



Διπλωματική Εργασία
του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και
Τεχνολογίας Υπολογιστών της Πολυτεχνικής Σχολής του
Πανεπιστημίου Πατρών

Δημητρίου Ιωάννης του Νικόλα

Αριθμός Μητρώου: 227491

Θέμα

**Υλοποίηση Κωδικοποιητή
για το Πρότυπο HEVC**

Επιβλέπων

Αθανάσιος Στουραίτης

Αριθμός Διπλωματικής Εργασίας:

Πάτρα, (Μάρτιος 2016)

ΠΙΣΤΟΠΟΙΗΣΗ

Πιστοποιείται ότι η Διπλωματική Εργασία με θέμα

Υλοποίηση Κωδικοποιητή για το Πρότυπο HEVC

Του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας
Υπολογιστών

Δημητρίου Ιωάννης του Νικόλα

Αριθμός Μητρώου: 227491

Παρουσιάστηκε δημόσια και εξετάστηκε στο Τμήμα Ηλεκτρολόγων
Μηχανικών και Τεχνολογίας Υπολογιστών στις
...../...../.....

Ο Επιβλέπων

Αθανάσιος Στουραίτης

Ο Διευθυντής του Τομέα

Ευθύμιος Χούσος

Αριθμός Διπλωματικής Εργασίας:

Υλοποίηση Κωδικοποιητή για το Πρότυπο HEVC

Φοιτητής:
Ιωάννης Δημητρίου

Επιβλέπων:
Αθανάσιος Στουραίτης

Περίληψη

Στην παρούσα διπλωματική εργασία γίνεται υλοποίηση αλγορίθμων οι οποίοι βρίσκουν εφαρμογή στην συμπίεση κινούμενων εικόνων. Συγκεκριμένα γίνεται υλοποίηση του αλγορίθμου Κινησιακά Αντισταθμισμένης Πρόβλεψης (Motion Compensated Prediction, MCP) και του Διακριτού Μετασχηματισμού Συνημίτονου δύο διαστάσεων (Discrete Cosine Transform, DCT) σύμφωνα με τις προδιαγραφές του προτύπου H.265/ HEVC που περιγράφει την συμπίεση κινούμενων εικόνων. Οι υλοποιήσεις γίνονται σε επίπεδο καταχωρητή (RTL) με χρήση του λογισμικού Simulink της Matlab. Στόχος των υλοποιήσεων είναι η βελτίωση της απόδοσης των συγκεκριμένων αλγορίθμων όσον αφορά τον χρόνο εκτέλεσης, τον ρυθμό επεξεργασίας δεδομένων (Throughput), την κατανάλωση ισχύος και την επιφάνεια ολοκλήρωσης. Για την επίτευξη του στόχου αυτού έγινε χρήση κατά κύριο λόγο των μεθόδων της διασωλήνωσης (pipelining) και της παράλληλης επεξεργασίας (parallel processing).

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 1 : ΕΙΣΑΓΩΓΗ	7
1.1 Η Ανάγκη για Κωδικοποίηση Κινούμενων Εικόνων	7
1.2 Βασικές Τεχνικές Συμπίεσης Εικόνων.....	8
1.3 Το Πρότυπο H.265/HEVC.....	14
1.4 Ζητήματα Υλοποίησης.....	16
ΚΕΦΑΛΑΙΟ 2 : Διακριτός Μετασχηματισμός Συνημίτονου	22
2.1 Θεωρητικό υπόβαθρο.....	22
2.2 Υπάρχουσες Υλοποιήσεις.....	35
2.3 Προτεινόμενη Υλοποίηση.....	36
2.4 Πειραματικές Μετρήσεις	55
ΚΕΦΑΛΑΙΟ 3 : Κινησιακά Αντισταθμισμένη Πρόβλεψη	62
3.1 Θεωρητικό υπόβαθρο.....	62
3.2 Υπάρχουσες Υλοποιήσεις.....	69
3.3 Προτεινόμενη Υλοποίηση.....	70
3.4 Πειραματικές Μετρήσεις	82
ΣΥΜΠΕΡΑΣΜΑΤΑ.....	88
ΒΙΒΛΙΟΓΡΑΦΙΑ.....	90

ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

Σχήμα 1.1: Τυπικό περιβάλλον για κωδικοποίηση εικόνας.....	11
Σχήμα 1.2: Κωδικοποιητής εικόνας.....	12
Σχήμα 1.3 :Block διάγραμμα υβριδικού κωδικοποιητή.....	14
Σχήμα 1.4 : (α) Αρχικό Σύστημα (β) Σύστημα 2-level pipelined.....	17
Σχήμα 1. 5 (α) Αρχικό Σύστημα (β) Σύστημα 2-level parallel processed	19
 Σχήμα 2.1 a) Πίνακας συνδιασποράς block εισόδου, b) Πίνακας συνδιασποράς block μετασχηματισμένου με τον DCT, c) Πίνακας συνδιασποράς block μετασχηματισμένου με τον KLT	29
Σχήμα 2.2 Καμπύλη ενέργειας ανά στοιχείο για σήμα εισόδου πριν και μετά τους μετασχηματισμούς.	30
Σχήμα 2.3 : Γράφημα ροής του IDCT 8 σημείων.....	42
Σχήμα 2.4 : Γράφημα ροής του DCT 8 σημείων.	43
Σχήμα 2. 5: Block διάγραμμα του προτεινόμενου 2D-DCT.	45
Σχήμα 2. 6: Block διάγραμμα του υποσυστήματος DCT 4 σημείων.....	47
Σχήμα 2. 7: Block διάγραμμα του υποσυστήματος DCT 8 σημείων.....	47
Σχήμα 2. 8: Block διάγραμμα του υποσυστήματος DCT 16 σημείων.....	48
Σχήμα 2. 9: Block διάγραμμα του υποσυστήματος DCT 32 σημείων.....	49
Σχήμα 2. 10: Κύκλωμα πολλαπλασιασμού με συγκεκριμένο συντελεστή.....	50
Σχήμα 2. 11: Κύκλωμα υπολογισμού των δεικτών γραμμών και στηλών (i,j).....	53
Σχήμα 2. 12: Βασικό κύκλωμα υπολογισμού τελικής διεύθυνσης.	53
Σχήμα 2.13 : Ολοκληρωμένο κύκλωμα παράλληλου υπολογισμού 32 διευθύνσεων στην ενδιάμεση μνήμη.	54
Γραφική Παράσταση 2.1 : PSNR ανακατασκευής εικόνας προς αριθμό DCT συντελεστών που χρησιμοποιούνται για την ανακατασκευή	56
Γραφική Παράσταση 2.2 : Λόγος συμπίεσης εικόνας προς αριθμό DCT συντελεστών που χρησιμοποιούνται για την ανακατασκευή	57
Γραφική Παράσταση 2.3: Ιστόγραμμα τιμών DCT συντελεστών.....	58
Εικόνα 2.1 : FRAME1	59
Εικόνα 2.2 : FRAME2	60
Εικόνα 2.3 : FRAME3	61
 Σχήμα 3. 1: Σύστημα MCP	63
Σχήμα 3. 2: Εκτίμηση Κίνησης κατά Block	64
Σχήμα 3. 3 : block διάγραμμα του συστήματος MCP	72
Σχήμα 3. 4: Προτεινόμενο κύτταρο SAD	73
Σχήμα 3. 5: 32 level parallel κύκλωμα υπολογισμού SAD	74
Σχήμα 3. 6: Κύκλωμα υπολογισμού BaseAddress	76
Σχήμα 3. 7 : Κύκλωμα πρόσθεσης κάθε πιθανής τιμής του d_{m2} στην BaseAddress.....	77
Σχήμα 3. 8 : Κύκλωμα πρόσθεσης μιας τιμής του d_{n2} στην BaseAddress	78
Σχήμα 3. 9 : Κύκλωμα παράλληλης πρόσθεσης κάθε πιθανής τιμής του d_{n2} στην BaseAddress.....	78
Σχήμα 3.10: Κύκλωμα υπολογισμού των στοιχείων του block σφάλματος.....	80
Σχήμα 3.11: Βασικό κύκλωμα υπολογισμού μιας διεύθυνσης στη μνήμη του τωρινού block και του block πρόβλεψης.....	81
Σχήμα 3.12: Ολοκληρωμένο κύκλωμα παράλληλου υπολογισμού 32 διευθύνσεων στη μνήμη του τωρινού block και του block πρόβλεψης	81

Γραφική Παράσταση 3.1 : PSNR πρόβλεψης για τρία συνεχόμενα πλαίσια	83
Γραφική Παράσταση 3.2 : Λόγος συμπίεσης για τρία συνεχόμενα πλαίσια	84
Εικόνα 3. 1: FRAME1	85
Εικόνα 3. 2: FRAME2	86
Εικόνα 3. 3: FRAME3	87

ΚΕΦΑΛΑΙΟ 1 : ΕΙΣΑΓΩΓΗ

1.1 Η Ανάγκη για Κωδικοποίηση Κινούμενων Εικόνων

Για μια ασπρόμαυρη ψηφιακή εικόνα 512 x 512 εικονοστοιχείων (pixels), όπου κάθε εικονοστοιχείο αναπαριστάται με 8 δυαδικά ψηφία (bits) απαιτούνται 2 Mbits πληροφορίας για την αναπαράσταση της, ενώ για μια έγχρωμη εικόνα 1000 X 1000 εικονοστοιχείων με 24 bits για κάθε εικονοστοιχείο, χρειαζόμαστε 3 Mbytes. Στην περίπτωση, εικόνων τηλεόρασης συμβατικής ευκρίνειας, όπου σε κάθε δευτερόλεπτο εμφανίζονται 30 στιγμιότυπα (frames) της εικόνας, ο ρυθμός δυαδικών ψηφίων ανέρχεται σε 180 Mbits/sec. Ο ρυθμός αυτός γίνεται 1 Gbit/sec, για εικόνες τηλεόρασης υψηλής ευκρίνειας (High Definition TV -HDTV). Είναι προφανές λοιπόν, ότι σε εφαρμογές, που απαιτείται αποθήκευση ή μετάδοση ενός τόσο μεγάλου πλήθους δεδομένων, γίνεται επιτακτική η ανάγκη συμπίεσης των δεδομένων, καθώς με την συμπίεση επιτυγχάνουμε :

- την μείωση του εύρους ζώνης συχνοτήτων που απαιτείται για την μετάδοση του video, αφού στον ίδιο χρόνο θα πρέπει να μεταδοθεί μικρότερο μέγεθος πληροφορίας.
- την μείωση του χώρου στη μνήμη που απαιτείται για την αποθήκευση του video στο σύστημα.

1.2 Βασικές Τεχνικές Συμπίεσης Εικόνων

Στην ενότητα αυτή, παρουσιάζονται μερικές από τις σπουδαιότερες τεχνικές συμπίεσης ακίνητων ψηφιακών εικόνων. Σκοπός της συμπίεσης ψηφιακών εικόνων (image compression) είναι η αναπαράσταση της εικόνας με όσο το δυνατόν λιγότερα δυαδικά ψηφία, διατηρώντας όμως το επίπεδο της ποιότητας και καταληπτότητας, που απαιτούνται για κάθε εφαρμογή. Ο όρος "συμπίεση ψηφιακών εικόνων" είναι ισοδύναμος με τον όρο "κωδικοποίηση ψηφιακών εικόνων" (image coding). Η κωδικοποίηση ψηφιακών εικόνων περιλαμβάνει δύο μεγάλες περιοχές εφαρμογών. Η μία περιοχή αφορά τη μείωση του εύρους ζώνης του καναλιού, που απαιτείται σε συστήματα μετάδοσης της εικόνας. Οι εφαρμογές μετάδοσης συμπεριλαμβάνουν τη ψηφιακή τηλεόραση, τις δορυφορικές επικοινωνίες, την τηλεδιάσκεψη (teleconferencing), την τηλεομοιοτυπία (facsimile). Η δεύτερη περιοχή αφορά την αποθήκευση (storage) πληροφορίας. Η αποθήκευση εικόνων βρίσκει εφαρμογές σε πολυμέσα (multimedia), δεδομένα εικόνων από δορυφορικά προγράμματα, δεδομένα video. Τα επιτρεπτά επίπεδα ποιότητας και καταληπτότητας μιας εικόνας διαφέρουν ανάλογα με την εφαρμογή. Σε εφαρμογές, όπως η αποθήκευση δεδομένων εικόνας από δορυφορικά προγράμματα ή εικόνων αντικειμένων με μεγάλη ιστορική αξία, συχνά είναι επιθυμητή η διατήρηση όλης της πληροφορίας, που έχει η αρχική εικόνα. Οι τεχνικές συμπίεσης, που επιτρέπουν την πιστή ανακατασκευή της αρχικής εικόνας, ονομάζονται τεχνικές χωρίς απώλεια πληροφορίας κατά τη συμπίεση (lossless compression). Σε μερικές εφαρμογές όμως, όπως η ψηφιακή τηλεόραση δεν απαιτείται η διατήρηση όλης της πληροφορίας της αρχικής εικόνας. Μπορεί σε τέτοιες εφαρμογές να θέλουμε εικόνα υψηλής ποιότητας, αλλά αρκετή πληροφορία της αρχικής εικόνας μπορεί να απολεστεί, χωρίς να γίνεται αυτό αντιληπτό από το ανθρώπινο μάτι. Σε εφαρμογές όπως τα RPVs (remote-piloted vehicles), η καταληπτότητα της εικόνας είναι πολύ σημαντική, αλλά είναι δυνατόν να είναι ανεκτή η απώλεια ενός ποσού πληροφορίας. Όσο μικρότερου βαθμού ποιότητας και καταληπτότητας μπορεί να είναι η εικόνα σε κάθε εφαρμογή, τόσο μικρότερος είναι ο απαιτούμενος ρυθμός δυαδικών ψηφίων

(bit rate) της εφαρμογής. Ο ρυθμός δυαδικών ψηφίων μετράται συνήθως με τον αριθμό δυαδικών ψηφίων ανά εικονοστοιχείο. Οι τεχνικές της δεύτερης περιοχής εφαρμογών ονομάζονται τεχνικές συμπίεσης εικόνας με απώλεια πληροφορίας (lossy compression). Οι τεχνικές, που δεν παρουσιάζουν απώλεια πληροφορίας, κατά τη συμπίεση της εικόνας βασίζονται στο γεγονός ότι ο ρυθμός των δεδομένων μιας εικόνας είναι γενικά μεγαλύτερος από το ρυθμό της πληροφορίας της εικόνας.

Για μια πηγή με n διακριτά σύμβολα με πιθανότητες

$$p_i, i = 1, \dots, n$$

ο μέσος ρυθμός πληροφορίας δίνεται από την εντροπία:

$$H(X) = \sum_{i=1}^n p_i \log_2 \left(\frac{1}{p_i} \right) = - \sum_{i=1}^n p_i \log_2 p_i,$$

Σύμφωνα με το θεώρημα του Shannon, μια πηγή εντροπίας H μπορεί να κωδικοποιηθεί χωρίς απώλειες, χρησιμοποιώντας $H + e$ bits κατά μέσο όρο ανά σύμβολο, όπου e οποιοσδήποτε θετικός αριθμός. Ο λόγος συμπίεσης, C , που ορίζεται σαν:

$$C = \frac{n}{n_\alpha}$$

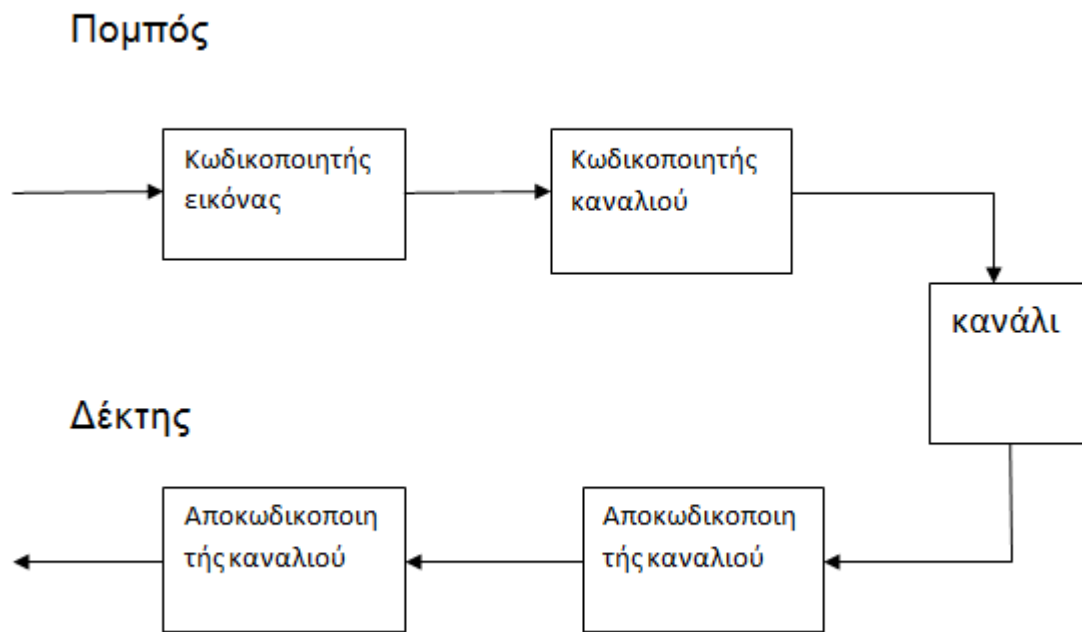
όπου n και n_α οι ρυθμοί δυαδικών ψηφίων των αρχικών και συμπιεσμένων δεδομένων αντίστοιχα, δίνεται από τη σχέση:

$$C = \frac{B}{B+e} = \frac{B}{H}$$

όπου B ο μέσος αριθμός δυαδικών ψηφίων ανά εικονοστοιχείο της αρχικής εικόνας. Γενικά, οι τεχνικές αυτής της κατηγορίας πετυχαίνουν λόγους συμπίεσης από 3 έως 5. Οι τεχνικές συμπίεσης με απώλεια πληροφορίας προσπαθούν να εκμεταλλευτούν

κάποιες ιδιότητες του αισθητηρίου της όρασης στο ανθρώπινο μάτι και να αφαιρέσουν πληροφορία χωρίς αυτή να γίνεται αντιληπτή. Για παράδειγμα μικρές απώλειες της χρωματικής πληροφορίας γίνονται δύσκολα αντιληπτές από το ανθρώπινο μάτι σε αντίθεση με τη θόλωση των περιγραμμάτων και το τρεμόπαιγμα (flickering) της εικόνας, που δημιουργούν κακή οπτική εντύπωση. Η απόδοση των τεχνικών συμπίεσης με απώλεια πληροφορίας μετράται με το λόγο συμπίεσης, την παραμόρφωση που προκαλούν και την υπολογιστική τους πολυπλοκότητα. Οι τεχνικές συμπίεσης με απώλεια πληροφορίας πετυχαίνουν λόγους συμπίεσης 40 έως 100 και χρησιμοποιούνται στις περισσότερες βιομηχανικές και εμπορικές εφαρμογές. Η παραμόρφωση μετράται είτε με το λόγο σήματος προς θόρυβο (Signal to Noise Ratio - SNR), είτε με το μέγιστο λόγο σήματος προς θόρυβο (Peak Signal to Noise Ratio - PSNR).

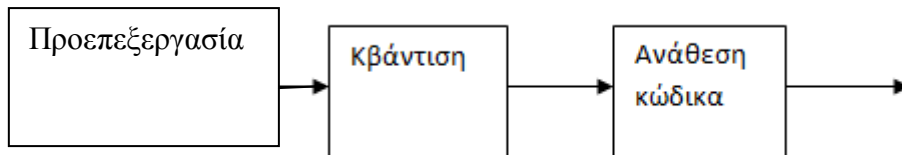
Στο σχήμα 1.1, δίνεται ένα τυπικό περιβάλλον για τη συμπίεση εικόνας. Η εικόνα αρχικά κωδικοποιείται με έναν κωδικοποιητή εικόνας (image coder), που ονομάζεται επίσης και κωδικοποιητής πηγής (source coder). Η έξοδος του κωδικοποιητή εικόνας είναι μια σειρά από δυαδικά ψηφία (string of bits), τα οποία και αναπαριστούν την αρχική εικόνα. Ο κωδικοποιητής καναλιού (channel coder), στη συνέχεια, μετασχηματίζει μέσω ενός τύπου διαμόρφωσης, τη σειρά των δυαδικών ψηφίων σε μια σειρά, που είναι πιο κατάλληλη για μετάδοση μέσα από ένα τηλεπικοινωνιακό κανάλι (communication channel). Το διαμορφωμένο σήμα μεταδίδεται στο τηλεπικοινωνιακό κανάλι. Το τηλεπικοινωνιακό κανάλι εισάγει θόρυβο και μέσα στο κανάλι δίνεται η δυνατότητα για διόρθωση του θορύβου.



Σχήμα 1.1: Τυπικό περιβάλλον για κωδικοποίηση εικόνας

Στο δέκτη, το σήμα που λαμβάνεται αποδιαμορφώνεται (demodulated) και απεικονίζεται πάλι σε μια σειρά δυαδικών ψηφίων από τον αποκωδικοποιητή καναλιού (channel decoder). Ο αποκωδικοποιητής εικόνας (image decoder) ανακατασκευάζει την εικόνα από τη σειρά των δυαδικών ψηφίων. Σε εφαρμογές αποθήκευσης δεδομένων εικόνας στο σχήμα 1.1, δεν υπάρχει τηλεπικοινωνιακό κανάλι και η σειρά των δυαδικών ψηφίων μετά τον αποκωδικοποίηση εικόνας, αποθηκεύεται σε κάποιο μέσο.

Ο κωδικοποίησης εικόνας του σχήματος 1.1 αποτελείται από τρία τμήματα, όπως φαίνεται στο σχήμα 1.2. Το *πρώτο* τμήμα, που είναι και το πιο σημαντικό είναι η προεπεξεργασία της εικόνας, ώστε αυτή να είναι πιο κατάλληλη για κβάντιση (quantization) και ανάθεση κώδικα (codeword assignment).



Σχήμα 1.2: Κωδικοποιητής εικόνας

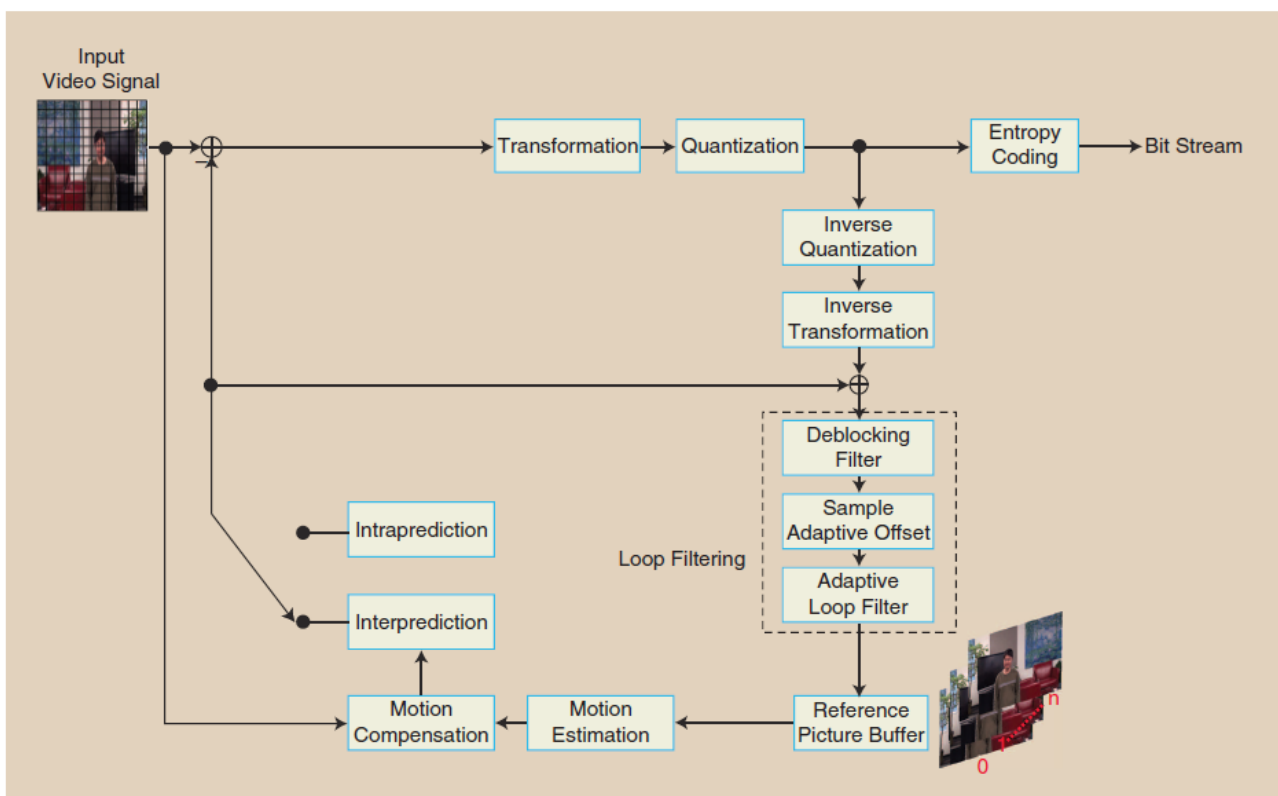
Το τμήμα προεπεξεργασίας του κωδικοποιητή είναι αυτό που καθορίζει τη μορφή πληροφορίας που κωδικοποιείται. Με κριτήριο τη μορφή των δεδομένων μετά το πρώτο στάδιο του κωδικοποιητή εικόνας, τα συστήματα κωδικοποίησης ταξινομούνται σε τρεις ευρείες κατηγορίες. Στην πρώτη κατηγορία, ανήκουν οι κωδικοποιητές κυματομορφής (waveform coders), όπου κωδικοποιείται είτε η ίδια η ένταση (intensity) της εικόνας είτε η διαφορά των εντάσεων μεταξύ ενός εικονοστοιχείου ή μιας ομάδας εικονοστοιχείων (συνήθως μιας ορθογώνιας περιοχής) της εικόνας με μια πρόβλεψη που γίνεται από τον κωδικοποιητή για τις τιμές αυτές. Η πρόβλεψη αυτή μπορεί να βασίζεται είτε σε γειτονικά εικονοστοιχεία της ίδιας εικόνας (Intra-prediction), είτε, αν η συμπίεση αφορά κινούμενες εικόνες, σε εικονοστοιχεία μιας ή περισσότερων προηγούμενων ή και επόμενων εικόνων (Inter-prediction). Στη δεύτερη κατηγορία ανήκουν οι κωδικοποιητές με χρήση μετασχηματισμών (transform coders). Σε αυτή την κατηγορία κωδικοποιητών, η εικόνα μετασχηματίζεται σε ένα πεδίο συχνοτήτων, όπως με το μετασχηματισμό Fourier ή με το μετασχηματισμό συνημίτονου, ή μετασχηματίζεται σε ένα πεδίο χρόνου-συχνότητας, όπως με το μετασχηματισμό κυματιδίου (wavelet). Τέλος, στην τρίτη κατηγορία κωδικοποιητών ανήκουν οι υβριδικοί κωδικοποιητές, οι οποίοι αποτελούν συνδυασμό των δύο προηγούμενων κατηγοριών, όπου συνήθως η διαφορά των αρχικών εικονοστοιχείων από την πρόβλεψη που κάνει ο κωδικοποιητής υφίσταται στη συνέχεια κάποιο μετασχηματισμό, όπως αυτοί που αναφέρθηκαν παραπάνω. Η απόδοση και η πολυπλοκότητα των διαφόρων κωδικοποιητών εικόνας ποικίλει κάθε φορά και εξαρτάται κυρίως από τη μορφή των δεδομένων που κωδικοποιούνται. Το δεύτερο τμήμα του κωδικοποιητή του σχήματος 1.2 αποτελεί η κβάντιση. Προκειμένου να αναπαρασταθεί η εικόνα με όσο το δυνατόν λιγότερα δυαδικά ψηφία, οι εντάσεις των εικονοστοιχείων (image

intensities) ή οι συντελεστές των μετασχηματισμών (transform coefficients) κβαντίζονται. Η κβάντιση μπορεί να είναι κβάντιση βαθμωτού μεγέθους (scalar quantization) ή κβάντιση διανύσματος (vector quantization). Η κβάντιση βαθμωτού μεγέθους είναι πολύ γνωστή σε εφαρμογές τηλεπικοινωνιακών συστημάτων. Η διανυσματική κβάντιση υπερτερεί της κβάντισης βαθμωτού μεγέθους για την κωδικοποίηση δεδομένων που έχουν στατιστική συσχέτιση. Η κβάντιση περιλαμβάνει ανάθεση των επιπέδων ανακατασκευής (reconstruction or quantization levels) και των ορίων απόφασης (decision boundaries). Στο τρίτο τμήμα του κωδικοποιητή εικόνας γίνεται η ανάθεση των κωδίκων που είναι μια σειρά από δυαδικά ψηφία, που αναπαριστούν τα επίπεδα κβάντισης. Οι τεχνικές ανάθεσης κωδίκων ταξινομούνται σε δύο κατηγορίες. Στην πρώτη κατηγορία οι κώδικες έχουν σταθερό μήκος (ίδιο αριθμό δυαδικών ψηφίων) (uniform length codeword assignment). Στη δεύτερη κατηγορία γίνεται ανάθεση λέξεων μικρότερου μήκους σε μηνύματα που έχουν μεγαλύτερη πιθανότητα εμφάνισης και λέξεων μεγαλύτερου μήκους σε μηνύματα με μικρότερη πιθανότητα εμφάνισης. Καθένα από τα τρία τμήματα του κωδικοποιητή προσπαθεί να εκμεταλλευτεί τον πλεονασμό της πληροφορίας στα δεδομένα της εικόνας και τους περιορισμούς, που τίθενται είτε από το σύστημα απεικόνισης της εικόνας (display device) είτε από το ανθρώπινο μάτι (human visual system). Τα τρία μέρη ενός κωδικοποιητή εικόνας σχετίζονται άμεσα μεταξύ τους. Για παράδειγμα, αν ο μετασχηματισμός στον κωδικοποιητή εικόνας αποσυσχετίζει σε μεγάλο βαθμό τα δεδομένα, μπορεί η τεχνική της διανυσματικής κβάντισης να μην υπερτερεί σημαντικά έναντι της κβάντισης βαθμωτού μεγέθους. Επίσης, αν τα επίπεδα κβάντισης επιλεχθούν έτσι ώστε κάθε επίπεδο να χρησιμοποιείται με ίδια πιθανότητα, τότε το κέρδος που μπορεί να επιτευχθεί χρησιμοποιώντας κώδικες μεταβλητού μήκους μπορεί να μην είναι μεγάλο σε σχέση με τους κώδικες σταθερού μήκους.

1.3 Το Πρότυπο H.265/HEVC

Το Πρότυπο High Efficiency Video Coding (HEVC) αποτελεί το πιο πρόσφατο πρότυπο για την συμπίεση κινούμενων εικόνων. Ακολουθεί κι αυτό, όπως και όλα τα πιο πρόσφατα προηγούμενα πρότυπα, την υβριδική προσέγγιση, δηλαδή εμπεριέχει συνδυασμό αλγορίθμων κωδικοποίησης κυματομορφής και κωδικοποίησης μετασχηματισμού.

Παρακάτω ακολουθεί ένα τυπικό block διάγραμμα υβριδικού κωδικοποιητή που μπορεί να δημιουργήσει μια ακολουθία από bits στην έξοδό του, η οποία να ακολουθεί τις προδιαγραφές του HEVC.



Σχήμα 1.3 :Block διάγραμμα υβριδικού κωδικοποιητή

Ως είσοδος στον κωδικοποιητή ορίζεται μία ακολουθία από εικόνες σε YUV format, που σημαίνει ότι κάθε εικόνα αναπαρίσταται από έναν πίνακα, οι τιμές των στοιχείων του οποίου δηλώνουν την φωτεινότητα για κάθε εικονοστοιχείο της εικόνας, και άλλους δύο πίνακες, οι οποίοι περιέχουν την χρωματική πληροφορία για κάθε εικονοστοιχείο.

Τυπικά ο αλγόριθμος κωδικοποίησης ακολουθεί τα παρακάτω βήματα:

Κάθε εικόνα χωρίζεται σε ορθογώνιες παραλληλόγραμμες περιοχές (blocks) από εικονοστοιχεία, και ο ακριβής τρόπος διαχωρισμού αποστέλλεται στον αποκωδικοποιητή. Η πρώτη εικόνα μιας ακολουθίας κινούμενων εικόνων (video) κωδικοποιείται με χρήση Intra-prediction. Για όλες τις υπόλοιπες εικόνες του video χρησιμοποιείται τυπικά Inter-prediction, πλην εξαιρέσεων. Η διαδικασία κωδικοποίησης με πρόβλεψη Inter-prediction περιλαμβάνει την επιλογή του κατάλληλου block εικονοστοιχείων, μέσα από κάποια γειτονική εικόνα αναφοράς, το οποίο θα χρησιμοποιηθεί για την πρόβλεψη καθώς και τον υπολογισμό του block σφάλματος. Η διεύθυνση του κατάλληλου block που επελέγη, το λεγόμενο διάνυσμα κίνησης (Motion Vector) στέλνεται στον αποκωδικοποιητή. Έτσι ο αποκωδικοποιητής, αφού λάβει το block σφάλματος, μπορεί από το διάνυσμα κίνησης να βρει το block αναφοράς και τέλος να εξάγει το αρχικό block.

Μετά την δημιουργία του block σφάλματος στην πλευρά του κωδικοποιητή, εφαρμόζεται σε αυτό ένας μετασχηματισμός. Στη συνέχεια, τα μετασχηματισμένα δείγματα κβαντίζονται, κωδικοποιούνται με κάποιο αλγόριθμο κωδικοποίησης εντροπίας και αποστέλλονται στον αποκωδικοποιητή.

Για να αποφευχθεί η συσσώρευση σφαλμάτων κατά την διαδικασία της Inter-prediction πρόβλεψης, είναι απαραίτητο οι εικόνες αναφοράς στην πλευρά του κωδικοποιητή να είναι ακριβή αντίγραφα των αποκωδικοποιημένων εικόνων στον αποκωδικοποιητή. Για αυτό είναι απαραίτητη η χρήση ενός βρόχου αποκωδικοποίησης στον κωδικοποιητή. Έτσι, τα κβαντισμένα δείγματα υφίστανται

μια διαδικασία αντίστροφης κβάντισης και αντίστροφου μετασχηματισμού. Στη συνέχεια, προστίθεται σε αυτά το block πρόβλεψης που δημιουργήθηκε κατά τη διαδικασία Inter-prediction, για να δημιουργηθεί το αρχικό block, το οποίο και αποθηκεύεται σε μια μνήμη, στην ίδια θέση που κατείχε και στην αρχική εικόνα, έτσι ώστε στο τέλος να έχουμε πλήρη την αρχική εικόνα. Τέλος, η εικόνα αυτή, πριν αποθηκευτεί στην μνήμη αποκωδικοποιημένων εικόνων, είναι απαραίτητο να διέλθει από μια συστοιχία φίλτρων, ώστε να εξομαλυνθούν οι ασυνέχειες που εμφανίστηκαν στα όρια των blocks λόγω της τμηματοποίησης της εικόνας και της ανεξάρτητης κωδικοποίησης κάθε τμήματος. Το σύνολο των διαδικασιών αντίστροφης κβάντισης, αντίστροφου μετασχηματισμού, πρόσθεσης του block πρόβλεψης και φιλτραρίσματος είναι ακριβές αντίγραφο των διαδικασιών που λαμβάνουν χώρα στον αποκωδικοποιητή.

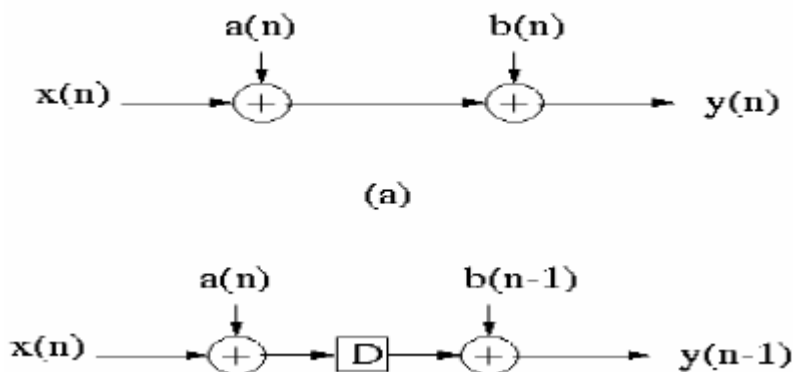
1.4 Ζητήματα Υλοποίησης

Η υλοποίηση αλγορίθμων συμπίεσης εικόνας για εφαρμογές πραγματικού χρόνου, οδηγεί σε τεράστιες υπολογιστικές απαιτήσεις που μπορούν να αντιμετωπιστούν με συστήματα, που αποτελούνται από επεξεργαστές ειδικών εφαρμογών (ASICs) με την τεχνική VLSI. Η έρευνα για την υλοποίηση αποδοτικών συστημάτων εστιάζεται πέρα από την ανάπτυξη γρήγορων αλγορίθμων συμπίεσης εικόνας και στην ανάπτυξη μεθοδολογιών για την αποδοτική σχεδίαση αρχιτεκτονικών, που υλοποιούν τέτοιους αλγορίθμους. Οι βασικότερες μεθοδολογίες που οδηγούν σε αύξηση του ρυθμού επεξεργασίας δεδομένων του συστήματος, καθώς και μείωση της καταναλισκόμενης ισχύος είναι οι τεχνικές διασωλήνωσης (pipelining) και παράλληλης επεξεργασίας.

1.4.1 Pipelining

Η συνεισφορά της μεθόδου γίνεται κατανοητή μέσω του ακόλουθου παραδείγματος:

Έστω τα δύο ακόλουθα συστήματα επεξεργασίας σήματος:



Σχήμα 1.4 : (α) Αρχικό Σύστημα (β) Σύστημα 2-level pipelined

Τα δύο αυτά συστήματα δημιουργούν στην έξοδό τους ακριβώς την ίδια ακολουθία $y(n)$, μόνο που στο δεύτερο σύστημα αυτή καθυστερεί κατά μία χρονική στιγμή, εξαιτίας του ενδιάμεσου καταχωρητή. Ωστόσο, για να δεχτεί το πρώτο σύστημα καινούργια είσοδο $x(n)$, πρέπει πρώτα να έχουν ολοκληρωθεί και οι δύο προσθέσεις, που σημαίνει ότι ο ελάχιστος κύκλος ρολογιού είναι ίσος με $2T_{add}$, όπου T_{add} ο μέγιστος χρόνος που χρειάζεται ένας αθροιστής για να υπολογίσει σωστό αποτέλεσμα. Αντίθετα, στο δεύτερο σύστημα ο χρόνος ρολογιού μπορεί να μειωθεί μέχρι έως και χρόνο T_{add} , καθώς το ενδιάμεσο αποτέλεσμα αποθηκεύεται στον καταχωρητή και δεν επηρεάζεται από αλλαγές στην έξοδο του πρώτου αθροιστή, παρά μόνο αφού περάσει ένας κύκλος ρολογιού. Έτσι, για την επεξεργασία N δειγμάτων το πρώτο σύστημα χρειάζεται χρόνο $2NT_{add}$, ενώ το δεύτερο $(N+1)T_{add}$. Λέμε ότι το σύστημα υπέστη διασωλήνωση 2 επιπέδων (2-level pipelined). Στην

περίπτωση αυτή, το κρίσιμο μονοπάτι (critical path) του συστήματος, δηλαδή η μέγιστη καθυστέρηση μεταξύ δύο καταχωρητών, μειώθηκε στο μισό.

Συνεπώς, η μέθοδος pipelining συνίσταται στην προσθήκη επιπλέον καταχωρητών μεταξύ των λειτουργικών μονάδων του συστήματος, με στόχο την μείωση της κρίσιμης καθυστέρησης. Με τον τρόπο αυτόν λοιπόν η διαδικασία pipelining αυξάνει τον ρυθμό επεξεργασίας του συστήματος. Ωστόσο, στην περίπτωση που δεν απαιτείται τόσο μεγάλος ρυθμός επεξεργασίας ή τα υπόλοιπα συστήματα δεν μπορούν να πετύχουν τόσο μεγάλους ρυθμούς, τότε η διαδικασία pipelining μπορεί να συνδράμει στην μείωση της κατανάλωσης ισχύος του συστήματος.

Ως γνωστόν, για την τεχνολογία CMOS, που είναι και η πιο συχνά χρησιμοποιούμενη, ο χρόνος υπολογισμού ενός δείγματος εξόδου για έναν κύκλωμα με συνολική χωρητικότητα κρίσιμου μονοπατιού C_{charge} , τάση τροφοδοσίας V_0 , τάση κατωφλίου V_t και σταθερά k δίνεται από

$$T_{pd} = \frac{C_{charge} \cdot V_0}{k(V_0 - V_t)^2}$$

ενώ η κατανάλωση ισχύος από

$$P_{CMOS} = C_{total} \cdot V_0^2 \cdot f$$

Σε ένα M-level pipelined σύστημα, η μείωση του critical path συνεπάγεται την μείωση της χωρητικότητας που πρέπει να φορτιστεί ή να εκφορτιστεί σε κάθε κύκλο ρολογιού κατά M φορές, συνεπώς αν δεν επιθυμούμε την μείωση του χρόνου T_{pd} ,

μπορούμε να μειώσουμε την τάση τροφοδοσίας V_0 κατά έναν παράγοντα β , με $\beta < 1$. Έτσι, αν βV_0 η νέα τάση τροφοδοσίας, τότε η κατανάλωση ισχύος γίνεται

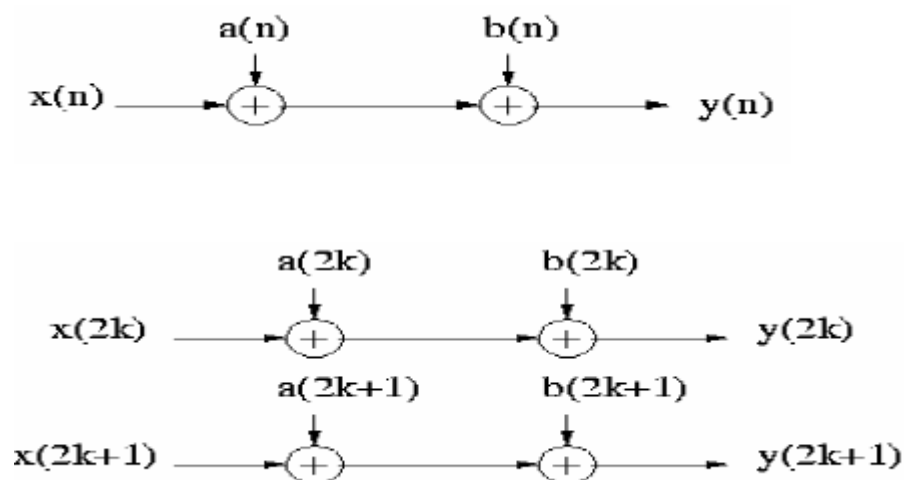
$$P_{pip} = C_{total} \cdot \beta^2 \cdot V_0^2 \cdot f = \beta^2 \cdot P_{seq}$$

όπου $\beta^2 < \beta < 1 \Rightarrow P_{pip} < P_{seq}$

και επιτυγχάνεται μείωση της κατανάλωσης.

1.4.2 Parallel Processing

Η διαδικασία αυτή συνίσταται στην δημιουργία ενός ή περισσότερων αντιγράφων του αρχικού συστήματος, έτσι ώστε να είναι εφικτή η επεξεργασία περισσότερων του ενός δειγμάτων του σήματος εισόδου κάθε χρονική στιγμή. Παρακάτω φαίνεται ένα παράδειγμα εφαρμογής της μεθόδου:



Σχήμα 1. 5 (α) Αρχικό Σύστημα (β) Σύστημα 2-level parallel processed

Φαίνεται λοιπόν ότι το δεύτερο σύστημα μπορεί στον ίδιο χρόνο με το πρώτο να δέχεται δύο δείγματα εισόδου και να υπολογίζει δύο δείγματα εξόδου. Συνεπώς και σε αυτή την περίπτωση έχουμε αύξηση του ρυθμού επεξεργασίας και συγκεκριμένα διπλασιασμό του, με τίμημα βέβαια τον διπλασιασμό της επιφάνειας ολοκλήρωσης. Ισοδύναμα, μπορούμε να πούμε ότι το σύστημα έχει υποστεί παραλληλισμό 2 επιπέδων (2-level parallel). Και εδώ, στην περίπτωση που ο αρχικός ρυθμός επεξεργασίας είναι αρκετός για τις απαιτήσεις της εφαρμογής, είναι εφικτό να επιτευχθεί αντ' αυτού μείωση της καταναλισκόμενης ισχύος.

Εφόσον πλέον σε κάθε κύκλο ρολογιού το παραλληλισμένο σύστημα επεξεργάζεται περισσότερα δείγματα (έστω L φορές περισσότερα), τότε για να μείνει αμετάβλητος ο αρχικός ρυθμός επεξεργασίας πρέπει η χρονική διάρκεια του κύκλου ρολογιού να αυξηθεί κατά L φορές. Συνεπώς από την εξίσωση

$$T_{pd} = \frac{C_{charge} \cdot V_0}{k(V_0 - V_t)^2}$$

συνεπάγεται ότι αύξηση του T_{pd} L φορές επιτρέπει την μείωση της τάσης τροφοδοσίας κατά έναν παράγοντα β . Ακόμα, εξαιτίας της δημιουργίας L αντιγράφων του αρχικού συστήματος, η συνολική χωρητικότητα που φορτίζεται ή εκφορτίζεται σε έναν κύκλο ρολογιού αυξήθηκε L φορές, ενώ η συχνότητα λειτουργίας καθενός από τα αντίγραφα μειώθηκε εξίσου. Συνεπώς τελικά η κατανάλωση του παράλληλου συστήματος γίνεται

$$\begin{aligned} P_{par} &= (LC_{charge})(\beta V_0)^2 \frac{f}{L} \\ &= \beta^2 C_{charge} V_0^2 f \\ &= \beta^2 P_{seq}, \end{aligned}$$

Τέλος, συνδυάζοντας τις δύο τεχνικές, αν στο ίδιο σύστημα εφαρμοστεί pipelining M επιπέδων και parallel processing L επιπέδων, τότε από τη σχέση που συνδέει αρχικό και τελικό χρόνο ρολογιού

$$LT_{\text{sec}} = \frac{(C_{\text{ch arg e}} / M) \cdot \beta V_0}{k (\beta V_0 - V_t)^2} = \frac{L \cdot C_{\text{ch arg e}} \cdot V_0}{k (V_0 - V_t)^2}$$

έχουμε ότι επιτυγχάνεται μείωση της κατανάλωσης ισχύος κατά β^2 , όπου το β υπολογίζεται από τη σχέση

$$LM \cdot (\beta V_0 - V_t)^2 = \beta (V_0 - V_t)^2$$

ΚΕΦΑΛΑΙΟ 2 : ΔΙΑΚΡΙΤΟΣ ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ ΣΥΝΗΜΙΤΟΝΟΥ

2.1 Θεωρητικό υπόβαθρο

Η διαδικασία του μετασχηματισμού περιγράφεται ως εξής:

Έστω $s(n_1, n_2)$ ένα block δεδομένων (στην περίπτωση μας, το block σφάλματος). Ο μετασχηματισμός του σε ένα block μετασχηματισμένων δειγμάτων $S(k_1, k_2)$ συνίσταται στον υπολογισμό του παρακάτω αθροίσματος :

$$S(k_1, k_2) = \sum_{n_1=0}^{n-1} \sum_{n_2=0}^{n-1} s(n_1, n_2) g(n_1, n_2, k_1, k_2) \quad k_1, k_2 = 0, 1, \dots, n-1.$$

όπου n η διάσταση του block εισόδου s , και $g(n_1, n_2, k_1, k_2)$ η συνάρτηση μετασχηματισμού. Η παραπάνω εξίσωση γράφεται ισοδύναμα σε μορφή πινάκων :

$$\mathbf{S} = \mathbf{A} \mathbf{s}$$

όπου $\mathbf{A}$ ένας τετραγωνικός πίνακας $n^2 \times n^2$ στοιχείων, που περιέχει τις τιμές της συνάρτησης g , και $\mathbf{s}, \mathbf{S}$ διανύσματα n^2 στοιχείων, που περιέχουν τα στοιχεία των αντιστοίχων blocks διατεταγμένα κατά γραμμές.

Για κάθε $g(n_1, n_2, k_1, k_2)$, απαιτούμε να υπάρχει συνάρτηση $h(n_1, n_2, k_1, k_2)$ για την οποία να ισχύει:

$$s(n_1, n_2) = \sum_{k_1=0}^{n-1} \sum_{k_2=0}^{n-1} S(k_1, k_2) h(n_1, n_2, k_1, k_2) \quad n_1, n_2 = 0, 1, \dots, n-1$$

Όστε να είναι εφικτή η ανακατασκευή του αρχικού block. Η συνάρτηση $h(n_1, n_2, k_1, k_2)$ ονομάζεται συνάρτηση του αντιστρόφου μετασχηματισμού. Αντίστοιχα σε μορφή πινάκων η εξίσωση γράφεται:

$$\mathbf{s} = \mathbf{A}^{-1} \mathbf{S}$$

όπου $\mathbf{A}^{-1}$ ο αντίστροφος του πίνακα $\mathbf{A}$.

Το στάδιο του μετασχηματισμού σε ένα σύστημα κωδικοποίησης κινούμενων εικόνων, έχει ως στόχο την μεταφορά του block σφάλματος σε κάποιο άλλο πεδίο, στο οποίο θα μπορεί να περιγραφεί με μικρότερο όγκο πληροφορίας. Γι αυτό ο μαθηματικός μετασχηματισμός που θα επιλεγεί θα πρέπει να παρουσιάζει σε όσο το δυνατόν μεγαλύτερο βαθμό τις παρακάτω ιδιότητες:

- a) Τα μετασχηματισμένα δείγματα θα πρέπει να είναι όσο το δυνατόν ασυσχέτιστα. Έτσι αφαιρείται η πλεονάζουσα πληροφορία που υπήρχε στο μη μετασχηματισμένο πεδίο και επιτυγχάνεται συμπίεση.
- b) Το μεγαλύτερο μέρος της ενέργειας του σήματος θα πρέπει να συμπυκνώνεται σε όσο το δυνατόν λιγότερα δείγματα. Στην περίπτωση αυτή τα υπόλοιπα δείγματα

μπορεί να θεωρηθούν μηδενικά και να αγνοηθούν, χωρίς να υπεισέρχονται σημαντικά σφάλματα.

Παρακάτω θα υπολογίσουμε τον πίνακα μετασχηματισμού A , ο οποίος ικανοποιεί με βέλτιστο τρόπο τις δύο παραπάνω ιδιότητες:

α) Θεωρώντας το block εισόδου ως ένα στοχαστικό σήμα, η συσχέτιση μεταξύ των δειγμάτων του σήματος δίνεται από την συνάρτηση συνδιασποράς:

$$COV_s[n_1, n_2] = E[s(n_1) - E[s(n_1)]] E[s(n_2) - E[s(n_2)]] =$$

$$= E[s(n_1)s(n_2)] - E[s(n_1)] E[s(n_2)] \quad n_1, n_2 = 0, 1 \dots n^2 - 1$$

Αντίστοιχα, σε μορφή πινάκων η παραπάνω εξίσωση γράφεται :

$$COV_s = E[\mathbf{s} \mathbf{s}^T] - \mathbf{m}_s \mathbf{m}_s^T$$

όπου $\mathbf{s}^T$ το ανάστροφο του διανύσματος $\mathbf{s}$ και

$\mathbf{m}_s = E[\mathbf{s}]$ το διάνυσμα μέσων τιμών των δειγμάτων του $\mathbf{s}$

Για να είναι τα στοιχεία του διανύσματος $\mathbf{S} = \mathbf{A}\mathbf{s}$, που θα προκύψει από τον μετασχηματισμό, ασυσχέτιστα μεταξύ τους, απαιτείται :

$$COV_S [n_1, n_2] = 0 \quad \text{για } n_1 \neq n_2$$

Με άλλα λόγια, απαιτείται ο πίνακας συνδιασποράς του $\mathbf{S}$ να είναι διαγώνιος. Ο πίνακας αυτός, COV_S , μπορεί να εκφραστεί μέσω του πίνακα συνδιασποράς του αρχικού διανύσματος, COV_s , ως

$$COV_S = E[SS^T] - m_s m_s^T$$

Δεδομένου ότι

$$S = As$$

$$m_s = E[As] = A E[s] = A m_s$$

έχουμε ότι :

$$COV_S = E[As (As)^T] - A m_s (A m_s)^T =$$

$$= E[A ss^T A^T] - A m_s m_s^T A^T =$$

$$= A(E[ss^T] - m_s m_s^T) A^T = A COV_s A^T$$

Γνωρίζουμε από την γραμμική άλγεβρα ότι, για κάθε πραγματικό και συμμετρικό πίνακα, έστω X , υπάρχει ορθογώνιος πίνακας, έστω Y , του οποίου οι γραμμές είναι τα ορθοκανονικά ιδιοδιανύσματα του X , τέτοιος ώστε :

$$Y X Y^T = \text{diag}(\lambda_1, \lambda_2, \dots)$$

όπου $\lambda_1, \lambda_2, \dots$ οι ιδιοτιμές του πίνακα X .

Λόγω της αντιμεταθετικής ιδιότητας:

$$COV_s[n_1, n_2] = E[s(n_1) - E[s(n_1)]] E[s(n_2) - E[s(n_2)]] =$$

$$E[s(n_2) - E[s(n_2)]] E[s(n_1) - E[s(n_1)]] = COV_s[n_2, n_1]$$

Αποδεικνύεται ότι κάθε πίνακας συνδιασποράς πραγματικού σήματος είναι πραγματικός και συμμετρικός.

Συνεπώς, αν ο πίνακας A επιλεχτεί ώστε οι γραμμές του να είναι τα ορθοκανονικά ιδιοδιανύσματα του πίνακα COV_s , τότε ο πίνακας

$$COV_s = A COV_s A^T$$

θα είναι διαγώνιος, και σε κάθε θέση της κύριας διαγωνίου του θα βρίσκονται οι ιδιοτιμές του πίνακα COV_s , διατεταγμένες με την ίδια σειρά που διατάχθηκαν τα ιδιοδιανύσματα στον πίνακα A .

b) Ως γνωστόν, η ενέργεια ενός στοχαστικού σήματος μεγέθους n^2 ορίζεται ως:

$$E_s = \sum_{n_1=0}^{n^2-1} E[(s(n_1) - m_s(n_1))^2] = E[(s - m_s)(s - m_s)^T]$$

Για το σήμα

$$S = As$$

η ενέργεια γίνεται :

$$E_s = \sum_{i=0}^{n^2-1} E[(S(i) - m_s(i))^2] = \sum_{i=0}^{n^2-1} E[(a_i s - a_i m_s)(a_i s - a_i m_s)^T]$$

όπου i ο δείκτης γραμμής του πίνακα A και a_i η i -οστή του γραμμή

Ακολουθώντας :

$$E_s = \sum_{i=0}^{n^2-1} E[a_i (s - m_s)(s - m_s)^T a_i^T] =$$

$$\sum_{i=0}^{n^2-1} \mathbf{a}_i E[(\mathbf{s} - \mathbf{m}_s)(\mathbf{s} - \mathbf{m}_s)^T] \mathbf{a}_i^T = \sum_{i=0}^{n^2-1} \mathbf{a}_i \text{COV}_s \mathbf{a}_i^T$$

Ψάχνουμε λοιπόν διανύσματα $\mathbf{a}_i$, τέτοια ώστε αν κρατήσουμε ένα υποσύνολο των μετασχηματισμένων δειγμάτων, έστω $M < n^2$, η ενέργεια του σήματος που θα διατηρηθεί θα είναι μέγιστη.

Αποδεικνύεται ([1]) πως και σε αυτή την περίπτωση, η βέλτιστη επιλογή για τα $\mathbf{a}_i$ είναι τα ορθοκανονικά ιδιοδιανύσματα του πίνακα COV_s .

Ο μετασχηματισμός που χρησιμοποιεί τα παραπάνω αποτελέσματα ονομάζεται Karhunen-Loeve Transform (KLT) και η υλοποίηση του περιλαμβάνει τις παρακάτω διαδικασίες:

Για κάθε νέο block εισόδου :

- 1) Υπολογισμός του πίνακα συνδιασποράς του block εισόδου
- 2) Υπολογισμός των ορθοκανονικών ιδιοδιανυσμάτων του παραπάνω πίνακα
- 3) Υπολογισμός των μετασχηματισμένων δειγμάτων
- 4) Αποστολή τόσο των μετασχηματισμένων δειγμάτων, όσο και του πίνακα μετασχηματισμού στον αποκωδικοποιητή.

Παρά το γεγονός ότι ο KLT είναι βέλτιστος μετασχηματισμός, όσον αφορά την αποσυσχέτιση των δειγμάτων του σήματος εισόδου και την συμπίκνωση της ενέργειας, εντούτοις παρουσιάζει κάποια πολύ σημαντικά μειονεκτήματα :

a) Είναι εξαρτημένος από τα δεδομένα (data dependent). Αυτό σημαίνει ότι για κάθε καινούργιο block εισόδου, απαιτείται να υπολογιστεί καινούργιος πίνακας μετασχηματισμού και να αποσταλεί και αυτός, εκτός από τα μετασχηματισμένα

δεδομένα, στον αποκωδικοποιητή, αφού αυτός δεν γίνεται να τον υπολογίσει μόνος του. Τόσο ο επιπλέον χρόνος που απαιτείται για τον υπολογισμό του πίνακα μετασχηματισμού, όσο και η ανάγκη για αποστολή του επηρεάζουν σημαντικά την απόδοση του αλγορίθμου.

b) Δεν υπάρχουν αποδοτικοί αλγόριθμοι για την υλοποίηση του KLT, με συνέπεια να απαιτούνται $O(n^4)$ αριθμητικές πράξεις για την υλοποίηση του μέσω του ορισμού. Ο αυξημένος αυτός αριθμός πράξεων που απαιτείται επηρεάζει σημαντικά την απόδοση του αλγορίθμου, τόσο αναφορικά με τον χρόνο εκτέλεσης, όσο και αναφορικά με την επιφάνεια ολοκλήρωσης, αν μιλάμε για υλοποιήσεις σε υλικό.

Τα παραπάνω μειονεκτήματα καθιστούν τον KLT μη εφαρμόσιμο σε εφαρμογές πραγματικού χρόνου. Γι' αυτό υπήρξε η ανάγκη χρήσης εναλλακτικών αλγορίθμων, ικανών να προσεγγίζουν την βέλτιστη απόδοση του, όσον αφορά την αποσυσχέτιση και την συμπύκνωση της ενέργειας, ενώ ταυτόχρονα να ικανοποιούν και τους πρακτικούς περιορισμούς (χρόνος εκτέλεσης, κατανάλωση ισχύος, επιφάνεια ολοκλήρωσης, ρυθμός επεξεργασίας).

Ο μετασχηματισμός που αποδείχτηκε ότι συγκεντρώνει τα παραπάνω στον μεγαλύτερο βαθμό είναι ο δισδιάστατος Διακριτός Μετασχηματισμός Συνημίτονου (2D DCT). Ο μετασχηματισμός αυτός, για block εισόδου διαστάσεων N, M δίνεται από τον τύπο:

$$X(k, l) = \left(\frac{2}{NM}\right)^{\frac{1}{2}} c_k c_l \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x(n, m) \cos\left(\frac{(2n+1)k\pi}{2N}\right) \cos\left(\frac{(2m+1)l\pi}{2M}\right)$$

όπου $k = 0, 1, \dots, N-1$ και $l = 0, 1, \dots, M-1$.

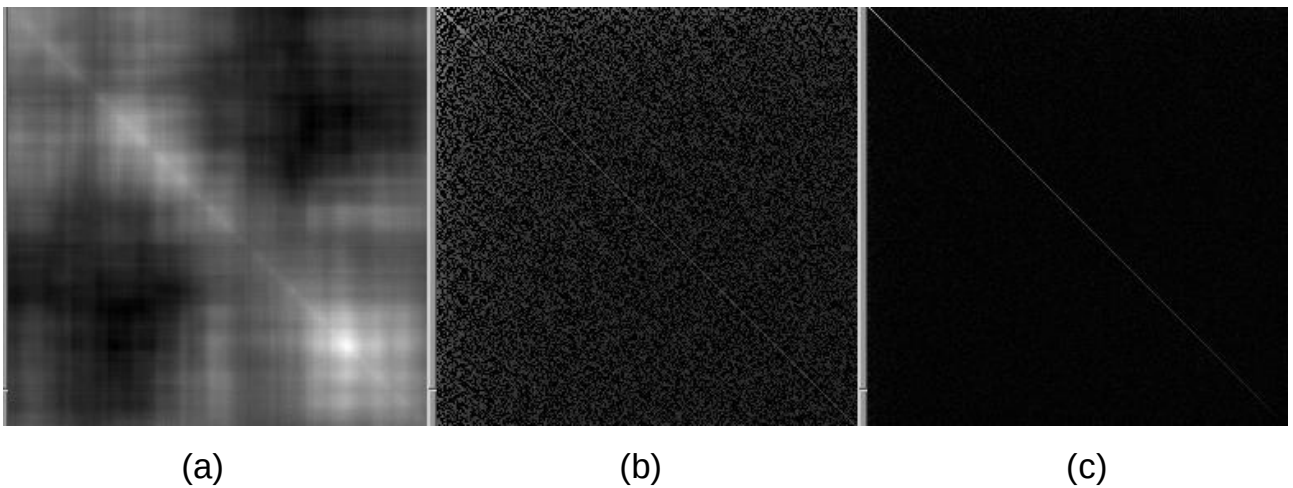
Ενώ ο αντίστροφος μετασχηματισμός:

$$x(n, m) = \left(\frac{2}{NM}\right)^{\frac{1}{2}} c_n c_m \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} X(k, l) \cos\left(\frac{(2n+1)k\pi}{2N}\right) \cos\left(\frac{(2m+1)l\pi}{2M}\right),$$

όπου $n = 0, 1, \dots, N-1$ και $m = 0, 1, \dots, M-1$.

$$c_k, c_l, c_m, c_n = \begin{cases} \frac{1}{\sqrt{2}} & , k, l, m, n = 0 \\ 1 & , \text{αλλού,} \end{cases}$$

Όσον αφορά τις δυνατότητες συμπίεσης του DCT, αποδεικνύεται ότι, τα ιδιοδιανύσματα του πίνακα συνδιασποράς ενός διανύσματος με υψηλή συσχέτιση μεταξύ των στοιχείων του είναι συνημιτονοειδείς συναρτήσεις. Συνεπώς οι συναρτήσεις βάσης του DCT προσεγγίζουν σε μεγάλο βαθμό τα ιδιοδιανύσματα τέτοιων blocks, και ο DCT δύναται να συμπίεσει τέτοια block σχεδόν όσο και ο KLT. Το γεγονός αυτό φαίνεται και από τα σχήματα 2.1 και 2.2:



Σχήμα 2.1 α) Πίνακας συνδιασποράς block εισόδου, β) Πίνακας συνδιασποράς block μετασχηματισμένου με τον DCT, γ) Πίνακας συνδιασποράς block μετασχηματισμένου με τον KLT

Στο σχήμα 2.1 φαίνεται ο πίνακας συνδιασποράς για ένα σήμα εισόδου, καθώς και ο αντίστοιχος πίνακας μετά την εφαρμογή των μετασχηματισμών DCT (b) και KLT (c). Παρατηρούμε ότι, με τον μετασχηματισμό KLT, όλα τα στοιχεία εκτός της κύριας διαγωνίου έχουν μηδενιστεί, συνεπώς το σήμα έχει αποσυσχετιστεί πλήρως. Την ίδια στιγμή, με τον DCT το σήμα δεν έχει αποσυσχετιστεί πλήρως μεν, καθώς πολλά στοιχεία εκτός της κύριας διαγωνίου είναι μη μηδενικά, εντούτοις οι τιμές τους έχουν μειωθεί σε μεγάλο βαθμό σε σχέση με τις αρχικές, που σημαίνει ότι τα στοιχεία αποσυσχετίστηκαν σε μεγάλο βαθμό.



Σχήμα 2.2 Καμπύλη ενέργειας ανά στοιχείο για σήμα εισόδου πριν και μετά τους μετασχηματισμούς.

Στο σχήμα 2.2 φαίνεται η συμπίκνωση της ενέργειας που προκαλούν οι παραπάνω μετασχηματισμοί. Η οριζόντια γραμμή δείχνει την κατανομή της ενέργειας του σήματος εισόδου, ενώ η υπερβολές την κατανομή της ενέργειας μετά τους μετασχηματισμούς. Βλέπουμε ότι, ενώ πριν το μετασχηματισμό, η ενέργεια κατανέμεται σχεδόν ομοιόμορφα σε όλα τα στοιχεία του σήματος, μετά τον

μετασχηματισμό το μεγαλύτερο μέρος της ενέργειας του σήματος συγκεντρώνεται σε πολύ λίγα στοιχεία.

Όσον αφορά τους πρακτικούς περιορισμούς, ο DCT αντιμετωπίζει σε μεγάλο βαθμό τα μειονεκτήματα του KLT καθώς:

a) Είναι μετασχηματισμός ανεξάρτητος από τα δεδομένα (data independent), που σημαίνει ότι ο πίνακας μετασχηματισμού είναι σταθερός και δεν απαιτείται ο υπολογισμός του για κάθε block εισόδου ούτε και η αποστολή του στον αποκωδικοποιητή.

b) Είναι διαχωρίσιμος, αφού η συνάρτηση μετασχηματισμού γράφεται ως

$$g(n_1, n_2, k_1, k_2) = \cos\left[\frac{(2n_1+1)k_1\pi}{2n}\right] \cos\left[\frac{(2n_2+1)k_2\pi}{2n}\right] = \\ = g_1(n_1, k_1) g_2(n_2, k_2)$$

Στην περίπτωση αυτή είναι δυνατή η χρήση του αλγορίθμου αποσύνθεσης γραμμών-στηλών (rows-columns decomposition), για την υλοποίηση του μετασχηματισμού. Σύμφωνα με αυτόν, ο δισδιάστατος μετασχηματισμός μπορεί να αντικατασταθεί από η μονοδιάστατους κατά μήκος των γραμμών του αρχικού block, και κατόπιν άλλους η μονοδιάστατους κατά μήκος των στηλών. Ο αλγόριθμος αυτός επιτρέπει την μείωση των απαιτούμενων αριθμητικών πράξεων από $O(n^4)$ σε $O(2n^3)$.

c) Για τον υπολογισμό του μονοδιάστατου DCT έχουν αναπτυχθεί πολλοί αλγόριθμοι, οι οποίοι μειώνουν την αριθμητική πολυπλοκότητα από $O(n^2)$ σε $O(n \log n)$. Έτσι, συνυπολογίζοντας και την συνεισφορά της ιδιότητας της διαχωρισιμότητας, η συνολική πολυπλοκότητα της υλοποίησης του 2D DCT πέφτει σε $O(n^2 \log n)$.

Οι γρήγοροι αλγόριθμοι για τον υπολογισμό του μονοδιάστατου DCT μπορούν να ταξινομηθούν στις παρακάτω κατηγορίες:

α) Αλγόριθμοι έμμεσου υπολογισμού (indirect algorithms), οι οποίοι υπολογίζουν τους συντελεστές DCT μέσω άλλων μετασχηματισμών. Τέτοιοι μετασχηματισμοί είναι ο διακριτός μετασχηματισμός Fourier, ο Walsh-Hadamard Transform (WHT) και ο Discrete Hadamard Transform (DHT).

β) Αλγόριθμοι στους οποίους οι συντελεστές του DCT υπολογίζονται άμεσα (direct algorithms), χρησιμοποιώντας τεχνικές όπως η παραγοντοποίηση της μήτρας των συντελεστών και ο αποδεκατισμός στο χρόνο και στη συχνότητα. Στην κατηγορία αυτή ανήκουν επίσης οι αλγόριθμοι επαναληπτικής δομής, οι αλγόριθμοι περιστροφών και οι αλγόριθμοι διάσπασης βάσης (split-radix).

α) Αλγόριθμοι έμμεσου υπολογισμού

i) Υπολογισμός του DCT μέσω του FFT

Οι αλγόριθμοι της κατηγορίας αυτής απεικονίζουν την αρχική ακολουθία n σημείων σε μία $(2n - 1)$ σημείων ακολουθία ή σε δύο ακολουθίες n σημείων και υπολογίζουν το μετασχηματισμό Fourier αυτών. Η ακολουθία του DCT προκύπτει μετά από μία βαθμίδα επεξεργασίας, στην οποία εκτελούνται οι αριθμητικές πράξεις πολλαπλασιασμού και άθροισης/αφαίρεσης. Παρέχουν το πλεονέκτημα της εκτέλεσης του αλγορίθμου σε αρχιτεκτονικές Fast Fourier Transform (FFT) αλγορίθμων. Στην κατηγορία αυτή υπάγονται και οι αλγόριθμοι επαναληπτικής δομής οι οποίοι χρησιμοποιούν FFT αλγορίθμους. Παρουσιάζουν αυξημένη

ταχύτητα εκτέλεσης και η πολυπλοκότητα τους εξαρτάται από το μετασχηματισμό απεικόνισης των δεικτών.

ii) Υπολογισμός του DCT μέσω του WHT μετασχηματισμού

Οι αλγόριθμοι της κατηγορίας αυτής οδηγούν σε αποδοτικές γενικά υλοποιήσεις, στις οποίες απαιτείται η αποθήκευση της μήτρας που συνδέει τους δύο μετασχηματισμούς. Οι αλγόριθμοι αυτοί δεν προτιμούνται για ακολουθίες με περισσότερα από 16 σημεία και η υπολογιστική τους πολυπλοκότητα εξαρτάται από τον αλγόριθμο υπολογισμού του Walsh-Hadamard Transform.

iii) Υπολογισμός του DCT μέσω του DHT μετασχηματισμού

Η υπολογιστική τους πολυπλοκότητα είναι ανάλογη με αυτή των αλγορίθμων που βασίζονται στον FFT. Για την υλοποίηση τους μπορούν να χρησιμοποιηθούν οι δομές για τον υπολογισμό του FFT, γενικά όμως χαρακτηρίζονται ως χαμηλής ταχύτητας αλγόριθμοι. Όπως οι αλγόριθμοι της προηγούμενης κατηγορίας, οδηγούν σε αποδοτικές γενικά υλοποιήσεις, στις οποίες απαιτείται η αποθήκευση της μήτρας που συνδέει τους δύο μετασχηματισμούς.

β) Αλγόριθμοι άμεσου υπολογισμού

i) Αλγόριθμοι αποδεκατισμού στο χρόνο

Οι αλγόριθμοι αποδεκατισμού στο χρόνο (Decimation in Time, DIT) είναι όλοι επαναληπτικής δομής και βασίζονται στη μέθοδο που προτάθηκε από τους Vetterli και Nussbaumer. Στην πρώτη βαθμίδα του αλγορίθμου επιτυγχάνεται μία αλλαγή της

διάταξης των σημείων της ακολουθίας εισόδου, στη δεύτερη βαθμίδα ο n σημείων μετασχηματισμός αντιστοιχίζεται σε μετασχηματισμούς $n/2$ σημείων και η διαδικασία συνεχίζεται προσδίδοντας επαναληπτική δομή στους αλγορίθμους. Μειονέκτημα αποτελεί η σχετικά δύσκολη τεχνική απεικόνισης των δεικτών που χρησιμοποιείται.

ii) Αλγόριθμοι αποδεκατισμού στη συχνότητα

Οι αλγόριθμοι αποδεκατισμού στην συχνότητα (Decimation in Frequency, DIF), όπως και αυτοί της προηγούμενης κατηγορίας είναι όλοι τους επαναληπτικής δομής και βασίζονται στη μέθοδο που προτάθηκε από τους Vetterli και Nussbaumer. Στην πρώτη βαθμίδα του αλγορίθμου επιτυγχάνεται μία αλλαγή της διάταξης των σημείων της ακολουθίας εισόδου, η οποία έχει ως αποτέλεσμα την ομαδοποίηση των συντελεστών της ακολουθίας του DCT σε αυτούς που έχουν άρτιους και σε αυτούς που έχουν περιττούς δείκτες. Παρουσιάζουν το ίδιο μειονέκτημα της δυσκολίας στην απεικόνιση των δεικτών.

iii) Αλγόριθμοι παραγοντοποίησης της μήτρας των συντελεστών

Ο υπολογισμός του DCT επιτυγχάνεται με αποδεκατισμό της μήτρας των συντελεστών σε μήτρες απλούστερης μορφής, ο οποίος προσδίδει επαναληπτική δομή στον αλγόριθμο. Ενώ δεν παρουσιάζει το μειονέκτημα της δύσκολης απεικόνισης των δεικτών, είναι αλγόριθμος σχετικά χαμηλής ταχύτητας.

iv) Αλγόριθμοι με χρήση τεχνικών περιστροφής

Στους αλγορίθμους της κατηγορίας αυτής ο DCT υπολογίζεται με χρήση τεχνικών περιστροφής - ολίσθησης (Planar Rotations). Αντικαθιστούν δηλαδή τις αθροίσεις και τους πολλαπλασιασμούς με περιστροφές/ολισθήσεις. Ενώ, η βασική μονάδα επεξεργασίας των αλγορίθμων έχει πολύ απλή δομή και οι απεικονίσεις των δεικτών

είναι εύκολες στην υλοποίηση τους, οι αλγόριθμοι παρουσιάζουν τη χαμηλότερη ταχύτητα εκτέλεσης σε σχέση με όλους τους άλλους αλγορίθμους.

2.2 Υπάρχουσες Υλοποιήσεις

Στο πρότυπο HEVC καθορίζεται η εκμετάλλευση της ιδιότητας της διαχωρισιμότητας του 2D-DCT με τον αλγόριθμο διαχωρισμού γραμμών-στηλών (row-column decomposition), κι έτσι το πρόβλημα υπολογισμού του δισδιάστατου μετασχηματισμού ανάγεται στον υπολογισμό $2n$ μονοδιάστατων, όπου n η διάσταση του Block εισόδου. Ως εκ τούτου, η πλειονότητα των υπαρχουσών υλοποιήσεων μελετά αποκλειστικά την βελτίωση της απόδοσης του DCT μίας διάστασης.

Σε κάθε μια από τις εργασίες [3],[5],[6],[11],[13]-[16], γίνεται χρήση τριών βασικών τεχνικών βελτίωσης της απόδοσης του 1D-DCT:

a) Διαχωρισμός του πίνακα μετασχηματισμού διάστασης n σε δύο υποπίνακες διάστασης $n/2$. Ο ένας περιέχει τις γραμμές με άρτιο δείκτη του αρχικού πίνακα, ενώ ο δεύτερος αυτές με περιττό δείκτη. Η τεχνική αυτή κάνει χρήση της κατακόρυφης συμμετρίας των στοιχείων του πίνακα μετασχηματισμού. Επειδή ο πίνακας με τις άρτιες γραμμές είναι ένας DCT $n/2$ σημείων, είναι εφικτή η αναδρομική εφαρμογή της τεχνικής αυτής. Με τον τρόπο αυτόν η αριθμητική πολυπλοκότητα μειώνεται, όπως αναφέρεται, από $O(n^2)$ με τον ορισμό σε $O(n^2/4)$, δηλαδή κατά 75%.

b) Υλοποίηση των πολλαπλασιασμών με απλές προσθέσεις και ολισθήσεις. Η συγκεκριμένη πρακτική εκμεταλλεύεται το γεγονός ότι οι συντελεστές του μετασχηματισμού είναι σταθεροί και προϋπολογισμένοι, συνεπώς η υλοποίηση τους είναι εφικτό να βελτιστοποιηθεί κατά περίπτωση.

c) Χρήση πολυπλεκτών σε κάθε στάδιο του μετασχηματισμού, ώστε, στις περιπτώσεις που απαιτείται, αντί για μετασχηματισμό n σημείων να εκτελούνται δύο μετασχηματισμοί $n/2$ σημείων παράλληλα από το ήδη υπάρχον υλικό. Στις συγκεκριμένες υλοποιήσεις χρησιμοποιείται μια επιπρόσθετη μονάδα μετασχηματισμού $n/2$ σημείων σε κάθε στάδιο του μετασχηματισμού, καθώς ο πίνακας με τις περιπτές γραμμές δεν ταυτίζεται με τον DCT $n/2$ σημείων. Η υλοποίηση αυτή εκτιμήθηκε ότι, ενώ αυξάνει τον ρυθμό επεξεργασίας του συστήματος κατά $32/n$ φορές, αυξάνει ωστόσο σημαντικά την επιφάνεια ολοκλήρωσης, όπως και την κατανάλωση ενέργειας, κατά 50%, λόγω της επιπρόσθετης μονάδας μετασχηματισμού $n/2$ σημείων σε κάθε στάδιο του μετασχηματισμού.

Στις εργασίες [8],[10] εφαρμόζεται μια παραγοντοποίηση του πίνακα μετασχηματισμού, κομμάτι της οποίας είναι ο πίνακας του μετασχηματισμού Hadamard, ο οποίος υλοποιείται χωρίς πολλαπλασιασμούς, καθώς τα στοιχεία του είναι είτε 1 είτε -1. Πρόκειται για έναν έμμεσο αλγόριθμο υπολογισμού του DCT. Το απαραίτητο υλικό για μια τέτοια υλοποίηση, έστω 16 σημείων, αναφέρεται ότι είναι 232 αθροιστές, που είναι ελαφρώς μεγαλύτερη από αυτές των γρήγορων άμεσων αλγορίθμων υπολογισμού του μετασχηματισμού συνημίτονου, αν υποθέσουμε ότι οι πολλαπλασιασμοί σε αυτούς υλοποιούνται με προσθέσεις και ολισθήσεις, όπως περιγράφηκε παραπάνω .

2.3 Προτεινόμενη Υλοποίηση

Η υλοποίηση έγινε σε επίπεδο καταχωρητή (RTL) με χρήση του εργαλείου Simulink του πακέτου Matlab, και βασίστηκε στον αλγόριθμο διαχωρισμού γραμμών-στηλών (row-column decomposition), ο οποίος εκμεταλλεύεται την ιδιότητα της διαχωρισιμότητας του δυσδιάστατου Διακριτού Μετασχηματισμού Συνημίτονου και ανάγει το πρόβλημα υπολογισμού του δισδιάστατου μετασχηματισμού στον υπολογισμό $2N$ μονοδιάστατων, όπου N η διάσταση του Block εισόδου. Για την

υλοποίηση του μονοδιάστατου Διακριτού Μετασχηματισμού Συνημίτονου βασίστηκε στον αλγόριθμο αποδεκατισμού στην συχνότητα (Decimation in Frequency, DIF), ο οποίος περιγράφεται ως εξής:

Αρχικά ξεκινώντας από τον ορισμό του αντίστροφου μετασχηματισμού (IDCT)

$$x(n) = \frac{2}{N} \sum_{k=0}^{N-1} e(k)X(k) \cos\left[\frac{(2n+1)k\pi}{2N}\right], \quad n = 0, 1, \dots, N-1,$$

$$e(k) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } k = 0, \\ 1, & \text{otherwise.} \end{cases}$$

Χάριν απλότητας αμελούμε τον παράγοντα βάθμωσης $2/N$, θέτουμε

$$\hat{X}(k) = e(k)X(k)$$

και αφού χωρίσουμε το άθροισμα σε άρτιους και περιττούς όρους ως προς k , η σχέση γίνεται

$$x(n) = \sum_{k=0}^{N-1} \hat{X}(k) \cos \frac{(2n+1)\pi k}{2N}$$

$$\begin{aligned}
&= \sum_{k=0}^{N/2-1} \hat{X}(2k) \cos \frac{(2n+1)\pi(2k)}{2N} \\
&+ \sum_{k=0}^{N/2-1} \hat{X}(2k+1) \cos \frac{(2n+1)\pi(2k+1)}{2N} \\
&= \sum_{k=0}^{N/2-1} \hat{X}(2k) \cos \frac{(2n+1)\pi k}{2(N/2)} \\
&+ \frac{1}{2 \cos \frac{(2n+1)\pi}{2N}} \sum_{k=0}^{N/2-1} 2\hat{X}(2k+1) \cos \frac{(2n+1)\pi(2k+1)}{2N} \cos \frac{(2n+1)\pi}{2N}.
\end{aligned}$$

Κάνοντας χρήση της ιδιότητας

$$2 \cos \frac{(2n+1)\pi(2k+1)}{2N} \cos \frac{(2n+1)\pi}{2N} = \cos \frac{(2n+1)\pi(k+1)}{N} + \cos \frac{(2n+1)\pi k}{N}.$$

Έχουμε ότι

$$\begin{aligned}
&\sum_{k=0}^{N/2-1} 2\hat{X}(2k+1) \cos \frac{(2n+1)\pi(2k+1)}{2N} \cos \frac{(2n+1)\pi}{2N} \\
&= \sum_{k=0}^{N/2-1} \hat{X}(2k+1) \cos \frac{(2n+1)\pi(k+1)}{N} + \sum_{k=0}^{N/2-1} \hat{X}(2k+1) \cos \frac{(2n+1)\pi k}{N} \\
&= \sum_{k=0}^{N/2-2} \hat{X}(2k+1) \cos \frac{(2n+1)\pi(k+1)}{N} + \sum_{k=0}^{N/2-1} \hat{X}(2k+1) \cos \frac{(2n+1)\pi k}{N}.
\end{aligned}$$

Αφού

$$\cos \frac{(2n+1)\pi(N/2-1+1)}{N} = 0$$

Θέτοντας $k'=k+1$ στο πρώτο άθροισμα έχουμε:

$$\begin{aligned} \sum_{k=0}^{N/2-2} \hat{X}(2k+1) \cos \frac{(2n+1)\pi(k+1)}{N} \\ = \sum_{k'=1}^{N/2-1} \hat{X}(2k'-1) \cos \frac{(2n+1)\pi k'}{N} \\ = \sum_{k'=0}^{N/2-1} \hat{X}(2k'-1) \cos \frac{(2n+1)\pi k'}{N} \end{aligned}$$

Έτσι ο IDCT γράφεται :

$$\begin{aligned} x(n) &= \sum_{k=0}^{N/2-1} \hat{X}(2k) \cos \frac{(2n+1)\pi k}{2(N/2)} \\ &+ \frac{1}{2 \cos \frac{(2n+1)\pi}{2N}} \sum_{k=0}^{N/2-1} [\hat{X}(2k+1) + \hat{X}(2k-1)] \cos \frac{(2n+1)\pi k}{2(N/2)} \end{aligned}$$

Τέλος θέτουμε:

$$\begin{aligned} G(k) &= \hat{X}(2k), \\ H(k) &= \hat{X}(2k+1) + \hat{X}(2k-1), \quad \text{for } k = 0, 1, \dots, N/2-1 \end{aligned}$$

Και:

$$\begin{aligned}
g(n) &= \sum_{k=0}^{N/2-1} \hat{X}(2k) \cos \frac{(2n+1)\pi k}{2(N/2)}, \\
h(n) &= \sum_{k=0}^{N/2-1} [\hat{X}(2k+1) + \hat{X}(2k-1)] \cos \frac{(2n+1)\pi k}{2(N/2)}, \\
&\text{for } n = 0, 1, \dots, N/2 - 1.
\end{aligned}$$

Φαίνεται λοιπόν ότι οι συναρτήσεις $G(k)$ και $H(k)$ αποτελούν τους DCT των ακολουθιών $g(n)$ και $h(n)$ αντίστοιχα, συνεπώς ισχύει:

$$\begin{aligned}
x(n) &= g(n) + \frac{1}{2 \cos \frac{(2n+1)\pi}{2N}} h(n), \\
x(N-1-n) &= g(n) - \frac{1}{2 \cos \frac{(2n+1)\pi}{2N}} h(n), \\
&\text{for } n = 0, 1, \dots, N/2 - 1
\end{aligned}$$

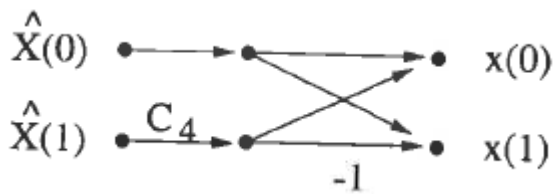
Αφού:

$$\begin{aligned}
\cos \frac{(2(N-1-n)+1)\pi k}{N} &= \cos \frac{(2n+1)\pi k}{N}, \\
\cos \frac{(2(N-1-n)+1)\pi}{2N} &= -\cos \frac{(2n+1)\pi}{2N}.
\end{aligned}$$

Με τον τρόπο αυτόν, ο IDCT N σημείων εκφράζεται συναρτήσει δύο μετασχηματισμών $N/2$ σημείων. Επαναλαμβάνοντας την διαδικασία με τον ίδιο τρόπο, ο IDCT μπορεί να κατακερματιστεί εκ νέου, έως ότου φτάσει να εκφραστεί μέσω μετασχηματισμών μόλις 2 σημείων, ο οποίος περιγράφεται από τις εξισώσεις:

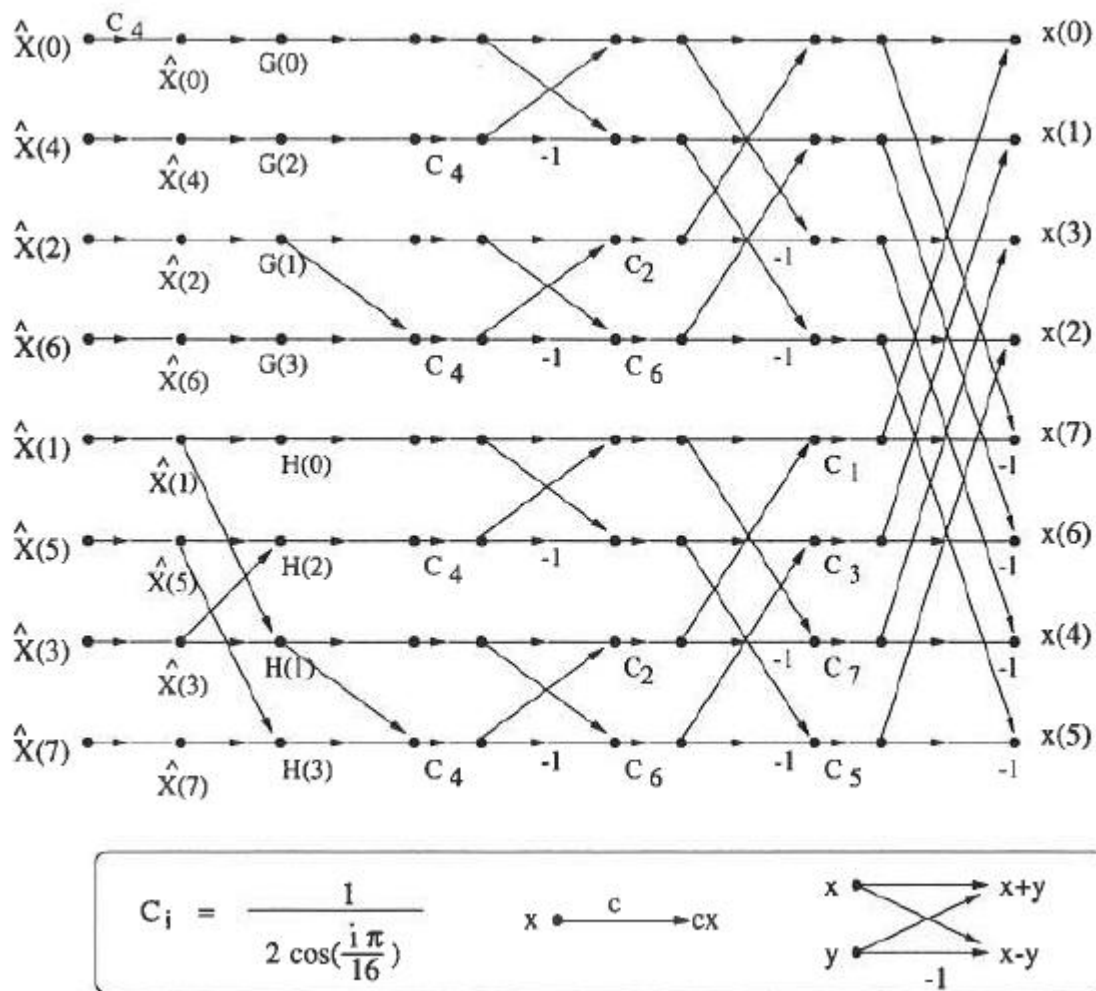
$$\begin{aligned}x(0) &= \hat{X}(0) + \hat{X}(1) \cos \frac{\pi}{4} \\x(1) &= \hat{X}(0) - \hat{X}(1) \cos \frac{\pi}{4}.\end{aligned}$$

Και το γράφημα ροής του είναι:



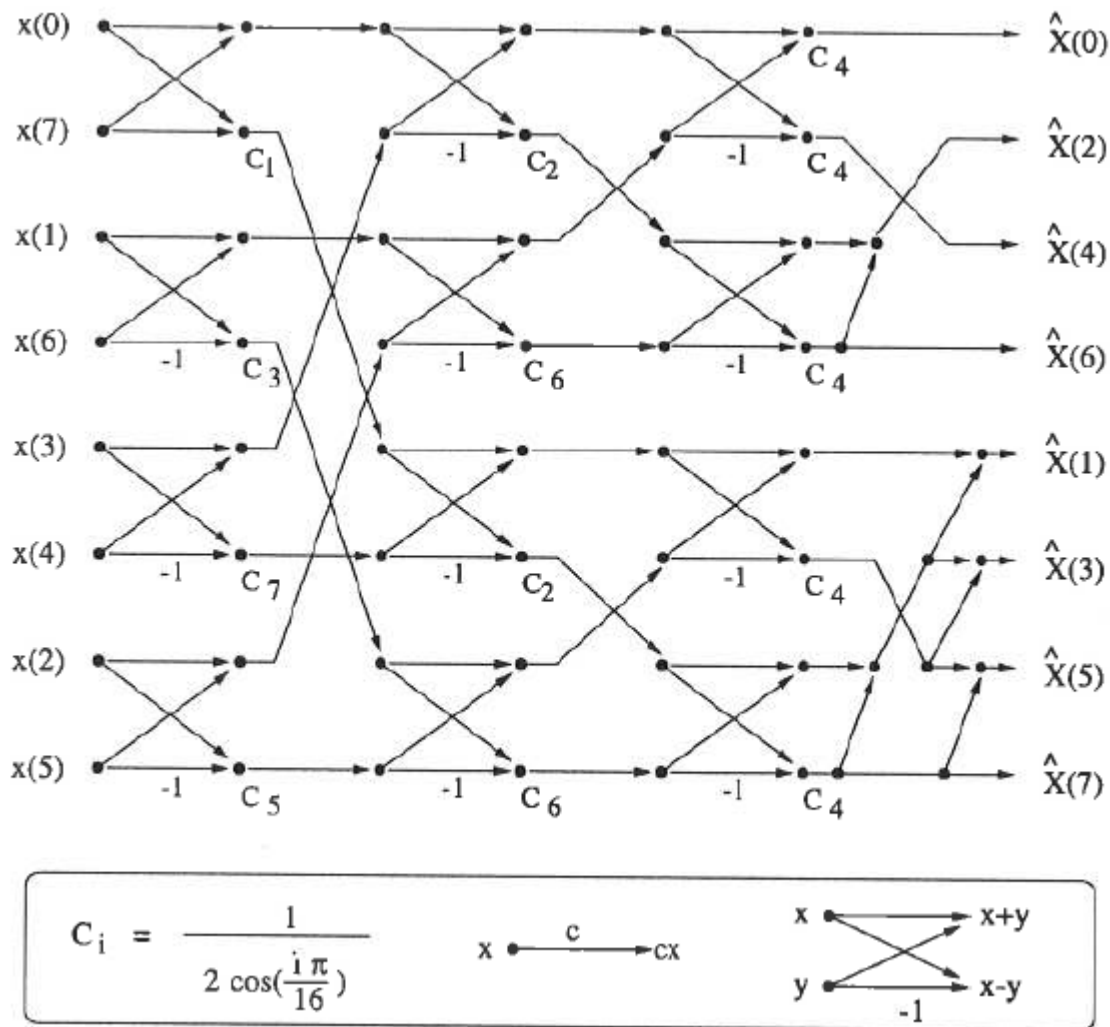
Το παραπάνω γράφημα αποτελεί το βασικό δομικό στοιχείο του αλγορίθμου αυτού, όπως και όλων των αλγορίθμων αποδεκατισμού, και λόγω του σχήματός του, καλείται αρχιτεκτονική πεταλούδας (butterfly architecture).

Έτσι τελικά, στο σχήμα 2.3 φαίνεται το γράφημα ροής του IDCT, ας πούμε ενδεικτικά για $N=8$.



Σχήμα 2.3 : Γράφημα ροής του IDCT 8 σημείων.

Είναι γνωστό, από τη θεωρία γράφων, πως για να παραγάγουμε τον γράφο ροής του συστήματος που θα δέχεται ως εισόδους τα $x(n)$ και θα υπολογίζει τα $X(k)$, δηλαδή τον DCT, αρκεί απλά να αντιστρέψουμε την φορά όλων των κατευθυντικών γραμμών του παραπάνω γράφου ροής. Με αυτό τον τρόπο, παράγεται τελικά το γράφημα ροής του DCT 8 σημείων, που παρουσιάζεται στο σχήμα 2.4.



Σχήμα 2.4 : Γράφημα ροής του DCT 8 σημείων.

Από το παραπάνω γράφημα φαίνεται ότι για την υλοποίηση του 1D-DCT 8 σημείων απαιτείται η εκτέλεση 13 πολλαπλασιασμών. Γενικά ο συγκεκριμένος αλγόριθμος απαιτεί την εκτέλεση $\frac{N \log_2 N}{2} + 1$ πολλαπλασιασμών, όπου N το μέγεθος του μετασχηματισμού.

Παρατηρούμε στο παραπάνω γράφημα, ότι οι 5 από τις 8 ακμές στην έξοδο του τελευταίου σταδίου πεταλούδων πολλαπλασιάζονται με τον συντελεστή C_4 . Συνεπώς, αν επιλέξουμε να υπολογίζουμε όχι την ακολουθία $X(k)$, αλλά την βαθμωμένη ακολουθία $X(k)/C_4$, τότε η μόνη αλλαγή που θα επιφέρουμε στο

γράφημα είναι η αφαίρεση του C_4 από τις 5 ακμές που υπάρχει στην παραπάνω μορφή και η εμφάνιση του στις άλλες τρεις που πριν δεν υπήρχε. Με αυτό τον τρόπο οι πολλαπλασιασμοί στο παραπάνω γράφημα γίνονται 11 από 13, ενώ γενικά μειώνονται σε $\frac{N \log_2 N}{2} - 1$.

Επιπλέον, παρατηρούμε στο παραπάνω γράφημα ότι στο δεύτερο στάδιο υπάρχουν δύο ακριβώς ίδια συστήματα πεταλούδων, που αποτελούν δύο μετασχηματισμούς 4 σημείων, ενώ στο τρίτο υπάρχουν 4 ίδια συστήματα, που αποτελούν τους μετασχηματισμούς 2 σημείων. Συμπεραίνουμε λοιπόν ότι στο παραπάνω γράφημα εμπεριέχεται όλο το απαραίτητο υλικό, όσον αφορά τις πεταλούδες, ώστε ένα σύστημα μετασχηματισμού N σημείων να μπορεί να εκτελέσει παράλληλα αν χρειαστεί 2 μετασχηματισμούς $N/2$ σημείων, 4 των $N/4$ σημείων και ούτω καθεξής. Η ιδιότητα αυτή είναι πολύ χρήσιμη σε εφαρμογές συμπίεσης video, όπου απαιτείται από το σύστημα μετασχηματισμού να υποστηρίζει και να μετασχηματίζει εισόδους διαφορετικών διαστάσεων, όπως το HEVC που καθορίζει ως πιθανά μεγέθη μετασχηματισμού τα 4,8,16,32 σημεία.

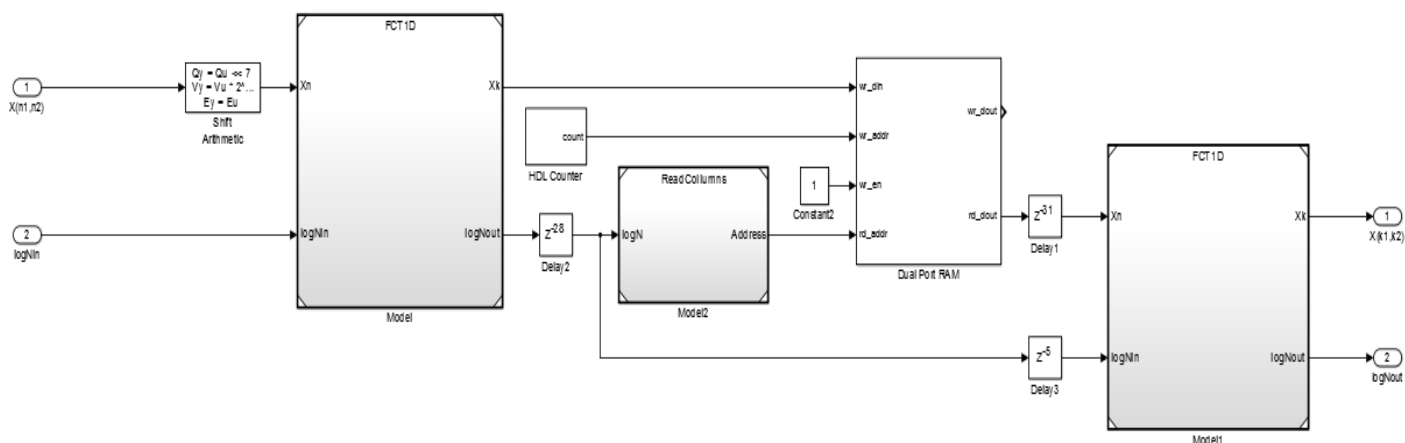
Ακόμα, γνωρίζοντας ότι όλοι οι συντελεστές του μετασχηματισμού είναι σταθεροί και αμετάβλητοι, μπορούμε να αντικαταστήσουμε τους πολλαπλασιαστές με κυκλώματα πρόσθεσης και ολίσθησης. Το απαραίτητο υλικό για κάθε τέτοιο κύκλωμα είναι $m-1$ αθροιστές και m ολισθήσεις, όπου m ο αριθμός των μη μηδενικών bits στην δυαδική αναπαράσταση του κάθε συντελεστή, ενώ η καθυστέρηση ενός τέτοιου κυκλώματος είναι $\text{ceil}(\log m)$, όπου $\text{ceil}(x)$ η στρογγυλοποιημένη προς τα πάνω τιμή του x και $\log m$ ο λογάριθμος του m με βάση το 2.

Τέλος, μπορούμε να εκμεταλλευτούμε το γεγονός ότι όλα τα πιθανά μεγέθη μετασχηματισμών είναι δυνάμεις του 2 και να κρατάμε ως πληροφορία του μεγέθους του μετασχηματισμού όχι το ίδιο το N , αλλά τον λογάριθμο του με βάση το 2 ($\log N$). Με τον τρόπο αυτό μειώνουμε από 7 σε 3 bits το μέγεθος όλων των καταχωρητών στους οποίους αποθηκεύεται η πληροφορία αυτή κατά την διάρκεια της

επεξεργασίας, συνεπώς έχουμε μείωση του απαραίτητου για αυτό υλικού πάνω από 50%.

Χρησιμοποιώντας τις παρατηρήσεις αυτές, καταλήγουμε στην παρακάτω υλοποίηση:

Το σύστημα δέχεται σειριακά τα στοιχεία των Blocks σφαλμάτων στην είσοδό του, καθώς και τον λογάριθμο του μεγέθους του μετασχηματισμού, και δίνει στην έξοδό του, επίσης σειριακά τα μετασχηματισμένα δείγματα. Αποτελείται από δύο όμοια υποσυστήματα υλοποίησης του 1D-DCT, ένα το οποίο αναλαμβάνει τον μετασχηματισμό των γραμμών και ένα που αναλαμβάνει τον μετασχηματισμό των στηλών, καθώς και μία ενδιάμεση μνήμη, στην οποία αποθηκεύονται προσωρινά τα ενδιάμεσα αποτελέσματα, όπως φαίνεται και στο σχήμα 2.5.

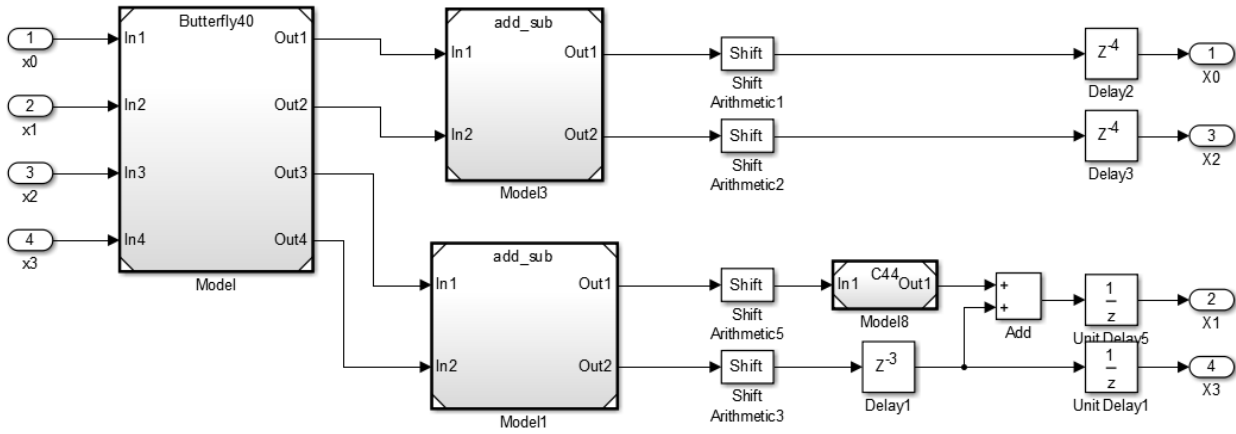


Σχήμα 2. 5: Block διάγραμμα του προτεινόμενου 2D-DCT.

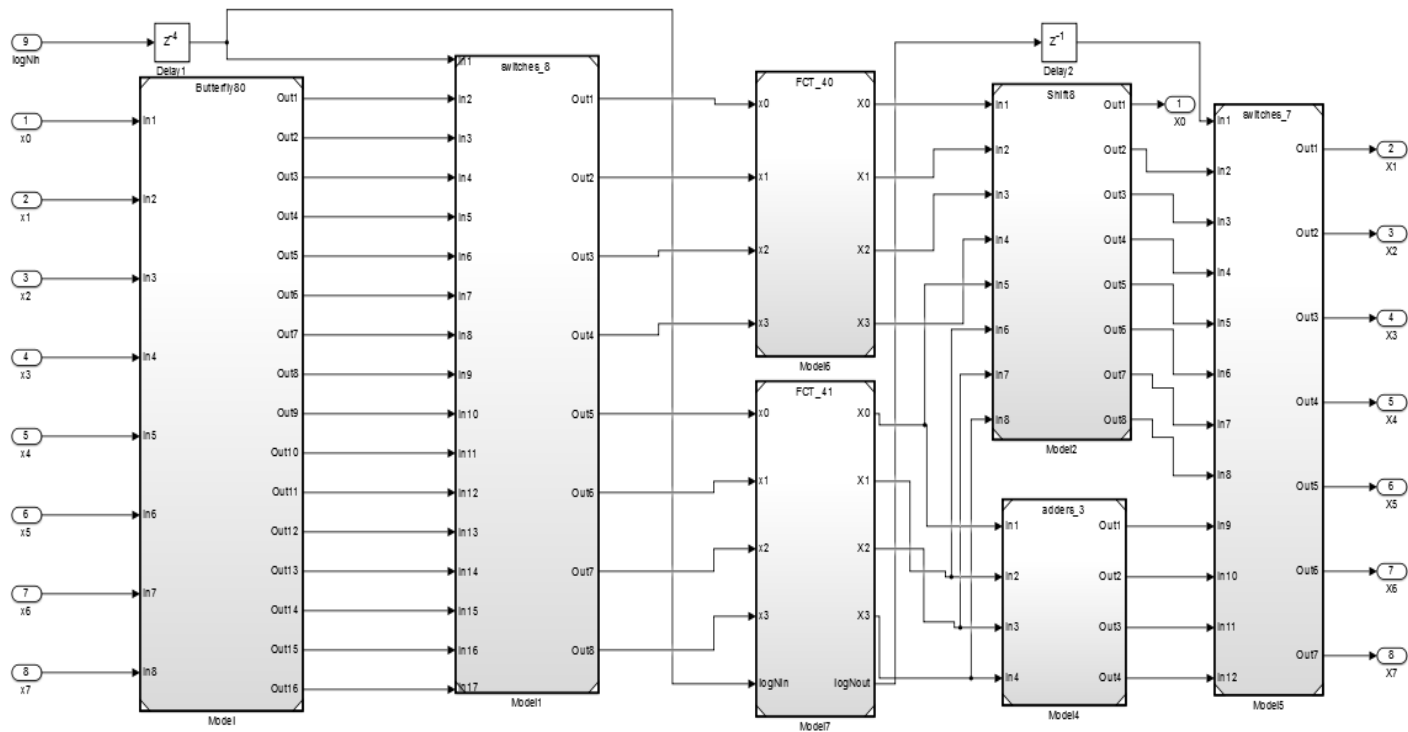
Το ελάχιστο μέγεθος της μνήμης αυτής είναι 32x32 στοιχεία των 16bits το καθένα, σύνολο δηλαδή 2KB. Ωστόσο στην συγκεκριμένη υλοποίηση επιλέχτηκε μνήμη διπλασίου μεγέθους, ώστε να επιτευχθεί ταυτόχρονη προσπέλασή της από τα δύο

υποσυστήματα μετασχηματισμών. Πιο συγκεκριμένα, όσο ο μετασχηματισμός γραμμών αποθηκεύει τα αποτελέσματα του στο πρώτο μισό της συγκεκριμένης μνήμης, την ίδια στιγμή ο μετασχηματισμός στηλών διαβάζει τα στοιχεία εισόδου του από το δεύτερο μισό της μνήμης. Με τον τρόπο αυτό έχουμε προφανώς ανάγκη για μνήμη διπλασίου μεγέθους, άρα και μεγαλύτερες απαιτήσεις στην επιφάνεια ολοκλήρωσης, ωστόσο κερδίζουμε τον διπλασιασμό του ρυθμού επεξεργασίας δεδομένων(throughput).

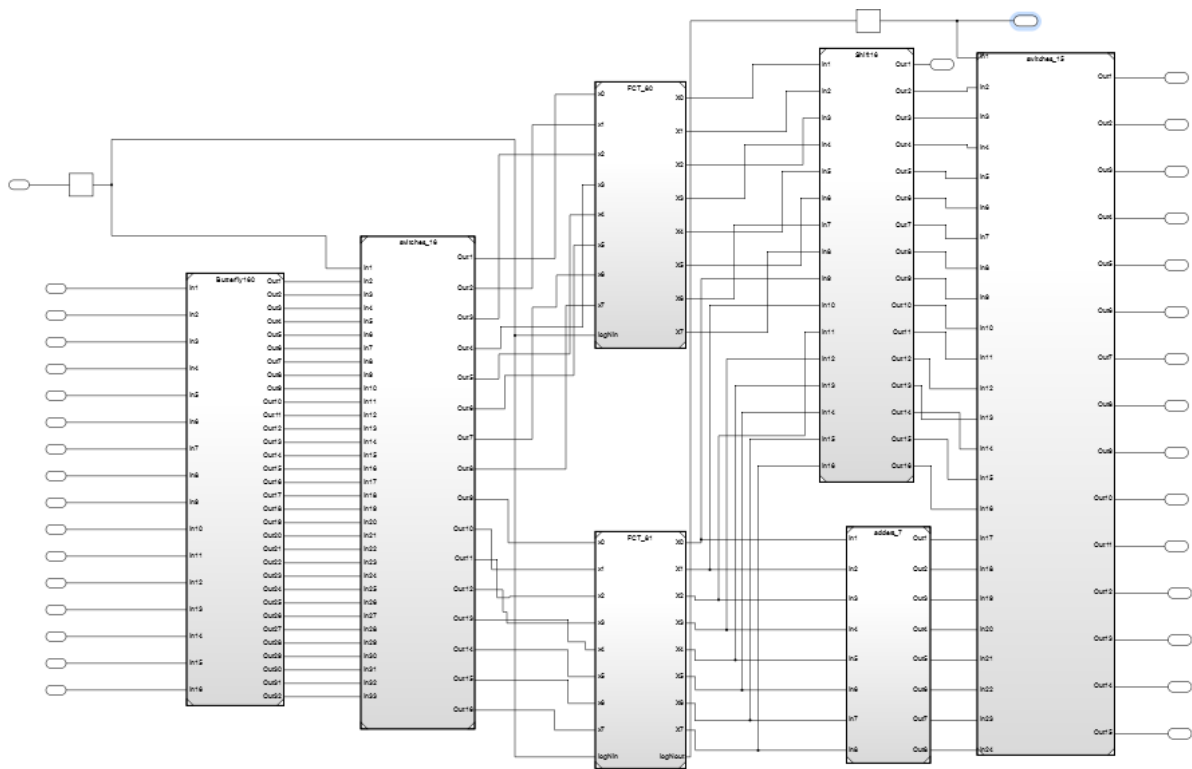
Καθένα από τα δύο όμοια υποσυστήματα υλοποίησης του 1D-DCT περιέχει το απαιτούμενο υλικό για έναν μετασχηματισμό 32 σημείων, του μεγαλύτερου δηλαδή μεγέθους που απαιτεί το πρότυπο HEVC. Στο υλικό αυτό προστίθενται $2N(\log N - 2)$ πολυπλέκτες (συγκεκριμένα 192 πολυπλέκτες για $N=32$). Πιο συγκεκριμένα, ανάμεσα σε κάθε δύο στάδια πεταλούδων, τοποθετούνται N πολυπλέκτες, οι οποίοι δέχονται ως εισόδους τόσο τις εξόδους των προηγούμενων πεταλούδων, όσο και τα αρχικά στοιχεία. Αν ο μετασχηματισμός του συγκεκριμένου μεγέθους απαιτεί την εκτέλεση των προηγούμενων πεταλούδων, τότε στην έξοδο των πολυπλεκτών περνάει το αποτέλεσμά τους. Αν όχι, τότε περνάνε τα αρχικά δείγματα. Η συγκεκριμένη λογική επαναλαμβάνεται σε κάθε στάδιο του μετασχηματισμού. Με τον τρόπο αυτό, το σύστημα είναι ικανό να υπολογίσει είτε έναν DCT 32 σημείων, είτε 2 DCTs 16 σημείων, είτε 4 DCTs 8 σημείων, είτε τέλος 8 DCTs 4 σημείων παράλληλα. Επεξεργάζεται δηλαδή σε κάθε κύκλο ρολογιού 32 στοιχεία, αυξάνοντας το throughput κατά $32/N$ φορές, όπου N το μέγεθος του μετασχηματισμού. Στα σχήματα 2.6-2.9 παρατίθενται τα υποσυστήματα μετασχηματισμού 4,8,16 και 32 σημείων



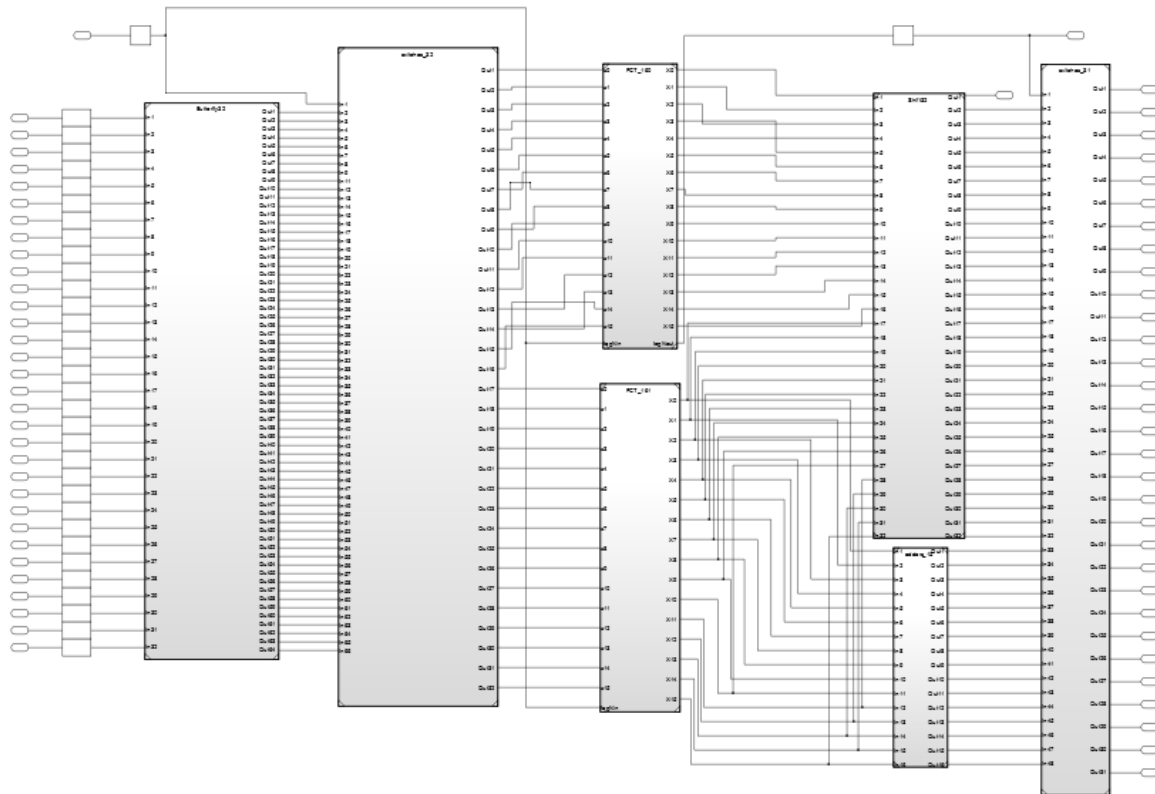
Σχήμα 2. 6: Block διάγραμμα του υποσυστήματος DCT 4 σημείων.



Σχήμα 2. 7: Block διάγραμμα του υποσυστήματος DCT 8 σημείων.

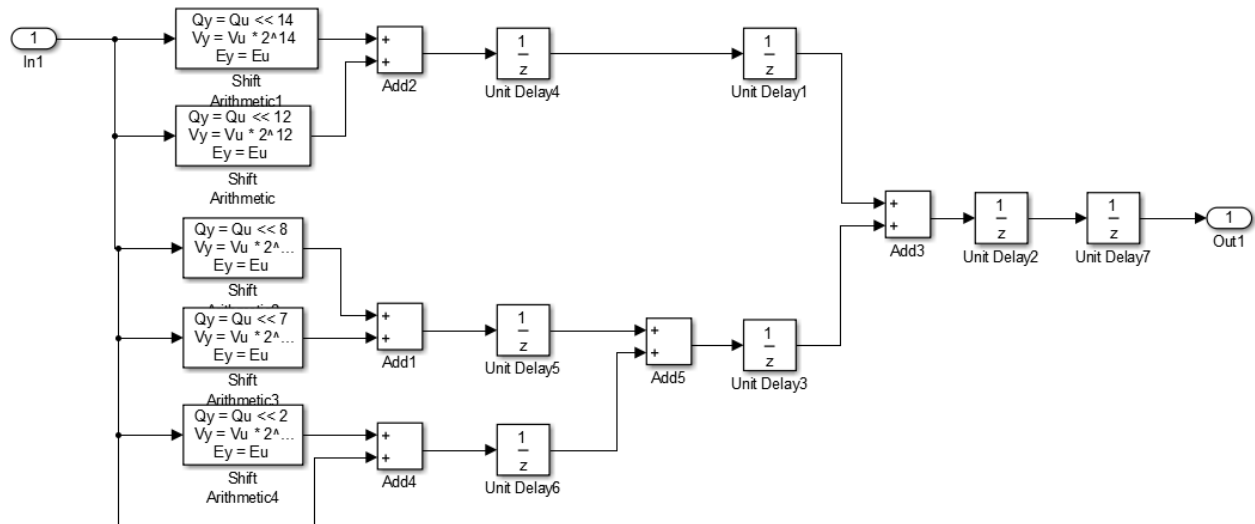


Σχήμα 2. 8: Block διάγραμμα του υποσυστήματος DCT 16 σημείων.



Σχήμα 2. 9: Block διάγραμμα του υποσυστήματος DCT 32 σημείων.

Όσον αφορά τους πολλαπλασιασμούς, αυτοί υλοποιούνται με κυκλώματα αθροίσεων και ολισθήσεων, όπως αυτά που περιγράφηκαν παραπάνω. Στο σχήμα 2.10 φαίνεται η δομή ενός τέτοιου κυκλώματος πολλαπλασιασμού.



Σχήμα 2. 10: Κύκλωμα πολλαπλασιασμού με συγκεκριμένο συντελεστή.

Η συγκεκριμένη πρακτική, εκτός του ότι επιταχύνει την εκτέλεση του πολλαπλασιασμού, μας δίνει την δυνατότητα να μειώσουμε τον ίδιο τον κύκλο ρολογιού κάτω από τον χρόνο εκτέλεσης του πολλαπλασιαστή με χρήση της μεθόδου διασωλήνωσης (pipelining), κάτι που σε περίπτωση που χρησιμοποιούνταν ενιαίο κύκλωμα πολλαπλασιαστή δεν θα ήταν εφικτό. Οι συντελεστές του μετασχηματισμού κβαντίζονται σε ακρίβεια 11 κλασματικών ψηφίων, που είναι η μέγιστη ακρίβεια η οποία εξασφαλίζει την προστασία του από υπερχείλιση (overflow). Η επιλογή αυτή έχει αρνητικό αντίκτυπο στην επιφάνεια ολοκλήρωσης, καθώς σε συντελεστές πολλών bits είναι πιο πιθανό να υπάρχουν πολλά μη μηδενικά bits, άρα αυξάνεται ο αριθμός των απαιτούμενων αθροιστών για την υλοποίηση του πολλαπλασιασμού, όμως κρατάει σε υψηλά επίπεδα τον **λόγο σήματος προς θόρυβο κβάντισης** (Signal to Quantization Noise Ratio, SQNR) των τελικών αποτελεσμάτων. Το SQNR αποτελεί ένα μέτρο ταύτισης των αποτελεσμάτων του συστήματος με τα ιδανικά.

Ειδικά η διατήρηση μεγάλου SQNR απαιτεί ειδική φροντίδα σε αλγορίθμους που περιέχουν παραπάνω από έναν πολλαπλασιασμούς στη σειρά. Για κάθε πολλαπλασιασμό δύο αριθμών, έστω n bits ο καθένας, απαιτείται καταχωρητής

μεγέθους $2n$ bits για την αποθήκευση του τελικού αποτελέσματος. Είναι λογικό ότι σε περιπτώσεις διαδοχικών πολλαπλασιασμών η απαίτηση αυτή δεν γίνεται να ικανοποιηθεί, καθώς θα απαιτούνταν καταχωρητές πολύ μεγάλου μεγέθους. Γι αυτό είναι απαραίτητο σε κάποιο σημείο ανάμεσα σε δύο διαδοχικούς πολλαπλασιασμούς να γίνεται δεξιά ολίσθηση κατά n bits, ώστε να αποθηκεύονται μόνο τα n πιο σημαντικά ψηφία του αποτελέσματος, να διατηρείται το επιθυμητό μέγεθος και να μην υπάρχει κίνδυνος υπερχείλισης. Σε περίπτωση που υπάρχουν προσθέσεις μεταξύ των διαδοχικών πολλαπλασιασμών, επιτυγχάνεται το μέγιστο δυνατό SQNR αν όλες οι προσθέσεις γίνουν με ακρίβεια $2n$ bits και μετά εκτελεστεί η ολίσθηση, αρκεί να έχει ληφθεί μέριμνα για την αποφυγή υπερχείλισης.

Ακόμα, γίνεται χρήση της μεθόδου διασωλήνωσης (pipelining), η οποία περιγράφηκε σε προηγούμενο κεφάλαιο, με σκοπό τόσο την μείωση της ελάχιστης χρονικής διάρκειας του κύκλου ρολογιού που απαιτείται για ορθή λειτουργία του συστήματος, όσο και της καταναλισκομένης ισχύος. Πιο συγκεκριμένα, μετά από κάθε αθροιστή του συστήματος τοποθετείται ένας καταχωρητής, και έτσι, θεωρώντας κυρίαρχη την καθυστέρηση των αθροιστών στο σύστημα, καταλήγουμε σε μείωση του κρίσιμου μονοπατιού κατά 27 φορές και ο κύκλος ρολογιού μπορεί να μειωθεί μέχρι να φτάσει περίπου τον χρόνο του ενός αθροιστή, δεδομένου ότι η ελάχιστη διάρκεια του δεν περιορίζεται από τον χρόνο προσπέλασης της ενδιάμεσης μνήμης.

Εναλλακτικά, μπορεί να επιτευχτεί μέσω του pipelining μείωση της καταναλισκομένης ισχύος. Από τον τύπο:

$$LM \cdot (\beta V_0 - V_t)^2 = \beta (V_0 - V_t)^2$$

για $L=1$, $M=27$ και τυπικές τιμές $V_0=5V$, $V_t=0.6V$ έχουμε

$$\beta=0.074 \Rightarrow \beta^2 = 0.005$$

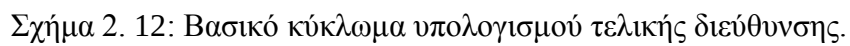
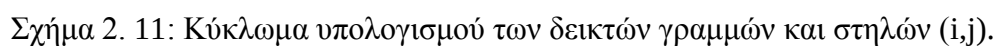
οπότε η καταναλισκόμενη ισχύς του κυκλώματος, στην περίπτωση που δεν μειώσουμε καθόλου τον κύκλο ρολογιού θα είναι περίπου 200 φορές μικρότερη.

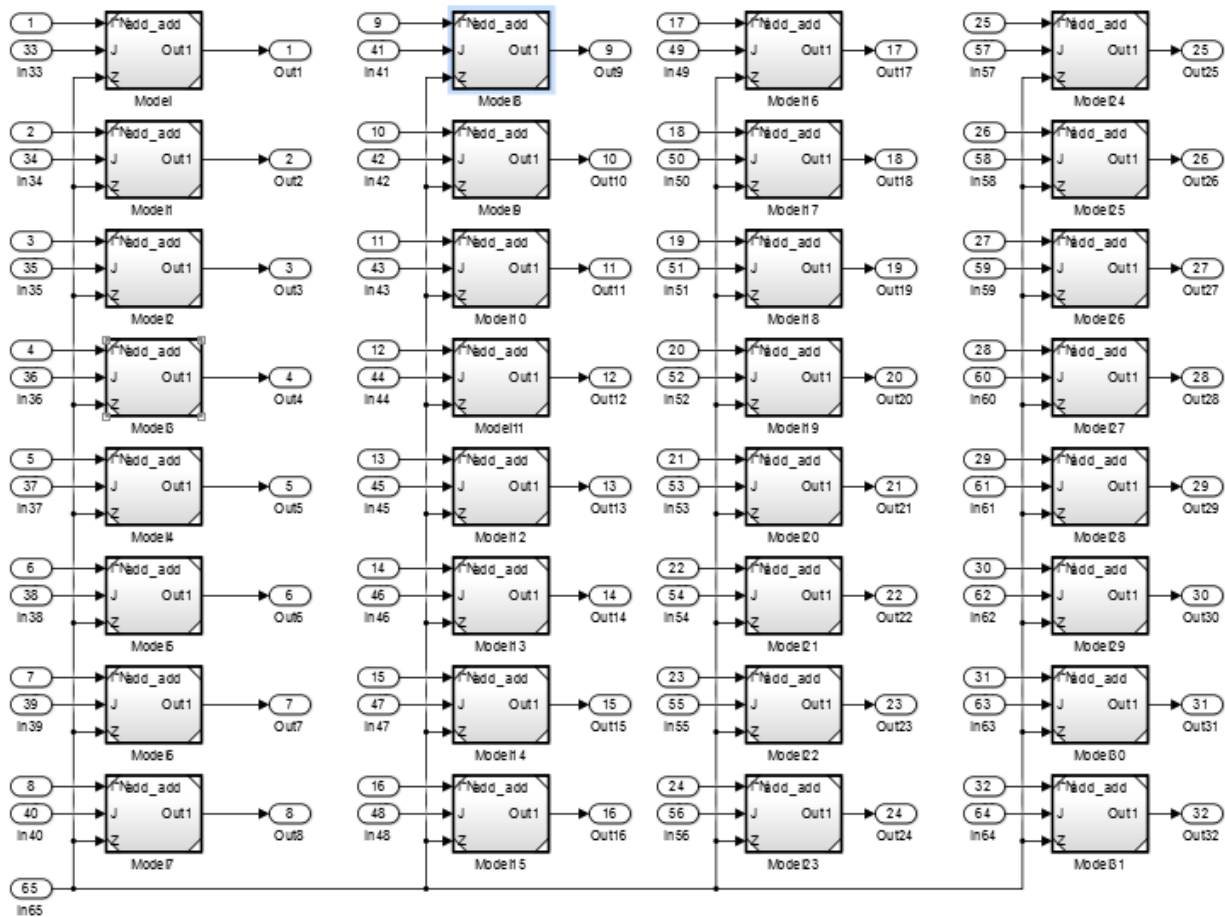
Τέλος, πρέπει να αναφερθεί ότι, για να γίνει εκμετάλλευση του μέγιστου throughput που μπορεί να επιτύχει το κύκλωμα του DCT μίας διάστασης, είναι απαραίτητο να λειτουργούν με τον ίδιο ρυθμό τα κυκλώματα ανάγνωσης και εγγραφής στην ενδιάμεση μνήμη. Κι αν η δημιουργία των κατάλληλων διευθύνσεων για εγγραφή κατά γραμμές μπορεί να γίνει με έναν απλό μετρητή (counter), η δημιουργία διευθύνσεων για ανάγνωση κατά στήλες είναι πιο πολύπλοκη, αφού απαιτεί προσθέσεις με την διάσταση του Block n . Συνεπώς, εφαρμόσαμε τόσο pipelining, όσο και parallel processing, ώστε να είναι δυνατή η δημιουργία 32 διευθύνσεων ανά χρόνο ενός αθροιστή.

Πιο συγκεκριμένα, η διεύθυνση που πρέπει να δημιουργείται είναι :

$$[Z+i*n+j]$$

όπου i, j οι δείκτες γραμμών και στηλών αντίστοιχα στο block, n η διάσταση του block και Z το offset μέσα στην μνήμη στο οποίο ξεκινάει το ζητούμενο block. Για να δημιουργούνται 32 τέτοιες διευθύνσεις ανά ένα χρόνο ενός αθροιστή πρέπει να εφαρμόσουμε pipelining 2 επιπέδων, δηλαδή να τοποθετήσουμε έναν καταχωρητή μεταξύ των δύο αθροιστών, και ακόμα parallel processing 32 επιπέδων. Το τίμημα που πληρώνουμε βέβαια είναι η επιπλέον επιφάνεια 64 αθροιστών, δηλαδή 32 φορές περισσότερων από την κλασική υλοποίηση, καθώς και ανάλογη αύξηση της κατανάλωσης ενέργειας, λόγω της αύξησης της συνολικής χωρητικότητας του κυκλώματος κατά 32 φορές. Στα σχήματα 2.11, 2.12 και 2.13 φαίνεται το δομικό διάγραμμα του ακολουθιακού κυκλώματος που παράγει τους δείκτες γραμμών και στηλών και το κύκλωμα υπολογισμού των τελικών διευθύνσεων αντίστοιχα. Να σημειωθεί πως το κύκλωμα του σχήματος 2.12 επαναλαμβάνεται 32 φορές στο προτεινόμενο σύστημα, ώστε να υπολογίζονται ταυτόχρονα 32 διευθύνσεις και να επιτυγχάνεται το επιθυμητό throughput, όπως φαίνεται στο σχήμα 2.13.





Σχήμα 2.13 : Ολοκληρωμένο κύκλωμα παράλληλου υπολογισμού 32 διευθύνσεων στην ενδιάμεση μνήμη.

Συνολικά λοιπόν, η υλοποίηση που προτείνεται πετυχαίνει αύξηση του ρυθμού επεξεργασίας (throughput) σε $T = \max(T_{\text{mem}}, T_{\text{add}}/32)$, όπου T_{mem} ο χρόνος προσπέλασης της ενδιάμεσης μνήμης και T_{add} ο χρόνος ολοκλήρωσης μιας πρόσθεσης, δηλαδή επιτυγχάνεται αύξηση κατά $1728/N$ φορές, όπου N το μέγεθος του μετασχηματισμού, σε σχέση με την κλασική υλοποίηση, ή εναλλακτικά, να μειώσει την κατανάλωση ισχύος κατά περίπου 200 φορές, αν δεν επιθυμούμε καθόλου αύξηση του throughput. Όσον αφορά την επιφάνεια ολοκλήρωσης, η προτεινόμενη υλοποίηση περιέχει το απαραίτητο υλικό για την υλοποίηση του 2D-DCT 32x32 σημείων με χρήση των τεχνικών διαχωρισμού γραμμών-στηλών (row-column decomposition) και αποδεκατισμού στην συχνότητα (Decimation in Frequency, DIF), με την προσθήκη των απαραίτητων pipeline καταχωρητών, $2N(\log N - 2)$ πολυπλεκτών (συγκεκριμένα 192 πολυπλέκτες για $N=32$), ώστε να

επιτυγχάνεται παράλληλος υπολογισμός μετασχηματισμών για $N < 32$, καθώς και χρήση διπλασίου μεγέθους μνήμης (4KB έναντι 2KB) και 32 φορές μεγαλύτερου μεγέθους κυκλώματος διευθυνσιοδότησης της (64 αθροιστές έναντι 2), ώστε να επιτευχθεί το παραπάνω throughput.

2.4 Πειραματικές Μετρήσεις

Για την μέτρηση της απόδοσης του συστήματος που υλοποιήθηκε, τρία συνεχόμενα πλαίσια 32×32 εικονοστοιχείων μιας ακολουθίας video χωρίστηκαν σε blocks διαστάσεων 8×8 εικονοστοιχείων, εφαρμόστηκε σε αυτά ο μετασχηματισμός DCT και στη συνέχεια κάθε block ανακατασκευάστηκε με χρήση του ιδανικού μετασχηματισμού IDCT, με χρήση όμως μόνο n_0^2 συντελεστών, όπου $n_0 = 1 \dots 8$, και σχηματίστηκαν το τωρινό frame $X(i, j)$, και τα ανακατασκευασμένα frame $X'(i, j)$, για κάθε n_0 , και μετρήθηκαν τα παρακάτω μεγέθη :

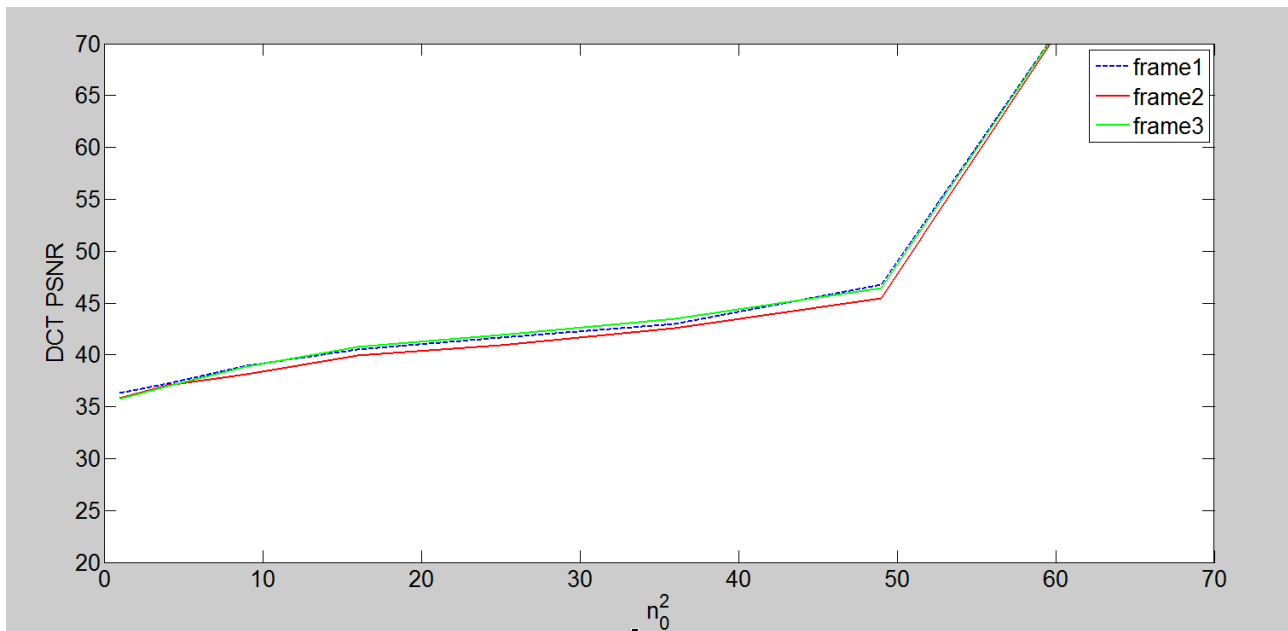
α) Λόγος Μέγιστου Σήματος προς Θόρυβο (Peak Signal to Noise Ratio, PSNR) που δίνεται από τον τύπο

$$\text{PSNR} = \log_{10} \frac{M^2 255^2}{E_{\text{error}}}$$

όπου E_{error} η ενέργεια της διαφοράς μεταξύ του frame εισόδου στον DCT $X(i, j)$ και του frame, $X'(i, j)$, που παράγει ο ιδανικός αντίστροφος μετασχηματισμός συνημίτονου (IDCT) από τα μετασχηματισμένα δεδομένα, ανάλογα με τον αριθμό των συντελεστών, n_0^2 , που χρησιμοποιούνται για την ανακατασκευή, η οποία δίνεται από

$$E_{\text{error}} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} (X(i, j) - X'(i, j))^2$$

όσο μικρότερο είναι το σφάλμα της ανακατασκευής, τόσο μεγαλύτερο PSNR έχουμε.



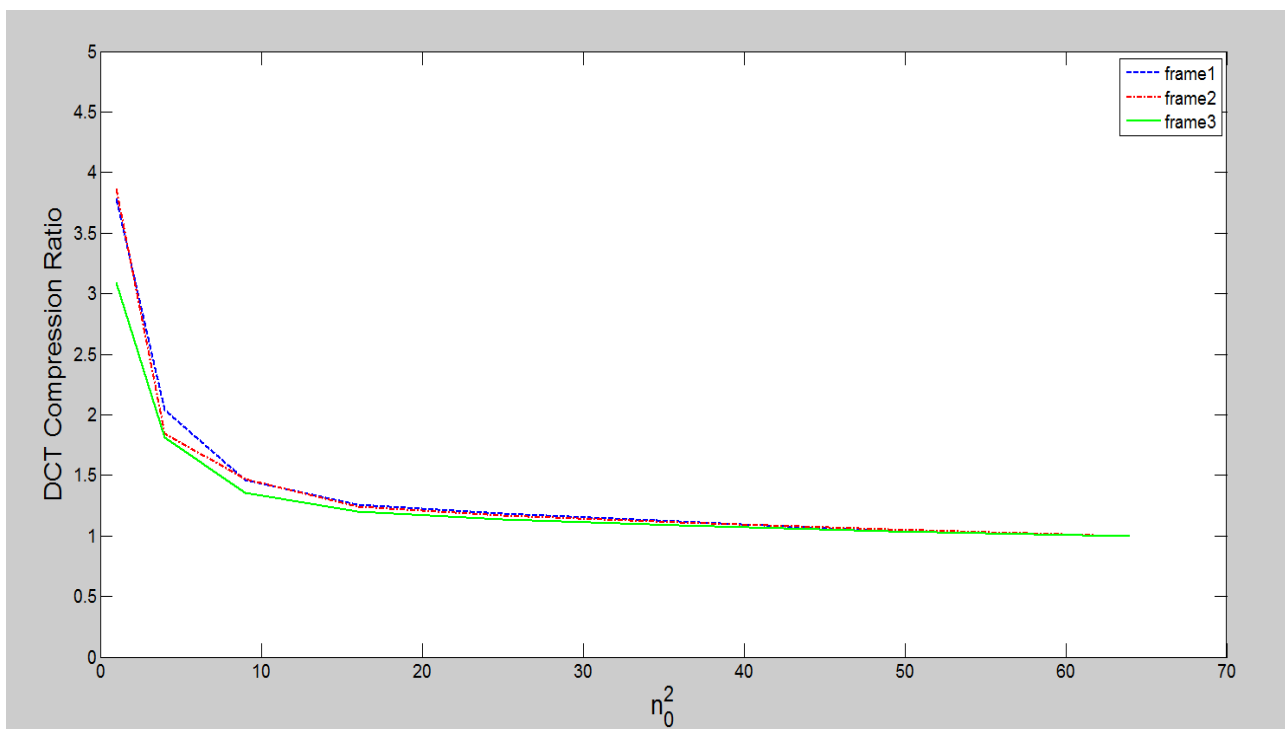
Γραφική Παράσταση 2.1 : PSNR ανακατασκευής εικόνας προς αριθμό DCT συντελεστών που χρησιμοποιούνται για την ανακατασκευή

Παρατηρούμε από την Γραφική Παράσταση 2.1 ότι, αν εξαιρεθεί η περίπτωση όπου για την ανακατασκευή χρησιμοποιούνται όλοι οι συντελεστές, όπου πρακτικά έχουμε τέλεια ανακατασκευή, ο ρυθμός μείωσης του PSNR είναι πολύ χαμηλός, κάτι που δείχνει ότι μπορούμε να αμελήσουμε συντελεστές υψηλών συχνοτήτων, ώστε να επιτευχτεί συμπίεση, χωρίς να υποστεί η εικόνα μεγάλη παραμόρφωση.

β) Λόγος συμπίεσης δεδομένων (Compression Ratio, CR), που δίνεται από τον τύπο:

$$CR = \sqrt{\frac{E_{in}}{E_{out}}}$$

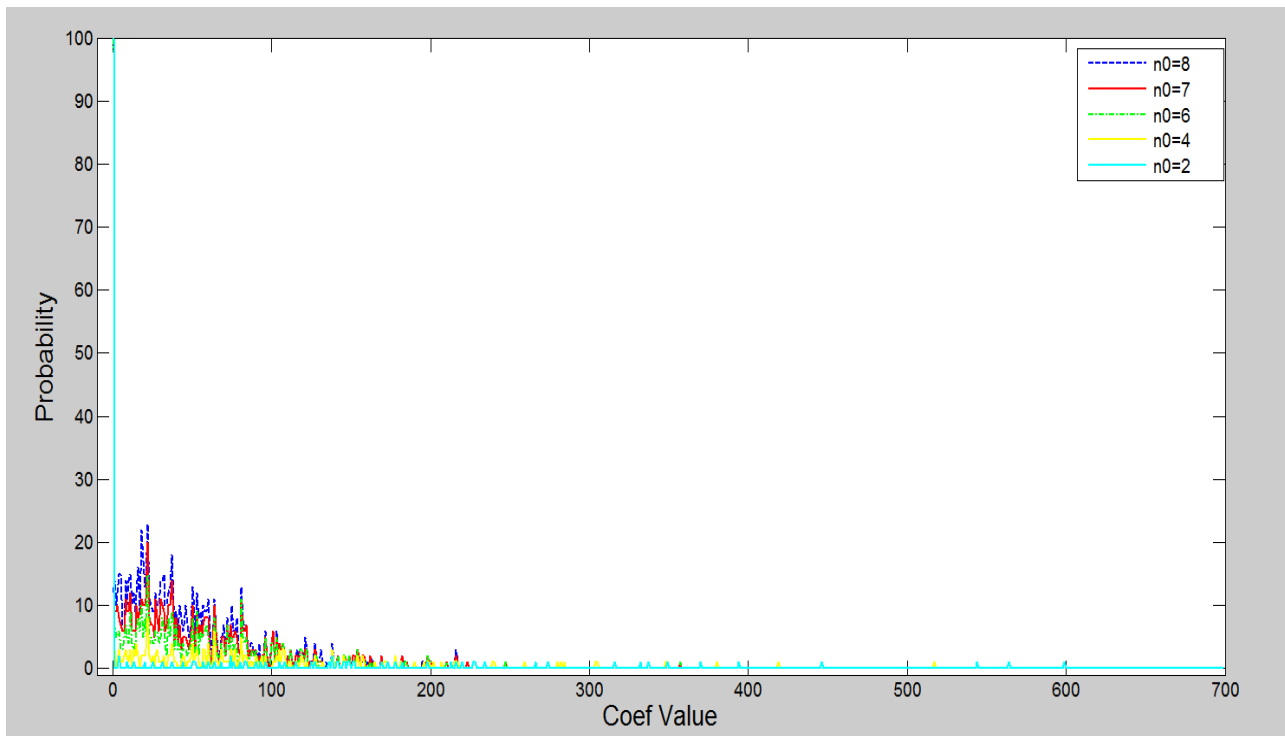
όπου E_{in} η ενέργεια του τωρινού frame $X(i, j)$ και E_{out} η ενέργεια του ανακατασκευασμένου frame $X'(i, j)$. Όσο μεγαλύτερος είναι ο λόγος αυτός τόσο μεγαλύτερη είναι και η συμπίεση των δεδομένων.



Γραφική Παράσταση 2.2 : Λόγος συμπίεσης εικόνας προς αριθμό DCT συντελεστών που χρησιμοποιούνται για την ανακατασκευή

Παρατηρούμε από την Γραφική Παράσταση 2.3 ότι όσο λιγότεροι συντελεστές χρησιμοποιηθούν για την αποστολή της εικόνας, άρα και την ανακατασκευή της από τον δέκτη, τόσο μεγαλύτερη συμπίεση επιτυγχάνεται. Στην περίπτωση που αποσταλούν όλοι οι συντελεστές, το σύστημα μετασχηματισμού δεν επιτυγχάνει καθόλου συμπίεση ($CR = 1$) κάτι που είναι λογικό, καθώς ο DCT είναι ένας μοναδιαίος μετασχηματισμός, που σημαίνει ότι διατηρεί αναλλοίωτη την ενέργεια του σήματος εισόδου.

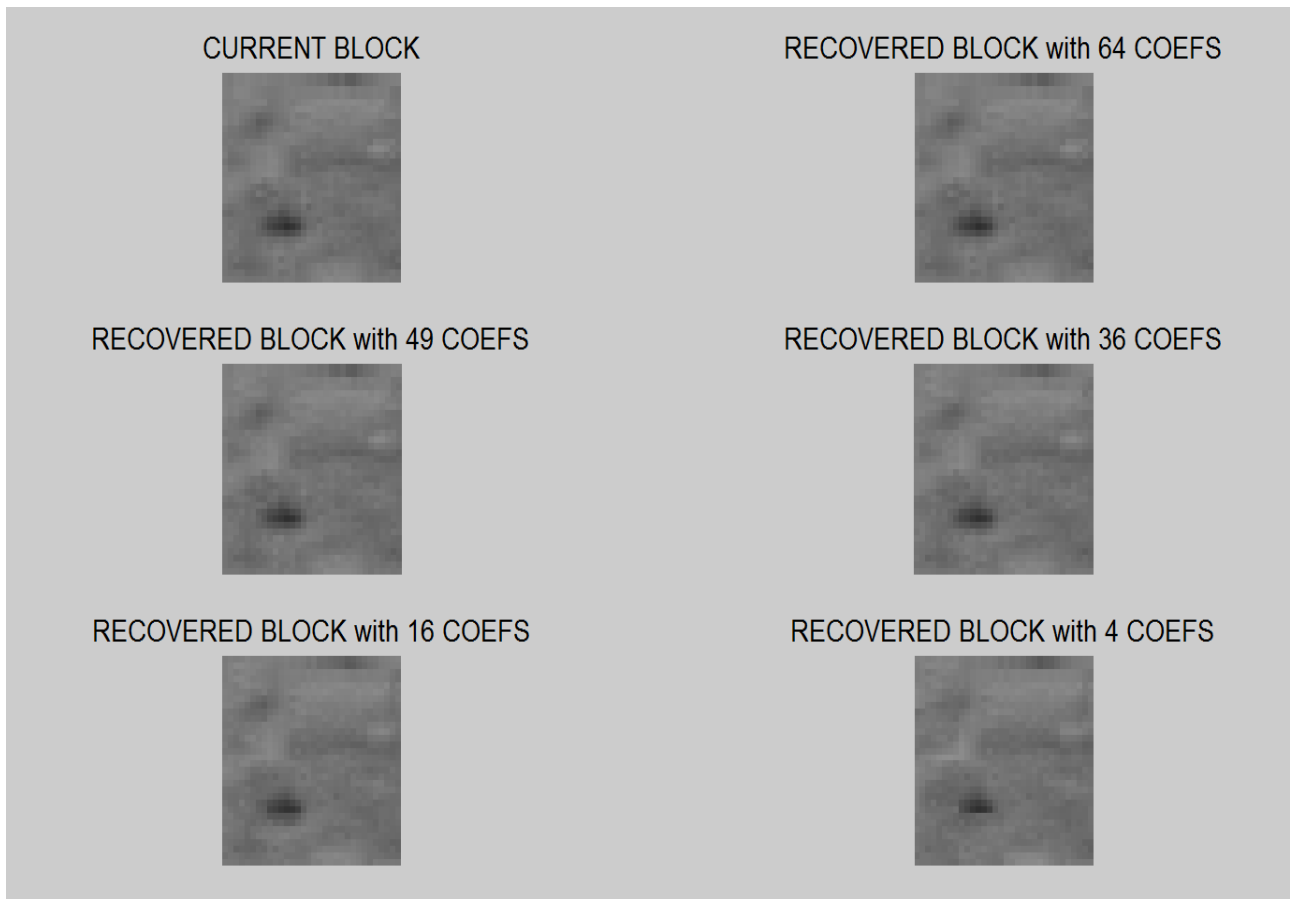
γ) Ιστόγραμμα των παραχθέντων DCT συντελεστών, που είναι ένα μέτρο της πιθανότητας ένας συντελεστής να έχει μια συγκεκριμένη τιμή. Όσο λιγότεροι συντελεστές έχουν μεγάλη πιθανότητα εμφάνισης, τόσο μικρότερη είναι η εντροπία τους, άρα τόσο περισσότερο μπορούν αυτοί να συμπιεστούν από έναν κωδικοποιητή εντροπίας.



Γραφική Παράσταση 2.4: Ιστόγραμμα τιμών DCT συντελεστών.

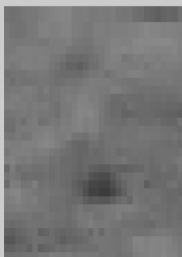
Παρατηρούμε από την Γραφική Παράσταση 2.5 ότι, αν και οι πιθανές τιμές των DCT συντελεστών ανήκουν στο σύνολο $[0, 32768]$, στην πραγματικότητα σχεδόν όλοι οι συντελεστές παίρνουν τιμές μικρότερες του 100 με πολύ μεγάλη πιθανότητα. Αυτό μας δείχνει ότι η εντροπία των συντελεστών είναι πολύ χαμηλή, που σημαίνει ότι μπορούν να υποστούν μεγάλη συμπίεση από έναν κωδικοποιητή εντροπίας, ο οποίος χρησιμοποιείται ως τελευταίο στάδιο συμπίεσης σε κάθε κωδικοποιητή κινούμενων εικόνων.

Στις παρακάτω εικόνες παρατηρούμε οπτικά τις διαφορές μεταξύ των αρχικών πλαισίων και των ανακατασκευασμένων, ανάλογα με τον αριθμό των DCT συντελεστών που χρησιμοποιούνται για την ανακατασκευή (64,49,36,16 ή 4 συντελεστές).



Εικόνα 2.1 : FRAME1

CURRENT BLOCK



RECOVERED BLOCK with 64 COEFS



RECOVERED BLOCK with 49 COEFS



RECOVERED BLOCK with 36 COEFS



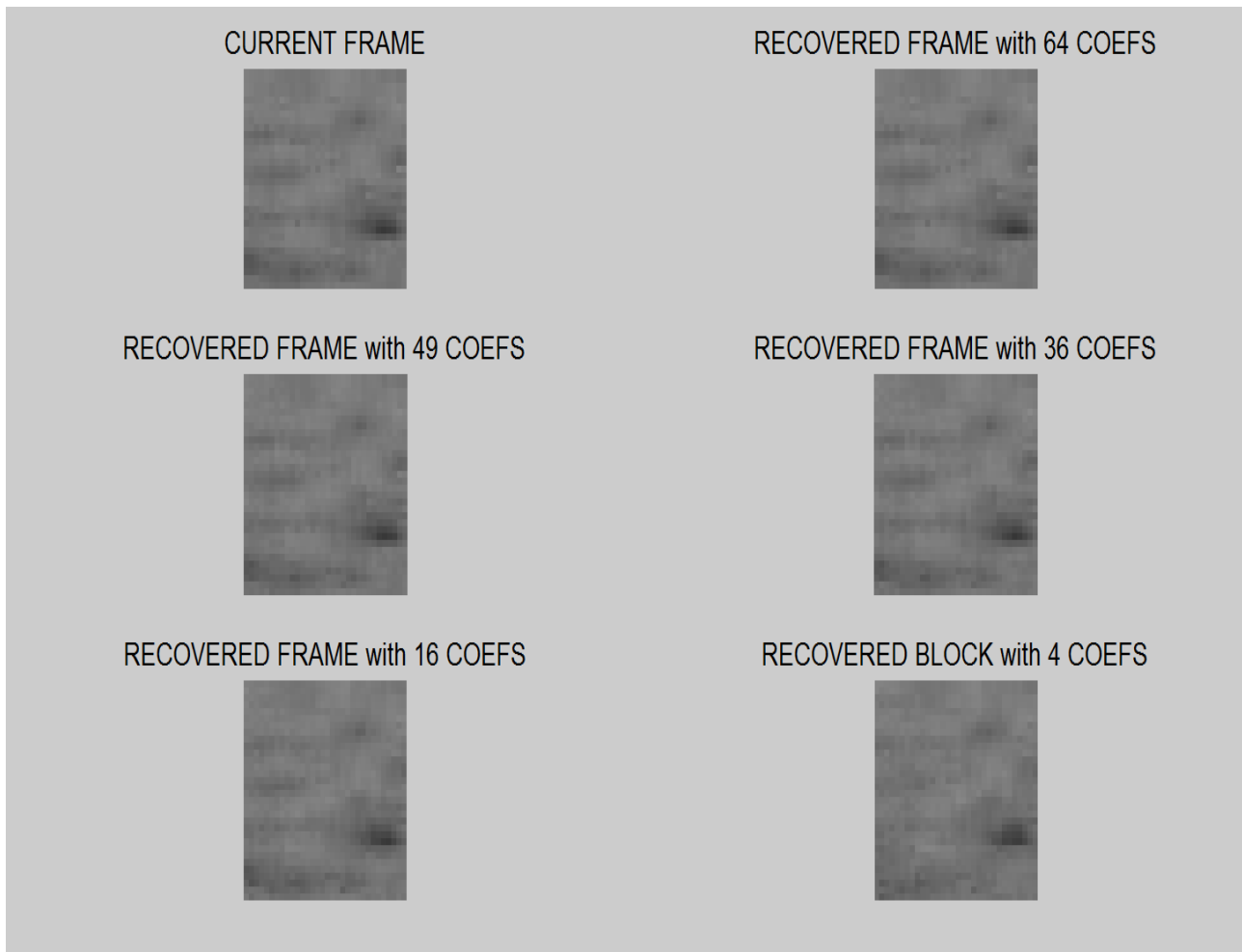
RECOVERED BLOCK with 16 COEFS



RECOVERED BLOCK with 4 COEFS



Εικόνα 2.2 : FRAME2



Εικόνα 2.3 : FRAME3

Παρατηρούμε ότι οπτικά οι διαφορές μεταξύ των εικόνων είναι ελάχιστες, με βασικό χαρακτηριστικό ένα ελαφρύ θάμπωμα των εικόνων όπου χρησιμοποιούνται λίγοι συντελεστές, λόγω της αφαίρεσης της ενέργειας υψηλών συχνοτήτων.

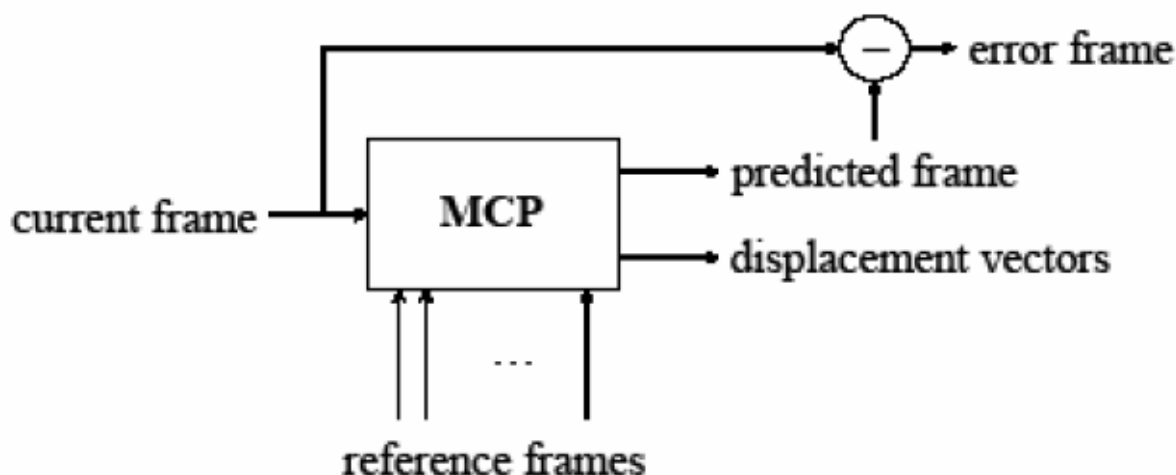
ΚΕΦΑΛΑΙΟ 3 : ΚΙΝΗΣΙΑΚΑ ΑΝΤΙΣΤΑΘΜΙΣΜΕΝΗ ΠΡΟΒΛΕΨΗ

3.1 Θεωρητικό υπόβαθρο

Όπως αναφέραμε σε προηγούμενο κεφάλαιο, το πρώτο βήμα στην κωδικοποίηση ακολουθιών video είναι ο χρονικός μετασχηματισμός της ακολουθίας, ώστε να επιτευχθεί η μείωση της πλεονάζουσας πληροφορίας μεταξύ διαδοχικών πλαισίων. Η κινησιακά αντισταθμισμένη πρόβλεψη (Motion Compensated Prediction, MCP), ως το βασικό κομμάτι της διαδικασίας Inter-prediction, αποτελεί ένα ισχυρό εργαλείο για την μείωση της χρονικής συσχέτισης μεταξύ διαδοχικών πλαισίων video και αποτελεί βασικό κομμάτι σε κάθε κωδικοποιητή video. Στο κεφάλαιο αυτό θα μελετήσουμε τον τρόπο λειτουργίας της κινησιακά αντισταθμισμένης πρόβλεψης.

Στην πιο γενική περίπτωση, η διαδικασία MCP αφαιρεί τη συσχέτιση μεταξύ περιοχών του τρέχοντος πλαισίου και προηγούμενων και επόμενων πλαισίων, τα οποία είναι διαθέσιμα τόσο στον κωδικοποιητή όσο και στον αποκωδικοποιητή. Τα πλαίσια αυτά αναφέρονται συχνά ως πλαίσια αναφοράς. Δεδομένων των πλαισίων αναφοράς και του τρέχοντος πλαισίου, η διαδικασία MCP δημιουργεί τα διανύσματα κίνησης (Motion Vectors, MVs), τα οποία περιέχουν τις παραμέτρους αντιστοίχισης του τρέχοντος πλαισίου με το πλαίσιο αναφοράς, καθώς και το προβλεπόμενο πλαίσιο. Από τη διαφορά του τωρινού πλαισίου και του πλαισίου που προέβλεψε η διαδικασία MCP παράγεται το πλαίσιο σφάλματος, το οποίο στην περίπτωση του κωδικοποιητή HEVC, όπως και των περισσότερων κωδικοποιητών, αποτελεί την είσοδο του συστήματος μετασχηματισμού, ενώ τα διανύσματα κίνησης

αποστέλλονται στον αποκωδικοποιητή. Στο σχήμα 3.1 βλέπουμε την δομή της διαδικασίας MCP.



Σχήμα 3. 1: Σύστημα MCP

Διατυπώνοντας τα παραπάνω σε μορφή εξισώσεων, αν $A_t[m, n]$ το τωρινό πλαίσιο, $A_{t-1}[m, n]$ το πλαίσιο αναφοράς και $[d_m, d_n]$ το διάνυσμα κίνησης που έχει προκύψει από την διαδικασία εκτίμησης κίνησης, τότε η έξοδος της διαδικασίας MCP θα είναι το πλαίσιο σφάλματος $H_t[m, n]$ που δίνεται από τη σχέση :

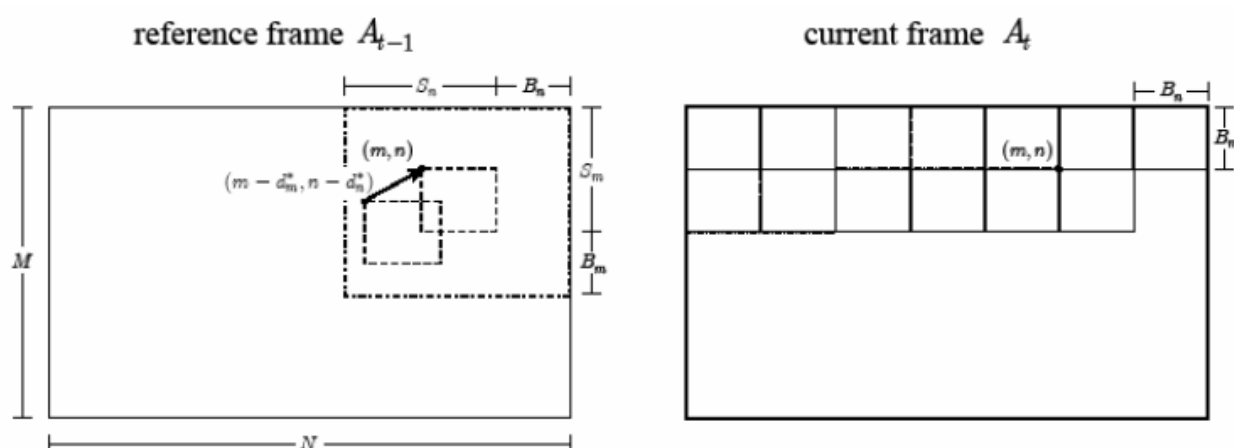
$$H_t[m, n] = A_t[m, n] - A_{t-1}[m - d_m, n - d_n]$$

Αν το τωρινό πλαίσιο και το πλαίσιο αναφοράς έχουν μεγάλη συσχέτιση μεταξύ τους, τότε το πλαίσιο σφάλματος θα περιέχει ένα μεγάλο αριθμό εικονοστοιχείων που έχουν τιμή κοντά στο μηδέν. Έτσι, το πλαίσιο σφάλματος θα έχει μικρότερη ενέργεια από το αρχικό πλαίσιο, οπότε θα έχει επιτευχτεί συμπίεση. Αυτή είναι και η μεγάλη συνεισφορά της διαδικασίας MCP. Στην περίπτωση που το τωρινό πλαίσιο

και το πλαίσιο αναφοράς είναι αρκετά διαφορετικά μεταξύ τους, τότε η διαδικασία MCP δεν θα επιφέρει πλεονέκτημα όσον αφορά τη συμπίεση. Συνήθως αυτό συμβαίνει όταν έχουμε εμφάνιση νέων αντικειμένων στην σκηνή, όταν έχουμε αλλαγή φωτεινότητας ή αλλαγή στο ζουμ της κάμερας.

Όπως αναφέρθηκε παραπάνω, η εξαγωγή των διανυσμάτων κίνησης γίνεται μέσω της διαδικασίας Εκτίμησης Κίνησης (Motion Estimation). Η μέθοδος που έχει επικρατήσει για την υλοποίηση της διαδικασίας αυτής είναι η Εκτίμηση Κίνησης κατά Block (Block-Based Motion Estimation).

Το τρέχον πλαίσιο $A_t[m, n]$, που αποτελείται από $M \times N$ εικονοστοιχεία, χωρίζεται σε μη επικαλυπτόμενα Blocks διαστάσεων $B_m \times B_n$ εικονοστοιχείων (για το HEVC $B_m, B_n = 4, 8, 16, 32$ ή 64). Για κάθε ένα από αυτά τα Blocks γίνεται αναζήτηση σε μια περιοχή $S_m \times S_n$ εικονοστοιχείων στο πλαίσιο αναφοράς και επιλέγεται με κάποιο κριτήριο ταιριάσματος το Block που ταιριάζει καλύτερα στο τρέχον Block, και με βάση αυτό υπολογίζονται το διάνυσμα κίνησης και το Block σφάλματος. Η διαδικασία της αναζήτησης φαίνεται παραστατικά στο σχήμα 3.2.



Σχήμα 3. 2: Εκτίμηση Κίνησης κατά Block

Η διαδικασία αυτή κάνει δύο βασικές παραδοχές:

- a) Η κίνηση όλων των εικονοστοιχείων που ανήκουν στο ίδιο Block είναι ίδια. Ισοδύναμα, το μέγεθος ενός αντικειμένου σε μια περιοχή της εικόνας είναι μεγαλύτερο από το μέγεθος του Block που επιλέγεται για την ίδια περιοχή.
- b) Υπάρχει μια μέγιστη απόσταση, την οποία μπορεί να καλύψει ένα αντικείμενο μεταξύ του τρέχοντος πλαισίου και του πλαισίου αναφοράς. Η υπόθεση αυτή περιορίζει τον αριθμό των εξεταζόμενων Blocks στο πλαίσιο αναφοράς, συνεπώς και την πολυπλοκότητα της αναζήτησης.

Το κριτήριο που χρησιμοποιείται κατά κύριο λόγο για την εύρεση της καλύτερης πρόβλεψης είναι το άθροισμα απολύτων διαφορών (SAD), καθώς παρέχει προβλέψεις μεγάλης ακρίβειας χωρίς την ανάγκη εκτέλεσης πολλαπλασιασμών. Έτσι λοιπόν επιλέγουμε ως Block πρόβλεψης εκείνο το Block του πλαισίου αναφοράς για το οποίο ελαχιστοποιείται το SAD.

Διατυπώνοντας μαθηματικά τα παραπάνω έχουμε

$$(d_m^*, d_n^*) : \arg \min_{(d_m, d_n)} \sum_{i=0}^{B_m-1} \sum_{j=0}^{B_n-1} C(A_t[m+i, n+j] - A_{t-1}[m+i-d_m, n+j-d_n])$$

όπου d_m , d_n ανήκουν στα διαστήματα $[-\frac{Sm}{2}, \frac{Sm}{2}]$ και $[-\frac{Sn}{2}, \frac{Sn}{2}]$ αντίστοιχα, ενώ $C(x)=|x|$, στην περίπτωση που χρησιμοποιούμε το SAD κριτήριο. Εναλλακτικό κριτήριο αποτελεί το άθροισμα τετραγωνικών σφαλμάτων (SSE), όπου $C(x)=x^2$, το οποίο όμως δεν προτιμάται λόγω του ότι απαιτεί την εκτέλεση πολλαπλασιασμών.

Η κλασικότερη μέθοδος επίλυσης του παραπάνω προβλήματος βελτιστοποίησης είναι ο αλγόριθμος πλήρους αναζήτησης (Full Search), στον οποίο εξετάζονται όλα τα πιθανά Blocks εντός των διαστημάτων $[-\frac{Sm}{2}, \frac{Sm}{2}]$ και $[-\frac{Sn}{2}, \frac{Sn}{2}]$ και επιλέγεται

αυτό που ελαχιστοποιεί το SAD. Ο αλγόριθμος αυτός, αν και παράγει πολύ καλές προβλέψεις και είναι κατάλληλος για VLSI υλοποιήσεις, εντούτοις είναι ιδιαίτερα απαιτητικός από άποψη υπολογισμών. Αν υποθέσουμε μέγιστη μετατόπιση w εικονοστοιχείων προς κάθε κατεύθυνση, τότε για την εύρεση της βέλτιστης λύσης απαιτούνται $(2w+1)^2 B_m B_n$ πράξεις απόλυτης διαφοράς και άθροισης. Για τον λόγο αυτό αναπτύχθηκαν διάφοροι αλγόριθμοι οι οποίοι μειώνουν το υπολογιστικό κόστος. Οι αλγόριθμοι αυτοί μπορούν να χωριστούν σε δύο βασικές κατηγορίες:

a) Αλγόριθμοι κατευθυντικής αναζήτησης (directional search) οι οποίοι μειώνουν τον αριθμό των Blocks που εξετάζονται. Πιο συγκεκριμένα, στην αρχή επιλέγονται συγκεκριμένα Blocks εντός της περιοχής αναζήτησης ως πιθανές προβλέψεις. Τα Blocks αυτά συνήθως σχηματίζουν νοητά ένα συγκεκριμένο μοτίβο. Για αυτά τα Blocks υπολογίζεται το SAD. Η αναζήτηση επαναλαμβάνεται με κέντρο την προηγούμενη καλύτερη πρόβλεψη για μοτίβο μικρότερης διάστασης, μέχρις ότου φτάσουμε σε μια ελάχιστη διάσταση, όπου και γίνεται η τελική αναζήτηση. Οι αλγόριθμοι αυτοί βασίζονται στην υπόθεση ότι το SAD είναι μονότονη συνάρτηση των d_m , d_n , κάτι που ωστόσο πολλές φορές δεν ισχύει, και έτσι οι αλγόριθμοι αυτοί παγιδεύονται σε λύσεις αρκετά χειρότερες από την βέλτιστη. Τέτοιοι αλγόριθμοι παρουσιάζονται στα [20],[24],[26],[28],[29],[32].

b) Ιεραρχικοί αλγόριθμοι (Hierarchical) οι οποίοι μειώνουν τον αριθμό των εικονοστοιχείων που λαμβάνονται υπόψιν σε κάθε σύγκριση μεταξύ δύο Blocks. Πιο συγκεκριμένα, η αναζήτηση χωρίζεται σε L στάδια, όπου συνήθως $L=2^l$, $l>0$. Στο πρώτο στάδιο, καλύπτεται όλη η περιοχή αναζήτησης, για τον υπολογισμό των SADs όμως χρησιμοποιούνται L λιγότερα εικονοστοιχεία από κάθε Block. Στη συνέχεια, σε κάθε επόμενο στάδιο γίνεται αναζήτηση σε μια κοντινή στην πρόβλεψη του προηγούμενου σταδίου περιοχή, χρησιμοποιώντας όμως διπλάσιο αριθμό

εικονοστοιχείων από ότι στο προηγούμενο επίπεδο. Το κατάλληλο Block πρόβλεψης επιλέγεται από το τελευταίο στάδιο, όπου χρησιμοποιούνται όλα τα εικονοστοιχεία. Για την παραγωγή των Blocks μειωμένων στοιχείων που απαιτούνται στα πρώτα L-1 στάδια, εκτελείται είτε υποδειγματοληψία των Blocks προς τις δύο κατευθύνσεις, είτε χρήση κάποιου κατωδιαβατού φίλτρου, ώστε μια ομάδα από εικονοστοιχεία να αντιπροσωπεύεται πλέον από ένα. Στην πρώτη περίπτωση η λύση η οποία επιλέγεται έχει τα ίδια προβλήματα με τους αλγόριθμους της προηγούμενης κατηγορίας, ενώ στην δεύτερη, αν και προσεγγίζεται καλύτερα η βέλτιστη λύση, εντούτοις απαιτείται σημαντική αύξηση της μνήμης του συστήματος, καθώς απαιτείται η ταυτόχρονη αποθήκευση τόσο της αρχικής όσο και της εικόνας που ελήφθη με υποδειγματοληψία. Τέτοιοι αλγόριθμοι παρουσιάζονται στα [21],[31].

Όποιος από τους παραπάνω αλγόριθμους κι αν χρησιμοποιηθεί, η αποτελεσματικότητα της διαδικασίας Motion Estimation επηρεάζεται σε μεγάλο βαθμό από τρεις παραμέτρους:

α) Τις διαστάσεις του Block. Αν το Block είναι μικρό, τότε επιτυγχάνεται καλύτερη ποιότητα πρόβλεψης. Αυτό συμβαίνει γιατί όσο μικρότερο είναι ένα Block, τόσο λιγότερες είναι οι πιθανότητες να συνυπάρχουν μέσα σε αυτό δύο αντικείμενα κινούμενα σε διαφορετικές κατευθύνσεις. Το τίμημα, ωστόσο, της χρήσης μικρών σε μέγεθος Blocks, και συνεπώς της πολύ καλής πρόβλεψης, είναι το γεγονός ότι θα απαιτηθεί μεγαλύτερος αριθμός Blocks για να καλυφθεί μια συγκεκριμένη περιοχή της εικόνας, άρα και περισσότερα διανύσματα κίνησης, κάτι που δρα αρνητικά στην συμπίεση του σήματος. Αντίθετα, Blocks μεγάλου μεγέθους οδηγούν σε μείωση των διανυσμάτων κίνησης, με αντίκτυπο την υποβάθμιση της ποιότητας της πρόβλεψης. Συνεπώς ο κωδικοποιητής οφείλει να επιλέγει κάθε φορά τις διαστάσεις του Block, έτσι ώστε να ελαχιστοποιείται η συνολική πληροφορία, δηλαδή το σύνολο των bits που απαιτούνται για να μεταδοθούν τόσο τα στοιχεία του Block σφάλματος, όσο και τα διανύσματα κίνησης.

b) Την μέγιστη απόσταση που μπορεί να καλύψει κάποιο αντικείμενο. Σε περίπτωση που επιλεγεί μεγάλη τιμή, τότε υπάρχει μεγαλύτερη πιθανότητα να βρεθεί η βέλτιστη λύση στην εικόνα αναφοράς. Ωστόσο, η μέγιστη αυτή απόσταση αυξάνει σε μεγάλο βαθμό την υπολογιστική πολυπλοκότητα, καθώς, αν για παράδειγμα επιλεγεί εξαντλητική αναζήτηση, οι απαιτούμενες αριθμητικές πράξεις είναι ανάλογες με το τετράγωνο της μέγιστης απόστασης. Στις περισσότερες περιπτώσεις μια μέγιστη απόσταση $w=15$ pixels επιφέρει ικανοποιητικά αποτελέσματα.

c) Την ακρίβεια της αναζήτησης. Παρότι αρχικά η διαδικασία του Motion Estimation αναπτύχθηκε για ακρίβεια αναζήτησης ενός εικονοστοιχείου (integer pixel accuracy), γρήγορα αποδείχτηκε ότι αυτό περιορίζει την ποιότητα της πρόβλεψης, αφού οι μετακινήσεις των αντικειμένων στην εικόνα είναι ανεξάρτητες από την χωρική συχνότητα δειγματοληψίας της. Μάλιστα, αποδείχτηκε ότι η αναζήτηση ακρίβειας μικρότερης του ενός εικονοστοιχείου μπορεί να επιφέρει κέρδη περίπου 2db στην ενέργεια του Block σφάλματος. Για να υλοποιηθεί λοιπόν αυτή η διαδικασία, απαιτείται η εικόνα αναφοράς να περάσει από κάποιο φίλτρο παρεμβολής, ώστε να αποκτήσει την απαιτούμενη ακρίβεια. Ωστόσο, αυτό σημαίνει ότι τα pixel της εικόνας αναφοράς θα αυξηθούν ανάλογα με την αύξηση της ακρίβειας, κάτι που αφενός αυξάνει την απαιτούμενη μνήμη για την αποθήκευση της εικόνας, αφετέρου αυξάνει και των αριθμό των υποψήφιων Blocks που πρέπει να εξεταστούν. Οι περισσότεροι κωδικοποιητές λύνουν το τελευταίο πρόβλημα, εκτελώντας πρώτα αναζήτηση ακρίβειας ενός εικονοστοιχείου, και στη συνέχεια η αναζήτηση μεγαλύτερης ακρίβειας γίνεται μόνο σε μια περιοχή της προηγούμενης λύσης.

3.2 Υπάρχουσες Υλοποιήσεις

Στην εργασία [30] γίνεται μια περιγραφή των βασικότερων παράλληλων αρχιτεκτονικών που χρησιμοποιούνται για τον υπολογισμό του SAD.

Στην πρώτη αρχιτεκτονική έχουμε παράλληλο υπολογισμό απόλυτων διαφορών που αφορούν το ίδιο SAD, άρα και το ίδιο διάνυσμα κίνησης, και άθροισή τους μέσω ενός δέντρου αθροιστών (adder's tree). Στην δεύτερη αρχιτεκτονική έχουμε παράλληλο υπολογισμό SADs που αναφέρονται σε διαφορετικά Blocks πρόβλεψης, άρα και διαφορετικά διανύσματα κίνησης, και συσσώρευση των αποτελεσμάτων σε διαφορετικούς καταχωρητές. Οι αρχιτεκτονικές του πρώτου τύπου παρουσιάζονται επίσης και στις εργασίες [23],[25],[27], όπου ο υπολογισμός κάθε SAD $N_1 \times N_2$ εικονοστοιχείων διαχωρίζεται σε υπολογισμούς SAD 4×4 εικονοστοιχείων οι οποίοι γίνονται παράλληλα και αθροίζονται για να παράγουν SAD όλων των πιθανών διαστάσεων, από 4×4 έως 64×64 . Η τεχνική αυτή εκτιμήθηκε ότι αυξάνει το ρυθμό επεξεργασίας κατά 16 φορές, καθώς υπολογίζονται παράλληλα 16 απόλυτες διαφορές ώστε να προκύψει το SAD 4×4 εικονοστοιχείων, ενώ την ίδια στιγμή αυξάνονται ανάλογα τόσο η επιφάνεια ολοκλήρωσης, όσο και η κατανάλωση ενέργειας κατά 16 φορές, όσον αφορά το κύκλωμα υπολογισμού του SAD. Βέβαια και σε αυτή την περίπτωση, μπορεί να επιλεχτεί αύξηση της χρονικής διάρκειας κύκλου ρολογιού, ώστε να επιτευχτεί μείωση της κατανάλωσης ενέργειας, ταυτόχρονα όμως θα έχουμε μείωση του κέρδους στον ρυθμό επεξεργασίας.

Στην εργασία [22] παρουσιάζεται ένας αλγόριθμος κατευθυντικής αναζήτησης δύο σταδίων, με δύο ξεχωριστές μνήμες για την αποθήκευση των περιοχών αναζήτησης που χρειάζεται κάθε στάδιο. Στο πρώτο γίνεται αναζήτηση σε μια περιοχή 64 εικονοστοιχείων προς κάθε κατεύθυνση, αλλά μόνο ανά 8 Blocks, ενώ στο δεύτερο στάδιο γίνεται αναζήτηση σε μια περιοχή 7 εικονοστοιχείων προς κάθε κατεύθυνση γύρω από την καλύτερη πρόβλεψη του προηγούμενου σταδίου. Επιπλέον, η

εργασία προτείνει τρόπους κοινής χρήσης της ίδιας μνήμης και από τα δύο στάδια αναζήτησης, ενώ τέλος προτείνεται η χρήση on-chip μνήμης που θα διατηρεί μόνο τα απαραίτητα για την αναζήτηση εικονοστοιχεία και όχι ολόκληρες τις εικόνες, καθώς η καθυστέρηση προσπέλασης της εξωτερικής μνήμης στην οποία είναι αποθηκευμένες οι εικόνες μπορεί να φτάσει μέχρι και χιλιάδες κύκλους ρολογιού. Στην υλοποίηση αυτή εκτιμήθηκε ότι ο μέγιστος αριθμός των blocks που εξετάζονται $225+225=450$ blocks σε μια περιοχή 64 εικονοστοιχείων προς κάθε κατεύθυνση. Η επιλογή αυτή οδηγεί σε χρόνο εκτέλεσης $450T_{SAD}$, δηλαδή περίπου στο μισό από αυτόν μιας κλασικής υλοποίησης πλήρους αναζήτησης σε μια περιοχή 16 εικονοστοιχείων προς κάθε κατεύθυνση. Όσον αφορά την ποιότητα της πρόβλεψης, παρουσιάζει τα συνηθισμένα προβλήματα των αλγορίθμων κατευθυντικής αναζήτησης, τα οποία ωστόσο αντισταθμίζονται σε μεγάλο βαθμό από το γεγονός ότι η αναζήτηση φτάνει στην εξαιρετικά μεγάλη απόσταση των 64 εικονοστοιχείων προς κάθε κατεύθυνση.

3.3 Προτεινόμενη Υλοποίηση

Η υλοποίηση έγινε σε επίπεδο καταχωρητή (RTL) με χρήση του εργαλείου Simulink του πακέτου Matlab, και βασίστηκε στον αλγόριθμο πλήρους αναζήτησης ταιριάσματος μπλοκ (Full Search Block Matching Algorithm), ο οποίος περιγράφεται παρακάτω:

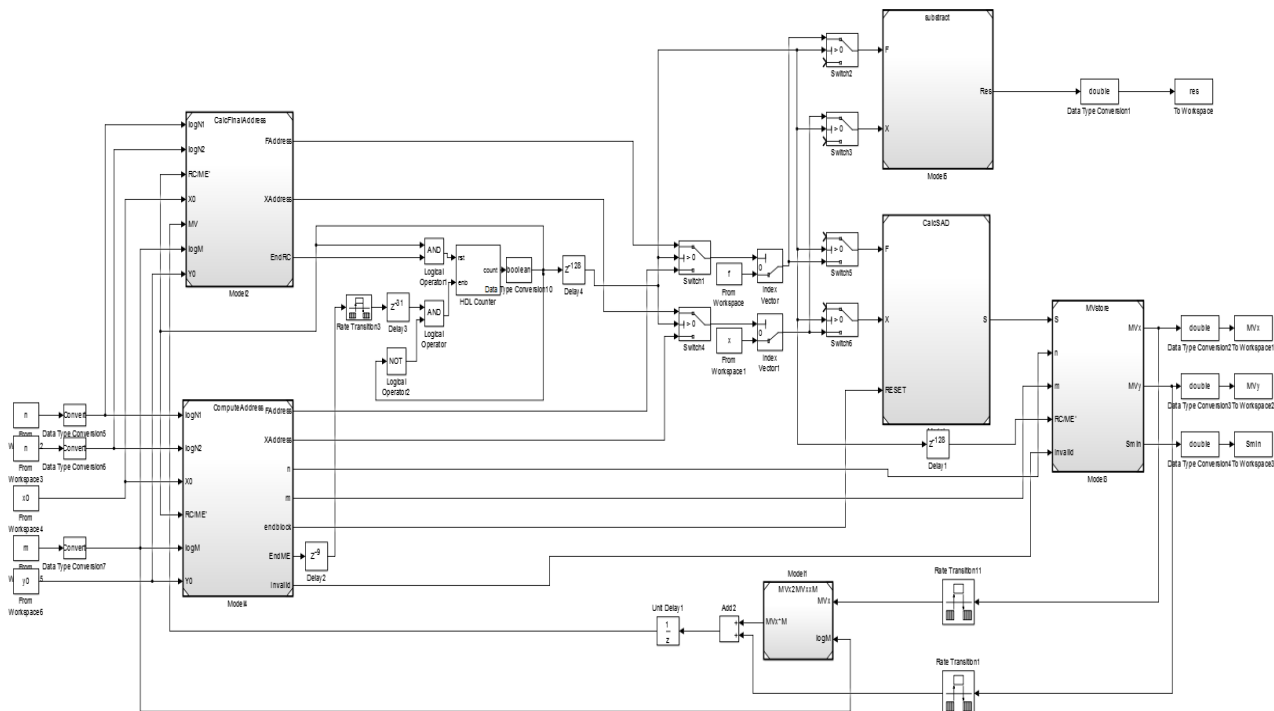
Αν $A_t[m, n]$ το τωρινό πλαίσιο, $A_{t-1}[m, n]$ το πλαίσιο αναφοράς, x_0 και y_0 η θέση του τωρινού block στο πλαίσιο A_t και n_1, n_2 οι διαστάσεις του block, τότε για κάθε d_m, d_n που ανήκουν σε ένα σύνολο $[-p, p]$ υπολογίζεται το άθροισμα απολύτων διαφορών (SAD). Το διάνυσμα κίνησης $[d_m, d_n]$ επιλέγεται έτσι ώστε να ελαχιστοποιεί το εν λόγω κριτήριο, ενώ ως block πρόβλεψης επιλέγεται το block $A_{t-1}[x_0 + m - d_m, y_0 + n - d_n]$, όπου $0 \leq m < n_1, 0 \leq n < n_2$.

Στη συνέχεια, υπολογίζεται το block σφάλματος (residual block):

$$H_t[m, n] = A_t[x_0 + m, y_0 + n] - A_{t-1}[x_0 + m - d_m, y_0 + n - d_n]$$

το οποίο αποστέλλεται για μετασχηματισμό, ενώ το διάνυσμα κίνησης στέλνεται στον αποκωδικοποιητή.

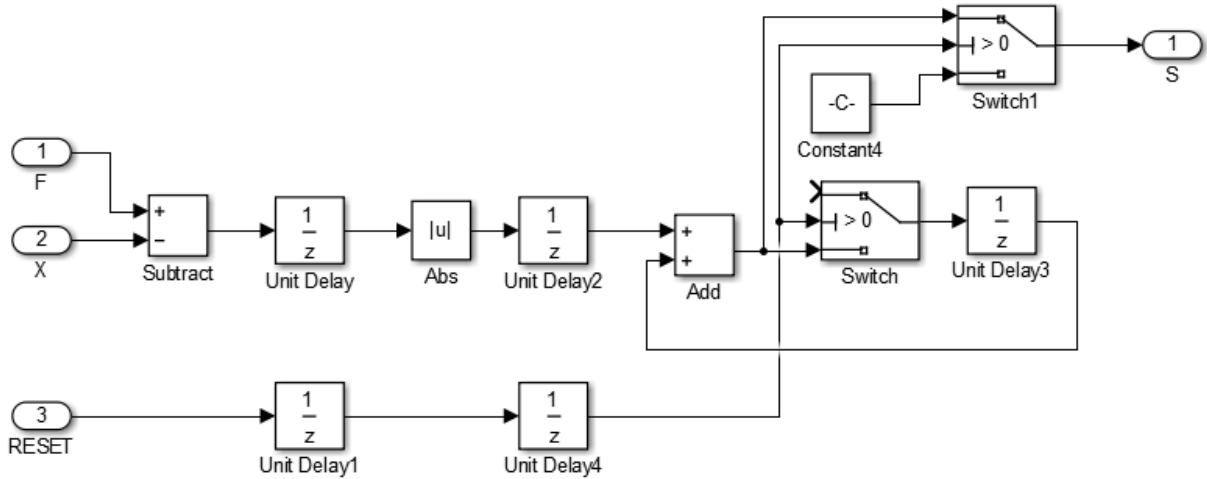
Ο αλγόριθμος αυτός παράγει καλύτερες προβλέψεις συγκριτικά με οποιονδήποτε άλλον από τους υπάρχοντες αλγορίθμους, ωστόσο είναι ιδιαίτερα απαιτητικός όσον αφορά τον χρόνο εκτέλεσης, καταλαμβάνοντας μάλιστα περίπου το 80% του συνολικού χρόνου σε έναν υβριδικό κωδικοποιητή όπως αυτός που περιγράφεται στο πρότυπο HEVC. Πιο συγκεκριμένα, απαιτείται να υπολογιστεί το SAD για κάθε ένα από τα blocks που βρίσκονται μέσα στην περιοχή αναζήτησης (σύνολο $(2p+1)^2$ blocks). Επειδή ο υπολογισμός ενός SAD μεταξύ δύο blocks $n_1 \times n_2$ εικονοστοιχείων απαιτεί $n_1 \times n_2$ υπολογισμούς απόλυτης διαφοράς μεταξύ δύο εικονοστοιχείων και $n_1 \times n_2$ αθροίσεις, ο συνολικός απαιτούμενος χρόνος για την εύρεση του διανύσματος κίνησης είναι $(2p+1)^2 n_1 n_2 (3T_{add})$, όπου T_{add} ο χρόνος εκτέλεσης ενός αθροιστή. Στο σχήμα 3.3 φαίνεται το block διάγραμμα του συστήματος MCP.



Σχήμα 3. 3 : block διάγραμμα του συστήματος MCP

Στόχος της παρούσας υλοποίησης είναι να μειωθεί ο παραπάνω χρόνος εκτέλεσης, χωρίς ταυτόχρονα να υποβαθμίζεται η ποιότητα των προβλέψεων του συγκεκριμένου αλγορίθμου. Για τον λόγο αυτό εφαρμόζονται οι τεχνικές *pipelining* και *parallel processing*.

Αρχικά, σε κάθε κύτταρο SAD τοποθετείται ένας καταχωρητής μεταξύ του υπολογισμού της διαφοράς και της απόλυτης τιμής και άλλος ένας καταχωρητής μεταξύ της απόλυτης τιμής και της άθροισης. Με τον τρόπο αυτόν ο ρυθμός επεξεργασίας δεδομένων του κυττάρου μπορεί να τριπλασιαστεί και ο χρόνος υπολογισμού του συνολικού SAD μεταξύ δύο blocks $n_1 \times n_2$ εικονοστοιχείων μειώνεται από $n_1 n_2 \cdot (3T_{add})$ σε $(n_1 n_2 + 2) T_{add}$. Δηλαδή μειώνεται κατά περίπου 66%. Το προτεινόμενο βασικό κύκλωμα υπολογισμού ενός SAD φαίνεται στο σχήμα 3.4.



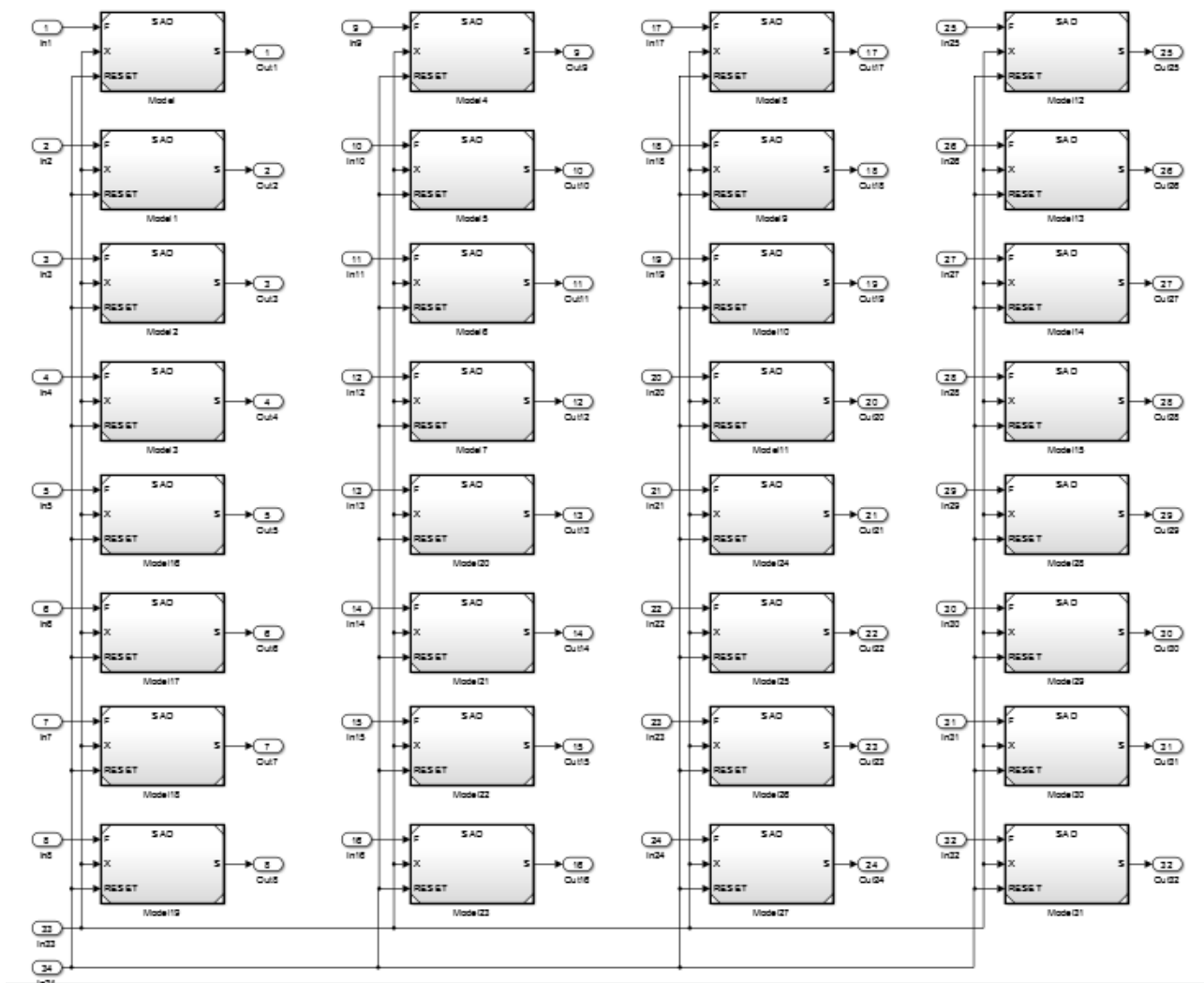
Σχήμα 3. 4: Προτεινόμενο κύτταρο SAD

Γενικά ο αλγόριθμος πλήρους αναζήτησης προσφέρεται για παράλληλη επεξεργασία, καθώς απαιτείται να γίνεται επαναληπτικά η ίδια διαδικασία και το μόνο που αλλάζει κάθε φορά είναι τα d_m , d_n . Έτσι λοιπόν, στην παρούσα υλοποίηση επιλέγεται ο παραλληλισμός 32 επιπέδων για το κύτταρο SAD, ώστε να υπολογίζονται ταυτόχρονα 32 από τα $(2p+1)^2$ SADs. Πιο συγκεκριμένα, αν θέσουμε:

$$d_m = 8d_{m1} + d_{m2} \quad \frac{-p}{8} \leq d_{m1} < \frac{p}{8}, \quad 0 \leq d_{m2} < 8$$

$$d_n = 4d_{n1} + d_{n2} \quad \frac{-p}{4} \leq d_{n1} < \frac{p}{4}, \quad 0 \leq d_{n2} < 4$$

Τότε αν τα SADs για κάθε τιμή των d_{m2} , d_{n2} υπολογιστούν ταυτόχρονα, ο χρόνος υπολογισμού των $(2p+1)^2$ SADs μειώνεται από $(2p+1)^2 T_{SAD}$ σε $(2p+1)^2 T_{SAD}/32$, δηλαδή μειώνεται κατά 32 φορές. Το ολοκληρωμένο κύκλωμα υπολογισμού των 32 SADs παράλληλα φαίνεται στο σχήμα 3.5



Σχήμα 3. 5: 32 level parallel κύκλωμα υπολογισμού SAD

Τελικά λοιπόν, ο συνολικός χρόνος εκτέλεσης του συστήματος υπολογισμού του SAD που έχει υποστεί 3-level pipelining και 32-level parallel processing είναι 96 φορές μικρότερος από αυτόν του αρχικού συστήματος. Βέβαια, το αντίτιμο για την βελτίωση αυτή είναι η αύξηση της επιφάνειας ολοκλήρωσης, καθώς για την υλοποίηση του νέου συστήματος απαιτούνται 32 φορές περισσότεροι αθροιστές (συγκεκριμένα 96 έναντι μόλις 3 στο αρχικό σύστημα) και 96 φορές περισσότεροι καταχωρητές (συγκεκριμένα 96 έναντι ενός). Ωστόσο ο συμβιβασμός αυτός γίνεται επειδή το κρίσιμο στην διαδικασία MCP είναι η μείωση του χρόνου εκτέλεσης.

Ακόμα, πρέπει να σημειωθεί ότι για να υποστηριχτεί η δυνατότητα του συστήματος υπολογισμού του SAD να επεξεργάζεται 32 δείγματα ανά T_{add} , πρέπει και το κύκλωμα που δημιουργεί τις διευθύνσεις προσπέλασης στην μνήμη που εμπεριέχει την εικόνα αναφοράς να δύναται να δημιουργήσει 32 διευθύνσεις ανά T_{add} . Με άλλα λόγια, θα πρέπει οι διευθύνσεις

$$[x_0 + m - d_m, y_0 + n - d_n] = [x_0 + m - (8d_{m1} + d_{m2}), y_0 + n - (4d_{n1} + d_{n2})]$$

να υπολογίζονται παράλληλα για κάθε τιμή των d_{m2} , d_{n2}

Αν υποθέσουμε ότι το μέγεθος της εικόνας είναι M , όπου M μια δύναμη του 2, τότε οι διευθύνσεις στην μνήμη θα είναι

$$[(x_0 + m - (8d_{m1} + d_{m2}))M + y_0 + n - (4d_{n1} + d_{n2})]$$

Αν υποθέσουμε ότι η τιμή $c_0 = x_0 + My_0$ είναι σταθερή για όλη τη διαδικασία και έχει προϋπολογιστεί, τότε η τελική διεύθυνση είναι:

$$[c_0 + (m - (8d_{m1} + d_{m2}))M + n - (4d_{n1} + d_{n2})] =$$

$$= [c_0 + M(m - 8d_{m1}) + n - 4d_{n1} + Md_{m2} + d_{n2}]$$

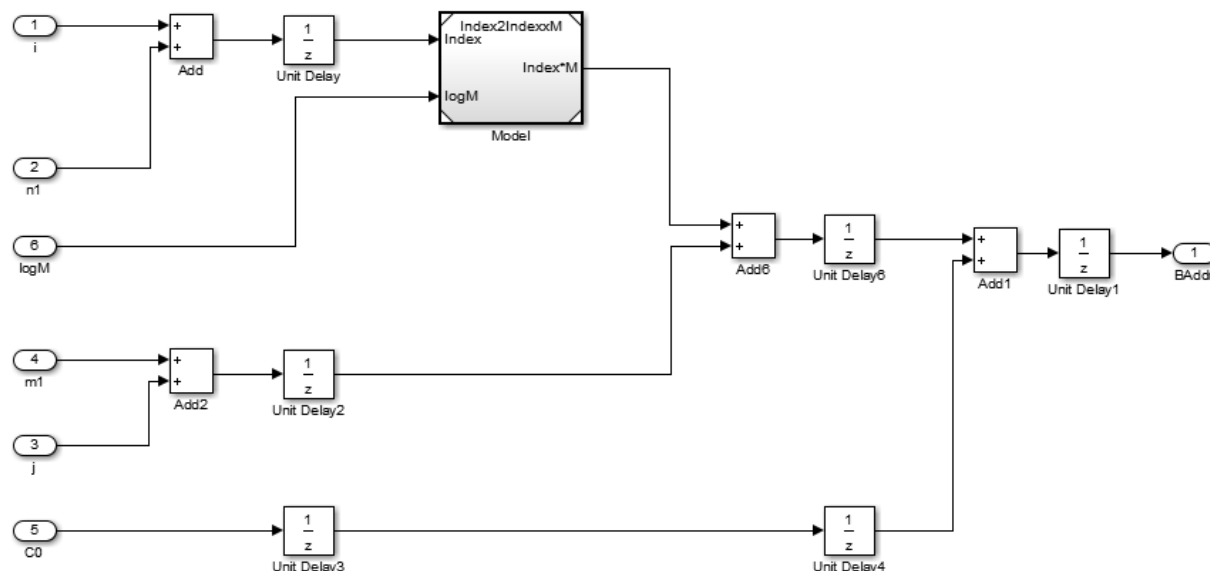
Αν θέσουμε

$$\text{BaseAddress} = c_0 + M(m - 8d_{m1}) + n - 4d_{n1}$$

τότε τελικά η διεύθυνση γίνεται

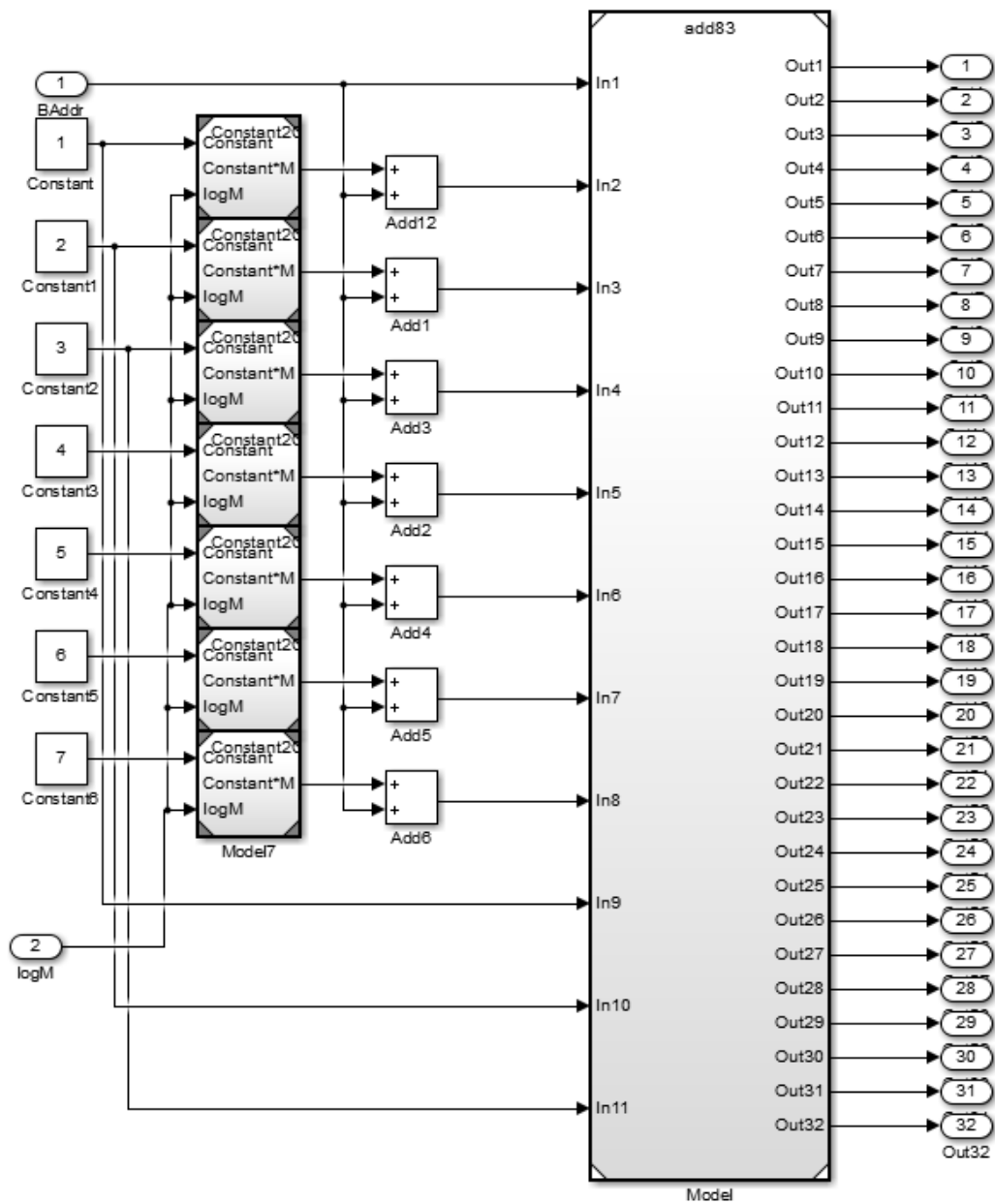
$$[\text{BaseAddress} + Md_{m2} + d_{n2}]$$

Η BaseAddress χρειάζεται 4 αθροίσεις για να υπολογιστεί και παραμένει σταθερή κατά τη διάρκεια ενός κύκλου ρολογιού, συνεπώς αρκεί η χρήση pipelining ώστε να υπολογίζεται μια BaseAddress ανά T_{add} . Το κύκλωμα υπολογισμού της BaseAddress φαίνεται στο σχήμα 3.6

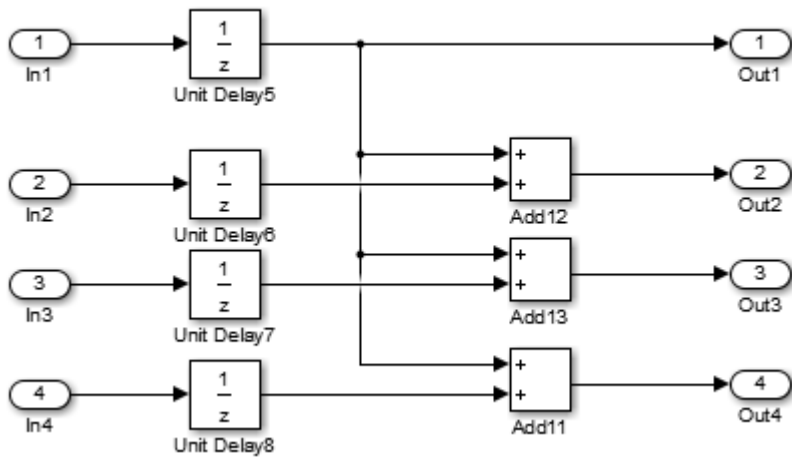


Σχήμα 3. 6: Κύκλωμα υπολογισμού BaseAddress

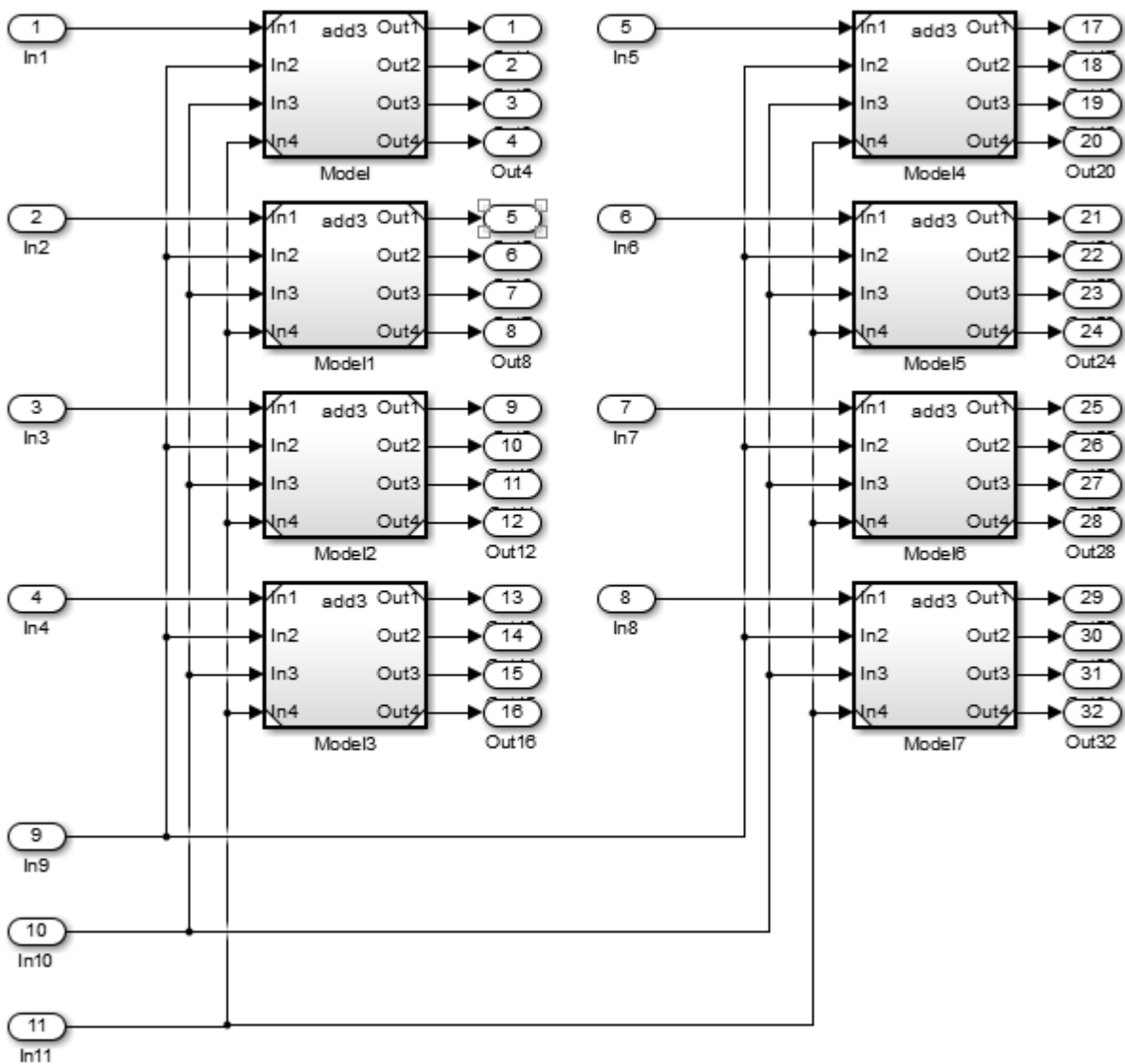
Για τον υπολογισμό της τελικής διεύθυνσης απαιτούνται αρχικά 7 προσθέσεις, αφού $0 \leq d_{m2} \leq 7$, ενώ στη συνέχεια σε καθένα από τα 8 αποτελέσματα πρέπει να προστεθούν οι πιθανές τιμές του d_{n2} , δηλαδή άλλες $3 \times 8 = 24$ αθροίσεις. Οπότε συνολικά το κύκλωμα χρειάζεται $7 + 24 + 4 = 35$ προσθέσεις για να υπολογίσει παράλληλα 32 διευθύνσεις στην μνήμη του πλαισίου αναφοράς. Τα απαραίτητα κυκλώματα υπολογισμού των τελικών διευθύνσεων του πλαισίου αναφοράς φαίνονται στα σχήματα 3.7, 3.8 και 3.9.



Σχήμα 3. 7 : Κύκλωμα πρόσθεσης κάθε πιθανής τιμής του d_{m2} στην *BaseAddress*



Σχήμα 3. 8 : Κύκλωμα πρόσθεσης μιας τιμής του d_{n2} στην *BaseAddress*



Σχήμα 3. 9 : Κύκλωμα παράλληλης πρόσθεσης κάθε πιθανής τιμής του d_{n2} στην *BaseAddress*

Όσον αφορά τις διευθύνσεις του τωρινού πλαισίου, αυτές είναι της μορφής:

$$[c_0 + m \cdot M + n]$$

και πρέπει να δημιουργούνται με ρυθμό μία ανά κύκλο ρολογιού, συνεπώς αρκεί η εφαρμογή pipelining 2 επιπέδων για την επίτευξη του επιθυμητού throughput.

Τέλος, μετά τον υπολογισμό όλων των απαιτούμενων SADs και την λήψη απόφασης για το διάνυσμα κίνησης $[d_m, d_n]$, απαιτείται ο υπολογισμός του block σφάλματος :

$$H_t[m, n] = A_t[x_0 + m, y_0 + n] - A_{t-1}[x_0 + m - d_m, y_0 + n - d_n] =$$

$$= A_t[c_0 + M \cdot m + n] - A_{t-1}[c_0 + M \cdot (m - d_m) + n - d_n] =$$

$$= A_t[c_0 + M \cdot m + n] - A_{t-1}[c_0 + M \cdot m + n - (M \cdot d_m + d_n)]$$

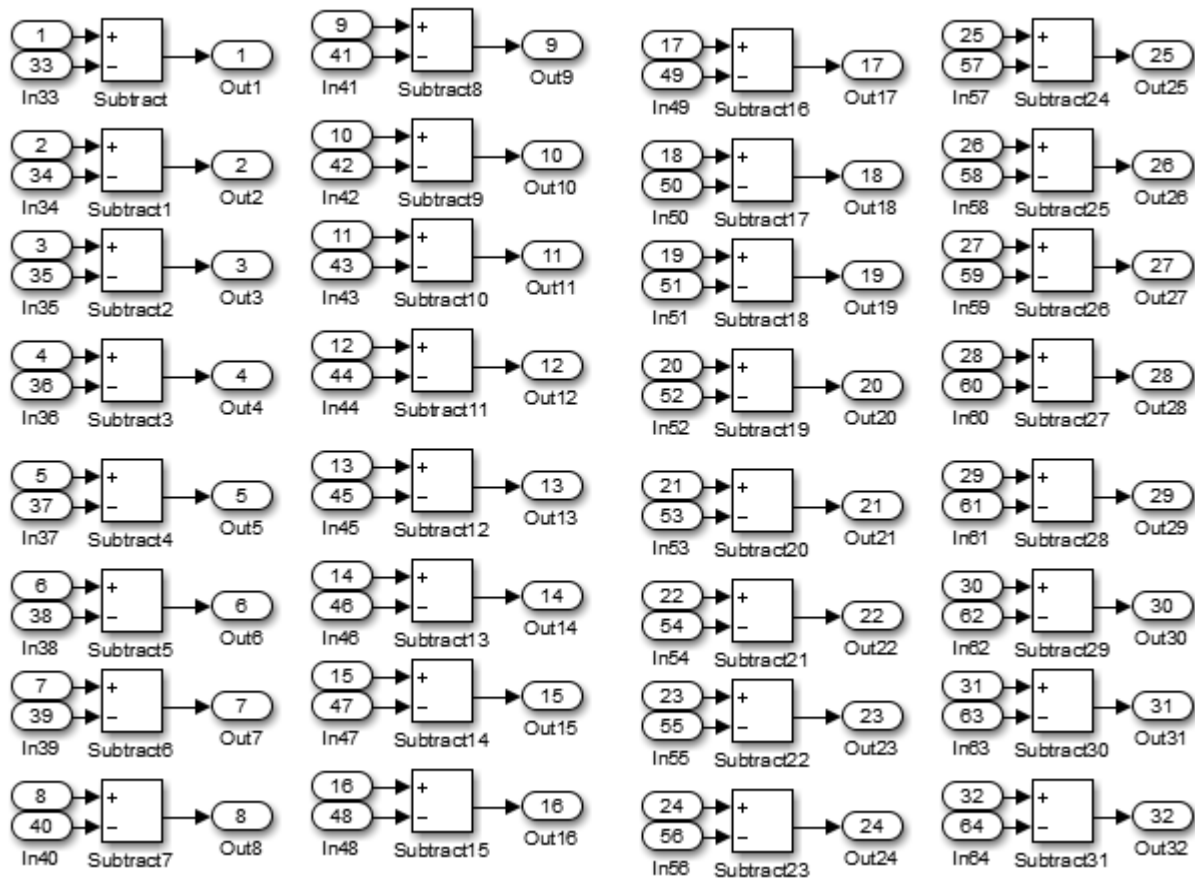
Επειδή οι τιμές :

$$c_0, M \cdot d_m + d_n$$

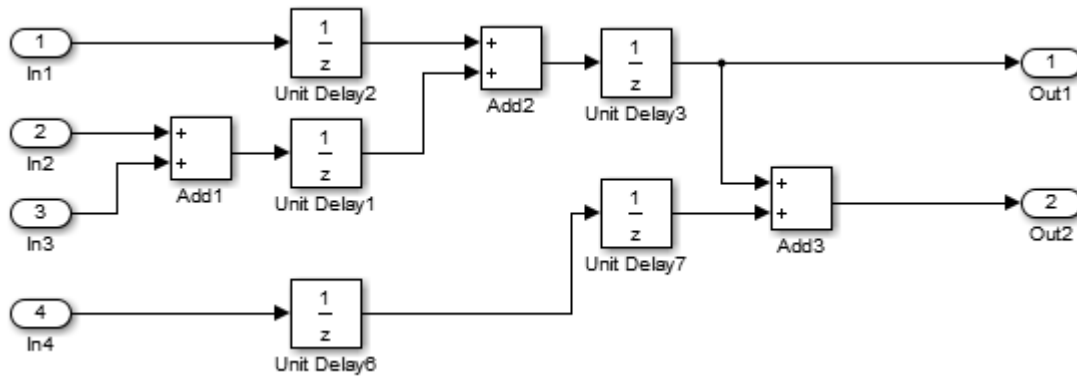
είναι σταθερές για δεδομένο block πρόβλεψης, μπορούν να έχουν προϋπολογιστεί.

Αν επιθυμούμε τα στοιχεία του block σφάλματος να υπολογίζονται με τον ίδιο ρυθμό που δέχεται δεδομένα ο DCT που σχεδιάστηκε στην παρούσα διπλωματική, καθώς αυτά δίνονται ως είσοδοι στον DCT, τότε θα πρέπει να υπολογίζονται 32 στοιχεία ανά T_{add} . Για να επιτευχτεί αυτό, απαιτείται εφαρμογή pipelining και parallel processing 32 επιπέδων τόσο στο σύστημα διευθυνσιοδότησης, όσο και στον υπολογισμό της διαφοράς. Η αρνητική συνέπεια της παραπάνω απόφασης είναι, και εδώ, η αύξηση της επιφάνειας ολοκλήρωσης, καθώς ο αριθμός των αθροιστών που απαιτούνται για τις δύο παραπάνω διαδικασίες αυξάνεται 32 φορές (συγκεκριμένα

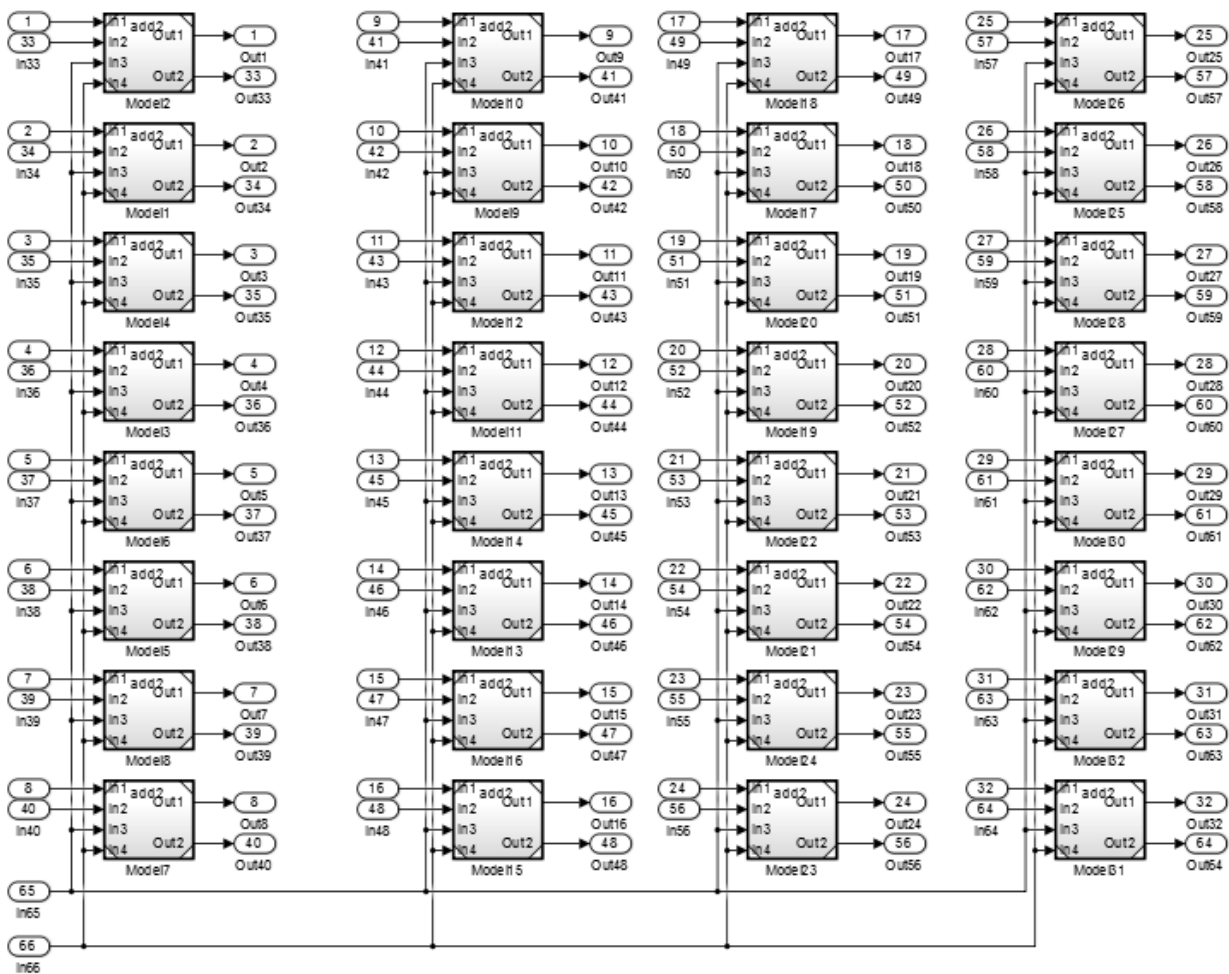
128 έναντι 4 αθροιστών σε μια συμβατική υλοποίηση). Το κύκλωμα υπολογισμού του block σφάλματος, καθώς και το βασικό δομικό κύκλωμα υπολογισμού μιας διεύθυνσης στη μνήμη του block πρόβλεψης και του τωρινού block φαίνονται αντίστοιχα στα σχήματα 3.10, 3.11 και 3.12.



Σχήμα 3.10: Κύκλωμα υπολογισμού των στοιχείων του block σφάλματος



Σχήμα 3.11: Βασικό κύκλωμα υπολογισμού μιας διεύθυνσης στη μνήμη του τωρινού block και του block πρόβλεψης



Σχήμα 3.12: Ολοκληρωμένο κύκλωμα παράλληλου υπολογισμού 32 διευθύνσεων στη μνήμη του τωρινού block και του block πρόβλεψης

Συνολικά λοιπόν, το προτεινόμενο σύστημα MCP επιτυγχάνει 96 φορές μικρότερο χρόνο εκτέλεσης από ότι ο κλασικός Full Search αλγόριθμος. Η μείωση αυτή του χρόνου εκτέλεσης έχει βέβαια αρνητικό αντίκτυπο τόσο στην επιφάνεια ολοκλήρωσης, όπου έχουμε αύξηση κατά 19 φορές (285 αθροιστές έναντι 15 στην κλασική υλοποίηση), όσο και στην κατανάλωση ενέργειας, όπου λόγω της μεγαλύτερης κατά 19 φορές συνολικής χωρητικότητας του κυκλώματος έχουμε αύξηση κατά 19 φορές. Η μεγάλη αυτή αύξηση της κατανάλωσης ενέργειας μπορεί να αντιμετωπιστεί με επιλογή κύκλου ρολογιού μεγαλύτερης διάρκειας, κάτι που ωστόσο θα επιφέρει αύξηση του χρόνου εκτέλεσης.

3.4 Πειραματικές Μετρήσεις

Για την μέτρηση της απόδοσης του συστήματος που υλοποιήθηκε, τρία συνεχόμενα πλαίσια 32x32 εικονοστοιχείων μιας ακολουθίας video χωρίστηκαν σε blocks διαστάσεων 8x8 εικονοστοιχείων, εφαρμόστηκε σε αυτά η διαδικασία MCP και τέλος σχηματίστηκαν το τωρινό frame $X(i, j)$, το frame πρόβλεψης $X_{pred}(i, j)$ και το frame σφάλματος $X(i, j) - X_{pred}(i, j)$ και μετρήθηκαν τα παρακάτω μεγέθη :

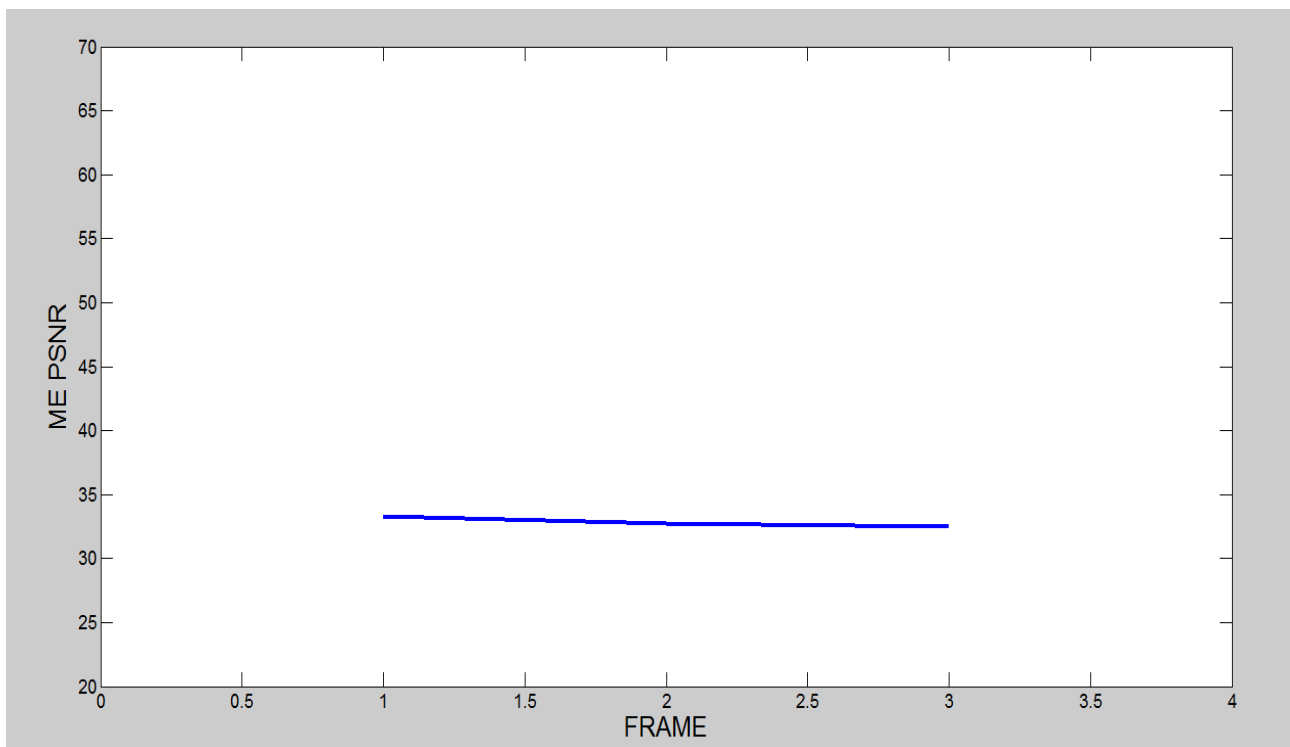
α)Λόγος Μέγιστου Σήματος προς Θόρυβο (Pick Signal to Noise Ratio, PSNR) που δίνεται από τον τύπο:

$$PSNR = \log_{10} \frac{M^2 255^2}{E_{residual}}$$

όπου $E_{residual}$ η ενέργεια του frame σφάλματος, η οποία δίνεται από

$$E_{residual} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} (X(i, j) - X_{pred}(i, j))^2$$

όπου $X(i, j)$ το τωρινό frame, $X_{\text{pred}}(i, j)$ το frame πρόβλεψης και M η διάσταση του frame. Μικρή ενέργεια frame σφάλματος, άρα και καλή πρόβλεψη, οδηγεί σε υψηλές τιμές του PSNR.

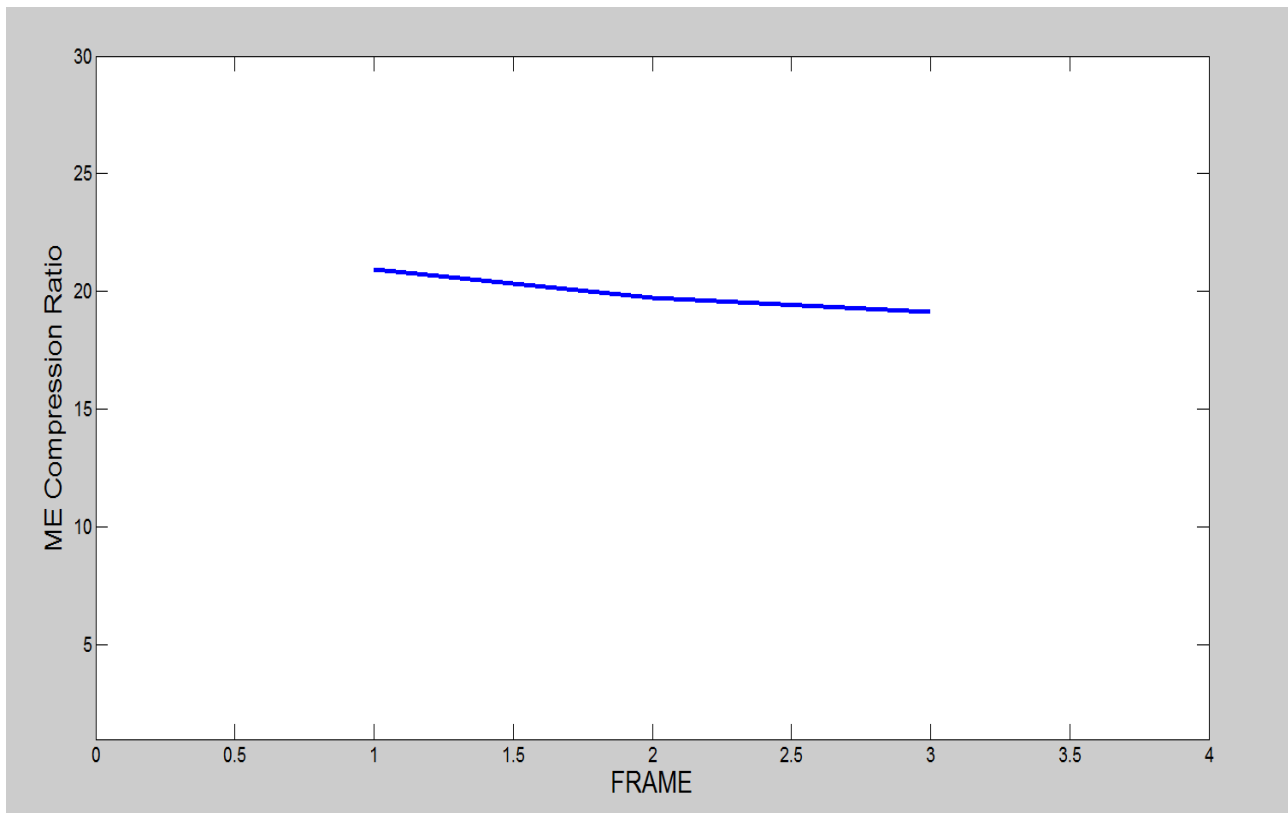


Γραφική Παράσταση 3.1 : PSNR πρόβλεψης για τρία συνεχόμενα πλαίσια

β) Λόγος συμπίεσης δεδομένων (Compression Ratio, CR), που δίνεται από τον τύπο:

$$CR = \sqrt{\frac{E_{\text{in}}}{E_{\text{out}}}}$$

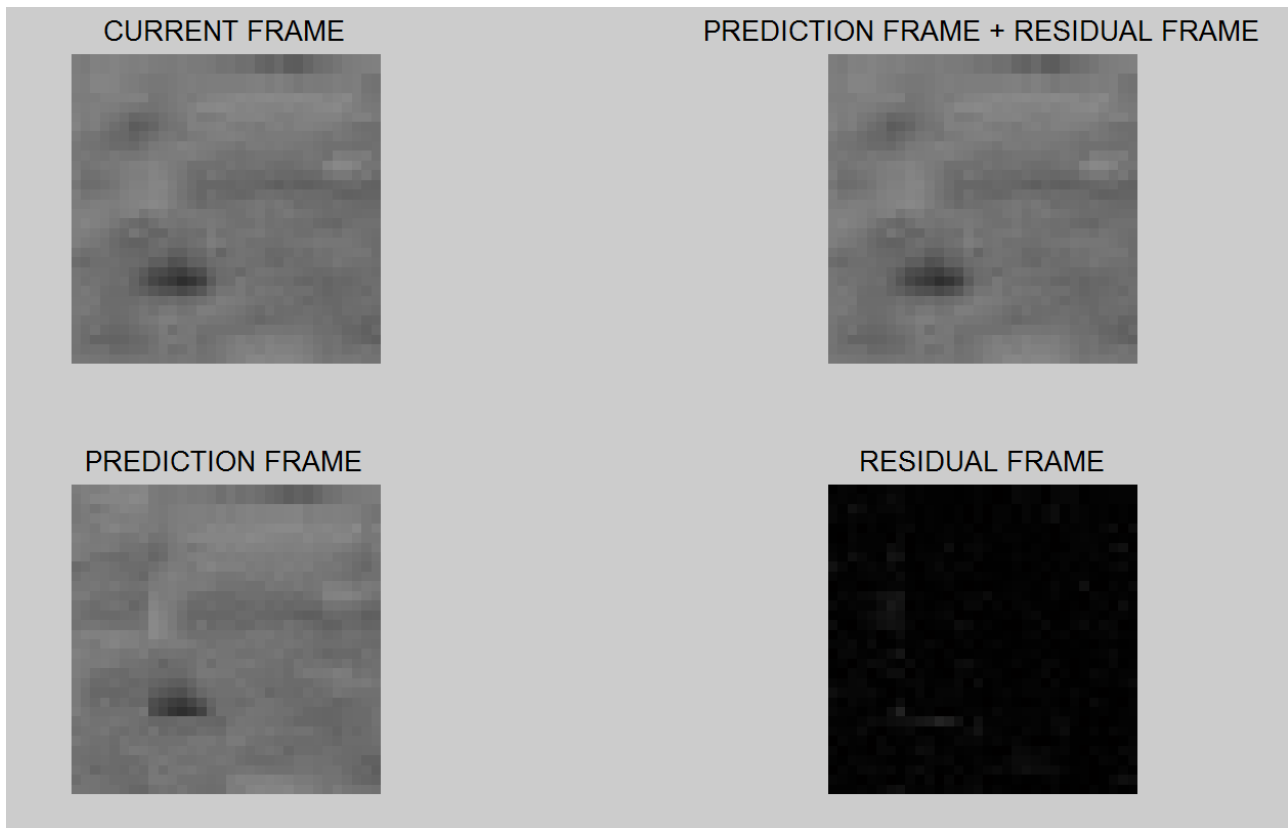
όπου E_{in} η ενέργεια του τωρινού frame $X(i, j)$ και E_{out} η ενέργεια του frame σφάλματος $X(i, j) - X_{pred}(i, j)$. Όσο μεγαλύτερος είναι ο λόγος αυτός τόσο μεγαλύτερη και η συμπίεση των δεδομένων.



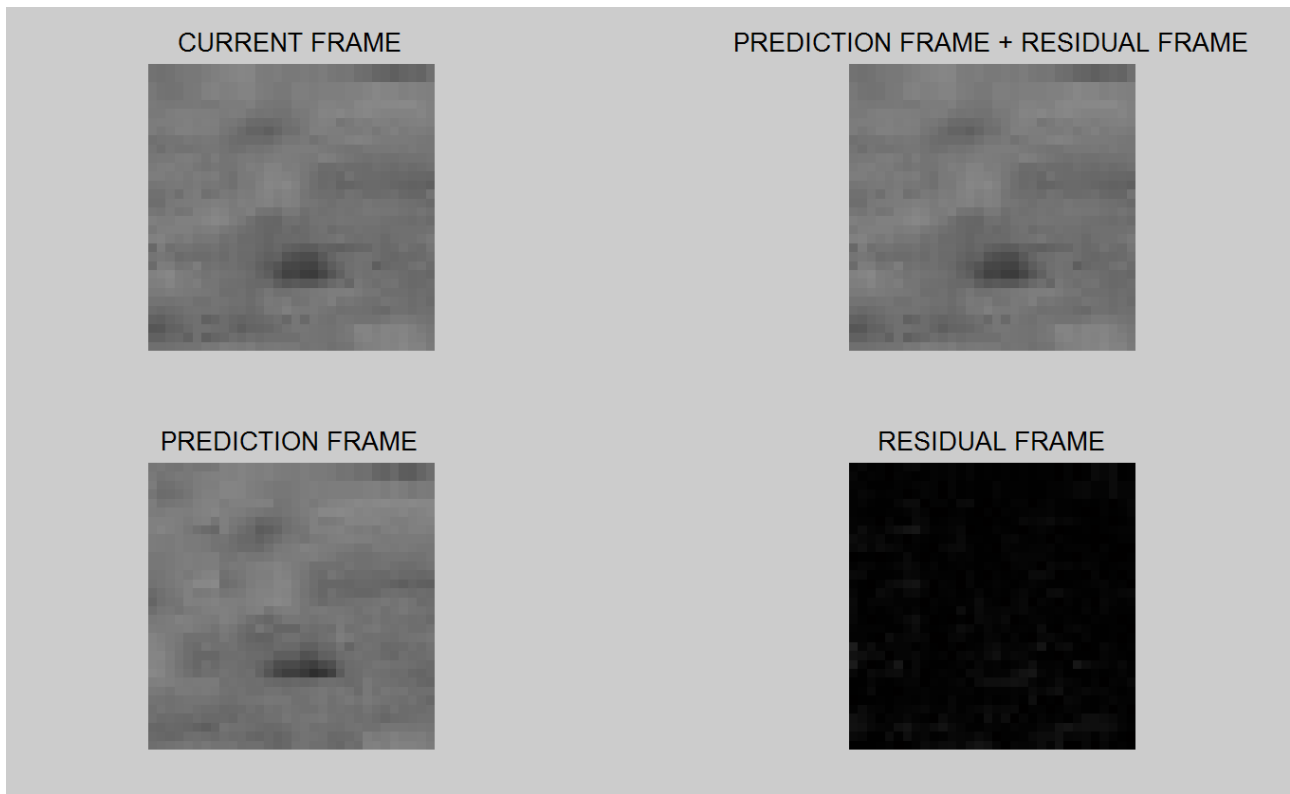
Γραφική Παράσταση 3.2 : Λόγος συμπίεσης για τρία συνεχόμενα πλαίσια

Παρατηρούμε από τις Γραφικές Παραστάσεις 3.1, 3.2 την πολύ σημαντική συνεισφορά του συστήματος MCP στην αφαίρεση του χρονικού πλεονασμού σε μία ακολουθία video, αφού επιτυγχάνει λόγο συμπίεσης περίπου 20 φορές και PSNR πρόβλεψης κοντά στα 35 db.

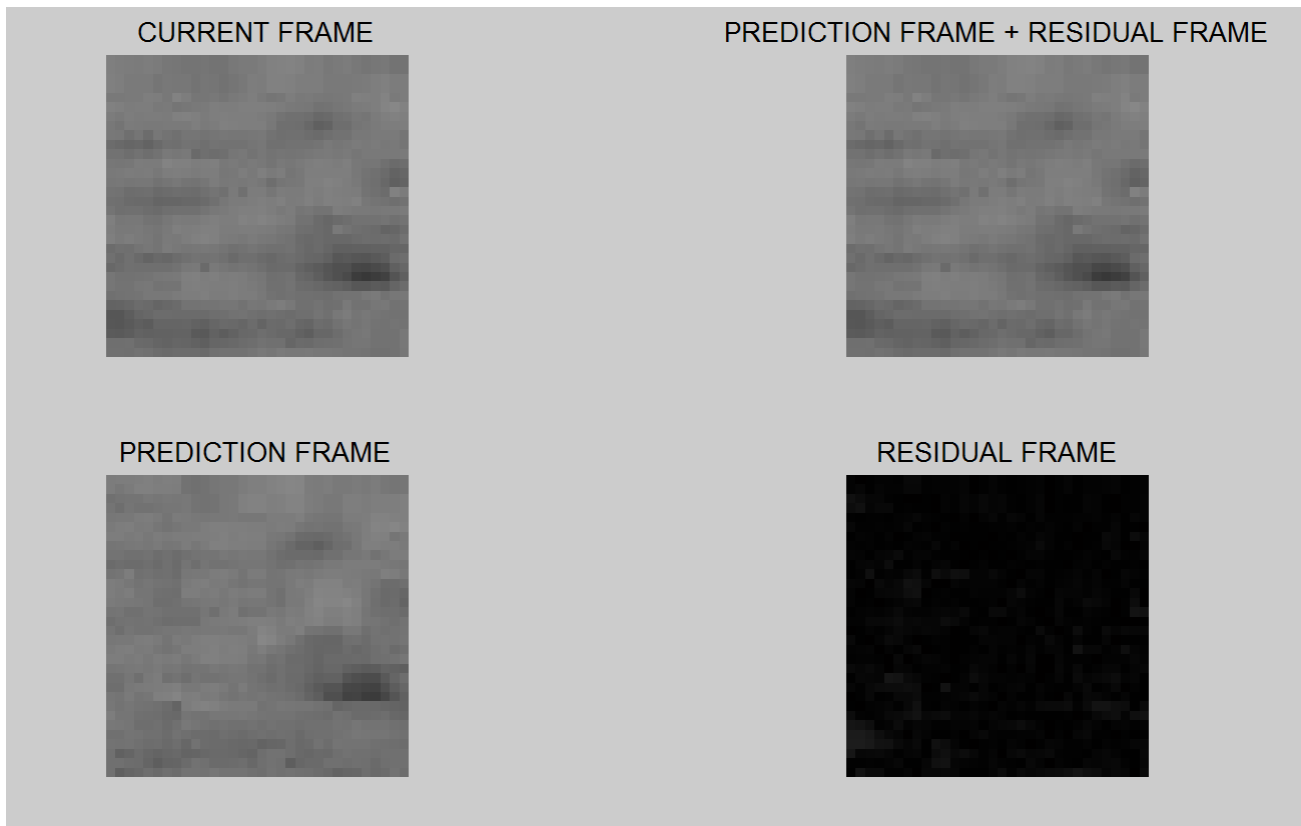
Στις παρακάτω εικόνες παρατηρούμε οπτικά τα αποτελέσματα της διαδικασίας Αντιστάθμισης Κίνησης.



Εικόνα 3. 1: FRAME1



Εικόνα 3. 2: FRAME2



Εικόνα 3. 3: FRAME3

Καταρχάς παρατηρούμε ότι το τωρινό frame και το ανακατασκευασμένο είναι ακριβώς ίδια και στις τρεις παραπάνω περιπτώσεις, κάτι που είναι λογικό, καθώς η διαδικασία MCP δεν εισάγει παραμόρφωση στην εικόνα. Ακόμα, παρατηρούμε ότι το frame πρόβλεψης είναι πολύ κοντινό στο τωρινό frame, και το frame σφάλματος είναι σχεδόν τελείως μαύρο, κάτι που σημαίνει ότι περιέχει πολύ λιγότερη ενέργεια από το τωρινό πλαίσιο. Η οπτική αυτή παρατήρηση επιβεβαιώνει τα παραπάνω μετρητικά αποτελέσματα σχετικά με την καθοριστική συμβολή του MCP στην αφαίρεση του χρονικού πλεονασμού σε μια ακολουθία video.

ΣΥΜΠΕΡΑΣΜΑΤΑ

Στα προηγούμενα κεφάλαια, περιγράφηκαν οι βασικότερες τεχνικές συμπίεσης εικόνων, καθώς και βασικές τεχνικές μετασχηματισμού αρχιτεκτονικής, όπως οι μέθοδοι *pipelining* και *parallel processing*, για την βελτιστοποίηση της απόδοσης ενός ψηφιακού συστήματος, όσον αφορά τον χρόνο εκτέλεσης, τον ρυθμό επεξεργασίας δεδομένων (*throughput*) και την κατανάλωση ενέργειας.

Αναλύθηκε η συνεισφορά του δισδιάστατου Μετασχηματισμού Συνημίτονου (2D Discrete Cosine Transform, 2D-DCT) στην κωδικοποίηση video, μέσω της αφαίρεσης της χωρικής συσχέτισης (*spatial correlation*) μεταξύ των δειγμάτων ενός block, καθώς και της συμπύκνωσης της ενέργειας του σήματος σε λίγους συντελεστές χαμηλών συχνοτήτων, καθώς και τα πρακτικά πλεονεκτήματα που έχει η χρήση του, έναντι του μαθηματικά βέλτιστου αλγορίθμου KLT. Ακόμα, έγινε σύντομη περιγραφή των υπαρχόντων αλγορίθμων υπολογισμού του DCT, καθώς και υλοποιήσεων αυτών και προτάθηκε μια νέα υλοποίηση. Η υλοποίηση αυτή βασίζεται στον αλγόριθμο αποδεκατισμού γραμμών-στηλών για την αναγωγή του υπολογισμού του δισδιάστατου μετασχηματισμού σε $2n$ υπολογισμούς μονοδιάστατων, όπου n το μέγεθος του μετασχηματισμού, και στον αλγόριθμο αποδεκατισμού στη συχνότητα (*decimation in frequency*) για την μείωση της πολυπλοκότητας του μονοδιάστατου μετασχηματισμού από $O(n^2)$ σε $O(n \log n)$. Η υλοποίηση αυτή πετυχαίνει να μειώσει την ελάχιστη χρονική διάρκεια κύκλου ρολογιού σε $T = \max(T_{\text{mem}}, T_{\text{add}}/32)$, όπου T_{mem} ο χρόνος προσπέλασης μνήμης και T_{add} ο χρόνος ολοκλήρωσης μιας πρόσθεσης, ή εναλλακτικά, να μειώσει την κατανάλωση ισχύος κατά περίπου 200 φορές, με τίμημα ωστόσο την επιπλέον επιφάνεια των απαραίτητων pipeline καταχωρητών, $2N(\log N - 2)$ πολυπλεκτών (συγκεκριμένα 192 πολυπλέκτες για $N=32$) ώστε να επιτυγχάνεται παράλληλος υπολογισμός μετασχηματισμών για $n < 32$, καθώς και διπλασίου μεγέθους μνήμης (4KB έναντι 2KB) και 32 φορές μεγαλύτερου μεγέθους κυκλώματος διευθυνσιοδότησης της (64 αθροιστές έναντι 2). Τέλος, μετρήθηκε πειραματικά ο

λόγος Μέγιστου Σήματος προς Θόρυβο (Peak Signal to Noise Ratio, PSNR), ο Λόγος συμπίεσης και το Ιστόγραμμα των συντελεστών, για τρία συνεχόμενα πλαίσια μιας ακολουθίας video, όταν χρησιμοποιούνται $n_0^2 \leq n^2$ συντελεστές μετασχηματισμού για την ανακατασκευή του αρχικού block, και επιβεβαιώθηκε έτσι η ιδιότητα συμπίκνωσης της ενέργειας του DCT, καθώς και η μικρή εντροπία των συντελεστών του, κάτι που επιτρέπει την περαιτέρω συμπίεση της ακολουθίας από έναν κωδικοποιητή εντροπίας.

Αναλύθηκε η συνεισφορά της διαδικασίας Κινησιακά Αντισταθμισμένης Πρόβλεψης (Motion Compensated Prediction, MCP) στην κωδικοποίηση video, μέσω της αφαίρεσης της χρονικής συσχέτισης (temporal correlation) μεταξύ των διαδοχικών εικόνων του video, που επιτυγχάνεται με την αναζήτηση και εύρεση της καλύτερης πρόβλεψης για κάθε block της τωρινής εικόνας σε μια περιοχή των χρονικά γειτονικών σε αυτή εικόνων. Περιγράφηκε η εξαντλητική αναζήτηση ως βασική προσέγγιση αντιμετώπισης, καθώς και εναλλακτικές προσεγγίσεις, οι οποίες μειώνουν τον απαιτούμενο χρόνο εκτέλεσης, μειώνουν ωστόσο και την ποιότητα της παραγόμενης πρόβλεψης, άρα και τον βαθμό συμπίεσης. Επιπλέον, αναλύθηκαν διαφορετικές υλοποιήσεις των παραπάνω προσεγγίσεων και προτάθηκε μια νέα υλοποίηση. Η υλοποίηση αυτή βασίζεται στον αλγόριθμο πλήρους αναζήτησης (Full Search) και αξιοποιώντας τις τεχνικές μετασχηματισμού αρχιτεκτονικών pipelining και parallel processing επιτυγχάνει μείωση του απαιτούμενου χρόνου εκτέλεσης κατά 96 φορές, με τίμημα ωστόσο την αύξηση τόσο της επιφάνειας ολοκλήρωσης, όσο και της κατανάλωσης ενέργειας κατά 19 φορές σε σχέση με την κλασική υλοποίηση του αλγορίθμου πλήρους αναζήτησης. Τέλος, μετρήθηκε πειραματικά ο λόγος Μέγιστου Σήματος προς Θόρυβο (Peak Signal to Noise Ratio, PSNR) και ο Λόγος συμπίεσης για τρία συνεχόμενα πλαίσια μιας ακολουθίας video και επιβεβαιώθηκε η καθοριστική συμβολή του MCP στην αφαίρεση του χρονικού πλεονασμού σε μια ακολουθία video.

BIBΛΙΟΓΡΑΦΙΑ

- [1]** A. K. Jain, "Fundamentals of Digital Image Processing", Prentice-Hall Inc., 1989.
- [2]** K. K. Parhi, "VLSI Signal Processing Systems, Design and Implementation", New York: John Wiley & Sons, Inc., 1999.
- [3]** P. Sreedevi, P. Rajanikumari, "Design and Implementation of Integer DCT Architectures for HEVC in VLSI Technology", International Journal of Eminent Engineering Technologies, vol. 2, Issue 3, April 2015.
- [4]** U. S. Potluri, A. Madanayake, R. J. Cintra, F. M. Bayer, S. Kulasekera, A. Edirisuriya, "Improved 8-Point Approximate DCT for Image and Video Compression Requiring Only 14 Additions", IEEE Transactions on Circuits and Systems, regular papers, vol. 61, no. 6, p. 1727-1740, June 2014.
- [5]** R. Conceicao, J. C. SOUZA JR., R. G. JESKE, B. ZATT, M. S. PORTO, L. V. AGOSTINI, "Low-Cost and High-Throughput Hardware Design for the HEVC 16x16 2-D DCT Transform", Journal of Integrated Circuits and Systems (Ed. Português), vol. 9, no.1, p. 25-35, 2014.
- [6]** J. D. Bolaños-Jojoa, J. Velasco-Medina, "Efficient Hardware Design of N-point 1D-DCT for HEVC", 20th Symposium on Signal Processing, Images and Computer Vision (STSIVA), p. 1-6, September 2015.
- [7]** Y. Y. Chung, N. W. Bergmann , "Implementation of DCT for Video Compression with Reconfigurable Technology", Hindawi Publishing Corporation EURASIP Journal on Advances in Signal Processing Volume 2011, Article ID 639043, 14 pages.
- [8]** A. Ahmed, M. U. Shahid, A. Rehman, "N Point DCT VLSI Architecture for Emerging HEVC Standard", Hindawi Publishing Corporation VLSI Design Volume 2012, Article ID 752024, 13 pages.

- [9]** Y. Chen, X. Cao, Q. Xie, C. Peng, "An Area Efficient High Performance DCT Distributed Architecture for Video Compression", The 9th International Conference on Advanced Communication Technology, vol.1, p. 238-241, February 2007.
- [10]** A. Ahmed, M. Awais, M. Maurizio, G. Masera, "VLSI Implementation of 16-point DCT for H.265/HEVC using Walsh Hadamard Transform and Lifting Scheme", IEEE 14th International Multitopic Conference, p. 144-148, December 2011.
- [11]** R. Jeske, J. C. de Souza Jr., G. Wrege, R. Conceição, M. Grellert, J. Mattos, L. Agostini, "Low Cost and High Throughput Multiplierless Design of a 16 Point 1-D DCT of the New HEVC Video Coding Standard", VIII Southern Programmable Logic Conference (SPL), p. 1-6, March 2012.
- [12]** A. Edirisuriya, A. Madanayake, R. J. Cintra, and F. M. Bayer, "A multiplication-free digital architecture for 16x16 2-D DCT/DST transform for HEVC", IEEE 27th Convention of Electrical Electronics Engineers in Israel, p. 1–5, November 2012.
- [13]** J. Zhu, Z. Liu, D. Wang, "Fully Pipelined DCT/IDCT/Hadamard Unified Transform Architecture for HEVC Codec", IEEE International Symposium on Circuits and Systems (ISCAS), p. 677-680, May 2013.
- [14]** P. K. Meher, S. Y. Park, B. K. Mohanty, K. S. Lim, C. Yeo, , "Efficient Integer DCT Architectures for HEVC", IEEE Transaction on Circuits and Systems for Video Technology, vol. 24, Issue 1, p. 168-178, January 2014.
- [15]** L. Hong, W. He, H. Zhu, Z. Mao, "A Cost Effective 2-D Adaptive Block Size IDCT Architecture for HEVC Standard", IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS), p. 1290-1293, August 2013.
- [16]** E. Kalali, E. Ozcan, O. M. Yalcinkaya, I. Hamzaoglu, "A Low Energy HEVC Inverse DCT Hardware", IEEE Third International Conference on Consumer Electronics - Berlin (ICCE-Berlin), vol. 60, Issue 4, p. 754-761, November 2014.
- [17]** Q. Shang, Y. Fan, W. Shen, S. Shen, X. Zeng, "Single-Port SRAM-Based Transpose Memory With Diagonal Data Mapping for Large Size 2-D DCT/IDCT", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.22, no.11, p. 2422-2426, November 2014.

- [18] H. Sun, D. Zhou, S. Kimura, S. Goto, "An area-efficient 4/8/16/32-point inverse DCT architecture for UHD TV HEVC decoder", IEEE Visual Communications and Image Processing (VCIP), p. 197-200, December 2014.
- [19] W. Yuan, P. Hao, C. Xu, "Matrix Factorization for Fast DCT Algorithms", IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 3, p. 3, May 2006.
- [20] P. Davis, M. Sangeetha, "Implementation of Motion Estimation Algorithm for H.265/HEVC", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, vol. 3, Special Issue 3, April 2014.
- [21] J. Lee, K. W. Lim, B. C. Song, J. B. Ra, "A fast multi-resolution block matching algorithm and its LSI architecture for low bit-rate video coding", IEEE Transaction on Circuits and Systems for Video Technology, vol. 11, no. 12, p. 1289-1301, December 2001.
- [22] M. E. Sinangil, A. P. Chandrakasan, V. Sze, M. Zhou, M. Sinangil, "Hardware-Aware Motion Estimation Search Algorithm Development for High-Efficiency Video Coding (HEVC) Standard ", IEEE International Conference on Image Processing (ICIP), p. 1529-1532, October 2012.
- [23] P. Nalluri, L. N. Alves , A. Navarro, "A Novel SAD Architecture for Variable Block Size Motion Estimation in HEVC Video Coding", IEEE International Symposium on System on Chip (SoC), p. 1-4, October 2013.
- [24] S. H. Yang, J. Z. Jiang, H. J. Yang, "Fast motion estimation for HEVC with directional search", Electronics Letters, vol. 50, Issue 9, p. 673-675, April 2014.
- [25] X. Yuan, L. Jinsong, G. Liwei, Z. Zhi, R. K. F. Teng, "A high performance VLSI architecture for integer motion estimation in HEVC", IEEE 10th International Conference on ASIC (ASICON), p. 1-4, October 2013.
- [26] H. Kibeya, F. Belghith, H. Loukil, M. A. B. Ayed, N. Masmoudi , "TZSEARCH pattern search improvement for HEVC motion estimation modules", 1st International Conference on Advanced Technologies for Signal and Image Processing, p. 95-99, March 2014.

- [27]** P. Nalluri, L. N. Alves, A. Navarro, "High speed SAD architectures for variable block size motion estimation in HEVC video coding", IEEE International Conference on Image Processing (ICIP), p. 1233-1237, October 2014.
- [28]** A. M. Tourapis, "Enhanced predictive zonal search for single and multiple frame motion estimation", SPIE 4671, Visual Communications and Image Processing, vol. 4671, pages 11, January 2002.
- [29]** X. Li, R. Wang, W. Wang, Z. Wang, S. Dong "Fast motion estimation methods for HEVC", IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), p. 1-4, June 2014.
- [30]** S. Agha, V. M. Dwyer, "Algorithms and VLSI Architectures for MPEG-4 Motion Estimation", Electronic Systems and Control Division Research, 2003.
- [31]** S. Kim, Suwon-Si, C. Park, H. Chun, J. Kim, "A Novel Fast and Low-complexity Motion Estimation for UHD HEVC" , Picture Coding Symposium (PCS), p. 105-108, December 2013.
- [32]** G. Sanchez, M. S. Porto, L. V. Agostini, "A hardware friendly motion estimation algorithm for the emergent HEVC standard and its low power hardware design", IEEE International Conference on Image Processing (ICIP), p. 1991-1994, September 2013.
- [33]** Bing-Fei Wu; Hsin-Yuan Peng; Tung-Lung Yu, "Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.16, no.10, p.1385-1398, October 2008.
-