République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

Université SAAD DAHLEB-BLIDA1

Faculté des Sciences

Département d'informatique

Mémoire de Fin d'Etude En vue de l'obtention du diplôme de Master en Informatique

Option : Système informatique et réseaux

Thème

Compression d'images dans les réseaux de capteurs sans fil

Réalisé par : - Benzada Narimane
- Bouhamidi Amira

Devant les jurys :

Mr.Ould Khaoua Mohamed	USDB1	Président
Mme.Boutoumi Bachira	USDB1	Examinatrice
Mme.Midoun Khadidja	USDB1	promotrice
Mme. Goumiri Soumia	CERIST	Encadreur

2021/2022

Remerciements

Tout d’abord nous tenons à remercier Allah pour nous avoir donné le courage, la force et la volonté pour réussir et de nous avoir éclaircir le chemin tout au long de notre vie.

Nos remerciements et nos profondes gratitudes vont à notre promotrice Madame **Midoun Khadidja** et notre encadreur Madame **Goumiri Soumia** pour leurs encadrements, leurs suivies et leurs conseils tout au long de cette période.

Nous tenons aussi à remercier les membres du jury pour leur précieux temps accordé à l’étude de notre mémoire.

Nos remerciements et notre gratitude vont aux professeurs et enseignants de département d'informatique ainsi que ses étudiants et son personnel côtoyés tout au long de notre cursus universitaire.

Que toute personne ayant œuvré de près ou de loin à la réalisation de ce projet par une quelconque forme de contribution, trouve ici le témoignage de notre plus profonde reconnaissance.

Dédicace

Nous dédions ce mémoire : À nos très chers parents pour leurs soutiens durant toute notre vie d'étudiants et sans eux nous ne serions jamais

Devenues ce que nous sommes.

A nos frères et sœurs

À toutes nos familles.

À tous nos amis sans aucune exception.

Benzada Narimane

Bouhamidi Amira

Résumé

Les réseaux de capteurs sans fil sont des systèmes distribués spécialement, composé de plusieurs dizaines de milliers de micro-capteurs. Ils sont aujourd'hui utilisés dans plusieurs domaines. L'image à une importance dans plusieurs applications basant sur les réseaux de capteurs sans fil, mais son traitement et sa transmission posent des problèmes car les images sont codées sur des milliers d'octets donc les transmettre sur un réseau de capteur ou ces nœuds sont alimenté par batteries consomme beaucoup d'énergie.

Pour minimiser l'énergie consommée, on s'est intéresse à la compression d'images pour réduire la quantité de données qui représente une image et donc augmenter la durée de vie du réseau. Nous avons travaillé sur l'algorithme de compression JPEG basé sur l'encodeur RLE, avec un modèle énergétique pour modéliser l'énergie consommée et pour évaluer tous les résultats, nous avons testé un autre algorithme JPEG mais qui est basé sur l'encodeur Huffman, et à la fin nous avons comparé les résultats des deux algorithmes selon trois paramètres (l'énergie consommée, la qualité des images et le taux de compression).

Mots clés : réseaux de capteurs sans fil, réseaux de capteurs d'images, compression d'images, conservation d'énergie, JPEG, transformation DCT, quantification, codage.

ملخص

شبكات الاستشعار اللاسلكية هي أنظمة موزعة بشكل خاص ، تتكون من عدة عشرات الآلاف من أجهزة الاستشعار الدقيقة. يتم استخدامها اليوم في عدة مجالات. الصورة مهمة في العديد من التطبيقات القائمة على شبكات الاستشعار اللاسلكية ، لكن معالجتها ونقلها يطرحان مشاكل لأن الصور يتم ترميزها على آلاف البايتات ، لذا فإن نقلها على شبكة مستشعر حيث يتم تشغيل هذه العقد بواسطة البطاريات يستهلك الكثير من الطاقة.

لتقليل الطاقة المستهلكة، نحن مهتمون بضغط الصور لتقليل كمية البيانات التي تمثل الصورة وبالتالي زيادة عمر الشبكة. لقد عملنا على خوارزمية ضغط JPEG بناءً على تشفير RLE ، مع نموذج طاقة لنمذجة الطاقة المستهلكة ولتقييم جميع النتائج ، قمنا باختبار خوارزميةJPEG أخرى ولكنها تعتمد على تشفير Huffman ، وفي النهاية قمنا بمقارنة نتائج الخوارزميتين وفق ثلاث معاملات (الطاقة المستهلكة ، جودة الصور ومعدل الضغط).

الكلمات المفتاحية : شبكات الاستشعار اللاسلكية، شبكات حساسات الصور، ضغط الصور،الحفاظ على الطاقة، تحويل DCT ، التكميم، الترميز.

Abstract

Wireless sensor networks are specially distributed systems, composed of several tens of thousands of micro-sensors. They are used today in several fields. The image is important in several applications based on wireless sensor networks, but its processing and transmission pose problems because the images are coded on thousands of bytes so transmitting them on a sensor network where these nodes are powered by batteries consumes a lot of energy.

To minimize the energy consumed, we are interested in image compression to reduce the amount of data that represents an image and therefore increase the lifetime of the network. We worked on the JPEG compression algorithm based on the RLE encoder, with an energy model to model the energy consumed and to evaluate all the results, we tested another JPEG algorithm but which is based on the Huffman encoder, and at the end we compared the results of the two algorithms according to three parameters (the energy consumed, the quality of the images and the compression rate).

Keywords: wireless sensor networks, image sensor networks, image compression, energy conservation, JPEG, DCT transformation, quantization, coding.

Présentation de l'organisme d'accueil

L'organisme d'accueil est le CERIST : signifiant Centre de Recherche sur l'Information Scientifique et Technique fondé en 1985 sous la tutelle du premier ministre de l'époque (MUSTAPHA KARIM RAHIEL), avait pour mission principale de mener toute recherche relative à la création, à la mise en place et au développement d'un système national d'information scientifique et technique.

- De 1994 jusqu'à 2000 : CERIST fut le premier fournisseur d'accès grand public de la toile en Algérie.
- Il devient ensuite, un réseau dédié à la communauté universitaire et de recherche à travers le réseau académique algérien de recherche (ARN).
- Il est le gestionnaire du nom de domaine.dz.

Le CERIST est chargé de la réalisation des programmes de recherche scientifique et de développement technologique dans le domaine de l'information scientifique et technique. A ce titre, il est notamment chargé de :

- Mener toute activité de recherche relative à la création, la mise en place et le développement du système national d'information scientifique et technique.
- Promouvoir la recherche dans les domaines des sciences et des technologies de l'informationet de la communication et de participer à leur développement.
- Contribuer à la coordination et à la mise en œuvre des programmes nationaux d'information scientifique et technique.
- Contribuer à l'édification et à la promotion de la société de l'information par la mise en placeet le développement de réseaux sectoriels d'information thématiques.
- Participer à la modernisation du système documentaire universitaire national par la mise en place notamment de bibliothèques virtuelles.
- Réunir les éléments nécessaires à la constitution de bases de données nationales dans les domaines des sciences et de la technologie et en assurer la diffusion.
- Promouvoir la recherche en matière de sécurité de l'information et des réseaux.

Le CERIST est donc le premier pilier en Algérie qui permet de soutenir les structures pesantes du développement et de l'information, de l'optimisation et du calcul intensif, de l'hébergement et du traitement des données.

Sommaire

Liste des figures

Liste des Tables

Liste des abréviations

RCSF : Réseau de Capteurs sans fil.

RCSFI : Réseau de Capteurs sans fil d'images.

JPEG : Joint Photographic Expert Group.

RLE : Run- Length- Encoded**.**

DCT : Discrete Cosine Transform**.**

DWT : Discrete Wavelet Transform.

Introduction Générale

Introduction générale

Les récentes avancées dans le domaine des technologies de communication sans fil et microélectroniques ont permis le développement de très petits capteurs à faible coût et consommant peu d'énergie, grâce à ce développement, un nouveau domaine de recherche a été créé qui est le domaine des réseaux de capteurs sans fil, qui traite, collecte et transmettre des données d'un environnement vers le monde extérieur [2].

Les réseaux de capteurs sans fil sont des systèmes embarqués composé d'un ensemble d'unités de traitement embarqués appelé « motes » sont alimentés par piles et qui communiques à travers des liens sans fil. Ces capteurs doivent être solides et doivent aussi survivre dans des conditions difficiles [1].

En raison de leur flexibilité, leur facilité de déploiement et leur faible cout, les réseaux de capteurs sans fil promettent de révolutionner notre vie à travers plusieurs domaines d'applications tels que la surveillance et la reconnaissance d'objets, la localisation et pistage d'objets, le contrôle de l'environnement et la cartographie de la biodiversité, l'agriculture de précision, la médecine et la santé, le bâtiment intelligent. Cependant la représentation des données visuelles exige une grande quantité d'information ce qui demande une énergie très élevée pour les transmettre. Donc il faut compresser les images avant la transmission pour réduire l'énergie consommée.

La compression d'images c'est l'une des méthodes les plus utilisées pour minimiser la consommation d'énergie dans les réseaux de capteurs sans fil. Plusieurs travaux de recherche qui sont publiés et qui gravitent autour d'un objectif commun et concentre sur la transmission des images avec des algorithmes de compression. Il est couramment admis que l'émetteur radio est l'un des composants les plus gourmands en énergie puisque la compression d'image demande des traitements et des transformations au niveau ce nœud donc le cout de traitement est pris en considération à cause de la grande quantité d'informations.

Par contre le traitement d'une simple valeur comme la température, la luminosité par exemple que sa cout est très petit et qui n'est pas pris en considération.

Notre objectif est de trouver un algorithme de compression adaptée aux contraintes des réseaux de capteurs sans fil (faible coût calculatoire, faible coût de stockage).

Ce mémoire est organisé en quatre chapitres comme suit :

Dans le premier chapitre, nous avons présenté les différents concepts liés aux réseaux de capteurs sans fil et les différentes techniques utilisées pour la conservation d'énergies.

Le deuxième chapitre, on a parlé d'une des méthodes de conservation d'énergie, ainsi que les travaux développés pour minimiser l'énergie lors de la transmission des images.

Dans le troisième chapitre on a cité Les différentes étapes de l'algorithme de compression JPEG dans laquelle notre conception a été basée, sont dans le troisième chapitre.

Dans le quatrième chapitre, on a abordé les différents tests pratiques réalisés.

Chapitre 1 : Introduction aux Réseaux de Capteurs sans fil

1.1 Introduction

Les réseaux de capteurs sans fil (Wireless Sensor Network (WSN) en anglais) constituent une catégorie de réseau sans fil comportant un très grand nombre de nœuds, ils sont considérés comme un type spécial de réseaux ad hoc.

Les nœuds des RCSF sont construits à partir des composants pas chers, et consistent en un grand nombre de micro-capteurs capable de récolter et de transmettre des données environnementales de manière autonome vers le monde extérieur.

Les nœuds capteurs sont utilisés dans plusieurs domaines d'applications : les opérations militaires et de sécurité, la surveillance environnementales, la médecine….

En vue de la batterie limitée des capteurs, la transmission de données ne peut pas être assurée directement entre tout couple de capteur. Les données sont donc transmises de proche en proche.

1.2 Les Réseaux de capteurs sans fil

Un réseau de capteur sans fil (RCSF) est un système de mesures. Il est composé d'un grand nombre des nœuds qui sont des micro-capteurs, capables de mesurer une ou plusieurs gradeurs physiques, de stocker, de traiter et de transmettre les données d'une manière autonome. Suivant des principes similaires aux réseaux mobiles ad hoc, la répartition des nœuds dans le champ de captage n'est pas nécessairement prédéfinie. Les nœuds capteurs vont découvrir leurs voisins et s'auto-organiser pour former un réseau de capteurs sans fil et remonter les données vers l'utilisateur final, via un réseau à infrastructure sur la base d'un modèle de communication multi-sauts.

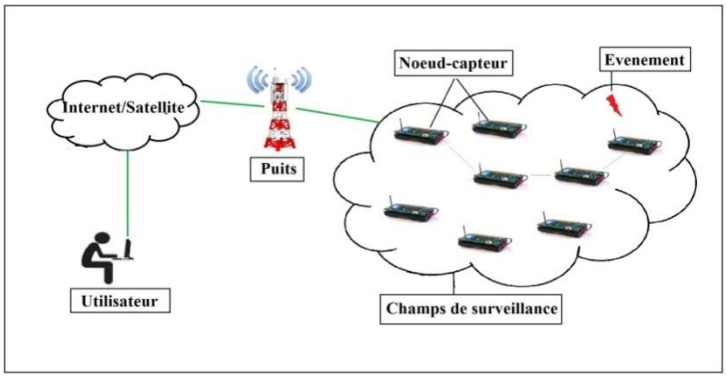


Figure 1.1 : Architecture de communication d'un réseau de capteurs sans fil [1]

Le réseau de capteur s'adapte aux changements d'état des nœuds, il est capable de fonctionner sans aucune intervention humaine avec une grande tolérance aux défaillances. Les nœuds-capteurs ont au moins deux rôles important : l'acquisition des données et la transmission de proche en proche pour l'acheminement des paquets via la destination. Le point de collecte appelé puits: c'est une interface entre le réseau de capteur sans fil et l'utilisateur final. L'interconnexion avec ce point se fait par un réseau à infrastructure comme l'internet liaison satellite. Cette architecture est illustrée dans la figure (1.1).

Les caractéristiques les plus importantes des nœuds capteurs sont leurs capacités d'auto-organisation de coopération, leur rapidité de déploiement, la tolérance aux erreurs et le faible coût. Un nœud de capteur est un dispositif de taille réduite constitué de plusieurs matériels [1]:

— **Unité de captage:** Dispositif matériel dont le rôle est de transformer un phénomène physique en une énergie électrique. Il existe plusieurs catégories de senseurs selon la nature de l'opération. Ainsi en peut les classer en des senseurs actifs ou passifs.

— **Unité de traitement :** contient c'est deux composants :

- **Mémoire :** La mémoire sauvegarde les données proviennent de senseur et les données proviennent à partir des autres nœuds capteurs.
- **Processeur:** Le processeur est le composant le plus important dans l'architecteur de nœud capteur car: il reçoit les données auprès de senseur, traiter ces données et décide quand et où l'envoyer. Il assure aussi le relais de données reçus à partir d'autres nœuds de capteurs. Le système d'exploitation utilisé dans les capteurs est TinyOS.

— **Unité d'alimentation :** Le bloc d'alimentation est important dans la phase de l'acquisition puisqu'il garde le fonctionnement de capteur au sein de réseau pour une durée limitée. La durée de vie d'un réseau de capteur dépend de niveau d'énergie de l'ensemble des nœuds, et comme ces nœuds sont utilisées pour captées des phénomènes physiques dans des champs de captage généralement difficiles, donc la phase de l'alimentation est aussi indispensable, et dès que la source d'énergie est la batterie traditionnelle, il faut surveiller l'état d'énergie dans le processus de communications entre ses entités. Les batteries doivent être changées de façon périodique.

— **Unité de Communication :** L'unité de communication est le composant responsable sur l'émission et la réception des paquets entres les nœuds sur un médium sans fil, de type radio fréquence. Le nœud capteur en principe est dédié pour accomplir une seul tâche, en effet sa capacité de calcule, de mémoire et de communication est limitée. Cette limitation est due à plusieurs explications: La première est liée à la contrainte de coût de fabrication de nœud capteur. La deuxième est liée à la contrainte d'énergie, et la dernière est liée à la contrainte de poids et de taille de nœud capteur [2].

La conception et la réalisation des réseaux de capteurs sans fil sont influencées par plusieurs paramètres [3], dont nous citons:

- **Durée de vie du réseau:** Cette métrique égale à la durée entre le déploiement du réseau jusqu'à l'instant où l'énergie du premier nœud s'épuise. La durée de vie du réseau varie de quelques heures à plusieurs années, selon l'application.
- **Ressources limités:** Les nœuds capteurs sont limités en terme de capacité de traitement et de mémoire. L'objectif est de mettre en œuvre des capteurs simples, petits et peu coûteux.
- **Bande passante limité :** Pour réduire l'énergie consommée lors de transfert de données entre les nœuds, le débit utilisé est de quelques dizaines de Kb/s. Un faible taux de transmission n'est pas un obstacle, puisque les fréquences de transmission ne sont pas importantes.
- **Facteur d'échelle :** Le nombre des nœuds publiés pour une application peut atteindre des milliers, dans ce cas le réseau fonctionne avec des densités de capteurs très grandes, et ce qui génère beaucoup de transferts internes, et nécessite que la station de base soit équipée de suffisamment de mémoire, pour stocker les informations reçues.
- **Topologie dynamique :** La topologie de réseau de capteurs elle peut changer à la cour du temps. À cause de l'expiration de l'énergie ou dans le cas de la défaillance d'un nœud capteur, soit en raison de déploiement de nœuds capteurs dans des environnements hostiles par exemple.
- **Agrégation de donnée :** Les données produites par les nœuds capteurs voisions sont très corrélées spatialement et temporellement, cela peut conduire à la réception d'une base d'informations redondante. La réduction de ces informations redondantes permet de réduire la consommation d'énergie dans le réseau, et donc d'augmenter sa durée de vie. L'un des techniques utilisées pour réduire la transmission d'informations redondantes est : l'agrégation de données, dans cette technique les nœuds intermédiaires collectent les informations prévenant de plusieurs sources.

1.3 Les Protocoles de Routage des RCSFs :

Les protocoles de routage établissent des routes entre tout nœud du réseau et la station de base pour assurer la fidélité de routage. Les protocoles sont classés en trois catégories : Protocoles de routage plat, protocoles de routage hiérarchique et protocoles de routage basé sur localisation.

1.3.1 Protocoles de Routage Plat :

Dans les réseaux plats tous les nœuds ont le même rôle. En effet le nombre des nœuds est très grand, il n'est pas possible d'attribuer un identifiant pour chaque nœud donc cette spécification conduit à un routage centré. La station de base envoie les requêtes aux nœuds qui situent dans des régions spécifiques, et puis quelques données sont demandées par le biais des requêtes, les attributs basés sur les noms sont nécessaires pour préciser les propriétés des données.

Les protocoles de routage à plat fonctionnent bien quand le réseau ne comprend pas un grand nombre de nœuds mais lorsque la taille du réseau devient de plus en plus importante, sa gestion devient plus difficile.

1.3.2 Protocoles basé sur la localisation :

Ce type de protocole consiste à partitionner le réseau en zones dans le quel chaque zone a un identifiant, une zone est constitué par un ensemble des nœuds, et le nœud est identifié par un EUI (End-system Unique Identifier), et enregistre dynamiquement l'identifiant de la zone à laquelle il appartient temporairement. A l'aide d'un GPS, il est possible de retourner l'information temporaire de localisation d'un nœud qui est appelée LDA (Location DependentAddress), et qui est un triplet de coordonnées géographiques (longitude, latitude, altitude), mais la précision de cette information est dépendant de type de l'application.

Dans ce cas la typologie nécessite un algorithme pour la gestion de localisation qui permet aux nœuds de déterminer les endroits approximatifs des autres nœuds. Ce type de topologie est mieux adapté aux réseaux avec une forte mobilité.

Avant l'envoi de données, le nœud source utilise un mécanisme pour découvrir la localisation de nœud destination, et puis inclus l'identifiant de zone de localisation et l'identifiant du nœud destination dans l'entête du paquet à envoyer.

1.3.3 Protocoles de Routage Hiérarchique :

La technique de hiérarchisation sert à diviser le réseau en sous-ensembles afin de faciliter la gestion du réseau surtout le routage. Dans ce type de protocoles les nœuds peuvent avoir des rôles supplémentaires, et sont partitionnés en deux types de groupes: la zone et le cluster.

Le cluster : est défini par un ensemble de nœuds qui possède un nœud chef nommé Cluster Head (CH), l'objectif de Cluster Head est de faire le relais entre les nœuds du cluster et la station de base directement ou via d'autres CHs. Le Cluster Head est caractérisé par des ressources énergétiques supérieures aux autres nœuds du réseau. Cette technique est appelée clusterisation.

— **La zone:** est définit par un ensemble de nœuds qui ne possède pas un nœud chef, ainsi, un cluster est une sous-classe d'une zone.

D'après les travaux des chercheurs [2], la consommation énergétique diminue lorsque le nombre de niveaux de la hiérarchie augmente.

1.4 Domaine d'application des réseaux de capteurs :

Les RCSFs ont été classés parmi les 21 technologies les plus importantes de ce siècle [3]. On les trouve dans plusieurs domaines d'applications comme:

1.4.1 Applications militaires:

Les premières applications des réseaux de capteurs ont concerné le domaine militaire [4,5]. L'idée était de déployé sur un secteur stratégique ou difficile d'accès pour surveiller tous les mouvements (ennemies), ou d'analyser le terrain avant d'envoyer des troupes (détections des armes chimiques, biologiques, ou de radiation), protéger les villes contre les attaques, tels que les menaces terroristes et aider les unités dans un champ de bataille.

1.4.2 Applications liée à la sécurité :

Les réseaux de capteurs sont aussi destinés pour la sécurité des zones privés et publiques. Des applications pourraient être mises en place pour protéger des parcs, des zones sauvages, des structures d'avion, des métros, de navires, et d'autres zones liées à la protection des ressources naturelles. La prévention d'accidents par la surveillance des routes ou des voies c'est aussi parmi les applications des réseaux de capteurs.

1.4.3 Applications médicales :

Les champs d'applications des réseaux de capteurs sont vastes dans le domaine de la médecine [6]. Parmi ces applications médicales : (1) la surveillance des patients à distance à l'aide d'un system en temps réel pour faciliter la communication entre les spécialistes qui sont disponibles pour la consultation à distance et les pompiers qui portent les malades. (2) Des applications pour la surveillance de la glycémie, le cancer ou la progression d'une maladie, ce qui permet aux malades de prendre en charge leur maladie. (3) Des applications pour la surveillance des maternités.

1.4.4 Applications environnementales :

Les réseaux de capteurs nous donne la possibilité de détecter des problèmes environnementaux [7] comme la pollution, étudier le comportement et les habituels des diverses

Espèces animales, la détection des incendies de forêts ainsi que la possibilité d'évaluer le taux de pesticides dans l'eau.

1.4.5 Application à la robotique :

Un réseau de capteur de caméras est appliqué pour des robots miniatures pour des applications de recherche et de sauvetage dans les zones urbaines. Dans cette application un ensemble de robots se déplace dans une zone sinistrée. Un robot équipé d'une caméra joue le rôle de source en enregistrant des images en les envoyant vert la station de base à partie d'autres robots (nœuds) qui se déplacent dans la région.

1.5 Les type des réseaux de capteurs :

Les RCSFs sont déployés sur terre, sous terre et dans l'eau selon l'environnement. Il ya cinq types de base de réseaux de capteurs [8]: les RCSFs terrestres, les RCSFs souterrains, les RCSF sous-marins, les RCSFs multimédia, les RCSFs mobiles.

1.5.1 RCSF terrestres :

Un RCSF terrestre (terrestrial WSN) consiste à un grand nombre de nœuds à faible coût déployé dans un espace donné. Les nœuds capteurs doivent être capables de transmettre les données à une station de base, puisque l'énergie de la batterie est limitée. Les capteurs terrestres peuvent être équipés d'une source d'énergies secondaires telles que les cellules solaires, ce type de RCSF est utilisé dans la capture et le monitorage environnemental.

1.5.2 RCSF souterrain :

RCSF souterrain (underground WSN) est un ensemble de nœuds, qui sont déployés sous le sol dans une cave ou mine pour surveiller les conditions souterrain, pour relayer les informations entre les nœuds et la station de base. Des nœuds sink sont déployés au-dessus sur la terre. Les RCSF souterrain sont plus chers que les RCSF terrestres en termes d'équipement, de déploiement et de maintenance. Le déploiement d'une RCSF sou terrain nécessite une planification soigneuse et, des considérations d'énergie et de coût. Les RCSFs souterrain sont utilisés dans plusieurs domaines d'application tel que : la surveillance agricole, la surveillance souterrain du sol et la surveillance des frontières militaires.

1.5.3 Les RCSF sous-marins

RCSF sous-marin (underwater WSN) est un ensemble de capteurs qui sont déployés sous l'eau, le nombre de nœuds de ce RCSF sont très peu par ce qu'ils sont très cher.

Des véhicules autonome sous-marins sont déployé pour exploré les données capturés. Les communications sans fil sont établies à travers des ondes acoustiques qui réside une bande passante limitée, une haute latence, un long délai de transmission. Les RCSFs sous-marins sont utilisés pour le contrôle de pollution, la prévention et la surveillance de désastres.

1.5.4 Les RCSF multimédias :

Les RCSF multimédia (multi-media WSN) est un ensemble de nœuds qui sont équipés des caméras ou des microphones. Ces nœuds capteurs sont capables de traiter, de stocker et de manipuler les données multimédia telles que le son, l'image et la vidéo. Les nœuds capteurs multimédias sont déployés d'une façon aléatoire pour garantir une couverture. Ils sont utilisés pour la surveillance et pour la traque des événements multimédia.

1.5.5 Les RCSF mobiles :

Les RCSF mobiles (mobile WSN) est une collection de nœuds qui ont l'aptitude de repositionner et s'organiser en réseau, ce qui représente la principale différence avec les nœuds statiques dans les autres types de RCSF. Les nœuds mobiles peuvent se communiquer quand ils sont a porté l'un de l'autre. Les challenges dans les RCSFs mobiles sont : le déploiement, l'auto-organisation, la couverture, le contrôle, le traitement de données, l'énergie et la maintenance. Les applications dans les RCSF mobiles incluent la recherche, le sauvetage, la surveillance d'environnement et la surveillance en temps réel à l'aide de robots.

1.6 La Consommation d'énergie dans les RCSF :

La consommation d'énergie des capteurs joue un rôle important dans la durée de vie des capteurs et donc celle du réseau car les capteurs sont munis d'une source énergétique limitée. Le changement des batteries des capteurs est usuellement impossible parce qu'il coûte plus cher qu'acheter un nouveau capteur.

La conservation d'énergie permet de consommer moins d'énergie pour toutes les couches d'un réseau de capteur. Allant de la couche physique et des techniques de modulation, jusqu'à la couche application et le développement de logiciels spécialisés. En effet, il ya quatre catégories de technique de conservation d'énergie : les techniques d'alternance entre les périodes d'activités et de sommeils ou "Duty-cycling", les techniques basées sur les protocoles de routage, et le contrôle de la topologie et les techniques orientées données basées sur l'agrégation de données [1].

1.6.1 Techniques du Duty-cycling :

Le principe est que tous les nœuds alternent entre périodes actives et de sommeil en fonction de l'activité du réseau. Un Duty-cycle est défini comme étant la fraction de temps où les nœuds sont actifs. Chaque nœud émetteur mettre la radio en mode veille (Low-power), à chaque fois que la communication n'est pas nécessaire, la radio doit être éteinte dès qu'il n'y a plus de données à envoyer et ou à recevoir, et devrait être prête dès qu'un nouveau paquet de données doit être envoyé ou reçu. Et comme les nœuds capteurs effectuent des tâches en coopération, il faut avoir un algorithme qui coordonne leurs dates de sommeil et de réveil [1] [3].

1.6.1.1 Protocoles Sleep/Wakeup :

Les protocoles Sleep/Wake up servent à limiter au maximum l'état en écoute en entrainant les nœuds dans un état endormi périodique, pour limiter l'utilisation de leurs énergies. Il existe trois types de réveil : les réveils périodiques, les réveils actifs, et les réveils organisés. Ces trois principaux de réveil sont utilisés pour construire sous forme de protocoles indépendants : à la demande, rendez-vous programmés, et régimes asynchrones. L'idée de bas de protocoles à la demande utilise l'approche la plus facile de la gestion d'énergie, le nœud devrait se réveiller seulement quand un autre nœud veut communiquer avec lui, mais le problème est comment savoir informer un nœud en sommeil qu'un autre nœud est disposé à communiquer avec lui. Et pour résoudre ce problème, ces systèmes utilisent généralement plusieurs radios avec différents compromis entre énergie et performances, c'est à dire : une radio à faible débit et à faible consommation pour la signalisation, et une radio à haut débit mais à plus forte consommation pour la communication de données. Le protocole STM est un exemple de ce type. La deuxième solution est le protocole, Sleep/Wake up rendez-vous programmé : qui consiste que chaque nœud doit se réveiller en même temps que ses voisins. Les nœuds se réveillent suivant un ordonnancement de réveil et restent actifs pendant un court intervalle de temps pour communiquer avec leurs voisins. Ensuite, ils se rendorment jusqu'au prochain rendez-vous. Et enfin, on trouve le protocole Sleep/Wake up asynchrone, qui consiste que chaque nœud peut se réveiller quand il veut et tant qu'il est capable de communiquer avec ses voisins [1].

1.6.1.2 Protocoles du niveau MAC :

Dans ce protocole, un ensemble des nœuds sont sélectionnés pour rester actifs, afin d'assurer la connectivité du réseau. Cependant, ces nœuds peuvent passer en mode sommeil quand ils n'ont pas de message à envoyer ou à recevoir. Il y a trois classes de protocoles MAC : les protocoles MAC reposant sur TDMA, les protocoles à accès aléatoire, et les protocoles hybrides.

— **Protocoles MAC reposant sur TDMA** : Dans le protocole MAC basé sur l'accès multiple par répartition temporelle (TDMA), le temps est divisé en trames où la trame est composée d'un certain nombre de slots de temps. Chaque nœud est attribué un ou plusieurs slots par trame, Il utilise ces slots pour l'émission/réception de paquets de/vers d'autres nœuds. Dans le cas général, les nœuds sont regroupés pour former des clusters avec un cluster head qui est chargé d'attribuer les slots de temps pour les nœuds de son cluster. Parmi les protocoles de TDMA, et qui est efficace en termes de conservation d'énergie on trouve TRAMA. Il divise le temps en deux parties, une période avec un accès aléatoire et une période avec un accès ordonnancé. La période d'accès aléatoire est consacrée à la réservation des slots et l'accès au canal est fondé sur la contention. Au contraire la période d'accès ordonnancée est constituée d'un ensemble de slots de temps attribués aux nœuds. L'algorithme de réservation des slots se fait en trois étapes: tout d'abord, les nœuds cherchent des informations sur un voisinage à deux sauts, qui sont nécessaires pour établir un ordonnancement sans collisions. Ensuite, les nœuds commencent une procédure d'élection afin d'associer chaque slot à un seul nœud. Chaque nœud aune priorité pour être le propriétaire d'un slot, cette priorité est calculée avec une fonction de hachage de l'identifiant du nœud et du numéro du slot. Le nœud avec la plus grande priorité devient le propriétaire du slot. Enfin, les nœuds envoient un paquet de synchronisation contenant la liste des voisins destinataires pour les transmissions suivantes [1].
— **Protocoles à accès aléatoire:** Les nœuds dans ce protocole partagent le canal de transmission, et ils ont le même droit d'accès. L'utilité est de garantir une adaptation dans le changement, la densité et dans la topologie du réseau. Mais l'inconvénient est l'accès concurrent qui provoque des collisions et donc la perte des paquets, et l'énergie au niveau de la couche MAC. CSMA et CSMA/CA [1]. Dans CSMA, le nœud émetteur écoute le canal avant l'envoi de paquet, s'il n'est pas occupé, le nœud transmet le message immédiatement. Sinon, Le nœud doit attend un temps aléatoire avant tenter de transmis ces paquets. Mais le problème de perte des paquets existe toujours. Pour cela une autre solution a été établie. CSMA/CA (CSMA/CA pour CSMA Collision Avoi dance) a été proposé, quand les nœuds voisins transmettent au même moment, les réseaux sans fil évitent les collisions en échangeant des messages de contrôle afin de réserver le canal avant chaque transmission de données [1].
— **Protocoles hybrides :** Ce type de protocoles, propose de combiner les deux méthodes : TDMA et CSMA, ces protocoles essaient d'avoir les avantages de deux méthodes en combinant entre les deux de manière intelligente. Il existe plusieurs exemples de protocoles hybrides, parmi ces exemples : Z-MAC (Zebra Media Access Control),ce protocole change dynamiquement le mode de transmission

entre CSMA et TDMA en fonction de la charge actuelle du réseau. En effet, il utilise CSMA comme un protocole de base pour l'accès au canal, et il utilise TDMA pour améliorer la résolution de contention entre les nœuds. Un autre exemple c'est Funneling-MAC, qui utilise CSMA et TDMA à la fois pour résoudre le problème de congestion et de perte de paquets observé par les nœuds proches du puits. Dansla réalité, TDMA est géré par le puits, et tous les nœuds utilisent CSMA par dé-faut, à moins de recevoir un message du puits les informant qu'ils doivent utiliser TDMA [1].

1.6.2 Les protocoles de routage efficace en énergie :

Les protocoles de routage jouent un rôle très important dans la conservation d'énergie, puisqu'ils prennent en considération les métriques relatives à la consommation d'énergie dans le calcul du coût d'une route vers le puits. Et pour rendre ca possible, ces protocoles choisissent le chemin ayant le plus petit nombre de sauts. Mais à condition que les autres métriques sont respectées, et particulièrement: le débit de transmission sur les liens radios, le niveau de charge des batteries des nœuds, et le niveau de fiabilité des liens radio, par ce que: les erreurs de transmission entrainent des retransmissions qui coûtent cher en énergie. Parmi les protocoles les plus efficaces en termes de consommation d'énergie entre les nœuds du réseau, et qu'on a déjà présentés précédemment, les protocoles LEACH, LEACH-C, et PEGASIS qui appartient à la famille de protocoles hiérarchiques. Enfin GEAR, qui représente la famille de protocoles basés sur la localisation [1].

1.6.3 Techniques centrées sur les données :

Les techniques basées sur les données sont nécessaires, si on prend par exemple les plans Duty-Cycling, on va trouver qu'ils ne prennent pas en considération les données prélevées par es nœuds, en réalité la collection des données influe négativement sur la consommation d'énergie de deux manières. La première c'est les échantillons inutiles : généralement les données échantillonnées contiennent des informations spatial et/ou temporelle sur le nœud récepteur, donc ce n'est pas nécessaire de communiquer ces informations redondantes à la station de base. Puisqu'un échantillonnage inutile implique une consommation d'énergie, même si le coût de l'échantillonnage est négligeable, cela causé par des consommations tout de long du chemin qu'emprunte le message. La deuxième c'est la consommation électrique du module de détection : on ne peut pas parler sur la conservation de l'énergie sur un réseau, par la limitation de communication sans parler sur la consommation de nœud capteur lui-même. Par la suite dans cette section on va présenter deux approches ou bien deux solutions qui peuvent être utiles à la réduction de la consommation d'énergie [3].

1.6.3.1 Réduction des données :

La réduction de données signifie qu'il faut limiter le nombre ou le volume de paquets

sur le réseau, parmi les méthodes de réduction des données : In-network processing, cette méthode consiste à rassembler les données par faire un calcule de moyenne de certaines valeurs au niveau des nœuds intermédiaires entre la source et le sink. Aussi, une compression de données peut être appliquée mais au niveau de nœud émetteur, et la décompression se fait au niveau de nœud récepteur [3].

1.6.3.2 Acquisition de données efficaces en énergie :

Selon [3], les chercheurs ont remarqué que dans certains gradeurs physiques qui sont mesurées ne change pas, entre deux échantillons, c'est le cas de température lorsque la dynamique est lente. Et c'est ce qui encouragé les chercheurs à exploiter la corrélation temporelle des données. Il existe en réalité deux catégories de techniques basées sur la collecte des données : techniques basées sur la prédiction et les techniques basées sur l'échantillonnage.

- **Technique basée sur la prévision des données:** Consiste a établi un modèle lors de la collection de données, pour que les valeurs futures puissent être prédites avec une certaine précision. Ce modèle permet de réduire le nombre de données acquis, et bien sûr la quantité de données à transmettre au puits [1].
- **Technique basée sur l'échantillonnage :** Les techniques basées sur l'échantillonnage sont divisées en trois classes : l'échantillonnage adaptatif, hiérarchique, actif fondé sur un modèle [3].

1.6.4 Contrôle de la topologie :

Le contrôle de la topologie consiste à diminuer la consommation de l'énergie par : l'élimination des nœuds redondants (mise en sommeil ces nœuds), et l'élimination de liens inutiles, par l'ajustement de la puissance de l'émetteur radio, et donc la portée de la communication. D'autre façons c'est la réduction de la topologie initiale de réseau mais à condition que le champ de captage doit être capturé [1].

1.7 Le cas des réseaux de capteurs d'images (RCSFI) :

Ce type de capteur est facile à réaliser, et peut être adapté facilement à des dispositifs tels que : les téléphones portables, les ordinateurs portables et les PDAs, cependant ces dispositifs sont dotée de ressources importantes en termes de mémoire et de capacité de calcul. Ces capteurs sont équipés avec des mini-caméras en technologie CMOS qui consomme très peu d'énergie comme par exemple: la caméra ADCM-1650 chez agilent

technologies. Ils peuvent prendre des photos qui peuvent être mémorisées en formats vectorielles ou matricielles.

Ce type RCSF diffère des autres types par:

— Les images sont représentées sur plusieurs milliers d'octets en fonction de leur taille et leur résolution. Pour cela, le nœud capteur doit générer plusieurs paquets pour transmettre une image entière contrairement aux réseaux de capteurs classique qui capturent des données scalaires simples codées sur quelques octets (2 ou 3 octets maximum) comme les valeurs de température et l'humidité,..., etc.).

— La plupart des capteurs traditionnels ont un champ de perception omnidirectionnelle, cela veut dire que l'orientation du capteur n'a pas d'impact sur la valeur du gradeur physique mesuré.

La consommation d'énergie est un problème fondamental qu'on le trouve beaucoup plus dans les réseaux de capteurs d'images comme dans les réseaux de capteurs en général. Puisqu'un nœud capteur d'image est alimenté par une batterie et comme les images sont codées sur des milliers d'octets ce qui nécessite de générée plusieurs paquets par les nœuds capteurs pour qu'ils puissent les transmettre et donc consommer beaucoup d'énergies. Parmi les techniques les plus utilisées pour résoudre le problème de la consommation d'énergie est la compression d'image.

1.8 Conclusion :

Les RCSF présente une nouvelle étape dans le développement des technologies de l'information et de communication.

Dans ce chapitre, nous avons donné une vue général sur les réseaux de capteurs sans fil, en présentant leur architecture, leur conception et réalisation, leurs diverses applications, leur types, et les différentes protocoles de routage utilisées dans ce type de réseau, nous avons aussi parlé de la capacité énergétique limité qui se caractérise les réseaux de capteurs, avec quelques techniques de conservation d'énergie.

Dans le chapitre suivant, nous aborderons sur une des techniques de conservation d'énergie qui est la compression d'images ou notre travail se concentre sur ça.

Chapitre 2 : La Compression d’images dans les RCSFIs

2.1 Introduction :

La grande transformation numérique que notre monde a connue a contribué de manière significative à augmenter la transmission de données à travers les réseaux en très peu de temps et avec une bonne qualité également. Cet énorme flux d'informations a grandement aidé à accéder et à traiter les informations plus rapidement et donc à augmenter le volume et la vitesse de production dans divers domaines de la vie.

Cependant, la transmission de données sur le réseau est un processus très couteux du coté énergétique, surtout si ces réseaux sont des réseaux de capteurs sans fil, les nœuds capteurs sont très limités en termes d'énergie et leurs remplacement est très couteux, et ici l'importance d'algorithmes de compression pour réduire la consommation d'énergie et ainsi prolonger au maximum la durée de vie du réseau.

Dans ce chapitre, nous parlons en détail des algorithmes de compression de données et d'images en particulier, car c'est l'objectif principal que nous voulons présenter dans notre recherche.

2.2 Les Caractéristiques d'une image :

Pour comprendre comment fonctionne la compression d'une image, nous devant tout d'abord savoir ce qu'est une image, quel sont ses différentes caractéristiques, ses types dans une première partie.

- **pixel :** En informatique, une image est un ensemble de points appelés pixels (contraction de Picture élément), l'ensemble de ces pixels est représenté dans un tableau à deux dimensions constituants l'image. Le pixel est l'unité de base permettant de mesurer la définition d'une image.
- **les images en niveau de gris :** Le niveau de gris d'une image permet d'obtenir plus de nuances que le simple noir et blanc, chaque pixel est un niveau de gris allant de 0(noir) à 255(blanc).Cet intervalle de valeurs signifie que chaque pixel est codé sur huit bits (un octet).
- **les images binaires (noire ou blanc) :** C'est le type des images les plus simples, un pixel peut prendre uniquement les valeurs noires ou blanches. On les utilise souvent pour scanner un texte quand il est composé d'une seule couleur.
- **les images en couleurs :** Une image en couleur est également un tableau rectangulaire de pixels. Ils sont représentés à l'aide de trois composants (par exemple RVB). Chaque pixel est un triplet de trois valeurs numériques, la signification de ces valeurs dépend du type de codage choisi. Le plus utilisé pour la manipulation des images numériques est l'espace de couleur Rouge, Vert, Bleu (RGB en anglais).
- **la définition et la taille d'une image :** On appelle définition d'une image le nombre de points (pixels) constituants l'image c'est-à-dire le nombre de colonnes de l'image que multiple son nombre de lignes. La taille de l'image est alors le nombre de pixels multiplié par le nombre de bits utilisé pour le codage des couleurs.

- **formes d'images** : Il existe deux formes d'images :

- **Les images vectorielles** : appelées aussi « cliparts » c'est des images numériques composées d'objets géométriques individuels (segments de droite, polygones, arcs de cercles…), définies chacun par divers attributs de formes, de position, de couleur…, l'inconvénient de ce type d'image est que pour atteindre une qualité photo réaliste, il faut pouvoir disposer d'une puissance de calcul importante.
- **Les images matricielles (ou images bitmap) :** sont composées d'une matrice de points à plusieurs dimensions, chaque dimension représente une dimension spatiale (hauteur, largeur, profondeur), temporelle (durée) ou autre (un niveau de résolution).

Les images actuelles comme les écrans d'ordinateurs reposent essentiellement sur des images matricielles, les descriptions vectorielles doivent préalablement être converties en descriptions matricielles avant d'être affichées comme images.

2.3 La compression d'image :

La compression est un processus visant à produire une représentation compressée d'une image. La compression d'image résout le problème de la réduction de la quantité de données nécessaire pour représenter une image, réduisant ainsi les exigences de stockage et de transmission d'image. La compression est obtenue en supprimant un ou plusieurs des trois types de redondance de données de base : redondance de codage, redondance inter-pixel et redondance psychovisuelle.

La redondance dans le codage résulte de la moindre utilisation de mots de code optimal, tandis que l'inter-redondance résulte des corrélations entre les pixels de l'image numérique. La répétition psycho-visuelle est due aux données que le système visuel humain ignore, qui représentent les informations visuelles non essentielles. L'objectif de la compression d'image est de réduire autant que possible le nombre de bits dans l'image d'origine tout en conservant sa précision et sa qualité. L'image est reconstruite en appliquant un processus inverse appelé le décodage, les systèmes de compression d'image sont caractérisés par deux blocs structurels : l'encodeur et le décodeur [19].

Il convient de noter qu'en compression d'image, tous les algorithmes de différents types sont évalués selon trois critères principaux, à savoir [17] :

- La qualité de reconstitution de l'image.
- Le taux de compression.
- La rapidité de codeur.

La compression peut être avec perte ou sans perte de données :

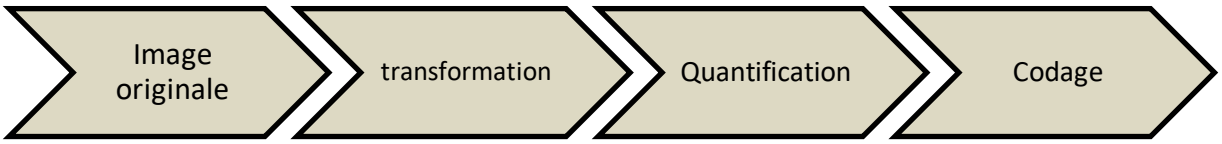


Figure 2.1 : Schéma d'un codeur d'image.

1- Compression avec perte (lossy compression) :

La compression avec perte de données conduit généralement à un niveau élevé et a un bon résultat lorsqu'elle est appliquée à certains types de fichiers, tels que l'audio, les vidéos et les images. Mais dans le cas des textes, cette compression entraine une détérioration de la qualité des données car elle réduit la taille du fichier en supprimant définitivement les informations redondantes, afin que cette suppression soit signalée par l'utilisateur, et ne soit pas constatée dans les autres types de données cités précédemment [20].

2- Compression sans perte (losless compression) :

La compression sans perte de données ou la compression dite « non destructive », conduit à la création de données originales qui sont complètement identiques aux données avant le processus de compression, c'est-à-dire que les algorithmes compriment les données sans les affaiblir ni détériorer leur qualité, en réduisant la taille du fichier est moindre par rapport à la compression avec perte de données, ce type est également utilisé pour les textes [20][21].

Dans le tableau suivant nous résumons les principaux points de différences et de similitudes entre les techniques de compression avec et sans perte [20] [22].

Base de comparaison	La comparaison avec perte	La comparaison sans perte
De base	La compression avec perte appartient à une famille de méthodes de codage de données, qui utilisent des estimations imprécises de la représentation du contenu.	Ce sont des algorithmes de compression de données, l'une de leurs caractéristiques les plus importantes est qu'ils permettent de reconstruire les données d'origine avec qualité et précision à partir des données compressées.
Utilisé dans :	Images, audio et vidéo.	Texte ou programme, images et son.

Algorithme :	Transformation en codage, DCT, DWT, compression fractale, RSSMS.	RLW, LZW, codage arithmétique, codage de Huffman, codage Shannon Fano.
Application :	JPEG, GUI, MP3, MP4, OGG, H-264, MKV.	RAW, BMP, PNG, WAV, FLAC, ALAC etc.
Capacité de stockage de données :	Plus par rapport à la méthode sans perte.	Moins par rapport à la méthode avec perte.

Tableau 2.1 : Répartition des parasites sur les hôtes étudiés.

2.4 Critères d'évaluation des algorithmes de compression :

Le taux de compression (eq2.1) c'est un des mesures de performance de méthodes de compression d'images fixes. Il est définit comme un rapport du volume des données après compression sur le volume initial de données :

$$\textbf{Taux} = \frac{\textbf{taille en bits de l'image compressée}}{\textbf{taille en bits de l'image originale}} \qquad (2.1)$$

Classiquement, en compression d'image on se base sur l'erreur quadratique moyenne (MSE-mean square error) (eq2.2) pour évaluer la qualité de chaque image, ou plutôt la mesure de dégradation engendrée par la compression :

$$\textbf{MSE} = \frac{1}{n*m}\sum_{i=1}^{m}\sum_{j=1}^{n} |xrec[i,j] - x[i,j])|^2 \qquad (2.2)$$

Dans la littérature, les mesures de qualité, PSNR (eq2.3) sont en général exprimées en rapport signal sur bruit, soit, dans le cas d'un signal représenté sur 8 bits :

$$\textbf{PSNR} = 10\log10\left(\frac{L^2}{MSE}\right) \qquad (2.3)$$

L : nombre d'échelons de luminance (255 pour 8 bits)

2.5 Les techniques de compression d'image :

Il existe plusieurs techniques de compression d'images, on présente quelques techniques :

2.5.1 RLE : Run-LengthEncoding :

L'idée principale de cette approche de la compression des données est de savoir si l'élément de donnée « d », apparait n fois consécutives dans le flux d'entrée, les itérations n sont remplacées par la paire unique « nd ». Cette méthode de compression de données est appelée codage de longueur d'exécution, ou RLE, comme le montre la figure suivante. RLE est fortement recommandé dans la compression de données graphiques.

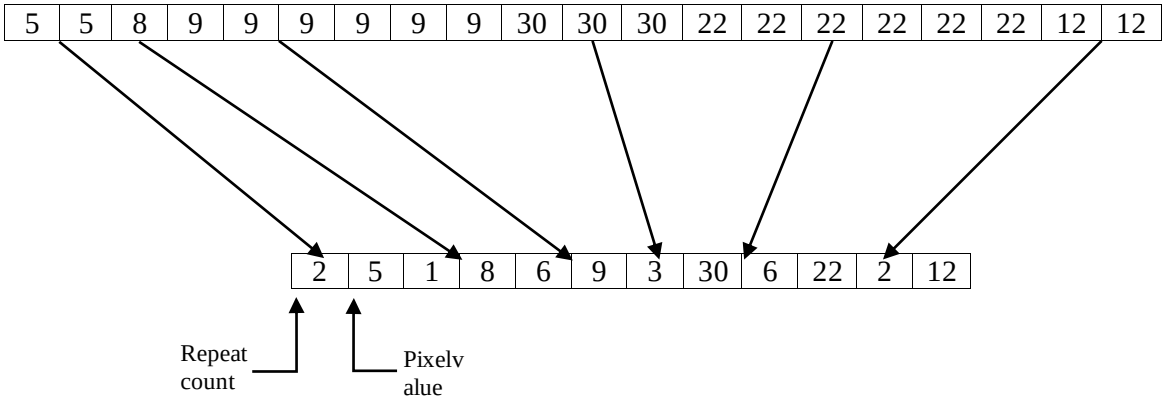


Figure 2.2 : Le principe de RLE.

RLE, est basé sur le principe que si on choisit au hasard un pixel dans l'image, il y a une forte probabilité que ses voisins aient la même couleur. Ainsi le compresseur scanne le bitmap ligne par ligne, à la recherche du nombre de pixels de la même couleur. Par exemple, si un bitmap commence par 15 pixels blancs suivis de 30 pixels noirs, suivis de 45 pixels blancs, le flux de sortie ne contiendra que ces nombres dans l'ordre : 153045.

Le compresseur suppose toujours que le bitmap commence par des pixels blancs, sinon, le bitmap commence par zéro pixel blanc, le flux de sortie doit commencer par une longueur de lecture de 0 et la résolution du bitmap doit également être réduite au début du flux de sortie.

Plus l'image est complexe, plus la compression est mauvaise. La figure (2.3) montre comment les lignes rasters traversent une zone uniforme. De sorte que la ligne de balayage entre à partir d'un point sur la circonférence de la zone et sort à partir d'un autre point, ces deux points ne sont utilisés par aucune autre ligne de balayage. Le nombre de ligne raster qui traversent une zone uniforme est d'environ la moitié de sa circonférence, qui est mesurée en pixels. Et parce que la région est unifiée, chaque ligne de sondage contribue à deux opérations dans le flux de sortie pour chaque région qu'elle traverse. Ainsi, le taux de compression dans une zone uniforme est approximativement égal au rapport [9] :

$$\textbf{Taux} = \frac{\textbf{2} * \textbf{half the length of the perimeter}}{\textbf{total number of pixels in the region}} \qquad (2.4)$$

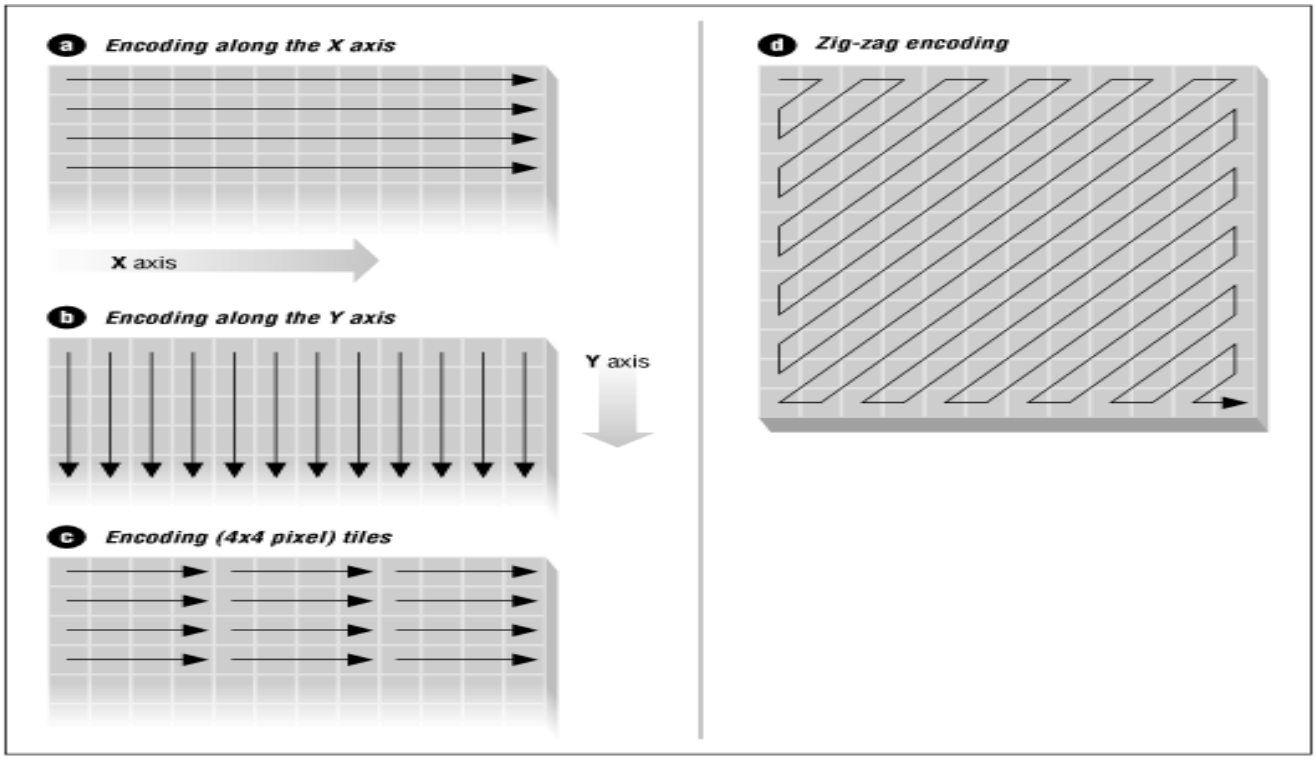


Figure 2.3 : La compression RLE

Il y a trois points importants qui doivent être mentionnés, qui est : premièrement, la longueur d'exécution ne peut pas être 0, donc elle est écrite [runlength minus one] sur le flux de sortie. Deuxièmement, dans une image couleur, il est très courant que chaque pixel soit stocké sur trois octets, ce qui représente l'intensité des composantes rouge, verte et bleue du pixel. Dans un tel cas, les opérations de chaque couleur doivent être codées séparément en séquences. Chaque séquence doit être sa longueur encodée séparément. La même méthode de compression pour les images en niveaux de gris est utilisée pour compresser les images en couleur. Troisièmement, il est recommandé d'encoder chaque ligne bitmap séparément. Donc, si la ligne se termine par 5 pixels avec une densité de 90 et que la ligne suivante commence par 8 de ces pixels, il est préférable d'écrire : 5, 90, 8, 90 sur le flux de sortie et il est préférable d'écrire « eol », ce qui signifie la fin de la ligne 5, 90, eol, 8, 90.

Parmi les inconvénients du compresseur RLE, il y a le fait que toutes les longueurs d'exécution doivent être repensées si l'image est modifiée. De plus, la sortie RLE peut souvent être plus longue que le stockage initial d'un bitmap non compressé [9].

2.5.2 Codage de Huffman :

Le codage de Huffman est une méthode très populaire pour compresser des données, en particulier des images. Cet encodeur est à la base de nombreux programmes courants utilisés sur les ordinateurs, et peut également être utilisé en une seule étape dans un processus de compression en plusieurs étapes. La méthode [Huffman 52], l'une des meilleures méthodes qui produit de bons codes, lorsque les probabilités des symboles

sont des puissances négatives de 2, tout comme dans la méthode de « Shannon-Fano », mais la seule différence entre elle est que la méthode de Shannon construit ses codes de loin de gauche à l'extrême droite, tandis que Huffman construit un arbre de codes de bas en haut, c'est-à-dire qu'il construit des codes de droite à gauche.

La méthode de Huffman, construit une liste contenant tous les symboles alphabétiques dans l'ordre décroissant de leurs probabilités, puis construit un arbre de bas en haut, de sorte que chaque feuille de l'arbre représente un symbole spécifique de la liste. Cela se fait en plusieurs étapes. Chaque fois que les deux symboles avec les plus petites probabilités sont sélectionnés, ajoutés en haut de l'arbre partiel, retirés de la liste, et remplacés par un symbole supplémentaire représentant les deux symboles, l'arbre devient complet lorsque la liste est réduite à un symbole supplémentaire représentant l'alphabet dans son intégralité [12].

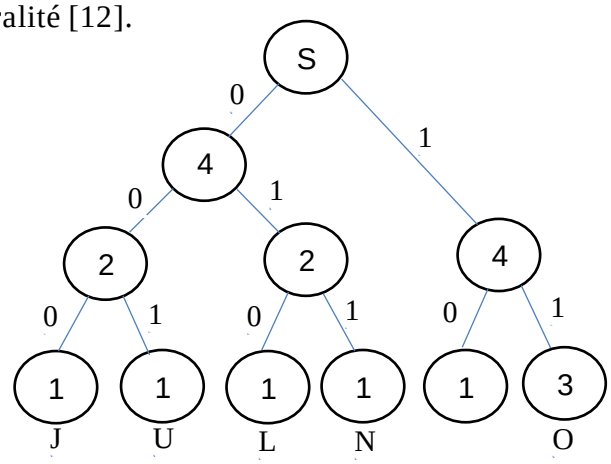


Figure 2.4 : Algorithme de compression sans perte de Huffman

La méthode de Huffman peut être résumée dans les étapes suivantes :

Etape1 : Lister les possibilités de code source, produire un ensemble de nœuds et mettre ces probabilités dans les feuilles d'un arbre binaire.

Etape 2 : Créer de nouvelles probabilités, chaque nouvelle probabilité représente la somme de deux nœuds avec les deux plus petites probabilités de la somme des nœuds.

Etape 3 : Créer un nœud majeur avec la nouvelle probabilité et marquer la branche de sous-nœud supérieur ou gauche avec la valeur « 1 », et la branche du sous-nœud inférieur ou droit avec la valeur « 0 ».

Etape 4 : Remplacer les deux nœuds par les deux plus petites probabilités pour avoir un nouveau(le nœud nouvellement produit). Si l'ensemble de nœuds ne contient qu'un seul nœud, arrêter, sinon passer à l'étape 2, [13].

2.5.3 Le codage LZW (LempelZivWelch) :

LZW porte le nom des scientifiques qui l'ont développé et ils sont : 'Abraham Lempel', 'Jakob Ziv' et 'Terry Welch'. Cet algorithme est basé sur un 'dictionnaire' sans perte. Ce type d'algorithme analyse le fichier et trouve des séquences de données qui se produisent plus d'une fois, ces séquences sont stockées dans un dictionnaire et à

l'intérieur du fichier compressé, puis les références sont placées à la place des données en double. Le compresseur LZW remplace les caractères avec des symboles individuels de longueur aléatoire mais le symbole doit contenir des bits de plus d'un caractère. Les 256 premiers symboles sont utilisés pour le jeu de caractères standard tandis que le reste des symboles est utilisé pour les chaines pendant que l'algorithme continu.

Le compresseur LZW, est la meilleure technologie pour réduire la taille des fichiers qui contiennent beaucoup de données répétitives, et c'est l'une des techniques les plus rapides et les plus simples parce qu'aucune donnée n'est perdue pendant ou après la compression. Cet algorithme est très efficace car il n'a pas besoin d'envoyer la table de chaines à l'algorithme de décompression, la table peut être reconfigurée exactement telle qu'elle était lors de la compression en utilisant les entrées comme données [14].

2.5.4 La compression JPEG (JoinPhotographic Experts Group) :

La norme JPEG, est une norme d'ISO/ITU-T, été créé à la fin des années 1980. Cette norme cible les applications d'images fixes en couleur. Le principe de cet algorithme est de diviser l'image en blocs de 8*8 pixels, ces blocs sont soumis à un traitement. D'abord une transformation cosinus discrète bidimensionnelle est effectuée, puis les paramètres de transformation sont quantifiés uniformément en combinaison avec une table de 64 éléments dont le but est de spécifier les pas de quantification, et la sélection d'une étape de quantification importante pour certains composants qui sont visuellement sans importance, car les informations pertinentes pour l'image, qui sont caractérisées par des signaux bidimensionnels Img (x ،y), sont concentrées aux fréquences spatiales les plus basses [16][17].

Une table standard est configurée selon le critère et n'est pas imposée. Enfin un codage entropique est mis en place, sans distorsion permettant l'utilisation des propriétés statistiques de l'image. Il commence par agencer les coefficients selon un balayage en zigzag pour les mettre correspondants aux fréquences les plus basses, cela donne une série de symboles. Le code de Huffman consiste à représenter les symboles les plus probables par des codes comportant un nombre de bits le plus petit possible [17].

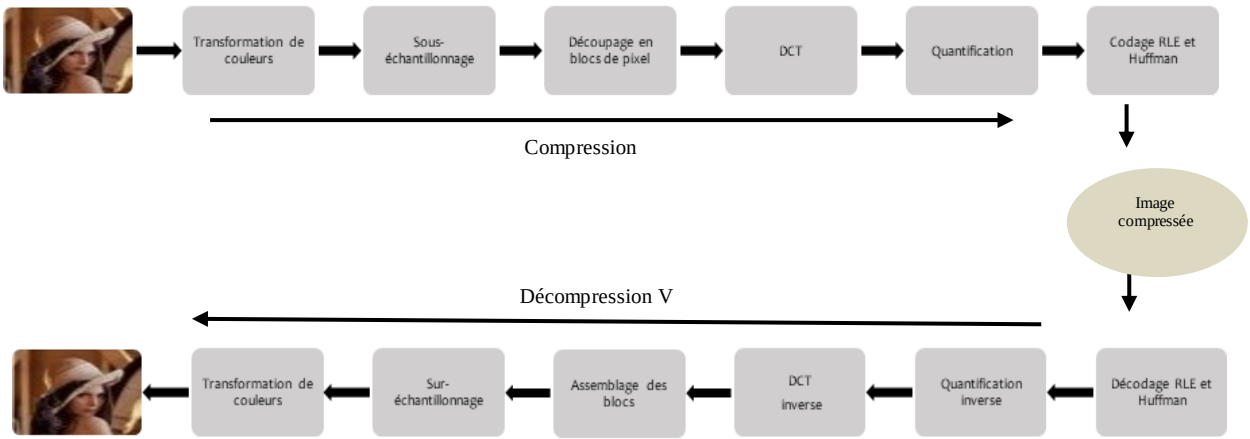


Figure 2.5 : L'algorithme de JPEG pour la compression d'image.

Les Étapes du processus JPEG pour les images couleur :

- Convertir l'espace colorimétrique de RGB en YCBCR (spécification des couleurs).
- Diviser l'image originale en blocs de 8*8 pixels.
- les valeurs de pixel à l'intérieur de chaque bloc vont de [127- 128], et les valeurs de pixel pour les images en noir et blanc vont de [0 - 255], donc chaque bloc est décalé de [0 - 255] a [-128 à 127].
- DCT est appliqué à chaque bloc, il fonctionne de gauche à droite et de haut en bas.
- Chaque bloc est compressé par quantification.
- La matrice quantifiée est codée entropie.
- La reconstruction de l'image compressée par un processus inverse qui utilise la transformation en cosinus inverse (IDCT) [17].

2.5.5 La compression JPEG2000 :

JPEG2000 a été créé, à la suite d'un partenariat entre ISO, IEC et ITU-T, ou cette norme est capable de fonctionner sans et avec perte de données en utilisant la transformation en ondelettes. Les performances de JPEG2000 sont bien meilleures que celles de JPEG. JPEG2000 rend les fichiers avec moins de poids pour une qualité d'image égale, et il offre de meilleurs contours plus nets et contrastés. Il combine également un certain nombre de fonctionnalités nouvelles et avancées telles que: l'évolutivité, la résistance aux erreurs de transmission, le codage sans perte, la versatilité de l'organisation des données, sans oublier les diverses extensions qui ciblent l'application (l'interaction, la sécurité et sans fil).

Toutes ces fonctionnalités ont rendu les professionnels de la photo très intéressés par cette norme, bien qu'elle ait peu d'applications, et que sa présence reste limitée sur le web [24].

JPEG2000, utilise la transformation en ondelettes des pixels de l'image, où les pixels de l'image sont convertis en fréquences dans lesquelles chaque pixel correspond à une seule fréquence. Ce processus aboutit à plusieurs sous-images, selon le nombre de divisions successives de l'image originale. Dans la plupart des cas, les nombres de basses fréquences dépasse le nombre de hautes fréquences, car la présence d'un plus grand nombre de hautes fréquences signifie que les pixels sont différents les uns des autres dans une même image, ce qui est rare dans les images. A la suite de cette étape, l'étape de quantification, ou la destruction des fréquences les plus hautes est éliminée. Selon un certain taux de compression, si la compression n'est pas destructive alors l'étape de quantification n'est pas effectuée [23].

2.6. Les méthodes de compression récentes :

2.6.1 La compression fractale

La compression fractale ne s'applique qu'aux images, et est basée sur la géométrie fractale de Benoit Mandelbrot, cette compression n'est reconnue par les navigateurs qu'à travers l'extension .fif . L'idée de cette méthode est basée sur la prémisse que chaque image est un ensemble de tailles variables et un ensemble fini de transformations géométriques qui sont appliquées à des sous-ensembles de motifs similaires. Par conséquent, la compression fractale consiste principalement à remplacer l'image entière par une série de formules mathématiques, qui permettent sa recomposition complète. Ainsi, le succès du processus de compression est proportionnel aux propriétés géométriques de l'image, car plus le nombre de ces formules est élevé, plus le nombre de formules mathématiques est faible. De là, nous concluons que la compression fractale est une bonne solution pour traiter les images de paysage. Dans ce cas, il faut mentionner que la perte de données se situe au niveau de la démarcation, du fait qu'il est impossible de réduire n'importe quelle image avec une grande précision à un ensemble de propriété géométrique appliquée aux blocs.

Cette méthode présente des avantages, notamment que la compression est totalement indépendante de la taille de l'image, car seules les propriétés géométriques sont prises en compte. L'image est découpée selon ses propriétés en un ensemble de blocs de taille variable, ce qui permet d'éviter les effets de pixels courants en JPEG. Comme lors de l'agrandissement de l'image reconstruite, on remarque un effet de flou, qui représente la perte de données résultant du processus de compression. Parmi ses inconvénients, on trouve que les formules mathématiques chargées de gérer la transformation géométrique des blocs, ne sont pas capables de masquer ces blocs à l'image, ce qui rend ce procédé inadapté à un traitement vidéo, qui se caractérise essentiellement par l'enchainement d'images distinctes [21].

2.6.2 La compression par ondelettes :

La compression par ondelettes, ou DWT (Discrete Wavelet Transform), s'appuie sur la théorie mathématique pour analyser les signaux. Les ondelettes peuvent être définies comme un ensemble de signaux élémentaires permettant de reconstruire un signal complexe. Ainsi, la compression en ondelettes consiste principalement à analyser une image qui est perçue comme une référence à un ensemble d'image basse résolution. Ce processus dépend de la différence entre les zones à faible contraste et les zones à contraste élevé, et se déroule en trois étapes principales :

La première étape, est un processus complexe, au cours duquel l'image est convertie en ondes. Tout d'abord, les pixels moyens de l'image d'origine sont mesurés deux par

deux le long de l'axe horizontal, puis des sous-échantillons en sont prélevés dans la direction horizontale. Étant donné que la largeur des échantillons est égale à la moitié de la largeur de l'image original, une deuxième image est crée dans le sens horizontal, à condition de calculer des erreurs entre l'image d'origine et l'image qui n'a pas été suffisamment échantillonnée.

Deuxièmement, le nombre de pixels dans les deux images obtenues est moyenné le long de l'axe vertical, puis les erreurs résultant du processus sont à nouveau déterminées

A l'issue de cette étape, nous obtenons quatre images, la première est une image approximative résultant de la prise de sous-échantillonnages horizontaux puis verticaux sans correction, tandis que les trois autres images représentent les détails de l'image originale.

Dans la deuxième étape, nous effectuons une évaluation quantitative des informations et supprimons les détails inférieurs à une certaine limite. Enfin, l'image est décompressée en utilisant le schéma inverse. Le décodage des informations fournit un ensemble d'ondes de différents niveaux, permettant de reconstruire l'image, les mêmes étapes sont répétées autant que nécessaire en prenant à chaque fois pour base la première sous-image obtenue (voir figure 2.6) [21].

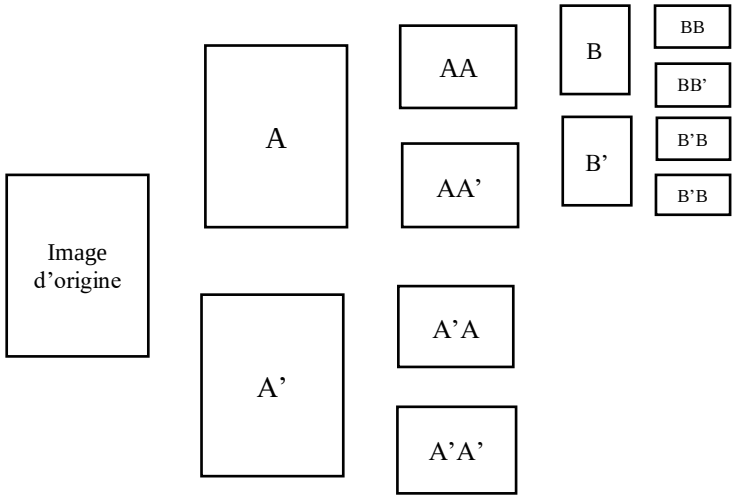


Figure 2.6 : Les étapes de compression par ondelettes

2.7 Traitement d'images dans les réseaux de capteurs sans fil :

L'unité de communication est l'un des composants électroniques les plus gourmands en énergie dans un nœud de capteur [20], le moyen d'économiser de l'énergie est de réduire la quantité de données àtransmettre. La solution la plus évidente est la compression.

Moins il y aura de données à transmettre moins le nœud capteur consommera d'énergie, cela n'est pas toujours valable car la compression a un coût d'énergie qui peut être très élevé. Autrement dit, le nœud capteur consomme plus d'énergie en ayant des

images compressées que des images non compressées.

Le problème est de disposer d'un algorithme de compression de données qui soit peu gourmand en énergie. Une liste d'algorithmes est disponible aujourd'hui, dont plusieurs concernant les images.

-La compression d'image dans les réseaux de capteurs sans fil est classifiée en deux types : compression locale et compression distribuée.

2.7.1 Compression locale :

Ici la compression d'image est réalisée au niveau de nœud source sans distribution de la charge de traitement des données avec d'autres nœuds, c'est-à-dire par des algorithmes locaux. Il existe plusieurs algorithmes de compression dans la littérature, certains peuvent fournir des taux de compression élevés mais ils ne sont pas applicables dans les réseaux de capteurs sans fil à cause des ressources limitées des nœuds capteurs.

Des travaux sur la compression locale pour évaluer les performances de plusieurs algorithmes de compression d'images traditionnelles, ces travaux ont analysé cinq algorithmes bien connus : JPEG, SS, SPITH, DCT et JPEG2000, les résultats ont montrés que le coût d'énergie de calcul est supérieur au coût de transmission de l'image non compressé pour, JPEG , SPITH, DCT et JPEG2000, L'algorithme SS est le seul algorithme qui amène des économies d'énergie par rapport aux cas sans compression. Le coût d'énergie et le coût d'implantation des algorithmes de compression dépendent aussi des caractéristiques de matériels.

Le problème qu'on trouve dans la compression locale est que le nœud capteur source consomme beaucoup d'énergie pour faire la compression et la transmission, alors que les nœuds intermédiaires font seulement la transmission et donc consommer moins d'énergie. La durée de vie de nœud source va diminuer ce qui implique la diminution de la durée de vie du réseau.

2.7.1.1Compression locale par JPEG :

C'est la technique de compression d'image la plus utilisée aujourd'hui. Elle est basée sur un découpage de l'image en blocs de 8*8 pixels en réduisant le coût de la DCT qui coûte plus en calcul, en ne calculant qu'une partie des coefficients parmi les 64 initiaux. Les blocs sont ensuite quantifiés et codés avec RLE et Huffman.

Parmi les moyens qui réduits le coût des étapes DCT c'est le masque de forme carrée qui est adopté par (Taylor et Dey, 2001) et le masque de forme triangulaire (TJPEG) qui est adopté par (Mammeri et al ; 2008). Un autre moyen aussi pour réduire le coût de la DCT c'est de faire les opérations dans un mode a virgule non flottante. Il calcule le nombre de bits minimums pour représenter les parties entières et décimales des valeurs réelles. (Chiasserini et Magli, 2002) ont obtenu une baisse de la consommation d'énergie d'un facteur de 8 environ sur un microcontrôleur StrongArm SA1110 [20].

2.7.1.2 Compression locale par JPEG2000 :

La complexité de l'algorithme JPEG2000 d'après (Moccagatta et Coban, 2000) était entre 38% et 264% supérieur à JPEG [21].

La technique de faible consommation énergétique qui introduit par (Wu et Abouzeid, 2004b) qui incorpore le standard JPEG2000 pour la compression d'images depuis un nœud caméra sans fil, cette technique était l'un des premiers travaux dont lequel il est considéré un scénario multi saut pour des réseaux de capteurs d'images. Ils ont formulé la transmission d'images comme un problème d'optimisation et ils ont proposé une heuristique appelée MTE (MinimizeTotalEnergy). Il est supposé que le nœud source connaisse le nombre de nœuds intermédiaires entre lui et le puis et chaque nœud est capable de connaitre son état de batterie. L'algorithme travaille comme suit : le nœud capteur compresse l'image capturée en appliquant la compression par ondelette, puis, il calcule un taux de compression et une dissipation d'énergie, elle est réalisée donc une nouvelle compression et des nouveaux taux de compression et dissipations d'énergie qui vont être comparés avec les anciennes valeurs. Ces étapes sont répétées jusqu'à un certain nombre d'itérations. Cette méthode mène à compresser l'image plusieurs fois sans avoir comment le nœud source va économiser de l'énergie.

Cette méthode vise à optimiser l'énergie dans les nœuds de transit mais elle n'est pas bonne pour la source.

2.7.2 Compression distribuée :

La nature distribuée des réseaux de capteurs a permis un bon nombre de travaux sur la compression distribuée des images. Deux types d'algorithmes de compression distribuée. Le premier (le plus développé) est basé sur la corrélation existante entre deux (ou plus) images capturées quand elles contiennent des scènes voisines. Le deuxième algorithme est basé sur la distribution du processus de compression d'une image à partir de plusieurs nœuds.

2.7.2.1 La compression distribution des images corrélées :

Le théorème présenté par Slepian et Wolf sur le codage des images corrélées indépendamment mais décodées toute ensemble. Ce théorème a inspiré plusieurs approches proposées pour le traitement collaboratif distribué des images sur les réseaux de capteurs sans fil. L'idée était la corrélation entre deux images ou plus d'une même scène mais originaire de plusieurs caméras, malgré la plupart des travaux publiés sur la compression distribuée ont travaillé sur le théorème de Slepian et Wolf mais l'application de théorème n'est pas l'exemple de difficultés. Gehrig et Dragotti ont étudié le cas d'images corrélées. Ils proposent une méthode distribuée pour profiter de la corrélation entre plusieurs vues capturées par plusieurs caméras adjacentes, en utilisant une certaine information géométrique afin de réduire le débit de transmission [20].

2.7.2.2 Compression basé sur la distribution du processus de compression :

Le travail proposé par (Wu et Abouzeid, 2004b), Un système de coopération pour distribuer le cout de calcul de DWT dans le cadre d'un codage JPEG2000.L'idée est de répartir la charge de travail du calcul de la transformée en ondelette entre les différents nœuds. Ce système est donné en deux variantes : une méthode de DWT parallèle et une méthode de carrelage.

1- Méthode de transformé en ondelette parallèle : l'image capturée est devisée en n blocs de données R1, R2, …, Rn, ou chaque bloc se compose d'une ou plusieurs rangées. Ces blocs sont transmis à certains nœuds voisins de la source. Ces nœuds effectuent une transformé en ondelette unidimensionnelle (1-D) pour chaque bloc de données qui lui a été envoyé, puis transmet le résultat à un nœud agrégat qui devise les données obtenues en m blocs I1, I2, I3, …, Im composé de colonnes. Ensuite il distribue ces blocs en les transmettant à des nœuds voisins. Ceux-ci effectuent la transformé en ondelette 1-D et les renvoient au nœud agrégat qui récupère tous les blocs et il obtient le résultat de la transformation bidimensionnelle (2-D) en ondelette de l'image.
2- Méthode de carrelage : Dans cette méthode l'image est divisée en tuiles (de blocs de données) puis ces tuiles sont envoyées à un nœud pour faire la transformation en ondelettes discrètes indépendamment .et enfin les résultats sont transmis à un nœud agrégat.

2.8 Travaux Connexes:

Dans ce travail, les chercheurs passent en revue les algorithmes de compression les plus importants et les plus courants utilisés pour réduire la consommation d'énergie dans les réseaux de capteurs sans fil, en indiquant les avantages et les inconvénients de chacun. Parmi ces algorithmes : jpeg, sphiht, ebcot, spec [37].

Alors que d'autres chercheurs se sont concentrés sur l'aspect matériel, l'objectif est L'objectif est d'obtenir un temps court lors du traitement d'image a faible consommation d'énergie, de sorte que la solution matérielle est conçue de manière a ce que les paramètres de compression puissent être modifiés au moment de l'exécution. Il est possible au niveau de WSNs d'ajuster le compromis entre l'utilisateur et les exigences variables ou immuables du réseau. Parmi les paramètres qui doivent être dynamique, le taille de l'image car ce paramètre est affecté par la puissance requise et le temps de réponse de circuit, ce paramètre peut être contrôlé par le nœud caméra afin d'améliorer sa durée de vie par rapport a l'énergie disponible dans sa batterie. Il peut également être modifie par l'utilisateur final en fonction des exigences de l'application et de la fiabilité du réseau. Lorsque la qualité de l'image diminue de la part du récepteur, l'utilisateur final

peut demander à réduire la taille de l'image dans l'espoir de réduire la charge de trafic sur le réseau, ce qui peut entrainer une perte de paquets. En plus de la taille de l'image, il existe un autre facteur qui est la quantification, qui affecte les valeurs des coefficients dans le tableau des gradeurs. La compression ajuste le rapport de données d'image, ajuste ainsi la quantité de données à envoyer dans le réseau, et affecte ainsi la consommation d'énergie sans fil et le temps de réponse du nœud [35].

D'autre part, on constate que certains chercheurs ont tenté de profiter du système de suivi oculaire miniature embarqué afin d'extraire une partie des algorithmes utilisés pour compresser les images. Ce travail vise a proposer plusieurs approches de conception et de mise en œuvre du traitement d'image et des algorithmes de compression pour un système de suivi oculaire miniature embarqué EyeDee™ contenant Weetsy™ système filaire/sans fil portable (cadre Weetsy™ et carte. Dans le travail suivant, les chercheurs ont proposé d'utiliser l'analyse des correspondances et la haute précision pour la compression d'image distribuée des réseaux de capteurs sans fil [38].

Et dans l'article intitulé « transmission d'images sur les réseaux de capteurs sans fil sous la contrainte de l'énergie. ». Les chercheurs ont proposé un algorithme de compression d'image original qui est très peu calculatoire est peu gourmand en énergie et qui assure aussi que l'image compressée a un maximum de résistance aux erreurs de transmission. L'objectif c'était de transmettre des images compressées sur un protocole de communication non fiable (acceptée toutes les pertes des paquets) mais l'image reste quand même acceptable même pour des taux de perte élevée pour que cela soit possible, ils ont été orientés vers un algorithme de compression par blocs (l'image original va être découpé en blocs de pixels qui vont être compressé un à un de manière indépendante). ICES c'est la méthode de compression avec perte, choisit dans cet article, est basée sur la suppression de pixel un par bloc. Cette méthode est utilisée pour réduire le nombre de bits nécessaire pour représenter un bloc de l'image. Pour évaluer les performances de l'algorithme de compression ils ont sélectionné un ensemble d'images test monochromes de taille 512*512 pixels et pour les évaluations portant sur la qualité d'image ils ont programmé l'algorithme ICES sur le langage c++ et le compilateur GNU g++. Le choix de tout petit bloc un des avantages des bons résultats par ce que l'efficacité de la stratégie de l'image est intimement liée à la taille des blocs [36].

Sur le même principe, nous ne constatons que les chercheurs dans l'article suivant « Compression d'image distribuée pour réseaux de capteurs utilisant l'analyse des correspondances et la super-résolution ». Les chercheurs ont décrit une méthode de codage distribué des images capturées par des capteurs équipés de caméras dans un réseau de capteur sans fil, dans le but que les informations mutuelles entre les caméras adjacentes sont exploitées pour alléger la charge de communication pour les nœuds capteurs qui souhaitent transmettre leurs images. Ils se sont intéressés à une nouvelle

méthode de compression d'image distribuée basée sur les points de caractéristiques, l'analyse de correspondance et les techniques de super-résolution d'images. Ils ont démontré qu'il existe un gain de codage exploité par un tel algorithme, encourageant le développement ultérieur de la méthode [33].

Dans le même contexte que les deux articles précédents dans l'article suivant, un schéma EDM a été développé qui est un schéma de mesure de divergence basé sur l'entropie pour prédire l'efficacité de compression de l'exécution d'un codage conjoint contenant plusieurs caméras corrélées. Ils ont pris les résultats de ce schéma pour proposer un protocole de codage multi-cluster qui est entièrement distribué.il fournit une approximation de la solution optimale, il construit une hiérarchique de codage, Le DCMP garantit aussi que chaque caméra de capteur est couverte par au moins deux clusters de codage différents. Les résultats de ce travail ont montré que l'EDM peut prédire efficacement l'efficacité du codage conjoint par des normes de codage pratiques. Cette compression proposée peut réduire le taux de codage allant jusqu'à 10% à 23% par rapport au codage indépendant classique [34].

Tableau comparatif :

Articles	L'idée clé	Les avantages	Les limitations
Compression D'images Distribuée pour RCSFs [33].	Une méthode de codage distribuée des images capturées est basée sur les points de caractéristiques, l'analyse de correspondance et les techniques de super-résolution d'images.	Alléger la charge de communication pour les nœuds capteurs qui souhaitent transmettre leurs images. Réduire la charge de la transmission sur le capteur principale et augmenter ainsi sa durée de vie opérationnelle	L'existence d'un gain de codage a exploité par un tel algorithme. Cela encourage un développement ultérieur de la méthode.
Un Cadre de Compression D'images pour les RCSFs [34].	Un schéma de mesure de divergence basé sur l'entropie. Proposer à travers les résultats de schéma EDM un protocole de codage multi-cluster.	Réduction du taux de codage allant jusqu'à 10% à 23%.	Chercher toujours à trouver des nouveaux algorithmes pour compresser et transmettre des images dans un réseau de capteur sans fil avec une bonne qualité d'image et avec moins de consommation d'énergie.
Solution de Compression D'images Matérielle [35].	Cet article traite les réseaux de capteurs d'image ou WCSNs, orientés vers le suivi et le comptage, et cet article traite le problème du grand espace pour représenter l'image et la grande consommation d'énergie lors de la compression au niveau du nœud. Ce travail présente également des solutions non conventionnelles ou il ne se concentre pas nécessairement sur les algorithmes de compression, mais plutôt sur la recherche de	Ce travail représente une solution pratique et non conventionnelle, car les propriétés physiques des nœuds qui composent les réseaux de capteurs sans fil sont limités, comme la mémoire et l'énergie ont toujours été un obstacle à la mise en œuvre de taches plus complexes sur ces nœuds. Ce travail ouvre la porte à la recherche dans le domaine de l'ingénierie des nœuds capteurs afin de les développer et de les améliorer.	Penser à trouver des solutions au niveau matériel n'est pas toujours possible en revenant à la condition liée à la taille du nœud capteur, qui se mesure en millimètres et donc la mémoire, le processeur et le niveau de stockage d'énergie doivent être limités, et aussi un changement au niveau de l'appareil entraine une augmentation du cout du nœud capteur.

	solutions au niveau matériel.		
Transmission D'images sur les RCSFs[36].	Un algorithme de compression par blocs, proposer une méthode de compression avec perte (ICES) et un algorithme de suppression de pixels auto-adaptatif. Le couplage de l'algorithme de compression avec une méthode d'entrelacement de pixels (AT).	Avec le couplage de ces deux ils ont obtenue des images de bonne qualité même avec des taux de perte de paquets de 40%. Le choix de tous petits blocs, c'est l'un des avantages pour avoir de bons résultats.	Chercher toujours à trouver des nouveaux algorithmes pour compresser et transmettre des images dans un réseau de capteur sans fil avec une bonne qualité d'image et avec moins de consommation d'énergie.
Une Enquêtes sur les Algorithmes de Compression D'images pour les Réseaux de Capteur Sans Fil[37].	Ce travail traite les types d'algorithme de compression utilisés dans les réseaux de capteurs optiques, allant des normes traditionnelles telles que JPEG aux méthodes de compression modernes, et discute les avantages et les inconvénients de chaque algorithme, et leurs effets sur les réseaux VSNs en termes d'énergie et de mémoire.	Cet article est très utile car il donne un avis complet sur tous les algorithmes de compression utilisés dans les réseaux de capteurs sans fil, et passe en revue leurs avantages, inconvénients et étapes de développement.	Les algorithmes de compression restent la principale préoccupation dans les réseaux de capteurs sans fil impliqués dans le capteur et le traitement d'image, car il est très difficile de trouver un algorithme de compression idéal qui nous donne une bonne qualité d'image compressé et consomme moins d'énergie et nécessite peu de mémoire pour le stockage.
Conception et Mise en œuvre du Traitement et de la Compression D'images [38].	Cette thèse étudie les mouvements oculaires lors des activités quotidiennes de l'être humain. Des dispositifs spéciaux collectent ces données ou images et sont soumis à des traitements tels que la compression et autres. Ce travail porte sur les algorithmes de	Ce travail aideà identifier les algorithmes de compression utilisés dans ce type d'études qui traitent les images provenant de l'œil humain et aide à comprendre leur fonctionnement, ce qui ouvre la porte pour bénéficier de ces algorithme dans les réseaux de capteurs sans fil.	Ces algorithmes ne conviennent pas aux réseaux de capteurs sans fil car les images après avoir été capturées par certains appareils sont envoyées à l'ordinateur. Ce dernier effectue le processus de traitement, qui a bien sûr plus de capacités physiques que le nœud capteur, qui est

	compression utilisés.		fondamentalement limité dans des capacités telles que l'énergie et la mémoire.

Tableau 2.2 : vue d'ensemble de certains des algorithmes proposés pour améliorer la qualité de la compression d'image dans RCSF

2.9 Conclusion :

Récemment, le nombre de recherches dans le domaine des réseaux de capteurs sans fil a augmenté, en raison de son importance. Toutes ces recherches s'accordent sur le problème de l'énergie lors de l'échange de données entre les nœuds. Trouver de nouveaux algorithmes de compression qui fournissent des données de hautes qualité et maintiennent une durée de vie plus longue du réseau est inévitable, et a cette fin. Dans le chapitre suivant, nous avons essayé d'écrire un nouveau format pour l'algorithme JPEG, en utilisant deux types d'encodeurs, le changement n'incluant que la dernière étape du processus de compression des données.

Chapitre 3 : Approche proposée pour la compression d’images

3.1 Introduction :

Les images naturelle représentent de grands volumes de données, voilà pourquoi Les centres de recherches en informatique dépensent de nombreuses heures sur des algorithmes de compression. Avec la compression on peut tous éliminer les informations redondantes, les informations inintéressantes, donc la compression est vitale dès lors que l'on veut transmettre les images sur les réseaux de capteurs sans fil.

Il existe dans nos jours plusieurs algorithmes de compression d'images, avec perte, sans perte. Dans notre mémoire on a travaillé sur l'algorithme de compression JPEG qui est un algorithme très utilisée.

3.2 Technique de base de la compression d'images :

La compression des images c'est l'un des méthodes les plus utilisées pour réduire la taille des images transmis dans un réseau de capteur sans fil et donc minimiser l'énergie consommée.

Plusieurs algorithmes de compression d'images existant (JPEG, JPEG2000,…, ect), on a travaillé sur l'algorithme JPEG où on a développé un schéma dont les éléments sont couramment utilisés dans la communauté scientifique de la compression d'image. Le JPEG c'est l'un des formats les plus complexes, son étude complète nécessite de solides bases mathématiques, il offre des taux de compression plus qu'intéressants.

Le JPEG est la norme internationale relative à la compression d'images fixes, la méthode de compression est « avec perte ».

Le schéma de principe d'un encodeur d'image souvent utilisé pour décrire le fonctionnement des algorithmes de compressions est celui présenté dans la FIGURE (3.1). La première étape de l'algorithme JPEG est une transformation qui a pour but de changer l'espace de représentation, l'étape de la quantification qui permet une réduction de nombre de valeurs représentées, et la dernière étape c'est le codage qui permet de passer d'une représentation de données vers une autre sans perte sur les valeurs quantifiées.

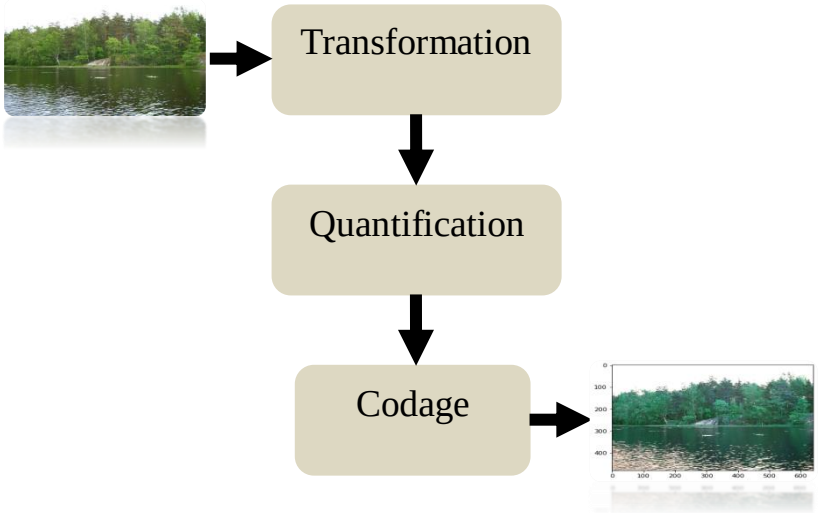


Figure 3.1 : Les étapes de l'algorithme JPEG

Dans les sections suivantes nous allons citer les différentes étapes détaillées de notre algorithme, notre objectif est de fournir une chaine de compression adaptée aux contraintes des réseaux de capteurs sans fil (faible coût calculatoire, faible coût de stockage).

3.2.1 Transformation :

Dans cette section on a parlé de la première étape de notre algorithme de compression.

Découpage en blocs :

A ce stade, l'image est découpée en blocs de 8*8 pixels, ou chaque bloc représente un tableau qui sera ensuite compressé séparément. Un des avantages de cette technique est que le traitement sera rapide, car le tableau ne peut contenir que 64 valeurs, mais en revanche, la somme des matrices à traiter est extrêmement grande.

Il est à noter que lorsque les dimensions de l'image ne sont pas des multiples de 8, il faut ajouter des pixels supplémentaires pour obtenir les dimensions requises. Les pixels ajoutés peuvent être noirs ou les pixels existants peuvent être copiés. Ce processus génère de petits défauts de compression sur le bord de l'image, mais la technique de duplication de pixels réduit ces défauts, mais elle est longue à calculer [1].

3.2.1.1 Techniques de décorrélation intra-images :

Une grande partie de l'information contenue dans une image est redondante, puisque les images sont un ensemble de pixels qui sont généralement corrélées, Cette étape de décorrélation vise à supprimer cette redondance.

Les techniques de décorrélation intègre des méthodes de transformation qui figurent parmi les techniques les plus connues et les plus employés, elles n'agissent pas directement sur l'image numérique mais sur le domaine de sa transformée. Elles permettent d'obtenir des taux de compression élevés tout en conservant une bonne qualité de l'image.

Ces méthodes sont utilisées par des standards internationaux pour le codage des images fixes comme JPEG et JPEG2000 et vidéo.

Dans cette section, nous comparons les deux approches les plus utilisées, la transformée en cosinus discrète(DCT) et la transformé en ondelettes(DWT).

La transformée en cosinus discrète(DCT) :

La DCT est une transformation mathématique complexe qui a pour but de transformer le domaine de représentation d'une image d'un domaine spatial en une représentation équivalente dans le domaine fréquentiel ce qui permettra de trier efficacement l'ensemble des données de l'image, permet aussi de supprimer certaines données sensibles à notre œil humain.

L'objectif du calcul de la DCT est d'éliminer les hautes fréquences et pour réaliser ça il faut séparer les bases fréquences et les hautes fréquences présentes dans l'image : deviser l'image en blocs, et appliquer aux chaque pixels de bloc sa répartition spectrale par DCT.

La DCT travaillent sur un signal discret, et elle est adapté pour décrire des signaux

stationnaires, elle range une grande partie de l'énergie du signal dans les bases fréquences.

Pour l'analyse d'un signal bidimensionnel (2D) comme les images, comme notre cas on a besoin d'une version 2D de la DCT, où l'image est décomposée en blocs afin de transformer chaque bloc indépendamment.

La DCT-2D est effectué sur une matrice carrée N*N de pixels et donne une matrice carrée N*N de coefficients fréquentiels comme montre la Figure (3.2), la définition de la 2-D DCT est donnée dans l'équation (eq3.1) :

$$\mathrm{F(u,v)} = \frac{2}{N} C(u)C(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} I(i,j) \cos\left(\frac{\pi(2i+1)u}{2N}\right) \cos\left(\frac{\pi(2j+1)v}{2N}\right) \qquad (3.1)$$

Où :

$$\mathrm{C(x)} = \begin{cases} 1, & si\ x \neq 0 \\ \frac{1}{\sqrt{2}}, & si\ x = 0 \end{cases}$$

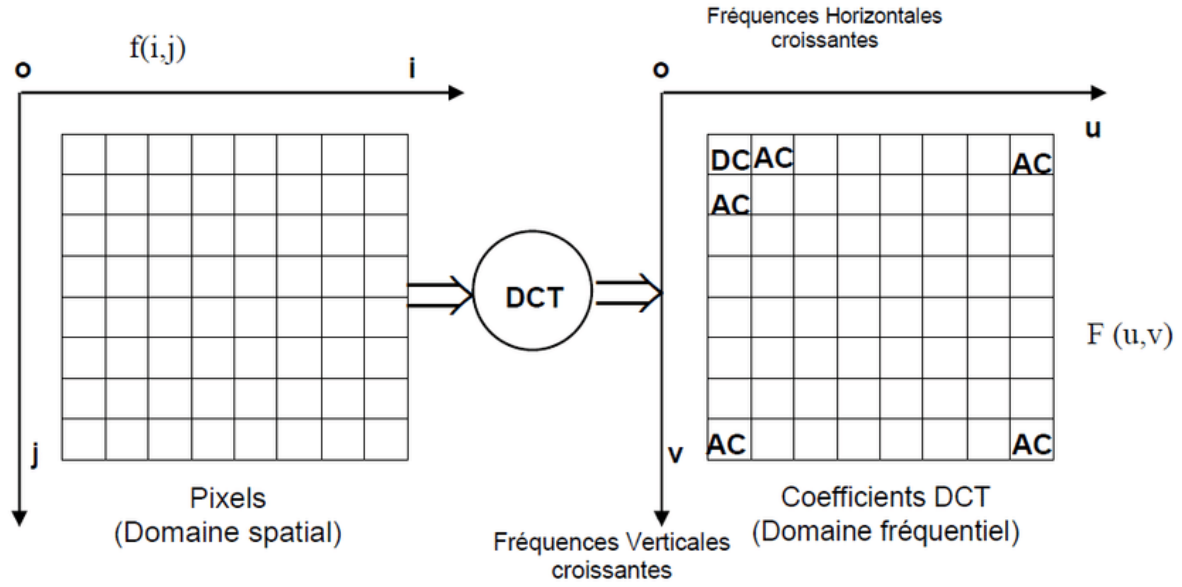


Figure 3.2 : Représentation de la DCT d'un bloc de 8 x 8 pixels

Dans la plupart de cas, on devise l'image en blocs 8*8 et on applique cette transformation sur l'image.

Les bases fréquences se trouvent en haut a gauche de la matrice, et les hautes fréquences en bas a droite.

la DCT est utilisé souvent dans les algorithmes de tatouage des images et la plupart des algorithmes basé sur cette transformée cachent le message secret dans les moyennes fréquences [11].

Parmi les avantages de la DCT, elle possède un ensemble de fonctions de base stable(les fonctions cosinus).

-elle offre actuellement le meilleur compromis puissance -complexité.

-la complexité des équipements augmente brutalement lorsqu'on cherche à descendre en dessous d'un seuil situé vers 50 MBPS.

-la DCT permet d'atteindre des performances meilleures à celles des techniques différentielles (10 à 30 MBPP en télévision) et de très bas débit pour des images de communications de qualité réduite (de 2 MBPS à 64 KBPS, selon la qualité recherché) [11].

La transformée en ondelettes discrète : DWT

La transformé en ondelette discrètes est une opération qui décompose un signal (une série d'échantillons numériques) en deux parties par projection sur un filtre passe-bas L et un filtre passe-haut H. Elle a été développée dans les années 80 et on peut la considérer comme une extension de l'analyse de Fourier.

Une transformé en ondelette 2D possède trois fonctions d'ondelettes qui sont généré à partir d'une fonction mère, la première fonction permet de récupérer les détails horizontaux, la deuxième récupère les détails verticaux et la troisième les détails diagonaux.

Le principe de la décomposition de DWT : en appliquant d'abord les filtres L et H sur les échantillons ligne par ligne puis en réappliquant L et H colonne par colonne cette fois-ci sur les échantillons résultants. Nous obtenons ainsi quartes sous bandes (sous-images) LL, LH, HL, HH comme présenté sur la Figure (3.3).

L'image est analysée dans différentes directions : les détails horizontaux sont contenus dans LH, les détails verticaux dans HL, les détails diagonaux dans HH et le bloc LL contient une image de plus base résolution (4 fois moins de pixels que l'image initiale).

LL1	HL1
LH1	HH1

Figure 3.3 : Principe de la transformée en ondelettes discrète

Parmi les avantages de DWT, Elle donne de très bons résultats, même avec des taux de compression élevée (plus de 90%).

-Une image compresser par la DWT peut être décompressé de deux manières différentes :

-Sa taille et fixe mais sa résolution augmente progressivement.

-Sa résolution est fixe mais sa taille augmente progressivement.

- Elle ne s'entraine pas des effets de mosaïque [11].

DCT vs DWT :

Chacune des deux méthodes présente des avantages comme on a cité mais dans notre cas il faut qu'on prend en considération les critères les plus importants, La consommation d'énergie

(le coût de transmission via les réseaux de capteurs sans fil), le degré de complexité (au niveau de l'implémentation).

Parmi les raisons pour lequel on a choisi de travailler avec la DCT c'est sa simplicité par rapport à la DWT même il y'a plusieurs travaux ont utilisé des algorithmes de compression, basées sur la DCT pour la transmission des images à travers les réseaux de capteurs sans fil.

La DWT offre une meilleure qualité d'image par rapport a la DCT, mais elle est moins adapté pour les applications des réseaux de capteur sans fil puisque elle nécessite davantage des ressources mémoire et consomme plus d'énergie ce qui nous n'arrange pas puisque les nœuds capteurs sont très limités en capacité de calcul et de mémoire, c'est pour cela nous s'intéresserons dans notre travail uniquement à la transformé en cosinus discrète.

3.2.2 Quantification :

A ce stade, l'image est compressée, c'est-à-dire que l'image perd ses informations, et donc sa qualité visuelle, mais en même temps c'est une étape qui fait gagner beaucoup de place, et au contraire DCT ne se comprime pas, DCT crée une matrice de 8*8 nombres pour chaque bloc. La quantification divise chaque matrice par une autre matrice, appelée matrice de quantification choisie par le programmeur, et se compose de 8*8 coefficients [2].

Le calcul suivant permettant la quantification (eq3.2) **:**

$$F^*(u,v) = \left[\frac{F(u,v)+\left[\frac{Q(u,v)}{2}\right]}{Q(u,v)}\right] \cong \left(\frac{F(u,v)}{Q(u,v)}\right) \qquad (3.2)$$

Pour la quantification inverse (eq3.3) :

$$\hat{F}(u,v) = F^*(u,v) \cdot Q(u,v) \qquad (3.3)$$

A ce stade également, les fréquences auxquelles l'œil humain est insensible s'affaiblissent lorsque les coefficients sont réduits à zéro notamment en bas droit de la matrice où se trouvent les hautes fréquences. Certaines informations de base existent dans le coin supérieur gauche sont conservées afin de représenter le bloc.

L'avantage est que lors de l'encodage du résultat dans le fichier, une longue chaine de zéro nécessitera très peu d'espace, mais si la quantification est trop élevée, il y aura peu des coefficients non nuls pour représenter fidèlement le bloc, par conséquent, la division en blocs devient visible à l'écran et l'image apparait pixélisée [2].

Dans l'exemple suivant (Figure 3.4), nous avons pris la matrice de quantification suivante :

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Figure 3.4 : Matrice représentant le niveau de quantification

Cela donne la matrice des fréquences quantifiée suivante (Figure 3.5) :

$$F^* = \begin{bmatrix} 79 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3.5 : Matrice quantifiée

A ce stade, 64 valeurs pour chaque bloc sont écrites les unes après les autres en suivant un chemin zigzag comme dans la Figure (3.6), où les basses fréquences sont situées en haut à gauche du bloc et donc il est lu en premier puis les hautes fréquences sont lues. Ce parcours se termine par une grande série de zéro, où les grandes fréquences sont complètement supprimées, et il ne reste donc que quelques chiffres, par exemple (-78,23,-65,0,2,23), ces chiffres représentent un bloc entier composé de 64 valeurs.

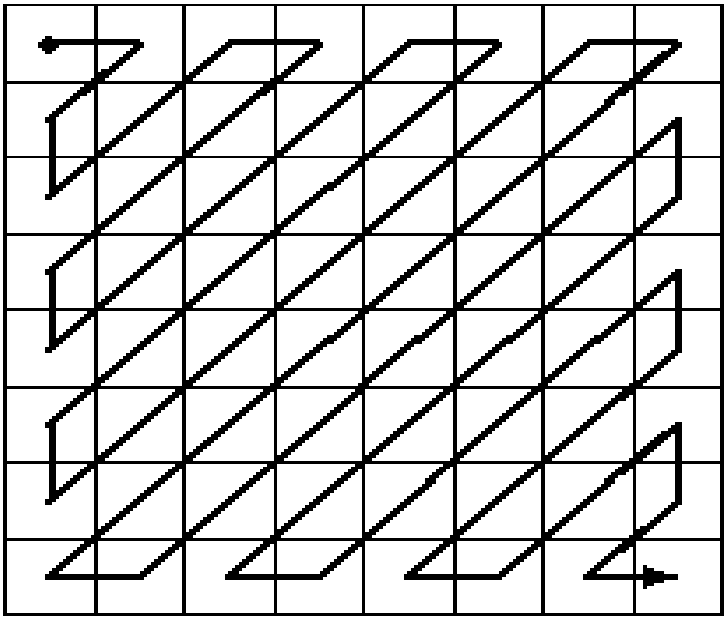


Figure 3.6 : Le parcours en zigzag

Dans certains cas, nous pouvons obtenir un résultat plus long comme (8,-6,34,83,93,0,0,0,0,0,0,4-3,57,-87), ou les zéro au milieu prennent beaucoup de place. Cet algorithme suit l'abréviation d'écrire des nombres dans la chaine, par exemple, au lieu d'écrire AAAABBCCC, cet algorithme suggère d'écrire 4A 2B 3C [3].

3.2.3 Codage :

On parle dans cette section des deux types de codage.

Le codage de Huffman :

Le codage de Huffman est un algorithme de compression de données sans perte est optimisé pour les chiffrements de symboles mais il ne permet pas d'obtenir les meilleurs taux de compression.

Le principe de chiffrement de Huffman est de créer un arbre composé d'un ensemble de nœuds comme montre la figure (3.7), d'abord le nombre d'occurrences de chaque caractère est déterminé. Par exemple, si l'on prend comme exemple la phrase suivante : 'Huffman', on constate ici que chacune des lettres : ' h','u','m','a','n', est répétée une seul fois, tandis que la lettre 'f' est répétée deux fois. Chacune des lettres précédentes forme une des feuilles de l'arbre, chaque feuille portant son poids, qui représente le nombre de fois que la lettre est répétée. Ce poids détermine l'emplacement de la feuille de l'arbre par rapport au nœud qui le précède : si le poids de la feuille est supérieur au poids du nœud précédent, la feuille est placée à sa droite, et si son poids est inférieur il est placé à gauche des nœuds.

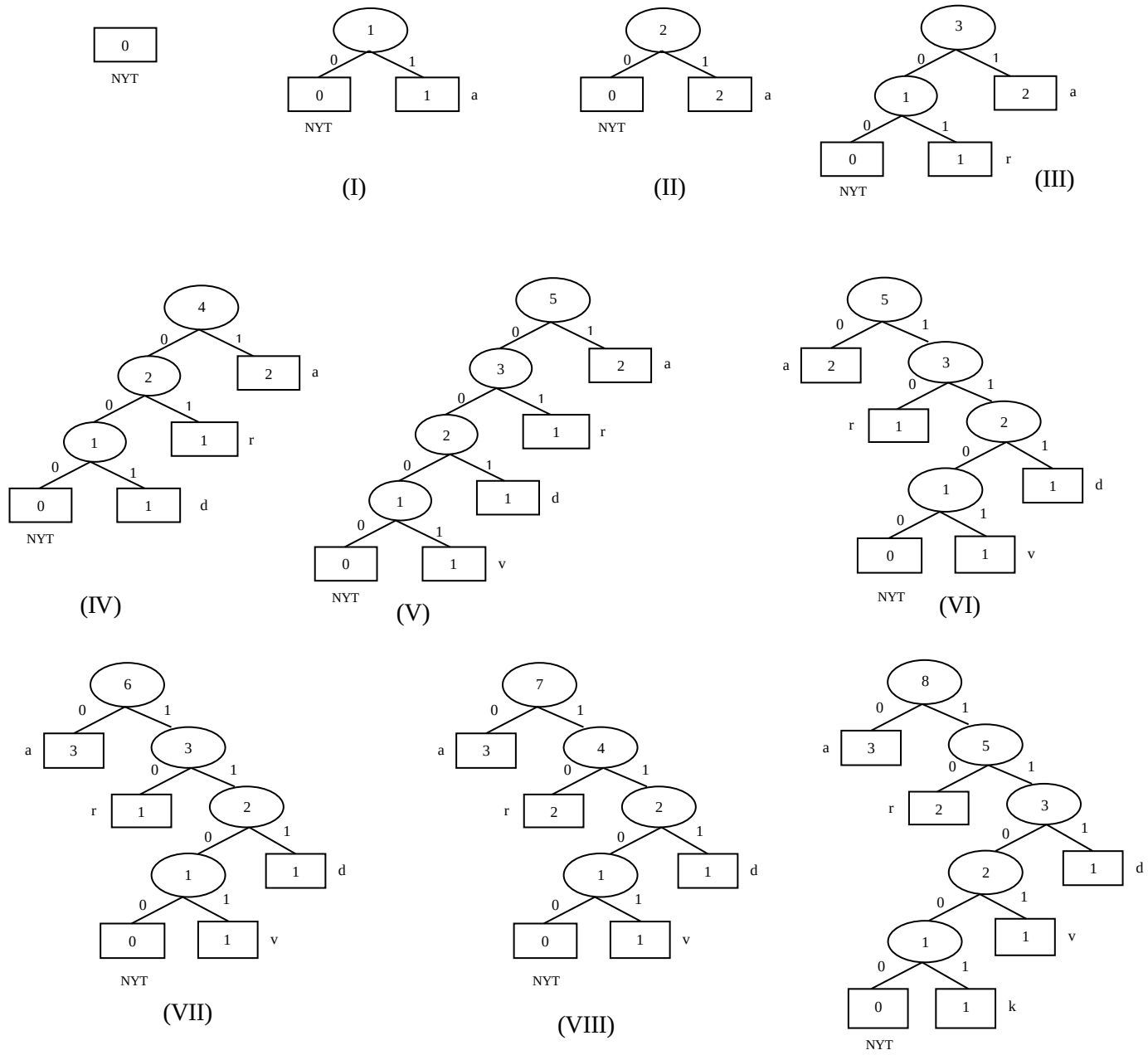


Figure 3.7 : L'arbre binaire

Afin d'obtenir le code binaire de chaque caractère, on remonte l'arbre de racine en feuille et on ajoute a chaque fois 0 ou 1 au code, selon la branche. Il est très nécessaire de partir de la racine pour obtenir les codes binaires partir des feuilles va provoquer une confusion lors du décodage.

Il existe trois types différents d'algorithme de Huffman, où chaque type définit une méthode pour construire l'arbre :

- **Le codage de Huffman adaptative :**Bien que le codage Huffman traditionnel soit bon et utile, il existe des applications qui nécessite toujours de connaitre à l' avance les données qui seront encodée, par exemple le transfert d'un flux vidéo en direct. Dans la méthode de Huffman traditionnelle, il n'est pas possible de savoir exactement ce qui sera transféré à l'avance, et comme solution la méthode de Huffman adaptative est utilisée.

Le codage de Huffman adaptative construit un arbre qui donne un schéma de codage binaire idéal comme dans le codage Huffman traditionnel. La seule différence est que son entrée est prétraitée avant de commencer à être encodée. En utilisant l'algorithme FGK (Faller-Gallager-Knuth), l'arbre est construit comme dans la notation traditionnelle de Huffman où les feuilles sont pour les lettres et les nœuds internes pour les pseudo-caractères, la fréquence du nœud interne sera égale à la somme des fréquences de ses enfants.

Malgré toutes les similitudes entre l'arbre Huffman traditionnel et l'arbre FGH, il existe deux différences principales :

Tous les nœuds de l'arborescence doivent avoir un frère où les nœuds peuvent être répertoriés de gauche à droite et de bas en haut dans l'ordre incrémentiel. Une autre différence est l'utilisation de nœuds vides, dans un arbre Huffman traditionnel, l'arbre est construit de bas en haut, de la feuille à la racine, à l'aide de notre table de fréquence. Dans la notation adaptative de Huffman, l'entrée est lue et l'arbre est construit en même temps sans compter les premières fréquences, donc l'arbre FGk doit être construit de haut en bas de la racine à la feuille. Le nœud vide sert de fratrie pour contenir un nouveau personnage qui sera ajouté, le but étant de conserver les biens de la fratrie [5].

- **Statique :** Dans ce type d'encodage, chaque octet a un code prédéfini par le programme. Par exemple, si nous voulons encoder le texte suivant : 'aabbaab ', nous allons d'abord lire le texte une fois de gauche à droite, afin de regrouper les caractères qui apparaissent dans le texte avec leurs itérations appelées poids, dans notre cas nous trouverons que la lettre 'a' se répète quatre fois, la lettre 'b' deux fois, et la lettre 'c'une fois. Deuxièmement, nous construisons un arbre binaire pondéré en fonction de ses itérations, où chaque nœud a un enfant gauche et un enfant droit et ils peuvent être vides, et chaque nœud a un poids et ses feuilles représentent l'alphabet du texte. Pour obtenir l'arbre complet, il faut suivre l'algorithme suivant : le départ se fait à partir de l'ensemble de nœuds isolés correspondant aux caractères, les deux caractères de moindre poids ont le même parent, les deux nœuds correspondant à ces caractères ont supprimés de la liste des nœuds à traiter et à remplacer par leur père auquel affecté d'un poids égal à la somme des poids des fils, puis nous commençons un nouvel ensemble de nœuds jusqu'à ce qu'il n'y ait qu'un seul nœud, qui deviendra la racine[6].

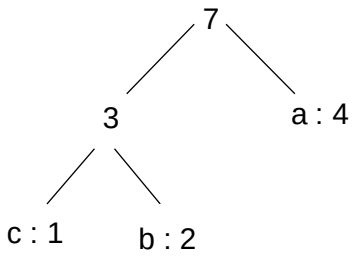


Figure 3.8 : L'arbre binaire pondéré

Chaque lettre de l'alphabet dans l'arbre a une notation binaire de 1 et 0 qui indique le chemin de la racine a la feuille ou 1 représente l'enfant droit et 0 représente l'enfant gauche, comme montre la Figure (3.10).

a	1
b	01
c	00

Figure 3.9 : L'encodage

- **Codage du Huffman semi-adaptatif :**

L'ensemble du fichier est lu afin de compter les occurrences de chaque octet, puis un arbre est créé à partir des poids de chaque octet, l'arbre reste inchangé jusqu'à la fin de la compression. Cette compression se traduit par un gain binaire supérieur ou égal au codage constant de huffman, mais pour décompresser il faut envoyer l'arbre, ce qui annulera le gain obtenu [7].

Codage RLE (Run length Encoded) :

RLE est une méthode de compression de données sans perte basée sur des itérations successives d'éléments, seules comptent la longueur et la taille des occurrences dans le texte. Tout d'abord, le texte à coder est balayé pour trouver une séquence de caractères correspondants, puis le caractère et le nombre d'itérations dans la séquence sont notés. Au lieu d'envoyer le texte suivant 'FFFFFRRRRTTTSSH', on envoie seulement : 'F5R4T3S2H2', dix caractères au lieu de 15 caractères, ainsi le taux de compression est de 33%. Cette méthode produit une compression uniquement s'il y a de nombreuses itérations. Si la donnée est binaire c'est-à-dire 1 ou 0, RLE peut être utilisé sans indiquer la lettre, les chiffres suffisent, le premier chiffre indique 0 ou 1, puis en alternance. Par exemple : 000011100 se code 4, 3,2. Mais si le texte ne contient que des chiffres, un séparateur doit être utilisé pour faire la distinction entre le caractère et le nombre de fois qu'il est répété. Par exemple : 55555533333377788888 se code : 6-5,6-3,3-7,5-8.

Huffman vs RLE :

Les facteurs	**Codage Huffman**	**Codage RLE**
Les avantages	• Il est facile à mettre en œuvre. • Produire une compression d'image sans perte.	• Il est facile a mettre en œuvre. • C'est une bonne alternative pour un algorithme de compression complexe
La rapidité	Rapide à exécuter	Rapide à exécuter
Application	ZIP, ARJ, MPEG, JPEG	TIFF, BMP, PDF
Les inconvénients	• Relativement lent. • Dépende du modèle statistique des données.	• Il ne peut pas atteindre les taux de compression plus élevés par rapport à

	• Le décodage est difficile en raison des différentes longueurs de code.	d'autres méthodes de compression élevées.

Tableau 3.1 : Comparaison entre les codages Huffman et RLE

3.3 Conclusion :

Aujourd'hui, il y'a beaucoup d'efforts dans plusieurs travaux pour adapter les algorithmes de compression d'images comme le JPEG, JPEG2000 aux contraintes des réseaux de capteurs sans fil. Dans ce chapitre, nous avons présenté les différentes étapes de la chaine de compression d'images. Nous avons parlé des deux types de l'étape de transformation, la DCT et la DWT avec une comparaison entre eux.

Ensuite nous avons évoqué l'étape de la quantification, et à la fin les différents types du codage. Dans le chapitre suivant, nous allons parler de l'évaluation de notre travail avec des résultats des tests.

Chapitre 4 : Implémentation & Expérimentation

4.1 Introduction :

La circulation des images dans les réseaux de capteurs sans fil nécessite beaucoup d'énergie, ce qui réduit la durée de vie du réseau, du fait que les nœuds capteurs disposent de ressources limitées, par exemple l'unité de traitement et autres. Et ici apparait l'importance des algorithmes de compression pour résoudre ce probleme. Dans le travail suivant, nous présentons deux types d'algorithmes et les comparons ainsi que leur efficacité a réduire la consommation d'énergie et a préserver l'information contenue dans l'image.

Dans ce chapitre, on va parler des logiciels utilisé ainsi que toute notre implémentation et les résultats des tests qu'on a trouvés.

4.2 Les outils d'implémentation utilisés :

Dans ce qui suit les outils que nous avons utilisé dans notre travail.

4.2.1Le matériel :

Nous avons utilisé un laptop de marque Acer avec les capacités suivantes :
Intel celeran CPU 3215U@ (1.70 GHZ).
Windows 8.1 professional.
4.00 GO de mémoire installé (RAM).
Type de system : système d'exploitation 64 bits.

4.2.2 Les logiciels :

Python :

Le langage de programmation utilisé est python version 3.7, Python est un langage simple, performant et associé à plusieurs milliers de bibliothèques. Il représente l'un des langages « open sources » les plus interactifs et le plus employé par les informaticiens avec une importante et riche documentation en ligne. . Ce langage s'est propulsé en tête de la gestion d'infrastructure, d'analyse de données ou dans le domaine du développement de logiciels.

Les bibliothèques utilisées :

Les principales bibliothèques utilisées sont : CV2, matplotlib, numpy, scipy, future…

Figure 4.1 : Logo python

4.3. Généralité sur la simulation :

La simulation est la modélisation informatique d'un système quelconque, avec une représentation de toutes les entités de ce système, leurs interactions ainsi que leurs comportements, elle met à la disposition de l'utilisateur un environnement d'expérimentation dont on peut faire varier les paramètres.

Les simulateurs de réseau existant :

Il existe plusieurs simulateurs de réseau tel que : NS2, OMNET++, JSIM, GLOMOSIM, OPNET…etc.

On va parler de notre simulateur OMNET++ et les difficultés qu'on a trouvées.

4.3.1 Simulateur OMNET++ IDE (Integrated Development Environment) :

c'est un environnement open source qui fournit des outils pour la création et la configuration des modèles de réseaux avec ou sans fil, il est basé sur des composants qui sont programmé en c++,puis assemblés en composants et modèles plus volumineux à l'aide d'un langage de description de topologie de haut niveau (NED).

Ce simulateur est composé aussi de l'IDE de simulation basé sur la plateforme Eclipse, d'interface graphique d'exécution de simulation interactive (Qtenv) et d'interface de ligne de commande pour l'exécution de la simulation (Cmdenv).

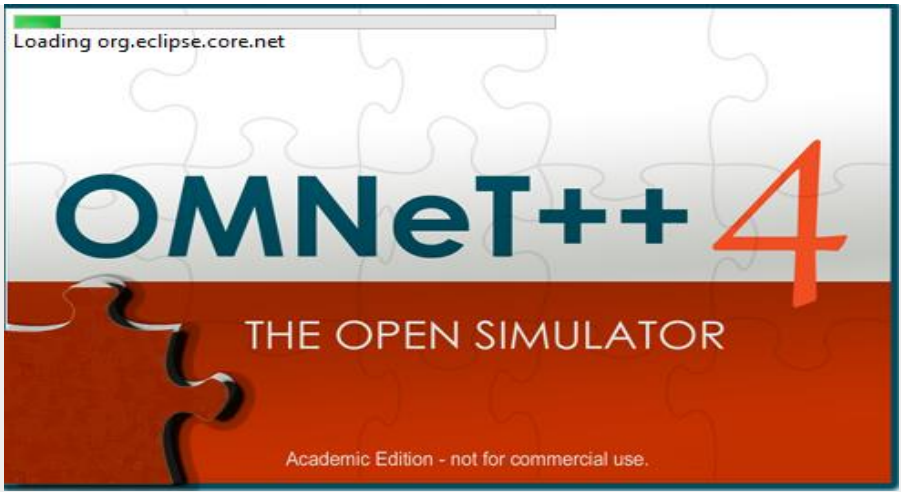
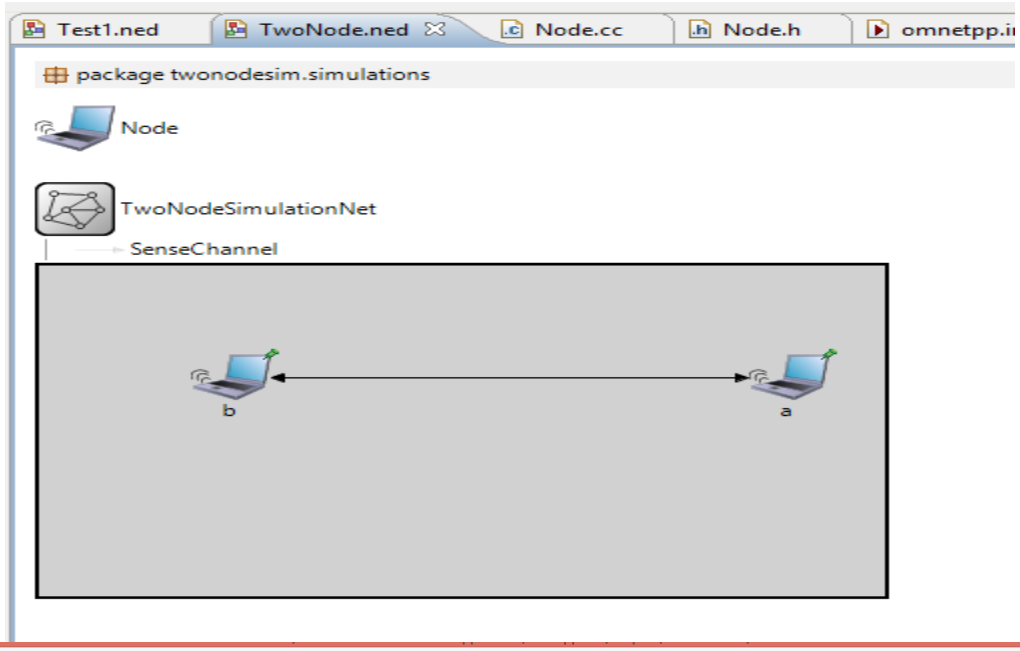
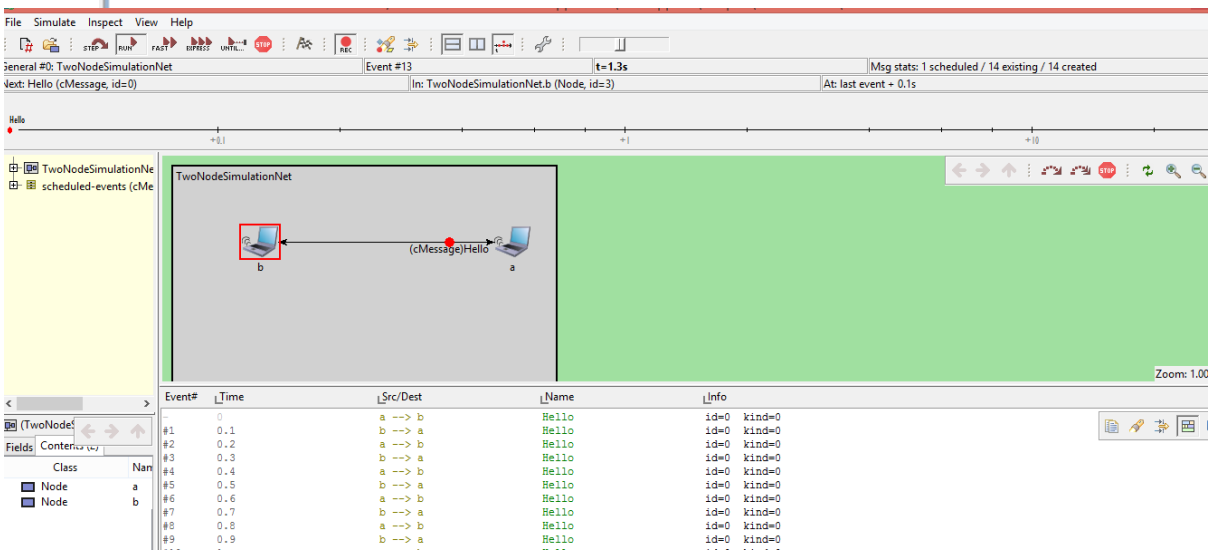


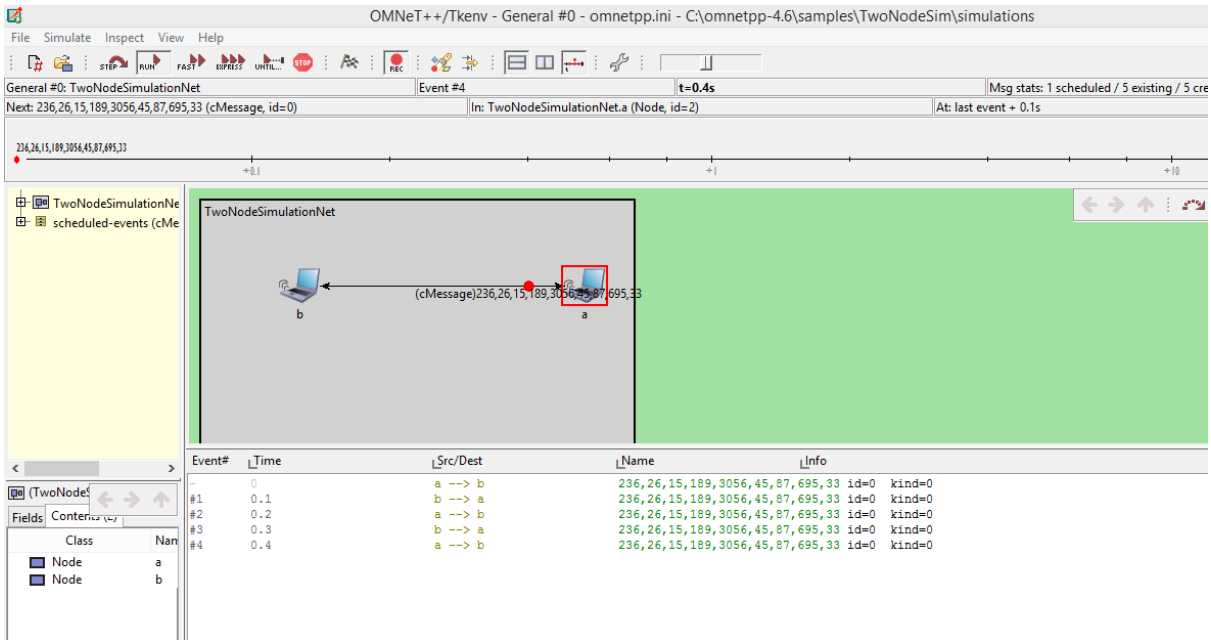
Figure4.2 : Lancement du simulateur OMNET++

4.4 Le problème rencontré :

Le problème qu'on a trouvé avec l'OMNET est le problème d'envoi de message entre les nœuds, on a réalisé une topologie avec fil avec l'envoi des messages entre les nœuds (des simples messages (Hello), des chiffres, des matrices) mais qu'on a essayé de réaliser ça avec une topologie sans fil ya malheureusement des erreurs, voici quelques résultats réalisé sur OMNET :







Dans notre travail on a voulu faire une topologie sans fil puis commencer d'envoyer entre les nœuds des messages personnalisés (Hello, chiffre..) puis tester l'envoi d'une image (qui est

représenté sous forme d'une matrice) sans compression pour voir la consommation d'énergie de cette image dans un RCSF, puis envoyer une image compressée et voir aussi la consommation d'énergie de cette image et comparer les résultats.

4.5 Résultats de la compression JPEG :

On a travaillé sur deux datasets qui contiennent chacun 30 images, le premier dataset contient des images en couleur et l'autre des images en niveau de gris.

On va présenter les résultats obtenus de l'algorithme de compression JPEG qu'on a fait sur Python, dont on a parlé de tous ces étapes dans le chapitre précédent.

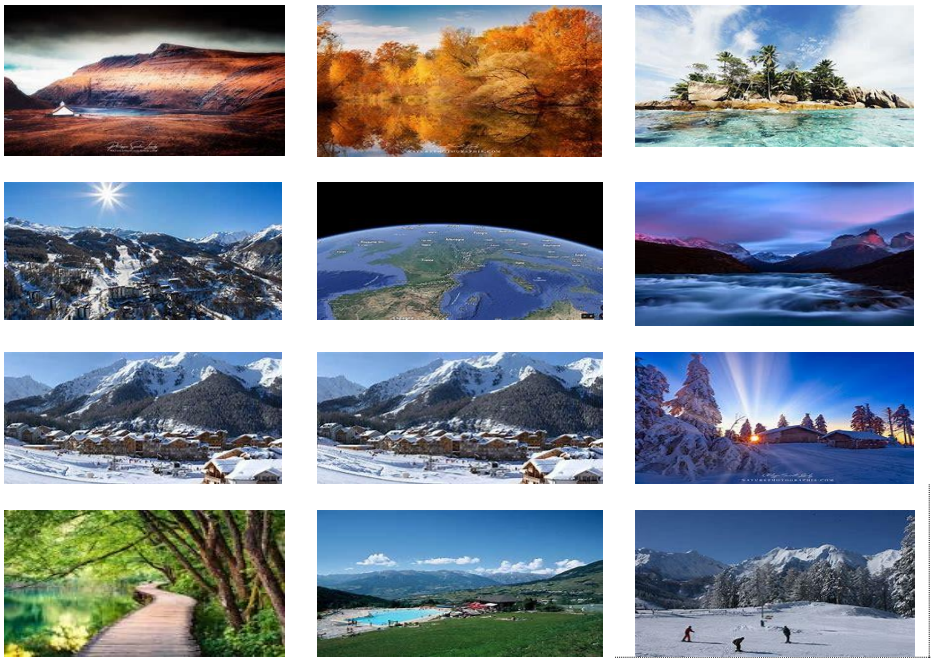


Figure 4.3 (a) : Des images en couleurs avant la compression

Figure 4.3 (b) : Des images en couleurs après la compression avec l'algorithme JPEG basé sur RLE

Figure 4.4 (a) : Des images en niveau de gris avant la compression

Figure 4.4 (b) : Des images en niveau de gris après la compression avec l'algorithme JPEG basé sur RLE

4.6 La modélisation :

Afin d'évaluer l'efficacité des deux algorithmes de compression proposés à réduire la consommation d'énergie dans les réseaux de capteurs sans fil, nous avons simulé un modèle énergétique qui a été présenté en 2015 par les chercheurs : Mohammad Abu Zahad, Mohammad Faraj et Abdu Hai Ali dans un article de recherche intitulé « Le modèle de consommation d'énergie du nœud capteur dans le réseau de nœud capteur sans fil » [39]. La simulation est bien sur concentrée sur la spécificité de ce type de réseau en termes de nombre maximum de paquets de données que peut transmettre par un nœud capteur.

Le modèle énergétique que nous avons simulé prend en compte les paramètres de la couche physique et de la couche MAC au niveau de chaque nœud capteur, contribue également à réduire la puissance de transmission et de communication via les canaux AWGN.

Les principaux composants du nœud capteur pendant le processus de transmission sont : le circuit numérique en bande de base, les analogues analogiques transmis et la puissance de l'amplificateur. En revanche, les principaux composants pendant le processus de réception sont : le circuit de réception analogique et le circuit numérique en bande de base. Ce modèle définit la puissance totale consommée par le nœud lors de l'émission et de la réception par les équations suivantes :

$$\mathbf{Ptx = Pdct+Pact+Pamp= Pto+ (\alpha+1) Pt \quad tel\ que : Pt = Ao\ d\alpha\ Pr} \qquad (4.1)$$

Equation 1 : L'équation pour calculer la puissance d'émission.

$$\mathbf{Prx = Pdcr + Plna + Pacr= Pro} \qquad (4.2)$$

Equation 2 : L'équation pour calculer la puissance de réception.

Dans ce modèle, la consommation électrique moyenne des protocoles MAC est calculée comme illustré à l'équation (3), via les activités de transmission (Ttx) et de réception (Trx), ou Pslp est la puissance gaspillée à l'état inactif.

$$\mathbf{P = TtxPtx + TrxPrx + (1 – Ttx - Trx) Pslp} \qquad (4.3)$$

Equation 3 : L'équation pour calculer la consommation électrique moyenne des protocoles MAC.

Les activités standard d'émission (Ttx) et de réception (Trx) peuvent être calculées comme suit :

$$\mathbf{Ttx = (3Tst + Lsync/Rr + Lrts/Rr + Ldata/Rr)\ 1/Tdata} \qquad (4.4)$$

Equation 4 : L'équation pour calculer l'activité standard d'émission

$$\mathbf{Trx = (Tst + Lcts/Rr)\ 1/Tdata} \qquad (4.5)$$

Equation 5 : L'équation pour calculer l'activité standard de réception.

La table (4.1) représente les valeurs constantes adoptées par les chercheurs pour définir ce modèle énergétique, par exemple la force nécessaire au nœud capteur pour recevoir un paquet de données (Pro) :

Paramètre Type	Paramètre Value	Paramètre Type	Paramètre Value
Pslp	37μw	Omac	10bytes
Pto	15.9mw	Nphy	72bytes
α	3.2	Pro	22.2mw
Tdata	1	Tst	195μsec
Ao	40db	Lsync, Lcts	24bytes
d	50m	Lrts	30bytes
B	100 khz	Ldata	82bytes
Ophy	7bytes	Rr	20kpps

Tableau 4.1 : Des paramètres généraux sur le modèle énergétique.

4.7 Résultats des tests :

Nous avons maintenant un modèle énergétique, que nous utiliserons par la suite comme référence pour comparer les deux algorithmes de compression que nous avons proposés, ce qui est notre objectif ultime dans ce travail. Le premier algorithme est l'algorithme JPEG utilisant l'encodeur RLE, le second est également l'algorithme JPEG mais utilisant l'encodeur Huffman.

Les trois facteurs que nous avons adoptés pour comparer les deux algorithmes sont les suivants : taux de compression, l'énergie consommée au niveau du réseau et la qualité d'image avant et après le processus de compression.

Afin de réaliser cette comparaison, nous avons utilisé deux ensembles d'images, chaque groupe contient 30 images comme en on a déjà dit. Pour plus d'informations sur les tailles des images avant et après la compression a l'aide des deux algorithmes proposés, veuillez consulter l'annexe D.

Le tableau (4.1) montre les différences les plus importantes entre les deux algorithmes utilisés dans le cas ou les images sont colorées :

Les algorithmes	L'énergie consommée(W)	La qualité	Le taux
Image compressé par JPEG basé sur RLE	21122954. 9	mauvaise	0.914
Image compressé par JPEG basé sur Huffman	27565561 .87	Très bien	0.73

Tableau 4.2 : Un tableau comparatif entre une image en couleur compressé par deux types d'encodeurs

Le tableau (4.3) montre les différences les plus importantes entre les deux algorithmes utilisés dans le cas ou les images sont en niveau de gris :

Les algorithmes	L'énergie consommée(W)	La qualité	Le taux
Image compressé par JPEG basé sur RLE	343633897	Mauvaise	0.953
Images compressés par JPEG basé sur Huffman	15659319.72	Très bien	0.64

Tableau 4.3: Un tableau comparatif entre une image en niveau de gris compressé par deux types d'encodeurs

Au final, nous concluons ce qui suit :

Pour les images en couleurs :

L'algorithme JPEG qui dépend de l'encodeur RLE donne un taux de compression très élevé et la puissance consommée au niveau du réseau est moindre par rapport a l'algorithme JPEG dépend de l'encodeur Huffman, mais la qualité des images est médiocre pour le même algorithme (voir le Figure 4.5).

Pour les images en niveau de gris :
Bien que l'algorithme de compression basé sur l'encodeur Huffman donne un taux de compression inférieur par rapport a l'algorithme JPEG qui dépend de l'encodeur RLE, il consomme moins d'énergie et la qualité des images est très excellente (voir le Figure 4.6).

Figure 4.5 : Des images en couleur après la compression avec l'algorithme JPEG basé sur Huffman

Figure 4.6 : Des images en niveau de gris après la compression avec l'algorithme JPEG basé sur Huffman

4.8 Conclusion :

Dans ce chapitre, nous avons parlé de toute notre implémentation, notre modèle énergétique. Ensuite, nous avons comparé entre deux algorithmes de compression d'images les plus courants dans les réseaux de capteurs, et nous avons montré les avantages de chaque algorithme ainsi que les limites de chacun.

On a trouvé que l'algorithme basé sur le l'encodeur RLE donne un taux de compression très élevées par rapport l'algorithme basé sur l'encodeur Huffman, mais la qualité visuelle est mauvaise. Les travaux sont toujours en cours dans ce domaine afin de rendre les algorithmes de compression plus adaptables aux caractéristiques de ce type de réseaux.

Conclusion Générale et Perspectives

Conclusion générale et perspectives

Les réseaux de capteurs constituent un axe de recherche très fertile et peuvent être appliqués dans plusieurs domaines différents. Cependant il reste encore de nombreux problèmes à résoudre dans ce domaine. Pour ce type des réseaux la problématique majeure est la consommation d'énergie car les nœuds capteurs ont une faible capacité d'énergie.

Le but des chercheurs dans ce domaine c'est trouver des solutions pour économiser l'énergie et donc prolonger la durée de vie d'un réseau de capteur.

Plusieurs algorithmes de compression existent dans le domaine du traitement d'images, même si les algorithmes classiques présentent de bonnes performances mais sont très couteux en énergie. Pour cela beaucoup de recherches et de travaux sont effectués pour mettre en place des algorithmes de compression qui sont adaptés aux contraintes des réseaux de capteurs sans fil.

Dans ce mémoire, nous avons cité un type particulier des RSCF qui est les réseaux de capteurs d'images, puis nous avons parlé de la méthode de compression d'images, après nous avons cité quelques travaux concernant la compression d'images dans les RCSFs.

Ensuite nous avons parlé de notre approche proposé pour la compression d'images dans les réseaux de capteurs sans fil, et la fin on a nous avons donné notre résultats de tests.

Comme perspective de ce travail, nous pensons que ça sera intéressant de combiner l'algorithme de compression JPEG basé sur l'encodeur RLE avec l'autre basé sur Huffman pour obtenir des meilleurs résultats peur être que de travailler avec chaque algorithme tous seul.

Annexe A : Code Source de L’algorithme de compression JPEG basé sur l’encodage RLE

```
from __future__ import division
import numpy as np
from scipy import misc
from scipy.fftpack import dct
import cv2
import matplotlib.pyplot as plt
class  JPEG8N:

        _Q = np.array([[16, 11, 10, 16, 24,  40,  51,  61 ],
                                        [12, 12, 14, 19, 26,  58,  60,  55 ],
                                        [14, 13, 16, 24, 40,  57,  69,  56 ],
                                        [14, 17, 22, 29, 51,  87,  80,  62 ],
                                        [18, 22, 37, 56, 68,  109, 103, 77 ],
                                        [24, 35, 55, 64, 81,  104, 113, 92 ],
                                        [49, 64, 78, 87, 103, 121, 120, 101],
                                        [72, 92, 95, 98, 112, 100, 103, 99 ]]).astype(float)

        _Q1 = None
        block_size = 8

        original_image = None
        image_block = None
        width = 0
        height = 0
        N = 1
        quality = 90

        def __init__(self, filename=None, N=1, mult=8):
                """Costruttore. E' possibileinizializzareunaimmaginesia da
                filename cheimpostandoiblocchimanualmente"""
                self.N = N
                self.block_size = 8*N
                if filename is not None:
                        self.original_image = self.__load_file(filename)
                        self.__resize()
                        self.__enblock()

input_dir = "c:/karri.jpg"
output_dir = "./output"


def __load_file(self, filename):
                """Metodo per ilcaricamentodell'immagine e conversione
                in array Python"""
                img = misc.imread(filename, flatten=True).tolist()
                img = cv2.imread("input_dir", cv2.CV_LOAD_IMAGE_GRAYSCALE).tolist()
                returnimg


def __resize(self):
                """Metodo di riadattamentodellemisuredell'immagine in Metodo
                chesia di altezza e larghezzamultipla di 8N"""
                img = self.original_image
                old_w, w = len(img[0]), len(img[0])
                old_h, h = len(img), len(img)
                while w % (8 * self.N) != 0:
```

```
                w += 1
        while h % (8 * self.N) != 0:
                h += 1
        last_px = img[-1][-1]
        last_row = img[-1]
        fori in range(old_h):
                img[i] = img[i] + [img[i][-1]] * (w-old_w)
        last_row = last_row + [last_px] * (w-old_w)
        img = img + [last_row] *(h-old_h)
        self.original_image = np.array(img)
        self.width = w
        self.height = h
        return True

def __enblock(self):
        array = np.array([[[[0.] * self.block_size] * self.block_size ] * (self.width //
self.block_size)] * (self.height // self.block_size))
        for y in xrange(self.height // self.block_size):
                for x in xrange(self.width // self.block_size):
                        for j in xrange(self.block_size):
                                fori in xrange(self.block_size):
                                        array[y][x][j][i] = self.original_image[self.block_size*y
+ j][self.block_size*x + i]
        self.image_block = array

def __dct2D(self, x, inverse=False):
        t = 2 if not inverse else 3
        temp = dct(x, type=t, norm='ortho').transpose()
        returndct(dct(x, norm='ortho', type=t, axis=0), norm='ortho', axis=1, type=t)#.transpose()

def __get_qf(self):

        ifself.quality< 1:
                self.quality = 1
        ifself.quality> 100:
                self.quality = 100
        ifself.quality< 50:
                return 5000/self.quality/100.0
        else:
                return (200-self.quality*2)/100.0

def __force_baseline(q):
        array = np.array([[0.] * (self.N*8)] * (self.N*8))
        for y in range(len(q)):
                for x in range(len(q[0])):
                        array[y][x] = q[y][x] if q[y][x] <= 255 else 255
        return array

def __quantitize(self, block, inverse=False):
                        array = np.array([[0.] * self.block_size] * self.block_size)
        if not inverse:
                for y in range(self.block_size):
                        for x in range(self.block_size):
                                array[y][x] = round(float(block[y][x]) / float(self.Q1[y][x]))
                array = np.round(np.divide(block, self.Q1))
        else:
                for y in range(self.block_size):
                        for x in range(self.block_size):
                                array[y][x] = block[y][x] * self.Q1[y][x]
```

```
                        array = np.multiply(block, self.Q1)
                return array

def set_Q1(self):
                if self.__get_qf() != 0:
                        self.Q1 = self.__stretch_matrix(np.around(np.multiply(self.__get_qf(),
self._Q))).astype(float)
                else:
                        self.Q1 = self.__stretch_matrix(np.ones(self._Q.shape)).astype(float)

def __stretch_matrix(self, matrix):
                returnnp.repeat(np.repeat(matrix, self.N, axis=0), self.N, axis=1)

def __normalize(self, block):
                array = np.array([[0.] * (self.block_size)] * (self.block_size))
                for y in range(self.block_size):
                        for x in range(self.block_size):
                                if block[y][x] < 0:
                                        array[y][x] = 0
                                elif block[y][x] > 255:
                                        array[y][x] = 255
                                else:
                                        array[y][x] = block[y][x]
                return array

def join(self, width, height):
                array = np.array([[0.] * width] * height)
                for y in xrange(len(self.image_block)):
                        for x in xrange(len(self.image_block[0])):
                                for j in xrange(self.block_size):
                                        fori in xrange(self.block_size):
                                                array[y*self.block_size+j][x*self.block_size+i] =
self.image_block[y][x][j][i]
                self.width = width
                self.height = height
                self.original_image = array

def compress(self, quality):
                self.quality = quality
                self.set_Q1()
                print"quantization matrix generated with quality=" +str(self.quality)
                for y in range(len(self.image_block)):
                        for x in range(len(self.image_block[0])):
                                self.image_block[y][x] =
self.__dct2D(self.image_block[y][x].astype(float), inverse=False)
                                self.image_block[y][x] = self.__quantitize(self.image_block[y][x],
inverse=False)

                return (self.image_block, self.quality)

defuncompress(compressed_image_structure):
                array = compressed_image_structure[0]
                bs = len(array[0][0])
                img = JPEG8N(filename=None, N=bs//8)
                img.quality = compressed_image_structure[1]
                img.image_block = compressed_image_structure[0]
                img.set_Q1()
                for y in range(len(img.image_block)):
                        for x in range(len(img.image_block[0])):
```

```
                                    img.image_block[y][x] = img.__quantitize(img.image_block[y][x],
inverse=True)
                                    img.image_block[y][x] =
img.__dct2D(img.image_block[y][x].astype(float), inverse=True)
                                    img.image_block[y][x] = img.__normalize(img.image_block[y][x])

                img.join(len(img.image_block[0])*bs, len(img.image_block)*bs)
                print "image uncompressed (" +str(img.width) + "x"+ str(img.height)+")"
                returnimg

im = cv2.imread(input_dir)
assert not isinstance(input_dir,type(None)), 'im not found'
implot = plt.imshow(im)
plt.show()
```

Annexe B : Code Source de L’algorithme de compression JPEG basé sur l’encodage Huffman

```
from scipy.fftpack import dct,idct
import scipy
import numpy as np
from PIL import Image, ImageDraw,ImageFont
import os
import matplotlib.pyplot as plt

Z = [[16, 11, 10, 16, 24, 40, 51, 61],
    [12, 12, 14, 19, 26, 58, 60, 55],
    [14, 13, 16, 24, 40, 57, 69, 56],
    [14, 17, 22, 29, 51, 87, 80, 62],
    [18, 22, 37, 56, 68 ,109 ,103 ,77],
    [24, 35, 55, 64, 81 ,104 ,113 ,92],
    [49, 64, 78, 87, 103, 121, 120, 101],
    [72, 92, 95, 98, 112, 100, 103, 99]]
temporal = {}
frecuencia = {}

g = 0;

def dct2(a):
    new_matriz = np.zeros((8,8));
    b = np.zeros((8,8));

    for i in range(8):
        for j in range(8):
            b[i][j] = a[i][j] - 128;

    c = scipy.fftpack.dct( scipy.fftpack.dct( b, axis=0, norm='ortho' ), axis=1, norm='ortho' )

    for i in range(8):
        for j in range(8):
            new_matriz[i][j] = np.fix(c[i][j]/Z[i][j]) # fix realiza redondeo a piso

    return new_matriz


def zigzag(matrix):
    global g
    con = 0
    matrix = np.array(matrix)
    rows=8
    columns=8
    aux = np.zeros((1,64))
    x = np.zeros((1,64))
    y = np.zeros((1,64))

    solution=[[] for i in range(rows+columns-1)]

    for i in range(rows):
        for j in range(columns):
            sum=i+j

            if(sum%2 ==0):
                solution[sum].insert(0,matrix[i][j])

            else:
                solution[sum].append(matrix[i][j])
```

```
    solucion = solution.reverse()
    f = open("zig.txt", "a")
    for i in solution:
        for j in i:
            if j == -0.0:
                j = abs(j)
                aux[0,con] = j
            else:
                aux[0,con] = j #Se guarda en vector el resultado del ordenamiento zigzag
            con = con + 1;

    indice = 0;

    for i in range(0,64):
        if(aux[0,i]!= 0):
            indice = i
            break

    for i in range(indice,64):

        temporal[g] = aux[0,i];

        if aux[0,i] in frecuencia:
            pass

        else:
            frecuencia[aux[0,i]] = aux[0,i]
        g = g+1

def save_probabilidades(dic,keys):
    pos = 0;
    file = open("result.txt","w")
    for i in probabilidades:
        file.write(str(keys[pos]) +"\t"+ str(dic.get(i,i))+"\n");
        pos+= 1


matrix = Image.open("C:/img30.jpg")
print(matrix.show())
matrix.show()
matrix = matrix.convert('L')
matrix.save("gray.jpg")
matrix.show()

alto, ancho = matrix.size

a = np.asarray(matrix,dtype=np.float32)

alternativo  = a;
alternativo = alternativo - 128;

Image.fromarray(alternativo.astype(np.uint8)).save("restada.jpg")
I = Image.open("restada.jpg");
I.show()
im2 = np.zeros((256,256))
```

```
for i in range(0,alto,8):
    for j in range(0,ancho,8):
        im2[i:(i+8),j:(j+8)] = dct2(a[i:(i+8),j:(j+8)])



Image.fromarray(im2.astype(np.uint8)).save("dct.jpg")
I = Image.open("dct.jpg");
I.show()

for i in range(0,alto,8):
        for j in range(0,ancho,8):
            zigzag(im2[i:(i+8),j:(j+8)])


f = open("dct.txt","w")
for i in range(0,alto):
    for j in range(0,ancho):
        if im2[i,j] == -0.0:
            f.write(str(abs(im2[i,j]))+" ")
        else:
            f.write(str(im2[i,j])+" ")
    f.write("\n")


keys = list(frecuencia.keys())

elementos = list(temporal.values())
print(elementos)
tam = len(keys)

probabilidades = {}

for i in keys:
    telemento = float(elementos.count(i));
    print i;
    probabilidades[i] = telemento/float(len(elementos));

    keys[i];
save_probabilidades(probabilidades,keys)

print("Se a realizado exitosamente el procesamiento de la imagen. \n Se ha generado un archivo de probabilidades.\n Se ha generado el archivo de texto que contiene la matriz resultante del procesamiento de JPEG")

print len(probabilidades)
zigzag(matrix)
```

Annexe C : Code Source de L’algorithme de modèle énergétique

```
Pro=22.2*10**6

def Energie_transmission(Pt0,a,A0,d,Pro):
    result1= (Pt0*10**6+(a+1)*(A0*d**a*Pro))
    return result1



def transmission_normalisée(Tst,Lsync,Lrts,Ldata,Rr):
    result2= (3*Tst*10**-6+((Lsync+Lrts+Ldata)/(Rr*125)))
    return result2




def reception_normalisée(Tst,Lcts,Rr):
    result3= (Tst*10**-6+(Lcts/(Rr*125)))
    return result3



Pslp= 37*10**-6
for i in range(100):
resultFinal_non_compressée=
346*(transmission_normalisée(195,24,30,82,20)*Energie_transmission(15.9,3.2,40,50,22.2)+Pro*recepti
on_normalisée(195,24,20)+(1-reception_normalisée(195,24,20)-
transmission_normalisée(195,24,30,82,20)))* Pslp


print("énergie total consommée lors de l'envoi d'une image non compressée  est = ",
resultFinal_non_compressée,"W")

print()
for i in range(100):
    resultFinal_compresséeRLE= 359
*(transmission_normalisée(195,24,30,82,20)*Energie_transmission(15.9,3.2,40,50,22.2)+Pro*reception_
normalisée(195,24,20)+(1-reception_normalisée(195,24,20)-
transmission_normalisée(195,24,30,82,20)))* Pslp


print("énergie total consommée lors de l'envoi d'une image compressée  en utilisant le codage RLE est=
",  resultFinal_compresséeRLE,"W")
print()
gainAlgoRLE = resultFinal_non_compressée -  resultFinal_compresséeRLE
print()
print("le gain en cas de codage RLE est",  gainAlgoRLE,"W")

print()


for i in range(100):
    resultFinal_compresséeHUFF=
101*(transmission_normalisée(195,24,30,82,20)*Energie_transmission(15.9,3.2,40,50,22.2)+Pro*recepti
on_normalisée(195,24,20)+(1-reception_normalisée(195,24,20)-
transmission_normalisée(195,24,30,82,20)))* Pslp


print("énergie total consommée lors de l'envoi d'une image compressée  en utilisant le codage
```

```
HUFFMAN est= ",  resultFinal_compresséeHUFF,"W")
print()
gainAlgoHUFF =   resultFinal_non_compressée -  resultFinal_compresséeHUFF
print("le gain en cas de codage HUFF est",  gainAlgoHUFF,"W")
print()
if(gainAlgoHUFF>gainAlgoRLE ):
    print("lalgorithme JPEG qui utilise le codage Huffman est plus éfficace")


else:
    print("lalgorithme JPEG qui utilise le codage RLE est plus éfficace ")
```

Annexe D : Dataset utilisées

Les images	La taille (bit) Image colorée	La taille (bit) Image noir et blanc
1	584120	465920
2	936088	215640
3	885584	238552
4	1093896	208472
5	1061528	412568
6	981184	617888
7	1039424	219752
8	980608	347536
9	1192288	568944
10	981064	518344
11	751336	126464
12	971944	397536
13	757176	644704
14	721432	180768
15	754160	692752
16	695136	493136
17	556544	535992
18	869576	153584
19	683552	205024
20	982648	415208
21	1207616	590176
22	800888	627224
23	973336	532800
24	939232	612304
25	693112	435504
26	938712	473256
27	1131424	507856
28	1488952	398400
29	706752	474520
30	798088	438704
La somme :	27157400	17544728

Les tailles des images originales

Les images	Huffman	RLE
1	442840	1104000
2	670080	808000
3	592024	746400
4	774320	976000
5	776144	944000
6	756296	912000
7	673728	808000
8	682864	832000
9	897136	1160000
10	765728	912000
11	509320	622400
12	754488	944000
13	549864	696000
14	464312	563200
15	560920	689600
16	567680	680000
17	385824	468000
18	570832	705600
19	542592	694400
20	735048	936000
21	943888	936000
22	596240	808000
23	765304	1000000
24	729040	928000
25	538400	928000
26	705400	928000
27	850184	1024000
28	989128	632800
29	514696	649600
30	613584	791200

Les tailles des images colorées compressés

Les images	Huffman	RLE
1	406312	620368
2	199432	307128
3	212312	331576
4	188808	287264
5	360328	504240
6	553256	705936
7	203088	310184
8	250512	316296
9	512728	696768
10	474064	3592000
11	120264	175720
12	370416	531744
13	575800	791504
14	152816	191000
15	634912	872488
16	482616	785392
17	438376	493544
18	148544	230728
19	187456	282680
20	366008	511880
21	446360	398808
22	566288	349912
23	474976	412560
24	478832	663152
25	401208	615784
26	440056	652456
27	467160	672320
28	378728	641760
29	433008	693712
30	386328	531744

Les tailles des images en niveau de gris compressées

Annexe E : L’installation d’Omenet et ses Framework.

Installation du simulateur OMNET++ :

L'installation d'OMNET++ se fait par différentes étapes suivant une procédure d'installation décrite dans le package téléchargé selon le système d'exploitation installé.

-une bibliothèque de simulation interne.

-un compilateur du langage descriptif de la topologie NED (nedc).

-un éditeur de réseaux graphiques pour les fichiers NED (GNED).

-un exécutable OMNET++.

-une interface graphique de simulation IDE.

-un outil de documentation de modèle (opp_neddoc).

-autres utilitaires (l'outil de création Makefile, etc).

-une documentation, des simulations types, etc.

La version d'OMNET qu'on a installé est la version 4.6.

Le lien de téléchargement d'OMNET : http://omnetpp.org/download/old.html.

Les plates-formes d'OMNET++ :

Il existe plusieurs extensions, plates-formes, et simulateurs basé sur OMNET++ par ce que le OMNET++ n'est pas spécialisé pour les réseaux de capteurs sans fil donc ces plates-formes vont introduire ce manque.

Parmi ces plates-formes : Castalia, Mixim, MobilityFrameWork… etc.

Les Framework :

Dans ce qui suit les Framework que nous avons téléchargé.

.INET Framework : est une bibliothèque de modèles open source pour l'environnement de simulation OMNET++.il fournit des protocoles, des agents et d'autres modèles pour les chercheurs et les étudiants travaillant avec des réseaux de communication. INET est particulièrement utile lors de la conception et de la validation de nouveaux protocoles ou de l'exploration de scénarios nouveaux ou exotiques.

Le lien de téléchargement d'INET :
https://github.com/inetframework/inet/releases/download/v3.3.0/inet-3.3.0-src.tgz.

On a téléchargé inet3.3.0 elle est compatible avec l'OMNET 4.6.

Castalia:

Castalia est un simulateur pour les réseaux de capteurs sans fil (WSN) basé sur l'OMNET++, les réseaux corporels (BAN) et plus généralement les réseaux d'appareils embarqués a faible consommation.il s'agit d'un simulateur générique avec un canal sans fil et un modèle de radio basé sur des données mesurés.

Castalia est utilisé par les chercheurs et les développeurs pour tester leurs algorithmes ou protocoles distribuées dans des modèles réalistes de canaux et de radio sans fil, avec un comportement de nœuds réaliste, notamment en ce qui concerne l'accés a la radio.

Pour travailler avec Castalia il faut télécharger deux versions, la version « Castalia 3.2.tar.gz »

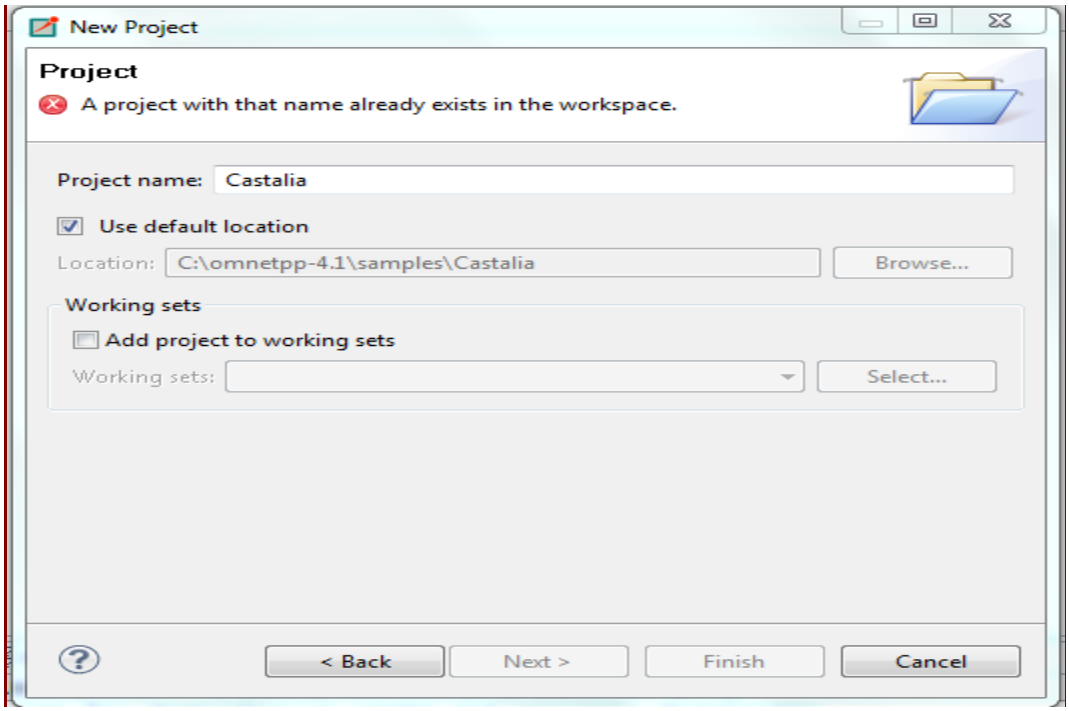
Et la version «Castalia-3.2_OMNeT-IDE_Windows_Linux-master » cette version on la trouvé et on a la télécharger a partir de ce lien:https://github.com/alexlacerda/Castalia-3.2_OMNeT-IDE_Windows_Linux, par contre la première version elle est plus disponible, donc ce n'est pas possible de travailler avec une seule version.

On a essayé aussi de travailler avec d'autres versions de Castalia comme la version 3.0 et 3.1 et 3.3 mais on aussi rencontrer des problèmes.

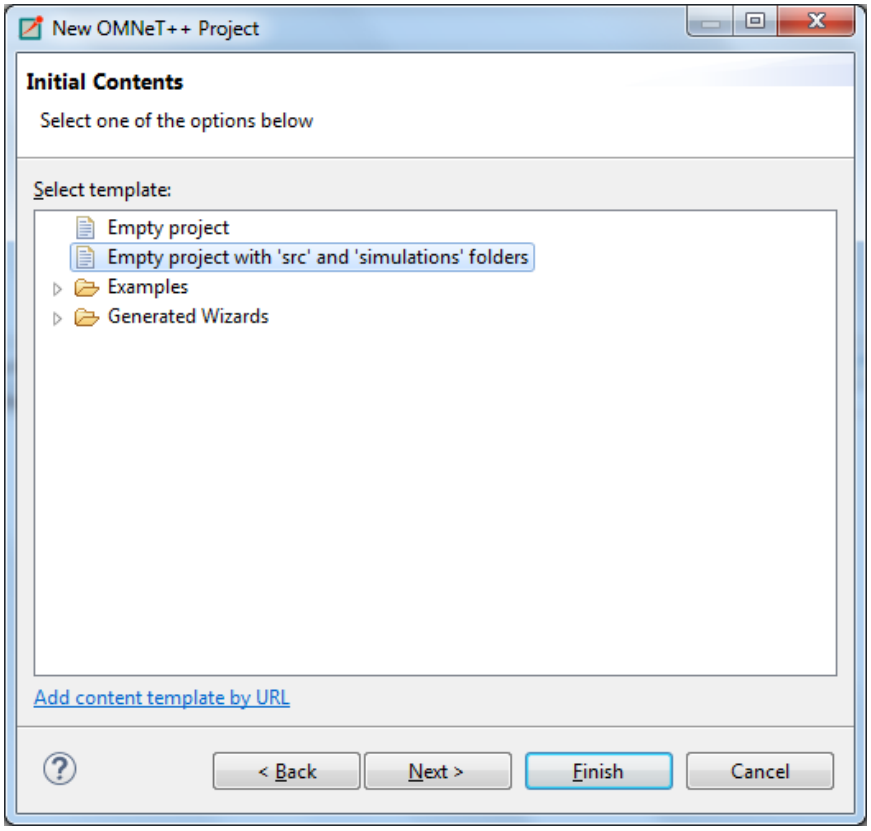
L'installation de Castalia 3.0 ou 3.1 ou 3.3 :

Les étapes sont les mêmes pour les trois versions et sont les suivants :

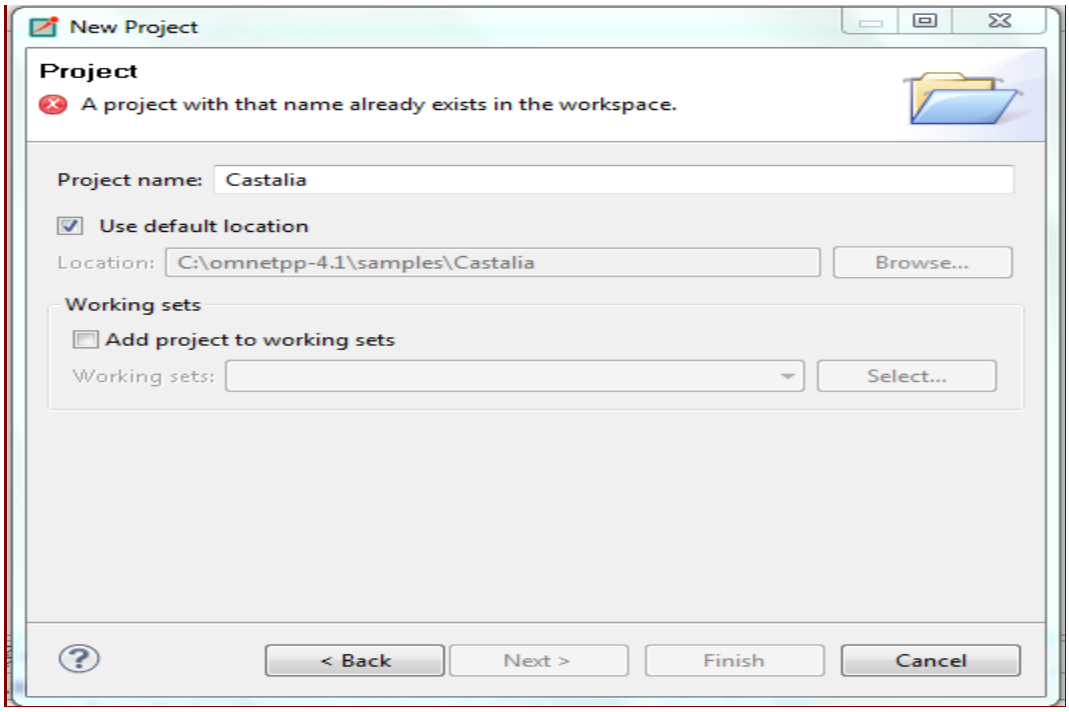
1. Décompresser de l'archive Castalia déjà télécharger.
2. Lancer le simulateur OMNET++ et créer un projet avec le nom Castalia.

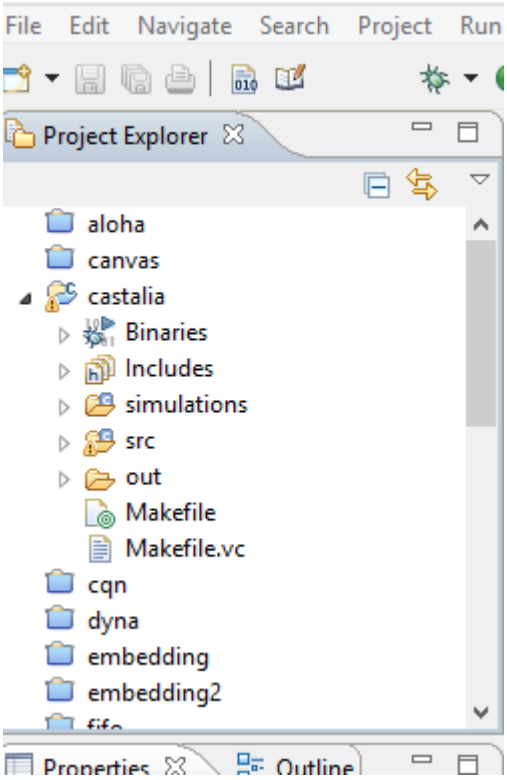


3. sélectionner l'option projet vide avec les dossiers 'src' et 'simulation'.

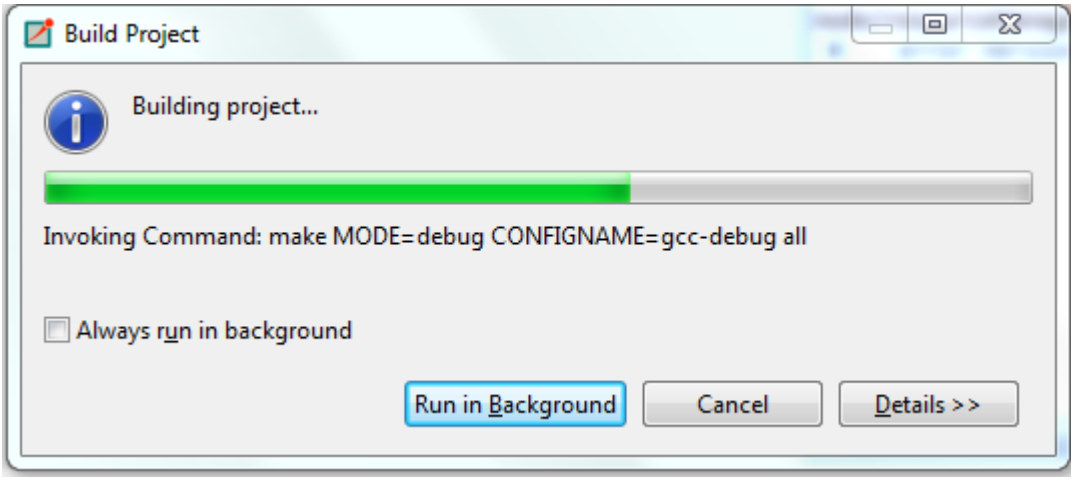


4. copiez le dossier src et simulation (Castalia 3.0) et collez-le dans le dossier src et simulation du projet Castalia, puis collez-le également dans le dossier bin.

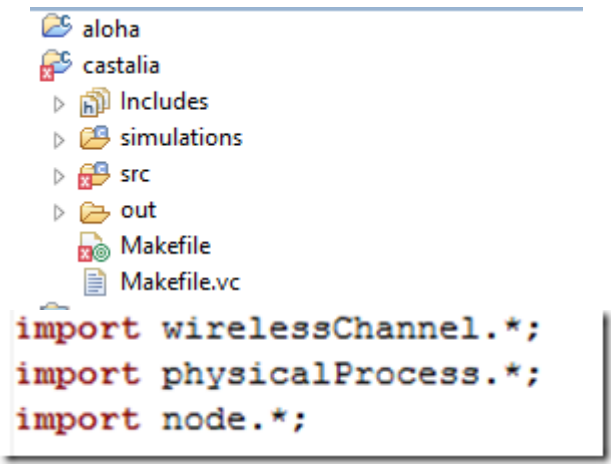




5-faire building pour le projet Castalia



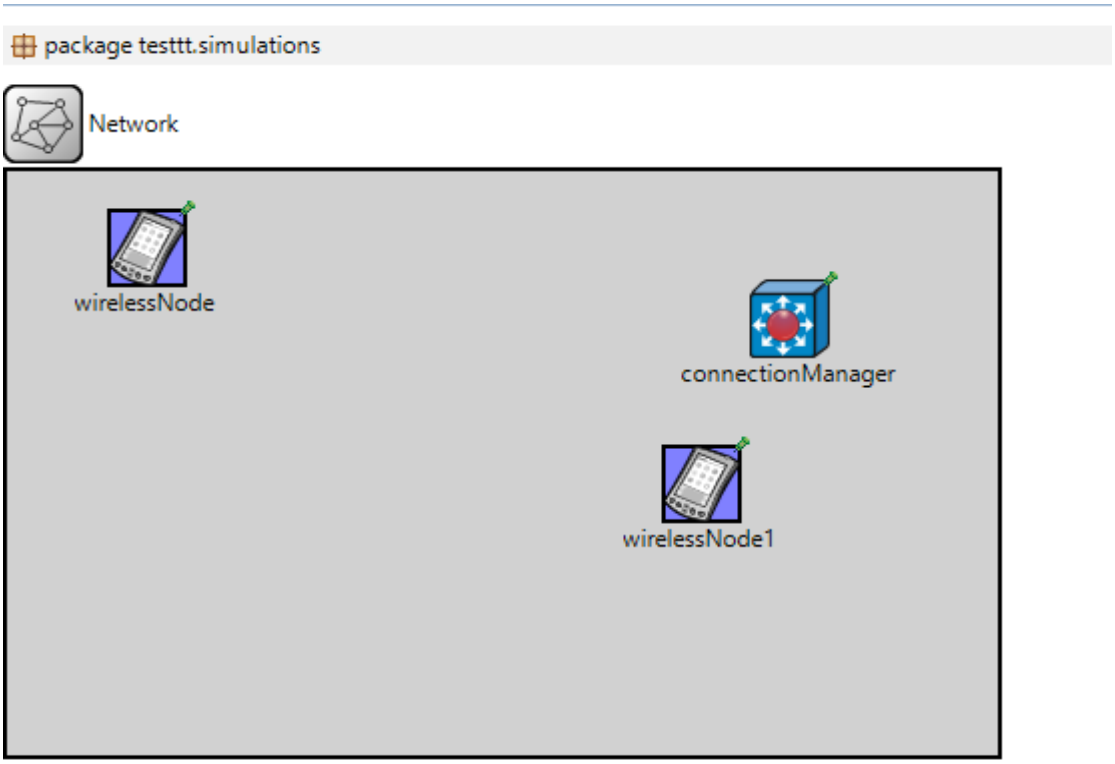
Finalement il y'a les erreurs dans les exemples prêts, on a essayé de les résoudre en ajoutant un (.*) à la fin.



Mais il y’a toujours les erreurs dans quelque exemples donc ce n’était pas possible de travailler avec Castalia.

MIXIM : est un simulateur qui intègre et développe plusieurs cadres existants pour les simulations sans fil et mobiles dans OMNET++ .il propose des modèles détaillées de propagation des ondes radio, d’estimation des interférences, de consommation d’énergie des émetteur et des récepteurs radio et des protocoles Mac sans fil.

Le problème qu’on trouvé avec MIXIM est que lorsque on modifie dans ces exemple il a toujours des erreurs, et lorsque on a essayé de faire un exemple en utilisant les modules de MIXIM ya toujours les erreurs dans le code source, on a pu faire seulement la topologie.



Bibliographie

[1] Cristian Rodrigo Duran Faundez. *Transmission d'images sur les réseaux de capteurssans fil sous la contrainte de l'énergie*. Theses, Université Henri Poincaré - Nancy 1,June2009.

[2] FadyShebli.*Réseauxdecapteurssansfil:minimisationdelaconsommationd'énergieetapplicationàlalocalisationdecibles*.PhDthesis,072008.

[3] Adel CHOUHA. *Traitement et Transfert d'images Par Réseau de Capteurs sans Fil*.Thèses,UniversitéHadjLakhder-Batna,March2011.

[4] BhargaviDalal and SampadaKukarni.Wireless sensor networks : Applications.InSiva S. Yellampalli, editor, *Wireless Sensor Networks*, chapter 1. IntechOpen, Rijeka,2021.

[5] RanjeetBKagadeandJSanthosh.Statecontextandhierarchicaltrustmanagementinwsnforintrusiondetection.In*Techno-Societal2020*,pages103–116.Springer,2021.

[6] Saeed Mohsen, AbdelhalimZekry, Khaled Youssef, and Mohamed Abouelatta.Aself-powered wearable wireless sensor system powered by a hybrid energy harvesterfor healthcare applications.*Wireless Personal Communications*, 116(4) :3143–3164,2021.

[7] AnnaLanzollaandMaurizioSpadavecchia.Wirelesssensornetworksforenvironmental-monitoring.*Sensors*,21(4):1172,2021.

[8] Ibrahima Diane. *Optimisation de la consommation d'énergie par la prise en comptede la redondance de mesure dans les réseaux de capteurs*. PhDthesis, 2014. Thèse dedoctorat dirigée par Mammeri, Zoubir et Niang, Ibrahima Réseaux télécoms, systèmesetarchitectureToulouse32014.

[9] M elle MakhmoukhDehia, & M elle MeloukSalima. (2017). Approche de minimisation de la consommation d'énergie dans les réseaux de capteurs sans fil.

[10]Makkaoui, L. (n.d.). Compression d'images dans les réseaux de capteurs sans fil. https://tel.archives-ouvertes.fr/tel-01750619v2.

[11]Reda, K., Zahir, A., & Samira, B. (2018). République Algérienne Démocratique et Populaire Systèmes des télécommunications Thème Compression d'images : Comparaison entre la méthode DCT et les ondelettes Ministère de l'enseignement supérieur et de la recherche scientifique Comparaison entre la méthode DCT et les ondelettes.

[12] Mignotte, M. diroift 6150 traitement d'images compression d'images.

[13] Al-Ani, M., Awad, F. H., Shaban AL-Ani, M., &HammadiAwad, F. (2013). THE JPEG IMAGE COMPRESSION ALGORITHM. International Journal of Advances in Engineering & Technology (Vol. 6). https://www.researchgate.net/publication/268523100.

[14] MrTounsiBillal,&MrZidoune Halim. (2016). compression de données. Université A/Mira de Béjaia.

[15] Peyré, G., Claude, G. P., & Et, S. (2016). Claude Shannon et la compression des données. https://fr.wikipedia.org/wiki/Streaming.

[16] Dufaux, F. (2011). Compression d'images. Département Traitement du Signal et des Images Telecom Paris Tech.

[17] http://www.google.com/mathworks.

[18] https://www.fileformat.info/mirror/egff/g0902.png.

[19] https://www2.ulb.ac.be/cours/acohen/travaux_2006_infodoc/CompressionNumerique/ImagesSite/arbre_Huffman.jpg.

[20] Abu Taleb, S. A., Musafa, H. M., Khtoom, M., &Gharaybih, I. K. (2010). Improving LZW Image Compression. In European Journal of Scientific Research (Vol. 44, Issue 3). http://www.eurojournals.com/ejsr.htm.

[21] http://igm.univmlv.fr/~dr/XPOSE2013/La_compression_de_donnees/images/compr-decompr-JPEG.png.

[22] https://www2.ulb.ac.be/cours/acohen/travaux_2006_infodoc/CompressionNumerique/TypeDonneesImageJPEG2000.htm.

[23] https://www2.ulb.ac.be/cours/acohen/travaux_2006_infodoc/CompressionNumerique/SansPerte.htm.

[24] https://www.techno-science.net/glossaire-definition/(JPEG-2000.html.

[25] https://zestdesavoir.com/articles/1532/(article-compression-JPEG).

[26] https://www.techno-science. net/ La compression JPEG (techno-science.net).

[27] http://le hollandaisvolant.net/science/JPG.

[28] Gabriele Monfardini - Corso di Basi di DatiMultimediali. (2006). Adaptive HuffmanCoding.

[29] http://www.techno-science.net/JPEG - La compression JPEG (techno-science.net).

[30] https://ben-tanen.com/adaptative-huffman/.

[31] https://www-igm.inv-mlv.fr/beal/Enseignementben-tanen.com/Polytechnique/Td5/huffman.html

[32] https://www.developpez.net/forums/d906116/genral-developpement/

[33] Wagner, R., Nowak, R., &Baraniuk, R. (2003). Distributed image compression for sensor networks using correspondence analysis and super-resolution. IEEE International Conference on Image Processing, 1, 597–600. https://doi.org/10.1109/icip.2003.1247032

[34] Wang, P., Dai, R., &Akyildiz, I. F. (2011). A spatial correlation-based image compression framework for wireless multimedia sensor networks. IEEE Transactions on Multimedia, 13(2), 388–401. https://doi.org/10.1109/TMM.2010.2100374.

[35] Kaddachi, M. L., Soudani, A., Lecuire, V., Torki, K., Makkaoui, L., &Moureaux, J. M. (2012). Low power hardware-based image compression solution for wireless camera sensor networks. Computer Standards and Interfaces, 34(1), 14–23. https://doi.org/10.1016/j.csi.2011.04.001.

[36] SCD_T_2009qsm. (n.d.).

[37] Mammeri, A., Hadjou, B., &Khoumsi, A. (2012). A Survey of Image Compression Algorithms for Visual Sensor Networks. ISRN Sensor Networks, 2012, 1–19. https://doi.org/10.5402/2012/760320.

[38] Morozkin, P. (n.d.). Design and implementation of image processing and compression algorithms for a miniature embedded eye tracking system. https://tel.archives-ouvertes.fr/tel-02953072.

[39] Un modèle de consommation d'énergie pour les réseaux de capteurs sans fil. (n.d.).https://doi.org/10.13140/RG.2.1.4779.9128.