



Πανεπιστήμιο Θεσσαλίας

Τμήμα Μηχανικών Η/Υ, Τηλ & Δικτύων – Π.Μ.Σ

Μεταπτυχιακή διατριβή:

Μέθοδοι βελτιστοποίησης Λογισμικού και Υλικού Πολυμεσικού
συστήματος σε ενσωματωμένες εφαρμογές.

Β α σ ί λ η ς Ε. Σ μ α ρ ά ι δ ο ς

Νοέμβριος 2008

Πρόλογος

Το J.P.E.G είναι ένα πρότυπο το οποίο γράφτηκε για την συμπίεση εικόνων. Το όνομα του προέρχεται από τα αρχικά του ονόματος της επιτροπής που το έγραψε (Joint Photographic Expert Group). Έγινε διεθνές το 1992. Ο στόχος της επιτροπής ήταν να εξελίξουν μια μέθοδο η οποία θα μπορούσε να συμπιέσει συνεχούς – τόνου σταθερές εικόνες η οποία θα κάλυπτε κάποιες συγκεκριμένες απαιτήσεις.

Ο στόχος της παρούσης μεταπτυχιακής διατριβής είναι να παρουσιάσει τεχνικές που πιθανόν βελτιώνουν την απόδοση του lossless JPEG κωδικοποιητή και στηρίζονται

- ❖ στους πίνακες Huffman
- ❖ στον τρόπο με τον οποίο γίνεται η κωδικοποίηση μιας εικόνας

και να ελέγξει εάν όντως πετυχαίνουν να βελτιώσουν την απόδοση του κωδικοποιητή.

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΒΙΒΛΙΟΘΗΚΗ & ΚΕΝΤΡΟ ΠΛΗΡΟΦΟΡΗΣΗΣ
ΕΙΔΙΚΗ ΣΥΛΛΟΓΗ «ΓΚΡΙΖΑ ΒΙΒΛΙΟΓΡΑΦΙΑ»**

Αριθ. Εισ.: 6709/1
Ημερ. Εισ.: 19-01-2009
Δωρεά: Συγγραφέα
Ταξιδετικός Κωδικός: Δ
621.367
ΣΜΑ

Περιεχόμενα

Λίστα πινάκων	4
Ευχαριστίες	5
ΕΙΣΛΓΩΣΗ - JPEG	7
ΚΕΦΑΛΛΑΙΟ 1 ^ο - Μαθηματικοί ορισμοί και χαρακτηριστικά εικόνας του JPEG.....	21
ΚΕΦΑΛΛΑΙΟ 2 ^ο - Κωδικοποίηση χωρίς απώλειες δεδομένων – Lossless encoding.....	33
ΚΕΦΑΛΛΑΙΟ 3 ^ο - Σύνταξη συμπιεσμένων δεδομένων	38
ΚΕΦΑΛΛΑΙΟ 4 ^ο - Διαδικασίες lossless κωδικοποιητή & αποκωδικοποιητή.....	51
ΚΕΦΑΛΛΑΙΟ 5 ^ο - Καθορισμός προσαρμοσμένων Huffman πινάκων	64
ΚΕΦΑΛΛΑΙΟ 6 ^ο - Τεχνικές βελτιστοποίησης lossless JPEG κωδικοποιητή.....	69
ΠΑΡΑΡΤΗΜΑ Α - Αναγνώριση προτύπων	98
ΑΝΑΦΟΡΕΣ	100

Λίστα πινάκων

Πίνακας 2.1 : <i>predictors</i> για <i>lossless</i> κωδικοποίηση	35
Πίνακας 2.2 : κατηγορίες διαφοράς για <i>lossless Huffman</i> κωδικοποίηση	37
Πίνακας 3.1 : <i>markers</i> του προτύπου JPEG	40
Πίνακας 3.2: επεξήγηση παραμέτρων επικεφαλίδας καρτέ	44
Πίνακας 3.3: παράμετροι επικεφαλίδας καρτέ	44
Πίνακας 3.4: παράμετροι επικεφαλίδας σάρωσης	45
Πίνακας 3.5: επεξήγηση παραμέτρων επικεφαλίδας σάρωσης	46
Πίνακας 3.6: επεξήγηση παραμέτρων καθορισμού πινάκων Huffman	48
Πίνακας 3.7: παράμετροι καθορισμού πινάκων Huffman	48
Πίνακας 3.8: επεξήγηση παραμέτρων <i>restart interval</i>	49
Πίνακας 3.9: παράμετροι καθορισμού <i>restart interval</i>	50
Πίνακας 4.1: <i>markers</i> που αναγνωρίζονται από την διεργασία « <i>Interpret markers</i> »	59
Πίνακας 6.1 : αρχεία .c πηγαίου κώδικα	70
Πίνακας 6.2 : αρχεία .h πηγαίου κώδικα	71
Πίνακας 6.3 : Αποτελέσματα κωδικοποίησης για τις διάφορες επιλογές	83
Πίνακας 6.4 : Αποτελέσματα κωδικοποίησης (βελτ 1 ^η)	87-88
Πίνακας 6.5 : Αποτελέσματα κωδικοποίησης (βελτ 2 ^η)	90-91
Πίνακας 6.6 : Αποτελέσματα κωδικοποίησης (βελτ 3 ^η)	95-96

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον καθηγητή και πρόεδρο του Τμήματος Μηχανικών Η/Υ , Τηλ & Δικτύων, κύριο Γεώργιο Σταμούλη. Η βοήθειά του όλα αυτά τα χρόνια ήταν ανεκτίμητη, η συμπαράσταση του αμέριστη, οι συμβουλές του πολύτιμες.

Επίσης, θα ήθελα να ευχαριστήσω τον καθηγητή του τμήματος, κύριο Ιωάννη Κατσαβουνίδη που μου έδωσε την ευκαιρία να πραγματοποιήσω αυτή την μεταπτυχιακή διατριβή. Σε όλη τη διάρκεια της διατριβής, με δίδαξε, με προβλημάτισε, με καθοδήγησε και με ενέπνευσε για να ολοκληρώσω το έργο μου.

Ακόμη θα ήθελα να ευχαριστήσω όλους τους μεταπτυχιακούς φοιτητές και υποψήφιους διδάκτορες του γραφείου Ε5 για την συνεργασία που είχαμε όλα αυτά τα χρόνια, φίλους και γνωστούς.

Τέλος, δεν έχω λόγια για την οικογένεια μου και τους γονείς μου που με στήριξαν όλα αυτά τα χρόνια, ως προπτυχιακό και μεταπτυχιακό φοιτητή στο τμήμα Μηχανικών Η/Υ , Τηλ & Δικτύων, με όλους τους δυνατούς τρόπους.

Στους γονείς μου,
Ευάγγελο κ Θεοδώρα που μου
έδωσαν την ευκαιρία να
σπουδάσω.
Σε όσους πίστευαν και υπέμειναν...

Εισαγωγή

JPEG

Περιεχόμενα

1. Εισαγωγή	8
2.Κωδικοποίηση και Αποκωδικοποίηση	9
2.1.Μέθοδοι συμπίεσης JPEG	10
2.1.1.Κωδικοποίηση βασισμένη σε DCT (DCT-based coding)	10
2.1.2.Κωδικοποίηση χωρίς απώλειες δεδομένων(Lossless coding)	12
2.2.Τρόποι Εκτέλεσης JPEG	13
2.3.Ακρίβεια δείγματος	16
2.4. Interleaving πολλαπλά συστατικά	16
2.5.Μονάδα πληροφορίας	17
2.6.Interleaved and non-interleaved	17
2.7.Μικρότερη κωδικοποιημένη μονάδα (MCU)	18
2.8Δομή συμπιεσμένων δεδομένων	18
3.Περίληψη	19

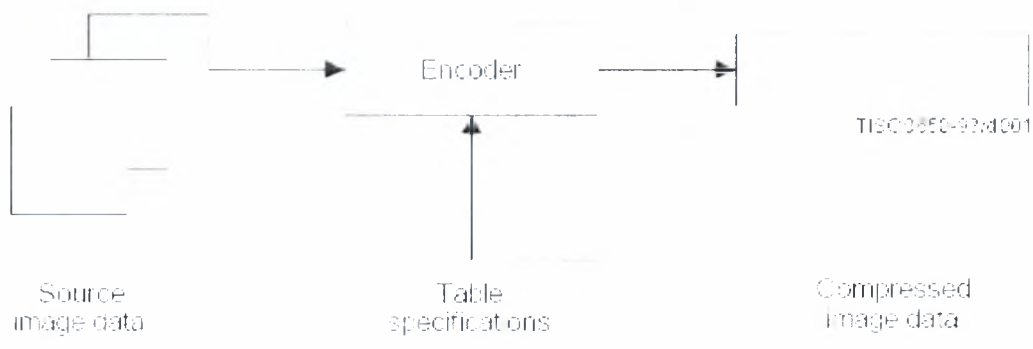
1. Εισαγωγή

Η ITU (International Telecommunication Union) είναι η ειδικευμένη αντιπροσωπεία των Ηνωμένων Εθνών στον τομέα των τηλεπικοινωνιών. Η CCITT (the International Telegraph and Telephone Consultative Committee) είναι ένα μόνιμο όργανο της ITU. Περίπου 166 χώρες μέλη, 163 επιστημονικές και βιομηχανικές οργανώσεις και 39 διεθνείς οργανισμοί συμμετέχουν στη CCITT που είναι το σώμα που καθορίζει τα πρότυπα παγκόσμιων τηλεπικοινωνιών. Η επιτροπή ISO/CCITT γνωστή ως JPEG έχει εργαστεί για να καθιερώσει τα πρώτα διεθνή πρότυπα συμπίεσης για τις συνεχούς τόνου σταθερές εικόνες, γκρι είτε έγχρωμες. Το πρότυπο JPEG στοχεύει να υποστηρίξει μια ευρεία ποικιλία των εφαρμογών για τις εικόνες συνεχούς τόνου καλύπτοντας τις εξής απαιτήσεις:

- ❖ Να μπορεί να παραμετροποιηθεί ο κωδικοποιητής έτσι ώστε να καλύπτει τις απαιτήσεις του εκάστοτε χρήστη ή εφαρμογής.
- ❖ Να μπορεί να εφαρμοστεί σε κάθε είδος συνεχούς τόνου ψηφιακή εικόνα ανεξαρτήτως διαστάσεων και εύρους χρωμάτων.
- ❖ Να έχει την κατάλληλη υπολογιστική πολυπλοκότητα για να πραγματοποιηθούν εφικτές υλοποιήσεις σε εφαρμογές λογισμικού, καθώς επίσης και σε εφαρμογές υλικού.
- ❖ Να καλύπτει τους ακόλουθους τρόπους εκτέλεσης:
 - Σειριακή κωδικοποίηση (Sequential encoding)
 - Προοδευτική κωδικοποίηση (Progressive encoding)
 - Ιεραρχική κωδικοποίηση (Hierarchical encoding)
 - Κωδικοποίηση χωρίς απώλειες δεδομένων (Lossless encoding).

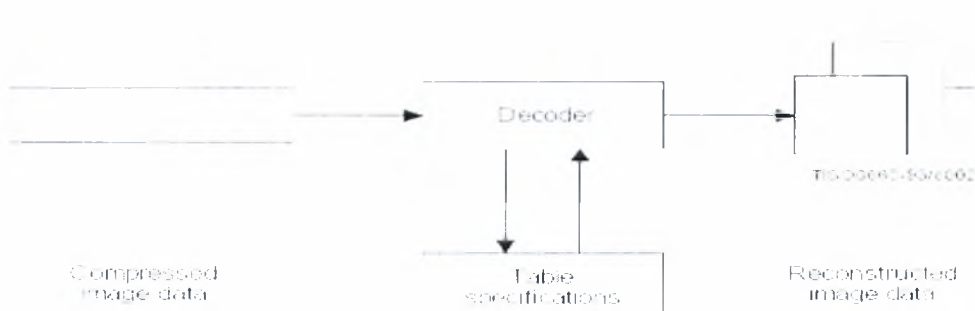
2. Κωδικοποίηση και Αποκωδικοποίηση

Ένας κωδικοποιητής πραγματοποιεί τη διαδικασία της κωδικοποίησης. Όπως φαίνεται στην [Εικόνα 2.1](#), ένας κωδικοποιητής παίρνει ως είσοδο μια εικόνα και κάποιους προκαθορισμένους πίνακες και με τη βοήθεια ενός διευκρινισμένου συνόλου διαδικασιών παράγει μια συμπιεσμένη εικόνα.



Εικόνα 2.1

Ένας αποκωδικοποιητής πραγματοποιεί τη διαδικασία της αποκωδικοποίησης. Όπως φαίνεται στην [Εικόνα 2.2](#), ένας αποκωδικοποιητής παίρνει ως είσοδο μια συμπιεσμένη εικόνα και κάποιους προκαθορισμένους πίνακες, και με τη βοήθεια ενός διευκρινισμένου συνόλου διαδικασιών αναδημιουργεί την αρχική εικόνα.



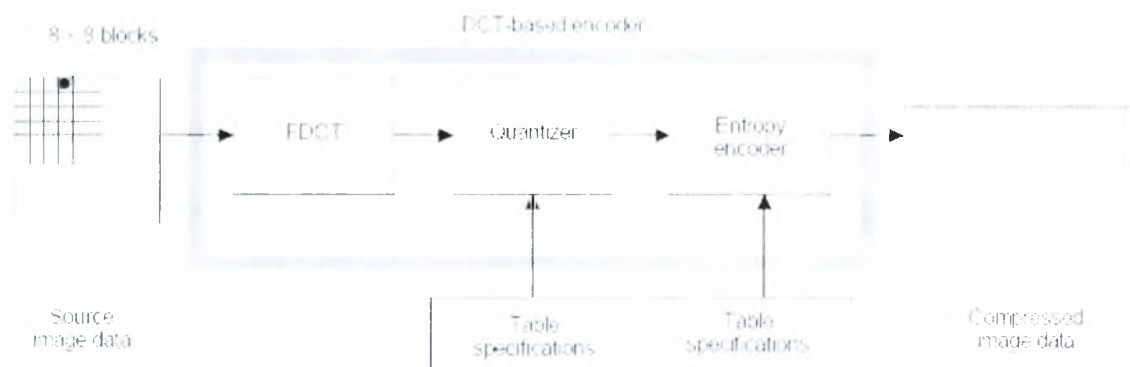
2.1 Μέθοδοι συμπίεσης JPEG

Το Jpeg περιλαμβάνει δυο διαφορετικές διαδικασίες κωδικοποίησης κ αποκωδικοποίησης.

- ❖ Lossy διαδικασία η οποία βασίζεται στον διακριτό μετασχηματισμό συνημίτονου (DCT) και είναι πολύ αποτελεσματική σε πολλές εφαρμογές.
- ❖ Lossless διαδικασία η οποία βασίζεται στην πρόβλεψη.

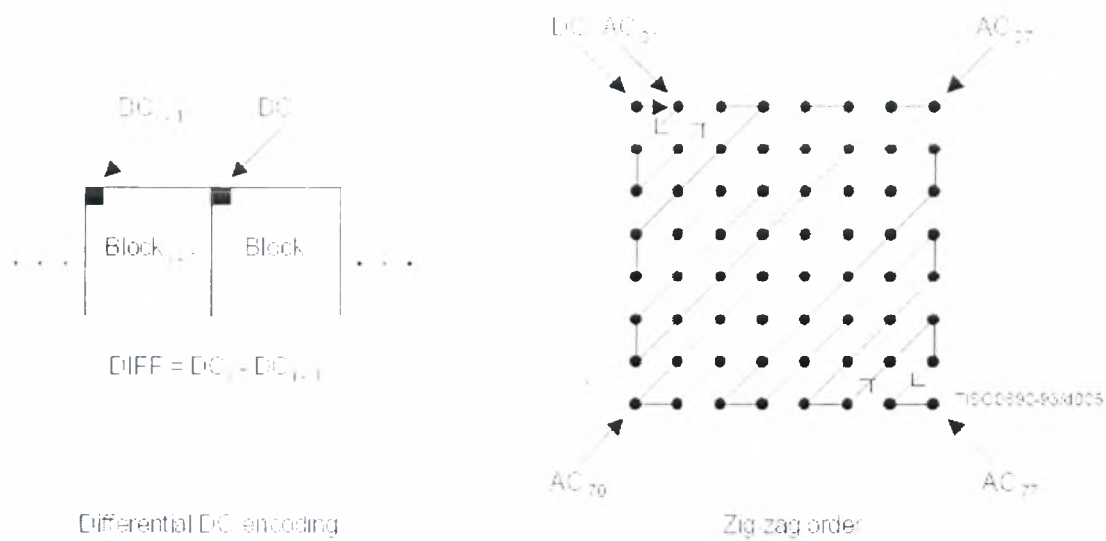
2.1.1 Κωδικοποίηση βασισμένη σε DCT [1]

Η **Εικόνα 0.3** παρουσιάζει την διαδικασία κωδικοποίησης που βασίζεται στον DCT. Παρουσιάζεται η εκδοχή στην οποία η εικόνα αποτελείται μόνο από ένα συστατικό. Τα κύρια βήματα της κωδικοποίησης είναι αρχικά ευθύ μετασχηματισμός συνημίτονου, η κβαντοποίηση και τέλος κωδικοποίηση εντροπίας. Συγκεκριμένα, τα δείγματα της εικόνας εισόδου ομαδοποιούνται σε μπλοκ 8X8 και το κάθε μπλοκ μετασχηματίζεται με τον ευθύ διακριτό μετασχηματισμό συνημίτονου (FDCT) σε ένα σύνολο από 64 τιμές οι οποίες αναφέρονται ως συντελεστές του DCT. Ο πρώτος από αυτούς τους συντελεστές αναφέρεται ως DC συντελεστής ενώ οι άλλοι 63 ως AC συντελεστές.



Κάθε ένας από αυτούς τους 64 συντελεστές κβαντοποιείται χρησιμοποιώντας την αντίστοιχη τιμή από τις εξήντα τέσσερις από έναν πίνακα κβαντοποίησης. Για αυτούς τους πίνακες κβαντοποίησης δεν υπάρχουν συγκεκριμένες τιμές αλλά ορίζονται κάθε φορά από την εφαρμογή προκειμένου να επιτύχει συγκεκριμένη ποιότητα εικόνας.

Μετά την κβαντοποίηση, οι συντελεστές προετοιμάζονται για κωδικοποίηση εντροπίας όπως φαίνεται στην

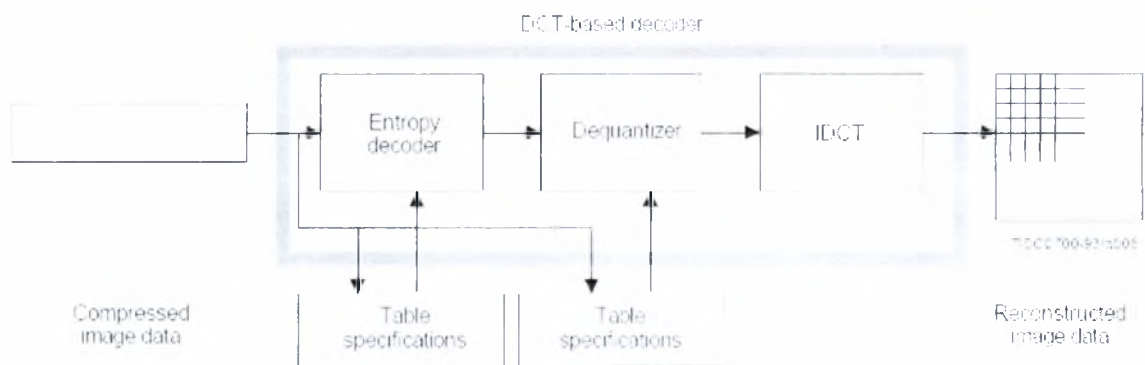


Εικόνα 1.1.1: Διαδικασία κβαντοποίησης και κωδικοποίησης εντροπίας

Ο κβαντοποιημένος DC συντελεστής του προηγούμενου μπλοκ χρησιμοποιείται για να προβλεφθεί ο κβαντοποιημένος DC συντελεστής του τρέχοντος μπλοκ και αυτό που κωδικοποιείται είναι η διαφορά τους. Όσον αφορά τους κβαντοποιημένους AC συντελεστές, αυτοί μετατρέπονται σε μια zig-zag ακολουθία μιας διάστασης όπως φαίνεται στην

Οι κβαντοποιημένοι συντελεστές εισάγονται στην διαδικασία κωδικοποίησης εντροπίας η οποία συμπιέζει τα δεδομένα περισσότερο. Υπάρχουν δύο τρόποι κωδικοποίησης εντροπίας. Η κωδικοποίηση Huffman και η αριθμητική κωδικοποίηση.

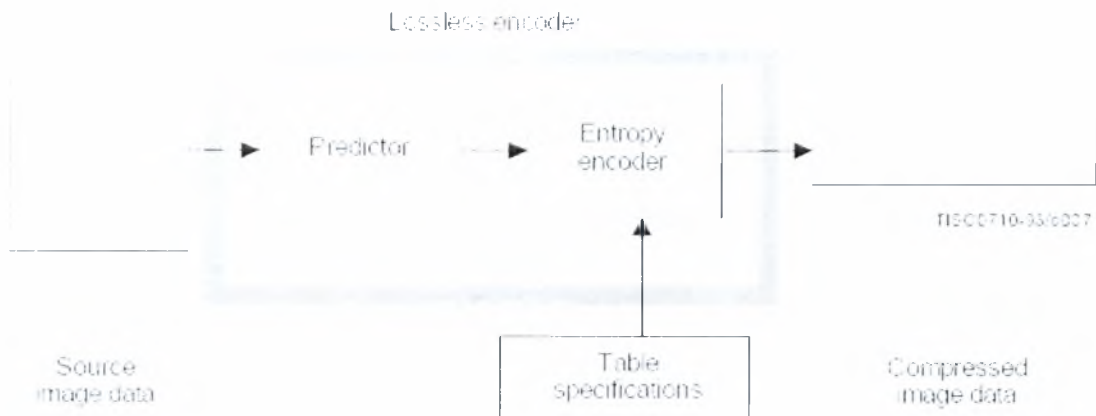
Η [εικόνα 4.5](#) παρουσιάζει τις κύριες διαδικασίες αποκωδικοποίησης οι οποίες βασίζονται στον διακριτό μετασχηματισμό συνημίτονου. Κάθε βήμα που παρουσιάζεται εκτελεί ουσιαστικά την αντίστροφη διαδικασία που παρουσιάστηκε στον κωδικοποιητή. Ο αποκωδικοποιητής αποκωδικοποιεί την zig-zag ακολουθία από τους κβαντοποιημένους DCT συντελεστές. Μετά την αποκβαντοποίηση, οι DCT συντελεστές μετασχηματίζονται σε μπλοκ των 8X8 χρησιμοποιώντας τον αντίστροφο διακριτό μετασχηματισμό συνημίτονου (IDCT).



[Εικόνα 4.5](#) Δεδομένα αποκωδικοποίησης

2.1.2 Κωδικοποίηση χωρίς απώλειες με πρόβλεψη (lossless prediction)

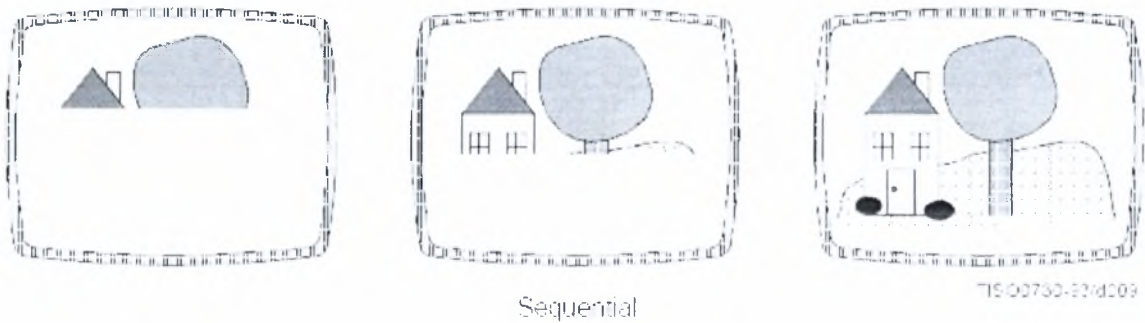
Το [πρωτόκολλο 2.1](#) παρουσιάζει την κύρια διαδικασία για την κωδικοποίηση χωρίς απώλειες δεδομένων. Ο μηχανισμός πρόβλεψης (predictor) συνδυάζει τις ανακατασκευασμένες τιμές από τα τρία γειτονικά δείγματα στις θέσεις α , β , γ για να διαμορφώσει μία πρόβλεψη του δείγματος στη θέση x όπως φαίνεται στην [εικόνα 2.1](#). Αυτή η πρόβλεψη τότε αφαιρείται από την πραγματική τιμή του δείγματος στη θέση x και η διαφορά κωδικοποιείται είτε με Huffman είτε με αριθμητική κωδικοποίηση.



Υπάρχουν συγκεκριμένοι τρόποι εκτέλεσης κάτω από τις οποίες καθορίζονται διαδικασίες κωδικοποίησης. Αυτοί οι τρόποι είναι οι ακόλουθοι:

- A. σειριακή κωδικοποίηση η οποία εκτελείται από πάνω προς τα κάτω και από τα δεξιά προς τα αριστερά σε μια εικόνα.
- B. προοδευτική κωδικοποίηση η οποία στηρίζεται στις πολλαπλές σαρώσεις.
- Γ. Ιεραρχική κωδικοποίηση
- Δ. Κωδικοποίηση χωρίς απώλειες δεδομένων

Για τον σειριακό τρόπο ο οποίος βασίζεται σε DCT, τα 8x8 μπλοκ από δείγματα εισάγονται μπλοκ - μπλοκ από αριστερά προς τα δεξιά και από πάνω προς τα κάτω όπως φαίνεται στην [εικόνα 11.1](#). Εφόσον ένα μπλοκ έχει μετασχηματιστεί με τον ευθύ μετασχηματισμό συνημίτονου, κβαντοποιείται και στη συνέχεια προετοιμάζεται για κωδικοποίηση εντροπίας.



Sequential

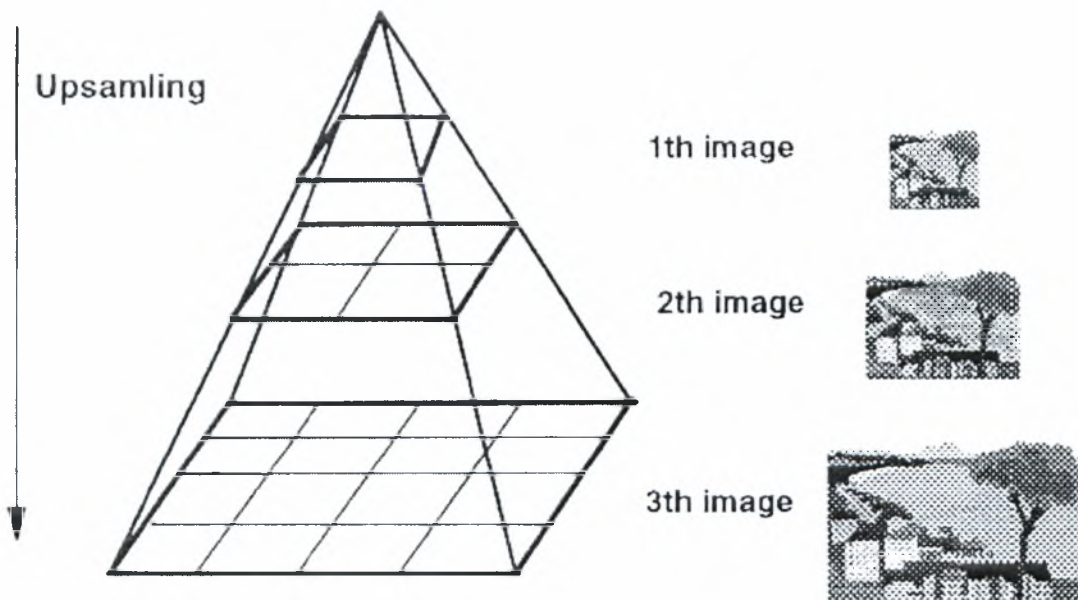
Για τον προοδευτικό τρόπο βασισμένο σε DCT, τα μπλοκ κωδικοποιούνται με την ίδια σειρά όπως παρουσιάστηκε παραπάνω αλλά σε πολλαπλές σαρώσεις στην εικόνα όπως φαίνεται στην [εικόνα 11.2](#). Αυτό επιτυγχάνεται τοποθετώντας έναν buffer ανάμεσα από τον κβαντοποιητή και την μηχανισμό που πραγματοποιεί η κωδικοποίηση εντοπίας. Καθώς κάθε μπλοκ μετασχηματίζεται με τον ευθύ μετασχηματισμό συνημίτονου και κβαντοποιείται, οι συντελεστές του αποθηκεύονται στον buffer. Οι DCT συντελεστές στον buffer κωδικοποιούνται μερικώς σε κάθε μια από τις πολλαπλές σαρώσεις.



Progressive

Υπάρχουν δυο διαδικασίες με τις οποίες οι κβαντοποιημένοι συντελεστές στον buffer κωδικοποιούνται μερικώς μέσα σε μια σάρωση. Πρώτα, μόνο μια συγκεκριμένη ζώνη από συντελεστές από την ακολουθία zig-zag χρειάζεται να κωδικοποιηθεί. Αυτή η διαδικασία καλείται φασματική επιλογή επειδή κάθε ζώνη περιέχει συντελεστές που καταλαμβάνουν χαμηλό ή υψηλό μέρος από το φάσμα συχνοτήτων για το κάθε 8x8 μπλοκ. Δεύτερον, οι συντελεστές μέσα στην τρέχουσα ζώνη δεν χρειάζεται να κωδικοποιηθούν με πλήρης ακρίβεια μέσα σε μια σάρωση. Κωδικοποιούνται αρχικά τα περισσότερο σημαντικά bits.

Στον ιεραρχικό τρόπο, μια εικόνα κωδικοποιείται σαν μια ακολουθία από καρέ. Αυτά τα καρέ συνήθως χρησιμοποιούνται για πρόβλεψη σε ακόλουθα καρέ. Η κωδικοποίηση ξεκινάει με μικρότερα καρέ και συνεχίζει με μεγαλύτερα. Το κάθε καρέ κλιμακώνεται και χρησιμοποιείται ως μηχανισμός πρόβλεψης στην επόμενη ανάλυση όπως φαίνεται στην

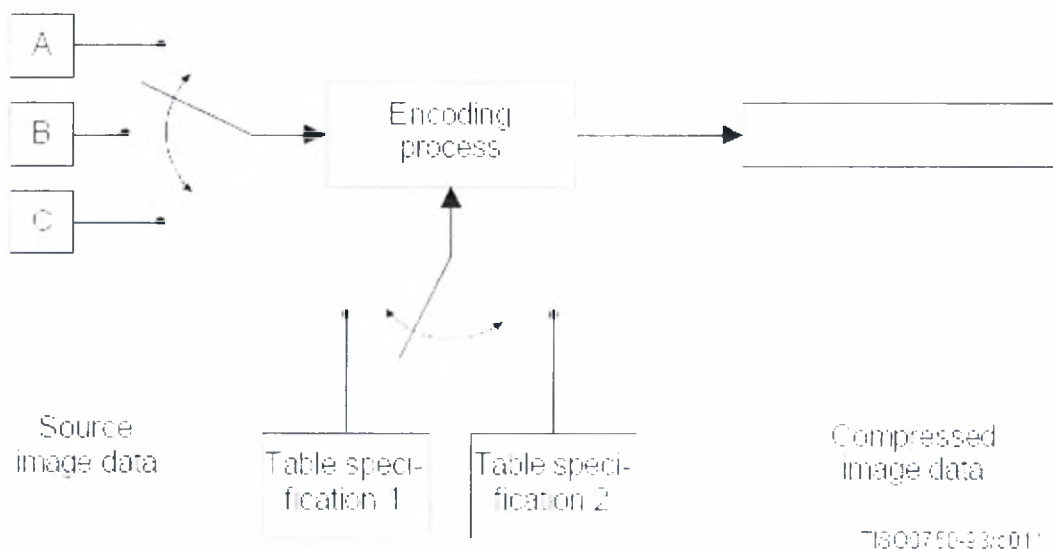


2.3 Ακρίβεια δείγματος

Για τις διαδικασίες που βασίζονται σε DCT, δυο διαφορετικές ακρίβειες δείγματος καθορίζονται: είτε 8 είτε 12 bits ανά δείγμα. Εφαρμογές οι οποίες χρησιμοποιούν δείγματα με άλλες ακρίβειες μπορούν να χρησιμοποιούσαν είτε 8 είτε 12 bits ακρίβεια ολισθαίνοντας τα δείγματα της εικόνας. Η baseline διαδικασία χρησιμοποιεί μόνο 8-bit ακρίβεια. Για τη διαδικασία χωρίς απώλειες δεδομένων, οι ακρίβειες δείγματος καθορίζονται από 2 έως 16 bits.

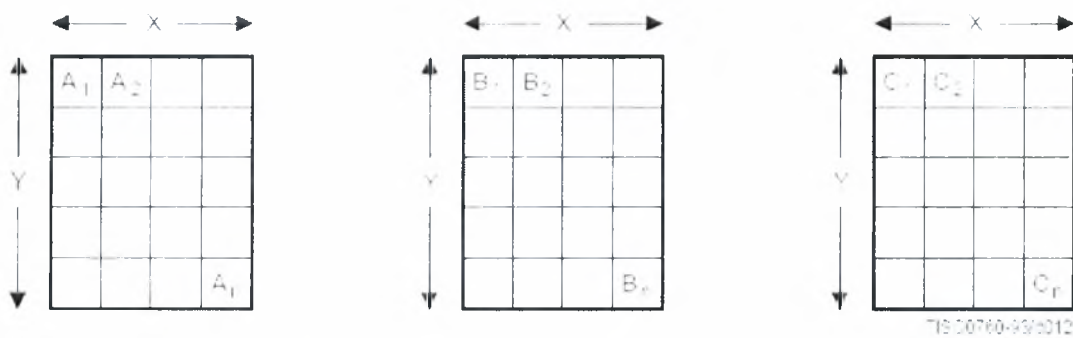
2.4 Interleaving πολλαπλά συστατικά

Η εικόνα 0.11 δείχνει ένα παράδειγμα για το πως μια διαδικασία κωδικοποίησης επιλέγει ανάμεσα από πολλαπλά στοιχεία μιας εικόνας καθώς επίσης και από μια σειρά από πίνακες δεδομένων. Στο συγκεκριμένο παράδειγμα, η εικόνα αποτελείται από 3 στοιχεία A,B,C και έχουν καθοριστεί 2 πίνακες δεδομένων για την κωδικοποίηση εντροπίας.



Μονάδα πληροφορίας (data unit) είναι ένα δείγμα για την lossless διαδικασία και ένα μπλοκ δειγμάτων 8X8 για τις διαδικασίες που βασίζονται σε DCT.

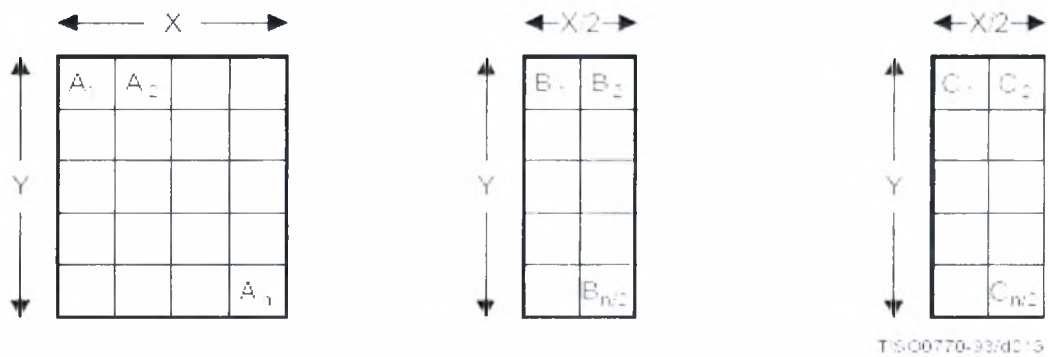
Στον σειριακό τρόπο εκτέλεσης, η διαδικασία είναι μη interleaved εάν ο κωδικοποιητής συμπιέζει όλα τα data units πρώτα από το στοιχείο A, ύστερα από B και έπειτα από το Γ. Η διαδικασία είναι interleaved όταν ο κωδικοποιητής κωδικοποιεί πρώτα μια μονάδα πληροφορίας από το A, ύστερα ένα από το B και ένα από το Γ. Υπάρχει περίπτωση και τα 3 συστατικά εικόνας να έχουν τις ίδιες διαστάσεις (εικόνα 0.12). Υπάρχει όμως και περίπτωση να μην έχουν τις ίδιες (εικόνα 0.13). Τότε οι μονάδες πληροφορίας έχουν την ακόλουθη μορφή:



Data unit encoding order: non-interleaved



Data unit encoding order: interleaved



$A_1, A_2, B_1, C_1, A_3, A_4, B_2, C_2, \dots, A_{n1}, A_n, B_{n2}, C_{n2}$
 Scan 1
 Data unit encoding order, interleaved

2.7 Μικρότερη κωδικοποιημένη μονάδα (MCU)

Το minimum coded unit ορίζεται ως η μικρότερη μονάδα που μπορεί να κωδικοποιηθεί. Εάν τα συμπιεσμένα δεδομένα είναι μη interleaved, το MCU θα είναι μια απλή μονάδα πληροφορίας. Αντίθετα, εάν τα συμπιεσμένα δεδομένα είναι interleaved, το MCU θα περιέχει ένα ή περισσότερα data units για κάθε συστατικό εικόνας.

2.8 Δομή συμπιεσμένων δεδομένων

Τα συμπιεσμένα δεδομένα εικόνας έχουν συγκεκριμένη δομή και ένα σύνολο από παραμέτρους. Τα διάφορα μέρη της πληροφορίας μιας συμπιεσμένης εικόνας αναγνωρίζονται από ειδικούς κωδικούς 2-bytes τα οποία καλούνται markers.

3. Περίληψη

Διαδικασία baseline (βασισμένη σε DCT)

- Διαδικασία βασισμένη στο διακριτό μετασχηματισμό συνημίτονου
- Τα δείγματα είναι 8-bit ανά συστατικό εικόνας
- Σειριακή
- Κωδικοποίηση Huffman: 2 AC και 2 DC πίνακες.
- Οι αποκωδικοποιητές θα εκτελέσουν σαρώσεις με 1,2,3 και 4 συστατικά εικόνας.
- Interleaved και non –interleaved σαρώσεις.

Εκτεταμένη διαδικασία βασισμένη σε DCT

- Διαδικασία βασισμένη στο διακριτό μετασχηματισμό συνημίτονου
- Τα δείγματα είναι 8-bit ή 12 bits
- Σειριακή ή ακολουθιακή
- Κωδικοποίηση Huffman ή Αριθμητική: 4 AC και 4 DC πίνακες.
- Οι αποκωδικοποιητές θα εκτελέσουν σαρώσεις με 1,2,3 και 4 συστατικά εικόνας.
- Interleaved και non –interleaved σαρώσεις.

Διαδικασία χωρίς απώλειες δεδομένων

- Διαδικασία πρόβλεψης
- Τα δείγματα έχουν ακρίβεια από 2 έως 16 bits.
- Σειριακή
- Κωδικοποίηση Huffman ή Αριθμητική: 4 DC πίνακες.
- Οι αποκωδικοποιητές θα εκτελέσουν σαρώσεις με 1,2,3 και 4 συστατικά εικόνας.
- Interleaved και non –interleaved σαρώσεις.

Ιεραρχική διαδικασία

- Πολλαπλά πλαίσια.
- Χρησιμοποιούνται εκτεταμένες διαδικασίες βασισμένες σε DCT ή σε διαδικασίες χωρίς απώλειες δεδομένων.
- Οι αποκωδικοποιητές θα εκτελέσουν σαρώσεις με 1,2,3 και 4 συστατικά εικόνας.
- Interleaved και non –interleaved σαρώσεις.

Κεφάλαιο 1^ο

Περιεχόμενα

1. Εισαγωγή.....	22
2. Εικόνα.....	22
2.1 Διαστάσεις και παράγοντες δειγματοληψίας.....	22
2.2 Ακρίβεια δείγματος.....	24
2.3 Μονάδα πληροφορίας (data unit).....	24
2.4 Προσανατολισμός.....	24
3. Σειρά κωδικοποίησης της πληροφορίας εικόνας.....	24
3.1 Ελάχιστη μονάδα κωδικοποίησης (MCU).....	25
3.1.1 Σειρά κωδικοποίησης για μη interleaved ($N_s = 1$).....	25
3.1.2 Σειρά κωδικοποίησης για interleaved ($N_s > 1$).....	25
4. Baseline αλγόριθμος.....	27

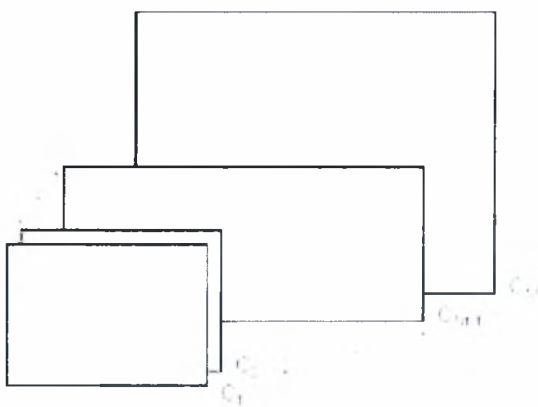
1. Εισαγωγή

Το κεφάλαιο αυτό έχει ως στόχο να παρουσιάσει χρήσιμες πληροφορίες σχετικά με τα χαρακτηριστικά της εικόνας και με την σειρά με την οποία τα συστατικά της εικόνας κωδικοποιούνται. Τέλος, έχει ως στόχο να παρουσιάσει κάποιους μαθηματικούς ορισμούς και να περιγράψει τον κύριο αλγόριθμο κωδικοποίησης εικόνας ο οποίος βασίζεται στον διακριτό μετασχηματισμό συνημίτονου και ονομάζεται baseline process.

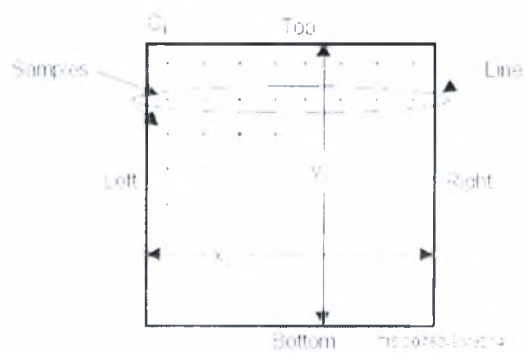
1.1 Εικόνα

1.1.1 Διαστάσεις και συστατικά

Όπως φαίνεται στην [εικόνα 1.1](#), μια εικόνα αποτελείται από N_c συστατικά. Το καθένα από αυτά έχει μοναδικό αναγνωριστικό C_i και αποτελείται από ένα ορθογώνιο πίνακα από δείγματα με x_i στήλες και y_i γραμμές. Οι διαστάσεις των συστατικών μιας εικόνας παράγονται από δύο παραμέτρους, X και Y όπου X είναι η μεγαλύτερη τιμή από αυτές τις μεταβλητές x_i σε όλα τα συστατικά του τρέχοντος πλαισίου και Y η μεγαλύτερη από τις τιμές της y_i αντίστοιχα.



a) Source image with multiple components



b) Characteristics of an image component

Για κάθε συστατικό εικόνας, οι παράγοντες δειγματοληψίας καθορίζονται να σχετίζονται με τις διαστάσεις του κάθε συστατικού σύμφωνα με τις ακόλουθες εκφράσεις:

$$x_i = \left\lceil X \times \frac{H_i}{H_{max}} \right\rceil \text{ and } y_i = \left\lceil Y \times \frac{V_i}{V_{max}} \right\rceil.$$

Όπου H_{max} και V_{max} είναι οι μεγαλύτεροι παράγοντες δειγματοληψίας για όλα τα συστατικά στο τρέχον καρέ.

Για παράδειγμα, ας θεωρήσουμε μια εικόνα η οποία αποτελείται από τρία συστατικά με μέγιστες διαστάσεις 512 γραμμές και 512 δείγματα ανά γραμμή ($X=512, Y=512$) και με τους ακόλουθους παράγοντες δειγματοληψίας:

Component 0	$H_0=4$	$V_0=1$
Component 1	$H_1=2$	$V_1=2$
Component 2	$H_2=1$	$V_2=1$

Συνεπώς $H_{max}=4$, $V_{max}=2$. Το κάθε συστατικό εικόνας θα έχει τις ακόλουθες διαστάσεις

Component 0	$x_0=512$	$y_0=256$
Component 1	$x_1=256$	$y_1=512$
Component 2	$x_2=128$	$y_2=256$

2.2 Ακρίβεια δειγμάτων

Ένα δείγμα είναι ένας ακέραιος με ακρίβεια P bits, με τιμή οποιαδήποτε ανήκει στο εύρος τιμών από 0 έως 2^P-1 . Όλα τα δείγματα σε όλα τα συστατικά της εικόνας έχουν την ίδια ακρίβεια δειγματος P . Στην lossless κωδικοποίηση, η ακρίβεια είναι από 2 έως 16 bits.

2.3 Μονάδα πληροφορίας (sample)

Μια μονάδα πληροφορίας είναι ένα δείγμα στην κωδικοποίηση χωρίς απώλειες δεδομένων (lossless) και ένα 8×8 μπλοκ από συνεχές δείγματα στην κωδικοποίηση που βασίζεται στον DCT.

2.4 Πρωτεύονα στοιχεία

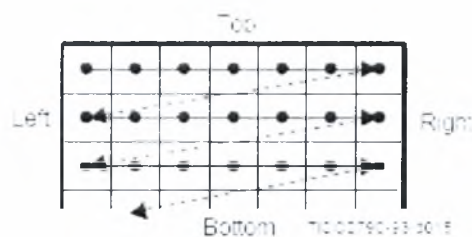
Η σειρά με την οποία οι μονάδες πληροφορίας κάθε συστατικού εικόνας εισέρχονται στην διαδικασία κωδικοποίησης είναι από αριστερά προς τα δεξιά και από πάνω προς τα κάτω μέσα σε ένα συστατικό εικόνας.

3. Σειρά κωδικοποίησης της πλαισίωσης

Η επικεφαλίδα σάρωσης (scan header) καθορίζει την σειρά με την οποία οι μονάδες πληροφορίας της εικόνας θα κωδικοποιηθούν και θα τοποθετηθούν μέσα στη συμπιεσμένη μορφή. Για την τρέχουσα σάρωση, εάν η παράμετρος του scan header είναι $N_s=1$ τότε δεδομένα από μόνο ένα συστατικό εικόνας (το οποίο καθορίζεται από την παράμετρο C_{s1}) θα βρίσκονται μέσα στο πεδίο της σάρωσης. Αυτά τα δεδομένα είναι μη interleaved εξ ορισμού. Εάν η παράμετρος είναι $N_s>1$, τότε δεδομένα από τα N_s συστατικά και συγκεκριμένα από το C_{s1} έως το C_{sN_s} θα είναι παρούσα μέσα στο πεδίο της σάρωσης. Αυτά τα δεδομένα θα είναι συνεχώς interleaved. Η σειρά των συστατικών στη σάρωση θα είναι σύμφωνη με την σειρά που έχει καθοριστεί στην επικεφαλίδα καρέ (frame header).

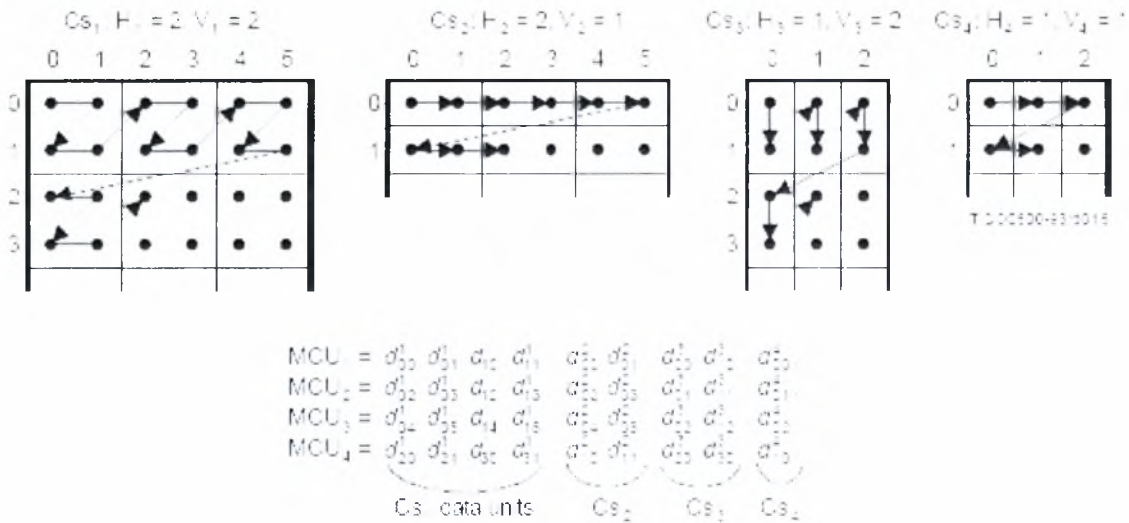
Για τα μη interleaved δεδομένα, το MCU είναι μια μονάδα πληροφορίας. Για τα interleaved δεδομένα, το MCU είναι μια ακολουθία από μονάδες πληροφορίας οι οποίες προσδιορίζονται από τους παράγοντες δειγματοληψίας των συστατικών εικόνων στη σάρωση.

Όταν η παράμετρος N_s ισούται με ένα (όπου N_s είναι ο αριθμός των συστατικών σε μια σάρωση), η σειρά των μονάδων πληροφορίας σε μια σάρωση θα είναι από αριστερά προς τα δεξιά και από πάνω προς τα κάτω όπως φαίνεται στην [εικόνα 1.2](#).



Όταν η N_s είναι μεγαλύτερη από ένα, κάθε συστατικό σάρωσης διαμερίζεται σε μικρούς ορθογώνιους πίνακες από H_k οριζόντιες μονάδες πληροφορίας και V_k κάθετες μονάδες πληροφορίας. Όπως φαίνεται στην [εικόνα 1.3](#), όταν ο αριθμός των συστατικών εικόνας είναι τέσσερα ($N_s=4$) το MCU_1 αποτελείται από μονάδες πληροφορίας που προέρχονται από πάνω αριστερά από κάθε συστατικό Cs_i . Αντίστοιχα, στο MCU_2 οι μονάδες πληροφορίας θα προέρχονται από τις διπλανές

περιοχές σε σχέση με αυτές του MCU_1 με φορά από αριστερά προς τα δεξιά και από πάνω προς τα κάτω.



Σχήμα 1.2.1: Σειρά σάρτησης των MCUs

Τέλος, αξίζει να σημειώσουμε κάτι ακόμη. Στις διαδικασίες κωδικοποίησης που στηρίζονται στον DCT, εάν το x_i δεν είναι πολλαπλάσιο του 8, η διαδικασία κωδικοποίησης θα επεκτείνει τον αριθμό των στηλών για να συμπληρωθούν τα δεξιότερα μπλοκ δειγμάτων. Εάν τα συστατικά εικόνας είναι interleaved, η διαδικασία κωδικοποίησης θα επεκτείνει επίσης τον αριθμό των δειγμάτων με ένα ή περισσότερα μπλοκ εάν χρειαστεί έτσι ώστε ο αριθμός των μπλοκ να είναι ακέραιο πολλαπλάσιο του H_i . Το ίδιο ισχύει και για την παράμετρο γ , επεκτείνοντας τώρα των γραμμών.

Για την διαδικασία χωρίς απώλειες δεδομένων, η μονάδα πληροφορίας είναι ένα δείγμα. Εάν τα συστατικά εικόνας είναι interleaved, η διαδικασία κωδικοποίησης θα επεκτείνει τον αριθμό των δειγμάτων εάν χρειαστεί έτσι ώστε ο αριθμός να είναι πολλαπλάσιο του H_i . Παρομοίως, η διαδικασία κωδικοποίησης θα επεκτείνει των

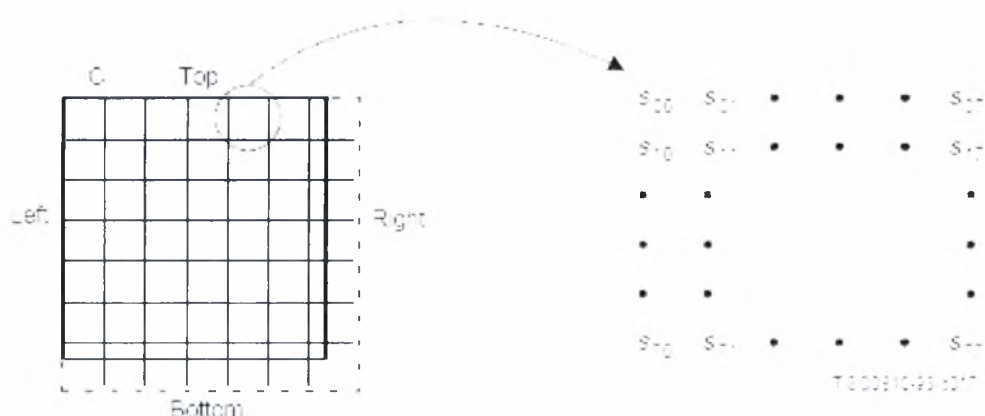
αριθμό των γραμμών εάν χρειαστεί έτσι ώστε ο αριθμός των γραμμών να είναι πολλαπλάσιο του V_1 .

Κάθε δείγμα που θα προστεθεί κατά την κωδικοποίηση για να συμπληρώσει το όχι πλήρες MCU θα αφαιρεθεί από την διαδικασία αποκωδικοποίησης.

Ο βασικός αλγόριθμος κωδικοποίησης ονομάζεται baseline αλγόριθμος και βασίζεται στον διακριτό μετασχηματισμό συνημίτονου. Στην συνέχεια περιγράφεται λεπτομερώς.

Πριν εφαρμοστεί ο ευθύς διακριτός μετασχηματισμός συνημίτονου (FDCT) σε ένα καρέ, τα δείγματα θα αφαιρεθούν κατά 2^{p-1} , όπου p είναι η παράμετρος της ακρίβειας. Έτσι, όταν το $P=8$, η αφαίρεση θα είναι κατά 128. Κατά την διαδικασία αποκωδικοποίησης, εφόσον εφαρμοστεί ο αντίστροφος διακριτός μετασχηματισμός συνημίτονου (IDCT), στα δείγματα θα προστεθεί η τιμή 128.

Η εικόνα 1.4 δείχνει ένα συστατικό εικόνας το οποίο έχει διαμεριστεί σε 8×8 μπλοκς για τον υπολογισμό του ευθύ μετασχηματισμού συνημίτονου. Η εικόνα 1.4 επίσης καθορίζει την διάταξη των δειγμάτων μέσα σε ένα μπλοκ υποδεικνύοντας τους δείκτες που χρησιμοποιούνται στους υπολογισμούς του μετασχηματισμού συνημίτονου.



Εικόνα 1.1: 2D DCT συνάρτηση

Γ. FDCT and IDCT

Οι παρακάτω ισότητες καθορίζουν τους υπολογισμούς του ευθύ και αντίστροφου μετασχηματισμού συνημίτονου.

$$\text{FDCT: } S_{xy} = \frac{1}{4} C_x C_y \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} s_{nm} \cos \frac{(2n+1)m\pi}{16} \cos \frac{(2y+1)x\pi}{16}$$

$$\text{IDCT: } s_{nm} = \frac{1}{4} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} C_x C_y S_{xy} \cos \frac{(2n+1)m\pi}{16} \cos \frac{(2y+1)x\pi}{16}$$

Όπου

$$C_x, C_y = 1/\sqrt{2} \text{ for } n, y = 0$$

$$C_x, C_y = 1 \text{ otherwise}$$

Δ. κβαντοποίηση και αποκβαντοποίηση DCT συντελεστών

Εφόσον ο ευθύς μετασχηματισμός συνημίτονου έχει υπολογιστεί για κάθε μπλοκ, κάθε ένας από τους 64 συντελεστές που παράγονται κβαντοποιείται. Το ποσό της κβαντοποίησης για κάθε συντελεστή S_{vu} είναι το στοιχείο Q_{vu} από τον πίνακα κβαντοποίησης και καθορίζεται από την παράμετρο T_{qi} . Ο κβαντοποιητής καθορίζεται από την ακόλουθη εξίσωση:

$$Sq_{vu} = \text{round} \left(\frac{S_{vu}}{Q_{vu}} \right)$$

Στρογγυλοποιημένος στον πλησιέστερο ακέραιο.

Στον αποκωδικοποιητή γίνεται η αντίστροφη διαδικασία

$$R_{vu} = Sq_{vu} \cdot Q_{vu}$$

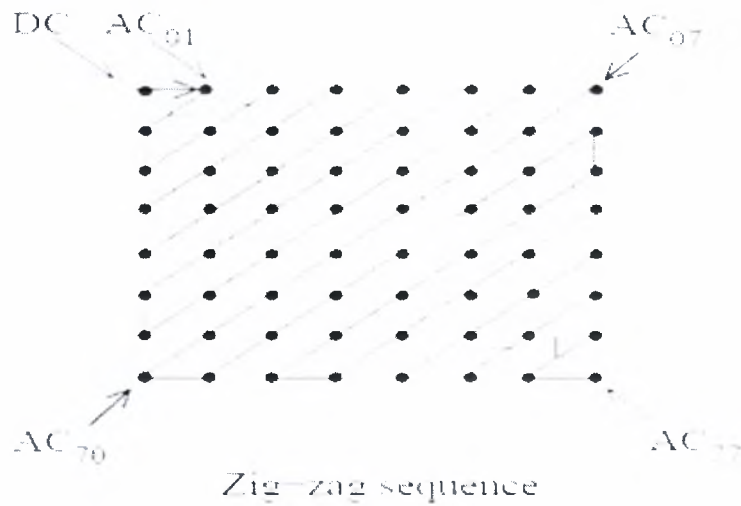
Ε. κωδικοποίηση DC συντελεστή

Μετά την κβαντοποίηση και κατά την κωδικοποίηση εντροπίας, ο κβαντοποιημένος DC συντελεστής Sq_{00} επιξεργάζεται ξεχωριστά από τους υπόλοιπους 63 κβαντοποιημένους AC συντελεστές. Η τιμή η οποία θα κωδικοποιηθεί είναι η διαφορά (DIFF) μεταξύ του κβαντοποιημένου συντελεστή του τρέχοντος μπλοκ (DC_i ή Sq_{00}) από αυτόν του προηγούμενου μπλοκ του ίδιου συστατικού εικόνας (PRED).

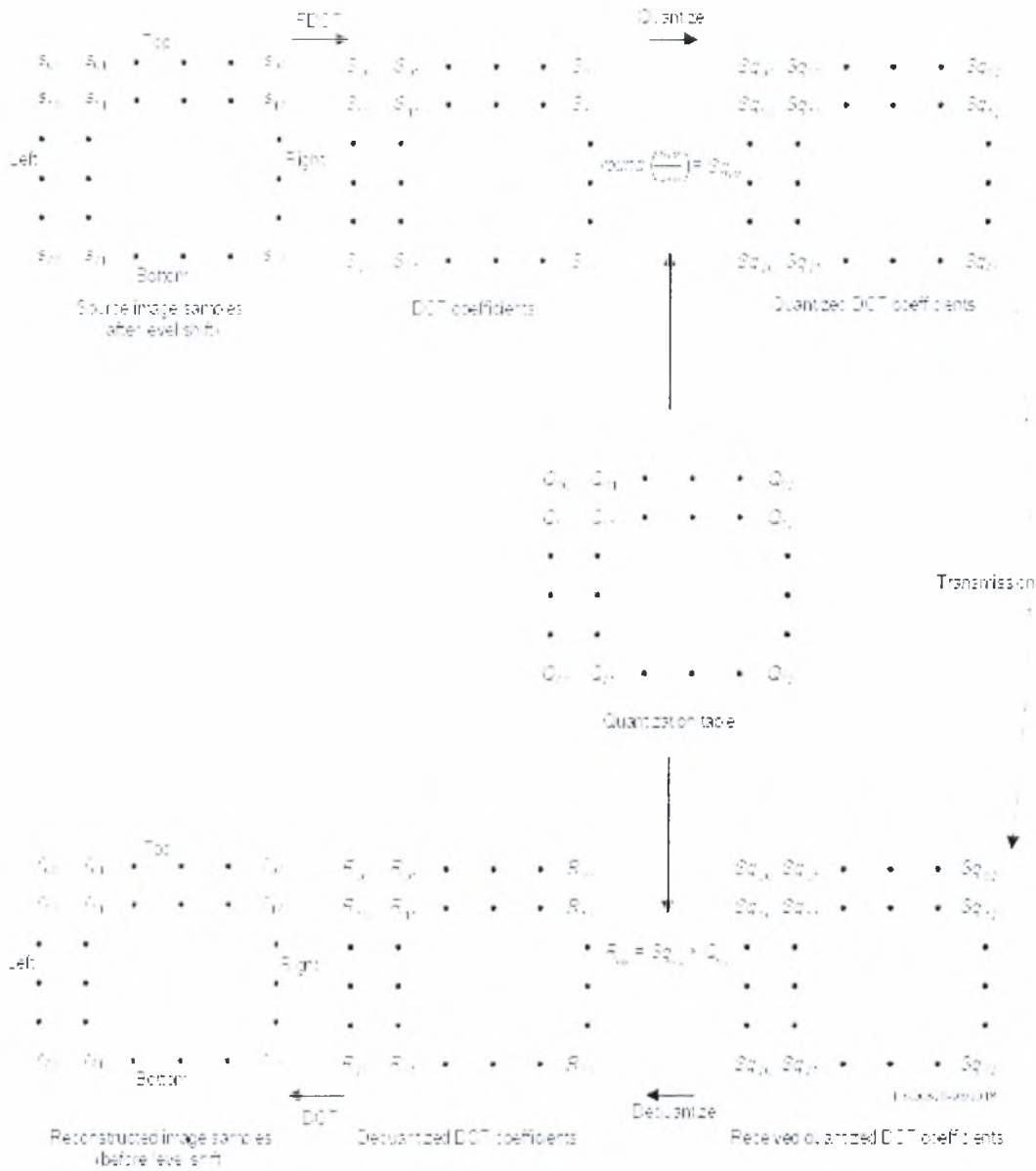
$$DIFF = DC_i - PRED$$

Z. Ακολουθία Zig-zag

Μετά την κβαντοποίηση και πριν την κωδικοποίηση εντροπίας, οι κβαντοποιημένοι AC συντελεστές μετατρέπονται σε μία zig-zag ακολουθία με την παρακάτω σειρά (πίνακας 1.5).



Στην [εικόνα 1.6](#) παρουσιάζεται η σχέση μεταξύ του μπλοκ 8x8 και των συντελεστών DCT.



Εικόνα 1.1: Αλγόριθμος κωδικοποίησης βασικής γραμμής.

Παρακάτω ακολουθεί ένα αριθμητικό παράδειγμα του baseline αλγορίθμου ([εικόνα 1.2](#)).

52	55	61	66	70	61	64	73	-76	-73	-67	-62	-58	-67	-64	-55	
63	59	66	90	109	85	69	72	-65	-69	-62	-38	-19	-43	-59	-56	
62	59	68	113	144	104	66	73	-66	-69	-60	-15	16	-24	-62	-55	
63	58	71	122	154	106	70	69	-128	-65	-70	-57	-6	26	-22	-58	-59
67	161	68	104	126	88	68	70	-61	-67	-60	-24	-2	-40	-60	-58	
79	65	60	70	77	68	58	75	-49	-63	-68	-58	-51	-65	-70	-53	
85	71	64	59	55	61	65	83	-43	-57	-64	-69	-73	-67	-63	-45	
87	79	69	68	65	76	78	94	-41	-49	-59	-60	-63	-52	-50	-34	

FDCT →

-415	-29	-62	25	55	-20	-1	3
7	-21	-62	9	11	-7	-6	6
-46	8	77	-25	-30	10	7	-5
-50	13	35	-15	-9	6	0	3
11	-8	-13	-2	-1	1	-4	1
-10	1	3	-3	-1	0	2	-1
-4	-1	2	-1	2	-3	1	-2
-1	-1	-1	-2	-1	-1	0	-1

Με κάποιον πίνακα κβαντοποίησης

-26	-3	-6	2	2	0	0	0
1	-2	-4	0	0	0	0	0
33	1	5	-1	-1	0	0	0
24	1	2	-1	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

→

26	-3	1	-3	2	-6	2	-4	1	4
1	4	1	1	5	0	2	0	0	1
2	0	0	0	0	0	-1	-1	EOB	

Κεφάλαιο 2^ο

Κωδικοποίηση χωρίς απώλειες δεδομένων – Lossless encoding

Περιεχόμενα

1. Εισαγωγή.....	34
2. Διαδικασίες <i>lossless</i> κωδικοποιητή.....	34
2.1 <i>Lossless</i> μοντέλο κωδικοποίησης.....	34
2.2 Πρόβλεψη.....	35
2.3 <i>Predictors</i>	36
2.4 Παράμετρος <i>point transform</i>	36
3. Διαδικασίες <i>lossless</i> κωδικοποιητή.....	37

1. Εισαγωγή

Το κεφάλαιο αυτό έχει ως στόχο να παρουσιάσει την μέθοδο κωδικοποίησης μιας εικόνας χωρίς απώλειες δεδομένων (lossless mode of operation). Η κωδικοποίηση μπορεί να πραγματοποιήσει είτε Huffman είτε αριθμητική κωδικοποίηση. Ο τρόπος αυτός κωδικοποίησης δεν στηρίζεται στον διακριτό μετασχηματισμό συνημίτονου αλλά στην πρόβλεψη η οποία πηγάζει από την σχέση ενός δείγματος με τα γειτονικά του.

Στην παρούσα εργασία, ο συγκεκριμένος τρόπος κωδικοποίησης αποτελεί το υπόβαθρο. Η κωδικοποίηση εντροπίας γίνεται με κωδικοποίηση Huffman.

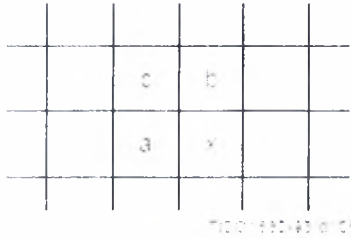
2. Διαδικασίες lossless κωδικοποιητή

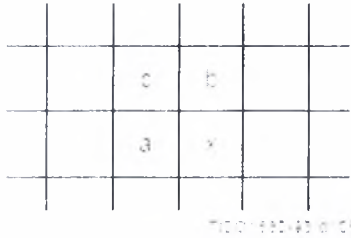
Σε αυτόν τον τρόπο κωδικοποίησης, η μονάδα πληροφορίας (data unit) είναι ένα δείγμα. Η ακρίβεια των δειγμάτων της εικόνας κυμαίνεται από 2 έως 16 bits. Τα δείγματα της εικόνας παρουσιάζονται ως μη προσημασμένοι ακέραιοι και δεν υφίστανται μετατόπιση επίπεδου πριν την κωδικοποίηση. Όταν ο κωδικοποιητής επανεκκινείται στην διαδικασία έλεγχου restart interval, η πρόβλεψη παίρνει την προκαθορισμένη της τιμή.

2.1 Lossless μοντέλο κωδικοποίησης

Το μοντέλο κωδικοποίησης που εξελίχθηκε για να κωδικοποιήσει τους DC συντελεστές στις διαδικασίες κωδικοποίησης βασισμένες σε DCT, επεκτείνεται για να επιτρέψει μια επιλογή από ένα σύνολο 7 μιας διάστασης και δυο διαστάσεων μηχανισμούς πρόβλεψης (predictor). Ο predictor γράφεται στην επικεφαλίδα σάρωσης. Ο ίδιος predictor χρησιμοποιείται για όλα τα συστατικά της σάρωσης. Κάθε συστατικό στη σάρωση μοντελοποιείται ξεχωριστά χρησιμοποιώντας προβλέψεις από τα γειτονικά δείγματα του ίδιου συστατικού.

2.2 Πρόβλεψη

Η  δείχνει την σχέση ανάμεσα στις θέσεις a,b,c των αναδημιουργημένων γειτονικών δειγμάτων που χρησιμοποιούνται για πρόβλεψη και της θέσης του δείγματος x που πρόκειται να κωδικοποιηθεί.



Εικόνα 2.1: Γειτονικά αναδημιουργημένα δείγματα

Ορίζουμε το P_x να είναι η πρόβλεψη της θέσης x και R_a, R_b, R_c να είναι τα αναδημιουργημένα δείγματα ακριβώς αριστερά, διαγώνια πάνω και πάνω από το τρέχον δείγμα. Οι επιτρεπόμενοι predictors, ένας εκ των οποίων επιλέγεται στην επικεφαλίδα σάρωσης παρουσιάζονται πιο κάτω ([ITU-T Rec. T81](#)).

Selection-value	Prediction
0	No prediction (see Annex J)
1	$P_x = R_a$
2	$P_x = R_b$
3	$P_x = R_c$
4	$P_x = R_a + R_b - R_c$
5	$P_x = R_a + (R_b - R_c) \cdot 2^{s-1}$
6	$P_x = R_b + (R_a - R_c) \cdot 2^{s-1}$
7	$P_x = (R_a + R_b) \cdot 2^{-s}$

⁴⁾ Shift right arithmetic operation

Εικόνα 2.2: Επιτρεπόμενοι predictors

2.3 Predictors

Οι επιλογές 1, 2 και 3 είναι για predictors μιας διάστασης και οι επιλογές 4,5,6 και 7 είναι δυο διαστάσεων. Ο οριζόντιος predictor χρησιμοποιείται στην πρώτη γραμμή των δειγμάτων στην αρχή της σάρωσης και στην αρχή κάθε restart interval. Ο επιλεγμένος predictor χρησιμοποιείται για όλες τις υπόλοιπες γραμμές. Το δείγμα που βρίσκεται ακριβώς από πάνω από το τρέχον (Rβ) χρησιμοποιείται στην αρχή κάθε γραμμής εκτός από την πρώτη. Στην αρχή κάθε γραμμής και στην αρχή κάθε restart interval, η τιμή πρόβλεψης 2^{P-1} χρησιμοποιείται, όπου P είναι η ακρίβεια του δείγματος. Εάν η παράμετρος point transform δεν είναι μηδέν, η τιμή πρόβλεψης στην αρχή των πρώτων γραμμών και στην αρχή κάθε restart interval είναι 2^{P-Pt-1} .

Για απλότητα της υλοποίησης, η διαίρεση με το 2 στους predictors 5 και 6 γίνεται με αριθμητική δεξιά ολίσθηση κατά ένα bit.

Η διαφορά μεταξύ της τιμής πρόβλεψης και της τιμής του δείγματος υπολογίζεται με υπόλοιπο 16. Στον αποκωδικοποιητή, η διαφορά αποκωδικοποιείται και προστίθεται το υπόλοιπο 16 στην πρόβλεψη.

2.4 Παράμετρος point transform

Στην κωδικοποίηση χωρίς απώλειες δεδομένων υπάρχει περίπτωση τα δείγματα της εικόνας να διαιρεθούν με μια ποσότητα 2^{Pt} όπου Pt η παράμετρος point transform. Στην lossless κωδικοποίηση επιλέγουμε να γίνει η διαίρεση αυτή με δεξιά ολίσθηση και όχι με αριθμητική πράξη. Ο λόγος είναι ότι με την ολίσθηση επηρεάζονται μόνο τα LSB (λιγότερα σημαντικά ψηφία) ενώ με αριθμητική πράξη επηρεάζεται όλη η ποσότητα.

SSSS	Difference values
0	0
1	-1 1
2	-3 -2 2 3
3	-7 -4 4 7
4	-15 -8 8 15
5	-31 -16 16 31
6	-63 -32 32 63
7	-127 -64 64 127
8	-255 -128 128 255
9	-511 -256 256 511
10	-1 023 -512 512 1 023
11	-2 047 -1 024 1 024 2 047
12	-4 095 -2 048 2 048 4 095
13	-8 191 -4 096 4 096 8 191
14	-16 383 -8 192 8 192 16 383
15	-32 767 -16 384 16 384 32 767
16	32 768

Πίνακας 1: Τύποι κωδικοποίησης

3. Διαδικασίες lossless κωδικοποιητή

Οι lossless αποκωδικοποιητές πρέπει να είναι ικανοί να αποκωδικοποιήσουν κωδικοποιημένα δεδομένα εικόνας με ακρίβεια από 2 έως 16 bits. Η μονάδα πληροφορίας είναι ένα δείγμα. Όταν ο αποκωδικοποιητής επανεκκινείται εξαιτίας της διαδικασίας ελέγχου restart interval η πρόβλεψη παίρνει την ίδια τιμή με αυτή που χρησιμοποίησε ο κωδικοποιητής. Τέλος, αποκωδικοποιούν τη διαφορά και προσθέτουν σε αυτή το υπόλοιπο 16 για να δημιουργηθεί το δείγμα εξόδου. Εάν είναι ενεργοποιημένη η παράμετρος Pt, τα δείγματα πολλαπλασιάζονται με ένα παράγοντα 2^{Pt} .

Κεφάλαιο 3^ο

Σύνταξη συμπιεσμένων δεδομένων

Περιεχόμενα

1. Εισαγωγή.....	39
2. Βασικά μέρη δομής συμπιεσμένων δεδομένων.....	39
2.1 Παράμετροι.....	39
2.2 Markers.....	39
2.3 Marker Κομμάτια.....	41
3. Σύνταξη Lossless κωδικοποίησης.....	41
3.1 Σύνταξη επικεφαλίδας καρέ.....	43
3.2 Σύνταξη επικεφαλίδας σάρωσης.....	45
4. Σύνταξη καθορισμού πινάκων και marker κομματιών.....	46
4.1 Σύνταξη καθορισμού πινάκων Huffman.....	47
4.2 Restart interval definition syntax.....	49

1. Εισαγωγή

Η δομή των συμπιεσμένων δεδομένων αποτελείται από μια ακολουθία από κώδικες που διατάσσονται σε bytes.

2. Βασικά μέρη δομής συμπιεσμένων δεδομένων

Τα συμπιεσμένα δεδομένα αποτελούνται κυρίως από παραμέτρους, markers και κομμάτια κωδικοποιημένων δεδομένων. Οι παράμετροι και οι markers συχνά οργανώνονται σε marker κομμάτια.

2.1 Παράμετροι

Οι παράμετροι είναι μη προσημασμένοι ακέραιοι με συγκεκριμένο μέγεθος και συγκεκριμένη τιμή η οποία καθορίζεται από την διαδικασία κωδικοποίησης και από τα χαρακτηριστικά της εικόνας. Το μέγεθος τους είναι συνήθως 4-bit, 1-byte είτε 2-bytes. Η ύπαρξη τους είναι πάρα πολύ σημαντική. Χωρίς αυτές, θα ήταν αδύνατον να ανακατασκευαστεί η εικόνα από τον αποκωδικοποιητή.

Για τις παραμέτρους με μέγεθος 2 bytes, το περισσότερο σημαντικό byte γράφεται πρώτα και ύστερα το λιγότερο σημαντικό. Οι παράμετροι που έχουν μήκος 4 bits, συνήθως έρχονται σε ζευγάρια και κωδικοποιούνται ως ένα απλό byte με την πρώτη 4-bit παράμετρο να καταλαμβάνει τα τέσσερα περισσότερο σημαντικά bit. Το ίδιο συμβαίνει και για τα εναπομείναντα μεγέθη παραμέτρων, τα περισσότερα σημαντικά bits γράφονται πρώτα.

2.2 Markers

Οι markers χρησιμοποιούνται στο να αναγνωρίζονται τα διάφορα δομικά μέρη των συμπιεσμένων δεδομένων. Πολλοί markers ακολουθούνται από μια λίστα από παραμέτρους όπως περιγράψαμε ενώ άλλοι όχι. Όλοι τους είναι δυο bytes. Το πρώτο byte είναι το X'FF το οποίο ακολουθείται από ένα άλλο byte το οποίο δεν πρέπει να είναι το 0 ή το X'FF.

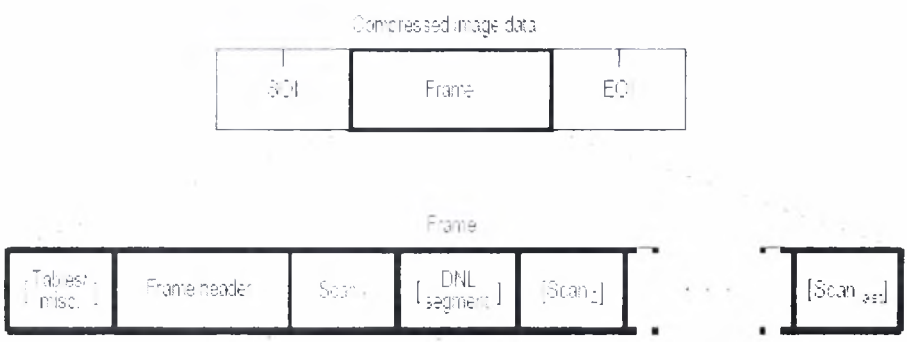
Ακολουθεί ο πίνακας 3.1 ο οποίος παρουσιάζει όλους τους πιθανούς markers. Με αστερίσκο είναι όσοι στέκονται μόνοι τους, τα οποία δεν είναι η αρχή ενός marker κομματιού.

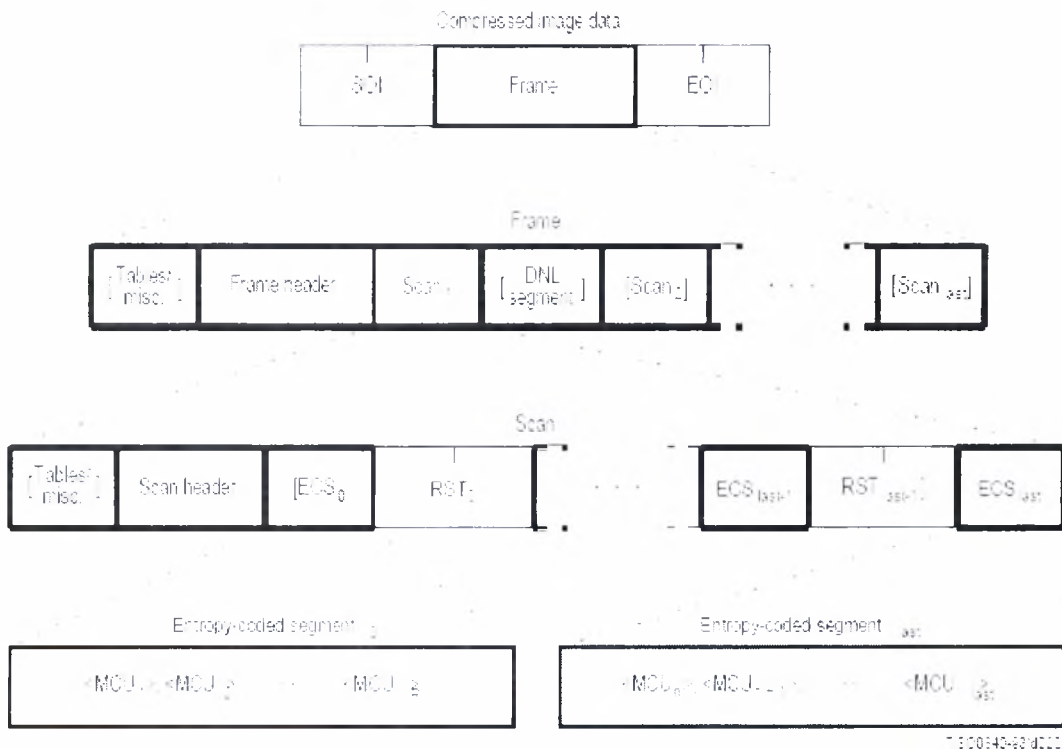
Code Assignment	Symbol	Description
Start Of Frame markers, non-differential Huffman coding		
*FFC0 *FFC1 *FFC2 *FFC3	SOF ₀ SOF ₁ SOF ₂ SOF ₃	Baseline DCT Extended sequential DCT Progressive DCT Lossless (sequential)
Start Of Frame markers, differential Huffman coding		
*FFC4 *FFC5 *FFC6	SOF ₄ SOF ₅ SOF ₆	Differential sequential DCT Differential progressive DCT Differential lossless (sequential)
Start Of Frame markers, non-differential arithmetic coding		
*FFC8 *FFC9 *FFCA *FFCB	JPC SOF ₈ SOF ₉ SOF ₁₁	Reserved for JPEC extensions Extended sequential DCT Progressive DCT Lossless (sequential)
Start Of Frame markers, differential arithmetic coding		
*FFCD *FFCE *FFCF	SOF ₁₃ SOF ₁₄ SOF ₁₅	Differential sequential DCT Differential progressive DCT Differential lossless (sequential)
Huffman table specification		
*FFC4'	DHT	Define Huffman table(s)
Arithmetic coding conditioning specification		
*FFC0'	DAC	Define arithmetic coding conditioning(s)
Restart interval termination		
*FFD0 through *FFD7	RST _n *	Restart with modulo 8 counting
Other markers		
*FFD8 *FFD9 *FFDA *FFDB *FFDC *FFDD *FFDE *FFDF *FFE0 through *FFEF *FFF0 through *FFFD' *FFFE	SOI EOI SOS DQT DNL DRG DHP EXP APP _n JPC COM	Start of image End of image Start of scan Define quantization table(s) Define number of lines Define restart interval Define hierarchical progression Expand reference components Reserved for application segments Reserved for JPEC extensions Comment
Reserved markers		
*FF00 *FF01 through *FFBF	TEM _n RES	For temporary private use in arithmetic coding Reserved

2.3 Marker Κομμάτια

Αποτελούνται από marker ο οποίος ακολουθείται από σχετικές παραμέτρους. Η πρώτη παράμετρος είναι η παράμετρος που καθορίζει το μήκος του συγκεκριμένου κομματιού και είναι 2 bytes. Αυτή η παράμετρος είναι υπεύθυνη για την κωδικοποίηση του αριθμού των bytes στο κάθε κομμάτι εκτός βέβαια από τον 2-byte marker. Τα κομμάτια marker που αναγνωρίζονται από τους markers SOF και SOS αναφέρονται σε κεφαλίδες, στην επικεφαλίδα καρέ και σάρωσης αντίστοιχα.

3. Σύνταξη Lossless κωδικοποίησης

Παρακάτω περιγράφεται η σύνταξη των συμπιεσμένων δεδομένων για lossless κωδικοποίηση. Η  δείχνει την υψηλού επιπέδου σύνταξη για όλα τα μέρη της συμπιεσμένης μορφής.



7300340-001011

Διακρίνονται τρεις markers:

SOI: Σηματοδοτεί την αρχή της πληροφορίας για την συμπιεσμένη εικόνα.

EOI: Σηματοδοτεί το τέλος της πληροφορίας για την συμπιεσμένη εικόνα.

RST_m: marker επανεκκίνησης. Ένας υπό συνθήκη marker ο οποίος τοποθετείται ανάμεσα από τα marker κομμάτια που προέρχονται από κωδικοποίηση εντροπίας μόνο εάν η παράμετρος επανεκκίνησης είναι ενεργοποιημένη. Υπάρχουν οχτώ μοναδικοί markers επανεκκίνησης (m=0-7) οι οποίοι επαναλαμβάνονται σε ακολουθία από 0 έως 7 αρχίζοντας από το 0 για κάθε σάρωση.

Αξίζει να σημειώσουμε ότι υπάρχει μόνο ένα καρέ το οποίο όταν συμπιεστεί πρέπει να ακολουθηθεί από το τον marker EOI.

Το δεύτερο επίπεδο καθορίζει ότι το καρέ πρέπει να αρχίζει με μια κεφαλίδα καρέ και θα περιέχει μια ή περισσότερες σαρώσεις. Διάφοροι ορισμοί πινάκων και marker κομμάτια μπορεί να προηγούνται της επικεφαλίδας καρέ. Εάν το κομμάτι DNL είναι παρών, πρέπει να ακολουθεί την πρώτη σάρωση. Κάθε σάρωση θα περιέχει πληροφορία από ένα έως τέσσερα συστατικά εικόνας.

Το τρίτο επίπεδο καθορίζει ότι μια σάρωση θα ξεκινάει με μια επικεφαλίδα σάρωσης και θα περιέχει ένα ή περισσότερα κομμάτια που έχουν υποστεί κωδικοποίηση εντροπίας. Διάφοροι ορισμοί πινάκων και marker κομμάτια μπορεί να προηγούνται της επικεφαλίδας σάρωσης. Εάν η παράμετρος επανεκκίνησης δεν είναι ενεργοποιημένη, θα υπάρχει μόνο ένα κομμάτι που έχει υποστεί κωδικοποίηση εντροπίας και δεν θα εμφανίζονται markers επανεκκίνησης. Εάν η παράμετρος επανεκκίνησης είναι ενεργοποιημένη, ο αριθμός των κομματιών που έχουν υποστεί κωδικοποίηση εντροπίας καθορίζεται από το μέγεθος της εικόνας και το καθορισμένο restart interval.

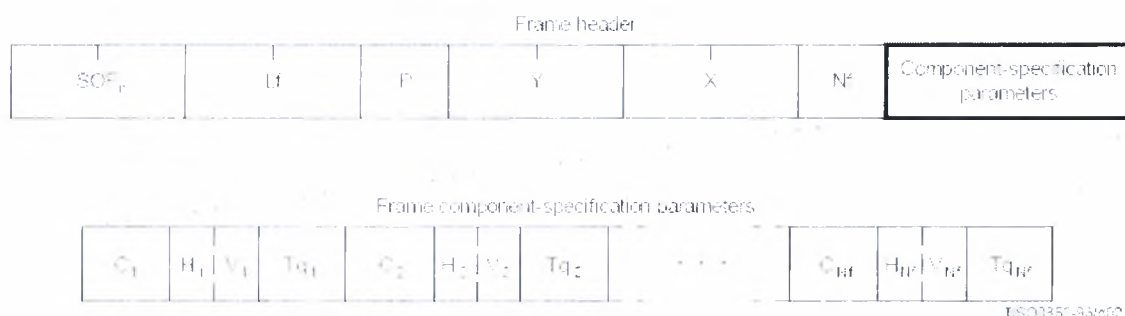
Το τέταρτο επίπεδο καθορίζει ότι το κωδικοποιημένο κομμάτι αποτελείται από μια ακολουθία από κωδικοποιημένα MCUs. Εάν η παράμετρος επανεκκίνησης είναι ενεργοποιημένη και έχει την τιμή R_i , κάθε κωδικοποιημένο κομμάτι εκτός από το

πρώτο θα περιέχει R_i MCUs. Το τελευταίο θα περιέχει όσα MCUs συμπληρώνουν την σάρωση.

Τέλος παρατηρούμε ότι ο καθορισμός των πινάκων που είναι απαραίτητοι για την αποκωδικοποίηση γράφονται σε συγκεκριμένες τοποθεσίες της κωδικοποιημένης πληροφορίας.

3.1 Σύνταξη επικεφαλίδας καρέ

Η h καθορίζει την επικεφαλίδα καρέ η οποία θα παρουσιάζεται στην αρχή του καρέ. Η επικεφαλίδα καθορίζει τα χαρακτηριστικά της εικόνας, τα συστατικά εικόνας στο καρέ και τους παράγοντες δειγματοληψίας για κάθε συστατικό εικόνας και καθορίζει τον προορισμό από όπου οι κβαντοποιημένοι πίνακες θα χρησιμοποιηθούν κατά την αποκωδικοποίηση.



Οι markers και οι παράμετροι που χρησιμοποιούνται επεξηγούνται στον πίνακα 3.2. Το μέγεθος και οι επιτρεπόμενες τιμές παρουσιάζονται στον πίνακα 3.3.

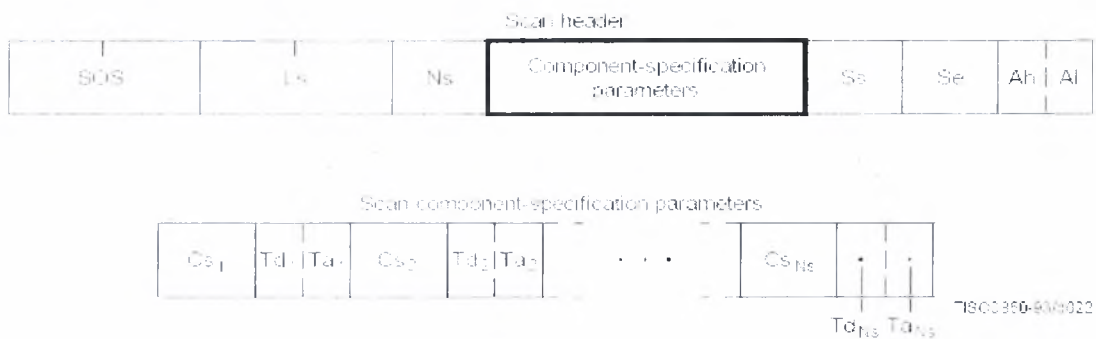
SOF3: marker για Lossless τρόπο κωδικοποίηση, κωδικοποίηση Huffman
Lf: μέγεθος επικεφαλίδας καρτέ
P: ακρίβεια δείγματος σε bits
Y: αριθμός γραμμών
X: αριθμός δειγμάτων ανά γραμμή
Nf: αριθμός συστατικών εικόνας στο καρτέ
Ci: αναγνωριστής συστατικού εικόνας
Hi: οριζόντιος παράγοντας δειγματοληψίας
Vi: κάθετος παράγοντας δειγματοληψίας
Tqi: δείκτης πίνακα κβαντοποίησης

Πίνακας 2.1: Πίνακας τιμών

Parameter	Size (bits)	Values			
		Sequential DCT		Progressive DCT	Lossless
		Baseline	Extended		
Lf	16	$3 + 3 \cdot Nf$			
P	8	8	8-12	8-12	2-16
Y	16	0-65 535			
X	16	1-65 535			
Nf	8	1-255	1-255	1-4	1-255
C _i	8	0-255			
H _i	4	1-4			
V _i	4	1-4			
T _{qi}	8	0-3	0-3	0-3	0

3.2 Σύνταξη επικεφαλίδας σάρωσης

Η h καθορίζει την επικεφαλίδα σάρωσης η οποία θα παρουσιάζεται στην αρχή της σάρωσης. Αυτή η επικεφαλίδα καθορίζει ποια συστατικά καθορίζονται στη σάρωση και καθορίζει από πού οι πίνακες εντροπίας θα χρησιμοποιηθούν κατά την αποκωδικοποίηση. Για την διαδικασία αποκωδικοποίησης χωρίς απώλειες, οι παράμετροι σάρωσης καθορίζουν την πρόβλεψη και το point transform.



Το μέγεθος και οι επιτρεπόμενες τιμές των παραμέτρων παρουσιάζονται στον [Πίνακα 3.1](#).

Parameter	Size (bits)	Values			
		Sequential DCT		Progressive DCT	Lossless
		Baseline	Extended		
Ls	16	0 + 2 * Ns			
Ns	6	1-4			
Cs _i	6	0-255 ^(a)			
Td _i	4	0-1	0-3	0-3	0-3
Ta _i	4	0-1	0-3	0-3	0
Ss	8	0	0	0-63	1-7b)
Se	6	63	63	Ss-63 ^(c)	0
Ah	4	0	0	0-15	0
Al	4	0	0	0-15	0-15

^(a) Cs_i shall be a member of the set of C_i specified in the frame header.
^(b) 0 for lossless differential frames in the hierarchical mode (see B.3).
^(c) 0 if Ss equals zero.

Οι markers και οι παράμετροι που χρησιμοποιούνται επεξηγούνται στον

SOS : αρχή marker σάρωσης
Ls: μήκος επικεφαλίδας σάρωσης
Ns: αριθμός συστατικών εικόνας στη σάρωση
Cs _j : Καθορίζει ποίο από τα Nf συστατικά εικόνας που καθορίστηκαν στη επικεφαλίδα σάρωσης θα είναι το j-οστό συστατικό στη σάρωση. Κάθε Cs _j θα πρέπει να ταιριάζει με ένα από τα Ci που καθορίστηκαν στην επικεφαλίδα σάρωσης με τη σωστή σειρά.
Td _j : δείκτης για πίνακες DC κωδικοποίησης εντροπίας. Καθορίζει έναν από τους τέσσερις πίνακες κωδικοποίησης εντροπίας.
Ta _j : δείκτης για πίνακες AC κωδικοποίησης εντροπίας. Καθορίζει έναν από τους τέσσερις πίνακες κωδικοποίησης εντροπίας.
Ss: επιλογή πρόβλεψης
Se: στην lossless κωδικοποίηση είναι μηδέν.
Ah: στην lossless κωδικοποίηση είναι μηδέν.
Al: point transform

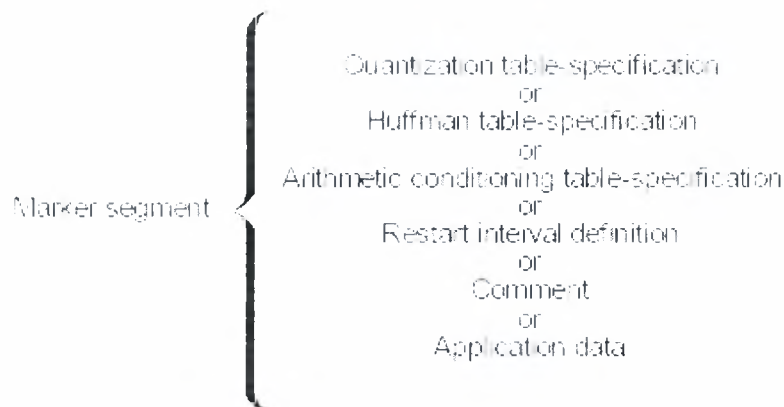
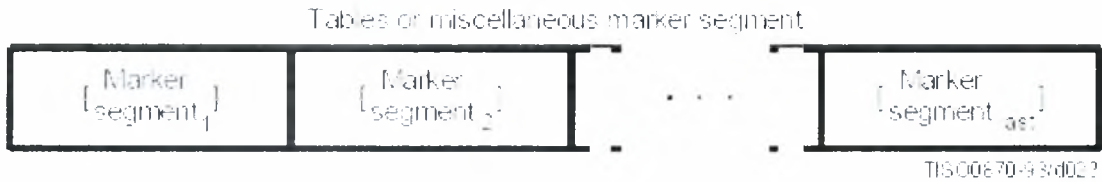
Πίνακας 2.3: Τύποι των marker

4. Σύνταξη καθορισμού πινάκων και marker κομματιών

Στην παράγραφο αυτή περιγράφονται ποια marker κομμάτια υπάρχουν () και πώς αυτά συντάζονται στην συμπιεσμένη μορφή. Τα κομμάτια αυτά είναι

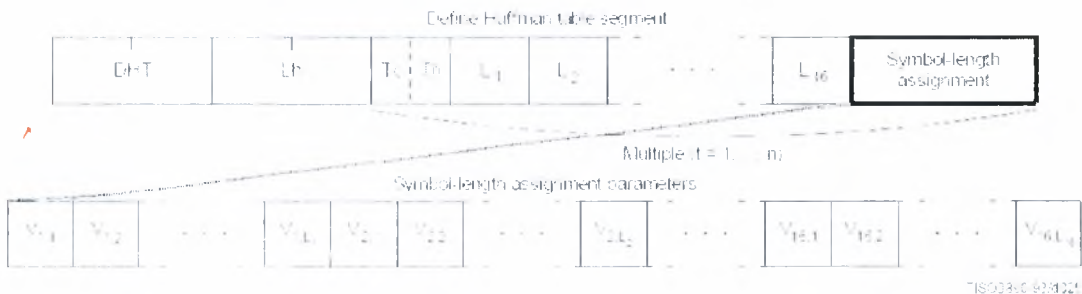
1. Καθορισμός πινάκων κβαντοποίησης
2. Καθορισμός πινάκων Huffman
3. Καθορισμός πινάκων αριθμητικής κωδικοποίησης
4. Καθορισμός restart interval
5. σχόλια
6. πληροφορίες εφαρμογής

Από αυτά, θα εξετάσουμε τον καθορισμό πινάκων Huffman και τον καθορισμό του restart interval.



4.1 Σύνταξη καθορισμού πινάκων Huffman

Η **00** καθορίζει το κομμάτι εκείνο της συμπιεσμένης εικόνας το οποίο εμπεριέχει τους πίνακες Huffman.



Οι markers και οι παράμετροι που χρησιμοποιούνται επεξηγούνται στον [Πίνακα 3.5](#).

DHT: marker που καθορίζει το Huffman Table
Lh: μήκος όλων των παραμέτρων του πίνακα Huffman
Tc: κλάση πίνακα .Μηδέν για DC ,1 για AC
Th: δείκτης για κάθε ένα πίνακα κωδικοποίησης.
L _i : αριθμός κωδίκων Huffman μήκους i (1 ≤ i ≤ 16). Στοιχεία του πίνακα BITS
V _{ij} : τιμές οι οποίες σχετίζονται με κάθε κώδικα Huffman. Στοιχεία του πίνακα HUFFVAL

Πίνακας 3.5: Lossless JPEG marker

Το μέγεθος και οι επιτρεπόμενες τιμές των παραμέτρων παρουσιάζονται στον [Πίνακα 3.6](#).

3.6

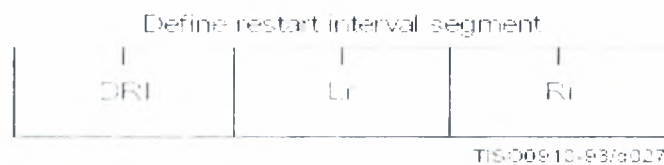
Parameter	Size (bits)	Values			
		Sequential DCT		Progressive DCT	Lossless
		Baseline	Extended		
Lh	16	$2 + \sum_{i=1}^n (17 + m_i)$			
Tc	4	0-1		0	
Th	4	0-1	0-5		
L _i	8	0-255			
V _{ij}	8	0-255			

- Το n καθορίζει τον αριθμό των πινάκων Huffman που καθορίζονται στο DHT .
- Η τιμή m_i καθορίζει τον αριθμό των παραμέτρων οι οποίες ακολουθούν τους 16 L_i παραμέτρους για τον πίνακα t και δίνεται από τη παρακάτω σχέση και είναι διαφορετικό για κάθε πίνακα.

$$m_i = \sum_{j=1}^{15} L_j$$

4.2 Σύνταξη Restart interval

Η m_i καθορίζει το κομμάτι εκείνο της συμπιεσμένης εικόνας το οποίο καθορίζει το restart interval.



Οι markers και οι παράμετροι που χρησιμοποιούνται επεξηγούνται στον [πίνακα 2.4](#).

DRI: καθορίζει τον marker του restart interval
Lr: καθορίζει το μήκος
Ri: καθορίζει τον αριθμό των MCU

Η τιμή του MCUR είναι ο αριθμός των MCUs τα οποία απαιτούνται για make up μιας γραμμής κάθε συστατικού στη σάρωση.

Το μέγεθος και οι επιτρεπόμενες τιμές των παραμέτρων παρουσιάζονται στον

Πίνακας 5.2

Parameter	Size (bits)	Values			
		Sequential DCT		Progressive DCT	Lossless
		Baseline	Extended		
Lr	16	4			
Ra	16	0-65 535			r - MCUR

Πίνακας 5.3

Κεφάλαιο 4^ο

Διαδικασίες lossless κωδικοποιητή & αποκωδικοποιητή.

Περιεχόμενα

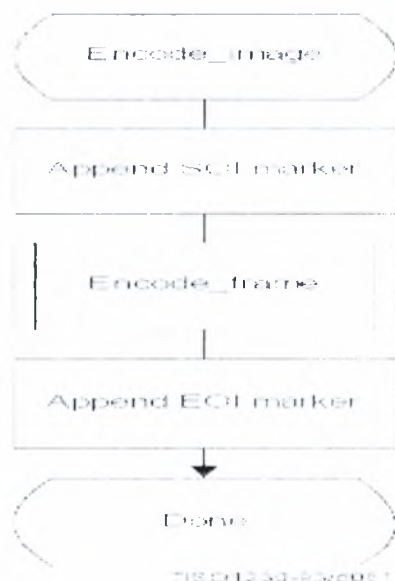
1. Εισαγωγή.....	52
2. Κωδικοποίηση εικόνας.....	52
2.1 Κωδικοποίηση καρέ.....	53
2.2 Κωδικοποίηση σάρωσης.....	54
2.3 Κωδικοποίηση restart interval.....	56
2.4 Κωδικοποίηση MCU.....	57
3. Αποκωδικοποίηση εικόνας.....	58
3.1 Αποκωδικοποίηση καρέ.....	60
3.2 Αποκωδικοποίηση σάρωσης.....	61
3.3 Αποκωδικοποίηση restart interval.....	62
3.4 Αποκωδικοποίηση MCU.....	63

1. *Εισαγωγή*

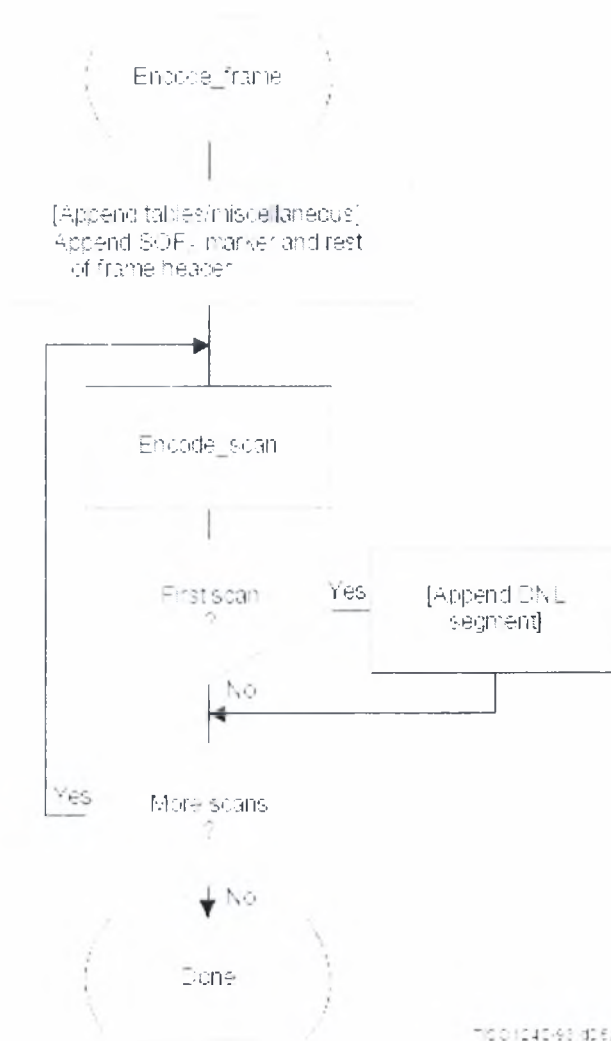
Σε αυτό το κεφάλαιο θα περιγράψουμε τις διαδικασίες με τις οποίες γίνεται η κωδικοποίηση μιας εικόνας. Στην ουσία, η κωδικοποίηση μιας εικόνας είναι μια ιεραρχική διαδικασία η οποία απαιτεί τις ακόλουθες υπορουτίνες: Αρχικά απαιτείται η κωδικοποίηση ενός καρέ. Η κωδικοποίηση ενός καρέ περιλαμβάνει την κωδικοποίηση μιας σάρωσης και αυτή με την σειρά της την κωδικοποίηση ενός restart interval εάν υπάρχει. Τέλος, φθάνοντας στο κατώτερο επίπεδο κωδικοποιείται ένα MCU, το οποίο περιλαμβάνει μια ακολουθία από δείγματα.

2. *Κωδικοποίηση εικόνας*

Η κωδικοποίηση εικόνας (`Encode_image`) περιλαμβάνει αρχικά την προσάρτηση του κατάλληλου marker (SOI) καθώς επίσης και την κωδικοποίηση κάθε καρέ (ένα καρέ υπάρχει στην lossless κωδικοποίηση) η οποία θα παρουσιαστεί πιο κάτω. Τέλος, εφόσον ολοκληρωθεί η κωδικοποίηση του καρέ, γίνεται προσάρτηση του marker ο οποίος υποδηλώνει το τέλος κωδικοποίησης της εικόνας (EOI) .



Η διαδικασία για την κωδικοποίηση ενός καρέ () προσανατολίζεται γύρω από τις σαρώσεις στο καρέ. Στην αρχή προσαρτείται η επικεφαλίδα του καρέ και στη συνέχεια κωδικοποιούνται οι σαρώσεις. Καθορισμοί πινάκων και άλλα marker κομμάτια προηγούνται των SOF_n markers (από τα αρχικά Start of Frame). Η διαδικασία τερματίζει όταν ολοκληρωθούν οι σαρώσεις.



2.2 Κωδικοποίηση σάρωσης

Μια σάρωση αποτελείται από μια απλή διαπέραση στα δεδομένα κάθε συστατικού εικόνας στη σάρωση. Προσδιορισμοί πινάκων και άλλα marker κομμάτια προηγούνται του SOS marker(00 00 00 00). Εάν περισσότερα από ένα συστατικά κωδικοποιούνται στη σάρωση, τα δεδομένα είναι interleaved. Εάν η παράμετρος που καθορίζει το restart interval είναι ενεργοποιημένη, τα δεδομένα τμηματοποιούνται μέσα στα restart intervals. Επίσης, ο RST_m τοποθετούνται στα κωδικοποιημένα δεδομένα ανάμεσα από τα restart intervals. Εάν η παράμετρος που καθορίζει το restart interval είναι απενεργοποιημένη, η διαδικασία ελέγχου είναι ίδια εκτός από το ότι ολόκληρη η σάρωση περιέχει μοναδικό restart interval. Τα συμπιεσμένα δεδομένα εικόνας που δημιουργούνται από μια σάρωση πάντα ακολουθούνται από ένα marker, είτε τον EOI είτε τον marker από το επόμενο marker τμήμα.

Η [εικόνα 2.7](#) δείχνει την διαδικασία ελέγχου για την κωδικοποίηση μιας σάρωσης. Ο βρόχος τερματίζεται όταν η διαδικασία κωδικοποίησης έχει κωδικοποιήσει τον αριθμό από τα restart intervals. Ο δείκτης m είναι ο μετρητής υπολοίπου του restart interval που απαιτείται για τον RST_m marker.

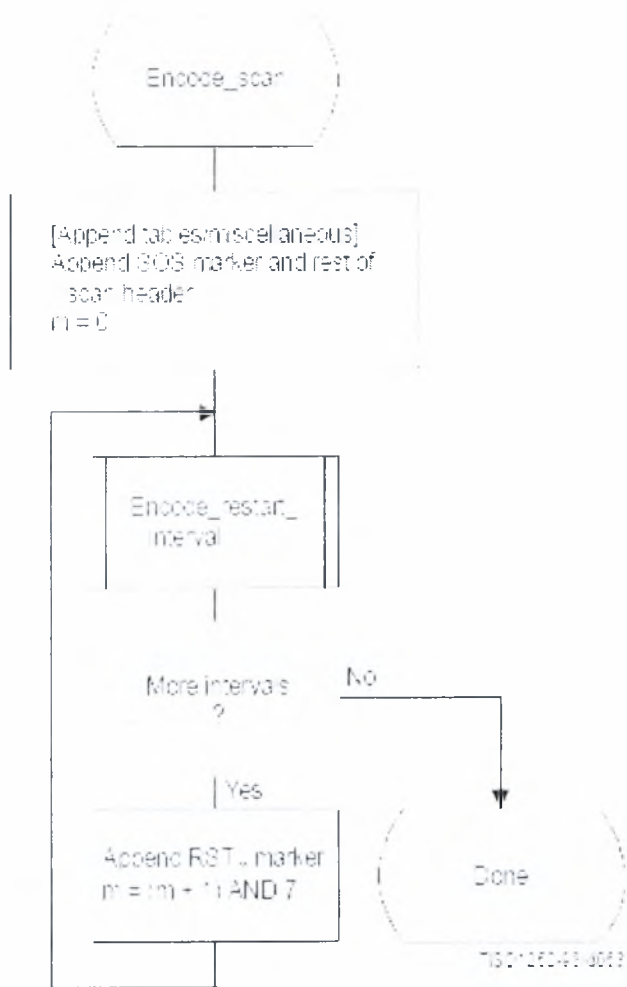
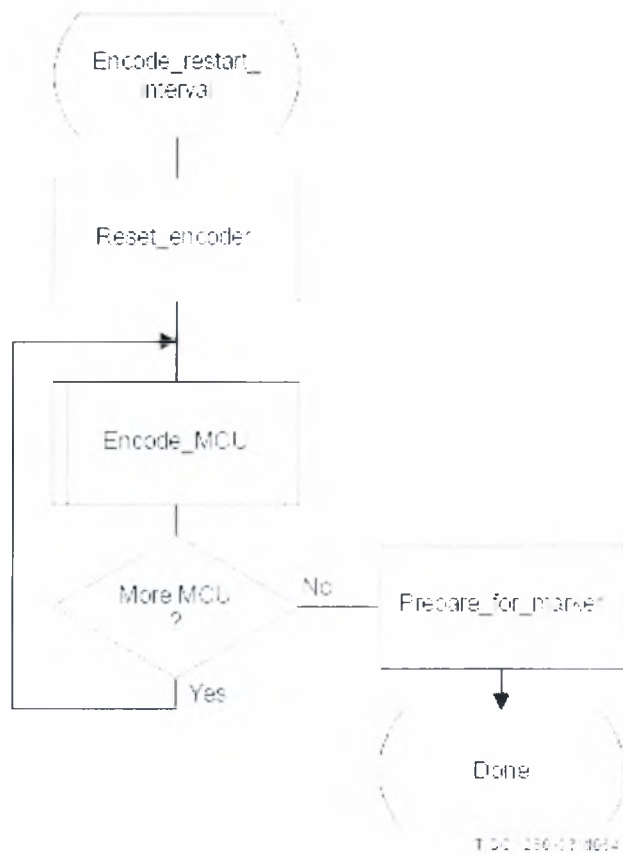


Figure 10.13

2.3 Κωδικοποίηση restart interval

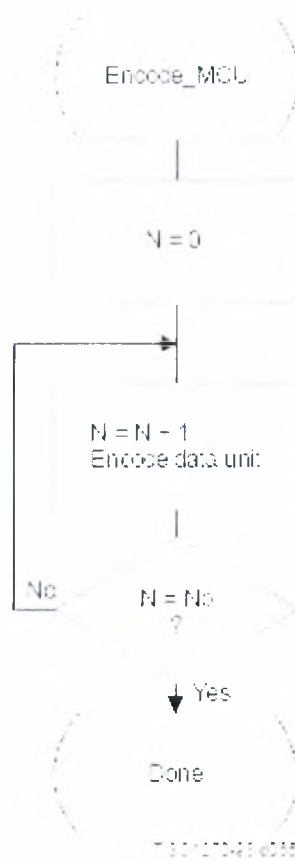
Στην εικόνα 4.4 παρουσιάζεται η διαδικασία ελέγχου για την κωδικοποίηση του restart interval. Ο βρόχος τερματίζεται είτε όταν η διαδικασία κωδικοποίησης έχει κωδικοποιήσει τον αριθμό των MCU είτε όταν έχει τελειώσει η σάρωση .



- Η διαδικασία ελέγχου “Reset_encoder” στον lossless τρόπο εκτέλεσης, επαναφέρει την πρόβλεψη στην προκαθορισμένη τιμή για όλα τα συστατικά στη σάρωση .
- Η διαδικασία “Prepare_for_marker” προσθέτει bits εάν χρειαστούν για να ολοκληρωθεί το τελευταίο byte.

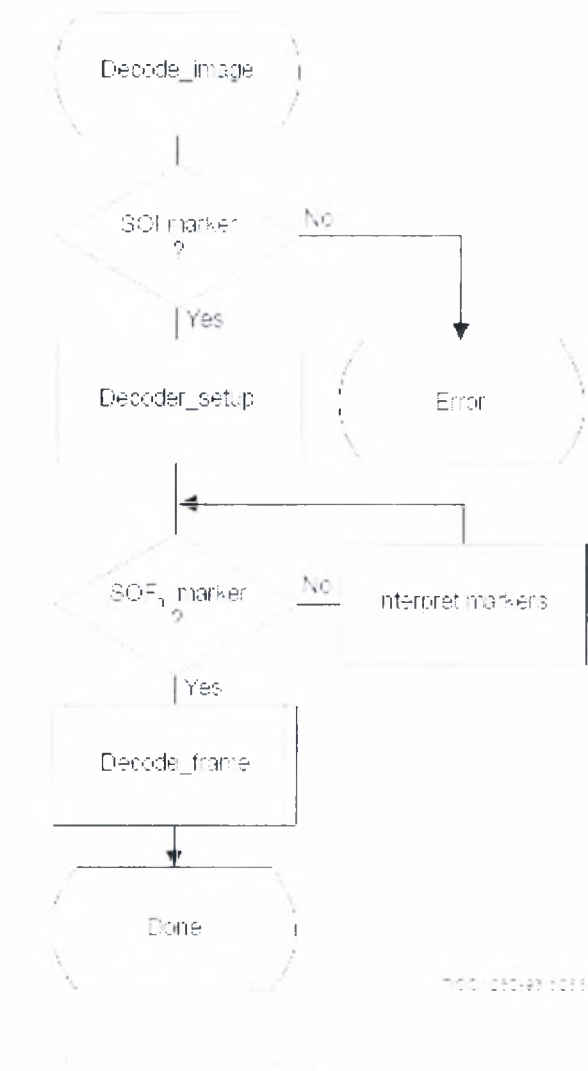
Μέσα στο δοθέν MCU, οι μονάδες πληροφορίας κωδικοποιούνται με την σειρά στην οποία βρίσκονται μέσα στο MCU (). Η διαδικασία ελέγχου κωδικοποίησης παρουσιάζεται στο παρακάτω σχήμα.

Το N_b αναφέρεται στον αριθμό των μονάδων πληροφορίας που βρίσκονται μέσα στο MCU.



3. Αποκωδικοποίηση εικόνας

Η διαδικασία αποκωδικοποίησης βασίζεται στην αναγνώριση των διαφόρων markers. Ο πρώτος marker πρέπει να είναι ο SOI. Η διεργασία “Decoder_setup” επανεκκινεί το restart interval . Ο επόμενος marker είναι συνήθως ο SOF_n . Εάν αυτός δεν βρεθεί τότε ένα από τα marker τμήματα που αναφέρονται στον [Πίνακα 2.1](#) έχουν ληφθεί. Όταν ο SOF_n ανιχνευτεί, ακολουθεί η αποκωδικοποίηση ενός καρέ ([Πίνακας 2.2](#)).



Marker	Purpose
DHT	Define Huffman Tables
DAC	Define Arithmetic Conditioning
DQT	Define Quantization Tables
DRI	Define Restart Interval
APP _n	Application defined marker
COM	Comment

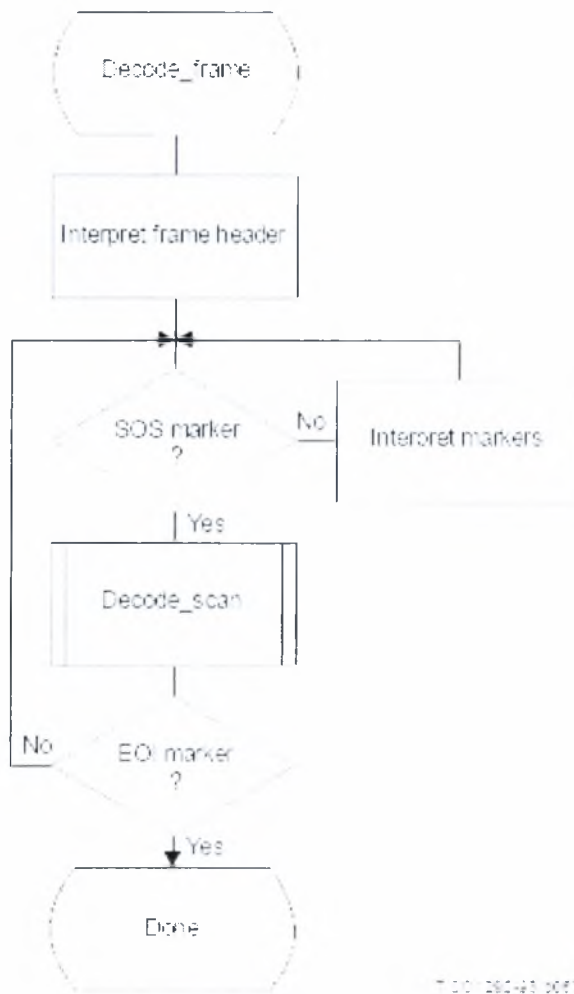
Η επιπρόσθετη λογική να μεταφράσεις αυτούς τους markers εμπεριέχεται στην διεργασία “Interpret markers”.

- Ο DHT θα μεταφραστεί από τις διεργασίες που χρησιμοποιούν κωδικοποίηση Huffman.
- Ο DAC θα μεταφραστεί από τις διεργασίες που χρησιμοποιούν αριθμητική κωδικοποίηση.
- Ο DQT θα μεταφραστεί από αποκωδικοποιητές που στηρίζονται στον διακριτό μετασχηματισμό συνημίτονου.
- Ο DRI θα μεταφραστεί από όλους τους αποκωδικοποιητές. Εξ ορισμού, οι διαδικασίες του “interpret markers” αφήνουν στο σύστημα τον επόμενο marker.

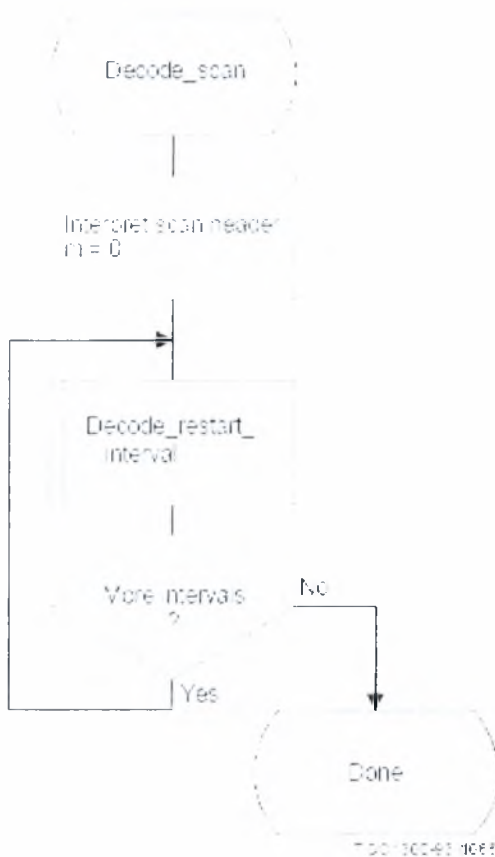
Αξίζει να σημειώσουμε ότι εάν ο marker SOI λείπει από την αρχή των κωδικοποιημένων δεδομένων, συνθήκη λάθους θα έχει ανιχνευτεί.

3.1 Αποκωδικοποίηση καρτέ

Στην **εικόνα 4.7** απεικονίζεται η διαδικασία αποκωδικοποίησης ενός καρτέ. Ο βρόχος τερματίζεται όταν ο marker EOI εντοπίζεται στο τέλος της σάρωσης. Όταν ο SOS marker ανιχνευτεί, ακολουθεί η αποκωδικοποίηση μιας σάρωσης.

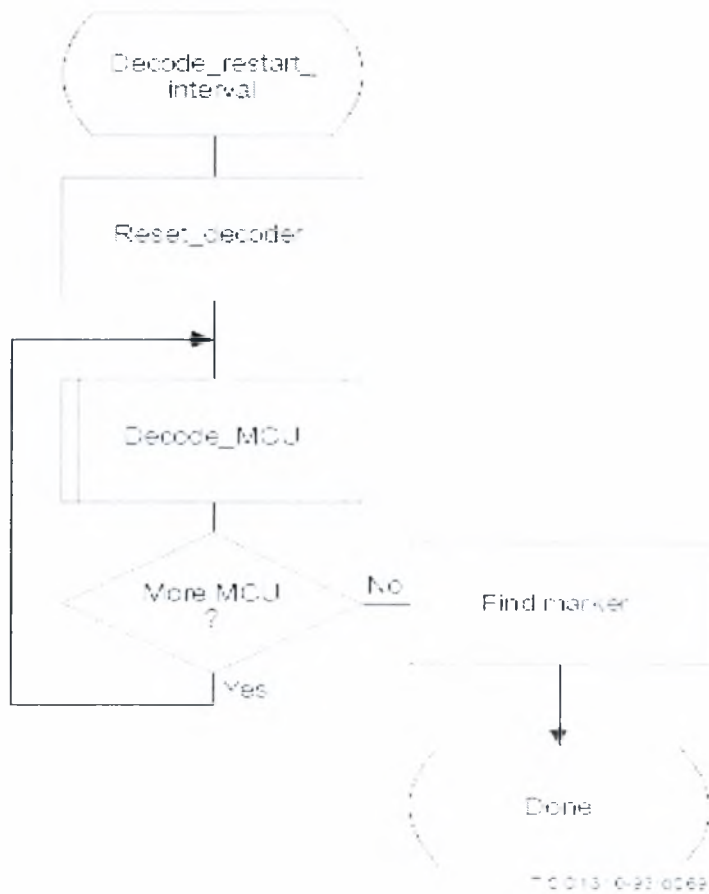


Η 4.8 δείχνει την διαδικασία αποκωδικοποίησης μιας σάρωσης. Ο βρόχος τερματίζει όταν ο απαιτούμενος αριθμός από τα restart intervals έχει αποκωδικοποιηθεί.



3.3 Αποκωδικοποίηση restart interval

Η διαδικασία αποκωδικοποίησης ενός restart interval παρουσιάζεται στην παρακάτω εικόνα (εικόνα 1.9). Η διαδικασία “Reset_decoder” επαναφέρει την πρόβλεψη στην καθορισμένη τιμή για όλα τα συστατικά εικόνας μέσα στη σάρωση και αποκωδικοποιεί κάθε MCU . Στο τέλος του restart interval , ο επόμενος marker τοποθετείται. Εάν κάποιο λάθος ανιχνευτεί, διορθώνεται.



Εικόνα 1.9

3.4 Αποκωδικοποίηση MCU

Η διαδικασία αποκωδικοποίησης ενός MCU παρουσιάζεται στην [εικόνα 3.10](#). Η παράμετρος N_b περιέχει τον αριθμό των μονάδων πληροφορίας μέσα στο MCU.



Εικόνα 3.10 Αποκωδικοποίηση MCU

Κεφάλαιο 5^ο

Καθορισμός προσαρμοσμένων Huffman πινάκων

Περιεχόμενα

1. Εισαγωγή.....	64
2. Παραγωγή Huffman πινάκων.....	65

Εισαγωγή

Η κωδικοποίηση Huffman χρησιμοποιείται στην κωδικοποίηση εντροπίας σε όλους τους τρόπους κωδικοποίησης.

Οι πίνακες Huffman καθορίζονται από μια λίστα 16 bytes (BITS) η οποία δίνει τον αριθμό των κωδίκων για κάθε μέγεθος κώδικα από ένα έως δεκαέξι. Ακολουθείται από μια λίστα με 8 bit τιμές συμβόλων (HUFFVAL) κάθε μια από τις οποίες ανατίθενται σε κάθε κώδικα Huffman. Οι τιμές των συμβόλων τοποθετούνται στη λίστα για να αυξήσουν το μέγεθος των κωδίκων. Μεγέθη κωδίκων μεγαλύτερα από 16 bits δεν επιτρέπονται. Επιπρόσθετα, οι κώδικες που δημιουργούνται έτσι ώστε όλοι οι 1-bit κώδικες κάθε μεγέθους να είναι προθέματα για μεγαλύτερους κώδικες.

Παρακάτω παρουσιάζεται η διαδικασία κατά την οποία οι πίνακες Huffman παράγονται από δυο λίστες (BITS και HUFFVAL). Ο τρόπος με τον οποίο αυτές οι δυο λίστες προσδιορίζονται δεν καθορίζεται. Όμως ακολουθούν τις βασικές αρχές της κωδικοποίησης Huffman.

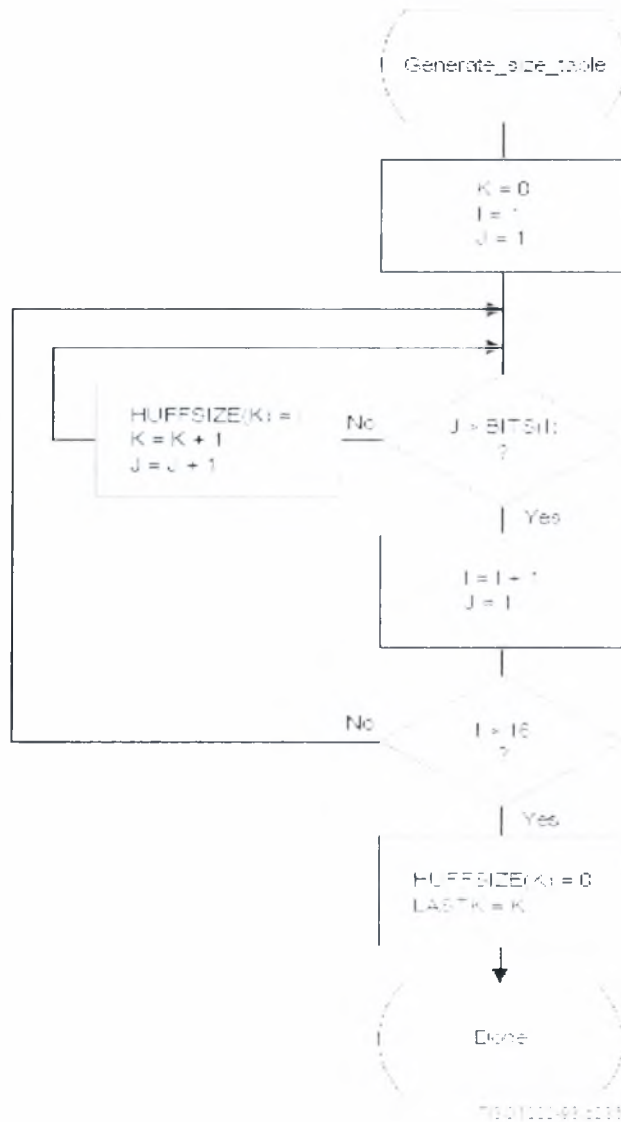
Ο DHT marker μέσα στα συμπιεσμένα δεδομένα υποδεικνύει την αρχή του ορισμού των πινάκων Huffman.

Παραγωγή Huffman πινάκων

Για την μετατροπή σε πίνακες από κωδικούς και μεγέθη κωδικών χρησιμοποιούνται 3 διαδικασίες.

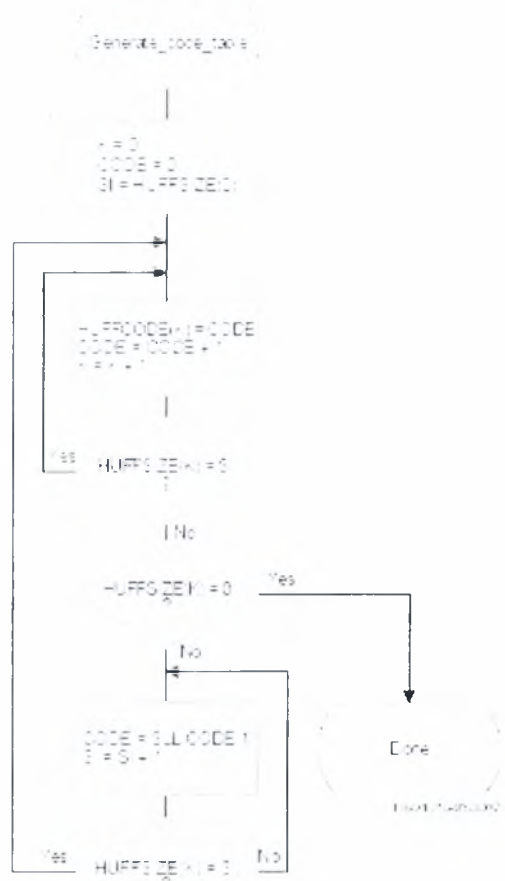
1. Η πρώτη δημιουργεί έναν πίνακα με μεγέθη κωδικών Huffman. Δηλαδή δημιουργείται ο πίνακας HUFFSIZE ().
2. Η δεύτερη παράγει τους κωδικούς από τον πίνακα που δημιουργήθηκε στο πρώτο βήμα. Δηλαδή δημιουργείται ο πίνακας HUFFCODE ().
3. Η Τρίτη δημιουργεί τους κωδικούς Huffman σε σειρά τιμών συμβόλου. Δηλαδή, δοθείσας μιας λίστας BITS (από 1 έως 16) η οποία περιέχει τους αριθμούς των

κωδίκων ανά μέγεθος και μια λίστα HUFFVAL η οποία περιέχει τις τιμές συμβόλων που σχετίζονται με τους κώδικες παραπάνω, δυο πίνακες δημιουργούνται (εικονα 7). Ο πίνακας EHUFCE και ο EHUFSE.



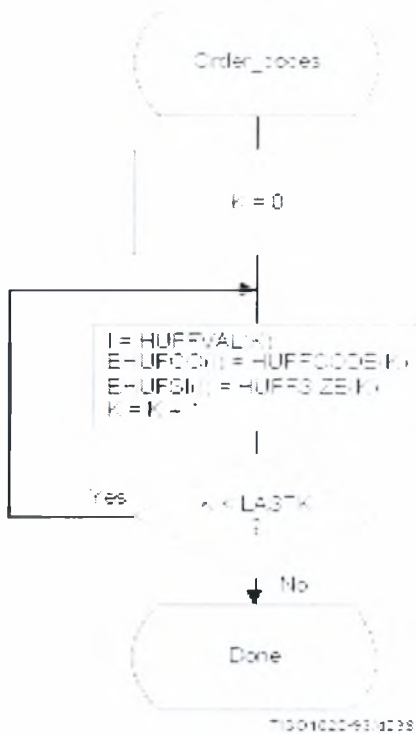
ΕΙΚΟΝΑ 7

- Η μεταβλητή lastk δείχνει την τελευταία είσοδο στον πίνακα.



- Το SLL CODE 1 σημαίνει αριστερή λογική ολίσθηση κατά ένα bit.

Οι δυο πίνακες HUFFCODE και HUFFSIZE έχουν δημιουργηθεί. Οι καταχωρήσεις στους πίνακες διατάσσονται σύμφωνα με αύξουσα αριθμητική τιμή και μέγεθος κώδικα Huffman. Οι πίνακες κωδικών της διαδικασίας κωδικοποίησης, EHUFCE και EHUFSE δημιουργούνται αναδιατάσσοντας τους κώδικες που έχουν καθοριστεί στους πίνακες HUFFCODE και HUFFSIZE σύμφωνα με τις τιμές συμβόλων που έχουν ανατεθεί σε κάθε κώδικα στον πίνακα HUFFVAL.



Κεφάλαιο 6^ο

Τεχνικές βελτιστοποίησης Lossless JPEG και ελαστικότητα

Περιεχόμενα

1. Εισαγωγή.....	70
2. Ανάλυση πηγαίου κώδικα.....	70
2.1 Δομή.....	70
2.2 Λειτουργία.....	71
2.2.1 Διαδικασία κωδικοποίησης.....	71
2.2.2 Διαδικασία αποκωδικοποίησης.....	72
2.2.3 Αρχεία εισόδου και εξόδου.....	72
2.2.4 Εικόνες ppm(<i>Portable PixMap</i>).....	74
2.3 Επιλογές κατά την διαδικασία κωδικοποίησης.....	75
2.4 Εργαλεία που χρησιμοποιήθηκαν.....	75
2.5 Παράδειγμα κωδικοποίησης και αποκωδικοποίησης.....	75
3. Τεχνικές βελτιστοποίησης.....	84
3.1 Βελτιστοποίηση 1η (<i>newecoder1.0</i>).....	85
3.2 Βελτιστοποίηση 2η (<i>newecoder2.0</i>).....	89
3.3 Βελτιστοποίηση 3η (<i>newecoder3.0</i>).....	92
Συμπεράσματα και Προτάσεις.....	97

1. Εισαγωγή

Η μεταπτυχιακή αυτή διατριβή έχει ως στόχο να βελτιώσει την απόδοση του Lossless JPEG κωδικοποιητή. Δηλαδή, του κωδικοποιητή που δεν στηρίζεται στον διακριτό μετασχηματισμό συνημίτονου αλλά στην πρόβλεψη. Ο πηγαίος κώδικας πάνω στον οποίο βασίστηκε η διπλωματική βρίσκεται στο url: http://compression-links.info/JPEGLS?search_score=%3E%3D5 με το όνομα `jpegls`.

2. Ανάλυση πηγαίου κώδικα

2.1 Δομή

Ο πηγαίος κώδικας αποτελείται από τα δέκα αρχεία γραμμένα σε γλώσσα προγραμματισμού C (`write.c` - `huffd.c`) και από 6 αρχεία επικεφαλίδας (`jpeg.h` - `jpeglib.h`).

<u>Αρχείο .c</u>	<u>περιγραφή</u>
write.c	Ρουτίνες για εγγραφή των Jpeg αρχείων
util.c	Ρουτίνες που χρησιμοποιούνται στον κωδικ / αποκωδικ.
read.c	Ρουτίνες για διάβασμα Jpeg αρχείων
predictor.c	Ρουτίνες για υπολογισμό πρόβλεψης
rhmtoljpg.c	Κύρια ρουτίνα για τη διαδικασία κωδικοποίησης Jpeg
pmread.c	Ρουτίνες για διάβασμα εικόνων
mcu.c	Ρουτίνες για διαχείριση mcu
ljpgtorpm.c	Κύρια ρουτίνα για τη διαδικασία αποκωδικοποίησης Jpeg
huffc.c	Ρουτίνες για κωδικοποίηση Jpeg
huffd.c	Ρουτίνες για αποκωδικοποίηση Jpeg

<u>Αρχεία .h</u>	<u>περιγραφή</u>
proto.h	Δηλώσεις συναρτήσεων
predictor.h	Υπολογισμοί πρόβλεψης
pnmtoljrg.h	Καθολικές μεταβλητές
mcu.h	δηλώσεις
jpeg.h	Βασικές δομές
io.h	Εγγραφή αρχείων

2.2 Λειτουργία

Τα αρχεία του πηγαίου κώδικα χρησιμοποιούνται σε δυο ξεχωριστές διαδικασίες. Ορισμένα από αυτά χρησιμοποιούνται στην διαδικασία κωδικοποίησης ενώ κάποια άλλα στην διαδικασία αποκωδικοποίησης. Κάποια χρησιμοποιούνται και στις δυο διαδικασίες.

2.2.1 Διαδικασία κωδικοποίησης

Η κωδικοποίηση ενός lossless JPEG κωδικοποιητή πραγματοποιείται σε πέντε βήματα.

1. Διάβασμα της εικόνας από το αρχείο εισόδου και αποθήκευση σε κάποιον buffer.
2. Δημιουργία είτε καθορισμένων Huffman πινάκων είτε προσαρμοσμένων ύστερα από επιλογή καλύτερου τρόπου πρόβλεψης σε όλο το αρχείο.
3. Εγγραφή επικεφαλίδας καρέ.
4. Εγγραφή επικεφαλίδας σάρωσης και άλλων marker κομματιών.
5. Κωδικοποίηση εικόνας και εγγραφή κωδικοποιημένης πληροφορίας στο νέο συμπιεσμένο αρχείο σύμφωνα με την επιλογή πρόβλεψης

2.2.2 Διαδικασία αποκωδικοποίησης

Η αποκωδικοποίηση ενός lossless JPEG κωδικοποιητή πραγματοποιείται σε τέσσερα βήματα.

1. Διάβασμα επικεφαλίδας καρέ.
2. Διάβασμα επικεφαλίδας σάρωσης .
3. Εγγραφή επικεφαλίδας αναδημιουργημένης εικόνας.
4. Αποκωδικοποίηση και εγγραφή αποκωδικοποιημένης πληροφορίας στο νέο αρχείο

2.2.3 Αρχεία εισόδου και εξόδου

Το αρχείο εισόδου στην διαδικασία κωδικοποίησης θα είναι μια εικόνα με κατάληξη της μορφής .ppm και το αρχείο εξόδου θα είναι της μορφής .ljpg. Το αρχείο εισόδου της διαδικασίας αποκωδικοποίησης θα είναι το αρχείο .ljpg που προέκυψε από την διαδικασία κωδικοποίησης και το αρχείο εξόδου θα είναι μια νέα εικόνα της μορφής .ppm (εικόνα 6.1).



Εικόνα 6.1

Οι εικόνες .ppm που χρησιμοποιήθηκαν σε αυτή την εργασία είναι οι :



Figure 1.10



Figure 1.11

2.2.4 *Εικόνες ppm(Portable PixMap)*

Το συγκεκριμένο format είναι ένας απλός τρόπος για την κωδικοποίηση μιας εικόνας. Αποτελείται από μια επικεφαλίδα και από το κύριο σώμα της εικόνας. Χρησιμοποιούμε το format για να αναπαραστήσουμε έγχρωμες εικόνες. Κάθε δείγμα αποτελείται από 3 byte (RGB) τα οποία αναπαρίστανται ως μη προσημασμένοι χαρακτήρες. Η τιμή (255,255,255) είναι το απόλυτο λευκό.

Η δομή της επικεφαλίδας ενός .ppm αρχείου είναι η ακόλουθη :

- αποτελείται από έναν μαγικό αριθμό –P6 αν η εικόνα έχει byte format δηλαδή ένα byte ανά συστατικό εικόνας ανά δείγμα είτε P3 αν έχει γραφτεί σαν κείμενο με ascii χαρακτήρες.
- Από σχόλια τα οποία αρχίζουν με '#’.
- Από έναν αριθμό που είναι το πλάτος της εικόνας.
- Από έναν αριθμό που είναι το ύψος της εικόνας.
- Από έναν αριθμό που είναι η μεγαλύτερη τιμή ενός δείγματος και υποδηλώνει την ακρίβεια του δείγματος.
- Από έναν χαρακτήρα whitespace. Συνήθως χαρακτήρας νέας γραμμής.

Το κύριο σώμα της εικόνας αποτελείται από μια ακολουθία από γραμμές, όση και το ύψος της εικόνας. Κάθε γραμμή αποτελείται από δείγματα , όσα και το πλάτος της εικόνας. Κάθε δείγμα αποτελείται από 3 συστατικά εικόνας (R,G,B) με αυτή τη σειρά. Για παράδειγμα,

```
P3
# example from the man page
4 4
15
 0 0 0   0 0 0   0 0 0   15 0 15
 0 0 0   0 15 7  0 0 0   0 0 0
 0 0 0   0 0 0   0 15 7  0 0 0
15 0 15  0 0 0   0 0 0   0 0 0
```

2.3 Επιλογές κατά την διαδικασία κωδικοποίησης

Κατά την διαδικασία κωδικοποίησης, έχουμε την ακόλουθες επιλογές.

1. Επιλογή καθορισμένων πινάκων Huffman (-d)
2. Επιλογή παραμέτρου point transform (-p)
3. Επιλογή προσαρμοσμένων πινάκων Huffman
4. Επιλογή restart interval (-r)


Οι πρώτες τρεις επιλογές επηρεάζουν το μέγεθος του συμπιεσμένου αρχείου ενώ η τέταρτη όχι.

2.4 Εργαλεία που χρησιμοποιήθηκαν

Για την ολοκλήρωση της εργασίας χρησιμοποιήθηκαν ορισμένα λογισμικά σχετικά με την ανάπτυξη πηγαίου κώδικα, image viewers , λογισμικά για την σύγκριση δυαδικών αρχείων καθώς και batch files για την αυτοματοποίηση της διαδικασίας. Συγκεκριμένα

- για την ανάπτυξη πηγαίου κώδικα χρησιμοποιήθηκε το «Microsoft Visual Studio 2005»
- Image viewer ο «IrfanView».
- για την σύγκριση δυαδικών αρχείων το λογισμικό «Ultra Compare Professional – Binary Compare»

2.5 Παράδειγμα κωδικοποίησης και αποκωδικοποίησης

Στην  φαίνεται ένα μέρος της δομής της εικόνας F-18.ppm .

D:\DIPLOMATIKI\REFERENCE_CODE\standardencoder\images\F-18new.jpg

00000000	FF	DE	FF	03	00	11	03	00	F0	01	40	03	00	11	00	01	φ2φΑ...Δ.Θ.....
00000010	11	00	02	11	00	FF	04	00	10	00	00	03	01	01	01	01φΑ.....
00000020	01	01	00	00	00	00	00	00	00	00	00	01	02	03	04	05
00000030	06	07	08	FF	04	00	1D	01	00	03	01	01	01	01	01	01	...φΑ.....
00000040	01	00	00	00	00	00	00	00	00	01	02	03	04	06	05	07
00000050	08	09	FF	04	00	10	02	00	03	01	01	01	01	01	01	00	..φΑ.....
00000060	00	00	00	00	00	00	00	00	01	02	03	04	05	06	07	08
00000070	FF	DA	00	00	03	00	00	01	10	02	20	05	00	00	F0	1B	φΥ.....φ.
00000080	F9	FF	00	F0	D4	71	EC	E7	F7	9F	DC	7E	B9	D8	F7	DF	φγ.80qig-ΥΥ~'2+3
00000090	A4	74	07	21	37	55	5E	5E	DC	FD	0D	58	00	00	00	00	κφ!...·0φIX....
000000a0	10	B0	00	02	9B	03	30	F3	1B	9B	D8	06	D8	6D	24	00	..'.><δ.>2.ύμ&
000000b0	12	A6	AA	1B	69	29	99	00	00	06	57	CB	55	4D	B4	A6	..φ.ιφ'...+EUMφ
000000c0	22	2E	F4	49	31	D4	74	32	86	D8	24	90	30	1A	9A	01	"..δΙ10τ2+3φ0.βΑ
000000d0	29	B3	B4	94	00	B6	00	1B	67	AA	18	08	18	44	08	1A)»'«ΙφΑ..+...DE.
000000e0	3A	D5	69	43	49	24	14	80	00	69	DE	01	20	01	32	B3	:δiCi&.ε.ιφ..2>
000000f0	8D	B4	3C	4D	90	EA	AA	AE	E2	21	B7	11	14	3D	61	30	κ'<κ&+α&ι...=α0
00000100	00	00	00	00	06	20	00	02	9B	00	04	0E	63	97	74	04>..ίc7τ.
00000110	0C	E8	AE	59	09	24	93	91	B6	DD	36	36	E6	66	65	01	ίε&κ.α"φγ6&α&ε.
00000120	A5	F4	4C	80	53	72	93	01	30	59	38	09	69	A6	80	67	Υ&L&Sr".*Υ".ιίεφ
00000130	9E	6E	A5	09	99	84	08	00	6D	82	CF	30	EE	ED	23	7B	ζη".*ΠI&κ,ί<ii+{
00000140	D1	4C	A4	86	E6	64	00	00	01	50	05	1B	6A	66	52	A0	ΝL&τ&d...ε...jφR
00000150	D5	58	50	00	01	00	92	4A	F4	D1	4C	CA	4D	B1	24	02	δ&ε...·J&NLE&+ε.
00000160	49	2B	B3	00	00	12	40	00	33	16	A4	57	7A	44	67	55	I+»...@.3.κWzDgU
00000170	4D	B4	32	15	BA	E8	13	00	04	90	06	38	02	B1	99	94	Υ'...ε&...ε&Α+R"
00000180	93	6D	34	A5	55	59	E7	9E	3A	6A	00	00	0D	A9	94	90	"m.φWYφ&φj...ε"
00000190	DE	5D	00	0E	9A	5C	44	80	D8	91	38	16	F8	49	24	9C	είί&κ&ε0'3.δiε'
000001a0	4C	10	24	92	50	E6	A9	30	03	AA	01	24	92	60	36	92	L.ε'ε&εε.·.ε'·6'
000001b0	02	98	49	5D	DA	48	59	00	00	1B	60	00	0A	68	14	4C	..IjJ&κ...·.k.L
000001c0	44	55	55	55	4C	CA	4A	AB	45	3A	53	00	08	09	26	DD	DUUVLE&κ&ε;f...εφ
000001d0	55	44	08	00	00	DB	00	00	43	00	14	44	36	EA	AA	66	UD&...·.CA.D&ε+ε
000001e0	40	B7	4E	20	4D	00	ED	02	AA	6D	53	A2	28	62	AA	80	@.N.Y.ι..*μφ&ε&ε

φ.18new.jpg

Μόλις ολοκληρώθηκε η διαδικασία κωδικοποίησης. Τα αποτελέσματα παρουσιάζονται στην

```
C:\Windows\system32\cmd.exe
Point transform parameter is 0.
Restart every 0 row(s).
ppm file
320 240
255
Use optimal Huffman table.
PSU HuffSymSum AddBitsSum
1      67211      41415
2      73250      49900
3      76571      57107
4      67898      41186
5      66963      40408
6      69910      44790
7      70043      45102
Selected PSU          : 5
Original file size    : 230415 bytes
Compressed file size  : 107760 bytes
Compression ratio     : 2.14
Press any key to continue . . .
```

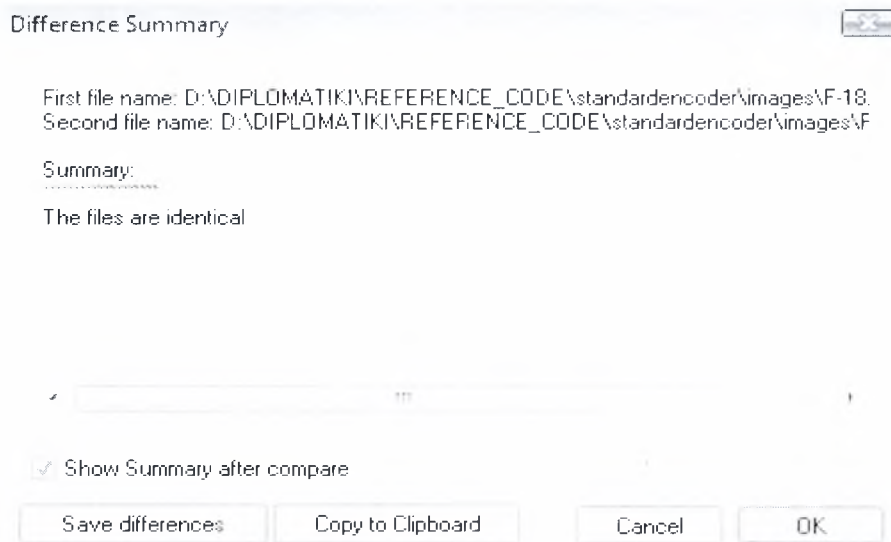
Εικόνα 6.7: Η δομή του αρχείου ppm

Κατά την αποκωδικοποίηση, ο lossless αποκωδικοποιητής θα δεχθεί ως αρχείο εισόδου το F-18new.ljrg και θα παράγει ένα νέο αρχείο F-18new.ppm. Ένα μέρος της δομής του φαίνεται στην παρακάτω εικόνα ([εικόνα 6.7](#)).

D:\DIPLOMATIKI\REFERENCE_CODE\standardencoder\images\F-18new.ppm

00000000	50 38 0A 33 32 30 20 32 34 30 0A 32 35 35 0A 07	F6,320,240,255,..
00000010	00 0E 06 00 02 00 01 00 58 72 20 9D 39 74 A6 AD,° -
00000020	69 A6 38 4C A1 36 4F A7 AD 5D A8 AC 60 A8 AC 60	i ! uOS-]'-"-
00000030	A3 AC 60 A3 AC 60 A3 AC 60 A3 AC 60 A9 AC 60 AA	-----@-*
00000040	AA 61 AA AA 61 AA AA 61 AA AA 61 AC AB 62 AC AB	*a*a*a*a*a-<b-<
00000050	62 A3 A7 5E A3 A5 5F A6 AB 65 A6 AB 65 A7 AC 66	b"§^" _ «e «eS-£
00000060	A3 AC 67 A7 AC 66 A7 AC 66 A7 AC 66 A5 AB 64 A7	^-gS-5S-5S-5Y«dS
00000070	AD 61 A3 AE 62 A7 AC 61 A3 AB 5F A5 AB 5F A5 AB	-a'0bS-aY«_Y«_Y«
00000080	5F A3 AB 5F A5 AC 5E A5 AD 64 A7 AF 56 A3 B0 57	_Y«_Y-^Y-Is^V^W
00000090	A7 AF 58 A4 AC 62 A7 AF 58 A6 AE 58 A7 AE 56 A9	S^U«-RS^V 0U§0Ve
000000a0	A3 5E AA AB 60 AA AC 61 AB AC 61 AA AB 60 A9 AB	<« «^ «-a«-a « @«
000000b0	60 AA AB 60 AA AC 5F A3 AE 68 A7 AC 56 AA B0 59	«^ «^ «0X§-V^V
000000c0	A9 AF 58 A7 AD 56 A3 AE 57 A3 AE 57 A3 AF 58 A5	@^X§-V^0W^0W^XY
000000d0	B0 5D A4 B0 5E A4 B0 5E A3 B0 5E A4 B0 5E A1 AE	^ 0V^ «^ «^ «^ «
000000e0	50 A1 AE 50 A5 AF 5E AA AE 62 AC AE 63 AB AC 62	V 0V^ «^ «b-00«-b
000000f0	AA AC 61 AA AB 60 AA AB 60 AA AB 60 AA AC 5F AB	«a «^ «^ «^ «_«
00000100	AF 58 AB AF 56 AB AF 56 AB AF 56 A9 AD 55 AA AE	^X«^V«^V«^V«-U^0
00000110	56 AB AF 57 AA AF 58 A4 AF 5E A3 AF 5F A3 B1 61	V«^W^X«^a^ _Y±a
00000120	A7 B4 64 A4 B1 61 A3 B2 62 A2 AF 5F A4 AF 60 A9	S^d±±aY^bc^_x^ «
00000130	AC 5F AA AC 5F AA AC 5F AA AC 5F AA AC 5F AA AC	-----
00000140	5F AA AC 5F AA AC 5F AB AC 5E AB AC 5E AB AC 5E	-----«-«-«-«-^
00000150	AB AC 5E AC AD 5F AD AD 5F AD AD 5F AC AE 5F A3	«-^--- -----0_£
00000160	AE 53 A4 B1 5D A4 B1 5D A2 AF 53 A7 B4 5F A6 B3	0[x±]x±]0^-[S^]^
00000170	5F A5 B2 5E A4 B1 5D A5 AF 5D A6 B0 5E A7 B1 60	_Y^±x±]Y^]^±S^
00000180	A5 B2 60 A9 B3 61 A7 B1 5F A5 AF 5E A5 AF 5E A9	^ ^@^a±_Y^Y^ «^0
00000190	B2 66 A7 B0 66 A7 B0 66 A9 B1 67 A7 B0 66 A3 B0	±±S^±S^±S^0±gS^±±°
000001a0	66 A3 B1 67 A3 B1 66 A4 B1 5F A4 B0 5E A3 B4 62	±±±g^±±±±_x^±^ «
000001b0	A3 B4 62 A4 B0 5E A5 B2 60 A5 B2 60 A3 B0 5E A2	^ «x^±Y^±Y^±Y^±°
000001c0	B2 5F A2 B3 60 A2 B3 60 A3 B4 61 A3 B4 60 5F B0	^_0^ «^ «^ «aB^Y^°
000001d0	50 A2 B3 5F A1 B2 5F A3 B3 63 A7 B3 61 A6 B2 60	V«^ _ ^ «0S^a^
000001e0	A6 B3 61 A4 B1 5F A4 B1 5F A4 B1 5F A4 B1 5F A4	^ax±_x±_x±_x±_x±

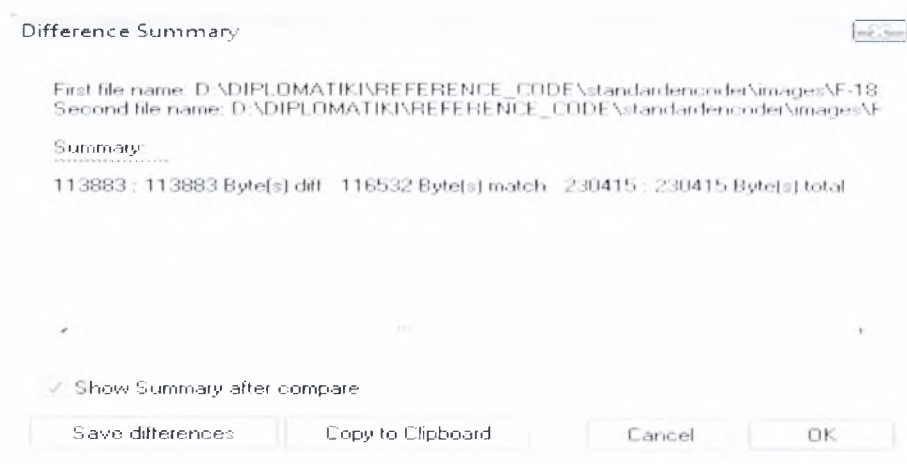
Η εικόνα φαίνεται αρκετά ίδια με την εικόνα. Το λογισμικό UltraCompare το επιβεβαιώνει κατά την σύγκριση()



Εικόνα 6.9. Σύγκριση αρχείων

Αυτό που πετύχαμε είναι να επιβεβαιώσουμε ότι όντως ο κωδικοποιητής και αποκωδικοποιητής είναι lossless, δηλαδή δεν έχουμε απώλεια πληροφορίας, το οποίο είναι και το ζητούμενο.

Εάν επιλέξουμε την παράμετρο point transform και επιλέξουμε να ελέγξουμε εάν όντως έχουμε κωδικοποίηση και αποκωδικοποίηση χωρίς απώλειες δεδομένων θα διαπιστώσουμε ότι κατά την σύγκριση τα αρχεία δεν είναι πανομοιότυπα (\neq).



Εικόνα 6.10. Σύγκριση αρχείων

Αυτό οφείλεται στο γεγονός ότι όταν η παράμετρος `point transform` είναι ενεργοποιημένη έχουμε μια μεταβολή στην ποσότητα των δειγμάτων. Παρόλα αυτά αν αποδείξουμε ότι η διαφορά κάθε byte μεταξύ της αρχικής εικόνας `.ppm` και της νέας εικόνας που δημιουργείται είναι μικρότερη από $2^{P_i} - 1$, τότε έχουμε μεταβολή στην τιμή ενός δείγματος στα λιγότερα σημαντικά bits χωρίς όμως να επηρεάζεται ολοκληρωτικά η τιμή και κατ' επέκταση το χρώμα της. Για παράδειγμα,

Έστω ο αριθμός δέκα ο οποίος γράφεται με 1010 και $P_i=3$. Τότε $2^{P_i} - 1 = 7$.

(10) 1010 $\ll 3 \Rightarrow$ 0001 $\gg 3 \Rightarrow$ 1000 (8)

Η διαφορά $10 - 8 = 2 < 7$ άρα είμαστε σωστοί. Όσο η διαφορά πλησιάζει στην επιτρεπόμενη τιμή τόσο χάνεται πληροφορία (`precision loss`) αλλά και τόσο μειώνεται η ποσότητα της συμπιεσμένης πληροφορίας



F-18.ppm



F-18newp1.ppm



F-18newp4.ppm



F-18newp6.ppm

Εικόνα 5.1 Η εικόνα ppm με 6-bit ανά pixel

Ο παρακάτω κώδικας εκτελεί τον έλεγχο.

```

if (formata==formatb)
{
    if (rowza==rowsb && colza==colzb)
    {
        if (!valuea==valueb)
        {
            maxval=valuea+valueb;
            do{
                cha=getc(infile1);
                chb=getc(infile2);
                if (cha-chb>maxval)
                {
                    printf("mistake\n");
                    exit(-1);
                }
            }while(cha!=EOF || chb!=EOF);
        }
        else
        {
            exit(-1);
        }
    }
    else
    {
        exit(-1);
    }
}
}

```

Τα αποτελέσματα στον απεικονίζουν το μέγεθος του συμπιεσμένου αρχείου για τις διάφορες επιλογές κωδικοποίησης.

	STANDARDENCODER	STANDARDENCODER
IMAGES	F-18.ppm	pepers.ppm
	SIZE(bytes)	SIZE(bytes)
ORIGINAL	230415	786447
DEFAULT HUFFMAN TABLES WITHOUT HEADERS	118256	490820
DEFAULT HUFFMAN TABLES WITH HEADERS *	118324	490888
ORIGINAL	230415	786447
CUSTOMIZED HUFFMAN TABLES WITHOUT HEADERS	107634	476222
CUSTOMIZED HUFFMAN TABLES WITH HEADERS *	107760	476347
ORIGINAL	230415	786447
P1	83067	405197
P2	61849	332655
P4	40138	189275

3. Τεχνικές βελτιστοποίησης

Όπως περιγράφηκε και στην παράγραφο 2.2.1 του παρόντος κεφαλαίου, η διαδικασία κωδικοποίησης ενός lossless JPEG κωδικοποιητή πραγματοποιείται σε πέντε βήματα.

1. Διάβασμα της εικόνας από το αρχείο εισόδου και αποθήκευση σε κάποιον buffer.
2. Δημιουργία είτε καθορισμένων Huffman πινάκων είτε προσαρμοσμένων ύστερα από επιλογή καλύτερου τρόπου πρόβλεψης σε όλο το αρχείο.
3. Εγγραφή επικεφαλίδας καρέ.
4. Εγγραφή επικεφαλίδας σάρωσης και άλλων marker κομματιών.
5. Κωδικοποίηση εικόνας και εγγραφή κωδικοποιημένης πληροφορίας στο νέο συμπίεσμένο αρχείο σύμφωνα με την επιλογή πρόβλεψης

Για την επιλογή του καλύτερου τρόπου πρόβλεψης, ακολουθείται η εξής διαδικασία: Αρχικά βρίσκονται οι συχνότητες των συμβόλων προς κωδικοποίηση σε όλη την εικόνα. Στη συνέχεια, για κάθε τρόπο πρόβλεψης από τους επτά δημιουργούνται οι πίνακες Huffman. Καλύτερος τρόπος πρόβλεψης θεωρείται αυτός που χρειάζεται την λιγότερη πληροφορία για την αναπαράσταση των συμβόλων (). Τέλος η κωδικοποίηση γίνεται με τους πίνακες Huffman που παράχθηκαν σύμφωνα με τον επιλεγμένο τρόπο πρόβλεψης για όλο το αρχείο.

*****	*	*****	****
	*		
	*		

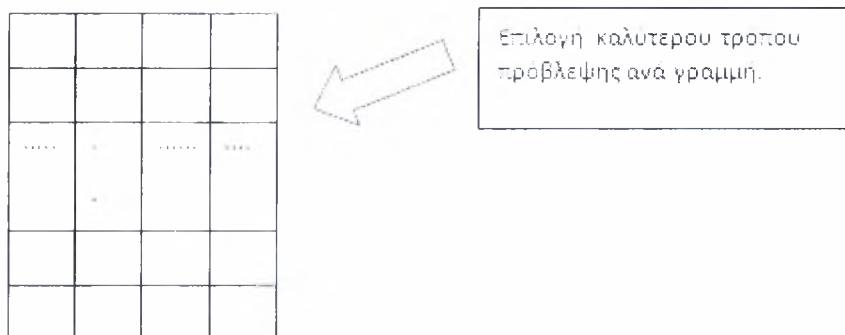


Επιλογή καλύτερου τρόπου πρόβλεψης σε όλη την εικόνα.

3.1 Βελτιστοποίηση 1^η (*newecoder1.0*)

Περιγραφή:

Η πρώτη βελτιστοποίηση που πραγματοποιήθηκε στην διατριβή αυτή είναι η κωδικοποίηση ανά γραμμή και όχι κωδικοποίηση σε όλο το αρχείο. Για να γίνει κωδικοποίηση ανά γραμμή, πρέπει να βρεθεί ο καλύτερος τρόπος πρόβλεψης για κάθε γραμμή (), να παραχθούν οι πίνακες Huffman για κάθε γραμμή και η κωδικοποίηση για κάθε γραμμής να γίνει σύμφωνα με τον επιλεγμένο τρόπο κωδικοποίηση και το αντίστοιχο πίνακα Huffman.



Συναρτήσεις:

Για τον σκοπό αυτό γράφτηκαν οι παρακάτω συναρτήσεις :

```
1. void HuffOptimize_line (CompressInfo *cPtr, int k);
```

```
// CompressInfo *cPtr : pointer to line CompressInfo
```

```
// int k: row number
```

Η συνάρτηση αυτή βρίσκει την καλύτερη πρόβλεψη για κάθε γραμμή και δημιουργεί τους κατάλληλους προσαρμοσμένους πίνακες Huffman για κάθε γραμμή.

```
2. void FreqCountAllSelValue_first_row(CompressInfo *cPtr)
```

```
//CompressInfo *cPtr : pointer to line CompressInfo
```

Η συνάρτηση αυτή βρίσκει την συχνότητα των συμβόλων μόνο της πρώτης γραμμής.

```
3.void FreqCountAllSelValue_each_row(CompressInfo *cPtr,int k)
```

```
// CompressInfo *cPtr : pointer to line CompressInfo
```

```
// int k:row number
```

Η συνάρτηση αυτή βρίσκει την συχνότητα των συμβόλων μόνο κάθε γραμμής εκτός από την πρώτη.

```
4. void StdPickSelValue_rows (CompressInfo *cPtr);
```

```
//CompressInfo *cPtr : pointer to line CompressInfo
```

Η συνάρτηση αυτή επιλέγει την καλύτερη πρόβλεψη για τους καθορισμένους πίνακες Huffman

```
5.void HuffEncode_each_row(CompressInfo * cPtr, MCU *curRowBuf, MCU *prevRowBuf)
```

```
// CompressInfo * cPtr: pointer to line CompressInfo
```

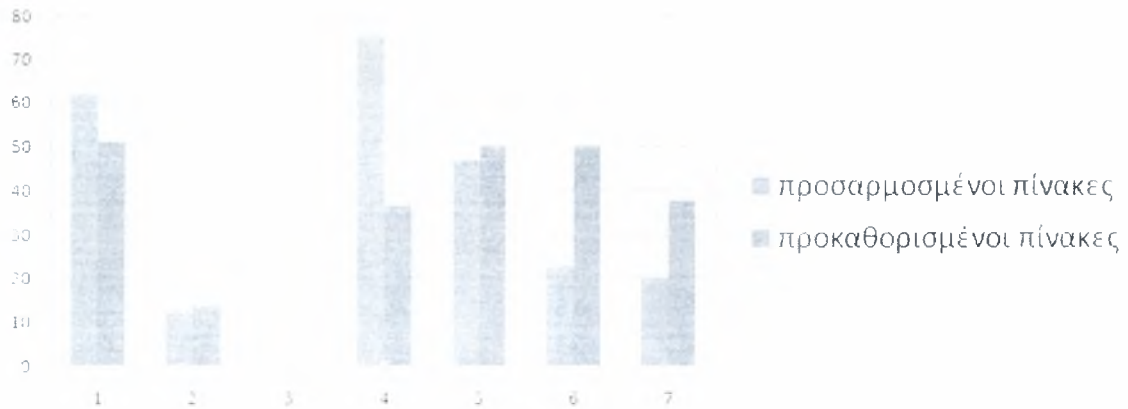
```
// MCU *curRowBuf:pointer to line
```

```
// MCU *prevRowBuf:pointer to previous line
```

Η συνάρτηση αυτή εκτελεί την κωδικοποίηση κάθε γραμμής.

Αποτελέσματα:

Η συχνότητα της κάθε πρόβλεψης για τους προσαρμοσμένους πίνακες Huffman και για τους καθορισμένους αναπαρίσταται στο παρακάτω ιστόγραμμα .



Εικόνα 6.3: Συγκριτική ανάλυση

Τα αποτελέσματα της κωδικοποίησης ανά γραμμή παρουσιάζονται στον πίνακα 6.4.

		STANDARDCODER	NEWENCODER1.0	
IMAGES	F-18.ppm	F-18.ppm	%	
	SIZE(bytes)	SIZE(bytes)		
ORIGINAL	230415	230415		
DEFAULT HUFFMAN TABLES WITHOUT HEADERS	118256	114080	-3.5	
DEFAULT HUFFMAN TABLES WITH HEADERS *	118324	122274	3.3	
ORIGINAL	230415	230415		
CUSTOMIZED HUFFMAN TABLES WITHOUT HEADERS	107634	98053	-8.9	
CUSTOMIZED HUFFMAN TABLES WITH HEADERS *	107760	117976	9.4	
ORIGINAL	230415	230415		
P1	83067	95713	15.2	
P2	61849	77062	24.5	
P4	40138	56060	39.6	

		STANDARDENCODER	NEWENCODER1.0	
IMAGES	pepers.ppm		pepers.ppm	%
		SIZE(bytes)	SIZE(bytes)	
ORIGINAL	786447	786447		
DEFAULT HUFFMAN TABLES WITHOUT HEADERS	490820	480029		-2.1
DEFAULT HUFFMAN TABLES WITH HEADERS *	490888	497471		1.34
ORIGINAL	786447	786447		
CUSTOMIZED HUFFMAN TABLES WITHOUT HEADERS	476222	458798		-3.6
CUSTOMIZED HUFFMAN TABLES WITH HEADERS *	476347	503945		5.7
ORIGINAL	786447	786447		
P1	405197	434953		7.34
P2	332655	363530		9.28
P4	189275	224288		18.4

Πίνακας 3.4. Συμπίεση του αρχείου

Συμπεράσματα:

1. Με αυτή τη βελτιστοποίηση πετυχαίνουμε καλύτερη συμπίεση δεδομένων, χωρίς όμως την εγγραφή των αρχείων επικεφαλίδας. Όταν όμως εγγράψουμε και τα αρχεία επικεφαλίδας τότε η χωρητικότητα του συμπιεσμένου αρχείου αυξάνεται. Αυτό εξηγείται ως εξής: Χωρίς την εγγραφή των αρχείων επικεφαλίδας, έχουμε λιγότερη χωρητικότητα διότι κάθε γραμμή έχει κωδικοποιηθεί με ξεχωριστό πίνακα Huffman. Αυτό πρακτικά σημαίνει ότι τα σύμβολα προς κωδικοποίηση θα έχουν μικρότερο μήκος αφού η συχνότητα των συμβόλων είναι μικρότερη. Όταν όμως εγγράψουμε και τα αρχεία επικεφαλίδας, πρέπει να εγγράψουμε όλους τους πίνακες Huffman κάθε γραμμής. Στην προκειμένη περίπτωση, εγγράφουμε στο συμπιεσμένο αρχείο 240 πίνακες.
2. Ένα άλλο αρνητικό στοιχείο είναι ότι χρειαζόμαστε αρκετούς κύκλους μηχανής. Για κάθε γραμμή, ελέγχουμε και τους επτά τρόπους πρόβλεψης και δημιουργούνται πίνακες Huffman για κάθε πρόβλεψη μέχρι να αποφασίσουμε ποιά πρόβλεψη είναι καλύτερη.

3. Τέλος, παρατηρούμε ότι με την ενεργοποίηση του point transform parameter, μειώνεται η χωρητικότητα του αρχείου.

3.2 Βελτιστοποίηση 2^η (newencoder2.0)

Περιγραφή:

Η δεύτερη βελτιστοποίηση την οποία προτείνουμε προκύπτει από το δεύτερο συμπέρασμα της πρώτης βελτιστοποίησης. Σύμφωνα λοιπόν με αυτό το συμπέρασμα, ο αρνητικός παράγοντας είναι ότι ενώ η καθαρή πληροφορία της εικόνας είναι μικρότερη από την αρχική, αναγκαζόμαστε να γράψουμε τους πίνακες Huffman για κάθε γραμμή με αποτέλεσμα η χωρητικότητα του τελικού συμπιεσμένου αρχείου να αυξάνεται σημαντικά. Αυτό που προτείνουμε είναι κωδικοποίηση ανά γραμμή αλλά συνολικά θα έχουμε 7 διαφορετικούς πίνακες Huffman, έναν για κάθε τρόπο πρόβλεψης. Για να γίνει αυτό, κάθε φορά που θα βρίσκουμε την καλύτερη πρόβλεψη ανά γραμμή θα προσθέτουμε τα ιστογράμματα. Στο τέλος, ο extended πίνακας με τα σύμβολα για την πρόβλεψη 1 για παράδειγμα, θα περιέχει όλα τα ιστογράμματα των γραμμών που κωδικοποιήθηκαν με την πρόβλεψη 1. Τέλος, θα δημιουργηθεί ο πίνακας Huffman με βάση τα αθροισμένα ιστογράμματα.

Ψευδοκώδικας για υπολογισμό Extended Huffman Table :

```
For (i=0; i<imagelines; i++){
Psv=Find_line_prediction ();
Add_frequencies (psv);
}
```

Συναρτήσεις:

Για το σκοπό αυτό, χρησιμοποιήθηκαν οι συναρτήσεις της προηγούμενης βελτιστοποίησης και ακόμη:

```
1. void FreqCountSumInit (void);
```

Η συνάρτηση αυτή αρχικοποιεί τον πίνακα με τις αθροισμένα ιστογράμματα

```
2. void GenExtendedHuffTable ( CompressInfo *cPtr, int k);
```

// CompressInfo *cPtr : pointer to line CompressInfo

// int k:row number

Η συνάρτηση αυτή δημιουργεί τους νέους extended Huffman πίνακες.

Αποτελέσματα:

Τα αποτελέσματα της κωδικοποίησης ανά γραμμή και δημιουργίας Extended Huffman Tables παρουσιάζονται στον πίνακα 6.5.

	STANDARDENCODER	NEWENCODER1.0		NEWENCODER2.0	
IMAGES	F-18.ppm	F-18.ppm	%	F-18.ppm	%
	SIZE(bytes)	SIZE(bytes)		SIZE(bytes)	
ORIGINAL	230415	230415		230415	
DEFAULT HUFFMAN TABLES WITHOUT HEADERS	118256	114080	-3.5	101231	-14.3
DEFAULT HUFFMAN TABLES WITH HEADERS *	118324	122274	3.3	101682	-14.06
ORIGINAL	230415	230415		230415	
CUSTOMIZED HUFFMAN TABLES WITHOUT HEADERS	107634	98053	-8.9	100818	-6.3
CUSTOMIZED HUFFMAN TABLES WITH HEADERS *	107760	117976	9.4	101621	-5.69
ORIGINAL	230415	230415		230415	
P1	83067	95713	15.2	79010	-4.8
P2	61849	77062	24.5	59638	-3.57
P4	40138	56060	39.6	39432	-1.75

STANDARDENCODER		NEWENCODER1.0		NEWENCODER2.0	
IMAGES	pepers.ppm	pepers.ppm	%	pepers.ppm	%
	SIZE(bytes)	SIZE(bytes)		SIZE(bytes)	
ORIGINAL	786447	786447		786447	
DEFAULT HUFFMAN TABLES WITHOUT HEADERS	490820	480029	-2.1	467563	-4.7
DEFAULT HUFFMAN TABLES WITH HEADERS *	490888	497471	1.34	468199	-4.62
ORIGINAL	786447	786447		786447	
CUSTOMIZED HUFFMAN TABLES WITHOUT HEADERS	476222	458798	-3.6	463519	-2.66
CUSTOMIZED HUFFMAN TABLES WITH HEADERS *	476347	503945	5.7	464335	2.52
ORIGINAL	786447	786447		786447	
P1	405197	434953	7.34	394662	-2.5
P2	332655	363530	9.28	324227	-2.5
P4	189275	224288	18.4	186490	-1.4

Συμπεράσματα:

1. Όσον αφορά το μέγεθος του αρχείου, παρατηρούμε ότι πετυχαίνουμε καλύτερα αποτελέσματα σε όλες τις περιπτώσεις. Αυτό οφείλεται στο γεγονός ότι η κωδικοποίηση γίνεται με 7 πίνακες Huffman. Αυτό συνεπάγεται ότι τα σύμβολα κωδικοποιούνται με λιγότερα bits σε σχέση με τον standardencoder ενώ γράφονται μόνο 7 πίνακες στα αρχεία επικεφαλίδας.
2. Παρόλο που κερδίζουμε σε χωρητικότητα, χάνουμε σε κύκλους μηχανής διότι ξανά για να επιλέξουμε την καλύτερη πρόβλεψη ανά γραμμή, δοκιμάζουμε όλους τους τρόπους πρόβλεψης και δημιουργούμε πίνακες Huffman για να επιλέξουμε τον καλύτερο.
3. Τέλος, παρατηρούμε ότι με την ενεργοποίηση του point transform parameter, μειώνεται η χωρητικότητα του αρχείου.

3.3 Βελτιστοποίηση 3^η (newencoder3.0)

Περιγραφή:

Στις προηγούμενες βελτιστοποιήσεις παρατηρούμε ότι πάντα στα μειονεκτήματα συγκαταλέγεται οι πολλοί κύκλοι μηχανής. Ο λόγος είναι ότι για να επιλέξουμε κάθε φορά την καλύτερη πρόβλεψη ανά γραμμή, πρέπει να δημιουργήσουμε για κάθε πρόβλεψη, πίνακες Huffman και να δούμε σε ποία περίπτωση χρειαζόμαστε λιγότερα bits για την αναπαράσταση των συμβόλων. Στην δεύτερη βελτιστοποίηση δε, εφόσον βρούμε και την καλύτερη πρόβλεψη ανά γραμμή, φτιάχνουμε εκ νέου 7 νέους πίνακες Huffman, έναν για κάθε πρόβλεψη αφού πρώτα έχουμε προσθέσει τα ιστογράμματα με τα σύμβολα για εκείνες τις γραμμές που έχουν την ίδια πρόβλεψη. Το ερώτημα είναι , μπορούμε να βρούμε έναν τρόπο πρόβλεψης της γραμμής έτσι ώστε να μειώνονται οι κύκλοι μηχανής και να πετυχαίνουμε καλύτερο ή τουλάχιστον ίδιο αποτέλεσμα;

Η απάντηση είναι ναι. Ο τρόπος προέρχεται από την αναγνώριση προτύπων (μοτίβου 2). Με λίγα λόγια, η αναγνώριση προτύπων είναι η περιγραφή και κατάταξη των αντικείμενων σε διάφορες κατηγορίες. Τα υπό κατάταξη αντικείμενα καλούνται πρότυπα.

Η αναγνώριση προτύπων και η εύρεση τρόπου πρόβλεψης της κάθε γραμμής σχετίζονται ως εξής: Αρχικά βρίσκουμε ορισμένα στατιστικά στοιχεία για κάθε γραμμή όπως ο μέσος όρος, η διασπορά και ο οριζόντιος, κάθετος και διαγώνιος συσχετισμός. Όποιος από τους τρεις συσχετισμούς είναι μεγαλύτερος, τότε η πρόβλεψη της γραμμής θα γίνει είτε οριζόντια (τρόπος πρόβλεψης 1) εάν ο οριζόντιος είναι μεγαλύτερος από τους άλλους δυο είτε κάθετα (τρόπος πρόβλεψης 2) εάν ο κάθετος συσχετισμός είναι μεγαλύτερος από τους άλλους είτε διαγώνια εάν ο διαγώνιος είναι μεγαλύτερος από τους άλλους 2.

Οριζόντιος συσχετισμός:

$$COF_{horizontal} = \frac{\sum_{i=0}^{n-2} x[i, j] * x[i, j + 1] - mean[i]^2}{(n - 1) * sigma[i]^2}$$

Κάθετος συσχετισμός :

$$COF_{vertical} = \frac{\sum_{i=0}^{n-1} x[i, j] * x[i + 1, j] - mean[i] * mean[i + 1]}{n * sigma[i] * sigma[i + 1]}$$

Οριζόντιος συσχετισμός:

$$COF_{diagonal} = \frac{\sum_{i=0}^{n-2} x[i, j] * x[i + 1, j + 1] - mean[i] * mean[i + 1]}{(n - 1) * sigma[i] * sigma[i + 1]}$$

Sigma:

$$sigma = \sqrt[2]{variation}$$

Variation:

$$variation = \frac{\sum_{i=0}^{n-1} sample^2}{n} - mean[i]^2$$

Mean:

$$variation = \frac{\sum_{i=0}^{n-1} sample}{n}$$

Συναρτήσεις:

Για το σκοπό αυτό, χρησιμοποιήθηκαν οι συναρτήσεις της προηγούμενης βελτιστοποίησης και ακόμη:

```

typedef struct statistical{

    double *sum;//αθροισμα δειγμάτων
    double *sumpower2;//αθροισμα τετραγώνων δειγμάτων
    double *mean;//μέσος όρος
    double *sigmapower2;//διασπορα στο τετραγωνο
    double *sigma;//ριζα διασπορας
    double *cor_h;//οριζοντιος συσχετισμος
    double *cor_v;//καθετος συσχετισμος
    double *cor_diag;//διαγωνιος συσχετισμος
}

1.void find_sums(StatisticalInfo *statcPtr,int k,int columns,int
components)

// StatisticalInfo *statcPtra :pointer to line StatInfo
// StatisticalInfo *statcPtrb :pointer to next line StatInfo
//int k: number of line
//int columns :number of lines
//int components:number of image components

```

Η συνάρτηση αυτή υπολογίζει το αθροισμα των δειγμάτων και το άθροισμα των τετραγώνων των δειγμάτων.

```

2.void find_mean(StatisticalInfo *statcPtr,int k,int columns,int
components)

```

Η συνάρτηση αυτή υπολογίζει το μέσο όρο των δειγμάτων της γραμμής

```

3.void find_sigmapower2(StatisticalInfo *statcPtr,int k,int columns,int
components)

```

Η συνάρτηση αυτή υπολογίζει το τετράγωνο της διασποράς

```

4.void find_sigma(StatisticalInfo *statcPtr,int columns,int components)

```

Η συνάρτηση αυτή υπολογίζει την τετραγωνική ρίζα της διασποράς .

```
5.void find_cor_horizontal(StatisticalInfo *statcPtr, int k, int columns, int components)
```

Η συνάρτηση αυτή υπολογίζει τον οριζόντιο συσχετισμό.

```
6.void find_cor_vertical(StatisticalInfo *statcPtra, StatisticalInfo *statcPtrb, int k, int columns, int components)
```

Η συνάρτηση αυτή υπολογίζει τον κάθετο συσχετισμό.

```
7.void find_cor_diagonal(StatisticalInfo *statcPtra, StatisticalInfo *statcPtrb, int k, int columns, int components)
```

Η συνάρτηση αυτή υπολογίζει τον διαγώνιο συσχετισμό.

Αποτελέσματα:

Τα αποτελέσματα παρουσιάζονται στον πίνακα 6.6.

	STANDARDENCODER	NEWENCODER1.0	NEWENCODER2.0	NEWENCODER3.0
IMAGES	F-18.ppm	F-18.ppm	F-18.ppm	F-18.ppm
	SIZE(bytes)	SIZE(bytes) %	SIZE(bytes) %	SIZE(bytes) %
ORIGINAL	230415	230415	230415	230415
DEFAULT HUFF TABLES WITHOUT HEADERS	118256	114080 -3.5	101231 -14.3	107410 -9.1
DEFAULT HUFF TABLES WITH HEADERS *	118324	122274 3.3	101682 -14.06	107773 -8.9
ORIGINAL	230415	230415	230415	230415
CUSTOMIZED HUFF TABLES WITHOUT HEADERS	107634	98053 -8.9	100818 -6.3	107346 -0.2
CUSTOMIZED HUFF TABLES WITH HEADERS *	107760	117976 9.4	101621 -5.69	107886 0.11
ORIGINAL	230415	230415	230415	230415
P1	83067	95713 15.2	79010 -4.8	82614 -0.5
P2	61849	77062 24.5	59638 -3.57	61488 -0.58
P4	40138	56060 39.6	39432 -1.75	39956 -0.4

		STANDARDENCODER	NEWENCODER1.0		NEWENCODER2.0		NEWENCODER3.0	
IMAGES	pepers.ppm		pepers.ppm	%	pepers.ppm	%	pepers.ppm	%
	SIZE(bytes)		SIZE(bytes)		SIZE(bytes)		SIZE(bytes)	
ORIGINAL	786447	786447		786447		786447		
DEFAULT HUFF TABLES WITHOUT HEADERS	490820	480029	-2.1	467563	-4.7	476251	-2.9	
DEFAULT HUFF TABLES WITH HEADERS *	490888	497471	1.34	468199	-4.62	476887	-2.8	
ORIGINAL	786447	786447		786447		786447		
CUSTOMIZED HUFF TABLES WITHOUT HEADERS	476222	458798	-3.6	463519	-2.66	471202	-1	
CUSTOMIZED HUFF TABLES WITH HEADERS *	476347	503945	5.7	464335	-2.52	472018	-0.9	
ORIGINAL	786447	786447		786447		786447		
P1	405197	434953	7.34	394662	-2.5	401680	-0.86	
P2	332655	363530	9.28	324227	-2.5	329438	-0.96	
P4	189275	224288	18.4	186490	-1.4	190238	0.5	

Επίπεδο κωδικοποίησης

Συμπεράσματα:

1. Παρατηρούμε ότι ναι μεν μειώνονται οι κύκλοι μηχανής, το μέγεθος του αρχείου μειώνεται.
2. Με ενεργοποιημένη την παράμετρο point transform parameter, έχουμε μείωση χωρητικότητας του αρχείου.

Συμπεράσματα και προτάσεις

Συμπεράσματα:

Με το πέρας αυτής της διατριβής διαπιστώνουμε ότι οι τεχνικές βελτιστοποίησης που προτείνουμε όντως πετυχαίνουν καλύτερη απόδοση του lossless JPEG κωδικοποιητή. Βέβαια η κάθε μια από τις τρεις βελτιστοποιήσεις έχει πλεονεκτήματα και μειονεκτήματα. Συνοπτικά

	Κύκλοι μηχανής	Χωρητικότητα	Χωρητικότητα με Pt
Βελτιστοποίηση 1 ^η	αυξάνονται	Αυξάνεται με την εγγραφή των headers, μειώνεται χωρίς.	αυξάνεται
Βελτιστοποίηση 2 ^η	αυξάνονται	μειώνεται	μειώνεται
Βελτιστοποίηση 3 ^η	μειώνονται	μειώνεται	μειώνεται

Προτάσεις:

Για περαιτέρω ενασχόληση με το συγκεκριμένο θέμα, προτείνουμε

- την επέκταση του κώδικα έτσι ώστε να αναγνωρίζονται και άλλοι τύποι εικόνων όπως bmp.
- αποκωδικοποίηση των παραγόμενων αρχείων .ljpg με τις βελτιστοποιήσεις από τον standard JPEG decoder.

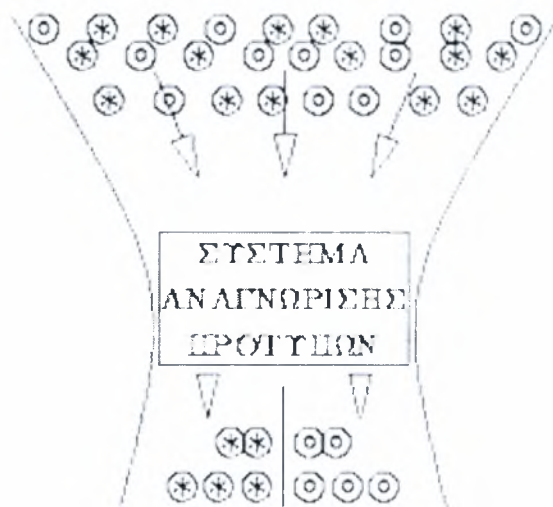
ΠΑΡΑΡΤΗΜΑ Α

Αναγνώριση προτύπων

1. Ορισμός

Η αναγνώριση προτύπων είναι ο επιστημονικός κλάδος που ασχολείται με την περιγραφή και την κατάταξη αντικειμένων σε ένα αριθμό κατηγοριών (εικόνα 1). Τα υπό κατάταξη αντικείμενα καλούνται πρότυπα. Υπάρχουν δύο κατηγορίες αναγνώρισης προτύπων:

- Σαφή πρότυπα (concrete items)
 - χαρακτήρες, εικόνες, αντικείμενα, ήχους.
- Αφηρημένα πρότυπα
 - Λύση ενός μαθηματικού προβλήματος ή ενός φιλοσοφικού επιχειρήματος.



2. Εφαρμογές

Η αναγνώριση προτύπων χρησιμοποιείται σε πολλές εφαρμογές όπως:

- Αναγνώριση ομιλίας.
- Αναγνώριση οπτικών χαρακτήρων
- Αναγνώριση χειρόγραφων κειμένων.
- Ταυτοποίηση ατόμων.
- Ιατρική διάγνωση.
- Γεωγραφικά συστήματα πληροφοριών.
- Βιομηχανικές εφαρμογές.
- Χρηματοοικονομικές Εφαρμογές.
- Εξόρυξη δεδομένων.

3. Μεθοδολογίες αναγνώρισης προτύπων

Υπάρχουν δυο μεθοδολογίες για την αναγνώριση προτύπων.

A. Στατιστική αναγνώριση προτύπων (statistical pattern recognition)

- Εξαγωγή χαρακτηριστικών γνωρισμάτων
- Δημιουργία διανύσματος χαρακτηριστικών στοιχείων
- Μαθηματικές - στατιστικές μέθοδοι, γραμμική άλγεβρα, θεωρία πιθανοτήτων

B. Συντακτική ή δομημένη αναγνώριση προτύπων (syntactic or structural pattern recognition)

- Δημιουργία πολύπλοκων ιεραρχικών περιγραφών.
- Δέντρα αποφάσεων (decision trees), λογικοί κανόνες, γραμματικές.

ΑΝΑΦΟΡΕΣ

1. Recommendation T.81 : Information Technology – Digital Compression and coding of continuous –tone still images – requirements and guidelines
2. Google and Wikipedia searching about
 - Huffman encoding
 - Lossless encoding
 - Jpeg
 - Pattern recognition



ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ



004000091711