وزارة التعليم العالي والبحث العلمي

BADJI MOKHTAR- ANNABA UNIVERSITY
UNIVERSITE BADJI MOKHTAR ANNABA

جامعة باجي مختار- عنابة

Faculté: Sciences de l'Ingéniorat.

Département: Electronique.

Domaine : Sciences et Technologies

Filière : Télécommunication

Spécialité : Systèmes des Télécommunications

Année : 2019/2020

MEMOIRE

Présenté en vue de l'obtention du diplôme de : MASTER

Implémentation de la transformation de Tchebichef discrète (DTT) sur le simulateur Sensevid pour l'IoT

Présenté par :

RAZKALLAH Hadi
ABEDGHERS Adel

Directeur de mémoire: M. KOUADRIA Nasreddine MCA , UBM Annaba

DEVANT Le JURY

Président : M. AMARA Fethi MCB , UBM Annaba

Examinateur :M. LAFIFI Saddek PROF, UBM Annaba

Année Universitaire **:** 2019/2020

Dédicace

Nous dédions ce travail aux :

Jeunes Algériens ambitieux qui aiment sa patrie.

Remerciements

*Nous tenons à remercier en premier lieu "**ALLAH**" qui m'a donné la patience pour finir ce travail, et que j'espère qu'il me guidera pour réussir dans le reste du chemin.*

Nous adressons les plus sincères remerciements à nos familles qui ont toujours encouragé.

*Nous tenons à remercier notre directeur de thèse, **Dr.KOUADRIA Nasreddine**, d'avoir nous dirigé dans ce travail avec compétence et professionnalisme, pour son encadrement de qualité et pour son coaching efficace.*

*Nous adressons à remercier le président de jury, **M. AMARA Fethi** , qui me fait l'honneur de présider ce jury , nous tenons également à remercier le membres du jury, **M. LAFIFI Saddek** d'avoir bien voulu examiner et juger ce travail.*

Nous tenons à remercier également à tous ceux qui nous a aidé de prêt ou de loin à finaliser cette thèse et qui ont contribué dans cette formation.

الملخص

تعتمد تقنيات أنترنت الأشياء على الأجهزة المنخفضة الاستهلاك في الطاقة كشبكات الإستشعار اللاسلكية للفيديو والذي يشكل تحدي كبير يعتمد أساسا على تقنيات الضغط بخوارزميات أقل تعقيد.غالبا ما يرافق هذا الضغط على بعض التدهور

الهدف من هذا العمل هو دمج خوارزمية تحويل تشيبيشاف المنفصل (DTT) في عملية ضغط الفيديوهات و مقارنتها مع خوارزمية الجيب التام المنفصل (DCT) بحساب معدلات إستهلاك الطاقة و قياس معايير النوعية المتعلقة بالفيديوهات المضغوطة (PSNR.SSIM).

في هذا العمل سيتم دمج الخوارزمية المذكورة ببرنامج المحاكات السانسفيد (SENSVID) و تحليل النتائج المتحصل عليها .

الكلمات المفتاحية : شبكات الاستشعار اللاسلكية البصرية, ضغط الصور , حساب الطاقة ، التعقيد الحسابي, التحويل الجيبي المنفصل, تحويل تشيبيشاف المنفصل , تقنيات التقليم .

Abstract

Internet of Things technology are based with low-power terminals such as wireless video sensors (WVS), which poses a great challenge mainly depends on compression techniques with low complex algorithms. These compressions are often accompanied by some degradations of the video.

The goal of this work is to integrate the discrete Tchebichef algorithm (DTT) in a WVS simulator which known as "SenseVid". The implementation is done on the video compression process and compared with the one using discrete cosine algorithm (DCT). This work also computes power consumption and measures parameters related to video quality (PSNR, SSIM).

By integrating the discrete Tchebichef algorithm (DTT) in the SenseVid simulator, we analyze the results obtained.

Keywords: wireless video sensor (WVS); SenseVid; image compression; energy consumption; computational complexity; discrete cosine transforms (DCT); discrete Tchebichef transform (DTT).

Résumé

La technologie de l'Internet des objets rassemble des terminaux à faible consommation d'énergie tels que les réseaux de capture vidéo sans fil. cela va poser un grand défi qui dépend principalement sur les techniques de compression avec des algorithmes moins complexes. Cette compression s'accompagne souvent avec quelque dégradation de la vidéo.

Le but de ce travail est d'intégrer l'algorithme discret de Tchebichef (DTT) dans le processus de compression vidéo et le comparer avec l'algorithme de cosinus discrète (DCT) et calculer la consommation d'énergie, mesurer les paramètres liés à la qualité du vidéo (PSNR). SSIM.

En intégrant l'algorithme discret de Tchebichef (DTT) dans le simulateur SenseVid, nous analysons les résultats obtenus.

Mots clés: réseaux de capteur vidéo sans fil, compression d'image, calcul d'énergie, complexité de calcul, transformée en cosinus discrète (DCT), transformation discrète de Tchebichef (DTT).

Liste des abréviations

BDCT	Binary Discrete Cosine Transform
bpp	bit per pixels
DCT	Discrete cosine Transform
DCT 1D	Discrete cosine Transform one dimension
DCT 2D	Discrete cosine Transform tow dimensions
DFT	Discrete Fourier Transform
DST	Discrete Sine Transform
DWT	Discrete Walvelet Transform
FFT	Fast Fourier Transform
GOP	Groupe of pictures
intDCT	Integer Discrete Cosine Transform
JPEG	Joint Photographic Experts Group
JPEG2000	Joint Photographic Experts Group 2000
KLT	Karhunen-Loeve Transform
MS	Mean square
MOS	Mean Opinion Score
MPEG	Moving Picture Experts Group
MTE	Minimize Total Energy
PSNR	Peak Signal to Noise Ratio
QoE	Quality of experience
QoS	Quality of service
RISC	Reduced instruction set computer
ROI	Region Of Interest
SSIM	Structural SIMilarity
SPIHT	Set Partitioning In Hierarchical Trees
TC	Taux de Compression
UIT-T	Union internationale des télécommunications
WISN	Wireless Image Sensor Network
WSN	Wireless Sensor Network
WVSN	Wireless Video Sensor Network

Liste des tableaux

Liste des figures

Table des matières

Introduction générale

L'être humain a tendance de rattacher sa vie de plus en plus avec les objets connecté à internet. Cette technologie cache derrière elle des diverses actionnaires (constructeurs informatique, mobiles, développeur, exploitant de réseaux de téléphonie mobile et fixe et Internet, les moteurs de recherche, les sociétés informatiques) ce qui a créé l'environnement nommé aujourd'hui par l'IoT (Internet of thing) .

Les avancées technologiques ont donné naissance à une nouvelle génération de réseaux appelés réseaux de capteurs sans fils (WSN). L'émergence de ces réseaux de capteurs sans fils ouvre la voie au déploiement de nouvelles applications de surveillance. Les WSNs forment un ensemble d'appareils électroniques miniaturisés autonomes, alimentés par batterie (pile), équipés de capteurs (camera) et peuvent communiquer entre eux sans fils.

Les réseaux de capteurs vidéo sans fils (WSVN) promet de révolutionner notre vie, provoque une dépendance avec l'environnement qui nous entoure. Grace à leur flexibilité, leur faible coût et la facilité de déploiement. Plusieurs domaines d'applications sont alors envisagés.
Nous citons par exemple, étudier la faune et la flore, la surveillance domaine de sécurité, la médecine et la santé, la logistique et les transports intelligents etc..

Problématique : pour avoir un réseau fiable et pratique il est nécessaire de préserver deux critères en même temps, consommer moins d'énergie et garder une bonne qualité de capture vidéo.

La consommation d'énergie des capteurs influence directement à la durée de vie du réseau qui est devenue le critère de performance fondamental. Plusieurs travaux de recherche [1] identifient que les activités de compression de vidéos avant la transmission est la phase la plus consommatrices en énergie.

SenseVid est un outil de simulation de WSN capable d'étudier, compresser et transmettre des vidéos en utilisant des encodages basés sur la DCT. Cependant, la transformée de Tchebichef discrète est une méthode qui est réputé être compétitive à la DCT et moins compliqué à implémenter. En effet, beaucoup de travaux récents publiés dans des revues de renommées

recommandent le remplacement de la DCT par la DTT dans les systèmes embarqués.

Afin d'ajouter des encodages utilisant la DTT, il a fallu ajouter cette transformation au programme SenseVid. Ce travail consiste alors à intégrer une implémentation de l'algorithme DTT à SenseVid et de faire différents tests et simulations pour comparer avec les DCTs déjà disponibles.

Organisation du manuscrit : Ce manuscrit est organisé en trois chapitres. Dans le chapitre 1, nous présentons le simulateur **SenseVid** utilisé dans la capture et l'évaluation des vidéos. Nous l'utilisons après pour intégrer un nouveaux algorithme de calcule. Nous commençons par présenter le fonctionnement général du simulateur, ces différents composants et ensuite nous faisons différents simulations pour le tester et comprendre son fonctionnement.

Dans le chapitre 2, nous présentons un état de l'art sur les transformations orthogonales utilisées dans la compression d'images d'une façon générale.

Le chapitre 3 constitue le cœur de notre travail de thèse, Nous présentons l'analyse et l'étude de la transformation discrète de **Tchebichef** (DTT), son intégration sur le simulateur **SenseVid** et les différents tests pour l'évaluer.

Chapitre 1

SenseVid: un outil de simulation et d'évaluation dédié aux réseaux de capteurs vidéo sans fil WVSN (*Wireless Video Sensor Networks*).

1- Introduction :

Les applications vidéo sont des éléments clés pour améliorer les réseaux de capteurs sans fil traditionnels (WSN). Elles offrent une diversité de donnée par rapport aux capteurs des données simples qui se chargent de traiter seulement la température, la luminosité, la pression atmosphérique ou les vibrations **[1].** En conséquence, les chercheurs du réseau de capteurs ont besoin d'outils adéquats et faciles à utiliser pour évaluer les performances de leurs propositions.

Plusieurs simulateurs sont disponible sur la littérature pour faire ces teste, citons pour l'exemple: **Evalvid** , **Sim-LIT**, **WiSE-MNet++, M3WSN**, **Joined-JPEG**, **EvalVSN [2-3]** et l'outil que nous avons utilisé dans cette thèse est le **SenseVid**.

SenseVid est un outil de transmission et d'évaluation vidéo open source installe sous system d'exploitation **UNIX**, prend en compte les caractéristiques spécifiques de WSN. En outre la compression intra-frame à faible énergie basée sur la transformation en cosinus discrète rapide (*pruned DCT*), une compression inter-frame de faible complexité est adoptée aussi pour permettre une prise en charge efficace des flux vidéo, un modèle d'énergie configurable est utilisé dans lequel les coûts de capture vidéo et l'encodage sont comptabilisés sur une image de base. Une diversification du trafic vidéo en fonction des niveaux de priorité est également fournie.

SenseVid peut être utilisé dans n'importe quel environnement de simulation ou de scénarios réel comme plaque d'essai. D'autres modules peuvent facilement être développés pour d'autres environnements. Pour évaluer la qualité de la vidéo reconstruite le PSNR (*Peak Signal to Noise Ratio*) et SSIM (*Structural SIMilarity*) sont implémentés en tant que métriques QoE (*quality of experience*) et qualité de service QoS (*quality of service* **[4].**

2- Plateforme et fonctionnalité :

SenseVid, est un nouvel outil qui suit le même principe d'EvalVid**,** utilisé dans la communauté de recherche en réseau pour évaluer la transmission vidéo via des métriques QoE , tels que PSNR et le SSIM **[5]** . En plus d'autres métriques QoS classiques comme le délai, la gigue et le taux de perte tout en étant spécifique à WVSN.

$$- PSNR = 10\log\left(\frac{I_{\max}^2}{MSE}\right) \quad (1)$$

$I_{\max}$: désigne la luminance maximale

MSE : Erreur quadratique moyenne (voir **[5] p43**).

$$SSIM(x,y) = \frac{2\mu_x\mu_y + (K_1L)^2}{\mu_x^2 + \mu_y^2 + (K_1L)^2} * \frac{2\sigma_{xy} + (K_2L)^2}{\sigma_x^2 + \sigma_y^2 + (K_2L)^2} \quad (2)$$

SSIM : (*Structural SIMilarity*) est une mesure de similarité entre deux images numériques (voir **[5] p44**).

En outre, Il adopte l'approche de trace de trafic, sur la base d'une vidéo d'entrée et d'un codec donné, un fichier de trace vidéo est généré. Chaque ligne du fichier de trace donne principalement, pour chaque trame, son type et sa taille. Un fichier de trace de l'expéditeur est également généré avec principalement le temps de transmission et la taille des paquets à envoyer.

Exemple1 :

#Rank	Type	Size (Bytes)	refPSNR	refSSIM	bpp	layers Size (bits)	captureEnergy (mJ)	encodingEnergy (mJ)	bit rate (kbps)
1	M	1619	23.297	0.827	0.511	12950	1.049	43.088	323.75
176	M	1749	23.7	0.843	0.552	13992	1.049	43.088	349.8
250	S	1907	23.803	0.832	0.602	15255	1.049	43.088	381.375

Tableau 1.1 : trois lignes (frames) d'une capture vidéo avec **SenseVid.**

Type: type de trame (frame) M ou S (principale ou secondaire).

- **PSNR** : Peak Signal to Noise Ratio.
- **SSIM** : Structural SIMilarity.
- **bpp** : bite par pixel.

Le fichier trace peut être utilisé par n'importe quel simulateur ou application réelle pour émuler la transmission d'une vidéo donnée. Basé sur le fichier de trace du récepteur qui donne la liste des paquets reçus et calcule les métriques de QoS et QoE obtenues.

Exemple 2:

Frame 1	Frame 12	Frame 45
		
Frame 77	Frame 102	Frame 143
		
Frame 166	Frame 237	Frame300
		

Tableau 1.2 capture video : hallQcf , de trame capture de (frame 1 …frame 300) , capture de vidéo : hall

Suivant les études publiées en **[6]** , les encodeurs, à savoir MPEG-4, H.263 ou H.264 sont gourmands en ressources d'énergie et ne sont donc pas adaptés aux réseaux contraints tels que le WVSN (Fig 1.1) . Ces codecs concentrent la complexité de calcul la plus élevée sur l'encodeur tandis que les capteurs vidéo n'ont pas suffisamment la puissance de traitement ni l'énergie suffisante pour exécuter des algorithmes de compression complexes **[7].**

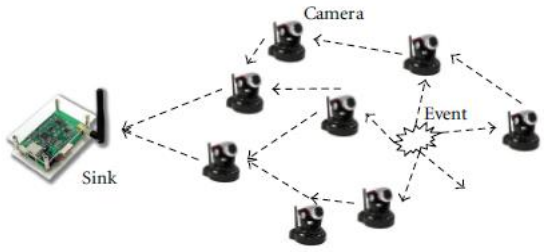
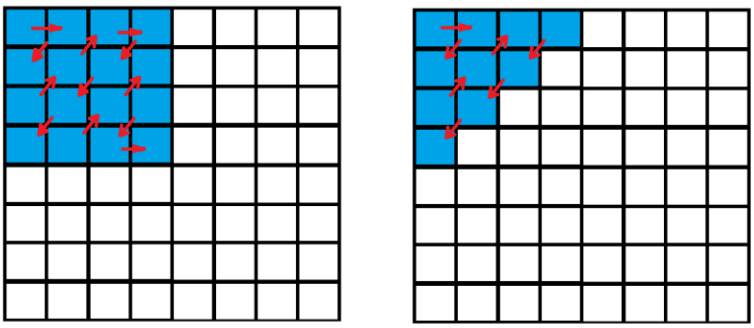


Figure 1.1 - réseaux de capteurs Vidéos sans fil

Le schéma de compression le plus populaire basé sur la DCT est le JPEG. L'image est divisée en petits blocs 8 × 8 sur lesquels le DCT est appliqué pour séparer les informations hautes et basses fréquences. Pour compresser l'image, une quantification est appliquée sur chaque bloc DCT éliminer les coefficients contenant le moins d'informations (composantes de fréquence les plus élevées). Un balayage en zigzag est ensuite appliqué pour réorganiser les coefficients en des basses et hautes fréquences. Enfin, un codage entropique est appliqué. Il consiste généralement en un encodage de longueur (**RLE**) suivi d'un codage **Huffman** ou arithmétique. La transformation DCT entraîne une faible utilisation de la mémoire car elle fonctionne sur de petits blocs et permet des taux de compression acceptables. Le calcul de la DCT prend au moins 60% de la puissance totale de l'encodeur.



1- Model Carré 2- Model Triangle

Figure 1.2 – model DCT rapide

2-1 DCT rapides implémentées en SenseVid:

1. LLM: Elle ne nécessite que 11 multiplications et 29 additions. Elle atteint la complexité multiplicative minimale réalisable. Ce algorithme peut etre approximé en appliquant la DCT zonale. On déduit ainsi le **SLLM** (square LLM) et le **TLLM** (triangle LLM) **[8]** (Voir figure 1.2).

2. binDCT: est une version DCT sans multiplication basée uniquement sur des opérations de décalage et d'addition. La binDCT 2-D permet en 16 bits une implémentation entière qui réduit encore la complexité DCT. En appliquant la DCT on déduit le **Sbin** (square Bin) et le **Tbin** (triangle TBin) **[9].**

Il existe de nombreux simulateur qui offrant le calcul de la DCT, mais **SenseVid** permet de différencier les niveaux de priorité des données à transmettre afin de s'adapter aux contraintes et dynamiques du réseau [4,p123]. Le tableau suivant résume les principaux simulateurs WSN comme suite Tableau (1.3) :

outile	Evalvid	Sim-LIT	WVSN	WiSE-MNet++	M3WSN	Joined-JPEG	EvalVSN	SenseVid
Langage	C	C++	C++	C++	C++	C++	Matlab	C++
Open source	✓		✓	✓				✓
Video support	✓			✓	✓		✓	✓
Codec	MPEG-4 H.263/4	custom	custom	N/A	MPEG-4 H.263/4	JPEG JPEG2000	custom	custom
Simulator	ns2	Tossim	Castalia	Castalia	Castalia	Castalia	ns2	cooja
QoE metrics	PSNR SSIM	PSNR	PSNR	PSNR SSIM	PSNR SSIM	PSNR	PSNR SSIM	PSNR SSIM

Tableau 1.3 (comparaison des simulateurs)

3- Principe de fonctionnement et architecture :

SenseVid implémente un modèle énergétique qui permet d'estimer l'énergie requise pour capturer et coder chaque trame. Une procédure de priorisation est également mise en place afin de pouvoir s'adapter à la dynamique du réseau en permettant ou non la transmission de données moins importantes. Dans ce qui suit nous allons résumer les principaux traitements inclus dans **Sensevid.**

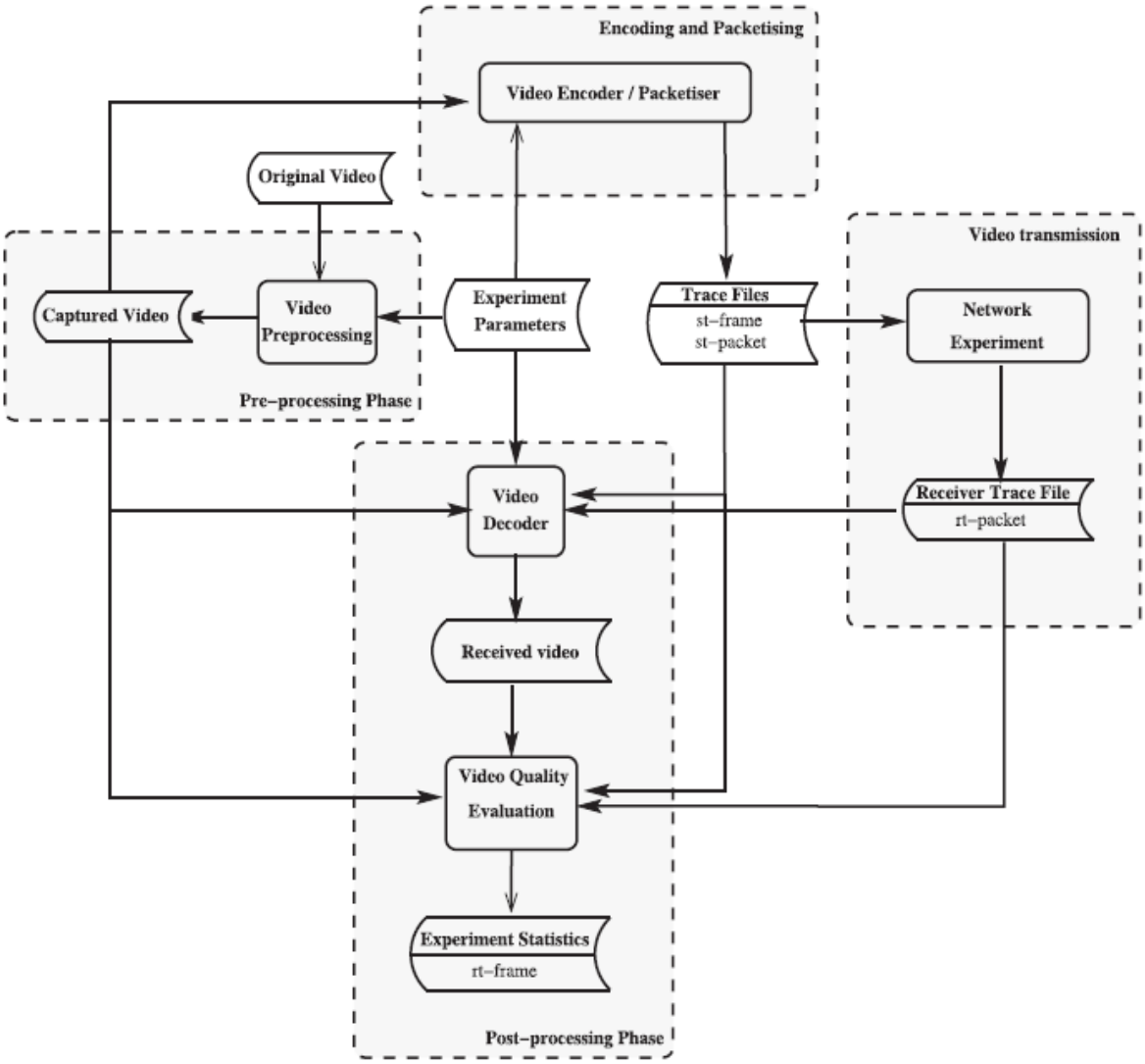


Figure 1.3 – Architecture de SenseVid

3-1 Prétraitement vidéo :

Le module de prétraitement vidéo permet de redimensionner les trames vidéo principalement à une résolution inférieure afin de réduire la vitesse de transmission requise. Dans le même contexte, la vidéo FFC (capture de fréquence d'image) d'origine peut être abaissée. L'envoi de séquences vidéo à 30 ou même 25 ips (images par seconde) est tout simplement irréalisable en raison des ressources limitées d'un WSN. Une vidéo QCIF (176 × 144) à échelle de gris compressée à 0,5 bpp nécessite un débit binaire utile de 316,8 kbps lorsqu'il est transmis à 25 ips.

3-2- Module encodeur vidéo / packetiser :

Le module encodeur vidéo / packetiser comprime le clip vidéo capturé et génère l'expéditeur fichiers de trace, à savoir st-frame et st-packet. Il convertit les images capturées en niveaux de gris où chaque pixel est codé en utilisant 8 bits avec plage [0,255]. Un algorithme de compression à faible énergie qui prend en compte à la fois la redondance spatiale et temporelle dans une séquence vidéo est mise en œuvre. Une trame est soit intra-codée et qualifiée de trame principale (trame M), soit inter-codée par rapport à la trame M précédente auquel il est appelé à une trame secondaire (trame S). La première trame est toujours codée en tant que trame M. Une suite la trame est codée M si elle est suffisamment différente de la trame M précédente; sinon, il est codé en S.

Le fait qu'une trame donnée soit codée en tant que trame principale ou secondaire est basé sur un paramètre utilisateur **γ (le coefficient GOP [10]** *groupe of pictures***)** qui permet pour contrôler le GOP de la vidéo compressée produite. **γ** prend des valeurs positives allant de 0 à l'erreur quadratique moyenne maximale (Vpeak = 255) deux images peuvent avoir. Pour chaque trame, l'erreur quadratique moyenne (MSE) (moyen square erreur **[5]**) par rapport à la trame M précédente est calculée.

$$MES = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} \left(I(m,n) - \hat{I}(m,n) \right)^2 \quad (3)$$

Où $M \times N$ est la taille de l'image , $\boldsymbol{I}$ et $\hat{\boldsymbol{I}}$ sont respectivement les amplitudes des pixels sur les images originale et dégradée **[5]**

Si MSE> γ2 alors la trame est considérée comme suffisamment différente de la trame M précédente et est donc codée M; sinon, c'est Codé en S. Notez que si γ = 0 alors toutes les trames sont codées en M. Pour augmenter le nombre de trames S, γ doit être augmenté. Réglage de γ à 255 entraîne le codage S de toutes les trames, à l'exception de la première trame.

3-2-1 Encodage de la trame principale (Main frame) :

Comme illustré sur la figure 4, chaque trame M est d'abord décomposée en blocs de 8 × 8 dont chacun est décalé de la plage [0,255] à signé les entiers vont de [−128, 127].

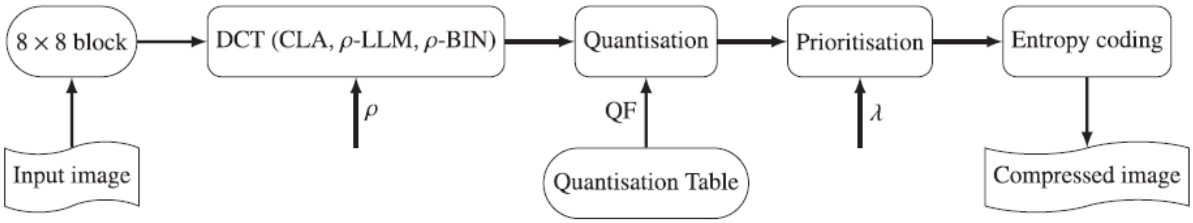


Figure 1. 4 – bloc de codage de la trame M

Ensuite, une DCT est appliqué sur chaque bloc. Afin de répondre aux exigences du WSN contraint, deux rapides DCT élagués sont mis en œuvre, le DCT ***Loeffler-Ligtenberg-Moschytz (LLM)*** [8] et ***le binDCT-C (BIN)*** [9]. Les deux motifs carré et triangulaires d'élagage DCT sont fournis. (SLLM,TLLM et SBIN,TBIN). La longueur du côté carré ou triangle **ρ (coefficient de la DCT)** est définie par l'utilisateur. par défaut La valeur est fixée à **8.** À des fins de comparaison, nous avons également implémenté la variante DCT traditionnelle, que nous appelons **CLA**.

La DCT produit des coefficients de bloc quantifiés à l'aide de la matrice de quantification standard JPEG. Compromis entre le niveau de qualité et le taux de compression peut être obtenu en sélectionnant un facteur de qualité approprié (QF) qui permet d'ajuster les valeurs de la matrice de quantification. L'utilisateur peut décider la qualité visuelle de la trame allant de la plus mauvaise (QF = 1) à la meilleure qualité (QF = 100).

Le bloc obtenu (B) est ensuite linéarisé en utilisant le balayage en zigzag traditionnel avec une procédure de priorisation facultative. Pour une M-frame, les priorités vont de 0 à 12 et suivent le balayage en zigzag du bloc. Pour le nombre maximum de priorités ($\lambda = 13$), le premier niveau de priorité ($l = 0$) contient le composant DC (b11) et les deux composants AC suivants (b12 et b21).

Le dernier niveau de priorité (l = 12) est composé des trois derniers coefficients du bloc linéarisé: b78, b87 et b88. Les coefficients de bloc restants sont assignés aux niveaux **l ∈ [1, 11]** où le coefficient **bij** obtient le niveau de priorité **l = i + j - 3**.
La figure suivante (1.5) , montre le balayage en zigzag et l'affectation de priorité.

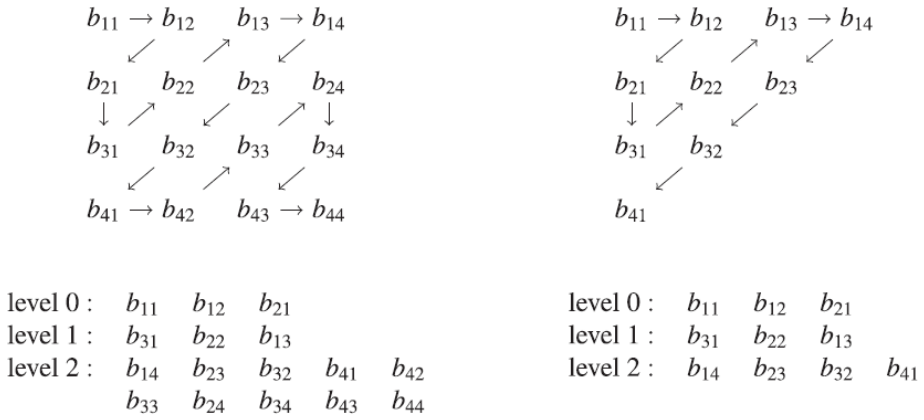


Figure 1.5 , montre le balayage en zigzag et l'affectation de priorité.

Lorsque l'utilisateur sélectionne trois niveaux de priorité (λ = 3) et un DCT élagué rapide avec une longueur de côté ρ = 4. Les formes carrées et triangulaires sont illustrés.

Enfin, un codage entropique sans perte est appliqué. L'utilisateur peut choisir parmi différents codeurs tels que **Huffmann [11]** et (**EG**) **exponentiels Golomb** un encodage de longueur de course (**RLE**) **[12].** Le codeur entropique par défaut consiste à coder tous les coefficients en utilisant le **code exponentiel-Golomb**.

3-2-2 Encodage de trame secondaire (secondary Frame) :

Une trame secondaire est inter-codée par rapport à la trame principale précédente. Autrement dit, ce qui est codé est la différence entre la trame actuelle et sa trame principale. Les blocs de différence résultants avec seulement des valeurs nulles qui sont ignorés car ils correspondent à des blocs qui sont exactement les mêmes que ceux de la trame M précédente. Afin de réduire la quantité de données à transmettre, seul un sous-ensemble des blocs restants sont encodés. Les blocs réputés similaires aux blocs de trame M précédents ont une priorité inférieure, sont compressés avec perte ou Simplement rejeté en fonction de deux paramètres fournis par l'utilisateur requis par la priorisation et les étapes de seuillage (Fig. 1.6).

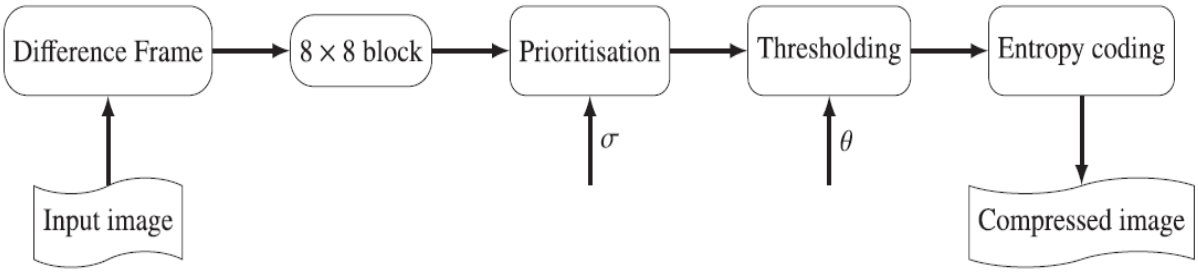


Figure 1. 6 – séquence de codage de la trame S

L'étape de priorisation consiste à attribuer un niveau de priorité à chaque bloc. Les niveaux de priorité vont de 0, le plus élevé, à 4, le niveau de priorité le plus bas. Les blocs de priorité la plus élevée sont ceux qui présentent le moins de similitude avec leurs blocs associés dans le cadre M précédent.

$$MS = \sum d_{ij}^2 / 64 \quad (4)$$

Les cinq niveaux de priorité sont liés aux notes MOS (***Mean Opinion Score*** **[13]**) et au carré moyen correspondant au MS (*mean square*) :

Pour chaque bloc de la trame de différence (**tableau 1.4**).

MS	PSNR	priorité
< 13	> 37	4
]13,51]	]31,37]	3
]51,205]	]25,31]	2
]205,650]	]20,25]	1
> 650	< 20	0

Tableau 1.4 : blocs de priorité pour la trame-S.

La priorité la plus élevée est attribuée aux blocs avec MS> 650 et la priorité la plus faible à ceux avec MS <13. La raison derrière cela est que si un bloc avec une priorité la plus élevée est manquant, alors en le reconstruit et le bloc basé sur le bloc codé M précédent aura une mauvaise qualité (PSNR <20 dB).

En revanche, la perte de blocs avec la priorité la plus basse n'affecte pas la qualité du bloc reconstruit qui obtient toujours une excellente qualité (PSNR> 37 dB).

Pour économiser les capteurs vidéo et les ressources réseau, l'utilisateur peut décider de coder uniquement un sous-ensemble des blocs d'une trame donnée en ajustent du paramètre σ qui définit le niveau de priorité le plus bas à considérer pour l'étape suivante (seuillage). Pour être plus adapté à WSN, la valeur par défaut σ est définie sur 0 où seuls les blocs avec le plus haut niveau de priorité sont conservés. En conséquence, l'énergie sera considérablement réduite. Cependant, si σ est réglé sur sa valeur maximale 4, alors tous les blocs de trame de différence sont codés. Ceci peut consomment une énergie excessive et peuvent entraîner un faible taux de compression qui ne justifie pas le choix d'inter-codage pour une trame donnée.

4- Le module d'énergie :

La négligence du coût de calcul (détection et traitement) par rapport au coût de communication peut convenir aux WSN. Ceci ne convient pas dans les WVSN où l'acquisition et la compression vidéo nécessitent des ressources de calcul importantes qui ne peuvent pas être négligées. En conséquence, un modèle de consommation d'énergie doit être fourni afin de pouvoir évaluer le coût global des applications vidéo, de la capture vidéo jusqu'à sa livraison finale. Tout modèle de consommation d'énergie dépend du nœud de capteur et ses capacités de calcule.

SenseVid donne séparément une estimation de l'énergie requise pour capturer et coder chaque trame. La capture d'énergie dépend principalement sur la taille du trame, la taille du trame détermine la taille du réseau de capteurs et suivant Le nombre d'opérations nécessaires pour convertir chaque pixel en une valeur numérique. Actuellement, nous avons adopté un modèle simple où l'utilisateur peut fournir l'énergie requise pour capturer un bloc 8×8, puis SenseVid calcule l'énergie de capture globale en fonction de la Taille du trame. L'énergie de compression est estimée en fonction de la façon dont une trame est codée.

Le coût de compression d'une trame codée-M (M-encoded frame) est estimé sur la base de la puissance requise pour M-encod en un bloc $P_{M\text{-}Bloc}$ donné par:

$$P_{M-block} = P_{DCT} + P_{Quantisation} + P_{entropy} \quad (5)$$

Où PDCT, PQuantisation et Pentropy sont des quantités d'énergie consommées par respectivement le calcule, la quantification et le codage entropique d'un bloc.

PDCT peut être calculé en utilisant:

$$P_{DCT} = \sum_{op} N_{op} * P_{op} \quad (6)$$

$\mathbf{P_{op}}$ est la consommation d'énergie d'opération **op** (addition, multiplication ou décalage) qui est principalement en fonction du nombre de cycles pour exécuter **l'op** et la puissance du processeur sous-jacent et la fréquence d'horloge.

$\mathbf{N_{OP}}$ est le nombre requis d'opérations de type **op** pour calculer le **DCT**, Ce nombre dépend du DCT sélectionné ainsi que de sa taille de zone (**ρ**) et elle est donné par :

$$N_{OP}^{\rho} = \begin{cases} (\rho+8) \times n_{op,\rho} & \text{if square DCT} \\ \rho \times n_{op,}\rho + \sum_{i=1}^{\rho} n_{op,i} & \text{if triangular DCT} \end{cases} \quad (7)$$

$\mathbf{n_{op,i}}$ est le nombre d'opérations de type op requises par la DCT 1-D de longueur latérale i.

Le tableau 6 résume le nombre d'opérations pour le LLM et le BIN DCT à élagage rapide mis en œuvre.

ρ	LLM						binDCT-C					
	1-D DCT		sq. 2-D DCT		tr. 2-D DCT		1-D DCT		sq. 2-D DCT		tr. 2-D DCT	
	Add	Mul	Add	Mul	Add	Mul	Add	Mul	Add	Mul	Add	Mul
8	29	11	464	176	413	152	30	13	480	208	420	172
7	28	11	420	165	348	130	28	12	420	180	346	139
6	26	10	364	140	280	102	28	12	392	168	290	115
5	25	9	325	117	223	77	28	12	364	156	234	91
4	24	9	288	108	169	59	27	11	324	132	174	63
3	23	8	253	88	118	38	19	6	209	66	96	26
2	20	6	200	60	66	18	13	2	130	20	46	6

Tableau 1.5 : nombre d'opérations suivant le coefficient du DCT

A ce stade, il convient de noter que le coût de fonctionnement du type opération est différenciée en fonction de l'arithmétique entière (BIN DCT) ou flottante (CLA et LLM). La puissance quantification par bloc $P_{quantification}$ est estimée comme suit:

$$P_{Quantisation} = \begin{cases} 8^2 \times \mathrm{P}_{div} & \text{classical DCT} \\ \rho^2 \times \mathrm{P}_{div} & \text{square pruned DCT of side length } \rho \\ \rho \times (\rho - 1) \times \mathrm{P}_{div} & \text{triangular pruned DCT of side length } \rho \end{cases} \quad (8)$$

Pdiv est la consommation d'énergie due au fonctionnement de la division. Enfin, la puissance d'entropie est estimée en fonction de coder entropique choisie. L'encodeur par défaut est (EG) l'exponentielle-Golomb d'où la puissance requise pour encoder une valeur non nulle peut être donnée par:

$$P_{EG} = \begin{cases} 9 & \text{if CLZ instruction is availabe} \\ 3 \times \lfloor \log_2(value + 1) \rfloor + 7 & \text{otherwise} \end{cases} \quad (9)$$

où CLZ est l'instruction *Count Leading Zeros*.

La puissance requise pour coder une trame S est calculée en utilisant:

$$P_{s-frame} = P_{diff} \times blocksNb + P_{priority} \times nonNullblocksNb + P_{thresh} \times finalBlocksNb + P_{entropy} \quad (10)$$

$blocksNb$, $nonNullblocksNb$ et $finalBlocksNb$ sont le nombre de blocs de trame, les blocs non nuls dans la trame de différence et le nombre de blocs retenus après seuillage respectivement.

$P_{diff} = 8^2 \times P_{sub}$	est la puissance consommée par le processus de soustraire le cadre principal.
$P_{priority} = 8^2 \times (P_{mul} + P_{add})$	est la puissance consommée par le calcul carré moyen dans la hiérarchisation étape.
$P_{thresh} = (P_{test} + P_{assignment}) \times 8^2$	est la puissance requise par l'étape de seuillage.
$P_{entropy} = numberOfNonNullValues \times P_{EG}$	est la puissance pour le codage entropique

Tableau 1.6 calcule de la puissance

5- Résultats numériques :

Dans cette section, nous souhaitons évaluer l'effet de certains paramètres sur la séquence vidéo compressée résultante sans compte tenu de sa transmission. Nous considérons deux clips vidéo QCIF à savoir le hall et les séquences vidéo de fleurs , composées respectivement de 300 et 250 images capturées à 25 ips. La vidéo de la salle

peut être qualifiée de séquence à faible mouvement avec un fond tandis que la vidéo de fleurs est une séquence de mouvement moyenne avec de nombreux petits détails.

Nous introduisons des valeurs de qualité (q) varie de 10 à 90 pendant la capture des deux vidéos (hall et flowers) avec toutes les DCT (CLA, S-T.LLM et S-T.Bin) et les trois encodeurs entropique (EG, RLE_EG, HUFFMAN) pour obtenir les différents résultats du :

1. **Valeur de l'encodeur énergie, Enc Energy (mj).**
2. **PSNR.**
3. **SSIM.**

Une fois la capture est terminé, le programme crée deux dossiers, **capturedFrames** (contient tous les trames capturé) et le **referenceFrames** (contient les trames de référence) et trois fichiers trace :

1. **Inputparameters** : les informations de la capture.

CODEC	SMPEG	Original resolution	176x144
GOP coef.	0	Target resolution	176x144
Quality coef.	50	Original FPS	25
M levels nb	1	Target FPS	25
DCT	CLA	Captured Frames	250
Zone size	4	Simulation Id	Cla50
Threshold	1	Packet payload size	96
Max. considered S level	0	Output directory	./
Entropy coding	HUFFMAN		

2. **st-packet :**

time++seqNb++pktSize++frameNbframeType++layerNb++blocksList

3. **st-frame** : Toutes les données ont été extraites de ce fichier trace.

5-1 facteur qualité / énergie :

Enc energy(mJ) capture avec RLE Enc entropique										
Vidéo	flaowr					Hall				
Q	10	30	50	70	90	10	30	50	70	90
CLA	43,087	43,088	43,088	43,088	43,088	43,087	43,087	43,087	43,088	43,088
tBIN	0,602	0,602	0,602	0,602	1	0,602	0,602	0,602	0,602	0,602
sBIN	0,315	0,315	0,315	0,315	0	0,315	0,315	0,315	0,315	0,315
tLLM	22,433	22,433	22,433	22,433	22	22,433	22,433	22,433	22,433	22,433
Sllm	12,697	12,697	12,697	12,697	13	12,697	12,697	12,697	12,697	12,697

Enc energy(mJ) capture avec EG Enc entropique										
CLA	43,24	43,323	43,377	43,44	43,605	43,223	43,287	43,329	43,383	43,551
tBIN	0,759	0,819	0,847	0,878	1	0,74	0,805	0,841	0,881	0,971
sBIN	0,441	0,482	0,5	0,521	1	0,43169	0,481	0,506	0,533	0,59
tLLM	22,625	22,688	22,718	22,749	23	22,607	22,689	22,73	22,773	22,863
Sllm	12,839	12,881	12,9	12,921	13	12,834	12,889	12,915	12,941	12,998
Enc energy(mJ) capture avec HUFFMAN Enc entropique										
CLA	43,088	43,088	43,088	43,088	43,088	43,088	43,088	43,088	43,088	43,088
tBIN	0,315	0,315	0,315	0,315	0	0,315	0,315	0,315	0,315	0
sBIN	0,602	0,602	0,602	0,602	0	0,602	0,602	0,602	0,602	0
tLLM	12,697	12,697	12,697	12,697	0	12,697	12,697	12,697	12,697	0
Sllm	22,433	22,433	22,433	22,433	0	22,433	22,433	22,433	22,433	0

Tableau 1.7 résultat du capturé facteur qualité / énergie

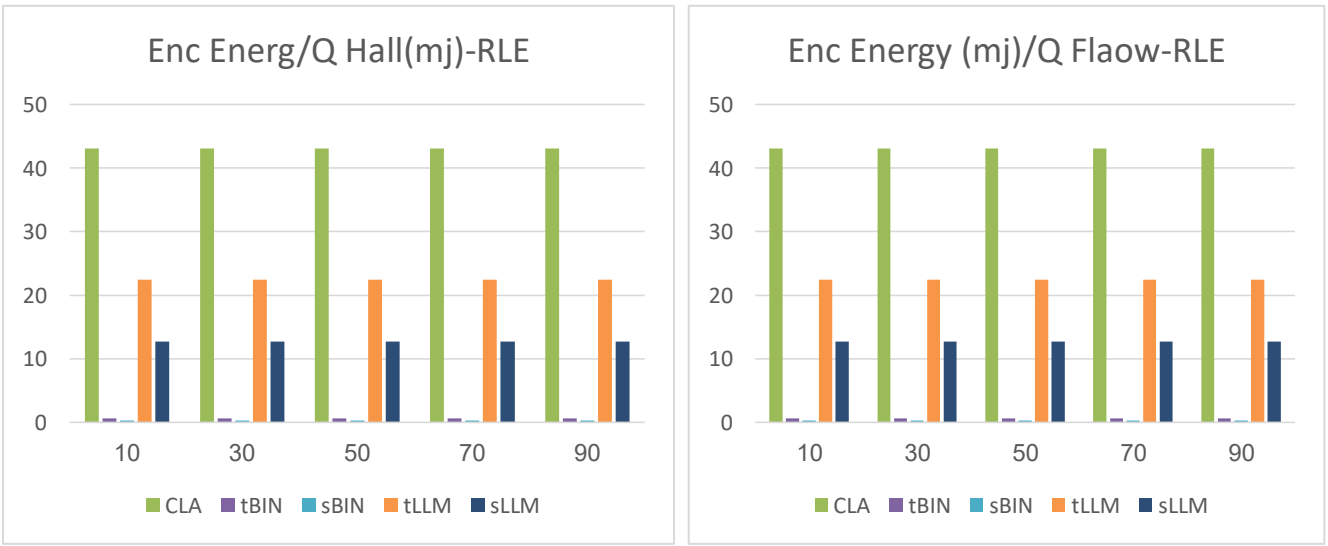


Figure 1.7 representation des donnee de capture facteur qualité / énergie

D'après ces résultats on peut constater:

- la consommation d'énergie de la DCT CLA et très élevé.
- le calcule rapide (tLLM et tBIN) consomme moins d'énergie et donc il est fortement souhaitable dans les WVSN.
- La DCT CLA consomme plus d'énergie par rapport au reste des transformations parce qu'elle s'agit de la DCT exacte.
- les restes des transformations (tBIN, sBIN, tLLM et sLLM) sont des techniques d'approximation DCT utilisées pour réduire la consommation d'énergie.

- On note également, selon les résultats, que la qualité vidéo d'origine n'a pas d'impact significatif sur la consommation d'énergie.

5-2 facteur qualité / PSNR :

PSNR capture avec RLE Enc entropique										
Vidéo	flaowr					Hall				
Q	10	30	50	70	90	10	30	50	70	90
CLA	23,766	27,373	29,848	33,056	40,603	26,432	30,732	32,947	35,276	41,129
tBIN	23,506	23,631	23,643	23,648	23,65	25,78	25,937	25,956	25,965	25,97
sBIN	22,334	22,352	22,354	22,354	22,354	24,741	24,775	24,779	24,78	24,78
tLLM	23,675	23,694	23,696	23,697	23,697	26,005	26,044	26,048	26,05	26,05
Sllm	22,405	22,411	22,412	22,412	22,412	24,844	24,859	24,86	24,861	24,861
PSNR capture avec EG Enc entropique										
CLA	23,766	27,373	29,848	33,056	40,603	26,432	30,732	32,947	35,276	41,129
tBIN	23,506	23,631	23,643	23,648	23,65	25,78	25,937	25,956	25,965	25,97
sBIN	22,334	22,352	22,354	22,354	22,354	24,741	24,775	24,779	24,78	24,78
tLLM	23,675	23,694	23,696	23,697	23,697	26,005	26,044	26,048	26,05	26,05
Sllm	23,766	27,373	29,848	33,056	40,603	24,844	24,859	24,86	24,861	24,861
PSNR capture avec HUFFMAN Enc entropique										
CLA	23,76	27,37	29,83	33,06	40,61	26,432	30,731	32,94	35,276	41,129
tBIN	22,33	22,35	22,35	22,35	0	24,741	24,775	24,779	24,780	0
sBIN	23,506	23,631	23,648	23,64	0	25,780	25,937	25,956	25,965	0
tLLM	22,405	22,411	22,412	22,412	0	24,844	24,85	24,860	24,861	0
Sllm	26,005	26,044	26,048	26,05	0	26,005	26,044	26,048	26,05	0

Tableau 1.8 résultat du capturé facteur qualité / PSNR

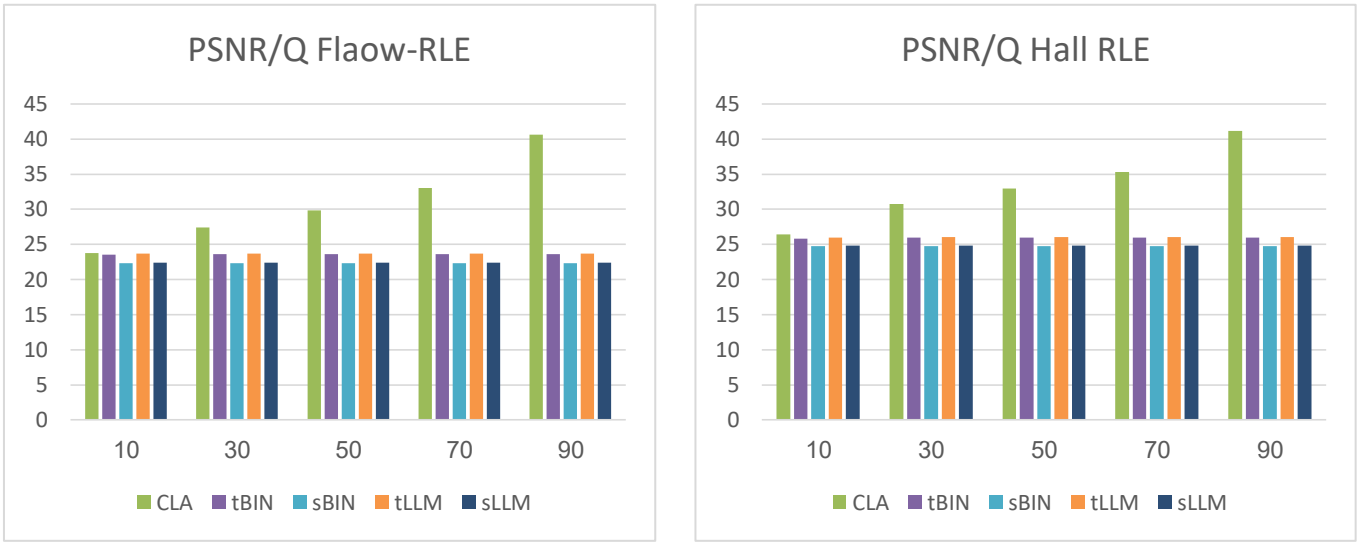


Figure 1.8 representation des donnee de capture facteur qualité / PSNR

- La DCT exacte représentée dans le CLA reste la meilleure si on veut garder un qualité maximale de la vidéo capturée.
- Les autres calcules donnent des résultats proche et acceptable.

- Tous les résultats sont proches si on utilise tBIN,sBIN, tLLM et sLLM cela explique que ces transformations perdent les détails de l'image.

- Au contraire, en utilisant CLA DCT les résultats sont différente, parce qu'il y a moins de perte dans les détails.

-

5-3 facteur qualité / SSIM :

SSIM capture avec RLE Enc entropique										
Vidéo	flaowr					Hall				
Q	10	30	50	70	90	10	30	50	70	90
CLA	0,844	0,933	0,961	0,979	0,995	0,829	0,915	0,942	0,961	0,984
tBIN	0,846	0,852	0,853	0,854	0,854	0,864	0,876	0,878	0,88	0,881
sBIN	0,789	0,791	0,791	0,791	0,791	0,838	0,844	0,845	0,845	0,845
tLLM	0,853	0,854	0,855	0,855	0,855	0,877	0,882	0,883	0,884	0,884
Sllm	0,792	0,793	0,793	0,793	0,793	0,844	0,848	0,848	0,848	0,848
SSIM capture avec EG Enc entropique										
CLA	0,844	0,933	0,961	0,979	0,995	0,829	0,915	0,942	0,961	0,984
tBIN	0,846	0,852	0,853	0,854	0,854	0,864	0,876	0,878	0,88	0,881
sBIN	0,789	0,791	0,791	0,791	0,791	0,838	0,844	0,845	0,845	0,845
tLLM	0,853	0,854	0,855	0,855	0,855	0,877	0,882	0,883	0,884	0,884
Sllm	0,792	0,793	0,793	0,793	0,793	0,844	0,848	0,848	0,848	0,848
SSIM capture avec HUFFMAN Enc entropique										
CLA	0,8447	0,9336	0,9617	0,9798	0,9953	0,8296	0,9159	0,9422	0,9614	0,9841
tBIN	0,7891	0,7910	0,7912	0,7913	0	0,8381	0,8442	0,8451	0,8455	0
sBIN	0,846	0,8528	0,8534	0,8540	0	0,8643	0,8763	0,8787	0,8801	0
tLLM	0,7925	0,7932	0,7933	0,7933	0	0,8449	0,8483	0,848	0,8487	0
Sllm	0,877	0,8828	0,8838	0,884	0	0,877	0,8828	0,8838	0,8841	0

Tableau 1.9 Résultat du capturé facteur qualité / SSIM

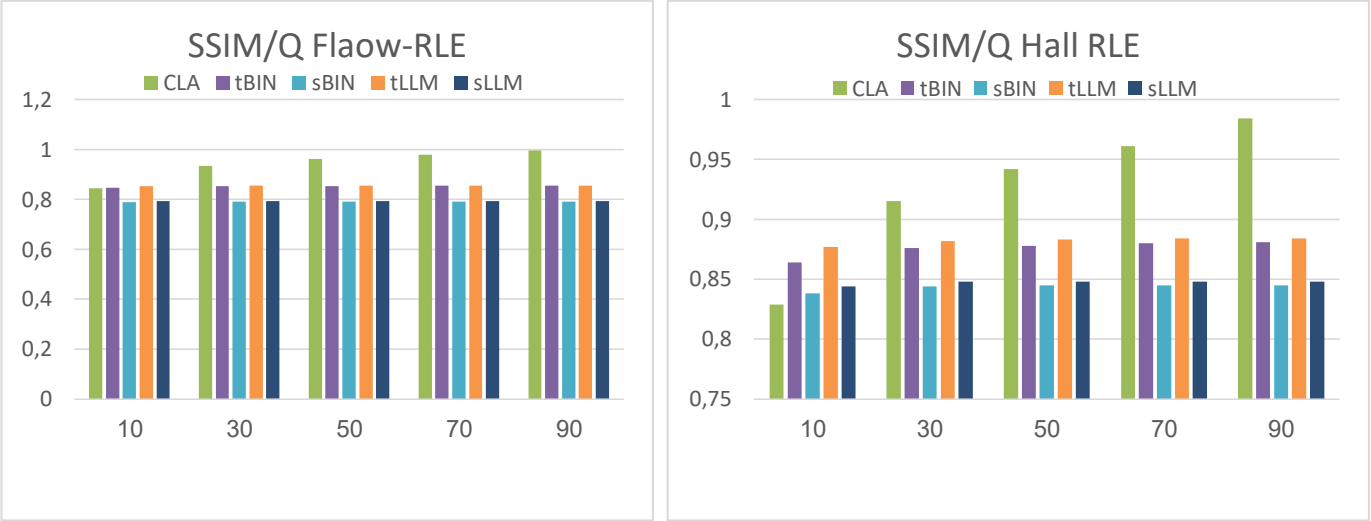


Figure 1.9 Représentation des données de capture facteur qualité / SSIM

- La qualité de la vidéo capturée est liée aux deux facteurs PSNR et SSIM , le calcule complet de la DCT reste toujours un facteur non négociable si en veut garder une qualité maximale dans le WVSN.

- Le facteur de qualité (Q) de la vidéo originale n'affecte pas beaucoup la quantité d'énergie consommée. Mais cela a un impact majeur sur la qualité de la vidéo encodée.

- La DCT CLA est préférée en qualité d'image. L'inverse en ce qui concerne la consommation d'énergie.

- En termes de consommation d'énergie, la DCT sBIN et tBIN sont privilégiées dans le domaine de WVSN, avec une qualité d'image très acceptable.

- Les transformations (BIN et LLM) affectent la qualité des images.

- Nous visons à réduire la consommation d'énergie et à améliorer la qualité dans le domaine des WVSN.

6- Conclusion :

Dans ce chapitre, nous avons présenté SenseVid, un outil de transmission et d'évaluation basé sur les traces de trafic destiné au WSN. SenseVid adopte des techniques de compression à faible coût calculatoire où le codage intra-trame est effectué à l'aide de DCT zonales et rapides.

SenseVid et un outil en cours de développement il est conçus en open source qui a permet aux chercheurs et utilisateur a paramétrer le logiciel et corrigée certain erreurs.

Nous avons aussi testé des différents scénarios avec le SenseVid et simulé l'impact de la variation des différentes DCT implémentées sur la qualité d'image et la consommation énergétique. L'impact du facteur de qualité a été aussi étudié.

Chapitre 2

Transformation orthogonales

1- Introduction :

En juin 2012, l'Union internationale des télécommunications a défini la norme IoT (**IoT**, Internet des objets **[22]** . En anglais Internet of Things ou IoT). Cette technologie s'appuie fortement sur les réseaux de capteurs, y compris les systèmes de capteurs vidéo sans fil ***WVSN***.

Nous avons traité dans le premier chapitre, les systèmes WVSN et l'emploi du simulateur **SeneVid** (outil de transmission et d'évaluation vidéo) pour évaluer ces systèmes dès la compression vidéos jusqu'au l'envoi. Dans ce chapitre nous traitons les principales techniques adoptées pour la compression des vidéos, notamment les transformations. Parmi ces techniques la transformée en cosinus discrète (DCT), la DWT (Discrète Wavelet Transform), la transformée de Karhunen-Loéve (KLT) et enfin la transformation discrète de Tchebychef **(DTT).**

Le codage par transformation fait partie intégrante des applications de traitement d'image et de vidéo. L'objectif principal est de mapper un ensemble de valeurs de pixels en quelques coefficients avec certaines propriétés souhaitées. Ces propriétés aident à reconstruire les pixels avec la moindre perte possible **[14].**

Les transformées et en particulier les transformées intégrales, sont utilisées principalement pour réduire la complexité des problèmes mathématiques. Les équations différentielles et les équations intégrales peuvent, par une application judicieuse des transformations appropriées, être transformées en équations algébriques dont les solutions sont plus faciles à obtenir. Il est donc important de dériver les mathématiques de base propriétés de ces transformations avant que son applications ne soient considérées **[15].**

2- La transformée de Fourier discrète (DFT):

La transformée de Fourier a été développée initialement pour étudier les fonctions de durée finie, et étendue aux fonctions périodiques. La transformation de Fourier repose sur le principe suivant : pratiquement toutes les fonctions sont décomposables en une somme de cosinus et de sinus à des fréquences différentes. Ainsi, lorsque l'on représente une fonction dans un repère Amplitude/Temps, la transformation de Fourier permet de la voir dans un repère Amplitude/Fréquence. On voit donc les composantes en fréquence d'un signal **[16],** appelés spectre du signal.

La transformée de Fourier du signal est calculée à l'aide de l'intégrale suivants :

$$X(t) = \int_{-\infty}^{+\infty} x(f) e^{-j2\pi ft} df \qquad (11)$$

Où, f est la fréquence du terme sinusoïdal.

On peut passer à la fonction d'origine à partir d'une transformée de Fourier en appliquant une transformation de Fourier inverse, la transformée inverse permet de reconstruire le signal à partir des sinusoïdes qui le constituent.

La transformée de Fourier inverse s'écrit comme suit :

$$X(t) = \int_{-\infty}^{+\infty} x(f) e^{i2\pi ft} df \qquad (12)$$

On appelle transformée de Fourier discrète (TFD) d'une suite de N termes x(0),x(1), . . . , x(N −1), la suite de N termes , X(0),X(1), . . . ,X(N −1), définis par :

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{nk}{N}} \qquad (13)$$

En pratique, les N termes x(n) peuvent être N échantillons d'un signal analogique échantillonné: xn = x(nTe), et les N termes X(k) correspondre à une approximation (à un facteur multiplicatif Te près) de la transformée de Fourier de ce signal aux N points de fréquence fk = k fe/N, avec k entre 0 et N −1, c'est-à-dire : f varie entre 0 et fe.

Les coefficients de Fourier qui sont inférieurs à un seuil donné sont rejetés (mise à zéro). La transformée de Fourier présente certains inconvénients :

- Les coefficients sont complexes.
- Perte de tous les aspects temporels du signal tels que le début et la fin, l'existence de singularité, etc
- Nécessité d'analyser le signal en temps et en fréquence.

2-1 Transformée de Fourier rapide (FFT) :

Cet algorithme est couramment utilisé en traitement numérique du signal pour transformer des données discrètes du domaine temporel vers le domaine fréquentiel, en

particulier dans les analyseurs de spectre, a été mis au point par **Cooley & Tukey** au début des années 1960 **[17]** . Il est également à la base des algorithmes de multiplication rapide (**Schönhage et Strassen**, 1971) **[18].**

La transformée de Fourier rapide sépare les fréquences paires des fréquences impaires lors du calcul d'une transformée de Fourier discrète pour diminuer le nombre d'opérations **[19].** Pour un signal X à N échantillons, toute fonction périodique du temps x(t) peut être décomposée en une somme infinie de sinus et cosinus dont les fréquences commencent à zéro et augmentent par multiples entiers d'une fréquence de base $f0$ =1/T, où T est la période de x(t).

La **FFT** profite des propriétés de symétrie et de périodicité de la TF pour réduire le calcule en temps. Dans ce processus, la transformation est partitionnée en une séquence de transformations de longueur réduite qui est collectivement réalisée avec un calcul réduit.

La technique **FFT** a également une limitation des performances comme méthode. La **FFT** est une transformation complexe qui fonctionne sur un nombre imaginaire et algorithme spécial. C'est une exponentielle complexe qui définit une sinusoïde complexe avec une fréquence **[20].**

3- Transformée discrète en sinus (DST) :

La transformée en sinus discrète (DST) est une transformée de Fourier similaire à la transformée de Fourier discrète (TFD), mais en utilisant une matrice purement réelle. Il est équivalent aux parties imaginaires d'un DFT d'environ deux fois la longueur, fonctionne sur des données réelles avec symétrie impaire (depuis la transformée de Fourier d'une fonction réelle et impaire est imaginaire et impair), où, dans certaines variantes, l'entrée et / ou de sortie les données sont décalées d'un demi-échantillon **[14].**

4- Transformée discrète en Cosinus (DCT) :

La transformée en cosinus discret (Discrete Cosines Transform. DCT) est l'élément essentiel de la compression du signal vidéo elle est utilisé pour réduire les redondances spatiales. La DCT est une transformation linéaire orthogonale. Elle est considérée comme une version simplifiée de la transformation de Fourier discrète (DFT). En effet, le noyau de projection utilisé pour la DFT est représenté par une exponentielle complexe (soit des

bases de sinus et cosinus) alors que le noyau de projection de la DCT est simplement une base de cosinus **[21].** Les coefficients de la transformée ne sont donc pas complexes mais réels ce qui présente un avantage pour le codage et la quantification, Il y a huit (8) variantes de la transformée discrète en cosinus. Seulement, les plus utilisées sont quatre (4). DCT I, DCT II, DCT III et DCT IV **[1].**

4-1 Transformée en cosinus discrète type I (DCT I) :

Les formules des transformations directe et inverse de DCT I sont présentées ci-dessous :

Transformation directe: Transformée en cosinus discrète type I (DCT I) :

$$X[k+1] = \sqrt{\frac{2}{N}}\alpha_k \sum_{n=0}^{N-1} \alpha_n x[n] \cos\left[\frac{\pi n k}{N}\right] \quad k = 0,...,N-1 \tag{14}$$

Transformation inverse :

$$x[n+1] = \sqrt{\frac{2}{N}}\alpha_n \sum_{k=0}^{N-1} \alpha_k X[k] \cos\left[\frac{\pi n k}{N}\right] \quad k = 0,...,N-1 \tag{15}$$

Avec, α_k constante définie comme suit :

$$\alpha_k = \begin{cases} \frac{1}{\sqrt{2}} & si\ k = 0 \text{ ou N} \\ 1 & si\ non \end{cases} \tag{16}$$

Cette forme de DCT I est définie pour les valeurs N entières supérieures à 2. La forme orthonormée matricielle de la transformation directe DCT type I peut être exprimée comme suit :

$$\left[C_{N+1}^{I}\right]_{k,n} = \sqrt{\frac{2}{N}}\left[\alpha_k \alpha_n \cos\frac{\pi n k}{N}\right] \quad k,n = 0,...,N-1 \tag{17}$$

et celle de sa transformée inverse peut être $\left[C_{N+1}^{I}\right]_{k,n}^{-1} = \left[C_{N+1}^{I}\right]_{k,n}^{T}$ exprimée comme :

Ou $\left[C_{N+1}^{I}\right]_{k,n}^{T}$ représente la matrice inverse, et $\left[C_{N+1}^{I}\right]_{k,n}^{-1}$ représente la matrice transposée.

4-2 Transformée en cosinus discrète type II(DCT II)

Les expressions des transformations directe et inverse de la DCT II sont présentées ci-dessous Transformation directe:

$$X[k]=\sqrt{\frac{2}{N}}\alpha_k\sum_{n=0}^{N-1}x[n]\cos\left[\frac{\pi(2n+1)k}{2N}\right]\quad k=0,...,N-1 \tag{18}$$

Transformation inverse :

$$x[n]=\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1}\alpha_k X[k]\cos\left[\frac{\pi(2n+1)k}{2N}\right]\quad n=0,...,N-1 \tag{19}$$

Cette forme est définie pour toutes les valeurs N positives. Elle est généralement désignée par DCT c'est la forme la plus utilisée. La forme orthonormée matricielle de la transformation directe DCT type II peut être exprimée Comme suit :

$$\left[C_N^{II}\right]_{k,n}=\sqrt{\frac{2}{N}}\left[\alpha_k\cos\frac{\pi(2n+1)k}{2N}\right]\quad k,n=0,...,N-1 \tag{20}$$

4-3 Transformée discrète en cosinus type III (DCT III) :

Les expressions des transformations directe et inverse de DCT III sont présentées ci-dessous Transformation directe:

$$X[k]=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1}\alpha_n x[n]\cos\left[\frac{\pi(2n+1)n}{2N}\right]\quad k=0,...,N-1 \tag{21}$$

$$x[n]=\sqrt{\frac{2}{N}}\alpha_n\sum_{k=0}^{N-1}X[k]\cos\left[\frac{\pi(2n+1)n}{2N}\right]\quad n=0,...,N-1 \tag{22}$$

Transformation inverse :

$$\left[C_N^{III}\right]_{k,n}=\sqrt{\frac{2}{N}}\left[\alpha_n\frac{\pi(2n+1)k}{2N}\right]\quad k,n=0,...,N-1 \tag{23}$$

4-4 Transformée en cosinus discrète type IV (DCT IV)

Les expressions des transformations directe et inverse de DCT IV sont présentées ci-suivante :

Transformation directe :

$$\left[C_N^{IV}\right]_{k,n} = \sqrt{\frac{2}{N}}\left[\alpha_k \cos\frac{\pi(2n+1)(2k+1)}{4N}\right] \quad k,n = 0,\ldots,N-1 \tag{24}$$

Transformation inverse :

$$x[n] = \sqrt{\frac{2}{N}}\sum_{k=0}^{N-1} X[k]\cos\left[\frac{\pi(2n+1)(k+1)}{4N}\right] \quad n = 0,\ldots,N-1 \tag{25}$$

La forme orthonormée matricielle de la transformation directe DCT type IV peut être exprimée comme suit :

$$[C_N^{IV}]_{k,n} = \sqrt{\frac{2}{N}}\left[\alpha_k \cos\frac{\pi(2n+1)(2k+1)}{4N}\right] \quad k,n = 0,1,\ldots,N-1 \tag{26}$$

L'analyse de ces formes montre les points suivants:

- Les transformées en cosinus de type I et de type IV sont représentables par des matrices involutives dont les inverses sont des matrices de la transformation directe elle-même,
- La transformée inverse de type II est similaire à la transformée directe de type III.

La DCT II est très utilisée en traitement du signal et de l'image et spécialement en compression **[1].**

4-5 Transformation DCT unidimensionnelle (DCT 1D) :

Pour une séquence d'entrée *x(n)* , *n* est un nombre entier naturel vrai entre [0, N-1] , les coefficients de la DCT (u) , *u* nombre entier naturel vari entre [0 , N-1], sont donnée par :

$$X(u) = C(u)\sum_{n=0}^{N-1} x(n)\cos\frac{(2n+1)u\pi}{2N} \tag{27}$$

Où N est un nombre entier naturel qui représente la taille de la DCT. Cette transformée est réversible permettant de restituer le signal d'origine (temporel). Pour cela il suffit

d'appliquer la Transformée en Cosinus Discrète Inverse : TCDI (IDCT « Inverse Discrete Cosine Transform ») au spectre du signal X . Cette transformation inverse est donnée par :

$$x(n)=\sum_{u=0}^{N-1}C(u)\,X(u)\cos\frac{(2n+1)u\pi}{2N} \qquad (28)$$

Les constantes de pondération C(u) utilisées dans les équations 3.1 et 3.2 sont données par :

$$C(u)=\begin{cases}\dfrac{1}{\sqrt{N}} & si\ u=0\\[2ex] \dfrac{2}{\sqrt{N}} & sinon\end{cases} \qquad (29)$$

En utilisant cette définition (équation 29) et en remplaçant C (0) par sa valeur dans l'équation 2.18, le coefficient DCT à la fréquence 0 est donné par :

$$X(0)=\frac{1}{\sqrt{N}}\sum_{n=0}^{N-1}x(n) \qquad (30)$$

Ainsi, la première valeur de la transformée représente la somme de toute les entrées pondérées Par 1/√N. Cette valeur est appelée DC (Direct Component) et représente l'offset de la transformée. Toutes les autres valeurs sont nommées coefficients AC (Alternative Component) Le calcul des coefficients de la DCT peut se faire avec une combinaison linéaire d'un ensemble de fonctions définies par :

$$F_n(u)=C(u)\cos\frac{(2n+1)u\pi}{2N} \qquad (31)$$

L'avantage d'une telle décomposition réside dans le fait que ces fonctions, indépendantes de x(n) , peuvent être pré-calculées à l'avance. Ces fonctions sont appelées également noyaux de la transformation. Ainsi, le calcul des coefficients de la DCT s'effectue en réalisant une combinaison linéaire entre le signal d'entrée x(n) et ces fonctions. Notons que toutes ces fonctions sont orthogonales. Les fonctions orthogonales sont indépendantes, ainsi aucune de ces fonctions ne peut être obtenue par une combinaison linéaire des autres. La forme matricielle associée à l'équation (30) est la suivante :

$$X_{N\times 1}=F_{N\times N}.x_{N\times 1} \qquad (32)$$

$F_{N\times N}$ est la matrice associé à (27), elle est définie comme suit :

$$F_{N\times N} = \left[C_N^{II}\right]_{u,n} = \left[C(u)\cos\frac{(2n+1)u\pi}{2N}\right] \quad u,n = 0,...,N-1 \qquad (33)$$

Enfin, notons que si la taille de la séquence d'entrée est supérieure à N, elle peut être divisée en plusieurs sous-séquences chacune de taille N. Par la suite la DCT peut être obtenue en utilisant les sous-DCT calculées avec les sous-séquences.

4-6 Transformation bidimensionnelle (DCT-2D) :

La DCT-2D est effectuée sur une matrice carrée N× N de pixels et donne une matrice carrée N × N de coefficients fréquentiels. Elle est très utilisée dans les applications de traitement d'images et de vidéo comme la compression et le codage. Le calcul des coefficients d'une matrice d'entrée peut se faire par l'équation (34).

$$Y(v,u) = \sum_{j=0}^{N-1}\sum_{i=0}^{N-1}\frac{C(v)}{2}\frac{C(u)}{2}y(i,j)\cos\frac{(2i+1)u\pi}{2N}\cos\frac{(2j+1)v\pi}{2N} \qquad (34)$$

C(u) et C(v) sont définis par l'équation (28) et u,v ,i , j et sont des entiers naturels vari entre [0.N-1]. La matrice d'entrée y(i,j) peut être recalculée à partir de y(u,v) en utilisant la DCT 2D inverse IDCT 2D définie par :

$$y(i,j) = \sum_{v=0}^{N-1}\sum_{u=0}^{N-1}\frac{C(v)}{2}\frac{C(u)}{2}Y(v,u)\cos\frac{(2i+1)u\pi}{2N}\cos\frac{(2j+1)v\pi}{2N} \qquad (35)$$

De la même manière que pour la DCT 1D, la DCT 2D peut être écrite sous la forme d'une combinaison linéaire d'un ensemble de fonctions appelées fonctions de base. Le calcul de ces fonctions est réalisé par l'équation (36).

$$F_{i,j}(u,v) = \frac{C(v)}{2}\frac{C(u)}{2}\cos\frac{(2i+1)u\pi}{2N}\cos\frac{(2j+1)v\pi}{2N} \qquad (36)$$

Par conséquent, l'équation permettant de calculer les coefficients de la DCT 2D est donnée par :

$$Y(v,u) = \sum_{j=0}^{N-1}\sum_{i=0}^{N-1}F_{i,j}(u,v)y(i,j) \qquad (37)$$

Les fonctions de base de la DCT 2D peuvent être générées à partir des fonctions de base 1D (voir équation (30)). Ceci est assuré grâce à la propriété de séparabilité du noyau de la transformation de la DCT2D.

Le développement des algorithmes de calcul rapide des transformées DCT se basent sur la possibilité de décomposer la matrice de définition sous forme d'un produit de matrices dont le calcul est plus simple, et permet de réduire le nombre de multiplications scalaires, en profitant des identités remarquables de périodicité et symétries des fonctions sinusoïdales. Ainsi, on peut décomposer toute transformée DCT de R^N en transformées plus simples en décomposant N en produit de facteurs premiers, et en composant des sous-transformées dans R^n où n est l'un de ces facteurs. En particulier, de nombreuses optimisations ont été développées quand N est une puissance de 2.

Cela revient à réécrire la matrice N × N sous forme de produit de sous-matrices identiques (disposées en pavage régulier et utilisant donc des coefficients réels communs ou différenciés uniquement par leur signe) et de matrices à coefficients unitaires ou nuls (-1, 0 ou 1), ces dernières ne nécessitant pas de multiplication.

4-7 LLM (Loeffler, Lieenberg, Moschytz) DCT:

La réduction du nombre des opérations gourmandes en énergie telles que les multiplications. Il a été démontré théoriquement que le nombre minimum pour calculer la DCT 8 points est 11 multiplications **[14].** L'un des algorithmes les plus célèbres dans la littérature, qui atteint cette limite est l'algorithme de Loeffler **[8].** D'autres algorithmes existent dans la littérature pour réduire la complexité de Calcul de la DCT. Les auteurs de **[23-24]** proposent de convertir les multiplications en des opérations simples, telles que les additions et les décalages. En effet, la DCT peut être implémentée en utilisant seulement des opérations d'additions et de décalages. Cependant, ces dernières sont des approximations et par conséquent elles dégradent la qualité des images finales. En outre, ces systèmes permettent un faible coût et une implémentation rapide par rapport à la DCT exacte. Une liste détaillée de ces différents algorithmes est disponible dans **[15].**

Les auteurs de [8] on trouvé une classe d'algorithmes DCT à 8 points nécessitant uniquement 11 multiplications et 29 ajouts. En fait, la même méthode peut être appliquée à la plupart des algorithmes connus **[1]** pour réduire le nombre de multiplications pour chacune.

Le nombre de multiplications a été réduit à la borne inférieure théorique sans augmentation du nombre d'additions par rapport aux d'autres algorithmes. la figure 2.1 illustre ce dernier . Les étapes de l'algorithme sont numérotées de 1 à 4:

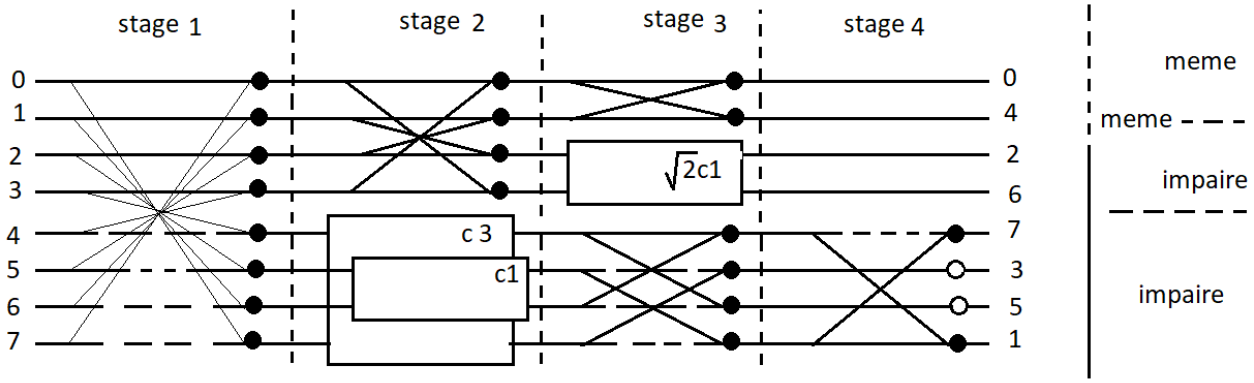


Figure 2.1 : Algorithme DCT à 8 points avec 11 multiplications.

Les algorithmes de Loeffler et de Chen [145] sont des algorithmes basés sur la méthode de calcul direct. Ces algorithmes sont les plus utilisés dans le calcul des coefficients de la DCT nécessitant qu'un nombre réduit

D'éléments de contrôle et de mémorisation. Le schéma bloc de la DCT de taille 8 points basée sur l'algorithme de Loeffler est illustré dans la Figure 2.2.

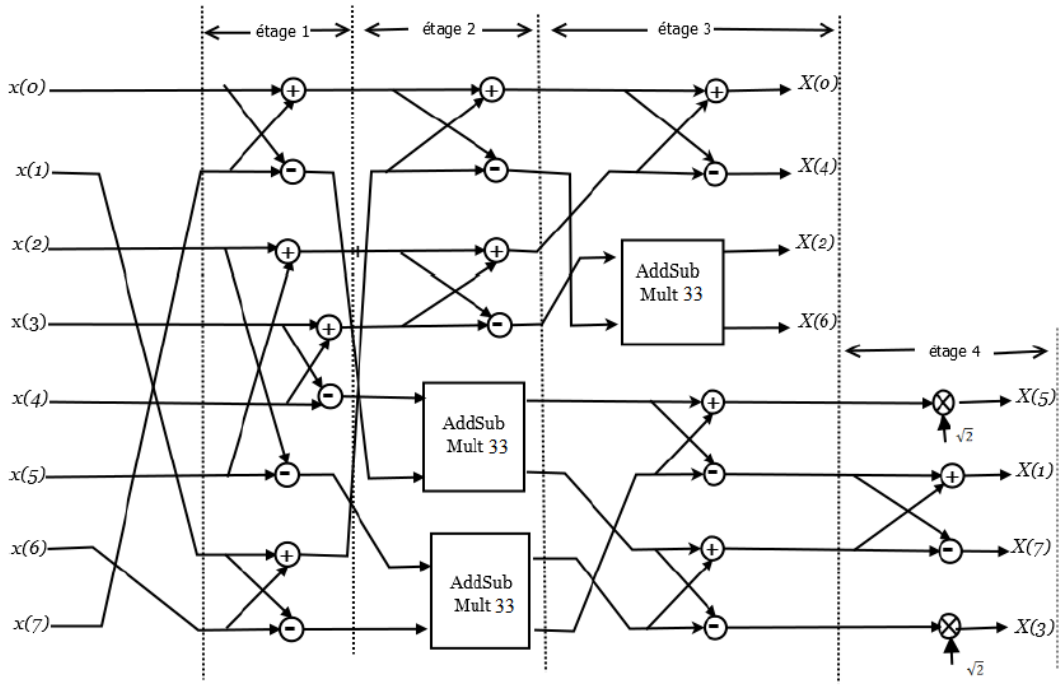


Figure 2.2 : Schéma bloc de l'algorithme de Loeffler 8 points.

Le bloc AddSubMult33 utilise 3 multiplieurs et 3 additionneurs Au total, l'algorithme de Loeffler consomme 10 blocs composé chacun d'un additionneur et un

soustracteur, 3 blocs de AddSubMult33 et 2 blocs de multiplications par constante. Par conséquent, le nombre total d'opérateurs arithmétiques utilisés par l'algorithme de Loeffler est 11 multiplieurs et 29 additionneurs.

4-8 BDCT, la DCT binaire:

Les auteurs dans [9], se sont intéressés à la recherche d'une matrice binaire qui se rapproche de la matrice DCT. L'idée la plus simple consiste à appliquer une fonction 'signe' sur la matrice DCT pour obtenir une nouvelle matrice dont les entrées sont seulement 1 ou -1. La matrice résultante de la fonction signe est appelée 'SDCT' . Cependant, cette matrice n'est pas orthogonale et par conséquent, la matrice inverse contient d'autres éléments autres que le 1 et le -1. Elle nécessite donc des multiplications. En outre, cette approche ne peut pas générer une DCT binaire (BDCT). Afin d'assurer l'orthogonalité, nous utilisons la matrice de Hadamard pour approcher la matrice DCT.

La procédure proposée dans [153] pour la construction est de comparer une ligne donnée de la matrice SDCT avec toutes les lignes de la matrice de Hadamard et la remplacer par le plus proche. La procédure est répétée pour toutes les lignes de la matrice SDCT. Nous illustrons ci-après cette procédure pour le cas N = 8. Donnons la matrice DCT (38) comme suit :

$$C_8 = \begin{bmatrix} e & e & e & e & e & e & e & e \\ a & b & c & d & -d & -c & -b & -a \\ f & g & -g & -f & -f & -g & g & f \\ b & -d & -a & -c & c & a & d & -b \\ e & -e & -e & e & e & -e & -e & e \\ c & -a & d & b & -b & -d & a & -c \\ g & -f & f & -g & -g & f & -f & g \\ d & -c & b & -a & a & -b & c & -d \end{bmatrix} \qquad (38)$$

a, b, c, d, e, f et g sont des nombres positifs. La matrice SDCT est obtenue comme suit :

$$\breve{C}_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} \quad (39)$$

La matrice SDCT n'est pas orthogonale, c.-à-d. sa matrice inverse est différente de sa matrice transposée.

$$\breve{C}_8^{-1} = \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 & 2 & 1 & 0 & 1 & 0 \\ 1 & 2 & 1 & 0 & -1 & -2 & -1 & 0 \\ 1 & 0 & -1 & -2 & -1 & 0 & 1 & 2 \\ 1 & 0 & -1 & 0 & 1 & 2 & -1 & -2 \\ 1 & 0 & -1 & 0 & 1 & -2 & -1 & 2 \\ 1 & 0 & -1 & 2 & -1 & 0 & 1 & -2 \\ 1 & -2 & 1 & 0 & -1 & 2 & -1 & 0 \\ 1 & -2 & 1 & -2 & 1 & 0 & 1 & 0 \end{bmatrix} \quad (40)$$

Dans ce qui suit, nous comparons chaque ligne de la matrice SDCT avec toutes les lignes de la matrice de Hadamard (WHT). La matrice WHT est la suivante :

$$H_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \quad (41)$$

Le résultat de cette comparaison en termes d'éléments de lignes différents de la matrice SDCT par rapport aux éléments de lignes de la matrice de Hadamard est résumé dans le tableau 2.1 .

	Lignes de la matrice H							
Lignes de la SDCT	1	2	3	4	5	6	7	8
1	0	4	4	4	4	4	4	4
2	4	4	4	4	0	4	4	4
3	4	4	4	4	4	4	0	4
4	4	2	2	4	6	4	4	2
5	4	4	4	0	4	4	4	4
6	4	2	6	4	2	4	4	2
7	4	4	4	4	4	0	4	4
8	4	0	4	4	4	4	4	4

Tableau 2.1 : Comparaisons des lignes de la matrice SDCT avec ceux de la matrice de HADAMARD pour N = 8 en termes de nombre d'entrées qui sont différentes. Après les remplacements convenables nous arrivons à la matrice BDCT [153] présentée ci a près :

$$\hat{C}_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} \quad (42)$$

En appliquant la procédure précédente sur un N arbitraire puissance de 2, nous obtenons la formule générale de la matrice binaire BDCT comme suit :

$$BDCT_{n,k} = (-1)\sum_{i=0}^{r-1}(n_i + n_{i+1})k_{r-1-i} \ , 0 \le n,k \le N-1 \quad (43)$$

Le BDCT a également été implémenté dans le codage vidéo norme H.263 +, basée sur un logiciel du domaine public H.263 + [9]. Le DCT dans le codeur de l'implémentation H.263 + sélectionnée est basé sur la factorisation de Chen avec virgule flottante multiplications, et le DCT dans le décodeur est la version à l'échelle de cette méthode avec multiplications à virgule fixe. Dans H.263 +, une étape de quantification uniforme est appliquée à tous les coefficients DCT d'un bloc. Dans la version basée sur BDCT, l'étape de quantification est modifié par la matrice de mise à l'échelle BDCT 2-D pour maintenir la compatibilité avec la norme.

5- Transformation de Karhunen-Loeve (KLT):

La transformation de Karhunen-Loève (KLT) est la transformation dite optimal. C'est-à-dire elle dé-corrèle les coefficients complètement. La DCT est la transformation la plus proche de la transformation KLT (optimal) dans la dé-corrélation des données **[14].**

les fonction , $\Phi_i(x)$ qui constituent la base orthogonale avec laquelle on peut approximer au mieux une fonction s(x) , doivent être solution de l'équation intégrale :

$$\int_{-A/2}^{+A/2} R(X, X')\Phi_i(X')dX' = \mathrm{E}\left[\left|a_j\right|^2\right]\Phi_i(X) \quad (44)$$

Avec R(x,x') fonction d'auto covariance de s(x) ,

- S(x) : fonction de carré intégrable ou d'énergie finie, définie sur l'intervalle [–A/2,A/2] ;

pour une image,s(x) représente l'évolution du niveau de gris suivant la direction x, de x= –A/2 à x= +A/2,

- a_i : coefficient i de la combinaison linéaire qui approxime s(x) (formule (45) **[25],**

$$a_m = \sum_{k=0}^{N-1} s(k\Delta X)\Phi_m(k\Delta X) \quad (45)$$

- Φi(x) : i[e] fonction de la base orthogonale.

La fig 2.3 suivante illustre l'histogramme d'une photo naturelle :

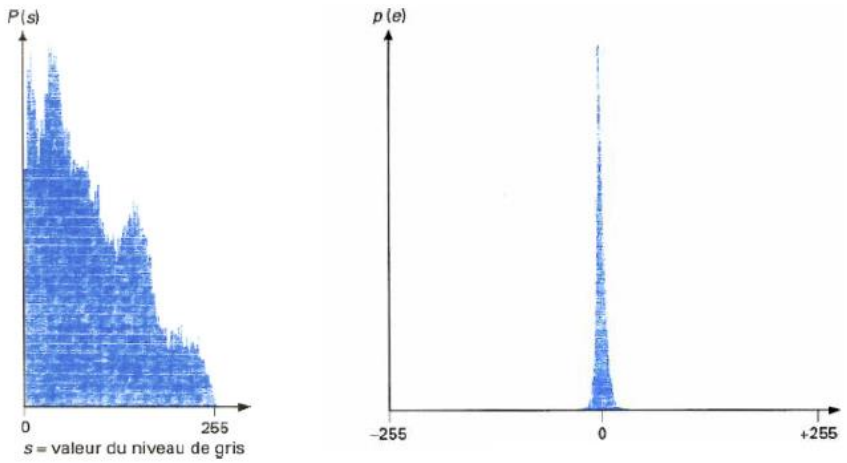


Fig 2.3 : a gauche, Histogramme des niveaux de gris d'une image naturelle ,s niveaux de gris.
a droit, Histogramme des différences de niveau de gris d'une image naturelle, e valeur de la déférence de niveaux de gris.

Bien que la génération de KLT implique l'estimation des matrices de corrélation / covariance avec leur diagonalisation conduisant à des valeurs propres et des vecteurs propres, une des références sous KLT indiquent leur applications dans la compression d'image, compression d'image multi-spectrale, segmentation d'image et indexation , filtrage récursif, restauration d'image , image représentation, récupération et analyse , codage d'images multicouches , neuronal regroupement , reconnaissance vocale , reconnaissance du locuteur , vérification du locuteur , sélection de caractéristiques , classification de texture , image et vidéo récupération ,etc **[14].**

6- Transformation en ondelette discret, (DWT) :

Les ondelettes c'est d'abord une théorie mathématique récente d'analyse du signale développée dans les années 80. On peut considérer qu'il s'agit d'une extension de l'analyse de Fourier.

Elles sont des fonctions générées à partir d'une fonction mère Ψ, par dilatations et translations. Dans le cas monodimensionnel, la fonction s'écrit :

$$\psi_{(a,b)}(t) = \frac{1}{\sqrt{a}} \Psi\left(\frac{t-b}{a}\right) \quad (46)$$

Où l'indice a représente un facteur d'échelle et l'indice b est un facteur de translation. On a un signal continu et on le décompose en une série de nombres qui décrivent des courbes qui s'additionnent pour reconstruire le signal. L'intérêt de cette théorie est au départ l'analyse des signaux et elle a déjà de nombreuses applications.

6-1 Décomposition en deux Dimensions (2D-DWT) et niveaux de résolution :

Le principe de la décomposition 2D par filtrage passe-bas (H0) et passe-haut (H1) sur une image est représenté en figure (III-2). L'image est d'abord filtrée dans la direction horizontale (Ligne), et sous-échantillonnée avec un facteur de 2, ensuite dans le sens vertical (Colonne), et sous-échantillonnée avec un facteur de 2 aussi. Nous obtenons ainsi quatre sous bandes (Sous-images) dans le domaine fréquentiel. Ces sous-bandes sont des matrices dont les dimensions sont réduites d'un facteur deux , figure 2.4

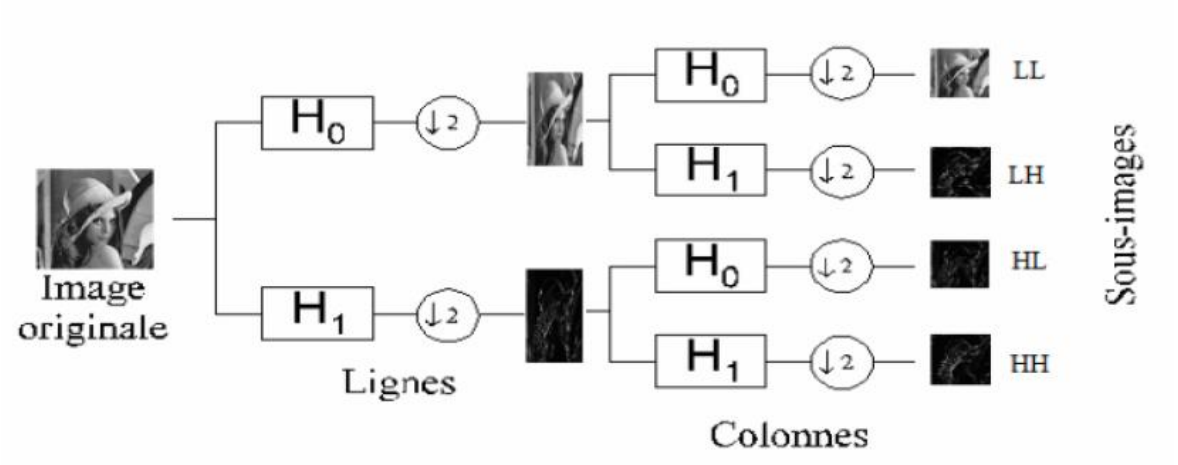


Figure (2.4): Principe de décomposition 2D par filtrage passe-bas (H0) et passe-haut (H1) dans le sens horizontal (ligne) et vertical (colonne).

LL (résulte du filtre passe bas (H0) dans les directions horizontale et verticale) correspond aux basses fréquences de l'image originale, ses coefficients sont donc les plus significatifs, LH (résulte du filtre passe - bas (H0) horizontal et passe- haut (H1) vertical) correspond aux détails verticaux de l'image originale, HL (résulte du filtre passe- haut (H1) horizontal et passe- bas (H0) vertical) correspond aux détails horizontaux de l'image originale et le HH (résulte du filtre passe-haut (H1) horizontal et vertical) correspond aux détails diagonaux de l'image originale. Nous obtenons ainsi le premier niveau de résolution de la décomposition en ondelettes. Pour obtenir une décomposition à plusieurs niveaux de résolution en ondelettes, il suffit de décomposer de la même façon la sous-bande LL, nous obtenons alors un deuxième niveau de résolution de la décomposition, et ainsi de suite comme le montre le diagramme de Mallat en figure (2.5). La figure (2.6) donne un exemple visuel de l'image Barbara décomposée en deux niveaux de résolution de la décomposition en ondelettes.

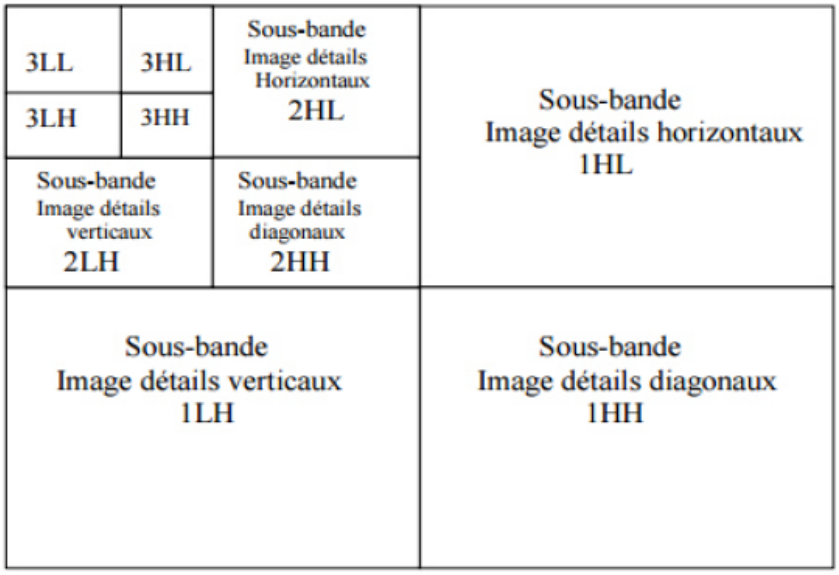


Figure (2.5): Diagramme de Mallat représentant les coefficients d'ondelettes de la Transformée, classés par sous-bandes de filtrage et niveau de décomposition.

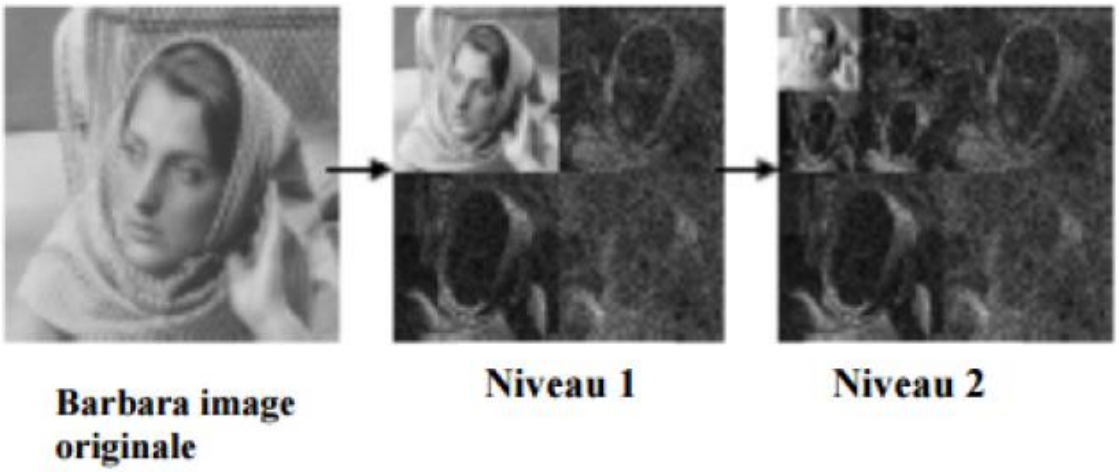


Figure (2.6): Exemple décomposition en ondelettes à 2 niveaux de résolution.

La compression par Ondelette est une technique récente qui donne de très bons résultats, Même avec des taux de compression élevés. Cette méthode reste encore marginale par rapport à l'utilisation de JPEG. Cependant la DWT est implémenté avec de la mémoire algorithmes intensifs et chronophages et a donc une ressource système très élevée et exigent [2]. Par conséquent, il ne convient pas aux applications multimédias sur la mémoire limitée des appareils portables, tels que les appareils photo numériques et les nœuds de capteurs dans les WMSN.

7- Transformation de Tchebichef Discrète (DTT) :

La DTT est basée sur les polynômes de Tchebichef orthogonaux discrets et a presque les mêmes propriétés que le DCT [27-28] elle conduit à des coefficients dé-corrélés. Elle garantit un bon compactage de l'énergie. Si l'on compare la complexité des différentes transformations archivé dans la littérature.

La capacité de compactage énergétique de la transformation utilisée dans un schéma de compression avec perte est mesurée à travers le gain de codage. Les auteurs de [29] ont montrée dans le tableau 2.2, le gain du codage (Cg) pour plusieurs transformations. Ce tableau montre que la DTT est le meilleur candidat pour la compression d'image avec perte par rapport à la transformation de Walsh-Hadamard (WHT), Haar, SDCT et la BDCT.

	DCT	DTT	WHT	BDCT	Haar	SDCT
Cg(dB)	8.8257	8.6756	7.9461	7.9461	7.9425	7.7905

Tableau 2. 2 : Gain de codage pour plusieurs transformations. Les résultats sont donnés pour un coefficient de corrélation $\rho = 0,95$.

La DTT possède un Cg considérable (égal à 8,6756 pour un coefficient de corrélation $\rho = 0,95$) ce qui va donner un PSNR élevé lorsqu'on place la transformation dans une chaîne de compression avec perte.

Les transformations précédentes citées ont été intégrées dans une chaine JPEG. Le PSNR a été utilisé comme métrique de distorsion. Les résultats représentés dans les figures ci-dessous montrent une comparaison de distorsion (RD) pour un ensemble d'images de test selon le nombre de coefficients retenus (un bloc de transformation contient 64 coefficients). Ayant un PSNR très proche du DCT, on peut conclure de ces courbes que la DTT est la meilleure candidat pour la compression d'images de faible complexité.

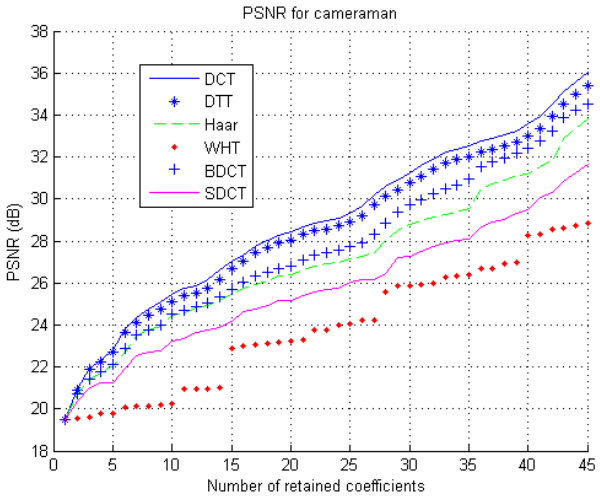


Fig. 2.7 : comparaison de distorsion avec les différentes transformations.

La transformée 1-D DTT directe et inverse d'un vecteur de 8 points sont données par les équations (46) et (47), respectivement :

$$Y_{8\times1} = C_{8\times8}.X_{8\times1} \quad (46)$$

$$X_{8\times1} = C^{-1}_{8\times8}.Y_{8\times1} = C^{1}_{8\times1}.Y_{8\times1} \quad (47)$$

Où $Y_{8\times1}$ et $Y_{8\times1}$ les vecteurs de l'entré et de la sortie, respectivement. La matrice $C_{8\times8}$ est donnée par l'équation (48) :

$$C_{8\times8} = D_{8\times8}.T_{8\times8} \quad (48)$$

Où $T8\times8$ est une matrice entière et $D8\times8$ est une matrice diagonale. Les deux matrices sont données respectivement par (49) et (50) :

$$T_{8\times8} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -7 & -5 & -3 & -1 & 1 & 3 & 5 & 7 \\ 7 & 1 & -3 & -5 & -5 & -3 & 1 & 7 \\ -7 & 5 & 7 & 3 & -3 & -7 & -5 & 1 \\ 7 & -13 & -3 & 9 & 9 & -3 & -13 & 7 \\ -7 & 23 & -17 & -15 & 15 & 17 & -23 & 7 \\ 1 & -5 & 9 & -5 & -5 & 9 & -5 & 1 \\ -1 & 7 & -21 & 35 & -35 & 21 & -7 & 1 \end{bmatrix} \quad (49)$$

$$D_{8\times8} = \frac{1}{2} diag\left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{42}}, \frac{1}{\sqrt{42}}, \frac{1}{\sqrt{66}}, \frac{1}{\sqrt{154}}, \frac{1}{\sqrt{546}}, \frac{1}{\sqrt{66}}, \frac{1}{\sqrt{858}} \right) \quad (50)$$

L'une des propriétés de la DTT c'est que la matrice $D_{8\times8}$ peut-être fusionnée dans l'étape de quantification, par conséquent elle ne génère aucun temps de calcule, la seule source de complexité c'est la $T_{8\times8}$. C'est ce qui rend l'algorithme de la DTT moins complexe par rapport à la DCT.

La transformée de la DTT s'applique sur les blocs des images selon la multiplication matricielle suivante (51) :

$$Y_{8\times8} = C_{8\times8}.X_{8\times8}.C'_{8\times8} \quad (51)$$

Où $S_{8\times8}$ et $R_{8\times8}$ sont les résultats de la transformation et le bloc de 8×8 à transformer. Cependant la transformée inverse (52) de la DTT sera comme suit :

$$X_{8\times8} = C'_{8\times8}.Y_{8\times8}.C_{8\times8} \quad (52)$$

7-1 Propriétés de la transformation DTT :

A .Symétrie :

Le calcul des lignes et colonnes contenues dans la DTT montre que cette opération semble être fonctionnellement identique. Cela signifie que la transformation est symétrique. Cette méthode est souvent utilisée dans le codage par transformation pour obtenir une réduction substantielle de la quantité de calcul mathématique. Considérons l'opération suivante:

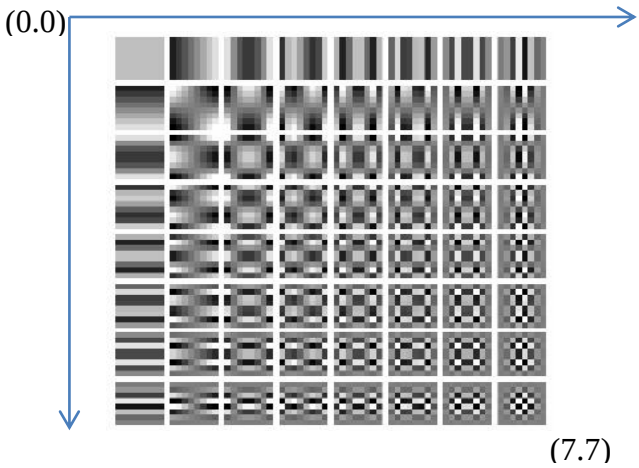
$$t_p(M-1-x) = (-1)\, p_{t_{p'}},\ p = 0,\ldots,M-1 \qquad (53)$$

B .Séparabilité :

la DTT 2D est séparable et pour l'obtenir on peut appliquer deux fois la transformation 1-D DTT.

C .Orthogonalité :

La DTT et orthogonale. Dans la figure 2.8, l'image de base de la transformation matricielle 8x8, où l'on trouve le plus bas les fréquences dans le coefficient (0,0) et la fréquence la plus élevée dans le coefficient (7,7). Pour la ligne de fonction de base, la fréquence augmente horizontalement tandis que la colonne augmente la fréquence verticalement.



Figue2.8 : Images de base 8 × 8 de Tchebichef orthonormé Polynômes.

7-2 Approximation et algorithme rapide pour la DTT :

Une classe d'approximations DCT a été introduite sur la base de la relation suivante: round (α · C), où round (·) est la fonction round telle que définie dans les langages C et Matlab , α est un paramètre réel et (C) , est la matrice DCT exacte . Les développeurs de l'article [31] ont proposé une approche similaire pour obtenir une approximation DTT à 8 points.

Ainsi, en introduisons une famille paramétrique de matrices DTT approximatives T (α), qui sont donnés par:

$$T(\alpha) = round(\alpha.T.D_0) \quad (54)$$

Avec :

$$D_0 = diag\left(\sqrt{\frac{6}{7}}, \frac{\sqrt{154}}{13}, \frac{\sqrt{66}}{9}, \frac{\sqrt{858}}{35}, \frac{\sqrt{858}}{35}, \frac{\sqrt{66}}{9}, \frac{\sqrt{154}}{13}, \sqrt{\frac{6}{7}}\right). \quad (55)$$

Le tableau (2.3) ci-dessous présente les opérations arithmétiques par lesquelles l'algorithme de la DTT passe.

Etape 1	$u0=x0+x7$ $u1=x1+x6$ $u2=x2+x5$ $u3=x3+x4$	$v0=x0-x7$ $v1=x1-x6$ $v2=x2-x5$ $v3=x3-x4$
Etape 2	$k0=u0+u2$ $k1=u1+u3$ $k2=u1-u3$	$z0=v0+v3$ $z1=v1-v2$ $z2=v1+v2$ $z3=v1-v3$ $z4=3(v2-3v3)$
Etape 3	$m0=k0+k1$ $m1=k0-k1$ $m2=2(3u0-2u2)$ $m3=k0-k2$	$w0=-(z1+z0)$ $w1=-(z0+z2)$ $w2=z0-z2$ $w3=2z1-z4$
Etape 4	$l0=m0+m2$ $l1=m3+m2$ $l2=4(2k2+u1)$ $l3=4(2u2-k1)$	$l4=w0-6v0$ $l5=2(2v3+3z2)$ $l6=8(2z1-z0+z3)$
Etape 5	$y0= m0$ $y1=l4-4z2$ $y2=l0-6u3$ $y3=l4+l5$	$y4=l1-l2$ $y5=l6+w2$ $y6=m1+l3$ $y7=w1+4w3$

Tableau 2.3 : Algorithme rapide de la DTT

Tous les coefficients dans la DTT sont des nombres entiers, donc les opérations de multiplication se transforment en opérations d'addition et shift comme il est présenté dans le tableau suivant :

$m\times 2= =\ll m$
$m\times 3= =\ll m+m$
$m\times 4= =\ll \ll m$
$m\times 6= =\ll(\ll m+m)$
$m\times 8= =\ll \ll \ll m$

Tableau 2.4 Représentation des opérations shift

On peut réduire encore la complexité de l'algorithme DTT en utilisant la version zonale PDTT (pruned DTT). On considère que la majeure partie de l'énergie du bloc est compactée dans les premiers 4 × 4 coefficients basse fréquence. Si on prend par exemple L = 4, on obtient les formules (56) et (57) respectivement :

$$Y_{4\times4} = C_{4\times8}.X_{8\times8}.C^{t}_{8\times4} = D(T_{4\times8}.X_{8\times8}.C^{t}_{8\times4})D \qquad (56)$$

$$X_{8\times8} = C^{t}_{8\times4}.Y_{4\times4}.C_{4\times8} = T^{t}_{8\times4}(D.Y_{4\times4}.D)T_{4\times8} \quad (57)$$

la matrice diagonale est donné par:

$$D_{4\times4} = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{42}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{42}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{66}} \end{bmatrix} \qquad (58)$$

8- Conclusion :

L'étape de transformation et la pièce maitresse de la compression. Dans ce chapitre nous avons présenté, les principaux calcule de la DFT est ces dérivée DST et la DCT . Nous avons abordé l'approche de Loeffler (LLM DCT) et le Binary DCT (Bin DCT), la transformation de Karhunen-Loeve (KLT) est ces applications et l'algorithmes de codage basés sur des ondelettes (DWT) utilisé dans les normes JPEG2000 et SPIHT .

Enfin nous avons traité la transformation Tchebychef Discrète (DTT) et sa version zonale rapide (Pruned-DTT) qui nécessite respectivement -41% et -52% d'additions et d'opérations par rapport à Loeffler DCT et la DTT exacte. Son avantage réside dans sa consommation d'énergie faible tout en maintenant un PSNR et un SSIM élevé.

Le prochain chapitre sera consacré à l'intégration de la DTT sur le simulateur **SenseVid** est discuté les résultats de différents test .

Chapitre 3

Implémentation de la DTT et simulation

1- Introduction :

Dans ce chapitre on se charge d'implémenter le code du calcul de la DTT au SenseVid pour pouvoir évaluer la qualité de la vidéo reconstruite. Le PSNR (*Peak Signal to Noise Ratio*) et l'SSIM (*Structural SIMilarity*) sont utilisés en tant que métriques de qualité**.**

La DTT est une transformée orthogonale dérivée des polynômes discrets de Tchebichef (DTP) **[29].** Comme nous l'avons vue au deuxième chapitre, la faible complexité des algorithmes de la DTT par rapport à la DCT utilise un code simple et moins complexe. La DTT est de plus en plus utilisée dans la littérature en raison de son gain de codage élevé, de son efficacité de transformation et de sa faible complexité informatique.

Tous les coefficients dans la DTT sont des nombres entiers, donc les opérations de multiplication se transforment en opérations d'addition et shift comme il est présenté dans le tableau (2.4):

les opérations arithmétiques par lesquelles l'algorithme de la DTT **[29]** passe en le comparant à la DCT (CLA) donne une différence remarquable (voir le tableau (3.1)) .

	1D		2D	
	DTT exact	DCT CLA	DTT exact	DCT CLA
Addition	44	29	704	464
Multiplication	0	11	0	176
Shift	29	0	464	0

Tableau 3.1- Comparaison entre la DTT et la DCT CLA.

D'après ce tableau on remarque que la DTT n'utilise pas de multiplications réelles. Ce qui va nous permettre à réduire la consommation d'énergie.

2- Intégration de la DTT sur le simulateur SenseVid :

L'algorithme utilisé est divisé en quatre étapes d'opérations arithmétiques (voir le tableau 2.3).

2-1 Code de la transformation directe (1D DTT):

La première étape :

```
// dtt exact 1D

Mat_<short> bin1DTT(Mat_<short> block){

Mat_<short> dtt1D = Mat::zeros(1,8,CV_16S);

short
u0,u1,u2,u3,v0,v1,v2,v3,k0,k1,k2,z0,z1,z2,z3,z4,m0,m1,m2,m3,w0
,w1,w2,w3,l0,l1,l2,l3,l4,l5,l6,y0,y1,y2,y3,y4,y5,y6,y7,
tmp,tmp1,tmp2,tmp3,tmp4;
```

(block .at<short> (0.0) représente le premier vecteur au pixel)

```
u0 = block.at<short>(0,0) + block.at<short>(0,7);

u1 = block.at<short>(0,1) + block.at<short>(0,6);

u2 = block.at<short>(0,2) + block.at<short>(0,5);

u3 = block.at<short>(0,3) + block.at<short>(0,4);

v0 = block.at<short>(0,0) - block.at<short>(0,7);

v1 = block.at<short>(0,1) - block.at<short>(0,6);

v2 = block.at<short>(0,2) - block.at<short>(0,5);

v3 = block.at<short>(0,3) - block.at<short>(0,4);
```

la 2eme étape :

```
k0 = u0 + u2;     z0 = v0 + v3;

k1 = u1 + u3;     z1 = v1 - v2;

k2 = u1 - u3;     z2 = v1 + v2;

z3 = v1 - v3;
```

```
tmp = v3 <<1;

tmp1 = v2 - tmp + v3;

tmp2 = tmp1 <<1;

z4 = v2 - tmp + v3 + tmp2;
```

la 3eme étape :

```
tmp = u0 <<1;

tmp1 = u2 <<1;

tmp2 = u0 + tmp - tmp1;

m0 = k0 + k1;

m1 = k0 - k1;

m2 = tmp2 <<1;

m3 = k0 - k2;

tmp = z1 <<1;

w0 = -(z1 + z0);

w1 = -(z0 + z2);

w2 = z0 - z2;

w3 = tmp + z4;
```

la 4eme étape :

```
tmp = k2 <<1;

tmp1 = u2 <<1;

tmp2 = tmp + u1;

tmp3 = tmp1 - k1;
```

```
l0 = m0 + m2;

l1 = m3 + m2;

l2 = tmp2 <<2;

l3 = tmp3 <<2;

tmp = v0 <<1;

tmp1 = v3 <<1;

tmp2 = z2 <<1;

tmp3 = z1 <<1;

tmp4 = (v0 + tmp) <<1;

l4 = w0 - tmp4;

tmp = tmp1 + z2 + tmp2;

l5 = tmp <<1;

tmp = tmp3 + z0 - z3;

l6 = tmp <<3;
```

la 5eme étape :

```
tmp = u3 <<1;

tmp1 = z2 <<2;

tmp2 = w3 <<2;

y0 = m0;

y1 = l4 - tmp1;

y2 = l0 + u3 + tmp;

y3 = l4 + l5;
```

```
y4 = l1 - l2;

y5 = l6 + w2;

y6 = m1 + l3;

y7 = w1 + tmp2;

dtt1D.at<short>(0,0) = y0;

dtt1D.at<short>(0,1) = y1;

dtt1D.at<short>(0,2) = y2;

dtt1D.at<short>(0,3) = y3;

dtt1D.at<short>(0,4) = y4;

dtt1D.at<short>(0,5) = y5;

dtt1D.at<short>(0,6) = y6;

dtt1D.at<short>(0,7) = y7;

return (dtt1D/2);

}
```

2.2 Code de la transformation directe (2D DTT):

```
Mat_<short> bin2DTT(Mat_<short> block){

Mat_<short> tempBlock = Mat::zeros(8,8,CV_16S);

Mat_<short>  temp2Block = Mat::zeros(8,8,CV_16S);

Mat_<short>  tempRow = Mat::zeros(1,8,CV_16S);

for (int j = 0; j < block.cols; j++){

tempRow = bin1DTT( block.row(j) );

tempRow.copyTo(tempBlock.row(j));
```

```
}
transpose(tempBlock,tempBlock);
for (int j = 0; j < tempBlock.rows; j++){
tempRow = bin1DTT( tempBlock.row(j) );
tempRow.copyTo(temp2Block.row(j));
}
transpose(temp2Block,temp2Block);
return temp2Block;
}
```

2.3 Code de la transformation inverse (2D iDTT):

```
Mat_<short> bin2iDTT(Mat_<short> block){
Mat_<short> tempBlock = Mat::zeros(8,8,CV_16S);
Mat_<short>  temp2Block =Mat::zeros(8,8,CV_16S);
Mat_<short>  tempRow =Mat::zeros(1,8,CV_16S);
for (int j = 0; j < block.cols; j++){
tempRow = bin1DTT( block.row(j) );
tempRow.copyTo(tempBlock.row(j));
}
transpose(tempBlock,tempBlock);
for (int j = 0; j < tempBlock.cols; j++){
tempRow = bin1DTT( tempBlock.col(j) );
tempRow.copyTo(temp2Block.row(j));
```

```
}

transpose(temp2Block,temp2Block);

return temp2Block;

}
```

Pour bien mener ce travail nous avons modifié plusieurs fichiers sur le simulateur. Dans ce qui suit nous allons expliquer les différentes étapes de modifications.

-La modification de *mainfarme.cc*

On a ajouté la matrice DTT et la matrix D comme suit :

```
if(codecP.DCT == "DTT"){

Mat D = (Mat_<int>(8,8)
<<27,19,17,27,41,68,87,104,20,20,24,32,44,99,102,94,24,22,27,
41,68,97,117,95,24,29,37,49,87,148,136,106,31,37,63,95,116,18
6,175,131,41,60,94,109,138,177,192,157,83,109,133,148,175,206
,204,172,123, 157,162,167,191,170,175,169);

dctBlock = DTT(floatBlock, codecP.DCT, codecP.zoneSize);

dctBlock.convertTo(dctFloatBlock, CV_32F);

divide(dctBlock,D, quantifBlock1,1 , CV_32F);

divide(quantifBlock1,
getQuantisationMatrix(codecP.qualityCoef), quantifBlock,
1,CV_16S);

}
```

On déclare le nombre des operations del' addition et de décalages :

```
int CLA_ADD_NB_DTT = 704;

int shiftBIN_DTT = 464;
```

-Calcule de l'energie :

```
double energy;

if(codecP.DCT == "DTT"){

dttCycles = CLA_ADD_NB_DTT * CYCLES_PER_FADD + shiftBIN_DTT *
CYCLES_PER_SHIFT;

quantifCyclesDtt = 128*CYCLES_PER_FMUL;

if(codecP.DCT == "DTT")

energy = (quantifCyclesDtt + dttCycles)* frameBlocksNb/1000*
POWER / PROC_CLOCK;
```

La modification de *sensvid.cc*

On ajoute les paramètres de simulation de la DTT comme suit:

```
case'a':

codecP.DCT = optarg;

if (codecP.DCT!="DTT"&& codecP.DCT!="sBIN"&&
codecP.DCT!="tBIN") {cout <<"NOT RECOGNIZED DTT"<< codecP.DCT
<<endl;     exit(EXIT_FAILURE);

}
```

-La modification de sensevid.h

On déclare les fonctions de la DTT dans le sensvid

```
//DTT functions

Mat DTT(Mat block, string DCT, int zoneSize);

Mat iDTT(Mat  dctBlock, string DCT, int zoneSize);

Mat_<short> tbin2DTT(Mat_<short> block, uchar zoneSize);

Mat_<short> sbin2DTT(Mat_<short> block, uchar zoneSize);

Mat_<short> bin1DTT(Mat_<short> block);
```

```
Mat_<short> bin1iDTT(Mat_<short> block);

Mat_<short> bin2DTT(Mat_<short> block);
```

Mat_<short> bin2iDTT(Mat_<short> block);

3- Comparaison des résultats:

3-1 Mesure énergétique :

Le but essentiel de notre thèse c'est de réduire la consommation de l'énergie de l'encodage en utilisant une transformation de faible complexité. Pour effectuer la simulation nous avons utilisé deux vidéos test. Selon le mouvement du vidéo (flowers est plus dynamique par rapport à hall vidéo). Le tableau (3.3) résume les résultats de capture :

ENERGY Flowers vid			
frame	CLA	DTT	Tllm
1	43,403	1,229	33,899
60	43,401	1,234	33,919
120	43,403	1,245	33,939
180	43,407	1,254	33,945
240	43,427	1,299	34,003
250	43,425	1,3	34,005

Tableau 3.2 : Energie de codage "flowers "

Dans ce qui suit on va présenter les résultats des simulations par figure :

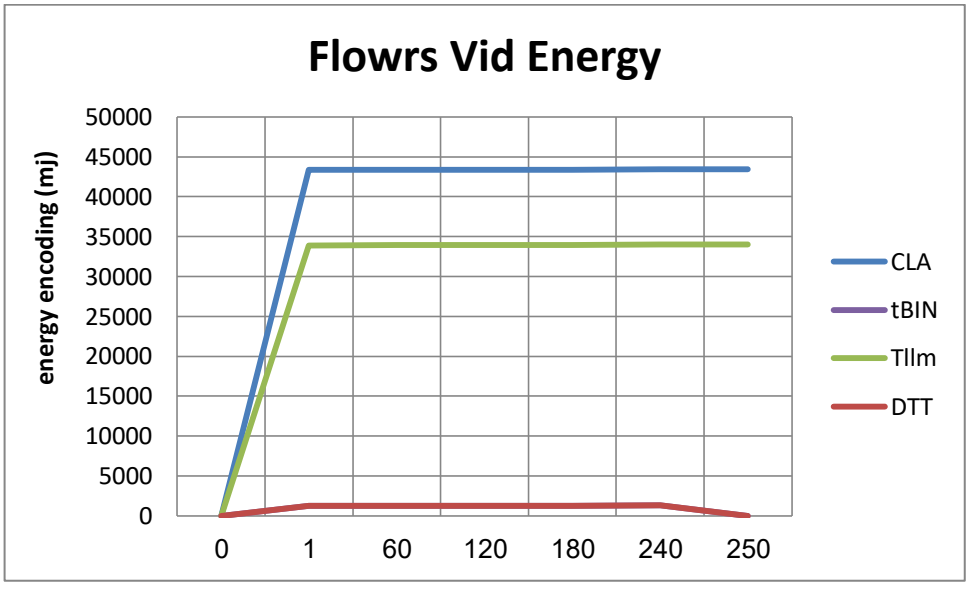


Figure 3.1 : énergie de codage "flowers "

ENERGY HALL vid (moin dynamique)			
Frame	CLA	DTT	Tllm
1	43,345	1,212	33,926
60	43,351	1,219	33,936
120	43,354	1,225	33,946
180	43,352	1,221	33,937
240	43,350	1,219	33,934
250	43,350	1,219	33,934

Tableau 3.3 : énergie de codage "Hall vidéo"

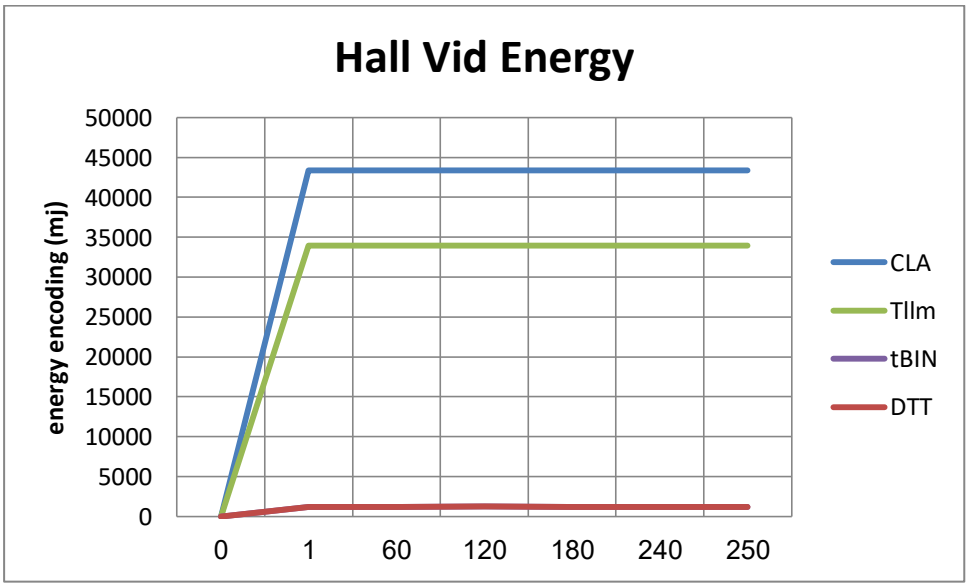


Figure 3.2 : énergie de codage "Hall vidéo"

D'après ces courbe on remarque que l''énergie nécessaire à l'exécution de la DTT est clairement inferieure aux autres DCTs quelle que soit le type de vidéo.

3-2 Mesure PSNR :

Dans ce qui suit on va présenter les résultats des comparaisons en termes de PSNR.

PSNR Flowers vid			
Frame	CLA	DTT	Tllm
1	29,828	28,424	28,604
60	30,695	29,353	29,563
120	31,241	30,199	30,425
180	31,115	29,901	30,151
240	31,623	30,458	30,751
250	31,432	30,413	30,705

Tableau 3.4 : PSNR flowers vidéo.

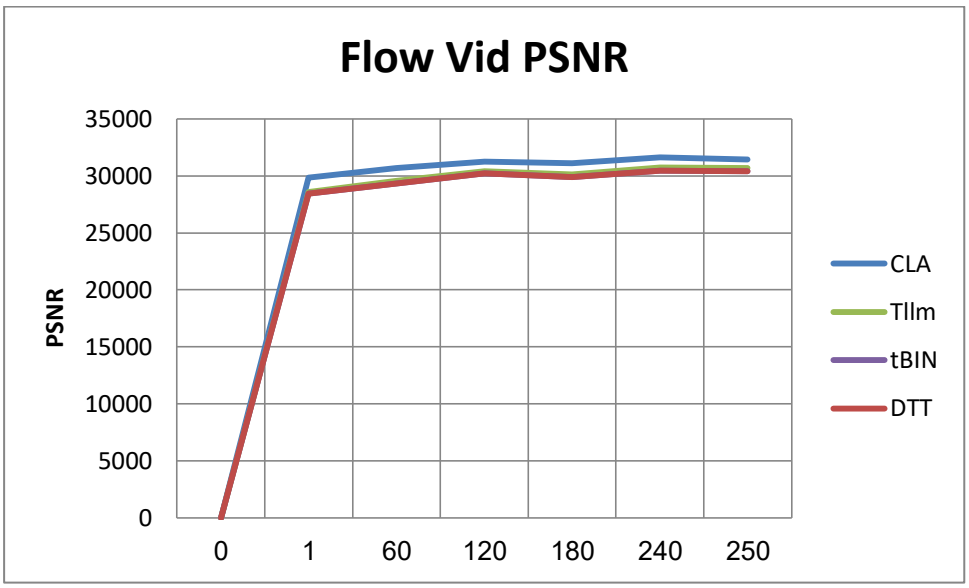


Figure 3.3 : PSNR flowers vidéo

PSNR Hall vid			
Frame	CLA	DTT	Tllm
1	34,158	35,550	36,040
60	33,939	35,309	35,838
120	33,707	35,021	35,483
180	33,907	35,230	35,718
240	33,929	35,264	35,773
250	33,919	35,330	35,813

Tableau 3.5 : PSNR Hall vidéo.

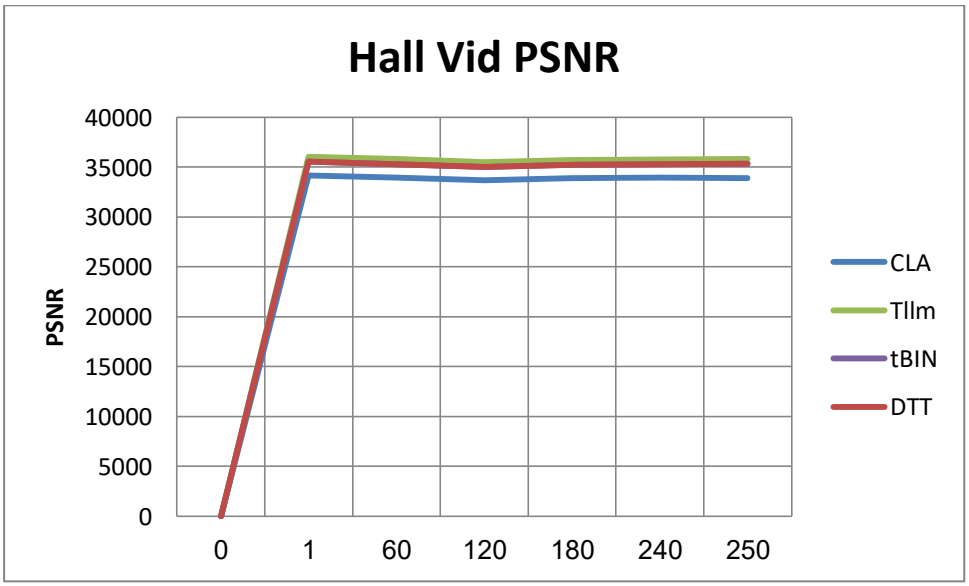


Figure 3.4 : PSNR Hall vidéo

3-3 Mesure SSIM :

Dans ce qui suit on va présenter les résultats des comparaisons en termes de PSNR.

SSIM Flowers vid			
Frame	CLA	DTT	Tllm
1	0.971	0.966	0.969
60	0.973	0.97	0.972
120	0.971	0.966	0.969
180	0.97	0.971	0.974
240	0.969	0.969	0.972
250	0.971	0.966	0.969

Tableau 3.6 : SSIM flowers vidéo.

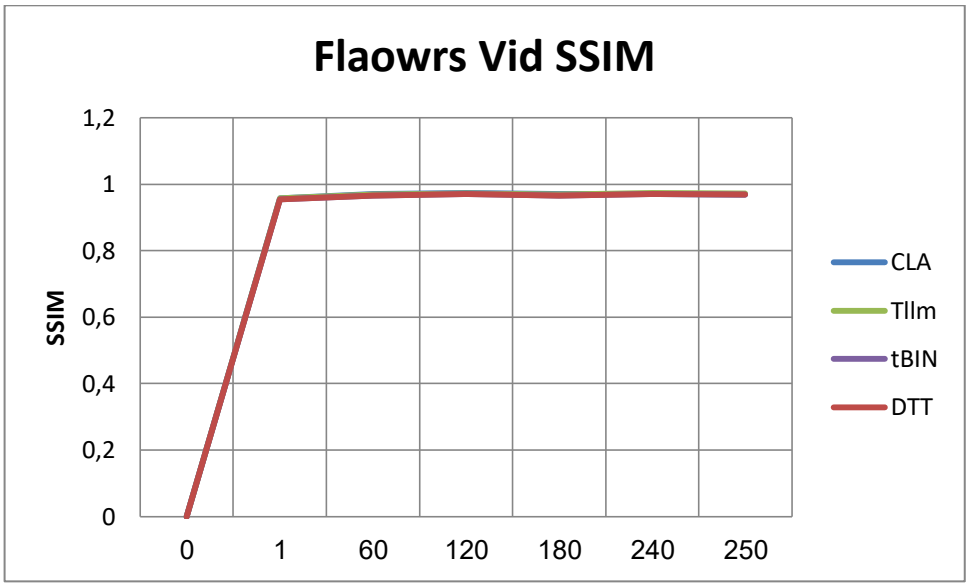


Figure 3.5 : SSIM flowers vidéo

PSNR Hall vid			
Frame	CLA	DTT	Tllm
1	0.952	0.977	0.983
60	0.951	0.976	0.982
120	0.95	0.975	0.981
180	0.952	0.977	0.982
240	0.952	0.976	0.983
250	0.95	0.976	0.982

Tableau 3.7 : SSIM Hall vidéo.

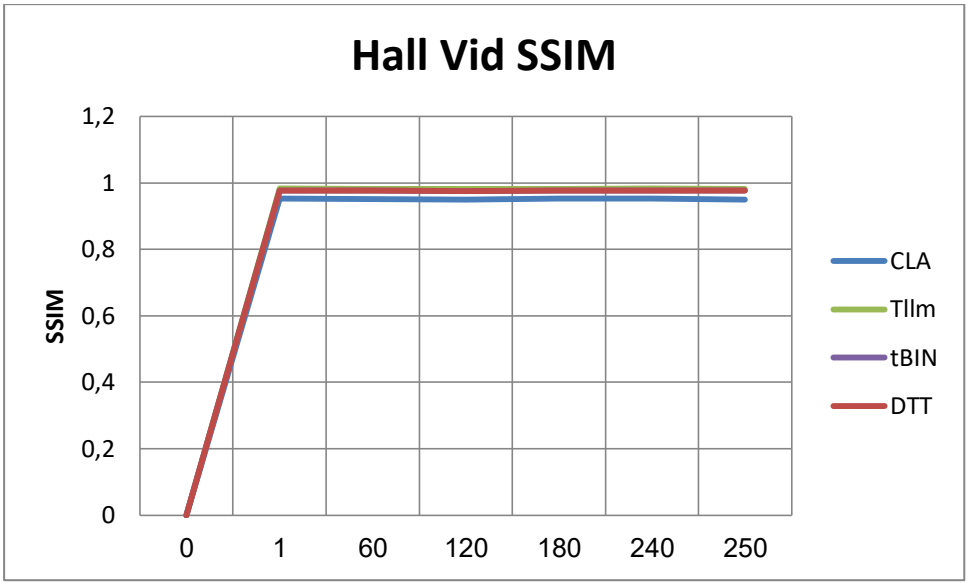


Figure 3.6 : SSIM Hall vidéo

Comme nous l'avons aperçu, l'algorithme DTT garde presque la même valeur du PSNR et SSIM ce qui donne avantage à la qualité vidéo à capturer.

figure 3.7 : Frame 75 capture avec la DTT

Figure 3.8 : Frame 75 capture avec la CLA

Figure 3.9 : Frame 75 capture avec la tLLM

Les figures précédentes montrent une comparaison visuelle des images reconstruites par différentes transformations. Il est clair qu'il n'y a pas une grande différence de qualité entre ces images reconstruites.

4- Discussion des résultats :

D'après ce que nous avons vu précédemment, les différentes méthodes de transformation (DTT,DCT et tLLM) ont donné des résultats identiques de point de vue visuel. Toutes les images capturées par le simulateur SenseVid semblent similaires.

Le résultat visuel malgré les informations qu'il donne pour l'œil humain, il reste insuffisant pour l'utiliser dans la comparaison entre les différentes méthodes. Cette insuffisance est due au manque de précision et les limites de distinction de l'oeil. C'est pour cela nous utilisons les métriques d'évaluation objectives (PSNR, SSIM).

La métrique SSIM, utilisée pour voir à quel point les images compressées sont identiques à l'image originale (1 est la valeur max). En appliquant cette métrique aux images reconstruites nous trouvons que toutes les valeurs sont proches de 1. En ce qui concerne le PSNR, les résultats de la méthode implémentée sont supérieurs et proche du 40 dB, si le PSNR est élevé la SSIM est proche de 1.

En ce qui concerne la consommation énergétique, la DTT est la meilleure candidate.

5- Conclusion :

Dans ce chapitre, nous avons implémenté la DTT exact. Une comparaison a été faite avec la DCT exact représentée par la CLA ainsi d'autre type de la DCT rapide tLLM et tBIN qui sont déjà prédisposées dans le simulateur SenseVid. Théoriquement parlant, la DTT exact propose un algorithme léger est moins complexe par rapport a la DCT exacte. En implémentant la DTT nous avons vérifié la rapidité et la faible consommation énergétique.

Pour conclure, nous pouvons dire que la DTT est fortement recommandée pour une implémentation dans les systèmes embarquée destinée à l'IoT. Cette dernière nécessite des algorithmes rapide à faible consommation énergétique pour satisfaire les contraintes des équipements miniaturisés (CPU, RAM., Batterie) d'un côté et pour augmenter l'autonomie de l'application d'un autre côté.

Conclusion générale

Le développement des logiciels de simulation est un domaine de recherche très vaste et créative. SenseVid est l'un de ces simulateur en open source permettent la possibilité de le modifier et le mettre à jour. Il permet aux étudiants et chercheurs en traitement d'images et vidéos à faire une évaluation basé sur les traces de trafic destiné au WVSN.

Dans ce document nous nous sommes intéressés à l'algorithme de la DTT qui est beaucoup moins complexe que celui de la DCT. Il utilise que des nombres entiers, donc les opérations de calculs seront plus simples et font accélérer l'exécution de la transformée. La simplicité de la DTT implique des opérations d'addition et de shift entières, contrairement à la DCT qui contient des opérations d'addition, shift et de multiplication flottantes (réels). Cette dernière et très gourmande en matière de ressources, c'est pour cette raison que la DTT a été choisie dans ce travail pour être appliquée dans les réseaux WVSN.

Comme tous les logiciels, on a constaté quelque erreurs pendant la simulation lié a la zone de seize (ρ) l'orque il s'agit une DCT Rapide en carrée (sBIN , sLLM) , une erreur de résultat lié au facteur de qualité 90 , par conséquent ces erreurs n'ont pas affecté à notre simulation en choisissant la CLA (DCT exacte) , tBIN et tLLM (PDCT) avec deux facteur de qualité 30 & 60 .

En conclusion, et pour répondre à notre problématique: l'utilisation de la DTT en compression d'images/vidéos diminue clairement la consommation d'énergie en préservent la même qualité du vidéos.

Références & Bibliographe :

- [1] Kouadria N, thèse de doctorat UBMA (2013), codage et compression d'image pour des réseaux capteur sans fil (p 9).

- [2] Erhad G, Juris H (2003). Computer performance evaluation modelling technique and tools (p 256).

- [3] E.orellama ,R.lecuria (2011) .A simulation framework for image quality assessment in wireless visual sensor networks under packet loss conditions, Computer Science Society (SCCC), 2011 30th International Conference of the Chilean.

- [4] Moufida M , (2018) , SenseVid . A traffic trace based tool for QoE Video transmission assessment dedicated to Wireless Video Sensor Networks.

- [5] AHMED Seghir Z , thèse de doctorat UMC (2012) , Evaluation de la qualité d'image (p.42,43,44,45).

- [6] MECHOUEK K , , thèse de doctorat UBMA (2017), Compression faible complexité et codage par régions d'intérêt d'images fixes dans les WSNs. (p17).

- [7] A. Mammeri, B. Hadjou, A. Khoumsi, (2012) A survey of image compression algorithms for visual sensor networks, ISRN Sensor Networks .

- [8] C. Loeffler, A. Ligtenberg, G.S. Moschytz, Practical fast 1-d dct algorithms with 11 multiplications, Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on, (1989), pp. 988–991vol.2.

- [9] Jie L, Trac D. Tran,Fast Multiplierless Approximations of the DCT Withthe Lifting Scheme.

- [10] Dr. Bovik A . The Essential Guide to Video Processing . (page 272).

- [11] Huffman, D. A. (1952). A method for the construction of minimum redundancy codes. Proceedings of the IRE, 40(9), 1098-1101.

- [12] Golomb, SW. (1966). Run-length encodings. IEEE Transactions on Information Theory, 12, 399-401.

- [13] Mean opinion score (MOS) terminology, Methods for objective and subjective assessment of speech and video quality. Recommendation ITU-T P.800.1.

- [14] Humberto Ochoa-Domınguez K. R. Rao , (2019 by Taylor & Francis Group), Discrete Cosine Transform.

- [15] V. Britanak ,P. C.Yip , K. R. Rao , (February 2006) , Discrete Cosine Transform .

- [16] G. BAUDOIN et J.-F. BERCHER , (Novembre 2001) , TRANSFORMÉE DE FOURIER DISCRÈTE , École Supérieure d'Ingénieurs en Électrotechnique et Électronique.

- [17] N. ROCKMORE , (2003) , The Cooley–Tukey FFT and Group Theory.

- [18] A. Bostan , (2010) , Algorithmes rapides pour les polynômes, séries formelles et matrices.

- [19] Bendelhoum M ,(these de doctorat , decembre 2018) , Évaluation des performances des images compressées par l'algorithme TOD-SPIHT, et transmises via le système MC-CDMA.

- [20] Ferda E, Nur Azman A, Nanna S (2011) , 1Faculty of Information and Communication Technology Universitas Dian Nuswantoro (UDINUS) Semarang, Indonesia , SPECTRUM ANALYSIS OF SPEECH RECOGNITION VIA DISCRETE TCHEBICHEF TRANSFORM.

- [21] Kouadria, N., Doghmane, N., Messadeg, D., & Harize, S. (2013). Low complexity DCT for image compression in wireless visual sensor networks. *Electronics Letters*, *49*(24), 1531-1532.

- [22] Présentation générale de l'Internet des objets (juin 2012) Y.2060 secteur de la normalisation des télécommunications de l' UIT.

- [23] Heyne, B., & Götze, J. (2007). A low-power and high-quality implementation of the discrete cosine transformation. Advances in Radio Science.

- [24] Liang, J., & Tran, T. D. (2001). Fast multiplierless approximations of the DCT with the lifting scheme. Signal Processing, IEEE Transactions on, 49(12), 3032-3044.

- [25] Jean-Paul DUBUS , technique de l'ingénieur, Compression d'images statiques et dynamiques.

- [26] B.Mohamed . 2013 , thèse de magister UBMA , Compression des Images en Couleurs Fixes en Utilisant la DWT.

- [27] Prattipati, S., Iswar, S., Swamy, M.N.S. and Meher, P.K. (2013) "A Fast 8 × 8 Integer Tchebichef Transform and Comparison with Integer Cosine Transform", en IEEE International Midwest Conferences on Circuits and Systems, Colubus, 4- 7 August 2013, 1294-1297.

- [28] N. Kouadria1, I. Mansri1, S. Harize1, N. Doghmane1, K. Mechouek1 , Lossy compression of color images based on discrete Tchebichef transform, Lossy compression of color images based on discrete Tchebichef transform.

- [29] N, Kouadria, K, Mechouek, K., S, Harize, and N, Doghmane, " Regionof-interest (2019) ,based image compression using the discrete Tchebichef transform in wireless visual sensor networks". Computers & Electrical Engineering, Elsevier, 73, pp.194-208.

- [30] Paulo A. M. Oliveira, Renato J. Cintra , 2016 , Low-complexity Image and Video Coding Based on an Approximate Discrete Tchebichef Transform , Sunera Kulasekera and Arjuna Madanayake, Member, IEEE.

- [31] Paulo A. M. Oliveira ,Renato J. Cintra∗ F´abio M. Bayer†,Sunera Kulasekera‡ Arjuna Madanayake , 28 janvier 2015 , A Discrete Tchebichef Transform Approximation for Image and Video Coding.