



Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Πολυτεχνική Σχολή

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Τομέας Ηλεκτρονικής και Υπολογιστών

ΣΥΣΤΗΜΑΤΑ ΠΟΛΥΜΕΣΩΝ

Εργασία : Απλοποιημένος codec JPEG

Χειμερινό εξάμηνο 2023/24

Δεϊρμεντζόγλου Ιωάννης
Τομέας Ηλεκτρονικής και Υπολογιστών
Α.Ε.Μ.: 10015
Email: deirmentz@ece.auth.gr

Περιεχόμενα

Εισαγωγή.....	3
Προεπεξεργασία.....	5
Διακριτός μετασχηματισμός συνημίτονου (DCT).....	6
Κβαντισμός.....	8
Zig-zag scanning και RLE.....	9
Κωδικοποίηση - Αποκωδικοποίηση Huffman.....	12
Demo 1	15
JPEG Encoder/Decoder	17
Καταγραφή Αποτελεσμάτων.....	19
Demo 2	32

Εισαγωγή

Σκοπός της προαιρετικής εργασίας στο πλαίσιο του μαθήματος είναι η υλοποίηση ενός κωδικοποιητή και αποκωδικοποιητή εικόνας σύμφωνα με το πρότυπο JPEG (ISO/IEC 10918-1:1994) στην εκδοχή του baseline sequential DCT-based, το οποίο παρέχεται στην εκφώνηση, με όσο το δυνατόν λιγότερες αποκλίσεις από το πρότυπο.

Το JPEG αποτελεί ένα πρότυπο συμπίεσης εικόνας που χρησιμοποιείται για την αποθήκευση εικόνων σε συμπιεσμένη μορφή. Η αξιοσημείωτη ποιότητα του JPEG είναι ότι επιτυγχάνει υψηλές αναλογίες συμπίεσης με μικρή απώλεια ποιότητας. Δηλαδή, επιτρέπει τη μείωση του όγκου δεδομένων της εικόνας χωρίς σημαντική απώλεια ποιότητας, καθιστώντας το JPEG ιδανικό για την αποθήκευση και τη μετάδοση ψηφιακών εικόνων με μια ισορροπία μεταξύ ποιότητας και μεγέθους αρχείου.

Ο βαθμός συμπίεσης μπορεί να ποικίλει ανάλογα με τις ανάγκες της εφαρμογής. Η μορφή JPEG είναι αρκετά δημοφιλής και χρησιμοποιείται σε πολλές συσκευές όπως ψηφιακές φωτογραφικές μηχανές και αποτελεί επίσης τη μορφή που επιλέγεται για την ανταλλαγή εικόνων μεγάλου μεγέθους σε περιορισμένου εύρους ζώνης περιβάλλον, όπως το Διαδίκτυο.

Η μορφή JPEG είναι αρκετά δημοφιλής και χρησιμοποιείται σε πολλές συσκευές όπως ψηφιακές φωτογραφικές μηχανές και αποτελεί επίσης τη μορφή που επιλέγεται για την ανταλλαγή εικόνων μεγάλου μεγέθους σε περιορισμένου εύρους ζώνης περιβάλλον, όπως το Διαδίκτυο. Ο αλγόριθμος JPEG είναι κατάλληλος για φωτογραφίες και πίνακες ρεαλιστικών σκηνών με ομαλές παραλλαγές τόνου και χρώματος, ενώ δεν προτιμάται σε εικόνες με πολλές άκρες και έντονες παραλλαγές καθώς αυτό μπορεί να οδηγήσει σε πολλές αλλοιώσεις στην προκύπτουσα εικόνα. Σε αυτές τις περιπτώσεις είναι καλύτερο να χρησιμοποιούνται χωρίς απώλειες μορφές όπως PNG, TIFF ή GIF. Για το λόγο αυτό το JPEG πρότυπο δεν χρησιμοποιείται σε ιατρικές και επιστημονικές εφαρμογές όπου η εικόνα πρέπει να αναπαράγει τα ακριβή δεδομένα όπως έχουν ληφθεί, μιας και η παραμικρή απώλεια μπορεί να οδηγήσει σε λάθος εκτιμήσεις.

Μία εικόνα JPEG ενδέχεται να υποστεί περαιτέρω απώλειες εάν επεξεργάζεται συχνά και στη συνέχεια αποθηκεύεται. Η διαδικασία της αποσυμπίεσης και επανασυμπίεσης μπορεί να υποβαθμίσει περαιτέρω την ποιότητα της εικόνας. Για να ξεπεραστεί αυτό το πρόβλημα, η εικόνα θα πρέπει να επεξεργαστεί και αποθηκευτεί σε μορφή χωρίς απώλειες και να μετατραπεί σε μορφή JPEG λίγο πριν την τελική μετάδοση στο επιθυμητό μέσο. Με τον τρόπο αυτό, εξασφαλίζονται οι ελάχιστες απώλειες λόγω συχνής αποθήκευσης. Τα αρχεία εικόνων που

αποθηκεύονται με μορφή JPEG συχνότερα χρησιμοποιούν τις επεκτάσεις .jpg ή .jpeg.

Η απλοποιημένη έκδοση του JPEG, γνωστή ως "baseline sequential DCT-based", αποτελεί την πιο βασική και ευρέως χρησιμοποιούμενη μορφή του πρωτοκόλλου JPEG. Στην εκδοχή "baseline sequential DCT-based" του προτύπου JPEG, η επεξεργασία της εικόνας γίνεται ανά τμήμα 8x8 πίξελ μέσω του μετασχηματισμού διακριτής μετασχηματισμού συνημίτονου (Discrete Cosine Transform - DCT). Αυτό σημαίνει ότι η εικόνα χωρίζεται σε μπλοκ 8x8 πίξελ και για κάθε ένα από αυτά τα μπλοκ εκτελείται ο μετασχηματισμός DCT.

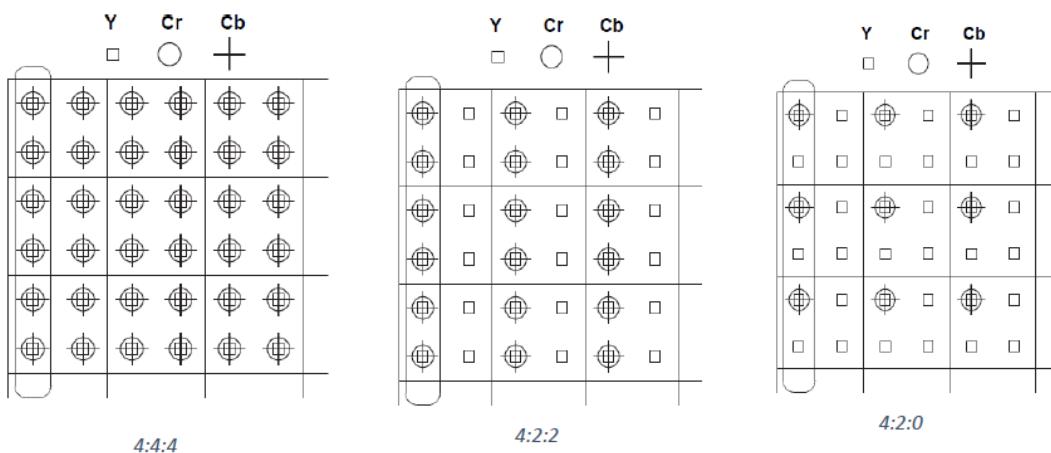
Οι τεχνικές συμπίεσης JPEG βασίζονται σε μια σειρά από διαδικασίες, μεταξύ των οποίων περιλαμβάνεται ένα στάδιο προεπεξαργασίας ο μετασχηματισμός σε διακριτό συνημίτονο (Discrete Cosine Transform - DCT), ο κβαντισμός, η αναδιάταξη (reordering) και η κωδικοποίηση Huffman.

Προεπεξεργασία

Στόχος του JPEG είναι να συμπιέσει μια εικόνα με τρόπο τέτοιο που να εισάγει τη λιγότερο πιθανή παραμόρφωση στην ανθρώπινη αντίληψη. Επομένως, το πρώτο βήμα είναι η μετατροπή της εικόνας από το χρωματικό μοντέλο RGB στο YCbCr μοντέλο που διαθέτει επίσης τρία κανάλια. Το κανάλι Y αντιπροσωπεύει τη φωτεινότητα ενός pixel και τα κανάλια Cb και Cr τον χρωματισμό (χωρισμένα σε μπλε και κόκκινα στοιχεία). Το χρωματικό μοντέλο YCbCr επιτρέπει μεγαλύτερη συμπίεση χωρίς σημαντική επίδραση στην ποιότητα της εικόνας. Η συμπίεση είναι πιο αποδοτική καθώς οι πληροφορίες που αφορούν τη φωτεινότητα, οι οποίες είναι και οι πιο σημαντικές για την τελική αντίληψη της ποιότητας της εικόνας, περιορίζονται σε ένα μόνο κανάλι.

Η υποδειγματοληψία είναι ένα προαιρετικό βήμα στη συμπίεση JPEG, το οποίο εισήχθη λόγω της ανατομίας των ανθρώπινων ματιών. Ενώ υπάρχει πολύ μεγάλη αντίληψη της φωτεινότητας από τον οφθαλμό, η αντίληψη της απόχρωσης και του κορεσμού του χρώματος είναι σημαντικά μικρότερη. Χρησιμοποιώντας την πληροφορία αυτή, η συμπίεση μπορεί να γίνει ακόμα πιο αποτελεσματική. Με την υποδειγματοληψία γίνεται μείωση της χωρικής ανάλυσης των στοιχείων Cb και Cr. Οι μορφές υποδειγματοληψίας για εικόνες JPEG παρουσιάζονται παρακάτω. Όταν πραγματοποιείται υποδειγματοληψία (εκτός από την περίπτωση 4:4:4), υπάρχει απώλεια πληροφορίας. Υποστηρίζονται τρεις περιπτώσεις υποδειγματοληψίας:

- I. **4: 4: 4**, όπου οι τρεις συνιστώσες δειγματοληπτούνται με τον ίδιο πίνακα πλέγματος.
- II. **4: 2: 2**, όπου οι συνιστώσες χρωματικότητας υποδειγματοληπτούνται οριζόντια και η ανάλυση τους μειώνεται κατά το ήμισυ σε σύγκριση με την φωτεινότητα.
- III. **4: 2: 0**, όπου οι συνιστώσες χρωματικότητας υποδειγματοληπτούνται με συντελεστή 2, τόσο οριζόντια όσο και κατακόρυφα. Έτσι η ανάλυση τους μειώνεται στο 1/4 σε σύγκριση με την φωτεινότητα.



Σχήμα 1 : Χωροχρονική Δειγματοληψία

- **convert2ycbcr (imageRGB, subimg)**

Η συναρτηση δέχεται δύο ορίσματα: την εικόνα σε μορφή RGB και έναν πίνακα υποδειγματοληψίας (subimg), ο οποίος περιγράφει την υποδειγματοληψία των χρωματικών συνιστωσών στον χρωματικό χώρο YCbCr. Επιστρέφει τις τρεις συνιστώσες Y, Cb και Cr της μετατραπείσας εικόνας. Η συναρτηση αρχικά ελέγχει την εγκυρότητα των εισόδων και εξασφαλίζει ότι οι διαστάσεις της είναι πολλαπλάσια του 8 και σε περιπτιωση που δεν είναι αφαιρούνται οι οριακές γραμμές και στήλες μέχρι να ικανοποιηθεί αυτή η συνθήκη όπως περιγράφεται από την εκφώνηση. Στην συνέχεια , πραγματοποιεί την μετατροπή της εικόνας από RGB σε YCbCr , σύμφωνα με τον προκαθορισμένο πίνακα υποδειγματοληψίας :

$$T = \begin{matrix} 0.299 & 0.587 & 0.114 \\ -0.1687 & -0.3313 & 0.5 \\ 0.5 & -0.4187 & -0.0813 \end{matrix}$$

Κάθε κανάλι (Y, Cb, Cr) υπολογίζεται χρησιμοποιώντας αυτόν τον πίνακα, και στη συνέχεια προστίθεται μια σταθερή τιμή (0 ή 128) για την εύρεση του τελικού αποτελέσματος στο εύρος [0, 255]. Τέλος , με βάσει τον πίνακα υποδειγματοληψίας αφού ελεγχθεί εάν αντιστοιχεί σε κάποια από τις υποστηριζόμενες μορφές υποδειγματοληψίας, η εικόνα υποδειγματολειπτείται, όπως περιεγράφηκε παραπάνω.

- **convert2rgb (imageY, imageCr, imageCb, subimg)**

Η συνάρτηση convert2rgb δέχεται ως είσοδο τα κανάλια Y, Cb και Cr στο χρωματικό χώρο YCbCr, καθώς και έναν πίνακα (subimg) που περιγράφει την μορφή υποδειγματοληψίας. Η συναρτηση αυτή επιστρέφει την ανακατασκευασμένη RGB εικόνα. Αρχικά, ελέγχει την εγκυρότητα της μορφής υποδειγματοληψίας. Στην συνέχεια, ο αντίστροφος πίνακας μετασχηματισμού (invT) χρησιμοποιείται για την μετατροπή από τον χρωματικό χώρο YCbCr πίσω στον χρωματικό χώρο RGB. Έπειτα, εφαρμόζει την δειγματοληψία ανάλογα με την μορφή που έχει επιλεγεί με παρόμοια λογική.

Διακριτός μετασχηματισμός συνημίτονου (DCT)

Ο διακριτός μετασχηματισμός συνημίτονου αποτελεί ένα από τα πιο βασικά δομικά στοιχεία για την συμπίεση JPEG. Ο DCT σχετίζεται με την ανάλυση Fourier όπου οι συναρτήσεις του χρόνου μπορούν να αποσυντεθούν στις συχνότητές τους. Μελέτη του ανθρώπινου ματιού αποκάλυψε ότι αυτό είναι καλό στην ανίχνευση της διακύμανσης της φωτεινότητας σε μια ευρεία περιοχή αλλά λιγότερο ευαίσθητο σε αλλαγές σε μια μικρή περιοχή, δηλαδή ότι η ευαισθησία μειώνεται με τις πληροφορίες συχνότητας. Ο DCT χρησιμοποιείται για τη μεταφορά των 8x8 μπλοκ εικόνων από το πεδίο του χώρου στο πεδίο της συχνότητας. Μια εικόνα συνεχούς τόνου μπορεί να

αναπαρασταθεί με μια σειρά από πλάτη, για κάθε συστατικό χρώματος, σε χώρο δύο διαστάσεων.

Πριν από τον υπολογισμό ενός DCT του μπλοκ, οι τιμές του μετατοπίζονται από ένα θετικό εύρος σε ένα με επίκεντρο το μηδέν. Για μια εικόνα 8-bit, κάθε καταχώριση στο αρχικό μπλοκ εμπίπτει στο εύρος [0,255]. Το μεσαίο σημείο του εύρους (στην περίπτωση αυτή, η τιμή 128) αφαιρείται από κάθε καταχώριση για να παράγει ένα εύρος δεδομένων που είναι κεντραρισμένο στο μηδέν, έτσι ώστε το τροποποιημένο εύρος να είναι [-128,127]. Αυτό το βήμα μειώνει τις απαιτήσεις δυναμικού εύρους στο στάδιο επεξεργασίας DCT που ακολουθεί. Η διαδικασία DCT σε σύστημα συμπίεσης εικόνας JPEG ξεκινά με μπλοκ δεδομένων εικόνας 8x8, $f(x, y)$. Αυτό το μπλοκ μπορεί να μετατραπεί σε ένα νέο μπλοκ 8x8, $F(x, y)$, με τον διακριτό μετασχηματισμό συνημίτονου (DCT). Το αρχικό μπλοκ $f(x, y)$ μπορεί να ληφθεί με τον αντίστροφο διακριτό μετασχηματισμό συνημίτονου (IDCT).

- **dctBlock = blockDCT(block)**

Η συνάρτηση blockDCT δέχεται ένα μπλοκ εικόνας ως είσοδο και υπολογίζει τους συντελεστές DCT για αυτό το μπλοκ. Αρχικά, γίνεται έλεγχος για την εγκυρότητα των διαστάσεων του μπλοκ. Στη συνέχεια, οι τιμές του μπλοκ μετατοπίζονται και υπολογίζονται οι συντελεστές DCT χρησιμοποιώντας την συνάρτηση DCT της NumPy. Η μετατόπιση επιπέδου επιτυγχάνεται με την αφαίρεση μιας τιμής από κάθε δείγμα, και η τιμή προς αφαίρεση καθορίζεται από την παράμετρο ακρίβειας. Συγκεκριμένα, η τιμή προς αφαίρεση υπολογίζεται ως 2^{P-1} , οπου P αντιπροσωπεύει την παράμετρο ακρίβειας. Οι συντελεστές DCT επιστρέφονται ως αποτέλεσμα.

- **block = iBlockDCT (dctBlock)**

Η συνάρτηση iBlockDCT υλοποιεί τον αντίστροφο Διακριτό Μετασχηματισμό Συνημίτονου (IDCT) για ένα μπλοκ των συντελεστών DCT. Αρχικά, πραγματοποιεί έναν έλεγχο για να διασφαλίσει ότι ο πίνακας $dctBlock$ έχει τις αναμενόμενες διαστάσεις 8x8. Στην συνέχεια, χρησιμοποιώντας την συνάρτηση IDCT της βιβλιοθήκης NumPy, υπολογίζεται ο αντίστροφος μετασχηματισμός DCT για τον πίνακα $dctBlock$. Έπειτα, από τις τιμές του μπλοκ αναιρείται η προηγούμενη μετατόπιση που πραγματοποιήθηκε πριν από την εκτέλεση του DCT. Η συνάρτηση επιστρέφει τον αντίστροφο DCT block μετά την αναίρεση της μετατόπισης, περιλαμβάνοντας τις αρχικές τιμές της εικόνας.

Κβαντισμός

Το ανθρώπινο μάτι εύκολα εντοπίζει μικρές διαφορές στη φωτεινότητα σε μια σχετικά μεγάλη περιοχή, αλλά δεν διακρίνει με ευκολία την ακριβή ισχύ των διακυμάνσεων φωτεινότητας σε υψηλές συχνότητες. Αυτό επιτρέπει τη σημαντική μείωση του όγκου των πληροφοριών στις περιοχές υψηλής συχνότητας. Ένα από τα κυριά βήματα συμπίεσης του κωδικοποιητή JPEG είναι η κβάντιση των συντελεστών DCT. Μετά την εκτέλεση του DCT παράγεται ένα block (διάστασης 8×8), όπου οι συντελεστές που αντιστοιχούν σε χαμηλές συχνότητες «μαζεύονται» πάνω και αριστερά, στην αρχή του block. Κάθε συντελεστής μειώνεται κατά έναν σταθερό παράγοντα και στρογγυλοποιείται στην πλησιέστερη ακέραιη τιμή. Ως αποτέλεσμα αυτού, πολλοί από τους συντελεστές υψηλότερης συχνότητας στρογγυλοποιούνται στο μηδέν και πολλοί από τους υπόλοιπους γίνονται μικροί θετικοί ή αρνητικοί αριθμοί, οι οποίοι χρειάζονται πολύ λιγότερα bit για να αποθηκευτούν. Έτσι, γίνεται να θυσιαστούν οι υψησυχνες λεπτομέρειες για περαιτέρω συμπίεση χωρίς να επηρεαστεί η ποιότητα της εικόνας.

Για τον κβαντισμό, χρησιμοποιείται ο εξής τύπος σε κάθε block :

$$\text{Κβαντισμένοι Συντελεστές DCT} = \text{round}\left(\frac{\text{Συντελεστές DCT}}{qScale * qTable}\right)$$

όπου ο κάθε συντελεστής DCT διαιρείται με την αντίστοιχη τιμή του πίνακα κβαντισμού (*qTable*), το *qScale* ρυθμίζει την ποιότητα την κλίμακα κβαντισμού και έπειτα το αποτέλεσμα στρογγυλοποιείται στον κοντινότερο γείτονα.

Η σκέψη είναι πως οι μικροί συντελεστές όταν διαιρεθούν με μεγάλο διαιρέτη (λογικά) θα μηδενιστούν, ενώ οι σημαντικές τιμές θα μείνουν ανεπηρέαστες. Επιπλέον, είναι προφανές πως για την αποκβαντοποίηση των τιμών ακολουθείται η αντίστροφη διαδικασία. Τέλος, αξίζει να σημειωθεί πως για τους συντελεστές φωτεινότητας και χρωματικότητας χρησιμοποιούνται διαφορετικοί πίνακες κβαντισμού.

- **qblock = quantizeJPEG (dctBlock, qTable, qScale)**

Η συνάρτηση `quantizeJPEG` εφαρμόζει τον στους συντελεστές DCT ενός μπλοκ και έχει ως είσοδο το `dctBlock` που είναι η έξοδος της συνάρτησης μετασχηματισμού `dct` που περιεγράφηκε παραπάνω, τον πίνακα κβαντισμού `qTable` και την κλίμακα κβαντισμού `qScale`. Αρχικά, πραγματοποιείται έλεγχος για το αν οι παρεχόμενοι πίνακες `dctBlock` και `qTable` είναι ορθογώνιοι με διαστάσεις 8×8 . Εάν δεν ικανοποιούνται οι συγκεκριμένες διαστάσεις η συνάρτηση επιστρέφει `error`. Στη διαδικασία κβαντισμού των συντελεστών DCT, οι 64 παράγοντες που προκύπτουν υπόκεινται σε κβαντισμό από έναν ομοιόμορφο κβαντιστή. Το μέγεθος του βήματος του κβαντιστή για κάθε συντελεστή ορίζεται από την αντίστοιχη τιμή του πίνακα κβαντισμού. Τέλος, επιστρέφεται ο πίνακας `qBlock` που περιέχει τους κβαντισμένους συντελεστές DCT.

- **dctBlock = dequantizeJPEG (qBlock, qTable, qScale)**

Η συνάρτηση dequantizeJPEG αναλαμβάνει τον αποκβαντισμό των κβαντισμένων συντελεστών DCT, εφαρμόζοντας την αντίστροφη λειτουργία. Αρχικά, πραγματοποιείται έλεγχος για το αν οι διαστάσεις του qBlock και του qTable είναι 8x8. Έπειτα, αντί για διαίρεση εφαρμόζεται πολλαπλασιασμός με το γινόμενο του qScale και του qTable σε κάθε στοιχείο του πίνακα εισόδου qBlock. Με αυτόν τον τρόπο αποκαθίσταται η αρχική τους τιμή πριν τον κβαντισμό. Η συνάρτηση επιστρέφει τους αποκβαντισμένους συντελεστές DCT στον πίνακα dctBlock.

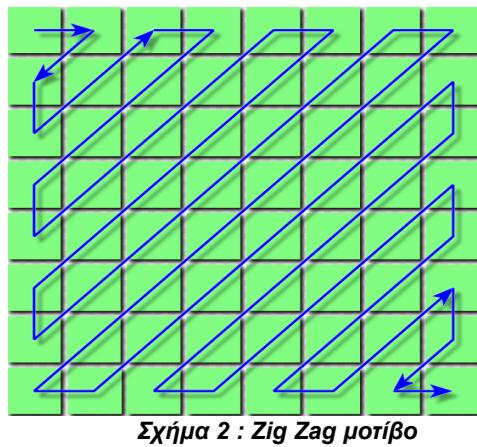
- **qTableC, qTableL = changedTables(num)**

Η βοηθητική συνάρτηση changedTables δημιουργεί δύο πίνακες κβαντισμού qTableL και qTableC, οι οποίοι αρχικά περιέχουν τους προκαθορισμένους πίνακες κβαντισμού για τη φωτεινότητα (Luminance) και την χρωματικότητα καναλιών (Chrominance) αντίστοιχα, σύμφωνα με τα πρότυπα JPEG. Αν η μεταβλητή εισόδου num είναι μηδέν, επιστρέφονται οι πίνακες κβαντισμού που ορίζει το πρότυπο. Σε διαφορετική περίπτωση, η συνάρτηση τροποποιεί τους πίνακες κβαντισμού, αντικαθιστώντας τα τελευταία 'num' στοιχεία με την τιμή 1000.

Zig-zag scanning και RLE

Μετά την κβάντιση, το μπλοκ εικόνας μετατρέπεται σε διάνυσμα ροής για τη διαδικασία κωδικοποίησης εντροπίας. Τα 64 στοιχεία δεδομένων είναι ευθυγραμμισμένα χρησιμοποιώντας μια σάρωση Zig-Zag του μπλοκ 8x8 για να συγκεντρωθούν τα στοιχεία χαμηλής συχνότητας στην αρχή της ροής. Αυτός είναι ένας βολικός τρόπος για να ληφθούν μεγάλες σειρές μηδενικών στοιχείων προς το τέλος της ροής αφού μετά την κβάντιση (ανάλογα με τον πίνακα κβάντισης που χρησιμοποιείται) οι συντελεστές που προκύπτουν από το μπλοκ είναι μηδενικοί στους υψηλότερους συντελεστές συχνότητας.

Το RLE των AC συντελεστών αποτελεί μια διαδικασία χωρίς απώλειες που έχει στόχο τη μείωση του αριθμού των συμβόλων, εκμεταλλεύμενη την ύπαρξη πολλών μηδενικών στις τιμές των συντελεστών, οι οποίοι προέρχονται κυρίως από υψηλές συχνότητες και μετά την Zig-Zag σάρωση βρίσκονται στο τέλος του διανύσματος. Αν αναδιατάξουμε τα στοιχεία, ξεκινώντας από την πάνω αριστερή γωνία, σε ένα διάνυσμα 64 στοιχείων σε Zig-zag μοτίβο (εικόνα) θα στοιχίσουμε τα στοιχεία από τους χαμηλόσυχνους συντελεστές στους πιο υψηλούς.



Στο διάνυσμα που παράχθηκε η πρώτη τιμή αποτελεί την DC συνιστώσα, ενώ οι υπόλοιπες συνθέτουν τις AC συνιστώσες. Για την κωδικοποίηση του DC όρου θα χρησιμοποιηθεί διαφορετική διαδικασία απ' ότι για τους AC. Στην περίπτωση του DC όρου, στην πραγματικότητα θα κωδικοποιηθεί η διαφορά (*DPCM- Differential Pulse Code Modulation*)

$$DIFF = DC_i - PRED$$

όπου DC_i ο (κβαντισμένος) DC όρος του τωρινού block και $PRED$, ο αντίστοιχος όρος του προηγούμενου προς κωδικοποίηση block (ίδιας συνιστώσας).

Για τις AC συνιστώσες μετράμε πόσα μηδενικά προηγήθηκαν των τιμών τους και δημιουργούνται τα ζεύγη

< precedingZeros, quantSymbol >

με σκοπό να παραχθεί ένας πίνακας μεγέθους $R \times 2$, οπου στην 1^η γραμμή του πίνακα θα τοποθετηθεί η κωδικοποίηση του DC ορού ως < 0 , DIFF >.

Σε αυτό το σημείο να σημειωθεί, πως στην συνέχεια βρίσκεται το στάδιο της Huffman κωδικοποίησης και μέσω αυτής ορίζονται δύο ειδικές περιπτώσεις. Πιο συγκεκριμένα

1. Δεν μπορούν να υπάρξουν περισσότερα από 16 συνεχόμενα μηδενικά (ZRL). Αν βρεθεί τέτοια περίπτωση πρέπει τα precedingZeros να διαχωριστούν. Για παράδειγμα αν βρέθηκαν 22 μηδενικά πριν την τιμή 1, τότε ο πίνακας των μηκών διαδρομής θα έχει την μορφή :

15	0	
[6	1]
...	...	

2. Αν τα συνεχόμενα μηδενικά βρίσκονται έως το τέλος του κβαντισμένου block, τότε τοποθετείται το ζεύγος του EOB (*EndOfBlock*) στον πίνακα των μηκών διαδρομής, δηλαδή $< 0, 0 >$.

Η παραπάνω διαδικασία έχει ως σκοπό να συγκεντρώσει τα μηδενικά σε ομάδες και να μειώσει περαιτέρω τα προς κωδικοποίηση απαιτούμενα κομμάτια πληροφορίας.

- **runSymbols = runLength(qBlock, DCpred)**

Η συνάρτηση `runLength` υπολογίζει τα σύμβολα μήκους διαδρομής σε έναν πίνακα κβαντισμένων συντελεστών DCT. Αρχικά, γίνεται έλεγχος για τη διάσταση του πίνακα `qBlock`. Αν αυτός δεν έχει τις αναμενόμενες διαστάσεις 8×8 ή εάν ο αριθμός `DCpred` δεν είναι `scalar`, τότε προκαλείται ένα σφάλμα. Στην συνέχεια, δημιουργείται ο πίνακας `idx` ο οποίος περιέχει τους δείκτες της ακολουθίας `zig zag` για τους συντελεστές DCT στις αντίστοιχες θέσεις. Ουσιαστικά είναι ο πίνακας `A.6` του προτύπου της εκφώνησης. Αυτός ο πίνακας χρησιμοποιείται για να διαταχθούν οι συντελεστές DCT. Χρησιμοποιώντας τον αλγόριθμο RLE, η συνάρτηση `sumppiezei` την ακολουθία των κβαντισμένων συντελεστών DCT σε μια λίστα `runSymbols`. Χρησιμοποιώντας τον αντίστροφο πίνακα `idx`, επιλέγονται τα μη μηδενικά στοιχεία του `qBlock`. Έπειτα, αρχικοποιείται η λίστα `runSymbols` με την αρχική DC συνιστώσα και ορίζεται η μεταβλητή `precedingZeros`. Για κάθε στοιχείο του `qBlock`, ελέγχεται αν είναι μηδέν ή όχι. Αν είναι μηδέν, αυξάνεται το `precedingZeros`. Στο τέλος της διαδικασίας, τα αποτελέσματα αυτά επιστρέφονται ως λίστα `runSymbols`.

- **qBlock = irunLength(runSymbols, DCpred)**

Η συνάρτηση `irunLength` επιτελεί την αντίστροφη λειτουργία από την προηγουμένη και δέχεται ένα σύνολο συμβόλων run-length και μια πρόβλεψη DC ως είσοδο και επιστρέφει τον αντίστροφο αποκωδικοποιημένο μπλοκ κβαντισμένων συντελεστών DCT. Αρχικά, η συνάρτηση ελέγχει εάν ο πίνακας `runSymbols` έχει διαστάσεις `Rx2` (R γραμμές και 2 στήλες) και εάν η `DCpred` είναι ένας αριθμός. Εάν όχι, παράγει ένα σφάλμα. Αρχικοποιεί έναν πίνακα `qBlock` με τον αρχικό κβαντισμένο συντελεστή DC και διατρέχει το σύνολο `runSymbols` ως εξής: Εάν η τρέχουσα γραμμή αντιστοιχεί σε μια σειρά 16 μηδενικών, προσθέτει 16 μηδενικά στο `qBlock`. Από την άλλη εάν η τρέχουσα γραμμή αντιστοιχεί στο τέλος του μπλοκ, προσθέτει μηδενικά για να συμπληρώσει το υπόλοιπο του μπλοκ. Διαφορετικά, προσθέτει τον αντίστοιχο αριθμό μηδενικών και τον κβαντισμένο συντελεστή στο `qBlock`. Έπειτα, ανακατασκευάζει το αρχικό 8×8 μπλοκ κβαντισμένων συντελεστών DCT από τα δεδομένα του `qBlock` χρησιμοποιώντας αντίστροφη αναδιάταξη ZigZag.

Κωδικοποίηση - Αποκωδικοποίηση Huffman

Σκοπός είναι τα σύμβολα που έχουν προκύψει από τα προηγούμενα βήματα να κωδικοποιηθούν βάσει της κατανομής πιθανότητας εμφάνισης τους. Σύμβολα με υψηλή πιθανότητα θα αντιστοιχούν σε μικρότερου μήκους κωδικούς, γεγονός που δίνει την ευκαιρία μείωσης χρήσης της μνήμης.

Το JPEG πρότυπο καθορίζει τυπικά Huffman Tables που χρησιμοποιούνται κατά την επεξεργασία εικόνας επιτρέποντας τη μείωση μεγέθους του JPEG αρχείου. Συνήθως, η διαδικασία της κωδικοποίησης των JPEG εικόνων χρησιμοποιεί συγκεκριμένα Huffman Tables που περιγράφονται στο header του jpeg αρχείου και ανακτώνται από τον αποκωδικοποιητή για την αντίστροφη διαδικασία της αποκωδικοποίησης.

Επομένως, χρησιμοποιήθηκαν οι πίνακες που ορίζει το πρότυπο, ως πίνακες αναφοράς. Αξίζει να σημειωθεί πως οι πίνακες αντιστοίχισης κωδικών λέξεων και συμβόλων είναι διαφορετικοί τόσο για τους DC και AC όρους όσο και για την χρωματικότητα και την φωτεινότητα. Για αυτό υλοποιείται η συναρτηση **ISOTables()** που αρχικοποιεί τους πίνακες αναφοράς για την κωδικοποίηση και την αποκωδικοποίηση Huffman.

Περιγραφή υλοποίησης

Η κωδικοποίηση Huffman δέχεται ως είσοδο τον πίνακα με τα μήκη διαδρομής, ωστόσο δεν κωδικοποιεί ακριβώς τα περιεχόμενα του. Πιο συγκεκριμένα η διαδικασία που ακολουθείται είναι η εξής:

- Για τον DC όρο πρέπει να βρεθεί η Κατηγορία στην οποία ανήκει η DIFF.
Για την Κατηγορία ισχύει :

$$-2^{Category} < Symbol < 2^{Category}$$

δηλαδή για τον υπολογισμό του *Category* μπορεί να γίνει χρήση της εξής κλαδικής σχέσης:

$$Category = \begin{cases} 0, & Symbol = 0 \\ \lfloor \log_2(|Symbol|) \rfloor + 1, & \text{else} \end{cases}$$

- Έπειτα, πρέπει να υπάρχει δυνατότητα να διαχωρίζονται διαφορετικά *Symbols* τα οποία κατατάσσονται στην ίδια Κατηγορία. Αυτό συμβαίνει με τα επιπλέον bits που τοποθετούνται μετά την κωδική λέξης της Κατηγορίας. Για να βρεθούν αυτά τα επιπλέον bits, υιοθετείται η εξής λογική:
 - *Category* = 0, δεν υπάρχουν επιπλέον bits.
 - *Category* ≠ 0, *Symbol* > 0, τα επιπλέον bits είναι η δυαδική αναπαράσταση του *Symbol*.
 - *Category* ≠ 0, *Symbol* < 0, τα επιπλέον bits είναι το συμπλήρωμα ως προς 1 της δυαδικής αναπαράστασης του *Symbol*.

- Για τους AC όρους ακολουθείται παρόμοια διαδικασία, με τις εξής διαφοροποιήσεις:
 - Δεν υπάρχει *Category* = 0.
 - Για τα μήκη διαδρομής [0 , 0] (EOB) και [15, 0] (ZRL) δεν υπάρχουν επιπλέον bits.
- Τέλος, για την εύρεση του huffStream κάθε block πρέπει να γίνει αντιστοίχιση των ζευγών < precedingZeros, *Category* > που βρέθηκαν στους κατάλληλους πίνακες αναφοράς, για την εύρεση των κωδικών λέξεων που τους αντιστοιχούν. Πιο συγκεκριμένα :
 - Για τους DC όρους η θέση, *index*, της κωδικής λέξης είναι *index* = *Category* + 1.
 - Για τους AC όρους ισχύει *index* = precedingZeros * 10 + *Category* + offset, με

$$offset = \begin{cases} 1, & \text{precedingZeros} \neq 15 \\ 2, & \text{precedingZeros} = 15 \end{cases}$$

Για την αποκωδικοποίηση του huffStream, χρησιμοποιείται η ιδιότητα του κώδικα Huffman ότι καμία κωδική λέξη δεν αποτελεί πρόθημα άλλης κωδικής λέξης. Πιο συγκεκριμένα:

- Εκτελείται μια αναζήτηση πάνω στο huffStream για την εύρεση της κωδικής λέξης του DC όρου. Όταν βρεθεί ακολουθείται η αντίστροφη διαδικασία που περιεγράφηκε παραπάνω για την εύρεση της Κατηγορίας και των επιπλέον bits.
- Από το σημείο που τελείωσαν τα ψηφία του κώδικα που αφορούν τον DC όρο, αρχίζει η αναζήτηση για όλους τους AC όρους και των επιπλέον bits τους. Για την επιστροφή από την θέση του πίνακα που βρέθηκε η κωδική λέξη, έστω *index*, ακολουθείται η εξής διαδικασία:

- $index=index-offset$, με $offset=\begin{cases} 1, & index < 151 \\ 2, & index > 151 \end{cases}$
- Υπολογίζεται το *Remainings* = *index mod 10*, όπου

$$Remainings \neq 0, Category = Remainings, precedingZeros = floor(index/10)$$

$$Remainings = 0, Category = 10, precedingZeros = floor(index/10) - 1$$

- Η διαδικασία ολοκληρώνεται όταν βρεθεί το EOB ή διαβαστεί ολόκληρο το huffStream.

- **huffStream = huffEnc(runSymbols)**

Η συνάρτηση huffEnc πραγματοποιεί την κωδικοποίηση Huffman ενός συνόλου συμβόλων run-length, χρησιμοποιώντας τους πίνακες Huffman που περνούνται ως είσοδοι. Αρχικά, ελέγχει εάν το runSymbols είναι Rx2 και επιλέγει τους πίνακες Huffman ανάλογα με το είδος των δεδομένων που επεξεργάζεται, είτε είναι φωτεινότητας (Luminance) είτε χρωματικότητας (Chrominance). Η μεταβλητή isLuminance είναι δυαδική και χρησιμοποιείται για να αποφασίσει ποιος πίνακας Huffman θα χρησιμοποιηθεί για την κωδικοποίηση των συμβόλων. Στην συνέχεια, για τον υπολογισμό του DC coefficient η συναρτηση υπολογίζει την κατηγορία του DC συντελεστή και τα επιπλέον bits που απαιτούνται για τον υπολογισμό του μέσω της βοηθητικής συνάρτησης getBinaryValue() και ακολουθείται η διαδικασία όπως περιεγράφηκε παραπάνω. Στο τέλος, το σύνολο των bits κωδικοποίησης εντοπίζεται στο strHuff, το οποίο μετατρέπεται σε bytes και επιστρέφεται ως λίστα huffStream.

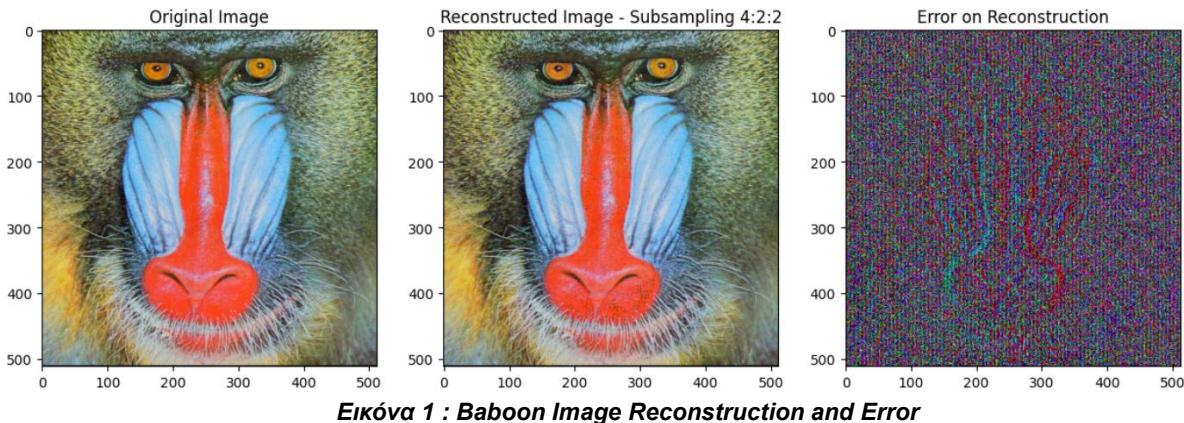
- **runSymbols = huffDec(huffStream)**

Η συναρτηση huffDec υλοποιεί την αποκωδικοποίηση Huffman και δέχεται ως είσοδο ένα stream από bits, που περιέχει την κωδικοποιημένη πληροφορία για ένα block. Στην έξοδο επιστρέφει τα αποκωδικοποιημένα σύμβολα μήκους διαδρομής. Αρχικά, η συναρτηση επιλέγει τους πίνακες Huffman ανάλογα με το είδος των δεδομένων που επεξεργάζεται, είτε είναι φωτεινότητας (Luminance) είτε χρωματικότητας (Chrominance). Η μεταβλητή isLuminance είναι δυαδική και χρησιμοποιείται για να αποφασίσει ποιος πίνακας Huffman θα χρησιμοποιηθεί για την αποκωδικοποίηση των συμβόλων. Στη συνέχεια, η συνάρτηση αναζητά την Huffman κωδικοποίηση για τον συντελεστή DC και ακολουθεί η διαδικασία αποκωδικοποίησης όπως περιεγράφηκε παραπάνω.

Demo 1

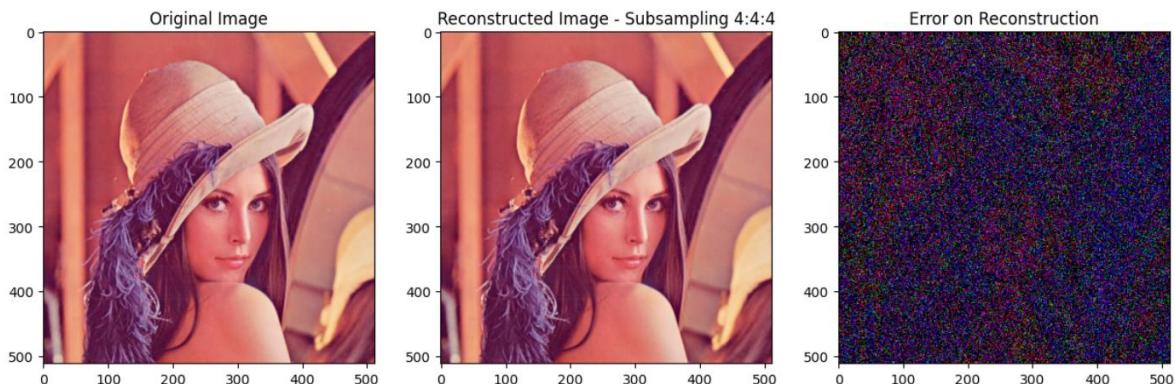
- Παρακατω παρουσιάζεται διαδικασία μετατροπής μίας εικόνας από RGB σε YCrCb και στη συνέχεια πίσω σε RGB χρησιμοποιώντας τις συναρτήσεις που παρουσιάστηκαν παραπάνω. Συγκεκριμένα, έγινε χρήση των συναρτήσεων **convert2ycbcr** και **convert2rgb**. Παρακατω, παρατίθενται τα αποτελέσματα για τις 2 εικόνες της εκφώνησης :

Για την πρώτη εικόνα, χρησιμοποιήθηκε υποδειγματοληψία 4: 2: 2



Εικόνα 1 : Baboon Image Reconstruction and Error

Για την 2^η εικόνα , χρησιμοποιήθηκε υποδειγματοληψία 4: 4: 4



Εικόνα 2 : Lena Image Reconstruction and Error

Παρατηρείται πως η καθε εικόνα ανακατασκευάστηκε σωστά, με το σφάλμα, που παρουσιάζεται στην δεξιά εικόνα, να οφείλεται στη παρεμβολή κατά την επιστροφή στον RGB χώρο για τον υπολογισμό των τιμών που έλειπαν. Χρησιμοποιήθηκε η παρεμβολή του κοντινότερου γείτονα, με τις τιμές που έλειπαν –λόγω της υποδειγματοληψίας- να υπολογίζονται ως η τιμή που είχε το pixel στα αριστερά τους. Στην περίπτωση της υποδειγματοληψίας 4: 2: 0, που δεν παρουσιάζεται εδώ, η παρεμβολή γίνεται σε παράθυρο μεγέθους 2×2 , με τις τιμές να ορίζονται ως η τιμή του πάνω αριστερά pixel.

2. Στην συνέχεια, παρουσιάζεται η διαδικασία μετατροπής μίας εικόνας από RGB σε YCrCb, ο υπολογισμός των κβαντισμένων DCT συντελεστών, και στη συνέχεια πίσω σε RGB χρησιμοποιώντας τις συναρτήσεις που υλοποιήθηκαν. Συγκεκριμένα, έγινε χρήση των συναρτήσεων **convert2ycbcr**, **convert2rgb**, **blockDCT**, **iBlockDCT**, **quantizeJPEG**, **dequantizeJPEG**.

Υλοποίηση

Με 2 βρόχους for διατρέχεται η εικόνα σε 8x8 blocks.

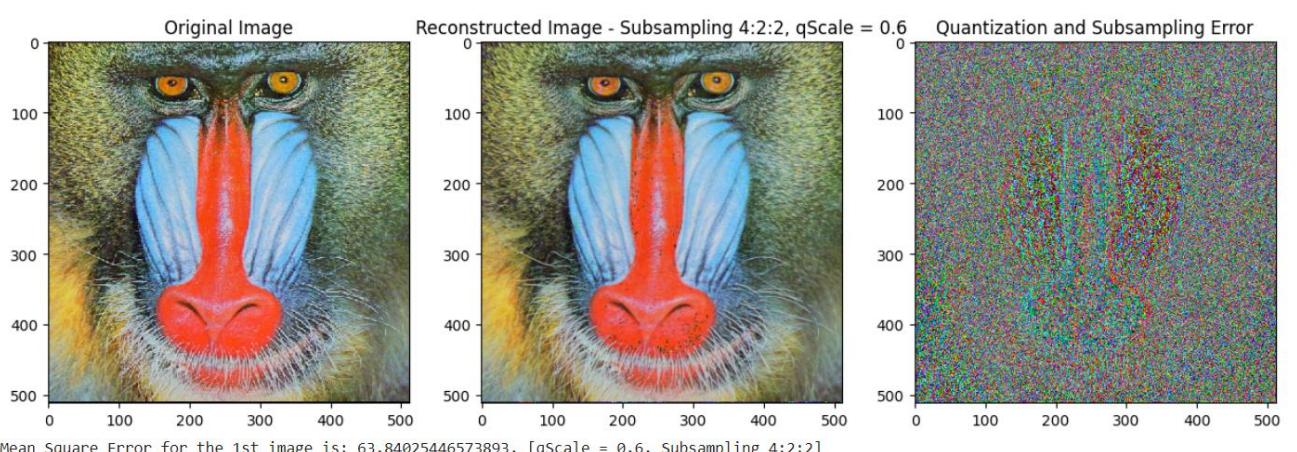
Επεξεργασία Y καναλιού:

Κάθε 8x8 block της συνιστώσας Y αποθηκεύεται στη μεταβλητή `blockY`. Εφαρμόζεται η μετασχηματισμός DCT στο `blockY` με τη συνάρτηση `blockDCT`, παράγοντας το `dctblockY`. Το μετασχηματισμένο block `dctblockY` γίνεται είσοδος στην `quantizeJPEG` παράγοντας το `quantblockY`. Στην συνέχεια, το `quantblockY` γίνεται είσοδος στην `dequantizeJPEG`, δημιουργώντας το `de_quantblockY`. Το `de_quantblockY` υποβάλλεται στον αντίστροφο μετασχηματισμό DCT με τη συνάρτηση `iBlockDCT`, παράγοντας το `idctblockY`. Το `idctblockY` ενσωματώνεται πίσω στο ανακατασκευασμένο κανάλι Y.

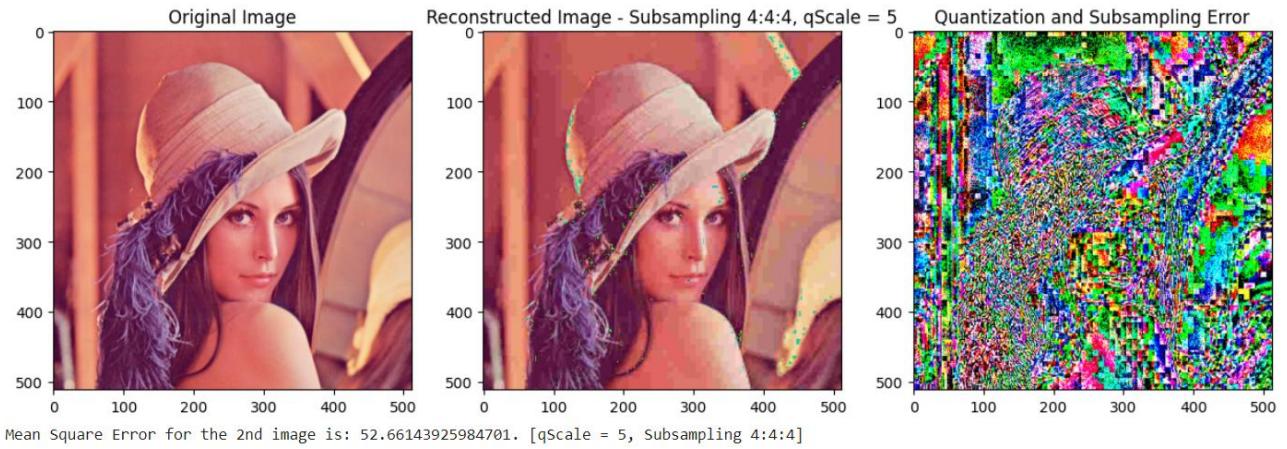
Επεξεργασία Cb και Cr καναλιών:

Τα βήματα επεξεργασίας για τις συνιστώσες Cb και Cr είναι παρόμοια με αυτά για τη συνιστώσα Y. Οι μετασχηματισμοί DCT, κβαντισμός, αποκβαντισμός και αντίστροφος μετασχηματισμός DCT εφαρμόζονται στα αντίστοιχα 8x8 blocks των συνιστωσών Cb και Cr. Τα αποτελέσματα των μετασχηματισμών αποθηκεύονται στις μεταβλητές `idctblockCr` και `idctblockCb` και ενσωματώνονται πίσω στις ανακατασκευασμένες συνιστώσες Cr και Cb αντίστοιχα.

Παρακατω, παρατίθενται τα αποτελέσματα για τις 2 εικόνες της εκφώνησης :



Εικόνα 3: Baboon Image and Reconstruction and Error



Εικόνα 4 : Lena Image Reconstruction and Error

JPEG Encoder/Decoder

- **JPEGenc = JPEGencode (img, subimg, qScale)**

Η συνάρτηση JPEGencode υλοποιεί τη διαδικασία κωδικοποίησης JPEG για μια εικόνα. Η JPEGencode έχει ως εισόδους την εικόνα img, qScale: την κλίμακα κβαντισμού, και τον 1x3 subimg που καθορίζει την υποδειγματοληψία. Παρακάτω παρουσιάζεται η ανάλυση της λειτουργίας της: Αρχικά, η συνάρτηση ελέγχει αν η εισαγόμενη εικόνα img είναι τρισδιάστατος πίνακας (3D matrix), ενώ ελέγχει επίσης αν το subimg είναι ένας πίνακας μεγέθους 1x3 και αν το qScale είναι μια σταθερά. Δημιουργούνται οι πίνακες Huffman για την κωδικοποίηση των DC και AC συνιστωσών για τις συνιστώσες για την φωτεινότητα (Luminance) και της χρωματικότητας (Chrominance) και αποθηκεύονται μαζί με τους πίνακες κβαντισμού qScaleL και qScaleC στην λίστα JPEGenc.

Ακολουθεί η μετατροπή σε YCbCr: Η εικόνα εισόδου μετατρέπεται από τον χώρο RGB στον χώρο YCbCr. Στην συνέχεια, τα στοιχεία του καναλιού Y (Luminance) χωρίζονται σε blocks μεγέθους 8x8 πριν από την επεξεργασία τους. Οι δείκτες idxHor και idxVer χρησιμοποιούνται για να καταγράψουν την τρέχουσα οριζόντια και κάθετη θέση του μπλοκ στην εικόνα. Για καθε μπλοκ, δημιουργείται ένα struct με το όνομα tmpBlockY το οποίο περιέχει το μπλοκ της συνιστώσας Y των pixels ('block') καθώς και τους οριζόντιους και κατακόρυφους δείκτες ('indHor' και 'indVer') που υποδεικνύουν τη θέση του μπλοκ στην εικόνα. Τα μπλοκ αποθηκεύονται σε μια λίστα που ονομάζεται Yblocks.

Για κάθε block Y κάθε συνιστώσας, εκτελείται ο μετασχηματισμός DCT, ο κβαντισμός συντελεστών DCT, υπολογισμός μηκών διαδρομής και η κωδικοποίηση Huffman. Το αποτέλεσμα που προκύπτει για το κάθε block αποθηκεύονται στον struct πίνακα JPEGencode. Στην πρώτη θέση του JPEGencode (μεγέθους $N \times 1$), αποθηκεύονται τα τεχνικά χαρακτηριστικά της κωδικοποίησης (πίνακες κβαντισμού και πίνακες αναφοράς για τον κώδικα Huffman). Παρόμοια διαδικασία ακολουθείται και για τις συνιστώσες Cb και Cr. Επιστρέφεται ο πίνακας JPEGencode που περιέχει τις πληροφορίες κωδικοποίησης για όλα τα blocks της εικόνας.

- **imgRec = JPEGdecode(JPEGenc)**

Προκειμένου να αποκωδικοποιηθεί μια JPEG εικόνα, χρειάζεται όλα να παραπάνω βήματα που περιγράφονται στον κωδικοποιητή JPEG να αντιστραφούν και να εκτελεστούν με την αντίθετη ροή. Η συνάρτηση αναλαμβάνει να ανακτήσει μια εικόνα από μια λίστα που περιέχει τα δεδομένα JPEG που έχουν κωδικοποιηθεί. Δημιουργούνται οι πίνακες Huffman για την κωδικοποίηση των DC και AC συνιστωσών για τις συνιστώσες για την φωτεινότητα (Luminance) και της χρωματικότητας (Chrominance) και αποθηκεύονται μαζί με τους πίνακες κβαντισμού qScaleL και qScaleC στο tableStruct.

Η συνάρτηση υπολογίζει τις διαστάσεις των εικόνων που αποκωδικοποιούνται από την λίστα JPEGenc. Αρχικά, εξάγονται όλες οι τιμές 'indHor' και 'indVer' από κάθε στοιχείο της λίστας JPEGenc και αποθηκεύονται στις λίστες indHorValues και indVerValues αντίστοιχα. Στη συνέχεια, βρίσκονται οι μέγιστες τιμές των indHor και indVer, και υπολογίζονται οι συνολικές γραμμές και στήλες της εικόνας Y πολλαπλασιάζοντας τις μέγιστες τιμές με το μήκος των blocks (8). Έπειτα, υπολογίζονται οι αριθμοί των blocks για κάθε κανάλι (Y, Cb, Cr). Βασιζόμενοι στη σχέση μεταξύ αυτών των αριθμών δημιουργούνται οι κατάλληλες πίνακες για την αποκωδικοποίηση των καναλιών Y, Cb και Cr.

Για κάθε στοιχείο της λίστας JPEGenc (εκτός από το πρώτο, το οποίο περιέχει τους πίνακες κβαντισμού και τους πίνακες Huffman) εκτελείται αποκωδικοποίηση Huffman, υπολογισμός των κβαντισμένων συντελεστών DCT από τα μήκη διαδρομής, αποκβαντισμός των συντελεστών DCT και εφαρμογή αντίστροφου μετασχηματισμού DCT.

Η μεταβλητή numberOfY αυξάνεται κατά ένα, υποδεικνύοντας ότι ένα ακόμα block Y έχει αποκωδικοποιηθεί επιτυχώς. Για τα κανάλια "Cb" και "Cr", η διαδικασία είναι παρόμοια. Τέλος, το σύνολο της εικόνας μετατρέπεται από YCbCr σε RGB.

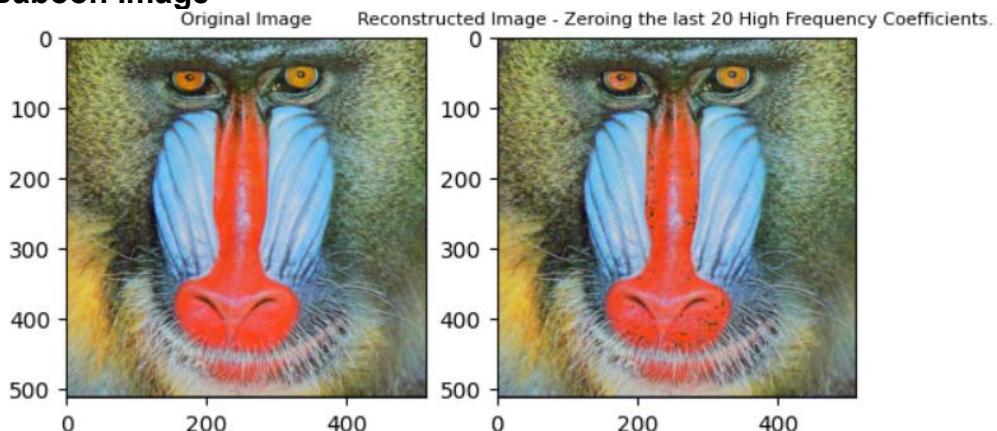
Καταγραφή Αποτελεσμάτων

Για τις 2 εικόνες που δίνονται από την εκφώνηση παρουσιάζονται αρχικά τα αποτελέσματα για τροποποίηση των πινάκων κβαντισμού και συγκεκριμένα μηδενισμό των 20, 40, 50, 60 και 63 πλέον υψίσυχων όρων των dctBlock για $qScale = 1$. Σημειώνεται ότι οι εικόνες πριν χρησιμοποιηθούν στον κωδικοποιητή και αποκωδικοποιητή JPEG έχουν ένα στάδιο προεπεξεργασίας : μετατροπή των τιμών των pixels στο εύρος [0, 255] και σε διαστάσεις που είναι πολλαπλάσιο του 8.

Μηδενισμός υψίσυχων όρων DCT – Πρώτη εικόνα

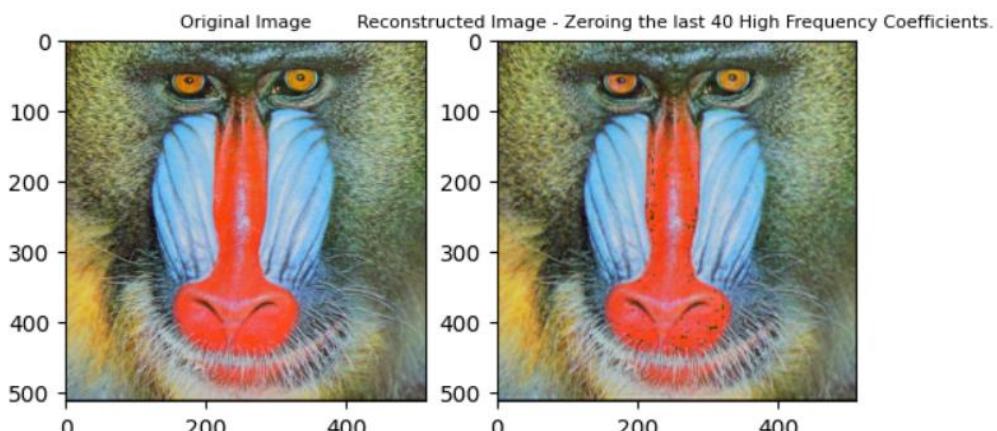
Για τον μηδενισμό των k υψίσυχων όρων, πρέπει να τροποποιηθεί το περιεχόμενο των k τελευταίων στοιχείων (σε Zig-Zag μοτίβο) των πινάκων κβαντισμού. Έτσι, αντικαθίστανται οι τιμές τους με τη τιμή 1000 (πρέπει ο διαιρέτης να είναι μεγάλος).

Baboon Image



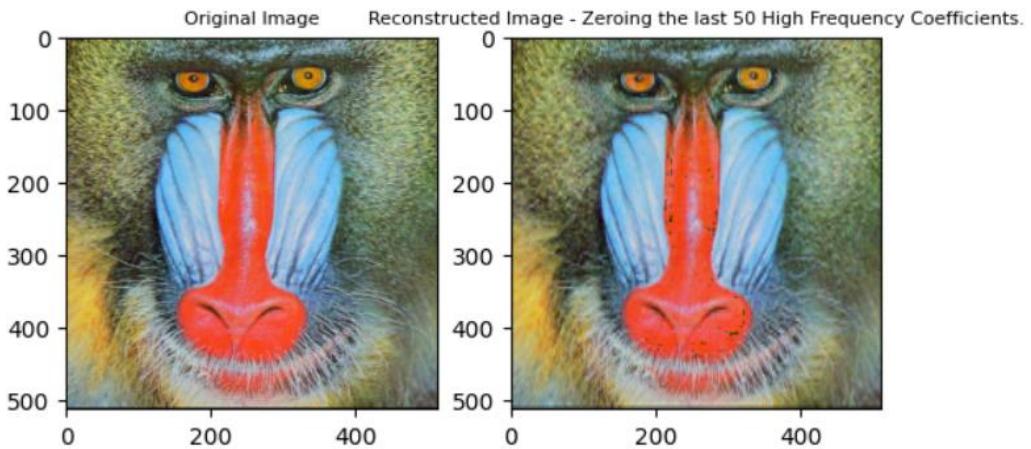
[Baboon Image] - Done zeroing the last 20 High Frequency Coefficients.

Eικόνα 6 : Μηδενισμός 20 υψίσυχων όρων



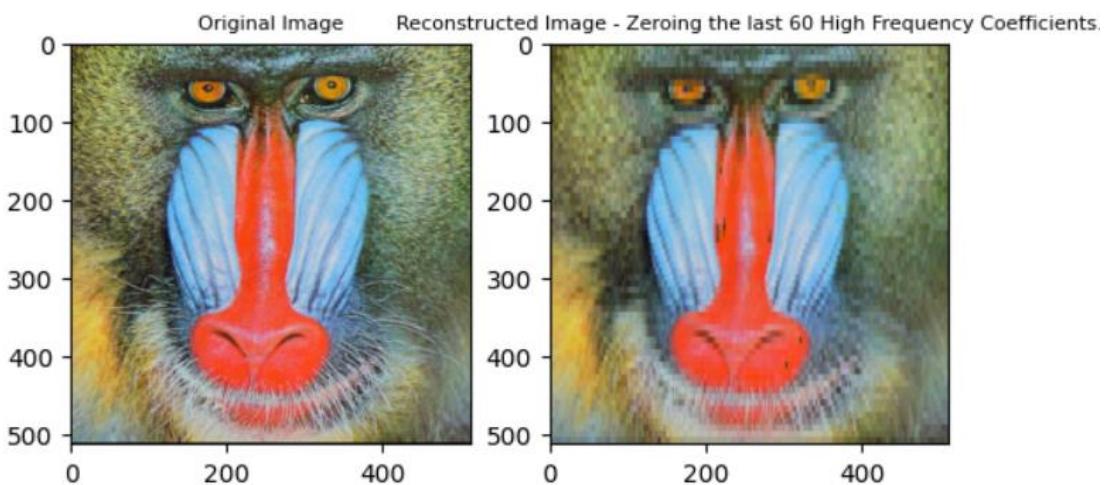
[Baboon Image] - Done zeroing the last 40 High Frequency Coefficients.

Eικόνα 7 : Μηδενισμός 40 υψίσυχων όρων



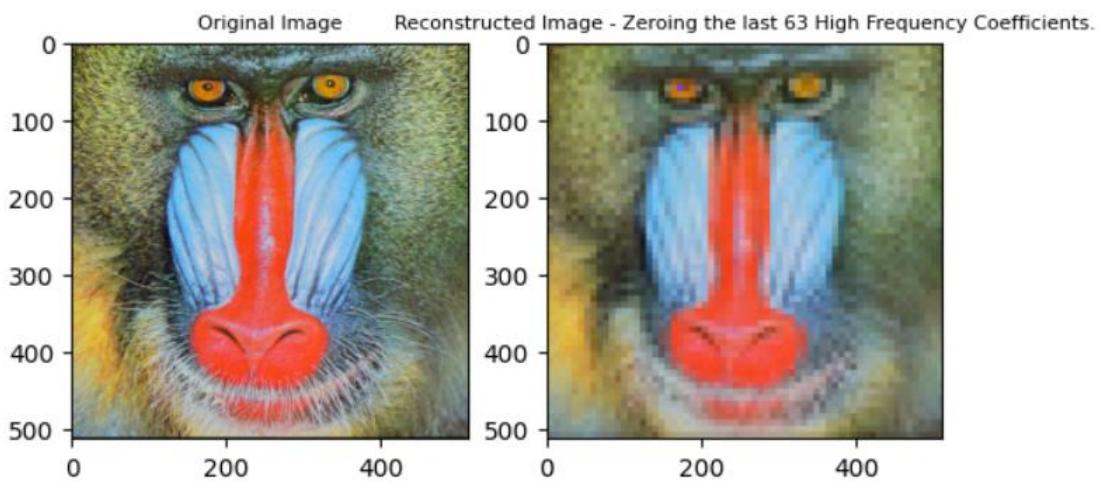
[Baboon Image] - Done zeroing the last 50 High Frequency Coefficients.

Εικόνα 8 : Μηδενισμός 50 υψίσυχνων όρων



[Baboon Image] - Done zeroing the last 60 High Frequency Coefficients.

Εικόνα 9 : Μηδενισμός 60 υψίσυχνων όρων



[Baboon Image] - Done zeroing the last 63 High Frequency Coefficients.

Εικόνα 10 : Μηδενισμός 63 υψίσυχνων όρων

Lena Image



[Lena Image] - Done zeroing the last 20 High Frequency Coefficients.

Εικόνα 11 : Μηδενισμός 20 υψίσυχνων όρων



[Lena Image] - Done zeroing the last 40 High Frequency Coefficients.

Εικόνα 12 : Μηδενισμός 40 υψίσυχνων όρων



[Lena Image] - Done zeroing the last 50 High Frequency Coefficients.

Εικόνα 13 : Μηδενισμός 50 υψίσυχνων όρων



[Lena Image] - Done zeroing the last 60 High Frequency Coefficients.

Εικόνα 14 : Μηδενισμός 60 υψίσυχνων όρων



[Lena Image] - Done zeroing the last 63 High Frequency Coefficients.

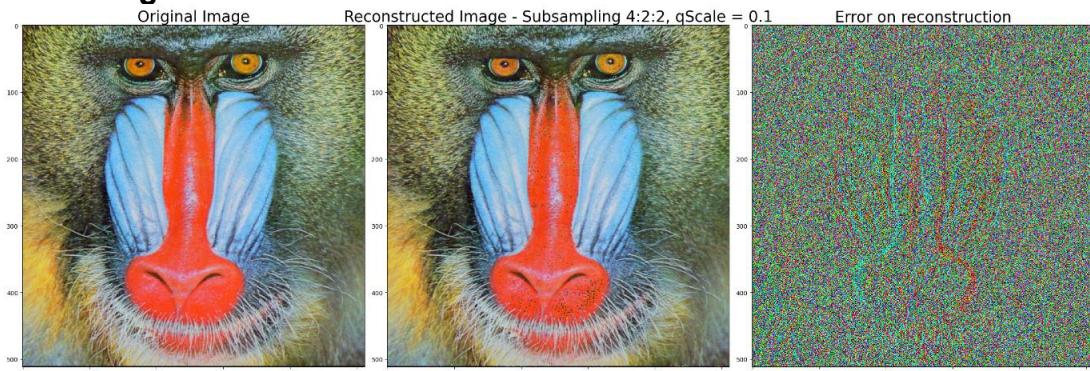
Εικόνα 15 : Μηδενισμός 63 υψίσυχνων όρων

Η αφαίρεση περισσότερων υψηλής συχνότητας όρων κατά τον κβαντισμό οδηγεί σε μείωση της λεπτομέρειας και αύξηση του θολώματος στην ανακατασκευασμένη εικόνα. Παρατηρείται ότι όσο περισσότεροι οροί αφαιρούνται τόσο μεγαλύτερο είναι το θόλωμα στην εικόνα. Συγκεκριμένα στην τελευταία το αποτέλεσμα είναι ιδιαίτερα θολό. Ωστόσο, παρατηρείται πως ακόμα και αν μηδενιστούν στην ακραία περιπτωση 63 όροι (όλους τους AC όρους!), μόνο η πληροφορία του DC όρου, αρκεί για να δοθεί η γενικότερη μορφή που περιέχεται στην εικόνα. Επιβεβαιώνεται έτσι για τον μετασχηματισμό DCT η εξαιρετική ικανότητα συγκέντρωσης της πληροφορίας σε ένα μικρό πλήθος δειγμάτων.

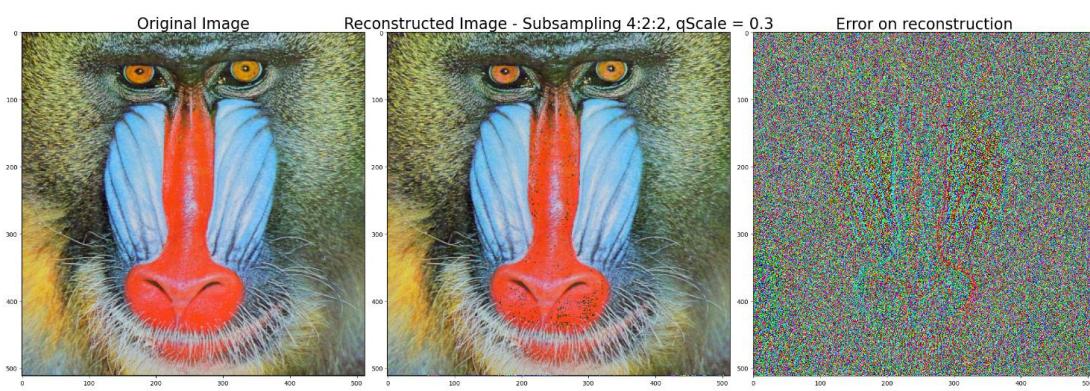
Σημείωση: Δεν δόθηκε διευκρίνηση για τον τύπο της υποδειγματοληψίας, έτσι χρησιμοποιήθηκε η 4:4:4 (με $qScale = 1$)

Παρακατω , παρουσιάζονται τα αποτελέσματα για τις δυο εικόνες για εφαρμογή των συναρτήσεων για διάφορες τιμές του qScale.

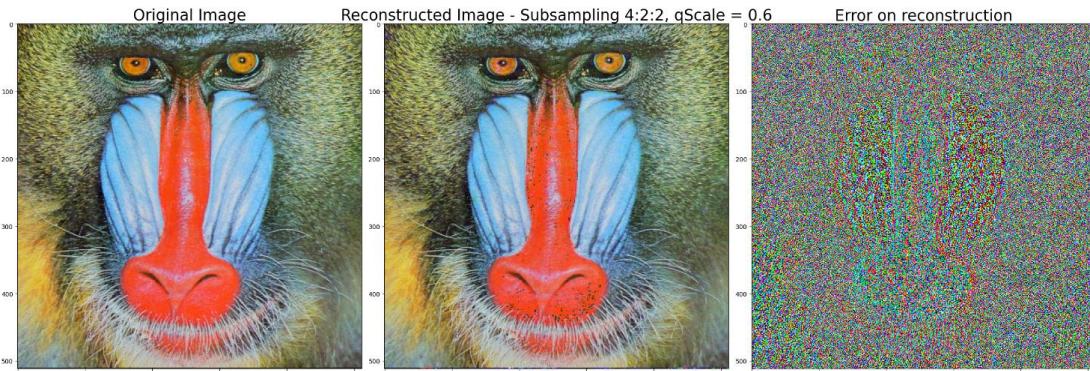
Baboon Image



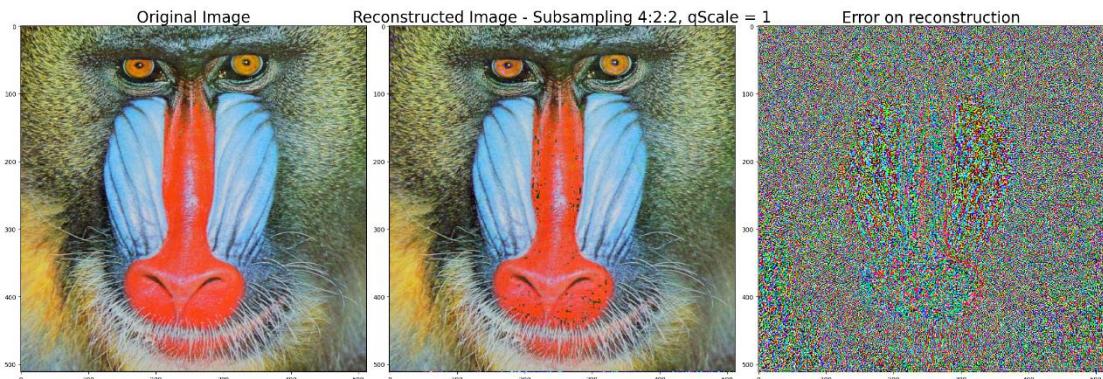
Eikόνα 16: Baboon Image and Reconstruction and Error for qScale = 0.1



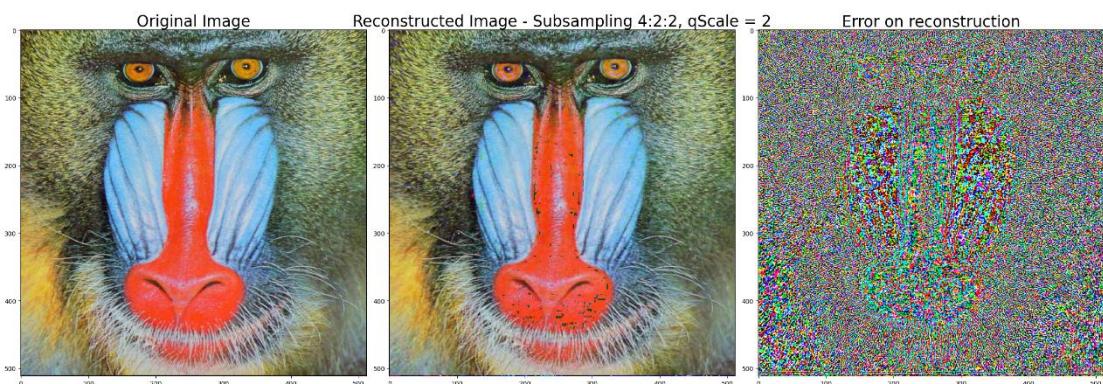
Eikόνα 17: Baboon Image and Reconstruction and Error for qScale = 0.3



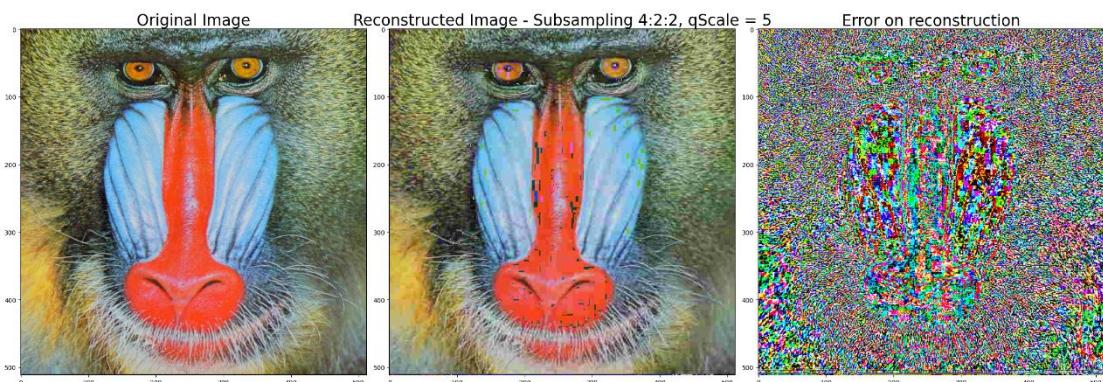
Eikόνα 18: Baboon Image and Reconstruction and Error for qScale = 0.6



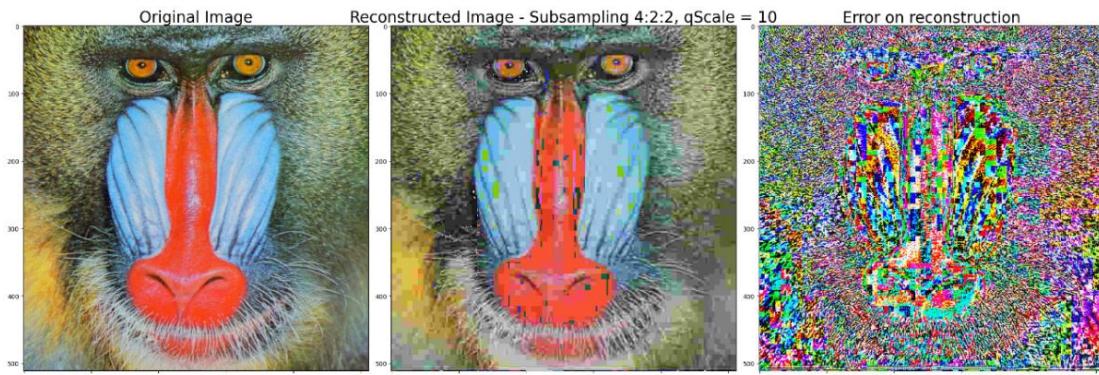
Eikóva 19: Baboon Image and Reconstruction and Error for qScale = 1



Eikóva 20: Baboon Image and Reconstruction and Error for qScale = 2



Eikóva 21: Baboon Image and Reconstruction and Error for qScale = 5



Εικόνα 22: Baboon Image and Reconstruction and Error for qScale = 10

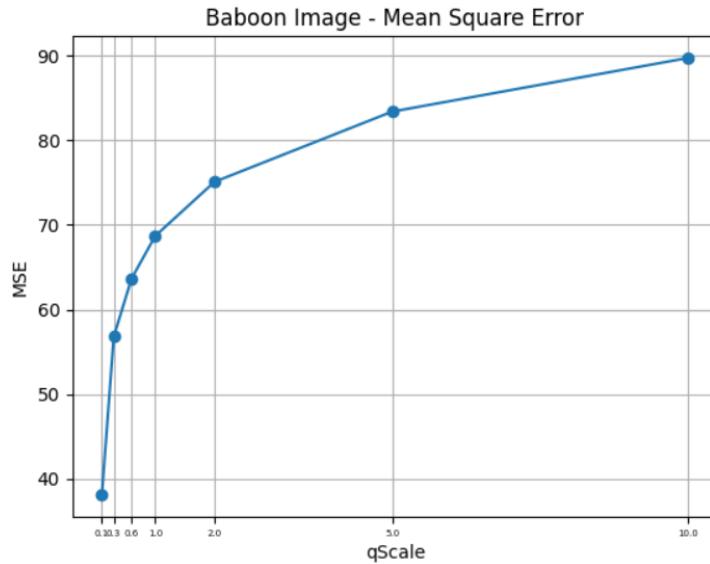
Παρατηρείται πως όσο αυξάνει το *qScale*, τόσο χαλάει το οπτικό αποτέλεσμα, ειδικά στις λεπτομέρειες και στις γρήγορες μεταβολές της εικόνας (πχ. στο κίτρινο του τριχώματος). Η παρατήρηση επιβεβαιώνεται και από την αύξηση του MSE όσο αυξάνεται το *qScale*.

Στην συνέχεια, παρουσιάζονται τα διαγράμματα για την αντίστοιχη τιμή του MSE (Mean Square Error).

Για κάθε τιμή του *qScale*, πραγματοποιείται η εξής διαδικασία :

Χρησιμοποιείται η συνάρτηση JPEGencode για να κωδικοποιήσει την εικόνα με την τρέχουσα τιμή του *qScale*. Χρησιμοποιείται η συνάρτηση JPEGdecode για να αποκωδικοποιήσει τα αποτελέσματα της κωδικοποίησης. Υπολογίζεται το μέσο τετραγωνικό σφάλμα (MSE) μεταξύ της αρχικής εικόνας και της εικόνας που αποκωδικοποιήθηκε. Υπολογίζεται το πλήθος των bit που χρησιμοποιήθηκαν για την κωδικοποίηση, καταμετρώντας το μήκος της ακολουθίας bit που παράγεται από τη συνάρτηση JPEGencode. Υπολογίζεται ο ρυθμός συμπίεσης (compression ratio), που ορίζεται ως ο λόγος του πλήθους των bit της αρχικής εικόνας προς το πλήθος των bit της κωδικοποιημένης εικόνας.

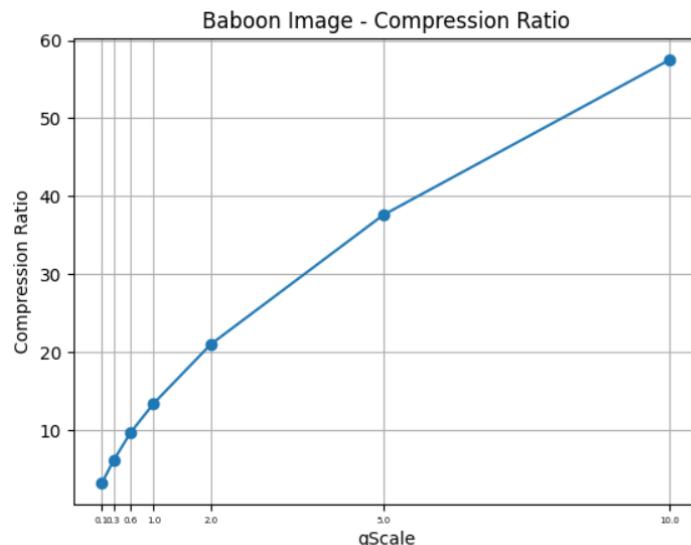
Ακολουθουν τα διαγράμματα :



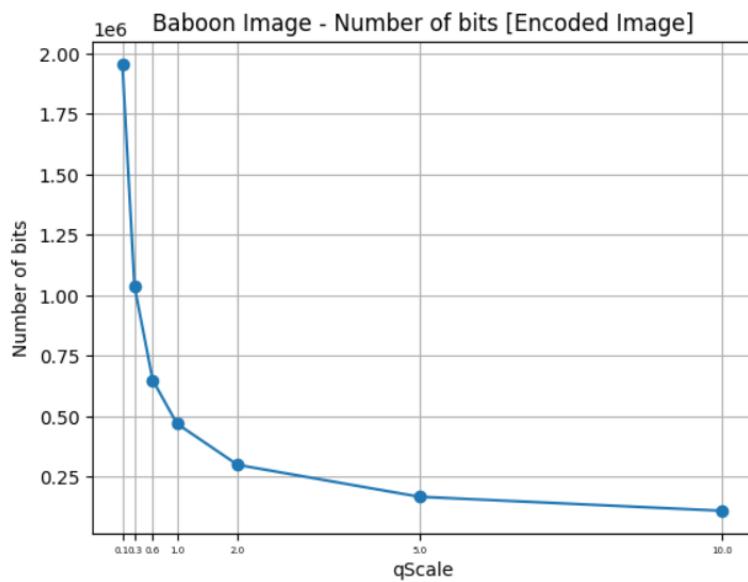
Διάγραμμα 1: MSE για κάθε τιμή qScale

Για να υπολογιστεί ο λόγος συμπίεσης, αρκεί να μετρηθεί το πλήθος των στοιχείων της αρχικής εικόνας και να γίνει σύγκριση με το συνολικό μέγεθος των huffStreams. Για την πρώτη εικόνα με υποδειγματοληψία 4: 2: 2 και διάφορες τιμές του qScale, τα αποτελέσματα δίνονται παρακάτω :

	qScale						
	0.1	0.3	0.6	1	2	5	10
Compression Ratio	3.2157	6.0616	9.6804	13.3920	21.0101	37.5672	57.5171



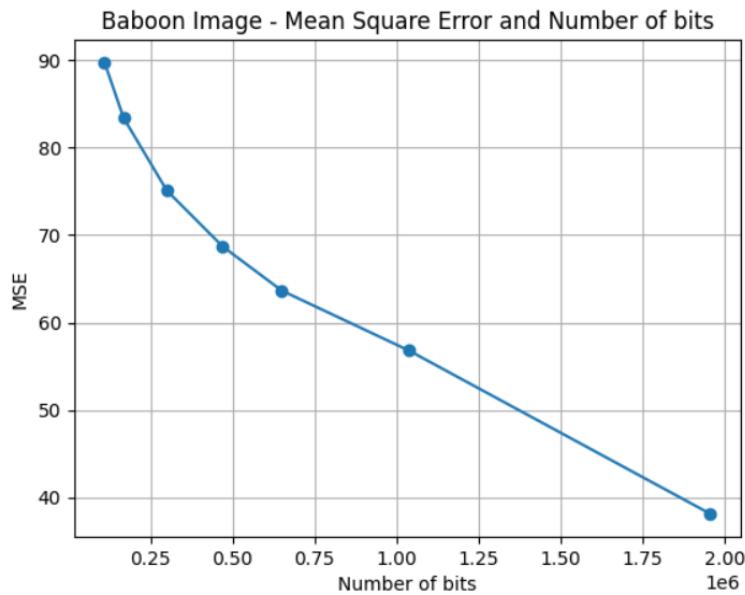
Διάγραμμα 2: Compression Ratio για κάθε τιμή qScale



Διάγραμμα 3: Number of bits για κάθε τιμή qScale

Ωστόσο, πρέπει να σημειωθεί πως με την αύξηση του *qScale*, παρατηρείται μείωση του αριθμού των bits της κωδικοποιημένης εικόνας.

Τέλος, δίνεται το γράφημα που δείχνει την σχέση μεταξύ του MSE και του αριθμού των bits της κωδικοποιημένης εικόνας και φαίνεται ότι η αύξηση του αριθμού των bits θα οδηγήσει σε μείωση του MSE.



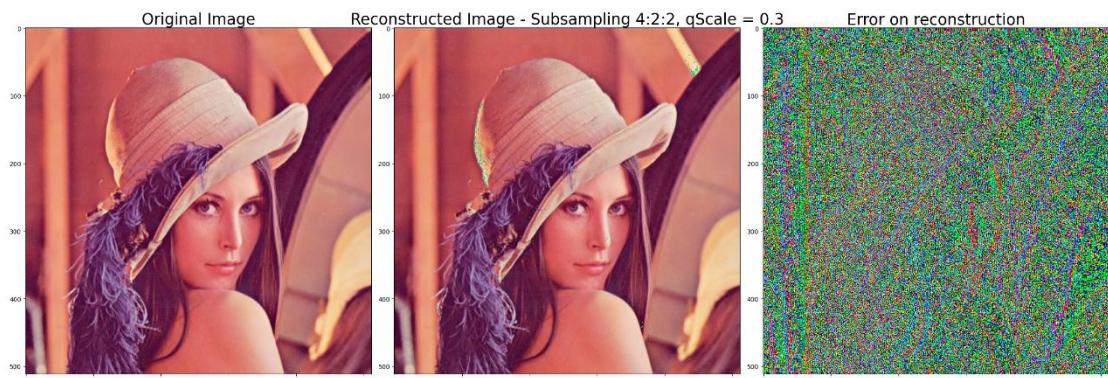
Διάγραμμα 4: MSE ως προς τον αριθμό bits

Lena Image

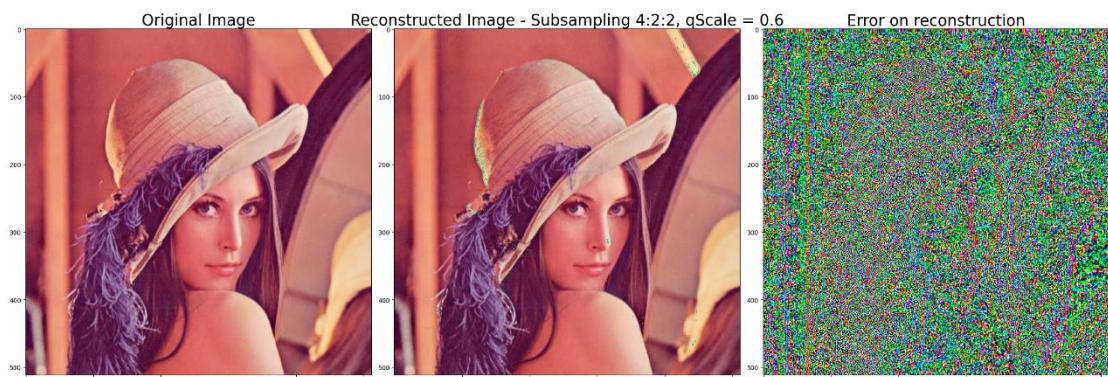
Στην συνέχεια, η ίδια διαδικασία επαναλαμβάνεται για την δεύτερη εικόνα. Δεν διευκρινίστηκε η υποδειγματοληψία, έτσι χρησιμοποιήθηκε 4: 4: 4, από το demo1 και παρουσιάζονται κατευθείαν τα αποτελέσματα :



Εικόνα 23: Lena Image and Reconstruction and Error for qScale = 0.1



Εικόνα 24: Lena Image and Reconstruction and Error for qScale = 0.3



Εικόνα 25: Lena Image and Reconstruction and Error for qScale = 0.6



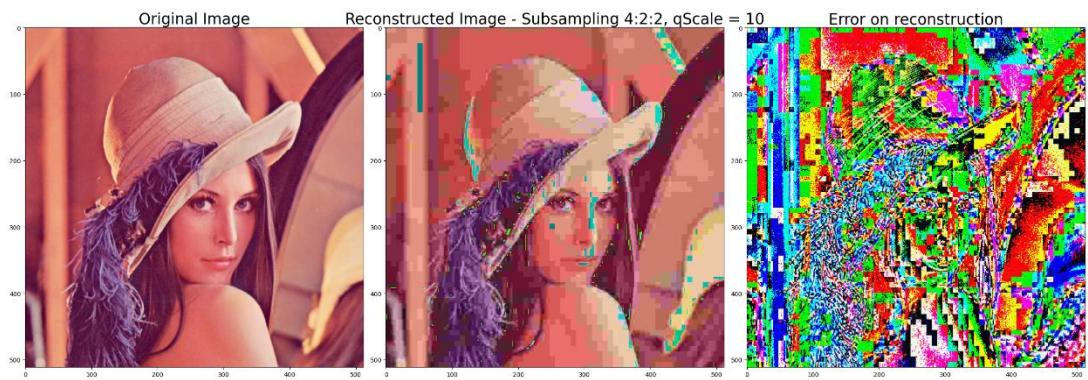
Εικόνα 26: Lena Image and Reconstruction and Error for qScale = 1



Εικόνα 27: Lena Image and Reconstruction and Error for qScale = 2

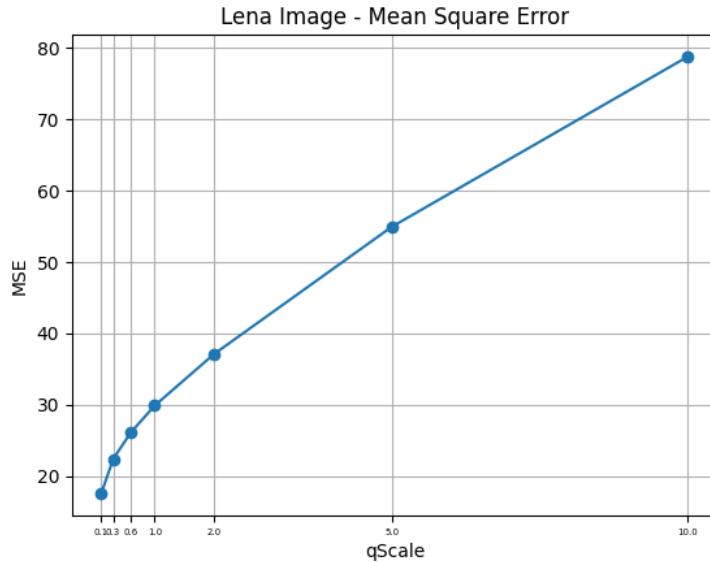


Εικόνα 28: Lena Image and Reconstruction and Error for qScale = 5



Εικόνα 29: Lena Image and Reconstruction and Error for qScale = 10

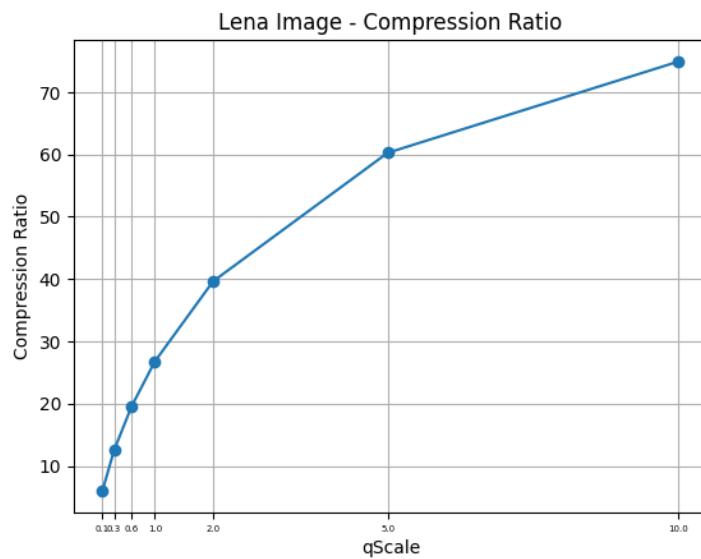
Παρατηρείται ξανά πως όσο αυξάνει το $qScale$, τόσο χαλάει το οπτικό αποτέλεσμα, ειδικά στις λεπτομέρειες και στις λεπτομέρειες της εικόνας (πχ. στην αίσθηση του βάθους που έδινε η φωτογραφία και στην ποιότητα των χρωμάτων). Η παρατήρηση επιβεβαιώνεται και από την αύξηση του MSE όσο αυξάνεται το $qScale$. Ακολουθουν τα διαγράμματα :



Διάγραμμα 5: MSE για κάθε τιμή $qScale$

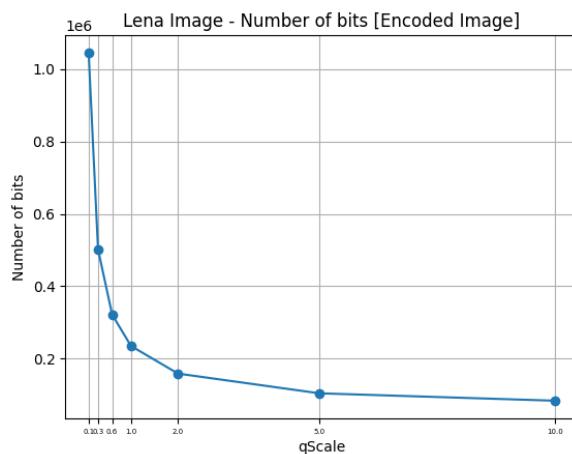
Με την ίδια μεθοδολογία, για την δεύτερη εικόνα με υποδειγματοληψία 4:4:4 και διάφορες τιμές του $qScale$, τα αποτελέσματα δίνονται παρακάτω

	$qScale$						
	0.1	0.3	0.6	1	2	5	10
<i>Compression Ratio</i>	4.8164	10.5356	16.4535	22.3755	32.1386	45.6034	53.8504



Διάγραμμα 6: Compression Ratio για κάθε τιμή $qScale$

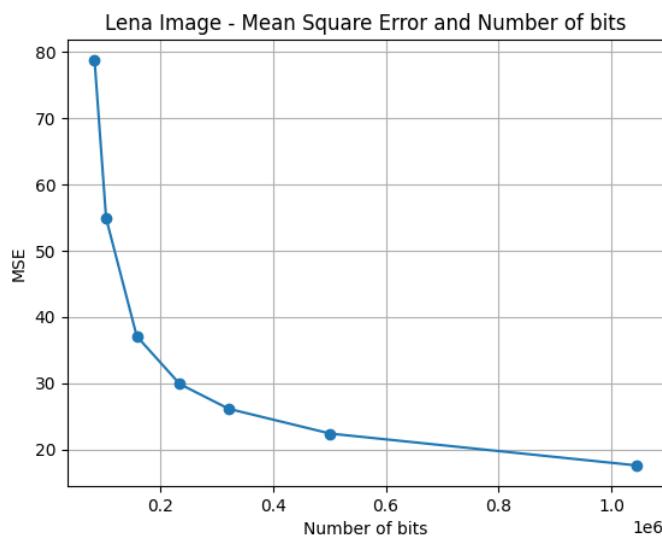
Παρατηρείται, πως όσο αυξάνει το *qScale* τόσο καλύτερος λόγος συμπίεσης επιτυγχάνεται. Βέβαια και προηγουμένως φάνηκε πως αύξηση του *qScale*, οδηγεί κι σε μεγαλύτερα σφάλματα. Τέλος, τα παραπάνω αποτελέσματα δεν αποτυπώνουν τον λόγο συμπίεσης για το πραγματικό μέγεθος της κωδικοποιημένης εικόνας, καθώς λείπουν στοιχεία κρίσιμα για την επιτυχημένη αποκωδικοποίηση της από κάποιο πρόγραμμα επεξεργασίας εικόνων (πχ. headers), ωστόσο αποδεικνύει πως τα διάφορα στάδια του προτύπου οδηγούν σε μείωση του μεγέθους της κωδικοποιημένης εικόνας.



Διάγραμμα 7: Number of bits για κάθε τιμή qScale

Ωστόσο, πρέπει να σημειωθεί πως με την αύξηση του *qScale*, παρατηρείται μείωση του αριθμού των bits της κωδικοποιημένης εικόνας.

Τέλος, δίνεται το γράφημα που δείχνει την σχέση μεταξύ του MSE και του αριθμού των bits της κωδικοποιημένης εικόνας και φαίνεται ότι η αύξηση του αριθμού των bits θα οδηγήσει σε μείωση του MSE.



Διάγραμμα 8: MSE ως προς τον αριθμό bits

Demo 2

Αναφέρθηκε και προηγουμένως πως ο μετασχηματισμός DCT, χρησιμοποιείται για την αποσυσχέτιση των δεδομένων. Μια τέτοια διαδικασία θα μειώσει την εντροπία ανά σύμβολο του συστήματος και θα οδηγήσει σε καλύτερη συμπίεση. Επιπλέον, κατά τον υπολογισμό των μηκών διαδρομής ευελπιστούμε σε μείωση των υπό προς κωδικοποίηση συμβόλων, με σκοπό την μείωση της συνολικής εντροπίας της εικόνας. Για την εντροπία ισχύει :

$$\text{Entropy} = \sum_i -f_i * \log_2(f_i) \quad [\text{per symbol}]$$

Παρακάτω γίνεται σύγκριση μεταξύ της εντροπίας ανά σύμβολο και της συνολικής εντροπίας της εικόνας σε τρία στάδια της κωδικοποίησης.

Spatial Domain – RGB: Η εντροπία υπολογίζεται ως το άθροισμα των εντροπιών των καναλιών R, G και B. Για να ισχύει κάτι τέτοιο, πρέπει να γίνει η παραδοχή ότι οι πιθανότητες εμφάνισης των τιμών είναι ανεξάρτητες για κάθε κανάλι χρώματος. Δηλαδή για μια τριάδα τιμών [r g b], ισχύει $p(x, y, z) = p(x)p(y)p(z)$. Τότε,

$$\text{Image Entropy} = \text{EntropyRed} * \text{length}(Red) + \text{EntropyGreen} * \text{length}(Green) + \text{EntropyBlue} * \text{length}(Blue)$$

Quantized DCT Coefficients: Όμοια με την παραπάνω περίπτωση.

$$\text{Image Entropy} = \text{EntropyY}*length(Y) + \text{EntropyCb}*length(Cb) + \text{EntropyCr}*length(Cr)$$

RunLengths: Υπολογίζεται η συχνότητα εμφάνισης του κάθε ζεύγους τιμών. Στην συνέχεια γίνεται χρήση του τύπου της εντροπίας.

$$\text{Image Entropy} = \text{Entropy} * \text{length}(Runlengths)$$

Υπολογισμος Εντροπιας

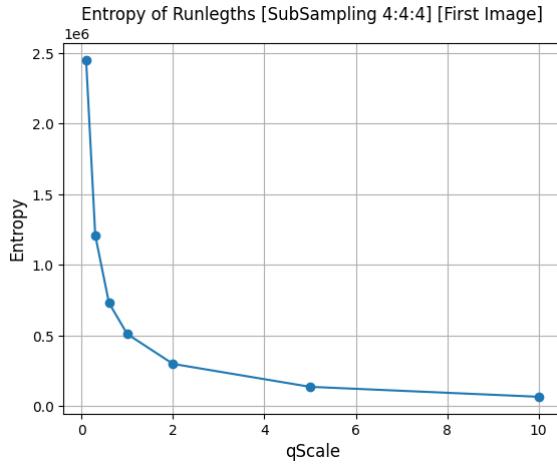
Baboon Image

Στην συνέχεια παρουσιάζονται οι εντροπίες υπολογισμένες για την πρώτη εικόνα, με $qScale = 0.6$ και υποδειγματοληψία 4: 2: 2, όπου η συνολική εντροπία μειώνεται, που ήταν και το ζητούμενο.

	Entropy	
	per Symbol	Total
Spatial Domain – RGB	22.9333	6011832.3956
Quantized DCT Coefficients	3.7226	785132.5043
RunLengths	5.0685	618788.4881

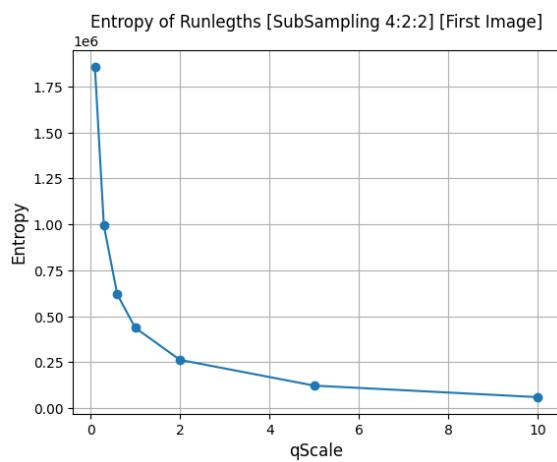
Για να παρατηρηθεί η σχέση μεταξύ υποδειγματοληψίας, *qScale* και συνολικής εντροπίας, παρακάτω παρουσιάζονται για τα Runlengths της πρώτης εικόνα οι διάφοροι συνδυασμοί υποδειγματοληψίας και *qScale*.

```
[Baboon Image] ~ The entropy for the Runlength is: 2.449018e+06. [SubSampling 4:4:4], [qScale = 0.10]
[Baboon Image] ~ The entropy for the Runlength is: 1.209370e+06. [SubSampling 4:4:4], [qScale = 0.30]
[Baboon Image] ~ The entropy for the Runlength is: 7.315975e+05. [SubSampling 4:4:4], [qScale = 0.60]
[Baboon Image] ~ The entropy for the Runlength is: 5.113598e+05. [SubSampling 4:4:4], [qScale = 1.00]
[Baboon Image] ~ The entropy for the Runlength is: 3.002670e+05. [SubSampling 4:4:4], [qScale = 2.00]
[Baboon Image] ~ The entropy for the Runlength is: 1.382248e+05. [SubSampling 4:4:4], [qScale = 5.00]
[Baboon Image] ~ The entropy for the Runlength is: 6.781445e+04. [SubSampling 4:4:4], [qScale = 10.00]
```



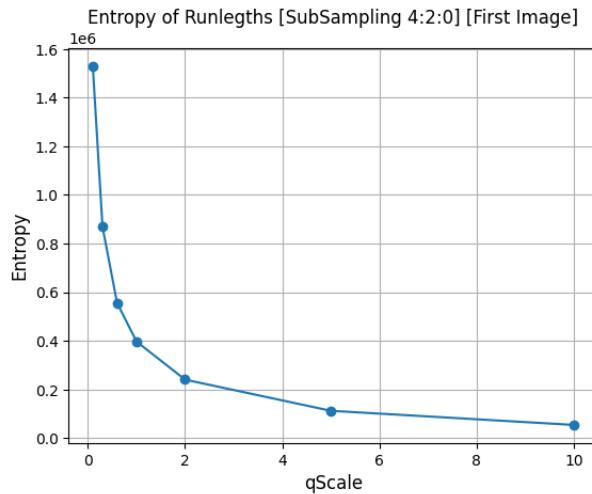
Διάγραμμα 9 :Baboon Image : Συνολική εντροπία Runlengths και *qScale*

```
[Baboon Image] ~ The entropy for the Runlength is: 1.857505e+06. [SubSampling 4:2:2], [qScale = 0.10]
[Baboon Image] ~ The entropy for the Runlength is: 9.941580e+05. [SubSampling 4:2:2], [qScale = 0.30]
[Baboon Image] ~ The entropy for the Runlength is: 6.187885e+05. [SubSampling 4:2:2], [qScale = 0.60]
[Baboon Image] ~ The entropy for the Runlength is: 4.377748e+05. [SubSampling 4:2:2], [qScale = 1.00]
[Baboon Image] ~ The entropy for the Runlength is: 2.629145e+05. [SubSampling 4:2:2], [qScale = 2.00]
[Baboon Image] ~ The entropy for the Runlength is: 1.229095e+05. [SubSampling 4:2:2], [qScale = 5.00]
[Baboon Image] ~ The entropy for the Runlength is: 6.004645e+04. [SubSampling 4:2:2], [qScale = 10.00]
```



Διάγραμμα 10 :Baboon Image : Συνολική εντροπία Runlengths και *qScale*

```
[Baboon Image] ~ The entropy for the Runlength is: 1.528867e+06. [SubSampling 4:2:0], [qScale = 0.10]
[Baboon Image] ~ The entropy for the Runlength is: 8.683910e+05. [SubSampling 4:2:0], [qScale = 0.30]
[Baboon Image] ~ The entropy for the Runlength is: 5.553849e+05. [SubSampling 4:2:0], [qScale = 0.60]
[Baboon Image] ~ The entropy for the Runlength is: 3.967224e+05. [SubSampling 4:2:0], [qScale = 1.00]
[Baboon Image] ~ The entropy for the Runlength is: 2.401655e+05. [SubSampling 4:2:0], [qScale = 2.00]
[Baboon Image] ~ The entropy for the Runlength is: 1.120430e+05. [SubSampling 4:2:0], [qScale = 5.00]
[Baboon Image] ~ The entropy for the Runlength is: 5.390419e+04. [SubSampling 4:2:0], [qScale = 10.00]
```



Διάγραμμα 11 :Baboon Image : Συνολική εντροπία Runlengths και qScale

Είναι φανερό πως η συνολική εντροπία αφενός μειώνεται κατά την αύξηση του *qScale* και αφετέρου για το ίδιο *qScale* η εντροπία είναι μικρότερη για μεγαλύτερες υποδειγματοληψίες.

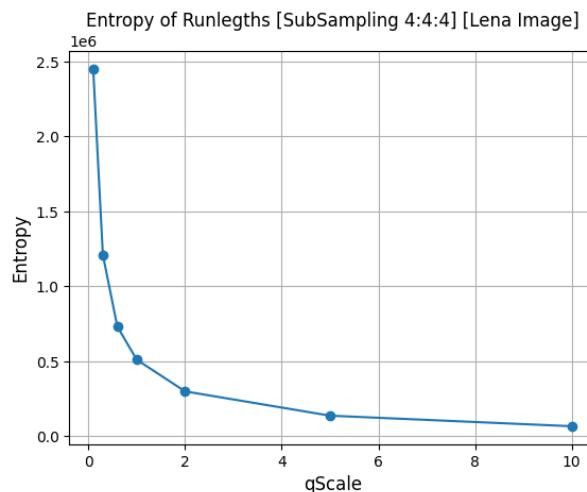
Lena Image

Ακολουθώντας το ίδιο σκεπτικό για την δεύτερη εικόνα, με *qScale* = 5 και υποδειγματοληψία 4: 4: 4.

	Entropy	
	per Symbol	Total
Spatial Domain – RGB	21.8155	5718820.0532
Quantized DCT Coefficients	2.1695	445996.3176
RunLengths	4.9593	298510.7536

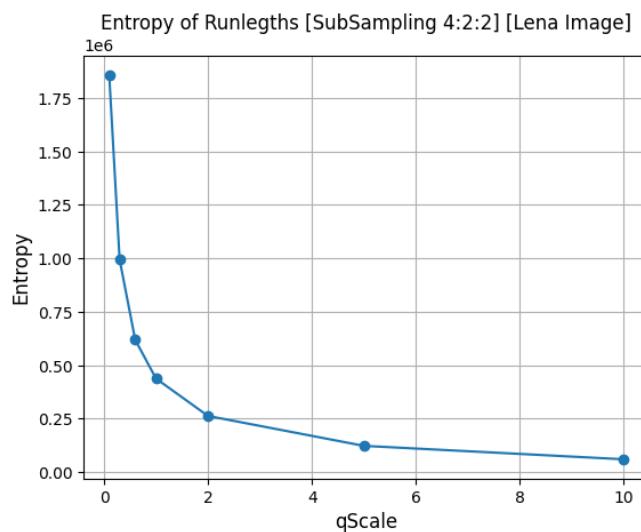
Ομοίως, γίνεται ανάλυση για την 2^η εικόνα:

```
[Lena Image] ~ The entropy for the Runlength is: 2.449018e+06. [SubSampling 4:4:4], [qScale = 0.10]
[Lena Image] ~ The entropy for the Runlength is: 1.209370e+06. [SubSampling 4:4:4], [qScale = 0.30]
[Lena Image] ~ The entropy for the Runlength is: 7.315975e+05. [SubSampling 4:4:4], [qScale = 0.60]
[Lena Image] ~ The entropy for the Runlength is: 5.113598e+05. [SubSampling 4:4:4], [qScale = 1.00]
[Lena Image] ~ The entropy for the Runlength is: 3.002670e+05. [SubSampling 4:4:4], [qScale = 2.00]
[Lena Image] ~ The entropy for the Runlength is: 1.382248e+05. [SubSampling 4:4:4], [qScale = 5.00]
[Lena Image] ~ The entropy for the Runlength is: 6.781445e+04. [SubSampling 4:4:4], [qScale = 10.00]
```



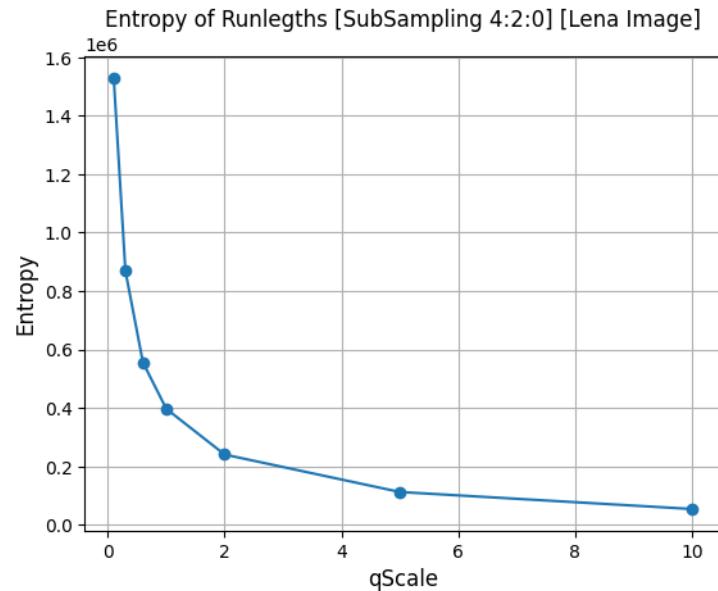
Διάγραμμα 12 :Lena Image : Συνολική εντροπία Runlengths και qScale

```
[Lena Image] ~ The entropy for the Runlength is: 1.857505e+06. [SubSampling 4:2:2], [qScale = 0.10]
[Lena Image] ~ The entropy for the Runlength is: 9.941580e+05. [SubSampling 4:2:2], [qScale = 0.30]
[Lena Image] ~ The entropy for the Runlength is: 6.187885e+05. [SubSampling 4:2:2], [qScale = 0.60]
[Lena Image] ~ The entropy for the Runlength is: 4.377748e+05. [SubSampling 4:2:2], [qScale = 1.00]
[Lena Image] ~ The entropy for the Runlength is: 2.629145e+05. [SubSampling 4:2:2], [qScale = 2.00]
[Lena Image] ~ The entropy for the Runlength is: 1.229095e+05. [SubSampling 4:2:2], [qScale = 5.00]
[Lena Image] ~ The entropy for the Runlength is: 6.004645e+04. [SubSampling 4:2:2], [qScale = 10.00]
```



Διάγραμμα 13 :Lena Image : Συνολική εντροπία Runlengths και qScale

```
[Lena Image] ~ The entropy for the Runlength is: 1.528867e+06. [SubSampling 4:2:0], [qScale = 0.10]
[Lena Image] ~ The entropy for the Runlength is: 8.683910e+05. [SubSampling 4:2:0], [qScale = 0.30]
[Lena Image] ~ The entropy for the Runlength is: 5.553849e+05. [SubSampling 4:2:0], [qScale = 0.60]
[Lena Image] ~ The entropy for the Runlength is: 3.967224e+05. [SubSampling 4:2:0], [qScale = 1.00]
[Lena Image] ~ The entropy for the Runlength is: 2.401655e+05. [SubSampling 4:2:0], [qScale = 2.00]
[Lena Image] ~ The entropy for the Runlength is: 1.120430e+05. [SubSampling 4:2:0], [qScale = 5.00]
[Lena Image] ~ The entropy for the Runlength is: 5.390419e+04. [SubSampling 4:2:0], [qScale = 10.00]
```



Διάγραμμα 14 :Lena Image : Συνολική εντροπία Runlengths και qScale

Είναι φανερό και σε αυτή την περίπτωση, πως η συνολική εντροπία αφενός μειώνεται κατά την αύξηση του *qScale* και αφετέρου για το ίδιο *qScale* η εντροπία είναι μικρότερη για μεγαλύτερες υποδειγματοληψίες.