Συστήματα Πολυμέσων και Εικονική Πραγματικότητα

Κωδικοποιητής/αποκωδικοποιητής MPEG-1

Ομάδα Κατανόησης Πολυμέσων

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Εργασία 2017-2018

// Εξωφυλλο με ονοματα αεμ δικας μας τιτλο μαθηματος και λοιπες μαλακιες που λεει στο pdf της εργασιας

//Φουτερ και header

//αριθμοι σελιδων

// Πινακας περιεχομενων που δημιουργείται αυτόματα από το word με βάση τα headings - clickable

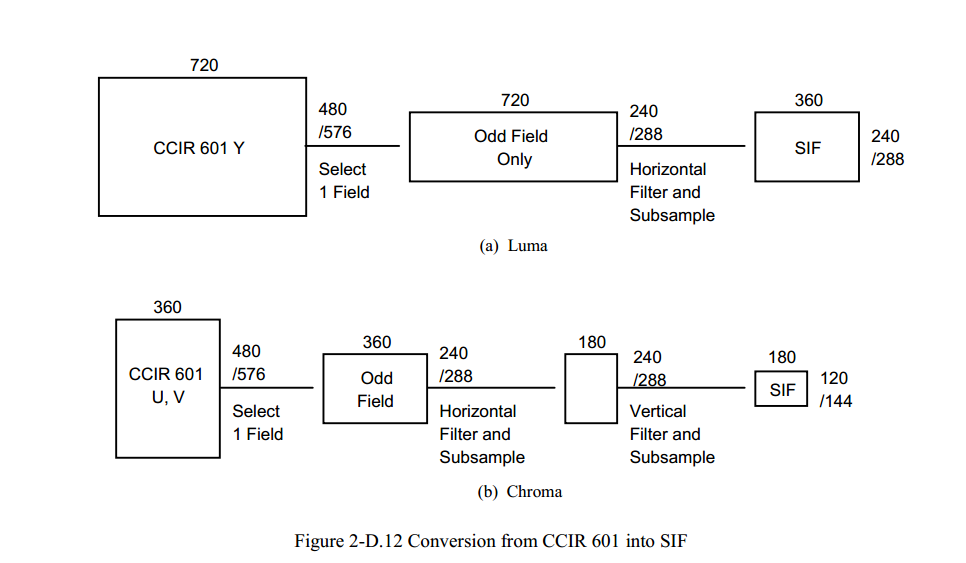
# MPEG Library

## 1.1 Προεπεξεργασία

Για την υλοποίηση της προεπεξεργασίας των εικόνων, υλοποιήθηκαν οι εξής δύο συναρτήσεις:

* function [frameY, frameCr, FrameCb] = ccir2ycrcb(frameRGB)
* function [frameRGB] = ycrcb2ccir(frameY, frameCr, FrameCb)

Οι αρχικές εικόνες που έχουμε είναι σε μορφή RGB μεγέθους 576x720, γι' αυτό αρχικά μετατρέπουμε την κάθε εικόνα σε μορφή CCIR 601, δηλαδή σε Y Cb Cr με αναπαράσταση 4:2:0. Για να δημιουργηθεί το μέγεθος των εικόνων που θέλουμε διατηρούμε μόνο τις περιττές στήλες των Cr και Cb, ενώ παράλληλα οι τιμές των Y, Cb και Cr προκύπτουν από τις αντίστοιχες πράξεις με τις τιμές των R, G και B. Στη συνέχεια, η διαδικασία που ακολουθείται φαίνεται στην παρακάτω εικόνα:



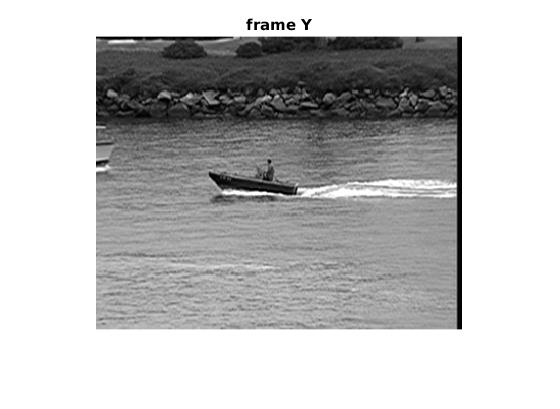
Τα FIR φίλτρα που εφαρμόζονται στην πρώτη συνάρτηση είναι:

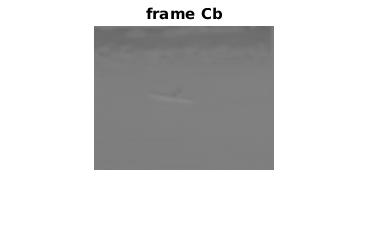
* για την φωτεινότητα: [-29, 0, 88, 138, 88, 0, -29] // 256
* για τις χρωματικότητες: [1, 0, 3, 0, 3, 0, 1] // 8

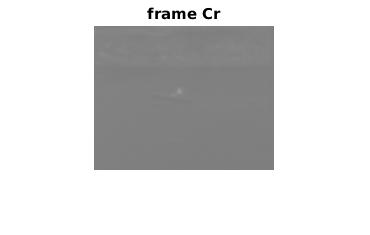
Στα αρχεία του MATLAB για την εφαρμογή των φίλτρων, εφαρμόζεται συνάρτηση που γράφτηκε από εμάς, όμως τα ίδια αποτελέσματα θα προέκυπταν και με την εφαρμογή της αντίστοιχης συνάρτησης conv2 του MATLAB.

Παρακάτω βλέπουμε για την εικόνα coastguard000.tiff τα αποτελέσματα από την εφαρμογή του φίλτρου:









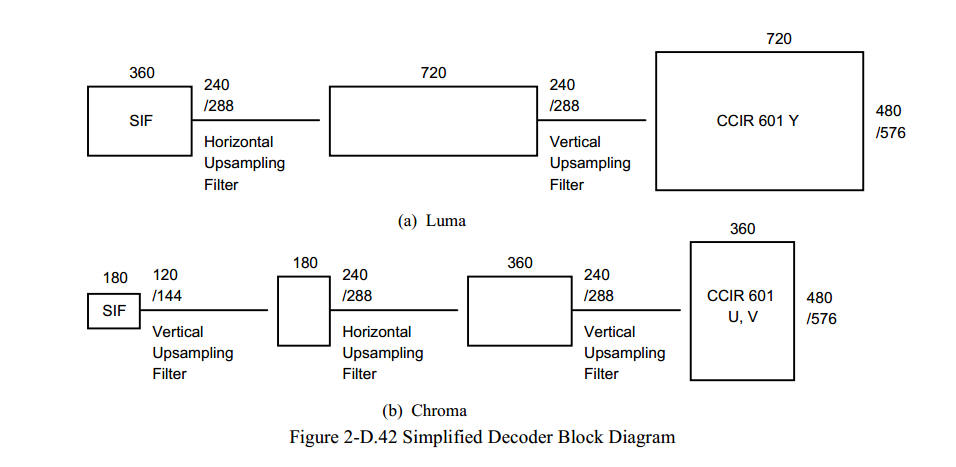
Τα αποτελέσματα της συνάρτησης είναι όπως θα τα περιμέναμε. Τα μεγέθη των Cr και Cb είναι διαφορετικά από του frame Y, και γι' αυτό το λόγο μειώνουμε το μέγεθος του τελευταίου στο μισό, ώστε να δούμε ενδεικτικά πως φαίνεται το συνολικό αποτέλεσμα, χωρίς να διατηρούμε αυτό το μέγεθος για τη συνέχεια.



Για την αντίστροφη συνάρτηση χρησιμοποιούνται τα εξής FIR φίλτρα:

* για την φωτεινότητα: [-12, 0, 140, 256, 140, 0, -12]// 256
* για τις χρωματικότητες: [1, 0, 3, 0, 3, 0, 1] // 8

Στη συνέχεια, ακολουθούνται διαδικασίες upsampling και interpolation με τη σειρά που φαίνονται στο παρακάτω σχήμα:



Στο τέλος, υπολογίζονται οι τιμές των R, G και B καναλιών μέσω των Y, Cb και Cr.

Το αποτέλεσμα που προκύπτει είναι η παρακάτω εικόνα, η οποία είναι σχεδόν πανομοιότυπη με την αρχική εικόνα.



## 1.2 Motion Estimation

Για την παρουσίαση της λειτουργίας της συνάρτησης Motion Estimation, η εφαρμογή γίνεται στην εικόνα “coastguard003.tiff” και συγκεκριμένα στο macroblock 0. Ως frame αναφοράς χρησιμοποιείται η εικόνα “coastguard001.tiff”. Το Motion Estimation εφαρμόζεται μόνο στο κανάλι Y, καθως εκεί είναι πιο ουσιώδης η ανίχνευση της κίνησης και όχι στα κανάλια χρώματος.

Το macroblock είναι μεγέθους 16x16 στο Y και 8x8 στα Cr και Cb. Για το λόγο αυτό, "κόβουμε" κάποια pixel από την δεξιά και αριστερή πλευρά της εικόνας, ώστε να χωράει ακέραιος αριθμός 16x16 τετραγώνων στην εικόνα, και πιο συγκεκριμένα το frameY γίνεται 352x288, με αποτέλεσμα να μπορούν να μπορεί να χωριστεί η εικόνα σε 22x18 = 396 macroblocks.

Στην συγκεκριμένη περίπτωση για το estimation χρησιμοποιήσαμε για λόγους ευκολίας και έλλειψης χρόνου Exhaustive Search. Αυτό σημαίνει ότι η συνάρτηση μας ψάχνει γύρω από το macroblock σε απόσταση από -7 έως +7 pixels προς κάθε κατεύθυνση, κάτι το οποίο είναι όμως υπολογιστικά αργό. Θα μπορούσαν να υλοποιηθούν και άλλοι αλγόριθμοι, όπως π.χ. Λογαριθμική αναζήτηση για βελτιστοποίηση της αναζήτησης αυτής.

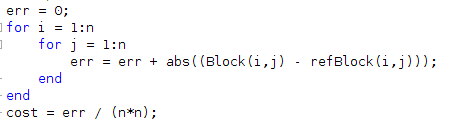
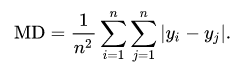
Στην motEstP κάθε macroblock προβλέπεται με αναφορά στο προηγούμενο I ή το προηγούμενο P frame, ανάλογα με το ποιό είναι πιο πρόσφατο. Στην συγκεκριμένη υλοποίηση της συνάρτησης όμως δεν υπάρχει καμία διαφοροποίηση είτε πρόκεται για I είτε για P πλαίσιο,

Το σφάλμα Ε εδώ προκύπτει από την σχέση: Εt+1 = Bt+1 – Mt, όπου Bt+1 το macroblock στο οποίο εφαρμόζουμε motion estimation και Mt το frame που προσεγγίζει καλύτερα το προηγούμενο.

Στην motEstB χρησιμοποιείται ως αναφορά θεωρούνται ταυτόχρονα το προηγούμενο I ή P πλαίσιο, καθώς και το επόμενο I ή P πλαίσιο.

Το σφάλμα Ε εδώ προκύπτει από την σχέση: Εt+1 = Bt+1 – (½) (Mt- + Mt+), όπου Bt+1 το macroblock στο οποίο εφαρμόζουμε motion estimation και Mt- και Mt+ τα frame που προσεγγίζουν καλύτερα το Bt+1 από την αναζήτηση στο προηγούμενο και επόμενο frame αντίστοιχα.

Στις δύο αυτές περιπτώσεις η συνάρτηση που χρησιμοποιείται για την εκτίμηση του σφάλματος είναι η Mean Absolute Difference (MAD), η οποία περιγράφεται στο αρχείο costFunction.m.



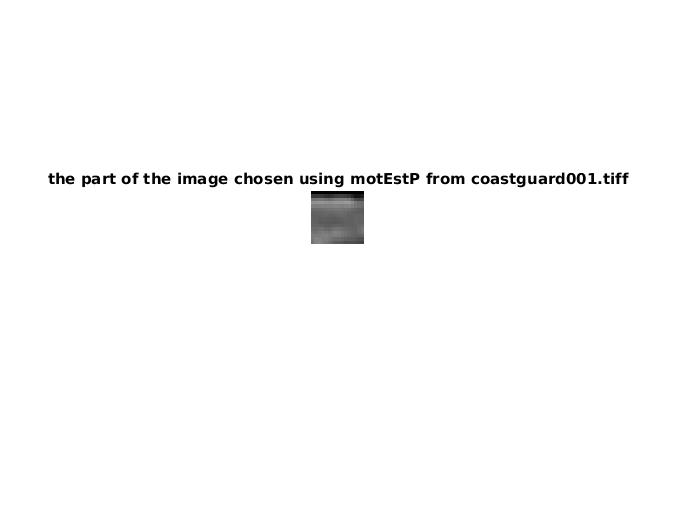
Η συνάρτηση αυτή υπολογίζει τη μέση τιμή των απόλυτων τιμών των διαφορών ανάμεσα στα pixels των macroblock. Αρχικά, αθροίζεται η απόλυτη τιμή των διαφορών και στο τέλος διαιρούνται με το πλήθος των στοιχείων n.

Το Motion Vector αποτελεί έναν πίνακα 2x2, ο οποίος μας δείχνει προς ποια κατεύθυνση μετατοπίστηκε το mactoblock μας ως προς το macroblock αναφοράς του προηγούμενου frame (1η στήλη) και ως προς του επόμενο frame (2η στήλη, εάν υπάρχει).

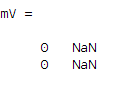
Πιο συγκεκριμένα, παρακάτω φαίνεται το macroblock 0 της εικόνας “coastguard003.tiff” στο οποίο εφαρμόζουμε την motEstP.



Μετά την εκτέλεση του αλγορίθμου, το macroblock που μας λέει ο αλγόριθμος ότι ταιριάζει περισσότερο από την εικόνα “coastguard001.tiff" φαίνεται παρακάτω:

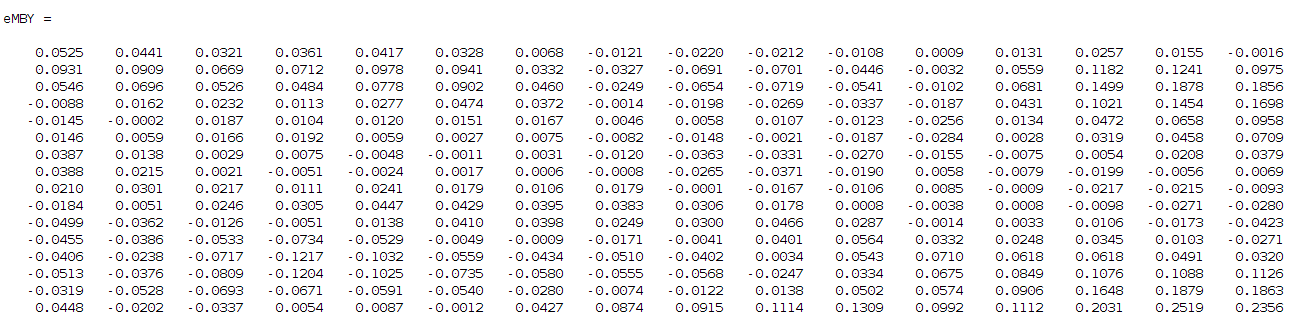


Βλέπουμε ότι όντως οι εικόνες αυτες είναι πάρα πολύ ίδιες. Το macroblock αυτό σύμφωνα με το αποτέλεσμα της συνάρτησης δινεται από το motion vector:

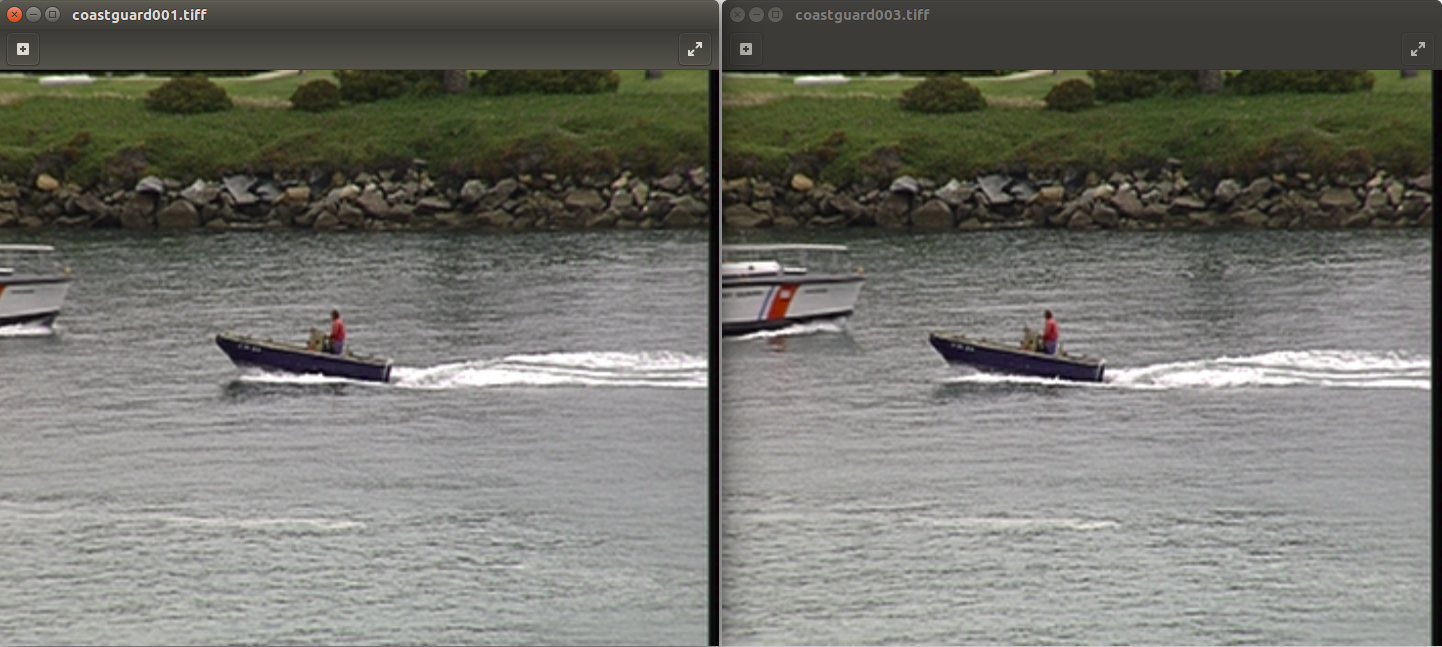


Δηλαδή από την ίδια θέση στην οποία ήταν και το macroblock στο οποίο εφαρμόσαμε την motEstP.

Το σφάλμα το οποίο προκύπτει ανάμεσα στα δύο αυτά macroblock και θα κωδικοποιηθεί στη συνέχεια είναι το εξής:

Βλέπουμε προφανώς ότι οι τιμές αυτές είναι πολύ μικρές και είναι οι χαμηλότερες που βρέθηκαν συγκριτικα με τα υπόλοιπα macroblock που βρίσκονται γύρω από το macroblock 0 που μελετήσαμε.

Παρακάτω βλεπουμε και τις δύο αυτές εικόνες, όπου μπορούμε να επιβεβαιώσουμε το αποτέλεσμα της συνάρτησης μας, καθώς τα πρώτα 16x16 pixels της εικόνας δεν έχουν αλλάξει αρκετά μεταξύ των δύο εικόνων.



Με την εφαρμογή της συνάρτησης και σε άλλα macroblock της εικόνας, παρατηρούμε ότι υπάρχουν διάφορα αποτελέσματα και προκύπτουν macroblock τα οποία κινούνται αρκετά μέσα στην εικόνα ή άλλα που παραμένουν σχετικά αμετακίνητα.

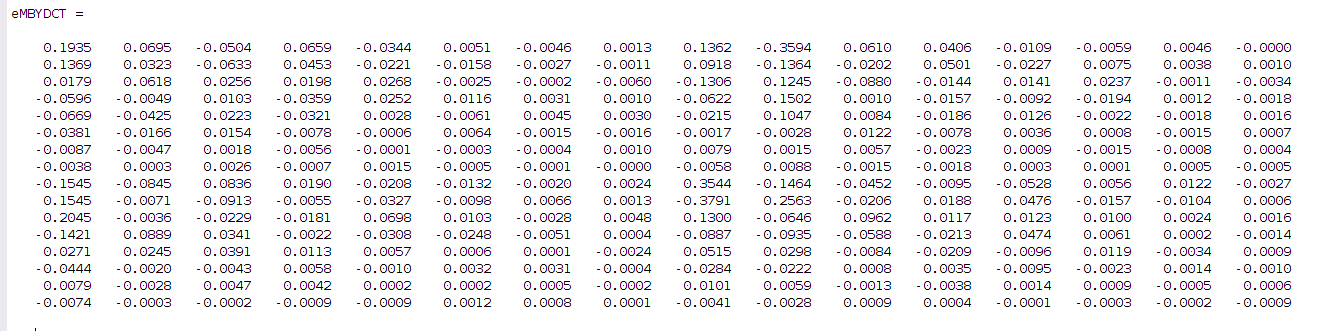
Οι αντίστροφες συναρτήσεις iMotEstP και iMotEstB εφαρμόζουν ακριβώς την αντίστροφη διαδικασία, καθως δέχονται ως όρισμα το motion Vector και το σφάλμα πρόβλεψης και επιστρέφουν το αρχικό macroblock. Η πρώτη, εφαρμόζει την αντιστροφή χωρίς κανένα σφαλμα, καθώς πάντα το αρχικό macroblock ταυτίζεται με το reconstructed, ενώ στην δεύτερη δημιουργείται ένα πολύ μικρό σφάλμα της τάξης 10-15, το οποίο προκύπτει από την ακρίβεια στις πράξεις του MATLAB και θεωρητικά δεν επηρεάζει σχεδόν καθόλου την διαδικασία της αποκωδικοποίησης.

## 1.3 Μετασχηματισμός DCT

Όλα τα πλαίσια τύπου I και τα σφάλματα που προέκυψαν για τα P και B frames στη συνέχεια κωδικοποιούνται μέσω του Διακριτού Μετασχηματισμού Συνημιτόνου (DCT). Ο μετασχηματισμός αυτός εφαρμόζεται σε κάθε block μεγέθους 8x8.

Για την υλοποίηση του DCT και του αντιστρόφου του στη συγκεκριμένη εργασία χρησιμοποιήθηκαν οι συναρτήσεις dct2 και idct2 του MATLAB.

Εφαρμόζουμε τον μετασχηματισμό αυτό στο αποτέλεσμα της motion estimation που εφαρμόστηκε στο macroblock 0 που αναφέρθηκε και πριν, στο κανάλι Y και παίρνουμε το παρακάτω αποτέλεσμα:



Με την εφαρμογή της αντίστροφης συνάρτησης DCT επιστρέφουμε πάλι στο αρχικό αποτέλεσμα.

## 1.4 Κβαντισμός

Οι συντελεστές που προέκυψαν από τον Μετασχηματισμό DCT στη συνέχεια περνούν μέσα από έναν κβαντιστή, ο οποίος χαρακτηρίζεται από τον πίνακα κβαντισμού και από την κλίμακα κβαντισμού.

Ο κβαντισμός γίνεται με διαφορετικό τρόπο στα I-frames και με διαφορετικό στα P και B, καθώς στα πρώτα ο DC συντελεστής (το πρώτο στοιχείο πάνω και αριστερά) κβαντίζεται ανεξάρτητα από την τιμή της κλίμακας κβαντισμού και απλά διαιρείται με το 8. Σε κάθε διαίρεση που πραγματοποιείται στις συναρτήσεις αυτές, χρησιμοποιείται επιπλέον και η συνάρτηση *fix*, η οποία στρογγυλοποιεί στον πιο κοντινό ακέραιο.

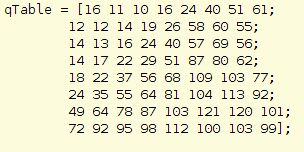
Δηλαδή κάθε DCT συντελεστής πολλαπλασιάζεται με το 8 και στη συνέχεια διαιρείται με το γινόμενο της κλίμακας κβαντισμού επί το αντίστοιχο στοιχείου του πίνακα κβαντισμού.

Βλέπουμε πως όταν η κλίματα κβαντισμού είναι 8, ουσιαστικά πραγματοποιειται μόνο η διαίρεση με το στοιχείο του πίνακα κβαντισμού, και εάν ο πίνακας αυτός περιέχει μόνο 1, τοτε πρόκειται για lossless compression.

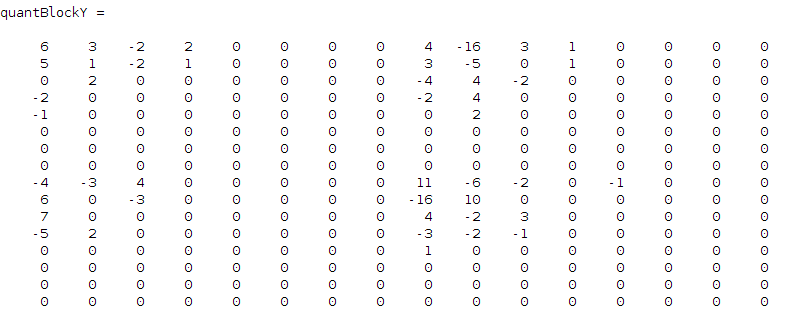
Με τον τρόπο αυτό, όλες οι τιμές πλέον ανήκουν σε κάποια από τις ζώνες του κβαντιστή και δεν αποτελούν αριθμούς, αλλά σύμβολα. Βέβαια στο MATLAB συνεχίζουν να έχουν την μορφή του αριθμού για ευκολότερη διαχείριση τους.

Η αντίστροφη διαδικασία σίγουρα επιφέρει κάποιο σφάλμα, καθώς η στρογγυλοποίηση στον κοντινότερο ακεραιο δεν μπορεί να αναιρεθεί.

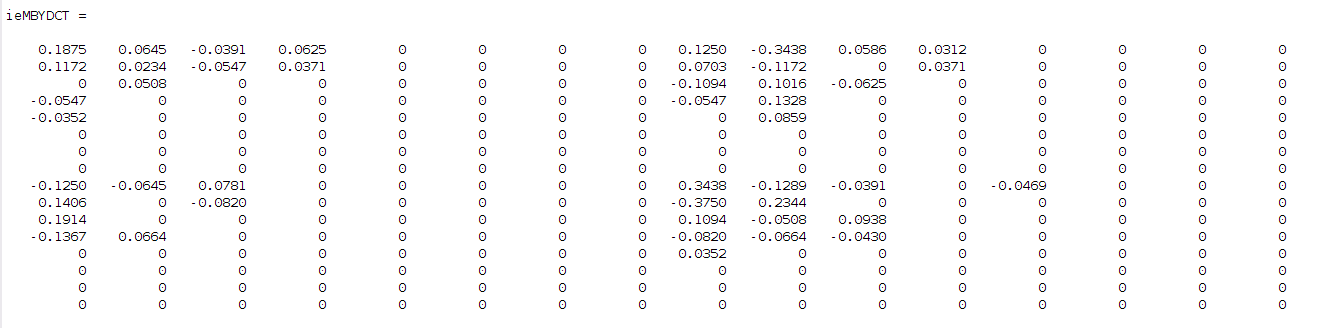
Εφαρμόζουμε τον κβαντισμό στο αποτέλεσμα της προηγούμενης συνάρτησης με πίνακα κβαντισμού τον *IJG Standard Table:*



Το αποτέλεσμα που προκύπτει φαίνεται στον παρακάτω πίνακα:

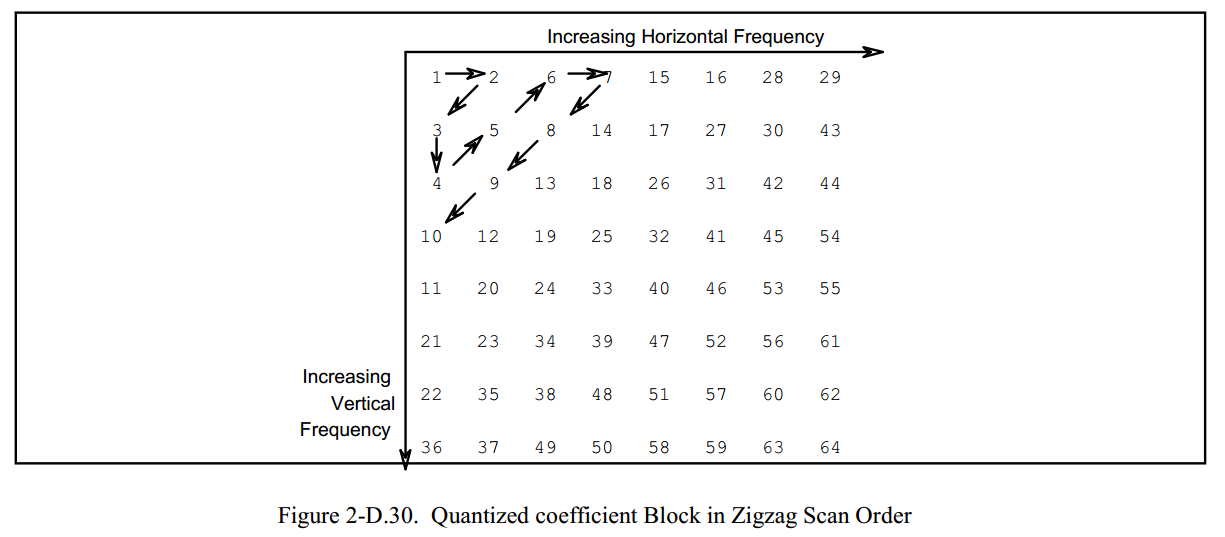


Κατά την αντιστροφή της διαδικασίας όμως βλέπουμε πως όπου προέκυψαν μηδενικά, παραμένουν μηδενικά ενώ τα υπόλοιπα στοιχεία αποκτούν τιμές.



## 1.5 Zig-zag scanning και RLE

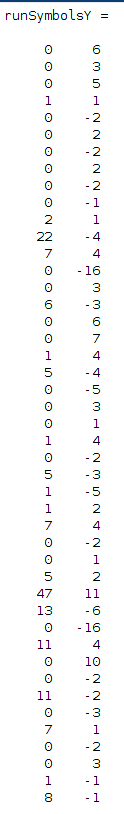
Κάθε block μεγέθους 8x8 που έχει προκύψει μετά τον κβαντισμό και περιέχει τα σύμβολα του κβαντιστή, μετατρέπεται σε ένα μονοδιάστατο διάνυσμα μήκους 64 μέσω του zig-zag scanning, ο οποίος περιγράφεται καλύτερα από το παρακάτω σχήμα:



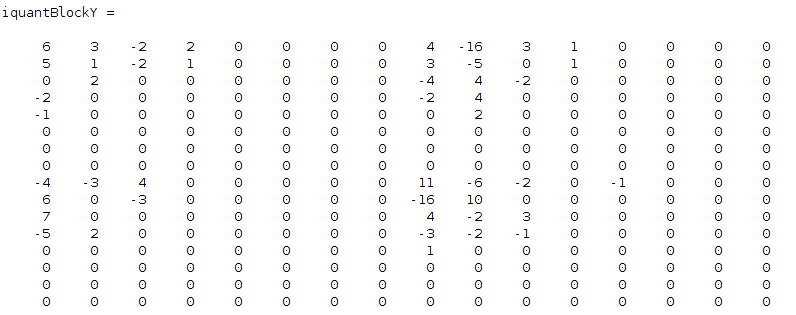
Στη συνέχεια, σε κάθε τέτοιο διάνυσμα εφαρμόζεται Run Length Encoding. Αποτέλεσμα αυτού είναι ένας πίνακας μεγέθους Rx2, ο οποίος περιέχει ζευγάρια της μορφής (precedingZeros, quantSymbol), τόσα ώστε να περιγράφεται πλήρως ολόκληρος ο πίνακας. (R είναι τα μήκη διαδρομής που εντοπίστηκαν στο block)

Μια διαφοροποίηση σε σχέση με το πρότυπο MPEG-1 είναι ότι εδώ περιλαμβάνεται και ο DC συντελεστής στην διαδικασία κωδικοποίησης RLE.

Στο αποτέλεσμα της προήγουμενης συνάρτησης, εφαρμόζουμε τον RLE και προκύπτουν τα παρακάτω ζεύγη της μορφής που έχει προαναφερθεί:



Η αντίστροφη διαδικασία με τη χρήση της *iRunLength* επαναφέρει τον πίνακα στην προηγούμενη του μορφή:



## 1.6 Κωδικοποίηση Μεταβλητού Μήκους Λέξης (Variable Length Coding)

Μετά την διαδικασία της RLE κωδικοποίησης, κάθε ένα από τα σύμβολα μήκους διαδρομής που προέκυψαν κωδικοποιούνται σε ένα stream από 0 και 1, με βάση τους πίνακες 2-D.15, 2-D.16 και 2-D.17 του προτύπου. Το μήκος του τελικού stream είναι μεταβλητό αναλογα με τα ζευγάρια που προέκυψαν.

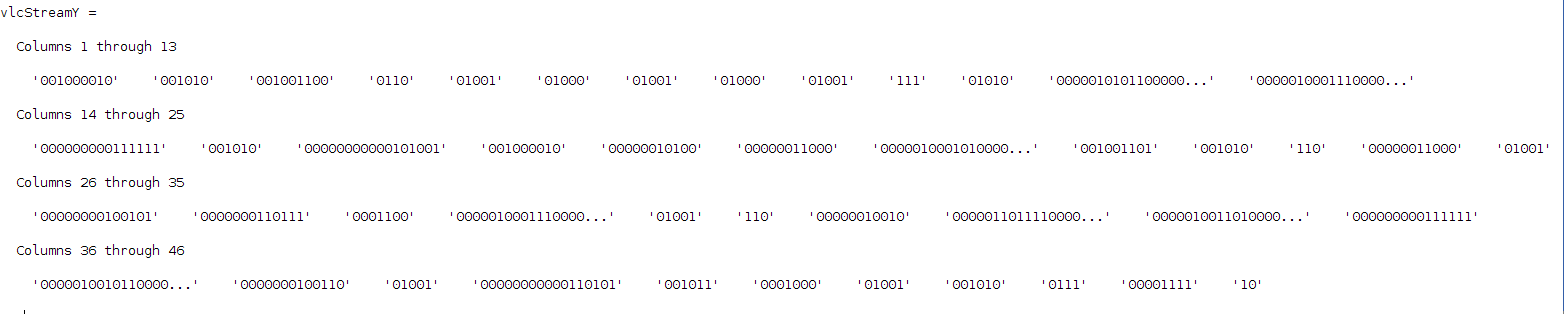
Επόμενως, η λειτουργία που επιτελεί αυτή η συνάρτηση είναι αρχικά να ελέγχει αν υπάρχει το σύμβολο μήκους διαδρομής στον πίνακα d15a, και εάν όντως υπάρχει να ορίζει το τελευταίο ψηφίο του VLC με βάση το πρόσημο του. Εαν δεν υπάρχει, τότε προστίθεται ένα escape code στο stream και με αναζήτηση στους άλλους δύο πίνακες, αρχικά για το Run και μετά για το Level, κωδικοποιείται το σύμβολο μήκους διαδρομής. Στο τέλος του block, πάντα προστίθεται η τιμή '10' που συμβολίζει το End Of Block.

Στο MATLAB καθε σειρά από 0 και 1 που αντιστοιχεί σε ένα σύμβολο μήκους διαδρομής, αποθηκεύεται σε ένα ξεχωριστό κελί ενός cell array, ώστε να είναι πιο εύκολη η αποκωδικοποίηση του, ενώ είναι επίσης εύκολη η μετατροπή του σε ένα char μεγάλου μήκους.

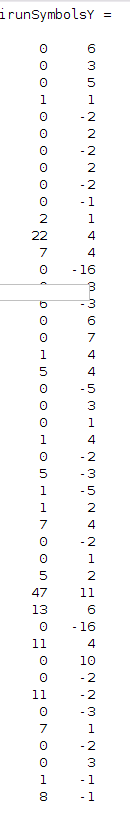
Η συνάρτηση ivlc εκτελεί ακριβώς την αντίστροφη διαδικασία, αναζητώντας στους ίδιους πίνακες για την υπάρξη των συμβόλων αυτών και γράφοντας τα ανάλογα σύμβολα μήκους διαδρομής με το σωστό πρόσημο.

Οι πίνακες h δεν χρησιμοποιήθηκαν καθόλου για την λειτουργία του VLC.

Εφαρμόζουμε λοιπόν VLC στο αποτέλεσμα της προηγούμενης συνάρτησης και προκύπτει το εξής stream:



Με την αντίστροφη συνάρτηση *ivlc* δημιουργούνται παλι τα ίδια ζευγάρια με runSymbols:



Άρα και αυτη η συνάρτηση επιφέρει τα αναμενόμενα αποτελέσματα.

## Demonstration

Στο αρχείο *demo1.m* εφαρμόζονται στο macroblock 0 της εικόνας “coastguard003.tiff” με τη σειρά όλες οι συναρτήσεις μέχρι να δημιουργηθεί το vlc stream και μετά εφαρμόζονται με την ανάποδη σειρά οι αντιστροφές συναρτήσεις. Παρατηρούμε ότι όλες οι συναρτησεις λειτουργούν σωστά, καθώς η είσοδος και η έξοδος που φαίνεται στη συνέχεια σχεδόν ταυτίζονται.

