρων και υποδειγματοληψίας όπως περιγράφεται στις σελίδες 57-60 από το δοσμένο αρχείο με το προτύπου MPEG. Επίσης να αναφέρω ότι για την μετατροπή σε YCrCb ακολούθησα την μέθοδο που αναφέρεται στο Wikipaidia(**https://en.wikipedia.org/wiki/YCbCr** στην ενότητα JPEG conversion).

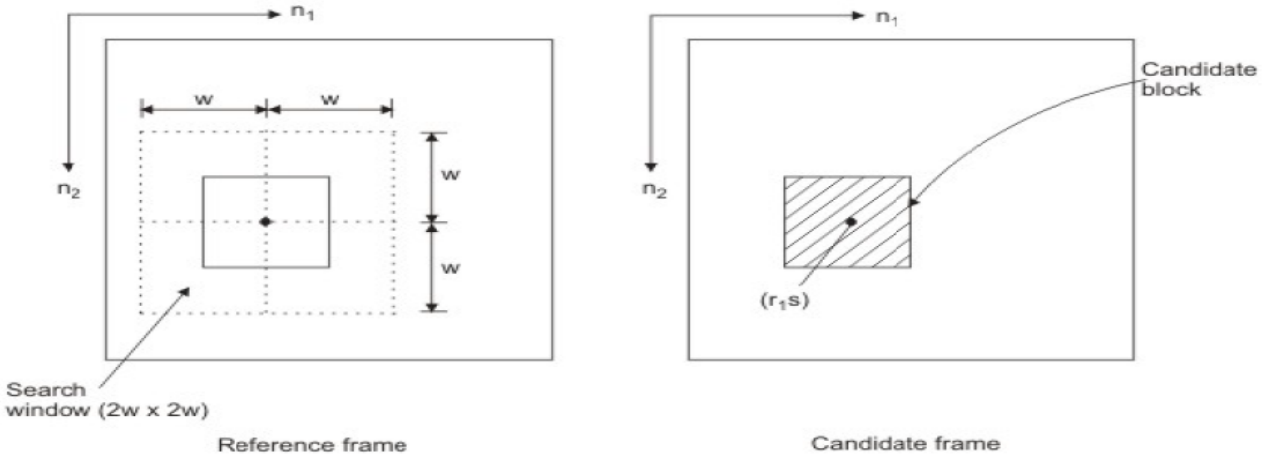
ycrcb2ccir(): Η συνάρτηση ακολουθεί ακριβώς την αντίθετη διαδικασία με την προηγούμενη συνάρτηση. Η διαδικασία είναι αυτή που περιγράφεται στις σελίδες 109-111 του προτύπου.

Παρακάτω φαίνεται το αποτέλεσμα της εικόνας coastguard3.tiff στην πρωτότυπη μορφή της και μετά από διαδοχική εφαρμογή των παραπάνω συναρτήσεων. Βλέπουμε ότι μετά από την διαδικασία της υποδειγματοληψίας και της αναδόμησης της εικόνας,η εικόνα δεν έχει αισθητές διαφορές με την πρότυπη.



Συναρτήσεις **motEstP(), iMotEstP(), motEstB(), iMotEstB() :**

Η συναρτήσεις έχουν ως στόχο την αναζήτηση ενός Macroblock από προηγούμενο ή επόμενο της σειρά προβολής frame (ανάλογα και με τον τύπο τους B ή P) που να μοιάζει περισσότερο στο τρέχον Macroblock του τρέχοντος frame. Για τον καθορισμό της ομοιότητας υπολογίζουμε το μέσο τετραγωνικό σφάλμα των υπο εξέταση macroblock. Τα όρια αναζήτησης για ένα macroblock καθορίζονται από ένα παράθυρο αναζήτησης με μέγιστο μήκος w=7 όπως ορίζει η εκφώνηση της εργασίας.



Συναρτήσεις: blockDCT(), iBlockDCT() :Απλή εφαρμογή τον αντίστοιχων συναρτήσεων Matlab.

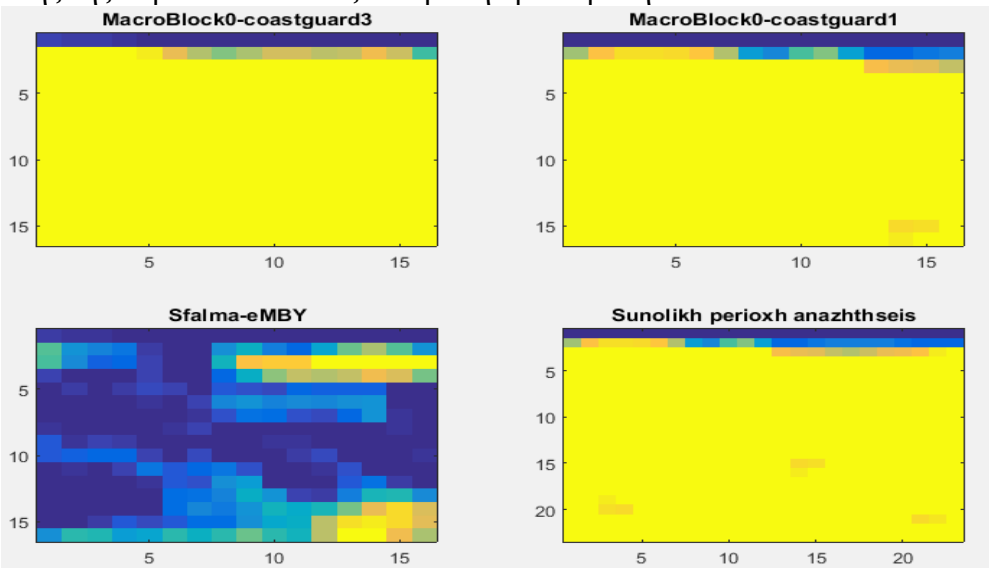
Συναρτήσεις κβαντισμού: Μεθοδολογία σελίδων 87-90.Εφαρμόζω τον τύπο με για τον κβαντισμό όπως περιγράφεται στην ενότητα. Η μόνη παρατήρηση που πρέπει να γίνει είναι σχετικά με τον κβαντισμό των I-frame κατά τον οποίο η DC τιμή του πίνακα των DCT συντελεστών κβαντίζεται με μοναδικό τρόπο σε σχέση με τις υπόλοιπες τιμές. Συγκεκριμένα όπως περιγράφεται στο πρότυπο ο συντελεστής DC παίρνει τιμές από 0 ως 2040 και κβαντίζεται πάντα με σταθερή τιμή ίση με 8.

Συναρτήση runLength(),iRunLength(): Στην συνάρτηση runLenght() έχω εφαρμόσει διαδικασία κατά την οποία το πρόγραμμα δημιουργεί έναν γραμμικό πίνακα όπου τα στοιχεία έχουν τοποθετηθεί ακολουθώντας την ZigZag διαδρομή όπως περιγράφεται στο πρότυπο. Στην συνέχεια μετρώντας τον αριθμό των μηδενικών δημιουργούμε τα σύμβολα μήκους διαδρομής.

Συναρτήση vlc(),ivlc(): Χρησιμοποιώ τους πίνακες που δίνονται για να κάνω μετατροπή των προηγούμενων συμβόλων σε vlc κώδικα. Η αναλυτική διαδικασία υλοποίησης με σχόλια φαίνεται στον κώδικα.

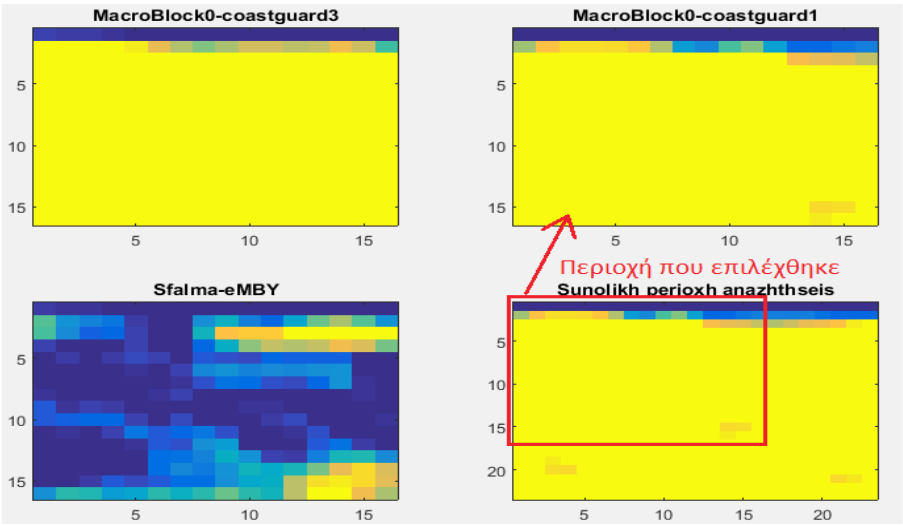
Demo1

Μετά την εκτέλεση του αρχείου demo1 με τα δεδομένα που ζητούνται έχουμε τα παρακάτω αποτελέσματα από την συνάρτηση mostEstP. Επίσης στον demo1 βρίσκεται και ο κώδικας της παρουσίασης της παραπάνω εικόνας από με την μετατροπή από RGB σε YCrCb και αντίστροφα.



- Πάνω αριστερά το αρχικό MacroBlock από την εικόνα coastguard3.tiff
- Πάνω δεξιά το Macroblock που επέλεξε η mostEstP συνάρτηση και κάτω δεξιά η συνολική περιοχή αναζήτησης. Υπενθυμίζω ότι η αναζήτηση γίνεται σε επίπεδο pixel,επομένως σύμφωνα με τα δοσμένα όρια [-7,7] και το ότι έχουμε το πρώτο block της εικόνας δηλαδή τα όρια αναζήτησης μόνο για θετικές τιμές [0,7] αφού για αρνητικές βγαίνουμε εκτός εικόνας,τελικά η συνολική περιοχή αναζήτησης περιορίζεται στον παραπάνω χώρο.
- Κάτω αριστερά εκτυπώνεται το σφάλμα eMBY.

Η επιστρεφόμενη τιμή του Motion Vector είναι [0 NaN;0 NaN].Δηλαδή η περιοχή που επιλέχθηκε από την συνολική περιοχή αναζήτησης είναι(κόκκινο τετράγωνο):



Παρατηρούμε ότι αυτή η περιοχή όντως είναι η καλύτερη επιλογή καθώς μοιάζει περισσότερο με την περιοχή που μας ενδιαφέρει(πάνω αριστερά). Επιπροσθέτως παρατηρώντας και το σχήμα του σφάλματος η περιοχές με μπλε αντιστοιχούν σε μηδενική τιμή,άρα και διαφορά μεταξύ των δύο εικόνων block. Να σημειωθεί ότι η συνάρτηση mostEstP δεν βρίσκει την περιοχή που μοιάζει απόλυτα στην αρχική μας περιοχή καθώς μπορεί να μην υπάρχει τέτοια περιοχή,αλλά μια περιοχή που μοιάζει περισσότερο σε σχέση με τις άλλες περιοχές,επομένως σφάλματα είναι λογικό να υπάρχουν.

Ενότητα 2 MPEG Processing

Demo2

Κατά την εκτέλεση του demo2 καλούνται οι συναρτήσεις για την κωδικοποίηση και την αποκωδικοποίηση για 4 frame και 1 GOP(λόγο χρόνο εκτέλεσης) για μορφή GOP = IBBP και κλίμακας κβαντισμού ίση με qScale = 8 και για τα frame coastguard0 ως coastguard3. Το πρόγραμμα φυσικά δουλεύει για οποιοδήποτε συνδυασμό αριθμού εικόνων,αριθμού GOP,μορφής GOP και γενικότερα καλύπτει της απαιτήσεις της εκφώνησης. Τα αποτελέσματα των εικόνων και της συμπίεσης είναι:

Με αυτά τα δεδομένα ο κωδικοποιητής πετυχαίνει κωδικοποίηση η οποία παράγει ένα αρχείο μεγέθους **87.625KBytes**. Η καταμέτρηση έχει γίνει από την δοσμένη συνάρτηση getSeqEntityBits. Το αρχικό μέγεθος των 4 εικόνων υπολογίζεται ως εξής. Αν θεωρήσουμε ότι έχουμε σύστημα GRB με 8bit αναπαράσταση χρώματος και εικόνες διάστασης 720x576 τότε χρειάζομαι 720*576*3 bytes για την κάθε εικόνα,άρα συνολικά και για τις 4 εικόνες 720*576*3*4 = 4976640 bytes δηλαδή περίπου **4.976MB**
Επομένως βλέπουμε ότι η συμπίεση που επιτεύχθηκε είναι 4976Kbytes/ 87.625KBytes = 56.8
Επομένως έχουμε μειώσει τον όγκο των δεδομένων **κατά 57(περίπου) φορές**.

Το μέσο απόλυτο σφάλμα για το κάθε frame είναι :
Mean_absolute_error of frame0 = 3.694568
Mean_absolute_error of frame1 = 9.923167
Mean_absolute_error of frame2 = 10.200811
Mean_absolute_error of frame3 = 8.893836

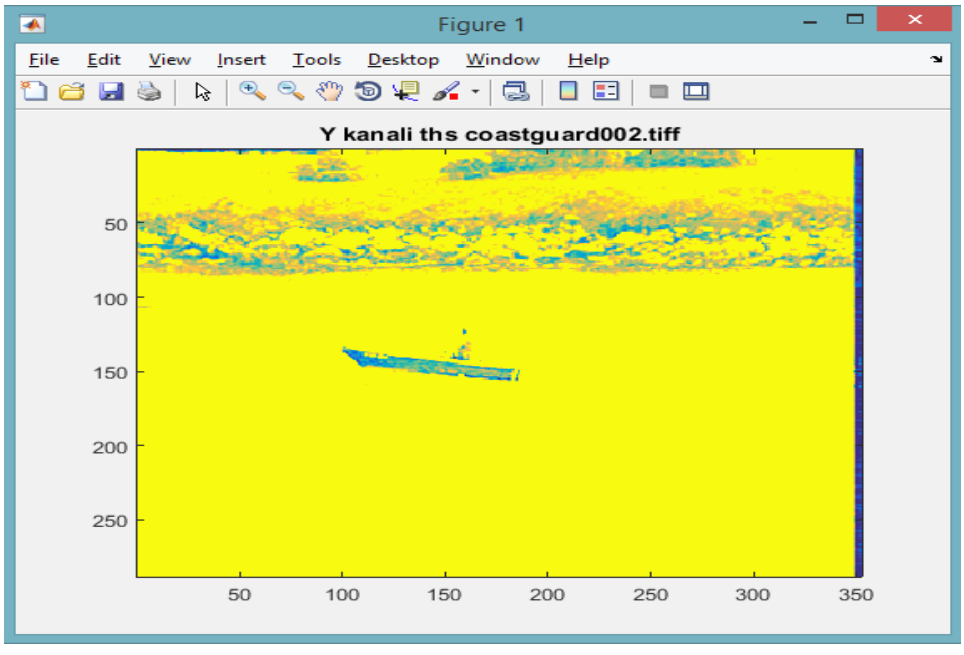
Επιπλέον μπορούμε να παρατηρήσουμε την εξάρτηση από την κλίμακα κβάντισης των αποτελέσματων σχετικά με το μέγεθος της συμπίεσης και την ποιότητα των εικόνων.

Για qScale = 16 έχουμε:
Μέγεθος συμπιεσμένου = 57.7 KBytes και σφάλματα ανά εικόνα
Mean_absolute_error of frame0 = 3.812056
Mean_absolute_error of frame1 = 10.019465
Mean_absolute_error of frame2 = 10.378625
Mean_absolute_error of frame3 = 9.188182

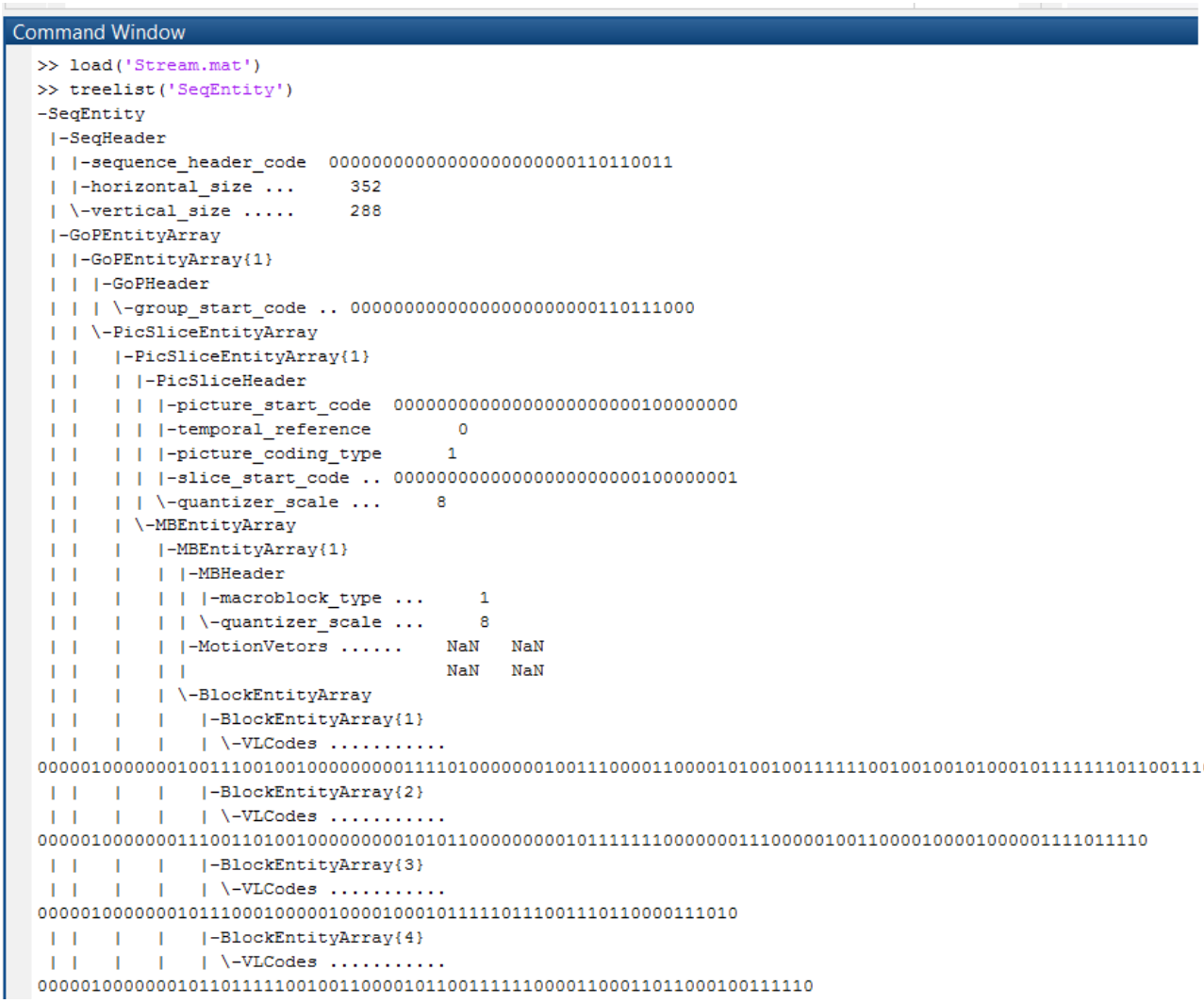
Δηλαδή παρατηρούμε μεγαλύτερη συμπίεση με μεγαλύτερα όμως σφάλματα,δηλαδή πετυχαίνοντας μεγαλύτερη συμπίεση το αντάλλαγμα είναι η ποιότητα της των παραγόμενων εικόνων.

Επιπροσθέτως για μικρότερη κλίματα κβάντισης qScale = 2 έχουμε:
Μέγεθος συμπιεσμένου = 202 KBytes και σφάλματα ανά εικόνα
Mean_absolute_error of frame0 = 3.647790
Mean_absolute_error of frame1 = 9.630150
Mean_absolute_error of frame2 = 9.898623
Mean_absolute_error of frame3 = 8.587063
Και αυτά τα αποτελέσματα επιβεβαιώνουν τα προηγούμενα αποτελέσματα καθώς βλέπουμε ότι με μείωση στον συντελεστή κβάντισης έχουμε αυξημένη ποιότητα αλλά μικρότερη συμπίεση.

Τέλος το κανάλι Y της εικόνας coastquard2.tiff φαίνεται παρακάτω



Επιπροσθέτως μετά την εκτέλεση της συνάρτησης treelist.m παρακάτω φαίνεται ένα απόσπασμα από το αποτέλεσμα που εκτυπώνεται με το αποτέλεσμα της μορφής της δομή που ζητείται να κατασκευαστεί. Στην εικόνα φαίνεται ότι η δομή έχει κατασκευαστεί ορθά σύμφωνα με την περιγραφή που δίνετε στην εργασία.



Ενότητα 3 MPEG Syntax

Η δομή των συναρτήσεων είναι η ίδια με της προηγούμενης ενότητας με της απαραίτητες προφανώς αλλαγές ώστε πλέον οι συναρτήσεις κωδικοποίησης να γράφουν σε ένα stream μορφής πίνακα όπως ζητείται από την εκφώνηση και οι συναρτήσεις αποκωδικοποίησης να διαβάζουν το stream αυτό για την αναδόμηση των εικόνων.

Εκτελώντας τον αρχείο demo3 με τα ίδια δεδομένα όπως και στην προηγούμενη περίπτωση προκύπτει ότι το μέγεθος τους συμπιεσμένου stream είναι ίσο με 82072 Bytes ή **82.072KBytes.** Συγκρίνοντας το αποτέλεσμα αυτό με το προηγούμενο μέγεθος 87.625KBytes βλέπουμε ότι το η συμπίεση μας αυξήθηκε 4976.664Kbytes/ 82.072KBytes = **60.63** . Το αποτέλεσμα αυτό είναι συνέπεια της κωδικοποίησης των Motion vector κατά dpcm που επιτυγχάνει αποσυσχέτιση των διαδοχικών δειγμάτων. Το παραπάνω προκύπτει από την παρατήρηση ότι διαδοχικά motion vector είναι πιθανό να έχουν ίση ή παραπλήσια τιμή με αποτέλεσμα η διαφορά τους να είναι μικρή ή και μηδενική. Χρησιμοποιώντας τους αντίστοιχους κώδικες VLC για την κωδικοποίηση των διαφορών βλέπουμε ότι η αναπαράσταση μικρών αριθμών χρειάζεται λιγότερα bits και κατά συνέπεια όσο πιο μικρή είναι η διαφορά τόσα λιγότερα bits χρειαζόμαστε.