Transmission d'images sur les réseaux de capteurs sans fil sous la contrainte de l'énergie

Cristian Duran-Faundez

Nancy-Université
Université Henri Poincaré
FACULTE DES SCIENCES & TECHNIQUES

U.F.R. : Sciences et Techniques Mathématiques, Informatique et Automatique
Ecole Doctorale : IAEM Lorraine
Département de Formation Doctorale : Automatique

CENTRE DE RECHERCHE EN AUTOMATIQUE DE NANCY
CRAN
CNRS UMR 7039

Thèse

présentée pour l'obtention du titre de

Docteur de l'Université Henri Poincaré, Nancy 1

en Sciences, spécialité Automatique,
Traitement du Signal et Génie Informatique

par **Cristian Duran-Faundez**

Transmission d'images sur les réseaux de capteurs sans fil sous la contrainte de l'énergie

Soutenue publiquement le 23 juin 2009

Membres du jury :

Rapporteurs :	Professeur Christine Fernandez-Maloigne	Université de Poitiers - Xlim-SIC
	Professeur David Simplot-Ryl	Université Lille 1 - INRIA Futurs
Examinateurs :	Professeur Christophe Chassot	INSA de Toulouse - LAAS
	Docteur Jean-Marie Moureaux	Université Henri Poincaré Nancy 1, CRAN
Directeurs de thèse :	Professeur Francis Lepage	Université Henri Poincaré Nancy 1, CRAN
	Docteur Vincent Lecuire	Université Henri Poincaré Nancy 1, CRAN

Université Henri Poincaré, Centre de Recherche en Automatique de Nancy

Faculté des Sciences, B.P. 70239, 54506 Vandœuvre lès Nancy

Nancy-Université
Université Henri Poincaré
FACULTE DES SCIENCES & TECHNIQUES

U.F.R. : Sciences et Techniques Mathématiques, Informatique et Automatique
Ecole Doctorale : IAEM Lorraine
Département de Formation Doctorale : Automatique

CENTRE DE RECHERCHE EN AUTOMATIQUE DE NANCY
CRAN
CNRS UMR 7039

Thèse

présentée pour l'obtention du titre de

Docteur de l'Université Henri Poincaré, Nancy 1

en Sciences, spécialité Automatique,
Traitement du Signal et Génie Informatique

par **Cristian Duran-Faundez**

Transmission d'images sur les réseaux de capteurs sans fil sous la contrainte de l'énergie

Soutenue publiquement le 23 juin 2009

Membres du jury :

Rapporteurs :	Professeur Christine Fernandez-Maloigne	Université de Poitiers - Xlim-SIC
	Professeur David Simplot-Ryl	Université Lille 1 - INRIA Futurs
Examinateurs :	Professeur Christophe Chassot	INSA de Toulouse - LAAS
	Docteur Jean-Marie Moureaux	Université Henri Poincaré Nancy 1, CRAN
Directeurs de thèse :	Professeur Francis Lepage	Université Henri Poincaré Nancy 1, CRAN
	Docteur Vincent Lecuire	Université Henri Poincaré Nancy 1, CRAN

Université Henri Poincaré, Centre de Recherche en Automatique de Nancy

Faculté des Sciences, B.P. 70239, 54506 Vandœuvre lès Nancy

Remerciements

Mes remerciements les plus sincères s'adressent en premier lieu à mes directeurs de thèse Francis Lepage et Vincent Lecuire, qui m'ont acueilli au sein du laboratoire et qui m'ont accordé leur confiance dès mon arrivée. En particulier je voudrais remercier Vincent, en qui j'ai trouvé plus qu'un encadrant, un ami. Avec lui j'ai appris et vécu beaucoup de choses pas seulement au niveau professionnel, mais aussi au niveau personnel. Tout ce qui est écrit dans cette thèse, entre autres choses, est le résultat de son engagement, de son dévouement, et de sa passion pour la recherche (sans compter une infinie patience envers moi). À lui, mon admiration et mes remerciements les plus profonds.

Sans doute que de nombreux facteurs ont une incidence sur le succès, la qualité et les résultats d'une thèse. La thèse n'est pas faite seulement entre un doctorant et ses encadrants. Les multiples réunions d'équipe, les discussions qui se génèrent et les relations amicales qui naissent avec les personnes qui travaillant dans le laboratoire, jouent un rôle tout aussi fondamental. Je remercie particulièrement les maîtres de conférences Nicolas Krommenacker, Jean-Philippe Georges et le Professeur Jean-Marie Moureaux, et bien sûr à mes collègues les futurs docteurs Nicolas, Idriss, Pierre, Carlos, Andrés, et tous les autres, qui m'ont toujours aidé et soutenu.

J'adresse, particulièrement, mes remerciements les plus distingués à Dominique Richier de l'IUT Nancy-Brabois, pour sa gentillesse, son aide désintéressée et son énorme patience avec les multiples expériences de consommation d'énergie que nous avons effectué au cours de la deuxième moitié de ma thèse.

Pour finir, je remercie mes parents et toute ma famille, de m'avoir soutenu et encouragé pendant toutes mes études et surtout pendant ces années loin de la maison.

Ce travail de thèse a été soutenue financièrement par l'Université du Bío-Bío du Chili et par une bourse « Président de la République » du gouvernement chilien.

La plateforme matérielle utilisée dans la cadre de cette thèse à été acquise avec une aide financière du CRAN et d'un projet BQR « Projets émergents - Jeunes chercheurs » par l'Université Henri Poincaré.

A mis padres
A mi familia

A la memoria de Berta Yolanda Rojas Moreno.

Table des matières

Introduction **xix**

I Positionnement : Etat de l'art sur la transmission d'images dans les réseaux de capteurs sans fil **1**

1 Élargir l'éventail d'applications pour les réseaux de capteurs sans fil **3**

1.1 Les réseaux de capteurs sans fil 4

1.1.1 Composition d'un capteur sans fil 5

1.1.2 Caractéristiques des nœuds de capteurs sans fil 6

1.1.3 Vue d'ensemble des plates-formes existantes 7

1.1.4 Applications des réseaux de capteurs sans fil 8

1.1.5 Problèmes généraux à relever 11

1.1.6 Principaux axes de recherche dans les réseaux de capteurs sans fil 13

1.2 Vers les réseaux de capteurs de vision 14

1.2.1 Applications 15

1.2.2 Spécificités des réseaux de capteurs de vision 15

1.2.3 Défis d'aujourd'hui en matière de recherche 17

1.3 Périmètre de notre travail 18

1.3.1 Contexte scientifique 18

1.3.2 Plateforme expérimentale 19

1.3.3 Mesure de la consommation d'énergie et du temps d'éxecution 20

1.3.4 Expérimentation : Pertes de données sur une plateforme réelle 21

1.3.5 Problèmes spécifiques à adresser 23

2 La transmission d'images sur réseaux de capteurs sans fil **25**

2.1 Applications des réseaux de capteurs d'image 25

2.1.1 Types d'applications 25

2.1.2 Scénarios d'application 27

2.2 Dispositifs de capture d'image 29

2.2.1 Caméras basées sur des composants commerciaux 31

2.2.2 Caméras conçues spécifiquement pour les réseaux de capteurs sans fil 33

2.2.3 Plate-formes de capteurs de vidéo 34

2.3 Traitement d'images dans les réseaux de capteurs 34
2.3.1 Compression locale 36
2.3.2 Compression distribuée 39
2.4 Transmission d'images sur réseaux de capteurs 43
2.4.1 Algorithmes de routage sur les réseaux de capteurs l'image 43
2.4.2 Transmission robuste d'images 46
2.5 Conclusion 47

II Contributions : Vers la transmission efficace d'images sur des réseaux de capteurs sans fil 49

3 Transmission d'images par un protocole semi-fiable 51
3.1 Principes techniques 53
3.1.1 Transformée en ondelettes d'une image 53
3.1.2 Prioritisation et paquétisation des données 54
3.1.3 Transmission semi-fiable 56
3.1.4 Protocole semi-fiable en boucle ouverte 57
3.2 Analyse de performances du protocole en boucle ouvert 58
3.2.1 Modélisation du protocole en boucle ouverte 59
3.2.2 Modélisation d'un *transcepteur* radio 60
3.2.3 Modélisation de la transformée en ondelettes dyadique 61
3.2.4 Modélisation de la qualité des images reçues 61
3.3 Résultats numériques 62
3.3.1 Paramètres d'entrée du modèle 62
3.3.2 Coût d'énergie avec un protocole fiable 63
3.3.3 Coût d'énergie avec un protocole semi-fiable 63
3.3.4 Impact de la politique de distribution des seuils d'énergie 67
3.4 Autres protocoles semi-fiables 67
3.4.1 Protocole semi-fiable en boucle fermée 68
3.4.2 Considération de la proximité au puits 71
3.5 Conclusion 73

4 Amélioration de la résistance aux pertes par entrelacement de pixels 75
4.1 Perte de données et dissimulation d'erreurs 76
4.1.1 Principes de la dissimilation des erreurs 76
4.1.2 Effets des pertes de paquets sur la qualité de l'image finale 76
4.2 Entrelacement de pixels 79
4.3 Entrelacement de pixels par automorphismes du Tore 81
4.3.1 Principes techniques des ATs 81
4.3.2 Proposition d'adaptation des AT pour les capteurs d'image 82
4.4 Expérimentation et analyse de résultats 84
4.5 Évaluation du coût d'énergie des AT* 86

4.6 Évaluation de la fonction d'entrelacement de pixels 87
4.7 Conclusion 88

5 Algorithme de compression d'images de faible complexité et résistant aux pertes de paquets 91
5.1 Principes techniques de ICES 93
5.1.1 Suppression de pixels 94
5.1.2 1-SAPR 94
5.1.3 Quantification scalaire uniforme 96
5.1.4 Paquétisation 96
5.2 Évaluation de la qualité des images 97
5.2.1 Comparaison de deux variantes de ICES 97
5.2.2 Comparaison de ICES avec des algorithmes de complexité similaire 98
5.2.3 Comparaison de ICES avec JPEG 101
5.3 Évaluation des ressources consommées sur un capteur d'image réel 101
5.3.1 Quantité de mémoire requise 101
5.3.2 Temps d'exécution et consommation d'énergie 104
5.4 Couplage ICES et AT 107
5.4.1 Principes techniques 107
5.4.2 Évaluation de la fonction d'entrelacement de blocs compressés avec ICES 108
5.4.3 Évaluation des performances 109
5.5 Conclusion 111

Conclusions 113

Liste des publications 119

Notice bibliographique 121

Table des figures

1.1 Schéma traditionnel d'un réseau de capteurs sans fil traditionnel 5
1.2 Anatomie générale d'un nœud de capteur 6
1.3 Quelques exemples de types de scénarios pour les réseaux de capteurs sans fil. 9
1.4 Banc d'essai utilisé pour les mesures de consommation d'énergie et du temps d'éxecution. 21
1.5 Trace de la puissance consommée par le capteur pour une application sous test. 22
1.6 Topologie expérimentale pour l'obtention de traces de pertes de paquets dans un réseau de capteurs d'image 22

2.1 Classification des réseaux de capteurs de vision en fonction de leur architecture de communication. 26
2.2 Quelques types d'application de réseaux de capteurs de vision. 27
2.3 Classification des algorithmes de compression d'image pour réseaux de capteurs sans fil . 36
2.4 Groupement des coefficients d'ondelettes en fonction de leur relation parent-enfant, comme proposé par (Wu et Chen, 2003). 38
2.5 Sélection de coefficients avec *Triangular JPEG* (Mammeri *et al.*, 2008). 39
2.6 Exemple de plusieurs nœuds caméra capturant des images corrélées. 40
2.7 Deux méthodes différentes pour l'application de JPEG2000 distribué. 42
2.8 Classification des algorithmes de routage pour les réseaux de capteurs sans fil. 44
2.9 Exemples de différentes méthodes de routage pour les réseaux de capteurs. 44

3.1 Schéma de transmission semi-fiable. 53
3.2 La TO dyadique appliquée une fois (a) ou deux (b). 53
3.3 Affectation de priorités sur les données d'uné image ayant une représentation multi-résolution bassée sur la transformée en ondelettes. 55
3.4 Relayage des paquets en fonction de leur priorité et de l'état de charge des batteries. . . . 57
3.5 Représentation du chemin entre la source et le puits. 58
3.6 Représentation d'un transcepteur radio. 60
3.7 Image originale utilisée pour les tests (128×128 pixels). 62
3.8 Formats de paquets utilisés comme paramètre dans notre modèle 64
3.9 Évaluation de la transmission semi-fiable par niveaux de résolution et par magnitudes de coefficients d'ondelettes avec une TO appliquée sur une image de 128×128 pixels. 65
3.10 Évaluation de la transmission semi-fiable par niveaux de résolution et par magnitudes de coefficients d'ondelettes avec deux TO appliquée sur une image de 128×128 pixels. 65

3.11 Exemples d'images reconstruites après application de la TO. 66
3.12 Comparaison des images reconstruites après l'application de différentes stratégies de priorités et de pertes de paquets. 67
3.13 Impact de l'application de différentes configurations des seuils d'énergie. 68
3.14 Consommation d'énergie pour la transmission d'une image avec un protocole en boucle fermée 70
3.15 Illustration de la problématique de la proximité au puits. 71
3.16 Effet des coefficients de contraction et de concavité. 72
3.17 Comparaison de performances d'une transmission semi-fiable en boucle ouverte sans et avec considération de proximité au puits. 73

4.1 Illustration de la perte d'un pixel sur une image monochrome et de son estimation à partir des pixels voisins. 77
4.2 Illustration de la perte d'un paquet sur une image monochrome et de la dissimulation des pixels manquants. 77
4.3 Visualisation des pertes du a la transmission non-fiable d'une image sans traitement au niveau de la source. 78
4.4 Exemple d'entrelacement de pixels pour la communication d'images sur un réseau sujet à des pertes de paquets. 79
4.5 Représentation de l'assignation de pixels selon la méthode de (Turner et Peterson, 1992). 80
4.6 Example de l'application des Automorphismes du Tore, avec $k = 1$, sur l'image de test 'Couloir' dans une résolution de 128×128 pixels. 82
4.7 Fonctionnement traditionnel des Automorphismes de Tore 83
4.8 Visualisation des pertes due a la transmission non-fiable d'une image sans traitement au niveau de la source. 84
4.9 AT* appliqués sur l'image « Corridor » de 128×128 pixels. 85
4.10 Qualité d'image observée en fonction du taux de pertes de paquets pour les scénarios avec et sans mélange de l'image. 86
4.11 Evaluation de la distance moyenne entre un pixel et ces voisins dans une image mélangée par AT*. 88
4.12 Qualités observées sur l'image « Corridor » en fonction des taux de perte de paquets avec différents clés de diffusion n. 89

5.1 Schéma de compression ICES. 93
5.2 Représentation du bloc $B_{p,q}$ de 2×2 pixels. 94
5.3 Débit/distorsion d'ICES vs. des algorithmes de complexité similaire. 99
5.4 Visualisation de l'image de test originale « Lenna » et la version reconstruite compressée par ICES à 3.75bpp, et comparaison de « l'œil Lenna » mis a l'échelle et reconstruit après sa compression par UQ à 4bpp, PR à 3.75bpp et ICES à 3.75bpp. 100
5.5 Visualisation de partie mise l'échelle de l'image originale « Corridor » (8bpp) et des images reconstruites compressées par UQ à 4bpp , PR à 3.75bpp et par ICES à 3.75bpp. 100
5.6 Débit/Distorsion de ICES vs. JPEG. 102

5.7 Visualisation de l'image « Lenna » reconstruite après compression par JPEG avec facteur de qualité Q=30 et ICES à 3.75bpp, et Comparaison de « l'œil Lenna » mis a l'échelle et reconstruite après compression par JPEG avec facteur de qualité Q=97 et Q=30, et par ICES à 3.75bpp. 103
5.8 Consommation d'énergie et du temps d'exécution pour les algorithmes de compression simples étudiés avec des débits binaires différentes. 105
5.9 Consommation d'énergie et du temps d'exécution pour les algorithmes de compression simples étudiés avec des débits différents. 107
5.10 Schéma de compression ICES. 108
5.11 Évaluation de l'influence de la clé de difusion n des automorphismes toriques dans la qualité d'images compressées et mélanges par blocs. 109
5.12 Comparaison de la qualité pour l'image « Corridor » après compression et transmission par ICES avec et sans mélange. 110
5.13 Comparaison de la qualité pour l'image « Corridor » après compression et transmission par JPEG* avec et sans mélange. 111

Introduction

L'émergence des réseaux de capteurs (et d'actionneurs) sans fil ouvre la voie au déploiement de nouvelles applications de surveillance (et de contrôle-commande) des grands systèmes, notamment ceux qui s'étendent sur de vastes étendues géographiques et qui requièrent une instrumentation à grande échelle. Ces applications amènent de nouveaux défis scientifiques et technologiques qui ont retenu l'attention d'un très grand nombre de chercheurs au cours des dernières années.

Les réseaux de capteurs sans fil représentent une révolution technologique des instruments de mesures, issue de la convergence des systèmes électroniques miniaturisés et des systèmes de communication sans fil. Il s'agit d'ensembles d'unités électroniques miniaturisées capables de mesurer certains phénomènes physiques dans l'environnement où ils sont déployés. En raison des contraintes de miniaturisation, et aussi de coût de fabrication, les *nœuds de capteurs* sont géneralement dotés de ressources très limitées en termes de capacité de calcul, d'espace de stockage de données, de débit de transmission et d'énergie embarquée. Ces limitations motivent un grande partie des problèmatiques de recherche dans le domaine des réseaux de capteurs sans fil, en particulier la contrainte de l'énergie qui est un problème fondamental. Il est couramment admis que le *transcepteur* radio est un des composants les plus gourmands en énergie, et donc que la plupart de l'énergie dissipée dans un nœud concerne la transmission et la réception de données. Si l'application le permet, il est donc préférable de transmettre des mesures quand un événement est détecté dans la zone de perception du nœud capteur (c'est-à-dire un changement considérable dans un phénomène mesuré) ou par demande directe plutôt que de transmettre les mesures périodiquement. Au delà du mode de fonctionnement de l'application, une des techniques les plus utilisées pour diminuer l'énergie dépensée pour la transmission des données est *l'agrégation des données*. L'agrégation de données, souvent appelée *fusion de données*, consiste à combiner les données provenant de différentes sources pour éliminer les redondances, ce qui a pour effet de réduire le trafic global du réseau. D'autres solutions existent pour diminuer la consommation d'énergie, comme par exemple l'adaptation des cycles d'endormisement (van Dam et Langendoen, 2003) ou de la puissance du transceiver radio (Cardei *et al.*, 2008).

Les applications potentielles des réseaux de capteurs sans fil couvrent de nombreux domaines, tels que : les applications militaires, la surveillance de l'environnement et la santé, entre autres. Traditionnellement, les réseaux de capteurs ont été conçus pour capturer des données scalaires simples, comme la température, la luminosité, le magnétisme, la pression ou les vibrations, par exemple, qui peuvent être codées sur quelques octets. Récemment, les progrès de la microélectronique ont permis la naissance d'une nouvelle génération de capteurs d'images miniatures ayant une très faible consommation d'énergie. Ils sont très intéressants dans les applications de surveillance, étant donné que la vision est certainement le plus puissant des sens humains (Horn, 1986). Mais les réseaux de capteurs d'images posent des problèmes supplémentaires par rapport aux réseaux de capteurs traditionnels en raison des caractéristiques parti-

culières de l'information qui est mesurée. En effet, alors que pour le codage d'une mesure de température, 2 ou 3 octets sont largement suffisants, ce qui peut être contenu dans un seul paquet, une image est généralement représentée sur plusieurs milliers d'octets (en fonction de la taille de l'image et de sa résolution). Par conséquent, le nœud capteur va devoir générer beaucoup de paquets pour transmettre l'image entière, et donc consomme beaucoup d'énergie. Une solution évidente pour diminuer la quantité de données envoyée, et donc l'énergie consommée dans le réseau, est de compresser l'image à la source. Néanmoins, la contrainte de la limitation des ressources noeuds, comme la capacité de traitement et de stockage de données, rend impossible ou inefficace en pratique l'exécution des algorithmes de compression standards. Des travaux en (Ferrigno *et al.*, 2005) ont montré en effet que des algorithmes bien connus comme JPEG ou JPEG2000 ont un coût d'énergie bien supérieur au gain qu'ils amènent sur le transceiver radio. Autrement dit, le capteur d'image épuiserait plus vite son énergie en envoyant des images compressées que des images non compressées.

Le développement de nouvelles méthodes de compression et de transmission d'images efficaces en énergie est indispensable pour que le déploiement de réseaux de capteurs d'images puisse être envisagé en pratique.

Ces trois dernières années, de plus en plus d'équipes de chercheurs apportent des contributions pour répondre à ces nouveaux défis. Mais même si l'on trouve de nos jours un nombre important de propositions dans la littérature, beaucoup de travaux sont encore trop théoriques, validés par simulation ou par modélisation mathématique sans prendre en compte les contraintes posées par les architectures matérielles réelles. Une validation expérimentale manque souvent pour prouver la faisabilité des propositions.

Cette thèse est une contribution au traitement et à la transmission d'images sur les réseaux de capteurs sans fil en considérant la contrainte de la consommation d'énergie et, de manière sous jacente, la contrainte des pertes de paquets. Cette deuxième contrainte est indispensable à prendre en compte puisque les transmissions sans fil sont faillibles (erreurs de transmission dues aux interférences, possibilité de collisions et de congestions, panne matérielle de nœuds du réseau, extinction de nœuds suite à l'épuissement de leurs batteries). Les pertes de paquets peuvent facilement être corrigées au niveau du protocole de communication par exemple avec un mécanisme basé sur les acquittements et les retransmissions de paquets, mais cela a un coût d'énergie qui doit être comptabilisé. La correction des pertes de paquets va permettre d'assurer une borne minimale de la qualité des images finales mais elle entraîne une augmentation de l'énergie consommée par le réseau et des retards de livraison des paquets. Notre objectif est de proposer des solutions qui fournissent un compromis entre l'énergie consommée par les capteurs et la qualité des images réçues. Le critère de performance peut être exprimé sous la forme d'un rapport énergie-distortion, incluant de fait le coût d'énergie de la compression et celui de la transmission. Comme la durée de vie des réseaux de capteurs d'image va dépendre avant tout à la durée de vie des nœuds source (ceux équipés de caméra), nous nous interessons spécialement à la dépense d'énergie sur ces nœuds.

Comme on l'a dit précédemment, plusieurs propositions que l'on trouve dans la bibliographie sont peu évaluées ou presentent quelques problèmes pratiques qui les rendent difficiles à mettre en œuvre, voir carrément inapplicables. De par la nature même des réseaux de capteurs, il est clair qu'on peut imaginer des algorithmes distribués pour faire la compression d'images. Le grand problème avec ce type d'approche et qu'il complexifie les scénarios de coopération des nœuds du réseau et rendent indispensable l'application de mecanismes de transmission fiable pour bien fonctionner. Cela coûte beaucoup d'énergie.

En plus, l'énergie consommée par le capteur d'image source est souvent negligée ou sous-estimée. Dans la réalité, les pertes de paquets peuvent être grandes et les topologies peuvent changer fréquemment. Nous avons voulu faire des propositions qui sont suffisamment ouvertes pour être appliquées dans n'importe quelle configuration de réseau, et qui soient très simples à implanter dans des capteurs réels.

Organisation du document

Le mémoire de thèse est organisé en 5 chapitres suivis d'une conclusion générale. Le positionnement de nos travaux est présenté sur les deux premiers chapitres et nos contributions sont détaillées dans les trois derniers.

Le chapitre 1 est une introduction aux *réseaux de capteurs sans fil*, et aux technologies associées. Il présente le fonctionnement général de ce type de réseau, ses applications potentielles et les principaux axes de recherche de ce domaine. Il introduit aussi le cas particulier des applications avec des images pour, finalement, détailler les objectifs de notre travail de recherche et le positionner par rapport aux travaux menés au niveau local et international.

Le chapitre 2 présente l'état de l'art de la recherche sur les réseaux de capteurs d'image. Il fournit une classification de ses applications et des technologies matérielles existantes. Il présente aussi les travaux les plus représentatifs, d'une part en traitement et compression d'images, et d'autre part en transmission d'images.

La deuxième partie du mémoire, composée des chapitres 3, 4 et 5, détaille nos contributions et les résultats de leur évaluation de performance. Notre première proposition, qui fait l'objet du chapitre 3 porte sur un protocole de communication semi-fiable. L'idée est de réaliser des économies d'énergie en relâchant la contrainte de fiabilité. Cela veut dire accepter, dans une certaine limite, des pertes de paquets. La différence avec un protocole non fiable est qu'ici, toutes les pertes de paquets ne sont pas admissibles. Les applications de transmission d'images peuvent s'appuyer sur un protocole de communication semi-fiable puisque les images naturelles ont, par nature, une certaine tolérance aux erreurs. La contrainte de ces applications est exprimée moins en termes de fiabilité qu'en termes de qualité de l'image finale. En d'autres mots, peu importe que des paquets soient perdus dans le réseau pourvu que la qualité de l'image finale n'en souffre pas trop.

Le chapitre 4 poursuit l'idée de relâcher la contrainte de fiabilité pour économiser de l'énergie, mais en la poussant à l'extrême cette fois-ci, c'est-à-dire en adoptant un protocole de communication non fiable. Dans ce cas, n'importe quelle perte de paquet est admise, et par conséquent la qualité minimale des images finales ne peut plus être bornée. Nous montrons en particulier que cette approche est viable à la condition d'appliquer une technique d'entrelacement de pixels à la source, avant transmission. Sur ce constat, nous avons adopté une technique d'entrelacement basée sur les automorphismes du Tore et nous proposons une adaptation efficace pour les dispositifs limités en ressources.

Le chapitre 5 traite de la compression des images à la source. Nous proposons un algorithme de compression d'image original qui, d'une part, est très peu calculatoire donc peu gourmand en énergie, et qui, d'autre part, assure que l'image compressée a un maximum de résistance aux erreurs de transmission. Notre but était clairement de transmettre des images compressées en se basant sur un protocole de communication non fiable, donc en acceptant, sans exclusive, toutes les pertes de paquets. En procédant comme cela évidemment, il faut que la qualité des images reste acceptable même pour des taux de pertes

élevés. Nous nous sommes donc orientés naturellement vers un algorithme de compression par blocs de très petites taille, 2×2 pixels, couplé à l'algorithme d'entrelacement de (blocs) de pixels donné au chapitre 4.

Première partie

Positionnement : Etat de l'art sur la transmission d'images dans les réseaux de capteurs sans fil

Chapitre 1

Élargir l'éventail d'applications pour les réseaux de capteurs sans fil

Bien souvent[1], il suffit de regarder autour de nous pour trouver des exemples d'application de capteurs de toutes sortes : Dans les machines à laver, les ascenseurs, les voitures, ou les téléphones portables ; l'intégration des capteurs dans les objets de notre vie quotidienne est en pleine expansion. C'est cette capacité à « percevoir le monde » qui fait que nos machines deviennent de plus en plus autonomes, et que nous pouvons nous libérer de nombreuses tâches qui étaient auparavant seulement possible sous supervision directe de l'homme.

Un *capteur* peut être défini comme un dispositif qui perçoit une propriété physique et qui mappe la valeur à une mesure quantitative (Gortz *et al.*, 2004). On trouve par exemple des capteurs de position, de vitesse, d'accélération, de pression, de mouvement, de luminosité, et de température, pour n'en citer que quelques uns parmi les plus simples. Des capteurs plus complexes, comme des capteurs de son ou d'images sont aussi très largement utilisés. Dans les usines modernes d'aujourd'hui, les systèmes de production sont bondés de capteurs qui surveillent et sécurisent les processus de fabrication. On y trouve par exemple des capteurs qui indiquent la position des matières premières, l'état des machines et la qualité du produit final, entre autres. Dans les voitures, on trouve des détecteurs de présence de passagers, d'ouverture des portes et de position (GPS).

Avec l'intégration des capteurs dans les systèmes embarqués, l'éventail des applications s'est élargi. Les *systèmes embarqués* sont des systèmes intégrés à fonctionalité spécifique qui permetent d'exécuter des tâches très particulières (parfois critiques) au sein de systèmes complexes, en complément de leur rôle, permettant un fonctionnement optimisé en termes d'efficacité, de fiabilité et de sécurité. L'intelligence de ce type de systèmes basés sur des microcontrôleurs dédiés peut, avec l'aide de capteurs intégrés, effectuer le processus de surveillance de certaines actions très spécifiques dans le même lieu où elles se produisent, ou envoyer rapidement des informations pertinentes pour les phénomènes détectés. Un exemple typique est l'ABS (*Antiblockiersystem*) pour l'aide au freinage d'urgence.

Si un périphérique avait besoin de communiquer avec d'autres appareils, une connexion par câble était la seule solution possible dans un premier temps. Avec le temps, le coût des puces en silicium a continué à baisser exponentiellement, et dans de nombreux cas, le coût des circuits intégrés de radio fréquence est

[1] Il faut être à la campagne ou au bout du monde et entièrement isolé pour ne pas être dans ce cas là.

maintenant inférieur à celui du câble et des connecteurs qu'ils remplacent. Un système de transmission sans fil intégré ajoute un lien de communication pour le matériel et le logiciel à intégrer sans besoins de connexions filaires rigides. Cette intégration de capteurs, de systèmes embarqués intelligents et de communications sans fil a conduit à la naissance d'une nouvelle gamme de dispositifs électroniques que ouvrent la voie à de nouvelles applications basées sur des capteurs sans fil. Ils font l'objet de plusieurs travaux de recherche et de développement, et ils sont le centre d'intérêt de cette thèse.

De toute évidence, les technologies de communication sans fil apportent plusieurs avantages (Industrial Technologies Program - U.S. Department of Energy, 2002). Tout d'abord la *portabilité*. En fonction de la connectivité, l'appareil sans fil peut être placé pratiquement n'importe où, puisqu'il n'y a plus de contrainte d'installation de câbles. Deuxièmement, des économies sur les coûts en raison de la suppresion du câblage. La *flexibilité* de ces réseaux est également un grand avantage, en effet (et cet avantage est étroitement liée à la portabilité), l'architecture de réseau sans fil n'a pas besoin d'être fixée à l'avance et les dispositifs peuvent communiquer les uns aux autres par la simple fait d'être dans la zone de couverture du signal radio. Le concept de réseau *ad-hoc* entre ici en jeu.

Les *réseaux de capteurs*[2] correspondent à un type particulier de réseau ad-hoc. Ce chapitre fait une description générale de la technologie utilisée pour les capteurs intelligents sans fil, l'élément le plus granulaire des réseaux de capteurs sans fil, puis présente les grandes classes d'applications, y compris celles faisant appel à des capteurs d'image.

1.1 Les réseaux de capteurs sans fil

Un réseau de capteurs sans fil (WSN) (Akyildiz *et al.*, 2002) est un système distribué de grande échelle mettant en communication un grand nombre d'entités autonomes communément appelées « *capteurs sans fil* », autant simplement « *capteurs* ». Ces capteurs forment donc les nœuds du réseau. Dans un scénario d'application classique, plusieurs nœuds capteurs sont déployés dans un certain environnement pour mesurer certains phénomènes physiques et faire remonter les informations collectées à une station de base, nommée le nœud *puits* (une porte d'entrée vers le monde extérieur qui fait l'interface entre le réseau de capteurs et l'utilisateur des données). Dans le cas le plus simple, les capteurs seront dans le voisinage direct du puits (un réseau de type étoile à un saut). Cependant, dans le cas d'un réseau à grande échelle, les capteurs ne sont pas tous dans le voisinage du puits et les messages seront acheminés du nœud source vers le puits en transitant par plusieurs nœuds, selon un mode de communication multi-sauts comme l'illustre la figure 1.1.

Un réseau de capteurs sans fil est un type particulier de *réseaux ad-hoc* qui sont utilisés pour l'interconnexion spontanée des systèmes informatiques. Dans un réseau ad-hoc, les entités sont en mesure de s'organiser entre elles pour former le réseau sans l'aide d'une infrastructure fixe définie à l'avance, ni d'une intervention humaine. Les nœuds ont la capacité de jouer le rôle de routeurs. Les principales différences entre les réseaux de capteurs sans fil et les réseaux ad-hoc traditionnels sont le problème de l'énergie et le facteur d'échelle. En effet, la taille d'un réseau ad hoc est habituellement considérée sur une échelle entre 10 et 100 nœuds, alors que la taille d'un réseau de capteurs est plutôt de l'ordre de plusieurs

[2] À partir de maintenant, tout au long de cette thèse, le terme *réseau de capteurs* sera utilisé indifféremment pour parler d'un *réseau de capteurs sans fil*.

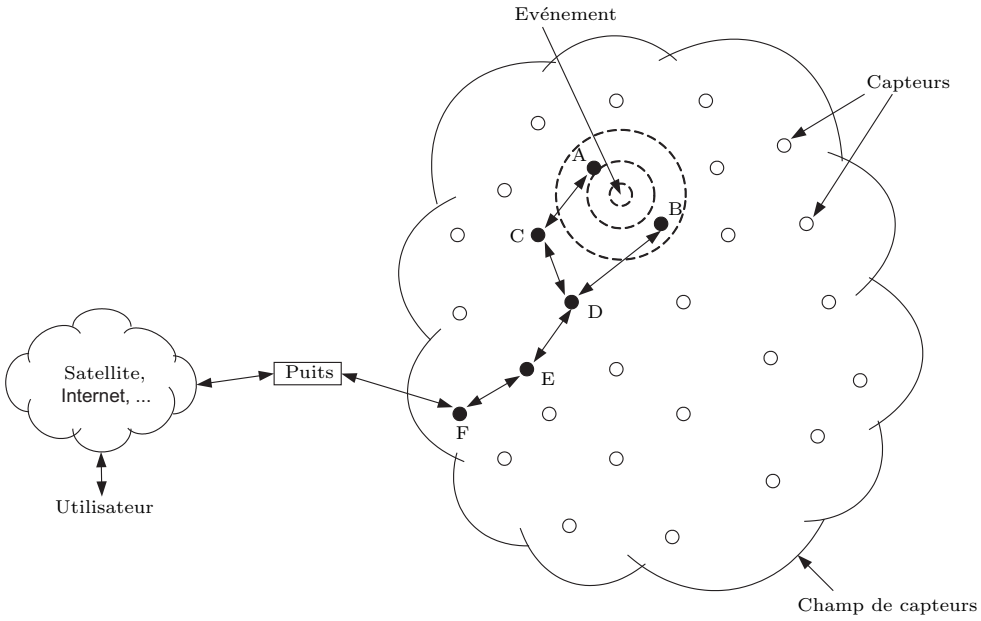


FIG. 1.1: Schéma traditionnel d'un réseau de capteurs sans fil traditionnel

centaines, voire des milliers de nœuds[3]. En outre, la densité de déploiement augmente considérablement, de l'ordre de 10 à 30 voisins usuellement. Par ailleurs, les ressources limitées des nœuds font que les réseaux de capteurs ont des différences dans leur fonctionnement, leur sécurité et leur fiabilité, selon leur application. De nouveaux protocoles de communication et d'auto-organisation doivent être développés en fonction des besoins des applications puisque les caractéristiques du réseau varient en fonction de l'application. Par exemple, certains applications considèrent que les nœuds, une fois déployés, sont fixes, d'autres considèrent qu'ils sont mobiles. Le réseau peut être homogène ou hétérogène (différents types de capteurs et de nœuds ou non). Il peut avoir un seul puits, ou plusieurs. En définitive, il y a beaucoup de scénarios envisageables et des protocoles génériques ne pourront pas être efficaces dans tous les cas. La tendance est au développement de protocoles dédiés à une application particulière.

1.1.1 Composition d'un capteur sans fil

Les capteurs sans fil considérés ici sont conçus comme de véritables systèmes embarqués, dotés de moyens de traitement et de communication de l'information, en plus de leur fonction initiale de relever des mesures. Ils représentent une révolution technologique des instruments de mesure, issue de la convergence des systèmes électroniques miniaturisés et des systèmes de communication sans fil.

Comme cela est illustré figure 1.2, un capteur sans fil est composé fondamentalement de quatre unités élémentaires :

Unité de Captage : Ce composant est l'unité qui contient le ou les capteurs embarqués sur le nœud.

[3]En août 2001, des chercheurs de l'Université de Californie, Berkeley, ont déployé le réseau de capteurs le plus important à ce jour là, composé de 800 nœuds 'Dots' (`http://webs.cs.berkeley.edu/800demo/`). En Décembre 2004, un groupe de l'Université de l'Ohio a déployé un réseau composé de 1000 nœuds de capteurs XSMs, et un *backbone* composé de 200 puits XSS, en faisant également le réseau ad-hoc sans fil 802.11b le plus important enregistré jusqu'à cette date là (`http://cast.cse.ohio-state.edu/exscal/`).

Habituellement, un *convertisseur analogique-numérique* (CAN) convertit les signaux provenant des capteurs (signaux analogiques) en signaux interprétables par l'Unité de Traitement (signaux numériques).

Unité de Traitement : Elle est généralement constituée d'un microcontrôleur dédié et de la mémoire. Cette unité fournit aux capteurs la capacité d'exécuter des calculs sur les données et les conserver selon un scénario programmé. Bien que ce ne soit pas obligatoire, il est souhaitable qu'il existe des moyens de reprogrammer facilement les capteurs dans le cas d'un changement dans les exigences de l'application.

Unité de Communication : Elle est le plus souvent constituée d'un transcepteur radio qui fournit au capteur la capacité de communiquer avec les autres au sein d'un réseau. Elle met en œuvre des protocoles de communication dépendant de la technologie utilisée (par exemple 802.11, 802.15.1, 802.15.4, etc. pour les technologies sans fil), tandis que les protocoles de plus haut niveau (routage, localisation, etc.) sont mis en œuvre dans l'Unité de Traitement. Certaines technologies radio permettent de changer la fréquence et la puissance de transmission.

Unité de Puissance : Comme il est souhaitable de s'affranchir de toute connexion par câble, le capteur doit disposer de sa propre source d'énergie qui alimente le reste des unités. Cette unité se trouve généralement sous la forme de batteries standard de basse tension.

En fonction des applications pour lesquelles ils sont conçus, les capteurs sans fil pourraient également avoir d'autres modules, comme une *Unité de Localisation*, afin d'identifier leur position géographique, par exemple en utilisant un récepteur GPS ou une technique de triangulation. Certaines applications pourraient aussi avoir besoin de capteurs équipés d'un *Mobilisateur* pour qu'ils puissent se déplacer. Enfin, s'il est nécessaire qu'un nœud soit maintenu en activité pendant une très longue période de temps, un *Générateur de Puissance*, tel que des cellules solaires, serait utile afin de tenir le nœud alimenté électriquement sans avoir à changer ses batteries.

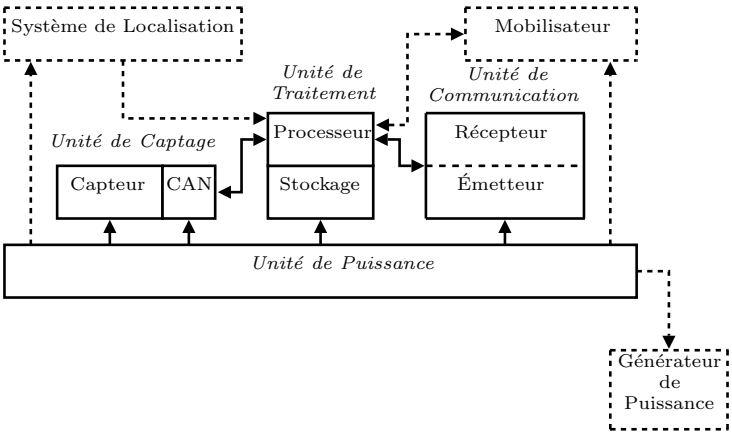


FIG. 1.2: Anatomie générale d'un nœud de capteur

1.1.2 Caractéristiques des nœuds de capteurs sans fil

En analysant la gamme des composants disponibles sur le marché et les prototypes présentés dans la littérature, il est évident que la principale caractéristique d'un nœud de capteurs sans fil est sa *petite*

taille. Depuis que les premiers nœuds de capteurs sans fil sont apparus il y a un peu plus de dix ans, la tendance est à la miniaturisation. Une deuxième caractéristique, évidente mais essentielle, est l'*autonomie* (pas seulement du point de vue de leur source d'énergie, mais aussi de leur fonctionnement). Ces deux premières particularités induisent plusieurs autres caractéristiques à considérer, en particulier la vitesse de calcul et la vitesse de transmission. Des performances élevées en termes de vitesse de traitement et de transmission impliquent une consommation d'énergie élevé. De manière générale, il est souhaitable que la durée de vie de la batterie de nœuds soit la plus grande possible, donc les différentes unités qui composent un nœud sont généralement très limitées en termes de ressources et de performance pour que leur consommation d'énergie soit extrêmement faible.

D'autres caractéristiques sont souvent utilisées comme spécificités des nœuds de capteurs dans la bibliographie, par exemple qu'ils aient un faible coût de production.

1.1.3 Vue d'ensemble des plates-formes existantes

Comme un certain nombre de technologies connues à ce jour, les nœuds de capteurs sans fil doivent être nés d'un projet militaire, ce qui entrave la mise en place d'une chronographie précise de leur développement. Cependant, le titre de premier prototype de nœuds de capteurs sans fil identifiable dans le bibliographie correspond sans aucun doute au module LWIM (Low-power Wireless Integrated Microsensors) développé dans le milieu des années 90 par l'Agence pour les Projets de Recherche Avancée de Défense (DARPA) des États-Unis et l'UCLA. Il s'agissait d'un géophone équipé d'un capteur de transmission radio-fréquence et d'un contrôleur PIC. Depuis un peu plus de 10 ans, la technologie des capteurs sans fil a beaucoup évolué. Les modules deviennent de plus en plus petits et les durées de vie prévues augmentent. Aujourd'hui, le marché de nœuds a été ouvert à l'industrie. Le fournisseur le plus connu est Crossbow Inc., avec son offre de capteurs Mica2 et MicaZ.

Le tableau 1.1 recense les différents composants actuellement disponibles sur le marché.

Le concept prévalent dans le développement de nœuds de capteurs est la conception modulaire. En effet, tous les noeuds de la table 1.1 sont en fait des cartes intégrées qui regroupent l'unité de communication et l'unité de traitement, tandis que l'unité de captage est conçue comme une carte distincte qui peut être attachée sur l'unité principale. Cela permet bien sûr de pouvoir réutiliser les mêmes unités pour différentes applications. Par exemple, un nœud Mica2 peut être combiné avec une carte MTS310 qui comprend un capteur de température, un capteur lumière, un capteur de son, un capteur de champ magnétique, et un accéléromètre à deux axes. De même, nous pouvons combiner le nœud Mica2 avec une carte MTS420 pour le doter d'un capteur d'humidité et d'un capteur de pression barométrique, et même d'un GPS pour le positionnement géographique. Une autre possibilité pour la même unité est l'ajout d'une carte d'acquisition MDA320.

Compte tenu des impératifs d'économie d'énergie que doivent respecter les nœuds de capteurs sans fil, un grand nombre de capteurs peuvent être basculés, par programmation, dans différents modes d'activité. Ainsi, un nœud de capteurs peut passer d'un mode actif, où le nœud est en pleine capacité de travail (toutes les unités sont opérationnelles), à un mode sommeil, où tout ou partie de ses éléments sont inactivés pour économiser l'énergie. Dans ce dernier mode, le minimum est laissé actif de sorte que le nœud puisse revenir à l'état actif s'il le juge nécessaire (par exemple, après un certain temps).

TAB. 1.1: Caractéristiques de noeuds de capteurs existants actuellement

Plate-forme	Fabricant	Unité de Traitement	Unité de Communication	Unité de Captage	Unité de Puissance
MICA2	Crossbow	Atmel ATMega128L (128 Ko de mémoire de programme, 4 Ko RAM) 512 ko mémoire flash pour des données EEPROM 4 Ko (configuration)	CC1000 (radio transcepteur multi-freq. 868/916 - 433 - 315 MHz, 38.4Kbaud)	Connecteur pour carte de capteurs externe	2.7 - 3.3V
MICAZ	Crossbow	Atmel ATMega128L 512 ko mémoire flash pour des données EEPROM 4 Ko (configuration)	Chipcon CC2420 (radio transcepteur 802.15.4, bande ISM de 2400 à 2483.5 MHz, 250 kbps)	Connecteur pour carte de capteurs externe	2.7 - 3.3V
IRIS	Crossbow	Atmel ATMega1281 (128 Ko de mémoire de programme, 8 Ko RAM) 512 ko mémoire flash pour des données EEPROM 4 Ko (configuration)	Radio transcepteur 802.15.4 (bande ISM, de 2400 à 2480 MHz, 250 kbps)	Connecteur pour carte de capteurs externe	2.7 - 3.3V
Imote2	Crossbow	Intel PXA271 256 ko mémoire SRAM 32 Mo mémoire SDRAM 32 Mo mémoire flash	TI CC2420 (bande ISM, de 2400 à 2483.5 MHz, 250 kbps)	Connecteur pour carte de capteurs externe	3.2 - 4.5V
Tmote Sky	Moteiv (Sentilla)	Texas Instruments MSP430 F1611 (10Ko RAM, 48Ko Flash, 128o stockage d'information)	Chipcon CC2420	Connecteur pour carte de capteurs externe	2.1 - 3.6V
BTnode rev3	ETH	Atmel ATMega128L 64+180 Kbyte RAM EEPROM 4 Ko	Bluetooth, CC1000	Connecteur pour carte de capteurs externe	DC externe 3.8 - 5V ou 2AA
Particle 2/29	TECO	PIC 18F6720 (20 MHz), Mémoire interne : 128Ko de mémoire de programme, 4Ko RAM, 1Ko EEPROM, 512 Ko Mémoire flash pour des données	TR1001 (RFM, bande passante 125Ko, bands ISM 868.35 ou 315 MHz)	Connecteur pour carte de capteurs externe	0.9 - 3.3 V

La plupart des fabricants adoptent des émetteurs RF à basse fréquence. Certains ont choisi de mettre en oeuvre un protocole d'origine récente conçu pour les modules sans fil industriels et spécifié dans la norme IEEE 802.15.4. Ce protocole de transmission opère dans la bande de fréquences des 2.4GHz. Les microcontrôleurs choisis sont généralement d'une faible vitesse et de très faible consommation d'énergie. De même, la mémoire disponible pour les programmes et les données est très réduite en comparaison avec celle des équipements informatiques d'aujourd'hui.

1.1.4 Applications des réseaux de capteurs sans fil

Plusieurs types d'applications peuvent être développées pour les réseaux de capteurs sans fil. Selon le mode de communication des données de mesure, nous identifions quatre grands scénarios d'applications :

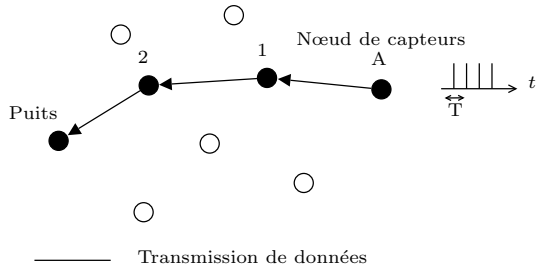
Applications périodiques : Les capteurs prennent des mesures dans des intervalles de temps réguliers,

et ils envoient les données au puits de manière périodique. Dans l'exemple de la figure 1.3(a), une image est capturée périodiquement par le noeud A, puis, il envoie les paquets vers le puits à travers les nœuds 1 et 2.

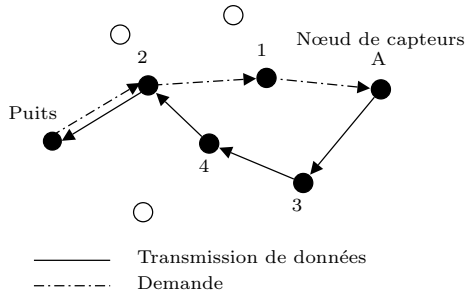
Applications à la demande (*On-Demand*) : Les capteurs attendent de recevoir un ordre du puits pour déclencher une mesure et l'envoyer. Cet ordre peut être généré par la demande manuelle d'un utilisateur humain ou d'une tâche automatique programmée. Dans l'exemple de la figure 1.3(b), une demande est adressée au noeud source A, le message est acheminé à travers les noeuds 2 et 1, et à sa réception, A active son unité de captage et envoie ces mesures vers le puits, cette fois par le chemin constitué des nœuds intermédiaires 3, 4 et 2.

Applications événementielles (Event-Driven) : Dans ce type d'applications, l'envoi de données vers le puits est déclenché lorsqu'un événement particulier est détecté. Les événements peuvent être causés par le dépassement d'un seuil dans les mesures récoltées par le capteur. Dans l'exemple de la figure 1.3(c), le noeud de capteurs A détecte un événement causé par un objet qui traverse sa zone de détection, et commence a envoyer ses mesures vers le puits à travers les nœuds 1 et 2.

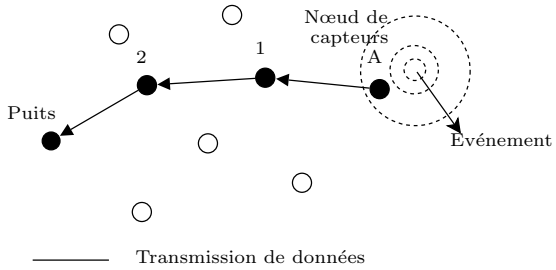
Applications hybrides : Toute alliance des cas précédents.



(a) Une *application de capture périodique*. Un nœud prend des mesures périodiquement, et envoie les paquets vers le puits à travers les nœuds 1 et 2.



(b) Une *application à la demande*. Une demande est adressée au nœud capteur A à travers les nœuds 2 et 1, puis A prend de mesures et les envoie par paquets vers le puits à travers les noeuds 3, 4 et 2.



(c) Une *application événementielle*. Un noeud de capteurs A détecte un événement causé par un objet qui traverse sa zone de détection, et commence a envoyer les valeurs de ces mesures.

FIG. 1.3: Quelques exemples de types de scénarios pour les réseaux de capteurs sans fil.

Les réseaux de capteurs sans fil ont trouvé un ensemble très vaste d'applications dans divers domaines (Arampatzis *et al.*, 2005), parmi lesquels on peut citer les applications militaires, environnementales, industrielles et de surveillance en général.

Applications militaires

Les premières applications potentielles des réseaux de capteurs ont concerné le domaine militaire. L'idée était de déployer un réseau de capteurs nanoscopiques (donc invisibles) sur des champs de bataille ou des zones ennemies pour surveiller le mouvements des troupes. Historiquement, le projet DARPA, qui a déjà été cité, a donné comme résultat les nœuds expérimentaux LWIM (très rudimentaires et assez volumineux) qui communiquaient selon une topologie en étoile. Les applications militaires sont les premières et certainement les plus représentatives des applications trouvées actuellement dans le domaine des réseaux de capteurs sans fil.

Dans (Arora *et al.*, 2005), nous trouvons les résultats d'une expérience intitulée « A Line in the Sand » (« Une ligne dans le sable ») où un réseau de capteurs sans fil était déployé dans un scénario de sécurité. Ce réseau était constitué de 90 nœuds Mica2 dotés de capteurs de métaux et de capteurs de mouvement TWR-ISM-002. Il a été déployé sur la base militaire de MacDill (Air Force Base) à Tampa (Floride), et d'autres zones d'expérimentation de la même ampleur. L'objectif du réseau de capteurs était de détecter et suivre les mouvements d'objets mobiles intrus. Le système devait être en mesure de classer les objets détectés dans le champs d'action du réseau. Trois différents groupes d'objectifs ont été classés en tenant compte des caractéristiques détectables telles que leur quantité de métal et de leur rapidité de mouvement : personne non armée, soldat et véhicule blindé. Les résultats de l'expérience montrent une précision largement acceptable dans la reconnaîssance des objets.

Applications environnementales

Une application très représentative a été effectuée dans l'île Grand Duck (44.09N, 68.15W), à Maine. Un réseau de 32 noeuds a été déployé pour la surveillance de l'habitat d'espèces protégées (Mainwaring *et al.*, 2002). Les unités déployées étaient des nœuds Mica et elles ont été utilisées pour étudier le comportement de l'océanite culblanc (*oceanodroma leucorhoa*), conformément aux changements climatiques. Les nœuds, dont certains ont été installés dans les nids des oiseaux, étaient capables de mesurer la température, la pression barométrique et d'humidité, et de transmettre les données dans un mode multi-saut jusqu'à un puits, puis vers une station de base accessible à partir d'Internet. Une application similaire peut être trouvée dans (Naumowicz *et al.*, 2008), concernant l'etude des oiseaux de mer dans une réserve nationale naturelle au Royaume-Uni.

De nombreuses applications de réseaux de capteurs se concentrent sur la mesure de phénomènes climatiques qui permettent d'étudier des changements dans l'environnement de certaines espèces animales ou végétales, afin de mieux comprendre leur comportement et, dans certains cas, supporter des études de réintroduction et de sauver des espèces qui sont en cours de disparition. Un exemple supplémentaire d'application est documenté dans (Biagioni et Bridges, 2002), pour l'étude à long terme des espèces végétales en danger.

D'autres applications environnementales sont destinées à la surveillance de certains phénomènes climatiques afin de détecter ou de prévoir certaines catastrophes naturelles telles que l'éruption des volcans

(Harvard Sensor Networks Lab, 2004 - 2008), les inondations (Schulz *et al.*, 2008) et les incendies de forêt (Doolin et Sitar, 2005).

Applications industrielles

Les technologies sans fil n'ont pas encore atteint leur apogée dans les industries, néanmoins nous commençons à voir aujourd'hui une augmentation du nombre de produits proposés pour ce milieu[4]. Certains protocoles comme la norme 802.15.4 sont en cours d'évaluation afin de déterminer s'ils peuvent supporter certaines des contraintes typiques des applications industrielles, telles que la communication temps réel (Salles *et al.*, 2008) et la robustesse aux erreurs de transmission (Willig *et al.*, 2002). Pour le moment, l'utilisation de technologies de réseaux de capteurs sans fil est encore, dans la plupart des cas, en stade expérimental.

En raison de leur sensibilité aux interférences, l'intérêt principal des réseaux sans fil dans l'industrie est concentré aujourd'hui sur les applications de maintenance prédictive. (Krishnamurthy *et al.*, 2005) par exemple, ont évalué les performances d'une application de réseaux de capteurs pour la prévision des pannes d'équipement dans des environnements industriels, sur la base de mesures de vibration. Ils ont discuté de la mise en œuvre d'une architecture de réseaux de capteurs dans une usine de fabrication de semi-conducteurs, en comparant deux plates-formes différentes : l'une basée sur des capteurs Mica2 et l'autre sur de capteurs Intel Mote (Nachman *et al.*, 2005). Une autre expérience a été réalisée dans un pétrolier dans la mer du Nord. Les résultats montrent le potentiel des réseaux de capteurs sans fil pour fournir des données de haute qualité sur des périodes de plusieurs mois. Ils sont utiles pour des applications de maintenance prédictive et disponibles à un coût relativement faible. D'autre part, les travaux de (Ramamurthy *et al.*, 2005; Ramamurthy *et al.*, 2007) sont axés sur l'intégration d'un contrôle intelligent basé sur la technologie des capteurs sans fil, non seulement pour la capture de l'information, mais aussi pour contrôler des actionneurs (des moteurs).

D'autres déploiements ont été réalisés dans les industries agricoles. Par exemple, les travaux menés par (McCulloch *et al.*, 2008) visaient à améliorer l'efficacité du système d'irrigation de pâturage, en Australie.

Autres applications

Les applications mentionnées ci-dessus sont les plus largement envisagées. Néanmoins, nous pouvons trouver d'autres domaines d'application des réseaux de capteurs. Dans le domaine de la médecine, on trouve des applications de surveillance de patients (UVA Department of Computer Science, 2005-2007). Dans la construction, on y trouve des applications de surveillance des structures de bâtiments (Xu *et al.*, 2004), ainsi que des applications liées à l'automatisation des maisons (Gauger *et al.*, 2008), le pilotage de robots (Kotay *et al.*, 2005), ...

1.1.5 Problèmes généraux à relever

Les problèmes posés par les réseaux de capteurs sans fil ont déjà été énoncés. Nous résumons ci-dessous les plus importants :

[4]WINA - Wireless Industrial Networking Alliance, *Wireless solutions by Industry*, `http://www.wina.org/WireSol/Pages/WirelessSolutionsbyIndustry.aspx`

Énergie : Comme on l'a déjà dit, l'énergie est considérée comme une ressource rare dans les applications de réseaux de capteurs sans fil. En effet, les nœuds généralement utilisent de batteries, souvent non rechargeables, et généralement n'ont pas de mécanismes de production d'électricité. Il est communément dit que les applications pour lesquelles les réseaux de capteurs sont focalisés suggèrent que le changement des batteries est difficile ou impossible. Selon l'application, les nœuds pourraient être dans des endroits difficiles d'accès, sur un champ de bataille, et ainsi de suite. Il en résulte que l'efficacité de la gestion de l'emploi de l'énergie disponible est une question souvent vitale pour le réseau.

Ressources limitées des noeuds : La demande exige une tendance vers la miniaturisation des nœuds, ainsi que vers l'élargissement de la durée de vie et la baisse du prix des unités. Les nœuds ont donc des ressources extrêmement limitées, en comparaison avec l'équipement informatique que nous avons de nos jours (tels que les ordinateurs portatifs et les PDAs), en termes de mémoire disponible, de capacité et de vitesse de traitement, de débit, ... En effet, des caracteristiques comme la haute vitesse de traitement et de transmission de données, ou une grande capacité de mémoire, sont des facultés qui amenent à une consommation énergétique très importante. Si on veut avoir de capteurs de taille microscopique, de faible consommation d'énergie et de faible coût de fabrication, on ne peut pas utiliser de microcontrôlleurs ou transcepteurs radio de haute vitesse.

Dimension et densité du réseau : Les réseaux de capteurs sont considérés comme des réseaux de très grande dimension, de l'ordre de plusieurs centains à plusieurs milliers de nœuds, déployés de manière dense (chaque nœud peut avoir plusieurs dizaines de voisins). La forte densité du réseau peut entraîner des problèmes de congestion, si les nœuds essaient de communiquer au même moment, donc des retards dans la diffusion de messages et des pertes de paquets.

La densité du réseau est généralement mise à profit pour partager le temps de travail entre les capteurs proches, et ainsi augmenter la durée de vie du réseau.

Le facteur d'échelle est également important pour la conception des protocoles de communication et des traitements de données. Le routage de paquets doit être effectué d'une manière économique en énergie, sans pour autant que les nœuds soient obligés de minoriser toutes les routes possibles. Pour maîtriser la quantité d'information à faire remonter au puits, des algorithmes de fusion de donnés sont aussi à envisager.

Environnement de communication non contrôlable : Il est habituel dans la littérature de prendre l'exemple d'un réseau de nœuds de capteurs déployé en larguant les capteurs depuis un avion. Pour ce type de déploiement, le positionnement des capteurs n'est pas contrôlé, de sorte que le réseau doit faire face à des problèmes de connectivité d'un certain nombre de nœuds qui se retrouvent en dehors de la zone de couverture des autres nœuds, soit parce qu'ils sont trop éloignés, soit parce qu'ils sont tombés dans des lieux qui entravent la propagation des ondes radio ou tout simplement parce qu'ils ont été détruits.

Les réseaux de capteurs héritent de tous les problèmes de l'usage d'une communication sans fil, tels que des problèmes d'interférences et des problèmes de sécurité (attaques). Les signaux radio émis par les nœuds peuvent être sérieusement endommagés par les interférences présentes dans le milieu. Les basses fréquences peuvent être perturbées par le bruit des machines ou d'autres agents que ne sont pas nécessairement communicants, tandis que les hautes fréquences sont perturbées par

d'autres équipements communicants que utilisent les mêmes bandes de fréquences.

Topologie dynamique : Les réseaux de capteurs sont des réseaux dont la topologie peut changer très fréquemment. Ces changements topologiques peuvent être dus à la mobilité des nœuds. Mais même pour les applications où les nœuds sont fixes, des changements peuvent se produire lorsque des nœuds sont ajoutés ou enlevés, soit par action directe de l'utilisateur, soit par le basculement de l'état des nœuds (actif/endormi), soit par l'épuisement de l'énergie, ou la panne des nœuds. Ce changement aléatoire de la disposition des nœuds exige que les nœuds puissent s'auto-organiser et cela passe par des méthodes efficaces en énergie et robustes au facteur d'échelle.

1.1.6 Principaux axes de recherche dans les réseaux de capteurs sans fil

Pour obtenir un premier aperçu de l'état de l'art sur les réseaux de capteurs, le lecteur novice est invité à consulter l'article « *survey* » de (Akyildiz *et al.*, 2002). Les axes de recherche sont de caractère pluri-disciplinaire, ils touchent les domaines de l'informatique, de l'automatique, du traitement du signal, de l'électronique, des nanotechnologies et des mathématiques.

Dans la section 1.1.3, nous avons montré que le développement de matériels pour les réseaux de capteurs est déjà un niveau avancé de l'évolution. Les efforts dans ce domaine sont faites principalement dans la conception de composants mieux intégrés, et peu gourmands en énergie. En parallèle, d'énormes efforts sont déployés aujourd'hui dans la miniaturisation des noeuds (Choi et Song, 2008). Nous avons discuté les principaux composants disponibles sur le marché. Dans la bibliographie, on trouve aussi une grande variété de prototypes non commercialisés (Vieira *et al.*, 2003). Sans aucun doute, les modules les plus utilisés sont ceux de la famille de Crossbow Mica, initialement conçu par l'Université de Californie (Hill et Culler, 2002*b*), et les noeuds Tmote Sky de Moteiv. D'autres modules qui sont développés par des universités sont : les nœuds Medusa MK-2 de l'UCLA (Savvides et Srivastava, 2002), les nœuds DIY de l'Université de Lancaster (Strohbach, 2004), les nœuds ZebraNet de Princeton, les nœuds XYZ de l'Université Yale (Lymberopoulos et Savvides, 2005) et le module Pushpin du MIT (Lifton *et al.*, 2002).

De pair avec le développement des composants matériels pour les réseaux de capteurs, la conception d'abstractions pour la programmation et la configuration des nœuds a également connue une grande évolution. Nous pouvons aussi compter aujourd'hui sur des systèmes d'exploitation et des langages de programmation pour systèmes embarqués, tels que TinyOS et NesC (UC Berkeley, n.d.) qui sont utilisés avec les motes Mica et Telos.

La recherche dans les protocoles de communication est aussi extrèmement active. Des travaux sur la couche liaison de données (MAC pour *Media Access Control*) pour les réseaux de capteurs intègrent la prise en charge des périodes d'endormissement des nœuds pour économiser l'énergie. Des exemples de protocoles MAC conçus pour les réseaux de capteurs sont B-MAC (Polastre *et al.*, 2004), S-MAC (Ye *et al.*, 2002), T-MAC (van Dam et Langendoen, 2003) et TRAMA (Rajendran *et al.*, 2003).

Il y a aussi beaucoup de travaux sur la conception d'algorithmes de routage des données. Certains chercheurs ont rompu initialement avec l'usage de l'inondation traditionnelle (dite « *flooding* »), et ont commencé à travailler sur des algorithmes de « *gossiping* » (« *bavardage* »), qui consiste à appliquer une politique probabiliste pour décider si un nœud relaye ou pas un paquet qu'il a reçu. Des algorithmes directionnels en utilisant la route la plus courte ou des variantes (la plus fiable, la moins coûteuse en énergie, ...) ont également été utilisés. La diffusion dirigée ou *Directed Diffusion* (Intanagonwiwat *et*

al., 2000) est un bon exemple d'un algorithme qui a été largement étudié.

Il existe d'autres algorithmes qui visent la génération d'une épine dorsale (*backbone*) pour rediriger des messages vers le puits, tel est le cas de SmartBone (Chuang et Chen, 2007) et le travail de (Lee *et al.*, 2007*b*).

Un autre ensemble de travaux a été consacrée à l'organisation de nœuds en groupes ou clusters, avec élection d'un nœud maître qui est responsable de la *fusion* et du routage de messages. La fusion de données (ou agrégation des données) est l'une des techniques les plus utilisées pour réduire la charge des nœuds, et obtenir des économies d'énergie. Sur son chemin vers le puits, il est très probable que les données provenant de deux nœuds proches l'un de l'autre vont passer par un même nœud intermédiaire. Dans ce cas là, ce nœud pourrait recueillir les informations provenant de ces deux sources, les fusionner, et créer un nouveau paquet avec les informations de deux sources. Ce système pourrait être étendu temporairement et quelques nœuds intermédiaires pourraient recueillir plusieurs mesures avant de créer le paquet qui fusionne toutes ces informations avant de transmettre les résultats de la fusion vers la passerelle.

Pour obtenir encore plus d'économies d'énergie, ces données fusionnées pourraient être représentées par une quantité d'informations plus petite en appliquant un algorithme de compression conçu pour les dispositifs limités en ressources, comme par exemple le *codage par ordonancement* (Coding by Ordering) proposé par (Petrovic *et al.*, 2003) ou la compression *Pipelined In-Network* de (Arici *et al.*, 2003).

En résumé, il y a un grand nombre de propositions dans le domaine des réseaux de capteurs sans fil. Néanmoins, l'état de développement de la recherche dans les réseaux de capteurs en est encore à un stade primaire et la plupart des problèmes identifiés restent ouverts.

La conception des matériels a encore un long chemin à parcourir pour obtenir des composants moins consommateurs d'énergie, de taille microscopique voir nanoscopique, des batteries, des matériaux innovants, ... En termes de protocoles de communication et de techniques de traitement des données, il y a aussi beaucoup de travail à faire. Bon nombre des articles publiés considèrent des scénarios très simplifiés, en négligant souvent des facteurs importants de la réalité. Il ya une forte nécessité pour les protocoles tolérants aux fautes et une gestion efficace de l'énergie (Kahn *et al.*, 2000; Stankovic, 2004; Aboelaze et Aloul, 2005).

L'élargissement de l'éventail de possibilités pour les réseaux de capteurs est aussi un problème latent. Au cours des dernières années, un nouveau domaine d'application a attiré à un groupe de chercheurs, motivés par ce nouveau défi : les applications basées sur des réseaux de capteurs d'images.

1.2 Vers les réseaux de capteurs de vision

Le développement des micro caméras et microphones a observé une forte évolution au cours de la dernière décennie, avec les évolutions des téléphones mobiles. Ces dispositifs deviennent de plus en plus petits et bon marché, et fournissent de plus en plus de performances en termes de rapidité et de qualité du signal. Jusqu'à il y a quelques années, l'usage d'un appareil photo impliquait la connexion d'un périphérique attaché au téléphone mobile, ou l'augmentation considérable de la taille de l'appareil. Aujourd'hui, nous trouvons ces micro-caméras embarquées dans pratiquement tous les téléphones cellulaires et les assistants numériques personnels, sans augmentation significative du coût de l'équipement, de son poids et de sa forme.

Les réseaux sans fil n'ont pas été en dehors de ce progrès et aujourd'hui, nous pouvons déjà voir les résultats des dernières avancées de microphones et micro-caméras CMOS, sous la forme de cartes de capteurs compatibles avec des nœuds sans fil, tels que ceux déjà présentés au début de cette thèse. Cela a permis d'envisager concrètement un nouveau type d'applications utilisant des *réseaux de capteurs sans fil multimédia* (Akyildiz *et al.*, 2007; Misra *et al.*, n.d.).

1.2.1 Applications

Parmi les nombreuses applications potentielles des réseaux de capteurs multimédia, celles utilisant des capteurs d'image sont appréciables pour tout ce qui concerne la reconnaissance, la localisation et le dénombrement d'objets par la vision. Certaines applications ont besoin d'identifier exactement le ou les objets qui traversent le champ du réseau de capteurs. Ce le cas par exemple dans surveillances environnementales comme d'étude du comportement des oiseaux, où il faut repérer qui entre et sort du nid, ou combien d'oeufs il y a dans le nid. Ce repérage n'est possible quà tavers la prise d'images. D'autres applications n'ont pas besoin directement d'images, mais la prise d'image peur servir à compléter et enrichir les mesures initiales. La surveillance des feux de forêt en constitue un exemple. Ce type d'application collecte des mesures de température pour détecter les départs de feux, mais la prise d'image va aider à avoir une idée plus précise de la situation pour se rendre compte de l'importance de l'incendie et de l'incidence du vent. Dans les cas mentionnés, les mesures de données scalaires vont nous aider à obtenir une certaine idée de ce qu'il se passe sur le terrain, mais la visualisation directe d'images permettra une classification plus efficace du phénomène étudié.

En fonction des exigences imposées à l'application, et bien évidemment en fonction du type de technologie disponible, les réseaux de vision peuvent être de deux types :

Réseaux de capteurs d'images fixes : Des capteurs d'images numériques peuvent prendre des photos qui peuvent être mémorisées en format matriciel ou vectoriel. Ce type de capteur est facile à réaliser et peut être adapté facilement à des dispositifs avec des ressources limitées, tels que les nœuds de capteurs sans fil.

Réseaux de capteurs de vidéo : Des capteurs d'images numériques peuvent aussi envisager de prendre des séquences d'images et de transmettre le flux vidéo vers le puits. Cette application exige des nœuds avec des capacités de calcul, de mémoire et de communication d'un tout autre ordre de grandeur que pour les images fixes. Les séquences d'image doivent être compressées fortement pour satisfaire à la contrainte de bande passante des liasons sans fil. Ces applications consomment nécessairement une quantité d'énergie bien supérieure à celles utilisant des images fixes.

En raison de la difficulté que comporte la transmission d'un flux vidéo, la plupart des prototypes de capteurs d'images sont dédiés aux images statiques.

1.2.2 Spécificités des réseaux de capteurs de vision

Bien évidemment, les travaux sur des images sont différents des travaux sur des signaux numériques ou analogiques plus simples, comme ceux des premiers réseaux de capteurs. Ces différences sont dues évidemment à la complexité du signal capturé. En effet, tandis que pour le codage d'un signal simple tel que le niveau de température ou la pression barométrique, un ou deux octets sont suffisants, le codage

d'une image numérique conduit à l'emploi de plusieurs centaines ou milliers d'octets. Cette différence de grandeur a des conséquences sur différents facteurs : capture du signal, besoins en mémoire, traitement du signal et transmission de données.

Capture du signal : La complexité du matériel est multipliée par rapport aux captures de phénomènes simples. En effet, un capteur de caméra CMOS est normalement composé de nombreux capteurs photo-sensibles que capturent les différentes intensités pour chaque pixel. Tandis que pour la capture d'un signal de lumière un seul photo-capteur est suffisant, pour capturer une image nous avons besoin de beaucoup plus (normalement un par pixel). Cette évidence entraîne avec elle un coût supplémentaire en énergie et en temps de capture.

Besoins de mémoire : Comme nous l'avons dit, tandis que pour le codage d'un signal simple sollicite quelques bits d'information (de 1 a 8 octets, en fonction de la précision du capteur), le codage d'une image numérique conduit à l'emploi de plusieurs centaines ou milliers d'octets. En particulier, la quantité de mémoire nécessaire dépend principalement de deux facteurs clés : La résolution de l'image et le format. En effet, une image de 128×128 pixels utilisera en principe 4 fois plus de mémoire qu'une image de 64×64. Maintenant, en fonction du format, une image peut être en noir et blanc, en niveaux de gris ou en couleur (Schettini *et al.*, 2003). En principe, le format définit le nombre de bits nécessaires pour coder un pixel (une intensité capturée par l'un des photo-capteurs). Une image en niveaux de gris est normalement codée sur 8 bits par pixel (désigné 8bpp), même si cette règle n'est pas obligatoire. Pour coder un pixel en couleur, nous pouvons le faire sur trois plans en utilisant, soit un codage RGB (Red, Green, Bleu), soit un codage YCrCb. Ceci implique normalement l'utilisation d'un octet par plan de couleur (24bpp).

Traitement du signal : Dans les applications traditionnelles de vision, il est commun de vouloir faire des traitements sur les images à la source, afin d'extraire une information (par exemple : l'emplacement ou la classification d'un objet), ou de compresser l'image afin de diminuer la quantité de données nécessaires pour la représenter. Alors que ces traitements sont aisés à mettre en œuvre dans des dispositifs informatiques dotés de beaucoup de ressources, comme les ordinateurs portables et les assistants numériques personnels, compte tenu des capacités limitées des matériels utilisés dans les réseaux de capteurs, le traitement d'image à la source devient très difficile. Les temps de calcul sont considérablement augmentés et l'énergie investie est parfois plus importante que celle économisée.

Transmission de données : Comme le transcepteur radio est l'un des composants les plus gourmands en énergie, les protocoles de communication ont un rôle important a jouer pour faire des économies d'énergie. Dans les applications traditionnelles (par exemple : la température ou le mouvement), on peut envisager d'enregistrer plusieurs mesures et les embarquer dans un seul paquet pour augmenter le rendement de la communication. Une des technique les plus utilisées est la fusion de données. Cela est possible parce que les mesures des différents capteurs sont généralement codées sur peu de bits et nous pouvons créer de paquets combinant des informations provenant de plusieurs sources. Dans le cas des images, la fusion de données n'est plus possible puisque les images sont transmises sur plusieurs centaines ou même milliers de paquets. Toutefois, les images naturelles ont des corrélations spatiales assez marquées et par conséquent la transmission d'images (et cela se produit également avec la transmission de la voix) offre une certaine tolérance aux pertes de paquets. En effet, on peut reconstruire une version aproximative de l'image originale même si une partie des informations est

perdue dans le réseau. Ces corrélations spatiales sont exploitées dans les algorithmes de compression, mais en contrepartie, les images compressées perdent leur tolérance aux pertes de paquets.

1.2.3 Défis d'aujourd'hui en matière de recherche

La vision est certainement le sens le plus puissant, mais aussi le plus complexe (Horn, 1986). Comme nous l'avons dit, les difficultés typiques sur les systèmes de vision, généralement associés à des problèmes de traitement à coût élevé, pourraient être multipliés lorsque nous devons faire face à d'énormes limitations en ressources, comme dans le domaine des réseaux de capteurs sans fil. Au delà des défis traditionnels des réseaux de capteurs sans fil (Kahn *et al.*, 1999; Aboelaze et Aloul, 2005), les applications des réseaux de capteurs d'images posent des défis particuliers.

A part les défis spécifiques à la conception des matériels, nous identifions dans le champ des réseaux de capteurs de vision des défis analogues à ceux classés par (Stankovic, 2004) :

Des protocoles de transmission et des algorithmes de compression d'images du monde réel : La plupart des propositions actuelles sont évaluées soit par analyse mathématique, soit par simulation. Elles considèrent des hypothèses simplifiant à l'extrême le mode de communication des capteurs, la topologie du réseau, le positionnement des nœuds, les caractéristiques des nœuds, ... , Ces hypothèses sont nécessaires pour simplifier les modèles mathématiques et les modèles de simulation utilisés pour évaluer les performances des propositions. Toutes les approches présentent d'excellents résultats par analyse mathématique et/ou simulation, mais qui y a-t-il de la réalité ? L'un des principaux défis pour les réseaux de capteurs de vision est de proposer des méthodes de compression et de communication réellement applicables, donc validées sur des plateformes réelles. Par exemple, la compression d'images selon la norme JPEG2000 a été largement discutée dans la bibliographie des réseaux de capteurs par ses indéniables qualités, en dépit de sa complexité. Au début, l'idée que le coût du traitement était négligeable prévalu, mais cette affirmation est certainement erronée (Wu et Abouzeid, 2004*b*; Ferrigno *et al.*, 2005). Pour la plate-forme mentionnée dans (Ferrigno *et al.*, 2005), JPEG2000 n'a pas donné de bons résultats. Quoi qu'il en soit, la faisabilité de JPEG2000 ou d'autres techniques de compression bien connues va dépendre des capacités des composants. La validation des modèles et des paramètres sur des plates-formes réelles doit donc être encouragée.

Temps réel : Le temps réel dans les réseaux de capteurs sans fil a été largement discuté dans la bibliographie (Oh *et al.*, 2006; He *et al.*, 2007). Certaines applications des réseaux de capteurs de vision sont soumises à des contraintes de temps réel, et par conséquent, la mise en place de mécanismes temps réel est nécessaire aussi bien en traitement d'images qu'en protocole de communication.

Gestion de l'énergie : Comme les noeuds ont par principe une source d'énergie très limitée, la gestion de l'énergie est le problème fondamental de la recherche dans les réseaux de capteurs. Ce problème est amplifié dans le cas des réseaux de capteurs de vision puisque les images forment des gros volumes de données. Prenons l'exemple suivant : (Shnayder *et al.*, 2004) ont évalué la puissance consommée et le temps d'exécution d'un mote Mica2 de Crossbow. Ils notent que le transcepteur radio d'un Mica2 consomme un courant de 3.72mA pour transmettre un octet à -20dBm (ce qui correspond à la puissance minimale de transmission), et cela prend environ 4.992E-004 secondes. Comme le Mica2 est alimenté avec une tension de 3V, nous avons une consommation d'énergie de 5,6μJ par octet

transmis. Maintenant, pour transmettre une image de 128×128 pixels, la consommation d'énergie est d'environ 91mJ sans compter les en-têtes de paquets et les champs supplémentaires à insérer dans les paquets (numéro de l'image, offset des données, ...). De plus, le coût de la capture d'image n'est pas négligeable, il coûte approximativement 90mJ pour une caméra Cyclops attachée à un mote Mica2. Cela est supérieur de plusieurs ordres de grandeurs au coût d'une mesure de lumière. Les capteurs d'images vont donc consommer beaucoup plus d'énergie que les capteurs traditionnels et donc, vont s'épuiser plus rapidement.

Abstractions de la programmation : Les défis sont les mêmes que pour les réseaux de capteurs sans fil traditionnels. Les développeurs ont besoin d'outils et de bibliothèques de programmation afin d'éviter l'exploration de la mise en œuvre de multiples niveaux.

Sécurité et confidentialité : La surveillance pour l'image est le mode de surveillance le plus intrusif qui soit. Les communications dans les réseaux de capteurs sans fil doivent faire face à des problèmes de sécurité parce que les nœuds sont généralement déployés dans des zones ouvertes faciles d'accés. Les attaques pourraient être réalisées aussi bien dans le matériel (par exemple en capturant et en prenant le contrôle d'un nœud) que dans les communications (attaques sur le protocole de communication). Pour cela, des mécanismes de sécurité doivent être utilisés et adaptés aux contraintes des réseaux de capteurs sans fil.

Jusqu'à présent, la qualité de l'image n'a pas été considérée comme une exigence essentielle dans les applications de réseaux de capteurs de vision, mais seulement souhaitable. En fait, une grande quantité de travaux considèrent qu'il vaut mieux diminuer la qualité des images pour gagner sur la consommation d'énergie et sur le temps d'exécution qui sont des critères prioritaires. Nous pouvons citer par exemple les travaux de (Chow *et al.*, 2006). (Downes *et al.*, 2006) ont expliqué comment on pourrait obtenir des informations utiles pour certaines applications avec un capteur optique Agilent ADNS-3060 capable de prendre des images de 30×30 pixels seulement. Quoi qu'il en soit, le niveau minimal de la qualité d'image va dépendre des applications.

1.3 Périmètre de notre travail

Les travaux menés au CRAN dans le domaine des réseaux de capteurs sont démarrés fin 2005. Cette thèse traite le cas des réseaux de capteurs d'image. Nous allons détailler nos objectifs scientifiques, la plateforme que nous avons développé pour les expérimentations, ainsi que les outils que nous avons utilisés pour les évaluations de performances.

1.3.1 Contexte scientifique

L'engouement pour les réseaux de capteurs d'images n'est apparu que très récemment. Il coïncide avec la commercialisation de mini-caméras en technologie CMOS, consommant donc très peu d'énergie, par exemple la caméra ADCM-1650 chez Agilent Technologies ou la MT9V011 chez Micron. Ces caméras alimentées par deux piles AA, peuvent fonctionner en continu pendant plus d'une journée ; des résultats encore meilleurs ont été obtenus par des prototypes de laboratoire (6 jours dans (McIlrath, 2001), 13 jours dans (Cho *et al.*, 2003) et 4 ans dans (Culurciello et Andreou, 2006) ! !). Une communauté scientifique en « réseaux de capteurs d'images » commence à se former, sous l'impulsion du *Center for Embedded*

Networked Sensing, UCLA, concepteur du capteur d'images Cyclops (Rahimi *et al.*, 2005). Elle se rassemble à travers l'organisation de plusieurs événements, parmi lesquels : le Workshop on Distributed Smart Cameras[5] combiné à la conférence ACM SenSys, la conférence ICDSC'2007[6], une session spéciale à la conférence IEEE ICIP'2007[7]. Un numéro spécial dans le journal Computer Networks est paru en novembre 2008 sur le thème « Wireless Multimedia Sensor Networks ».

Au début de le thèse, à la fin de 2005, il y avait très peu de travaux dans le domaine des réseaux de capteurs de vision. Les principaux pionniers sont le CENS (Center for Embedded Networked Sensing) de l'UCLA (Los Angeles), le Wireless Sensor Networks Lab de l'Université Stanford et le Sensor Networks Research Group de l'Université de Massachusetts. Au niveau national, il n'y a pas à notre connaissance de travaux préliminaires sur la transmission des images sur les réseaux de capteurs.

Toutefois, il faut noter qu'un projet de plateforme nationale du CNRS, RECAP (REseaux de CAPteurs), a démarré en 2004 pour soutenir et regrouper les activités de recherche en France sur les réseaux de capteurs. Les laboratoires partenaires sont le CITI (INSA Lyon, INRIA Rhône-Alpes), le LAAS (Toulouse), le LIFL (Université des Sciences et Technologies de Lille) et le LIP6 (Université Pierre et Marie Curie, Paris 6). Le CRAN est devenu en 2006 laboratoire-partenaire du projet CNRS RECAP. Nous participons aussi au projet ANR « Jeunes chercheuses et jeunes chercheurs » TCAP - Transport de flux vidéo sur réseaux de capteurs pour la surveillance à la demande - en association avec le LIUPPA (Laboratoire Informatique de l'Université de Pau et des Pays de l'Adour). Ce projet, qui a été sélectionné au titre du programme 2006 de l'ANR, a démarré en janvier 2007 pour une durée de 36 mois.

Notre contribution dans ce projet concerne principalement le codage et la transmission d'images sous la contrainte de la consommation d'énergie et des pertes de paquets. D'autres axes de recherche de ce projet abordent le contrôle de congestion (Maimour *et al.*, 2008), le routage multi-chemin (Maimour, 2007; Maimour, 2008) et le développement de composants logiciels (Louberry *et al.*, 2007).

1.3.2 Plateforme expérimentale

Une aide financière du CRAN, puis de l'Université Henri Poincaré dans le cadre d'un BQR « Projets émergents - Jeunes chercheurs », nous a permis d'acquérir les matériels nécessaires au développement d'une plateforme expérimentale de réseaux de capteurs d'images. Cette plateforme d'une vingtaine de nœuds, acquise principalement chez Crossbow Inc., est constituée de :

- 16 nœuds Mica2 (MPR400) et 6 nœuds Mica2Dot,
- 3 cartes de capteurs MTS510 (pour Mica2Dot), 3 cartes de capteurs MTS310 et 8 cartes de capteurs MTS300,
- une passerelle MIB510, 4 passerelles MIB520 et une passerelle Stargate (SPB400),
- 4 caméras Cyclops[8] fournits par la societé Pentar, Inc.

Les capteurs disponibles sur chaque carte sont résumés dans la table 1.2. La description des passerelles disponibles est résumé dans la table 1.3.

[5] `http://www.iti.tugraz.at/dsc06/`
[6] `http://www.icdsc.org/`
[7] `http://www.icip2007.org`
[8] Distribués par Agilent jusqu'en novembre 2007. Aujourd'hui, elles ne sont plus disponible pour l'achat.

TAB. 1.2: Quelques cartes de capteurs de notre plateforme

Capteur	MTS510	MTS300	MTS310	Cyclops
Accelerometre (2 axes)	✓		✓	
Photorésistance	✓	✓	✓	
Champ magnetique			✓	
Microphone	✓	✓	✓	
Thermistance		✓	✓	
Capteur d'image				✓

TAB. 1.3: Passerelles de notre plateforme

	MIB510	MIB520	SPB400 (Stargate)
Description	Carte de Programmation/Passerelle	Carte de Programmation/Passerelle	Système informatique embarqué (Linux) pour la programmation de capteurs, la realisation de taches et le relayage d'information.
Connecteurs pour motes	Mica2, Mica2Dot, MicaZ	Mica2, MicaZ	Mica2, MicaZ.
Connecteurs pour cartes de capteurs	Capteurs Mica (MTS, ...), caméra Cyclops	No	No
Portes de communication	RS-232	USB	RS-232, USB, Compact Flash, et PCM/CIA

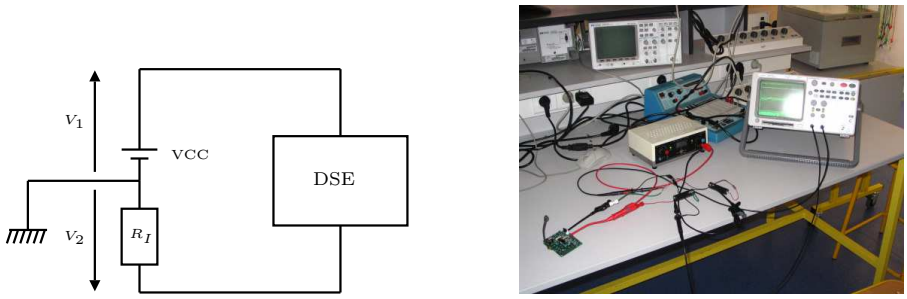
1.3.3 Mesure de la consommation d'énergie et du temps d'éxecution

Pour mesurer les performances de nos applications en termes de consommation d'énergie et de temps d'exécution, nous avons utilisé le banc d'essai illustré dans la figure 1.4. Ce banc d'essai permet de mesurer la puissance consommée par le capteur en fonction du temps, pour la durée d'un cycle basique d'éxecution, qui comprend la capture d'une image, le traitement des données, la paquetisation des données et la transmission des paquets. Dans le banc d'essai, une résistance de petite valeur ($R_I = 1\Omega$) ést connectée en série avec le Dispositif Sous Evaluation (DSE, c'est-a-dire, notre nœud capteur d'image composé d'un mote Mica2 et une caméra Cyclops), et une alimentation délivrant une tension continue. Les tensions sur l'alimentation ($V_1 \approx 3$ Volts) et celle sur la résistance (V_2) sont récupérées toutes les 0.5ms en utilisant un oscilloscope numerique Agilent 54622A.

Des calculs simples donnent le courant qui circule a travers le DSE et, à partir de cela, la puissance consommée instantanée (P en Watts). A l'instant t, $P(t)$ est donné par :

$$P(t) = \frac{[V_1(t) - V_2(t)] . V_2(t)}{R_I} \tag{1.1}$$

donc, la quantité d'énergie consommée par le DSE (E en Joules) entre les instants T_A et T_B est calculé comme :



(a) Modèle de base du banc d'essai.

(b) Vue du banc d'essai.

FIG. 1.4: Banc d'essai utilisé pour les mesures de consommation d'énergie et du temps d'éxecution.

$$E = \int_{T_A}^{T_B} P(t).dt \tag{1.2}$$

Un exemple de trace enregistrée est montré figure 1.5. Cette trace illustre un cycle de travail d'un nœud de capteur exécutant la capture d'une image de 32×32 pixels, une compression à 6bpp et la transmission des paquets avec une puissance de transmission de -20dBm. Le signal observé est très répresentatif de toutes les traces obtenues pendant nos experimentations. Dans une première phase, un augmentation rapide de la puissance consommée signale l'étape de capture de la image. Nous observons que la capture de l'image (avec un Mica2 connecté) est gourmande en termes de puissance consommée. La caméra consomme une puissance d'environ 92.69 mW pendant la capture, qui dure approximativement 0.97 secondes, c'est-à-dire, un coût d'énergie d'environ 90.64 mJ, par capture. Nous avons noté que l'énergie consommée pour la capture d'image ne varie pas de manière significative par rapport à la dimension ou aux caractéristiques de l'image. Après la capture de l'image, nous pouvons reconnaître l'étape de traitement et de transmission des données par paquets. Dans la trace de la figure 1.5, nous pouvons clairement identifier la transmission de 29 paquets obtenus comme le résultat de la compression à 6bpp (separés par des interruptions du système). Le temps d'éxecution utilisé pour le traitement et la transmission des données est d'environ 1.5 secondes, et la consommation d'énergie est d'environ 112 mJ. Comme résultat, un cycle d'exécution complet pour cette application en évaluation dure approximativement 2.5 secondes et consomme $112 + 90.64 \approx 212$mJ.

1.3.4 Expérimentation : Pertes de données sur une plateforme réelle

Afin d'obtenir des paramètres réalistes pour la simulation et d'évaluer les performances de nos propositions sous la contrainte des pertes de paquets, nous avons déployé un petit réseau composé d'un puits et de 5 nœuds de capteurs dont l'un d'entre eux est un capteur d'image. La topologie du réseau est visualisée figure 1.6.

La capture et la transmission des images sont réalisées grâce aux exemples d'applications `captureRadioTest/MoteRelay` et `MoteRelay`, conçues dans la version 1.x du langage de programmation NesC/TinyOS, et disponibles sur le dépôt CVS du firmware Cyclops (Center of Embedded Network Sensing, 2004). Le nœud source (nœud 1) est composé d'un mote Mica2 et d'une caméra Cyclops connectée.

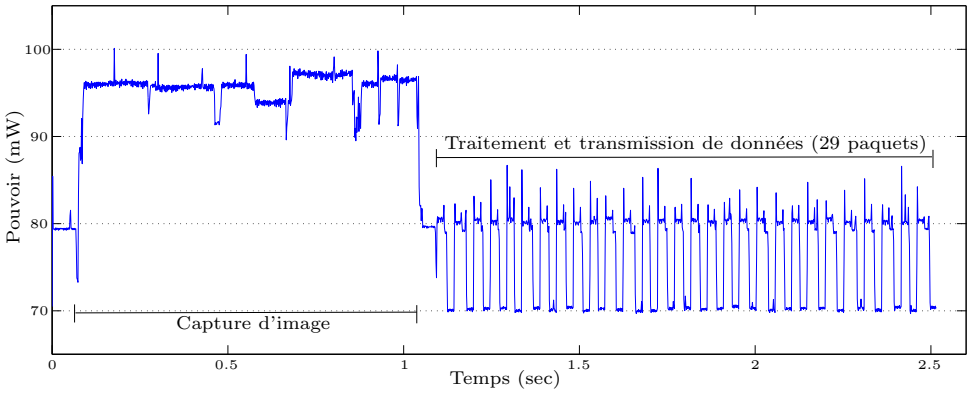
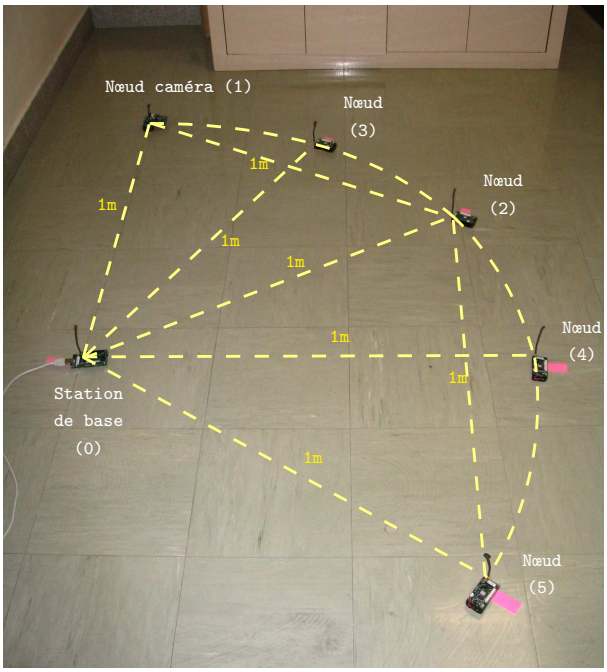


FIG. 1.5: Trace de la puissance consommée par le capteur pour une application sous test.



Composition de la Topologie :

- `Station de base (0)` : Station USB MIB520 + mote Mica2.
- `Nœud caméra (1)` : Mote Mica2 + capteur/caméra Cyclops.
- `Nœud (2-5)` : Mote Mica2.

FIG. 1.6: Topologie expérimentale pour l'obtention de traces de pertes de paquets dans un réseau de capteurs d'image

La caméra Cyclops, programmée avec l'application `captureRadioTest` capture des images monochromes de 128 × 128 pixels codées sur 8bpp et les envoie au mote Mica2 jointe. Celui-ci, programmé avec l'application `MoteRelay`, envoie les paquets vers la station de base (nœud 0) sans acquittement. La station de base est composée d'un mote Mica2 connecté a une station MIB520. Le Mica2 est programmé avec l'application `GenericBase`. Cette application est chargée de récupérer les données qui ont été reçues par la liaison radio et de les transmettre à la carte MIB associée. La carte MIB est reliée à un ordinateur de bureau via le port USB qui récupère les messages reçus et les enregistre dans un fichier pour exploitation future. La communication est réalisée selon la structure de messages de base `TOS_Msg`, utilisé par defaut sur TinyOS1.x, et qui réserve 29 octets pour des données utilisateur. 2 octets sont réservés pour un entête défini par la structure `serialDumpHeader_s`, qui indique le nombre restant d'octets à transmettre dans

la séquence. On est donc capable de transmettre un nombre maximum de 27 octets de données utiles par paquet.

Afin de forcer des grandes pertes dans le réseau, on rajoute de 1 à 5 unités Mica2 additionnelles qui génèrent du trafic de fond en envoyant des paquets à interval régulier compris entre 20ms et 125ms. Les pertes observées sur les traces collectées varient entre 0 et 92%. Les traces obtenues seront utilisées pour la simulation de pertes et l'évaluation des propositions réalisées dans cette thèse dans les chapitres 4 et 5.

1.3.5 Problèmes spécifiques à adresser

L'objectif général de la thèse est de développer des procédures de traitement et de transmission d'images assurant un compromis entre la qualité des images reçues et l'énergie consommée pour les transmettre de bout-en-bout. Nos propositions comprennent des protocoles de communication pour la transmission d'images efficace en énergie ainsi qu'une méthode de compression d'image de faible complexité et robuste aux pertes de paquets.

Nous sommes conscients que la qualité parfaite de l'image pourrait être obtenue si nous appliquons une transmission complètement fiable (avec par exemple, un protocole basé sur des accusés de réception et des retransmissions), mais cela coûte cher en énergie, ce qui n'est évidemment pas souhaitable dans une application de réseaux de capteurs sans fil. D'autre part, si aucun traitement n'était effectué à la source, et si aucun mécanisme pour garantir un certain niveau de fiabilité n'était mis en œuvre, nous négligerions forcement la qualité de l'image. Nous cherchons donc à obtenir un compromis entre les ressources investies pour compresser et transmettre les images et la qualité des images reconstruites au niveau du destinataire. Plus précisément, nous recherchons un compromis entre :

- Le taux de compression, sachant que plus on compresse, et moins l'image sera tolérante aux pertes de paquets,
- la tolérance aux pertes, sachant que moins l'image est tolérante aux pertes de paquets, et plus le protocole de communication devra assurer la correction de ces pertes,
- la consommation des ressources, sachant que compresser l'image et corriger les pertes de paquets coûte de l'énergie,
- la qualité des images reconstruites.

En effet, les diverses contraintes auxquelles sont confrontés les réseaux de capteurs sont par nature antagonistes. Nous savons que l'application d'un algorithme qui fournit une forte compression peut non pas seulement exiger trop de ressources (dans l'étape de traitement) au niveau de la source, mais aussi provoquer une diminution drastique de la tolérance aux pertes (Nous touchons ces principes avec plus de profondeur dans les chapitres suivants). Bien évidemment, avec une forte compression, la perte d'un paquet concerne la perte de beaucoup plus d'informations que la perte d'un paquet de données non compressés. D'autre part, bien que la transmission d'une image sans compression peut être une option possible, la quantité d'énergie et de temps investi (dans l'étape de transmission) sera, en principe, plus élevé.

Chapitre 2

La transmission d'images sur réseaux de capteurs sans fil

Malgré le peu de temps qui s'est écoulé depuis l'émergence du domaine de recherche des réseaux de capteurs sans fil, il existe aujourd'hui un nombre significatif de travaux traitant des capteurs d'image incluant le développement de composants matériels et logiciels, de protocoles de communication (routage, contrôle de flux, ...), et de traitements sur les données (compression, ...). Ce chapitre présente un état de l'art de ces travaux. Il est structuré en quatre parties.

Dans la première partie, nous présentons et classifions les principaux scénarios d'application des réseaux de capteurs d'images, en les illustrant par des cas pratiques récemment mis en œuvre. Puis nous présentons les travaux les plus représentatifs concernant les dispositifs de capture d'image (deuxième partie), le traitement des données de l'image aussi bien par de algorithmes locaux que des algorithmes distribués (troisième partie), et enfin la transmission de données de l'image (quatrième partie).

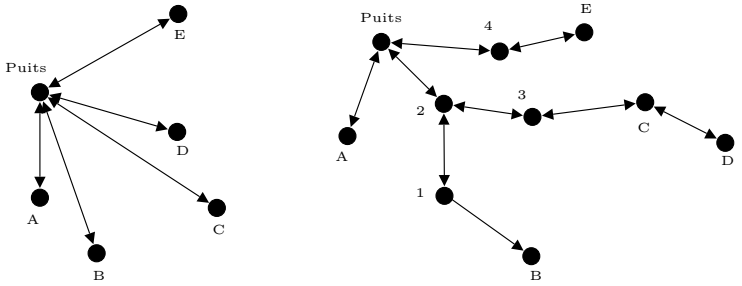
2.1 Applications des réseaux de capteurs d'image

Les réseaux de capteurs d'image concernent toutes les applications qui touchent à la détection, la localisation, le dénombrement et le pistage d'objets par la vision. Dans cette section, nous allons classifier ces applications de manière générique et en présenter quelques unes parmi les des plus représentatives qui ont été expérimentés jusqu'à aujourd'hui.

2.1.1 Types d'applications

Nous pouvons distinguer deux grandes familles d'application pour les réseaux de capteurs de vision en fonction de l'architecture du réseau considérée : réseaux à un saut (*single-hop network*) ou réseau multi-sauts (*multi-hop network*). Cette classification peut être généralisée pour tous les réseaux de capteurs sans fil. Dans le premier cas, un réseau de capteurs de vision est conçu comme un ensemble de nœuds caméra (et éventuellement d'autres types de capteurs) qui communiquent directement avec le puits comme montré figure 2.1(a). Un tel réseau a une couverture géographique limitée à la portée de communication du puits, quelques mètres à quelques centaines de mètres. Dans le deuxième cas, le réseau a une couverture

géographique beaucoup plus étendue puisque les nœuds éloignés du puits vont pouvoir transporter leur information en passant par un ou plusieurs nœuds intermédiaires, comme montré figure 2.1(b).



(a) Réseau de capteurs d'image à un saut.

(b) Réseau de capteurs d'image multi-sauts.

FIG. 2.1: Classification des réseaux de capteurs de vision en fonction de leur architecture de communication.

La disposition et le type de nœuds utilisé peut varier considérablement. On peut trouver des applications qui ne contiennent que les nœuds de capture d'images, mais il existe également des applications associant des nœuds de différents types qui se complètent mutuellement. Pour une application de détection des feux de forêt par exemple, on peut associer de capteurs de température et des capteurs d'image, les premiers fournissent l'événement déclanchant la prise d'image par les seconds. Ces nœuds peuvent éventuellement être regroupées fonctionnellement, comme suggéré par les travaux de (Kulkarni *et al.*, 2005*a*; Kulkarni *et al.*, 2005*b*) sur l'établicement des architectures multi-couches.

De même, les applications de capteurs d'images peuvent aussi être classifiées selon les modèles de surveillance ou de collecte des données des réseaux de capteurs traditionnels :

- Surveillance périodique
- Surveillance a la demande
- Surveillance sur déclenchement d'événements
- Surveillances hybrides.

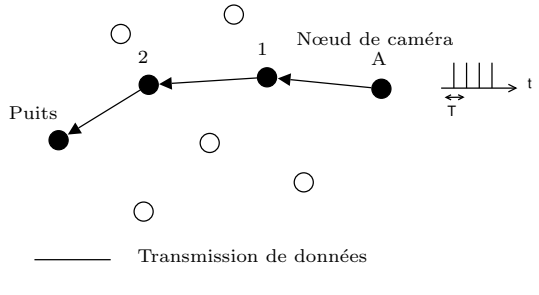
Les principes de ces classes d'application sont schématisés figure 2.2.

Les applications événementielles peuvent elles-mêmes être divisées en deux catégories en fonction du type d'événement :

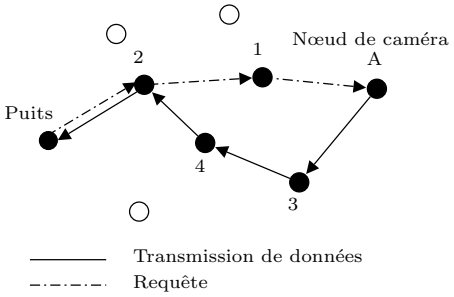
Événements internes : Les événements sont détectés par le nœud de capture d'image.

Événements externes : Les événements sont des messages reçus d'autres nœuds, qui notifient ainsi qu'un certain phénomène (l'événement) s'est produit dans leur zone de perception.

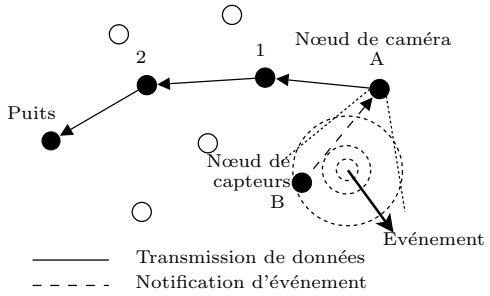
Dans les applications événementielles, des événements peuvent être détectés grâce à des capteurs de type scalaire (comme la température, la pression, les vibrations, etc), qui sont alors utilisés pour détecter un phénomène (cf. figure 2.2(c)). Ils peuvent aussi être détectés par des capteurs d'image qui vont alors appliquer un algorithme (reconnaissance de contours, de couleurs, ...), en vue d'identifier des informations intéressantes dans les images prises.



(a) Une application de capture périodique.



(b) Une application sur demande.



(c) Une *application événementielle*. Un nœud de capteurs B, équipé d'un capteur scalaire (vibrations, par exemple) détecte un événement causé par un objet qui traverse sa zone de perception. En sachant que le nœud caméra A peut prendre une photo de cette région, il lui envoie un message de notification. Finalement, le nœud A saisit une image et la transmet par paquets de données vers le puits à travers les nœuds 1 et 2.

FIG. 2.2: Quelques types d'application de réseaux de capteurs de vision.

2.1.2 Scénarios d'application

Il existe de nombreuses possibilités d'application des réseaux de capteurs d'image dans des scénarios réels. Elles concernent principalement les applications militaires, la surveillance environnementale, la sûrete et la sécurité des sites industriels, la surveillance des réseaux routiers et l'aide au déplacement de mobiles autonomes.

Applications militaires

Les réseaux de capteurs de vision peuvent être très utiles pour l'espionnage militaire et la surveillance des champs de bataille. Dans l'expérience menée à la base McDill aux USA, « Une ligne dans le sable » (décrite section 1.1.4), un réseau de capteurs de vision pourrait être utilisée pour la reconnaissance et la classification des cibles, par exemple. Le déploiment d'un réseau de capteurs sur les champs de bataille peut être réalisé manuellement ou aléatoirement. Dans le premier cas, des troupes de reconnaissance peuvent marcher dans le champ de bataille pour positionner et cacher stratégiquement des capteurs d'image. Comme les capteurs sont, par définition, des dispositifs très petits, ils seront à priori faciles à dissimuler. L'orientation des caméras doit être effectuée rigoureusement si on veut couvrir visuellement toute la zone à observer. Cela peut être un problème lorsque le réseau de capteurs est déployé aléatoirement, par

exemple en larguant les capteurs depuis un avion ou un drone. Comme les caméras ont un angle de vue limité, le risque est grand que des nœuds tombent dans une mauvaise position (pointant vers le sol, le ciel, d'un même côté q'un autre, etc). Certains articles (Tezcan et Wang, 2008) considérent des caméras motorisées pour qu'elles puissent être orientées correctement après déploiement.

Vigilance environnementale

Les réseaux de capteurs de vision sont aussi utiles pour la vigilance environnementale. Comme les nœuds consomment très peu d'énergie, ils peuvent être déployés dans des endroits stratégiques pour de longues périodes de temps (de l'ordre de plusieurs mois), afin d'obtenir des images d'intérêt scientifique sur de larges zones géographiques, par exemple près des nids, des abreuvoirs et réserves d'eau naturelles. Grâce à cela, les observateurs de la nature peuvent étudier le comportement et les habitudes des diverses espèces animales, en obtenant des scènes de lieux, qui peuvent être très éloignées, sans avoir besoin de se déplacer physiquement. En outre, des espèces naturellement timides qui rejettent la présence humaine pourraient être étudiées à travers les caméras, permettant son étude et l'ouverture de nouvelles portes à la science. Des systèmes de support pour les tâches de gardes de parcs, comme la détection des incendies de forêt, pourraient être développés.

Quelques expériences peuvent être signalées pour ce type d'application. Par exemple, un réseau de capteurs infrarouges Cyclops a été déployé dans la réserve des Montagnes James San Jacinto (Californie), comme rapporté dans (Srivastava *et al.*, n.d.), dans les nids d'oiseaux pour les étudier au cours de la saison de nidification. Ils ont aussi utilisé ces capteurs pour des études herpétologiques. De même, dans (Wawerla *et al.*, 2008), un système composé de caméras sans fil a été utilisé pour étudier le comportement des ours grizzly dans le parc Ni'iinlii Njike, en Yukon, Canada, juste en-dessous du cercle arctique.

Sûreté et sécurité de zones sensibles

Evidemment, les réseaux de capteurs de vision peuvent être appliqués pour la sécurité des zones privées et publiques. Néanmoins, le véritable intérêt des réseaux de capteurs de vision sans fil n'est pas dans la surveillance des établissements fermés (par exemple, des industries, des bureaux, des magasins commerciaux, des résidences, etc.). Pour ce type de demande, une longue liste de produits est disponible sur le marché. Des webcams rotatives sans fil, micro-caméras et autres dispositifs existent déjà pour la vidéo-surveillance. Ils s'appuient sur des méthodes de compression et des protocoles de communication normalisés, et des technologies de transmission à haut débit filaires ou sans fil.

Des systèmes de surveillance sans fil et limités en énergie pourraient être mis en place pour protéger des parcs, des zones sauvages, et d'autres zones liés a la protection des ressources naturelles, avec des caméras capables de dénoncer des chasseurs illégaux qui traversent les clôtures pour s'infiltrer dans les zones interdites, par exemple. D'autres applications pourraient être trouvées dans la surveillance des lieux privés ouverts, comme dans les industries forestières ou agricoles, par exemples.

Suivi du trafic routier

Des réseaux de capteurs de caméras peuvent être déployés pour le suivi et le contrôle de la circulation routière. Des algorithmes d'analyse d'images peuvent être utilisées pour faire le dénombremment des véhicules ou des personnes pour estimer le niveau de trafic en fonction des heures de la journée.

Applications a la robotique

Dans (Bae et Voyles, 2006) un réseau de caméras sans fil est employé pour des robots miniatures dans des applications de recherche et de sauvetage dans les zones urbaines. Dans cette expérience, une série de petits robots de capacité limitée se déplace dans une zone sinistrée. Un robot équipé d'une caméra joue le rôle de source, en enregistrant des séquences d'images et en les transmettant vers le puits à travers de multiples autres nœuds (robots) qui se déplacent dans la région. Comme les robots sont constamment en mouvement, et que leurs déplacements peuvent être rapides et imprévisibles, les auteurs ont mis en œuvre un système de routage pour le transport des images brutes. La communication était basée sur la technologie Bluetooth pour avoir une faible latence et éviter les collisions.

Une autre expérience est exposée dans (McCormick *et al.*, 2006). Ici, un réseau de caméras permet la surveillance, le suivi et le contrôle d'agents mobiles (robots) avec peu ou aucune capacité de perception de son environnement. Les auteurs mettent un ensemble de caméras dans le plafond d'une chambre, toutes pointées vers le sol. Grace à un système distribué de localisation et une communication réciproque entre les robots et le réseau, un robot est en mesure de connaître sa position et de corriger sa direction courante pour s'orienter vers sa destination.

2.2 Dispositifs de capture d'image

Les capteurs d'image de faible consommation d'énergie ont fait l'objet de grands progrès au cours des dernières années. En réponse à la forte demande du marché, nous pouvons trouver des capteurs d'image de plus en plus petits et de résolutions de plus en plus grandes, destinés principalement à être intégrés dans les téléphones portables, les ordinateurs portables, et les PDAs. Cependant, ces dispositifs sont dotés de ressources importantes en termes de mémoire et vitesse de calcul. Les dévelopements ont été centrés sur l'offre de meilleures qualités d'image et de taux de compression plus élévés puisque la demande des utilisateurs porte surtout sur ces aspects. L'autonomie en énergie est aussi importante, mais c'est de l'ordre de la journée. Dans les réseaux de capteurs sans fil de vision, comme dans les réseaux de capteurs en général, le problème de la consommation d'énergie est d'un tout autre ordre de grandeur, les nœuds devant avoir une autonomie de l'ordre du mois, voir de l'année. Dans beaucoup d'applications, la résolution des images n'a pas besoin d'être très élevée. Pour compter des œufs dans un nid par exemple, une image de 64×64 pixels sur 16 ou 32 niveaux de gris est suffisante. Selon (Cao *et al.*, 2005), les nœuds de capteurs d'image doivent avoir une capacité de calcul et de mémoire très importante, répondre à des contraintes de temps réel, et avoir un *transcepteur radio* haut débit, tout en consommant peu d'énergie. Les dispositifs disponibles actuellement ne sont pas encore capable d'atteindre ces niveaux d'exigence.

Plusieurs auteurs fondent leurs prototypes de capture d'image sur des composants commerciaux de faible consommation d'énergie. Par exemple (McCormick *et al.*, 2006) ont utilisé un capteur d'image ADCM 1670 [1] de Agilent pour fournir des capacités de vision dans une application de pistage d'objets. (Downes *et al.*, 2006), de l'Université de Stanford, ont utilisé le même capteur d'image dans leur architecture integrée de capteurs sans fil qui comprend un module de capteur d'image, de traitement de données et de communication tout en un. Le « mote » est constitué d'un microprocesseur ARM7 32 bits

[1] *Agilent ADCM-1670 CIF Resolution CMOS Camera Module, UART output* (2003). Datasheet. Agilent Technologies, Inc.. `http://www.agilent.com/`.

d'Atmel, avec 64ko de mémoire RAM et 256Ko de mémoire flash, un module radio Chipcon CC2420 qui est basé sur le standard 802.15.4, et plusieurs interfaces. Une autre architecture de capteur d'image sans fil est proposée dans (Cao *et al.*, 2005). Celle-ci comprend un transcepteur radio Chipcon CC1000, un processeur S3C44BOX de chez Samsung, une caméra VGA, une SDRAM comme mémoire principale pour le processeur, une SRAM pour le module de traitement d'images et de la mémoire flash. L'article discute aussi de l'application de méthodes de compression. Les auteurs comparent les algorithmes EZW, SPIHT et SPECK. Ils concluent que SPECK est celui qui peut représenter la meilleure option car il est beaucoup moins complexe que les autres, et pourtant mieux adapté aux dispositifs avec de contraintes de ressources. (Köppe *et al.*, 2004), de l'Université de Berlin, ont proposé un dispositif intégrant un module de capture d'image C328-7649 de COMedia, avec ses nœuds de capteurs ESB, avec un chip de caméra VGA, un module JPEG et une interface série, en vue de l'extension de la platforme ScatterWeb [2]. Des autres exemples de capteurs d'image conçus pour les réseaux de capteurs sans fil ont été décrits par (Ferrigno *et al.*, 2005) et (Karlsson *et al.*, 2007).

La table 2.1 montre une comparaison de quelques dispositifs de capture d'image commerciaux répresentatifs de la bibliographie des réseaux de capteurs de vision.

Tab. 2.1: Comparaison de dispositifs commerciaux utilisés sur les applications des réseaux de capteurs de vision.

	ADNS-3060 [3]	ADCM-1700 [4]	C328-7640 [5]	Quickcam Pro 4000 [6]
Fabricant	Agilent	Agilent	COMedia	Logitech
Résolution maximale	30×30	CIF (352×288)	VGA (640×480)	VGA (640×480)
Images par seconde	6400 fps (programmable)	15 fps (avec rés. CIF)	0.75fps	30 fps
Connectivité	20 pins	Connector 18 pins	Connector 4 pins	USB
Consommation de puissance	198mW, 6469 fps, 24MHz clock	42mW typiquement, sortie CIF, 13 MHz clock	198mW	>600mW
Tension d'alimentation	3.3V	2.65V - 3.1V	3.3V	∼5V

Il y a quelques années, certains groupes se sont consacrés au développement de dispositifs de caméra specifiquement conçus pour les réseaux de capteurs sans fil. La plupart ont suivi la philosophie de modularité des composants technologiques adoptés par les fabricants commerciaux de nœuds de capteurs

[2] `http://www.scatterweb.com/`

[3] *Agilent ADNS-3060 High-performance Optical Mouse Sensor* (2004). Datasheet. Agilent Technologies, Inc.. `http://www.agilent.com/`. October.

[4] *Agilent ADCM-1700-0000 Landscape CIF Resolution CMOS Camera Module* (2003). Datasheet. Agilent Technologies, Inc.. `http://www.agilent.com/`. November.

[5] *C328-7640 JPEG Compression VGA Camera Module*. Datasheet. COMedia Ltda.. `http://www.comedia.com.hk/`. November.

[6] `http://www.logitech.com/`

(c'est-à-dire le développement de cartes de capteurs distinctes des unités de radio/processeur). Ils ont développé des capteurs d'image de faible consommation pour être compatible avec les motes disponibles dans le marché. La plupart utilisent pour la partie caméra des composants commerciaux existants, et ils intègrent un processeur de faible consommation et de la mémoire. Les autres on préféré développer eux-mêmes des nouveaux composants et expérimenter de nouvelles approches au niveau du capteur lui-même. Finalement, quelques auteurs ont construit des capteurs d'image sans fil en combinant de très complexes technologies materielles, par exemple des webcams ou des cartes Wi-Fi, en permettant ainsi la transmission de flux vidéos grâce a une haute vitesse de traitement et de haut débit de transmission. Plus de détails des principaux représentants de ces tendances sont donnés par la suite.

2.2.1 Caméras basées sur des composants commerciaux

Plusieurs nœuds et cartes de capture d'images pour des applications de réseaux de capteurs sans fil ont été developpés en utilisant des composants électroniques commerciaux (dit en anglais composants COTS par *Commercial Off-The-Shelf*), par exemple, des caméras CMOS, des microcontrôleurs, des mémoires, ... Un exemple est MeshEye (Hengstler et Aghajan, 2006; Hengstler *et al.*, 2007). C'est un mote-caméra intelligent conçu pour la surveillance distribuée. La mote MeshEye intègre un microcontrôleur Atmel AT91SAM7S, une mémoire flash MMC/SD, deux capteurs ADNS-3060 originalement utilisés pour des souris optiques (le mote permet jusqu'au huit de ces capteurs), une caméra CMOS ADCM-2700 et un transcepteur radio CC2420 de Texas Instruments qui respecte la norme 802.15.4. Cet mote peut être alimenté via une interface mini-usb ou par des batteries standard AA.

Dans (Swarm-Intelligent Systems Group, 2004) on peut trouver une carte-caméra conçue pour pouvoir être enfichée sur les motes Micaz de Crossbow (Crossbow Technology Inc., n.d.).Une autre carte-caméra est exposée dans (Kleihorst *et al.*, 2006), qui utilise un module de communication Bluetooth, et qui permet l'intégration de deux capteurs d'image VGA.

Une étape importante a été franchie avec la naissance de la caméra Cyclops (Rahimi *et al.*, 2004; Rahimi *et al.*, 2005) développée par une équipe de l'UCLA. De même que l'apparition des nœuds Mica (Hill et Culler, 2002*a*; Hill et Culler, 2002*b*), aujourd'hui développés et distribués par Crossbow, a permis à la communauté scientifique internationale de disposer d'un support d'expérimentation de référence, les caméras Cyclops ont ouvert un grand espace de recherche dans le domaine des réseaux de capteurs d'image.

La caméra Cyclops a été développée par les laboratoires Agilent et le CENS (Center for Embedded Network Sensing) de l'UCLA. Elle permet la capture et le traitement d'images de faible résolution avec une relativement faible consommation d'énergie. Quatre versions de Cyclops ont été développés : Cyclops1, Cyclops2A, Cyclops2B et Cyclops2C (seule cette dernière version est traitée dans cette thèse, parce qu'elle intègre des améliorations significatives par rapport aux versions plus anciennes). Cyclops est composé d'un module CMOS de capture d'images de moyenne qualité ADCM-1700 (Agilent), un microcontroller ATmega128L de Atmel avec 128Ko de mémoire flash pour le stockage du code d'application et 4Ko de mémoire RAM (la même qui est utilisée dans les motes Mica de chez Crossbow), un CPLD XC2C256 CoolRubber de chez Xilinx, une mémoire SRAM TC55VCM208A de Toshiba avec 64Ko et une mémoire flash AT29BV040A d'Atmel pour le stockage de données. Cyclops a aussi un connecteur de 51 pins que lui permet d'être attachée à un mote Mica2 ou Micaz de Crossbow.

Le microcode de la caméra Cyclops a été écrit avec le langage de programmation NesC/TinyOS[7] et l'intégralité des sources est disponible sur le site du CENS de l'UCLA[8].

La dernière version de Cyclops supporte différentes dimensions d'image, qui peuvent être sélectionées a volonté. La résolution maximale pour le capteur ADCM est CIF (352×288), mais le microcode programmé par défaut limite la taille maximale à 128 × 128 pixels, dû probablement à des restrictions du materiel. Elle peut générer des images de trois formats différents : monochrome codé 8bpp, couleur RGB (24 bits) et YCbCr couleur (16 bits). La carte Cyclops peut aussi avoir différents états, qui défissent l'énergie consommée par la plateforme. Par exemple, quand on capture une image, Cyclops consomme 42mW, 0.7mW quand on est en mode endormi et moins de 1μW en état OFF. Afin d'économiser de l'énergie, Cyclops est capable d'utiliser des ressources ou de les libérer selon les besoins du moment.

Cyclops est une bonne initiative dans la route vers un capteur d'image peu gourmand en énergie. Cependant, ces consommations d'énergie pourraient être trop importantes pour des applications de réseaux de capteurs de long durée. En outre, Cyclops présente des contraintes surtout dûes à la faible vitesse du processeur qui induit des temps de traitement assez longs. Malheureusement, la caméra Cyclops a été sortie du marché et n'est plus disponible pour l'achat depuis la fin de 2007.

Une autre carte caméra est la CMUcam3 (Rowe *et al.*, 2007), la troisième version des systemes CMUcam (toute l'information à propos du projet est disponible sur `http://www.cmucam.org/`). Essentiellement, la CMUcam3 est composée d'une caméra CMOS Omnivision OV6620 ou une OV7620 comme module de capture d'image, en permettant la capture d'images en résolution CIF aux formats couleur RGB et YCbCr, un microcontroleur NXP LPC2106 (ARM7TDMI de 60MHz) avec 64Ko de mémoire RAM et 128Ko de mémoire flash, et un *frame buffer* de vidéo FIFO Averlogic AL4V8M440 de 1Mo.

Cet dispositif peut être connecté a une mote Telos (Polastre *et al.*, 2005) de Berkeley ou à un module Tmote Sky [9].

Afin d'obtenir des économies d'énergie, la caméra peut travailler selon trois modes d'opérations : *active*, *idle* et *power down*. Cependant, la consommation d'énergie de la CMUcam est bien plus grande que celle de la caméra Cyclops. Le bénéfice réel de la CMUcam est dans la rapidité de traitement. L'algorithme traditionel JPEG peut être calculé sur une image en résolution CIF en environ 0.82 secondes. Une API basé sur C appelé `cc3` a été mis à disposition pour fournir des abstractions pour la programmation de la caméra. Plusieurs composants ont été développés (`cc3` et des autres codes sont disponibles sur le site du projet CMUcam).

Comme on peut le remarquer, la plupart des caméras basées sur de composants commerciaux utilisent des microcontroleurs de faible consommation, pour obtenir des économies d'énergie en sacrifiant à la vitesse de calcul et la capacité de stockage. L'ajout de mémoire externe est devenu une nécessité. L'intégration de processeurs de signaux numériques (DSP, *Digital Signal Processor*) ou de circuits logiques programmables (FPGA, *Field-Programmable Gate Array*) en est encore au début. Un exemple récent est la plate-forme introduite par (Karlsson *et al.*, 2007), qui est composée d'un module radio-microcontrôleur Flck-3, un DSP et une carte caméra. Cet type de dispositif permet une haute vitesse de traitement de données, inaccessible avec d'autres types d'architecture.

[7] *TinyOS : An operating system for networked sensors* (2004). UC Berkeley. `http://www.tinyos.net/`.
[8] *CENS - CVS Repository* (2004). Center of Embedded Network Sensing. `http://cvs.cens.ucla.edu/`.
[9] *Tmote sky - low power wireless sensor module* (2006). Datasheet. Moteiv Corporation. `http://www.moteiv.com/`.

2.2.2 Caméras conçues spécifiquement pour les réseaux de capteurs sans fil

Assurément, la conception basée sur des composants commerciaux prévaut dans le développement de capteurs d'image sans fil. Ils fournissent plus d'abstraction et de facilité d'utilisation, au prix de plus de consommation d'énergie. D'énormes économies d'énergie peuvent être obtenues en appliquant des approches conception, comme exposés dans (Culurciello et Andreou, 2006). Les auteurs présentent ALOHA, un capteur prototype CMOS conçu spécifiquement pour des applications de réseaux de capteurs. ALOHA intègre une représentation de l'information basée sur des événements. L'idée de base est simple : des événements sont exécutés quand des pixels individuels atteignent un seuil de tension déterminé. Le pixel exécute une requête au circuit récepteur, en manipulant son addresse sur le *bus* par activation d'une cellule ROM sur l'intersection des ligne et colonne. La technique d'accès ALOHA (Abramson, 1970) (et ceci explique la raison du nom), originalement utilisé pour des réseaux d'ordinateurs, est utilisé pour résoudre les accès multiples au niveau du bus. Ceci permet l'accès au bus quand des données sont disponibles. Des techniques comme la reconstruction d'histogrammes et *intra-événements* peuvent être appliquées afin de reconstruire l'image originale (Culurciello *et al.*, 2003).

Jusqu'à présent, deux versions du capteur d'images ALOHA ont été développés : (1) l'ALOHAim1, avec une grille de 32×32 pixels (Culurciello et Andreou, 2004) et un détecteur de contention analogique pour les collisions, et (2) l'ALOHAim2, avec une grille de 64×64 pixels organisés comme 4 quadrants de 32×32 pixels indépendants (Culurciello et Andreou, 2006) et un détecteur de contention numérique.

La consommation de puissance signalée pour le capteur d'image ALOHA est d'environ 795μW pour ALOHAim1, en atteignant un taux effectif de mise à jour de 4.88Kfps, et 5.75μW pour l'ALOHAim2, en atteignant un taux effectif de mise à jour de 2.44Kfps. Les propriétés d'échelle de la consommation de puissance du capteur ALOHA (P_N) sont liées au nombre de pixels (N), et ceci peut être estimé en appliquant la formule $P_N = P_{N_0} \log_2 \left(1 + \frac{N}{N_0}\right)$, où P_{N_0} est la consommation de puissance d'un capteur ALOHAim1 et N_0 est le nombre de pixels (32×32).

Certaines évaluations pratiques ont été rapportées pour cette plate-forme. En (Teixeira *et al.*, 2005), la latence et la consommation de puissance ont été évalués pour un nœud simple composé d'un ALOHAim1 et d'un mote Mica2. Les auteurs montrent de la dissipation de puissance d'environ 111mW et 60.4mW pour le mote Mica2, pendant les phases de transmission et collecte d'images, respectivement. Avec ceci, les durées théoriques initialement anoncées par l'utilisation du capteur ALOHA sont baissées de plusieurs mois à quelques jours, dû a la consommation des nœuds sans fil. L'impact du nombre d'événements est aussi commenté quand un deuxième nœud, composé d'une mote Mica2dot et une ampoule incandescente à faible consommation est activée pendant la phase de capture afin d'incrémenter la qualité des images résultantes.

ALOHA intègre une nouvelle philosophie dans le domaine du développement des réseaux de capteurs, selon laquelle l'information est captée et transmise seulement quand il y a besoin, et tout le traitement des données (en principe) est exécuté au niveau du capteur materiel lui-même, ce qui permet des économies d'énergie très importantes en laissant de côté la nécessité d'intégrer des composants supplémentaires comme c'est le cas pour les architectures commerciales. (Teixeira *et al.*, 2006) ont motivé ce type de capteurs d'image basé sur des événements, en présentant des premiers résultats sur la modélisation et l'évaluation des capteurs d'image dans le contexte des réseaux de capteurs sans fil.

2.2.3 Plate-formes de capteurs de vidéo

La transmission de vidéo sur réseaux de capteurs n'a pas été largement étudiée jusqu'à présent. Quelques auteurs ont proposé l'intégration d'architectures complexes basées sur de composants COTS de très haut niveau comme des cartes électroniques avec des systèmes d'exploitation embarqués et des webcams. L'une de ces plate-formes est Panoptes (Feng *et al.*, 2005*a*; Feng *et al.*, 2005*b*), une plate-forme de capture de vidéo basé sur des composants commerciaux. Panoptes a été développé dans deux versions différentes. La première, initialement rapportée dans (Feng *et al.*, 2005*b*), est basée sur une carte Applied Data Bitsy, en utilisant un processeur Intel StrongARM de 206MHz, tandis que la deuxième (Feng *et al.*, 2005*a*), utilise la plate-forme Stargate de chez Crossbow. Les deux versions utilisent une webcam USB, et intègrent une architecture logicielle intégrée sur un noyau Linux embarqué, de la compression par JPEG, du filtrage, et un méchanisme de *streaming* dynamique basé sur des priorités. En plus, le standard 802.11 a été utilisé pour la communication sans fil.

En termes de consommation d'énergie, comme Panoptes est complètement programmable, la plate-forme vidéo est capable de travailler selon différents états de système. Dans l'étape de capture, la consommation d'énergie atteint, en moyenne plus de 5 Watts, et 58mW en mode endormi (version Data Bitsy). C'est énorme en comparaison avec la consommation d'énergie rapportée pour les systèmes Cyclops (section 2.2.1) et ALOHA (section 2.2.2). Ceci laisse Panoptes (et des autres dispositifs) hors de possibilité d'être appliqué sur des réseaux de capteurs contraints en énergie. Avec ces exigences en matière de puissance, la source d'alimentation ne peut pas être indépendante (par exemple, en appliquant des batteries de faible tension) pour être employé sur une zone difficile d'acces pour de longues périodes de temps.

Comme Panoptes, quelques autres prototypes de capture vidéo focalisés sur des applications de petite échelle sur des environements très contrôlés et basés sur des composants commerciaux ont été conçus jusqu'à présent. Un exemple est le nœud Meerkats (Margi *et al.*, 2006) développé dans le Département de Génie Informatique de l'Université de Californie. Meerkats a des caractéristiques similaires au nœud Panoptes. Il est composé d'une carte Stargate de chez Crossbow (qui intègre un système Linux embarqué) armé avec une carte sans fil PCMCIA 802.11b et une webcam QuickCam Pro 4000 de chez Logitech. Afin d'alimenter la plate-forme et de lui fournir d'une certaine autonomie, les auteurs ont incorporé une batterie de Lithium-Ion de 2-cellules personnalisées, capable de fournir 7.4 Volts et 1000mAh.

Ce type de plate-formes répond sans aucun doute aux besoins des applications vidéo en termes de débit de communication et de vitesse de traitement d'image. Cependant, elles ne répondent pas aux contraintes de consommation d'énergie. Même si on peut observer une certain économie d'énergie en comparaison avec des autres systèmes de surveillance commerciaux, ces économies ne sont pas encore suffisantes pour envisager leur déploiement en pratique.

2.3 Traitement d'images dans les réseaux de capteurs

Comme le transcepteur radio est l'un de composants électroniques les plus gourmands en énergie dans un nœud de capteurs (Akyildiz *et al.*, 2002) il est clair qu'un moyen d'économiser de l'énergie est de réduire la quantité de données à transmettre. Une solution a été exposée dans la section 2.2.2 au niveau du capteur matériel. Cependant, la solution la plus évidente est la compression de données. En effet, moins il y aura de données à transmettre et moins le tranceiver radio consommera d'énergie. Nous

allons voir que ce raisonnement n'est pas toujours valable car la compression a un coût d'énergie qui peut être très élevé. Le problème est de disposer d'un algorithme de compression de données qui soit peu gourmand en énergie et qui ait un bon rapport débit-distorsion (Ferrigno *et al.*, 2005). Une longue liste d'algorithmes de compression est aujourd'hui disponible (Salomon, 2004), dont plusieurs concernant la compression d'images. Il existe dans la littérature, par exemple, différentes méthodes de compression basées sur des approches vectorielles dans le domaine des ondelettes (par exemple (Fischer, 1986; Jeong et Gibson, 1993; Guillemot *et al.*, 2008)) qui permettent d'atteindre à bas débit de meilleures performances en termes de qualité de reconstruction que les approches scalaires. Cependant même lorsque leur complexité calculatoire est réduite (comme par exemple dans (Gaudeau *et al.*, 2008)) ces méthodes semblent encore trop coûteuses pour une implantation dans le contexte des réseaux de capteurs. Certainement, la compression de données n'est pas le sujet le plus nouveau, nous pouvons voir aujourd'hui une grande évolution dans ce domaine. Cependant, la limitation de ressources des nœuds de capteurs, comme la mémoire ou la vitesse des processeurs, rend inapplicables la plupart des algorithmes de compression existants utilisés dans l'informatique traditionelle (Kimura et Latifi, 2005).

Dans la litérature des réseaux de capteurs sans fil, plusieurs algorithmes ont été proposés pour la compression de données. Nous avons par exemple quelques approches de compression distribuée (Kusuma *et al.*, 2001; Pradhan *et al.*, 2002), Data Funneling (Petrovic *et al.*, 2003), pipelined in-network compression (Arici *et al.*, 2003) ou bien S-LZW (Sadler et Martonosi, 2006). Dans cette section, nous concentrons notre étude dans les techniques de compression d'image spécifiques aux réseaux de capteurs. Pour que la compression de données soit rentable à la source, il faut que le traitement des données et la transmission des données compressées coûte moins d'énergie qu'un scénario en absence de compression. En effet, quelques auteurs (Ferrigno *et al.*, 2005; Wu et Abouzeid, 2004*b*) ont démontré que la complexité de certains algorithmes de compression conduisent à des consommations d'énergie plus importantes que la transmission simple d'une image sans aucun traitement. L'un des premiers travaux qui considèrent le compromis entre consommation énergétique par traitement et par communication a été présenté par (Maniezzo *et al.*, 2002). Les auteurs ont montré l'existence d'un nombre optimal de nœuds M impliqués dans une strategie de compression qui minimise la consommation globale de puissance. Plus tard, (Ferrigno *et al.*, 2005) ont présenté un travail plus approfondi, dans lequel ils évaluent plusieurs algorithmes de compression traditionnels en analysant le compromis entre energie consommée et traitement de données.

Plusieurs propositions semblent montrer une certaine attraction pour les propriétés de la célèbre transformée en ondelettes (Mallat, 1999). Ceci est principalement dû au fait que les ondelettes permettent la décomposition des images en plusieurs niveaux de résolution. Traditionellement, dans le domaine du traitement du signal, l'élimination des coefficients résultants de haute résolution implique l'élimination du bruit. Dans le traitement d'images, cela implique l'élimination des détails dans les images résultantes. Comme certains coefficients sont plus importants que d'autres dans la reconstruction de l'image finale, une transmission basée sur des priorités ou par chemins multiples peut être mise en œuvre. Des exemples d'application d'ondelettes pour la transmission d'images sur réseaux de capteurs de vision sont donnés dans (Wu et Chen, 2003; Yu *et al.*, 2004; Wu et Abouzeid, 2004*a*; Wu et Abouzeid, 2004*b*; Wu et Abouzeid, 2005; Wu et Abouzeid, 2006).

La figure 2.3 illustre une classification générale pour les algorithmes de traitement et compression d'images que nous avons trouvé dans la litérature.

En principe, les méthodes de traitement traditionelles ont été conçues pour travailler localement sur

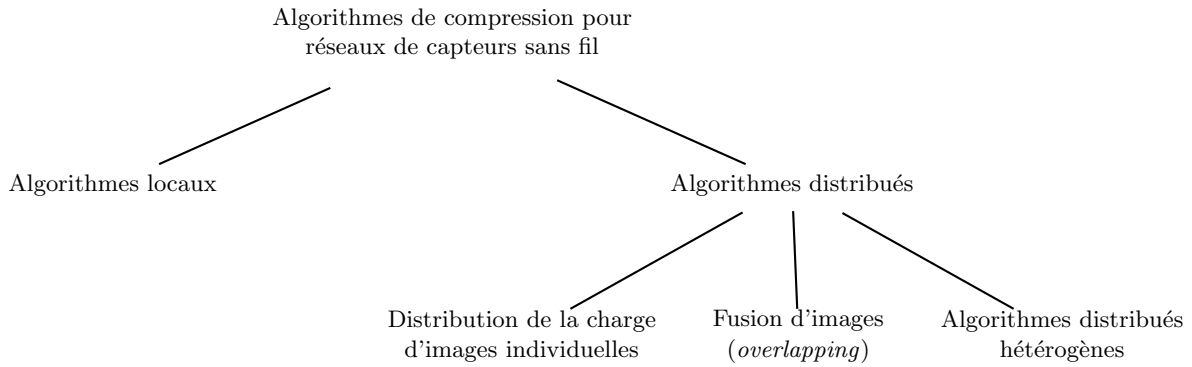


FIG. 2.3: Classification des algorithmes de compression d'image pour réseaux de capteurs sans fil

un seul et même processus. Dans les réseaux de capteurs, cela implique l'exécution des calculs au niveau du nœud source seulement (ou dans un seul nœud). D'autre part, la nature distribuée des réseaux de capteurs nous permet d'imaginer des approches qui considèrent la distribution du traitement de données entre plusieurs capteurs. D'une manière génerale, deux types d'algorithmes de compression distribués ont été rapportés. Le premier est basé sur la distribution du processus de compression d'une image a travers plusieurs nœuds, et le deuxième est basé sur la corrélation existante entre deux ou plus images principalement quand elles contiennent des scènes voisines. Quelques approches hétérogènes pourraient combiner les deux stratégies.

2.3.1 Compression locale

Dans le domaine des réseaux de capteurs sans fil, quelques propositions considèrent la compression au niveau de la source, c'est-à-dire par des algorithmes locaux, sans distribution de la charge de traitement de données avec d'autres nœuds (par exemple, S-LZW). La compression va servir à réduire le volume des données que la source aura à transmettre. Le traitement de données à la source est aussi nécessaire pour anticiper des possibles pertes d'information pendant la transmission de paquets jusqu'au nœud puits, ou pour contrôler la quantité de données à envoyer (et par incidence la qualité de l'image) selon les conditions du réseau.

Il existe plusieurs algorithmes de compression dans la litérature. Certains peuvent fournir des taux de compression élevés mais toutefois, ils ne sont pas applicables dans les réseaux de capteurs en raison des limitations des ressources des nœuds de capteurs. (Ferrigno *et al.*, 2005) ont présenté une plate-forme pour évaluer les performances de plusieurs algorithmes traditionnels de compression d'images sur un nœud de capteur. Ils ont analysé cinq algorithmes bien connus : JPEG2000, SS, DCT, SPITH et JPEG. Les résultats montrent que pour JPEG2000, DCT, SPITH et JPEG, le coût d'énergie des calculs est supérieur au coût de transmission de l'image non compressée. Les résultats montrent que SS est le seul des algorithmes testés qui amène des économies d'énergie par rapport au cas sans compression. Il amène une réduction de la consommation d'énergie d'environ 29%. De toute manière, le coût d'énergie et le coût d'implantation (quantité de mémoire requise notament) dépendent des caracteristiques du matériel aussi. La plate-forme de Ferrigno *et al.* est basée sur un microcontrôleur PIC16LF877, qui est bien plus rapide, et bien plus gourmand en énergie que le microcontrôleur Atmega128L utilisé dans les capteurs d'image Cyclops (voir section 2.2.1).

Il est attendu de la compression locale des données plusieurs avantages :

Extension de la durée de vie du nœud source. En effet, moins la source aura de données à transmettre, et moins d'énergie elle consommera au niveau du transcepteur radio. Cette affirmation est vraie tant que la complexité de l'algorithme de compression adopté sera suffisamment fiable pour être rentable. Le processus de compression ne doit pas coûter plus cher en termes de consommation d'énergie que le gain qu'il amène sur la communication, sinon la présence d'un processus de compression pourrait diminuer la vie utile du nœud.

Extension de la durée de vie des nœuds intermediaires. Pour les mêmes raisons, la réduction de la quantité de données à la source sera nécessairement bénéfique pour les nœud chargés de relayer les paquets entre le nœud source et le puits. Ils recevront moins de paquets de données, donc ils auront moins de paquets et d'acquittements à transmettre.

Contribution à la diminution des congestions du réseau. Une diminution de la quantité de données circulant sur le réseau va entraîner une diminution des risques de congestion du réseau, donc une diminution des pertes de paquets et des retards de transmission.

Contribution à la tolérance aux pertes. Quelques renforcements sur la tolérance aux pertes de paquets peuvent être atteints par l'application de quelques mécanismes de traitement à la source, comme par exemple le mélange ou le tatouage d'images.

Quelques systèmes de compression locaux proposés pour les réseaux de capteurs de vision sont présentés ci-dessous.

Schéma basé sur le codage SPIHT

L'une des premières propositions pour la compression d'images dans les réseaux de capteurs a été introduite par (Wu et Chen, 2003). Les auteurs ont proposé un schéma basé sur le codage SPIHT (Said et Pearlman, 1996), des blocs de données sont générés par relation parent-enfant de coefficients d'ondelettes. Cette relation parent-enfant est effectuée afin de renforcer la robustesse de SPIHT aux erreurs de transmission. L'algorithme fonctionne comme suit : premièrement, l'image capturée est décomposée en multiple résolutions en appliquant une transformée en ondelettes discrète. Puis les coefficients d'ondelettes sont regroupés en fonction de leur relation parent-enfant comme montré figure 2.4. Chacun de ces groupes est codé indépendamment par l'algorithme SPIHT. De cette manière, les erreurs possibles lors de la transmission affecteront seulement le bloc erroné, en permettant alors la recontruction de l'image avec des pertes possibles d'information.

Les expériences ont été réalisées avec un processeur Intel StrongARM SA 1110 et un émetteur radio LMX3162. Les auteurs proposent un schéma de transmission basé sur RCPC/CRC pour permettre la protection aux erreurs. Les résultats montrent des économies d'énergie en termes de traitement de données et une réduction effective de la propagation des erreurs.

Compression locale par JPEG

Vu le nombre important d'algorithmes de compression qui sont utilisés dans l'informatique traditionnelle, il est évident que certains auteurs aient tenté d'utiliser les standards qui ont déjà fait leur preuve en termes de ratio débit/distorsion. La technique de compression d'image la plus répandue de nos jours

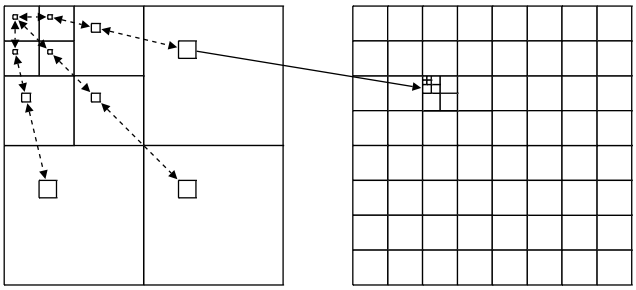


FIG. 2.4: Groupement des coefficients d'ondelettes en fonction de leur relation parent-enfant, comme proposé par (Wu et Chen, 2003).

est sans doute le standard JPEG. Il est basé sur un découpage de l'image en blocs de 8×8 pixels et sur la transformée en cosinus discrète (DCT). Les blocs sont ensuite quantifiés et codés avec RLE et Huffman. De toutes les étapes de l'algorithme JPEG, c'est la DCT qui coûte le plus en calculs. Le calcul classique de chaque coefficient DCT ($G_{i,j}$) pour un bloc de 8×8 est réalisé par :

$$G_{i,j} = \frac{1}{4}.C_i.C_j.\sum_{x=0}^{7}\sum_{y=0}^{7} x_{i,j}.\cos\left(\frac{(2x+1)i\pi}{16}\right).\cos\left(\frac{(2y+1)j\pi}{16}\right) \tag{2.1}$$

ou $C_f = \begin{cases} \frac{1}{\sqrt{2}}, & f = 0, \\ 1, & f > 0, \end{cases}$ and $0 \leq i, j \leq 7$.

Avec une telle complexité, il existe un besoin pour optimiser cette opération, pour la rendre plus rapide (on parle donc de DCTs rapides) et donc applicable sur des composants électroniques plus limités que les ordinateurs de nos jours, comme les appareils photo-numériques, les PDAs, et, bien sûr, les nœuds de capteurs d'image sans fil. Un exemple récent est l'approche de (Lee *et al.*, 2007*a*), qui ont adopté l'algorithme JPEG en utilisant l'algorithme LLM (Loeffler *et al.*, 1989) pour calculer la DCT dans un mode à virgule non flotante. Ils calculaient le nombre de bits minimums pour représenter les parties entières et décimales des valeurs réelles. Ce travail est intéréssant, mais des contraintes du temps ne nous ont pas permis de le considérer pour comparaison dans cette thèse.

Par ailleurs, pour diminuer la quantité de calculs à faire sur chaque bloc de coefficients DCT, (Mammeri *et al.*, 2008) proposent l'application d'un algorithme connu comme *Triangular JPEG* (T-JPEG). A lieu de traiter un bloc de coefficients DCT entier de $k \times k$ coefficients ($k = 8$ pour le cas de JPEG traditionnel), ils vont sélectionner une région réduite et représentative du bloc, qui correspond à une région triangulaire au coin haut-gauche du bloc $k \times k$, de longueur de cathète ρ, avec $\rho \leq k$, en accord avec le parcours en zigzag des coefficients AC du bloc, comme schématisé figure 2.5.

En procédant comme ceci, la quantité de coefficients a traiter passe de k^2 à $C_\rho = \frac{\rho.(\rho+1)}{2}$. La méthode coûte donc moins cher en énergie.

Compression locale par JPEG2000

JPEG2000 (Christopoulos *et al.*, 2000) est un algorithme de compression d'images basé sur un processus très complexe qui inclut une transformée en ondelettes dyadique, une allocation de bits, une quantification et un codage entropique. (Wu et Abouzeid, 2004*b*) ont introduit une technique de faible consommation énergétique qui incorpore le standard JPEG2000 pour la compression d'images depuis un

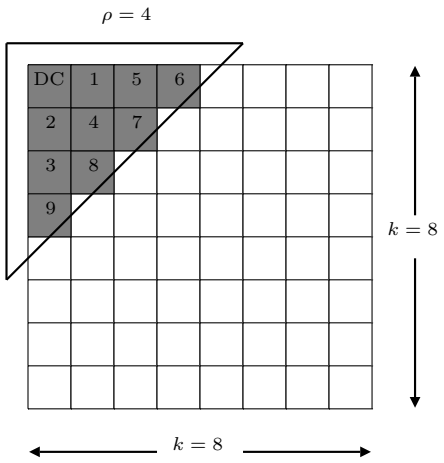


FIG. 2.5: Sélection de coefficients avec *Triangular JPEG* (Mammeri *et al.*, 2008).

nœud caméra sans fil. C'est l'un des premiers travaux dans lequel nous pouvons voir qu'il est consideré un scénario multi-saut pour réseaux de capteurs de vision. Ils ont formulé la transmission d'image comme un problème d'optimisation et ils ont proposé une heuristique appelé MTE (*Minimize Total Energy*). Ici, il est supposé qu'un nœud source a une connaissance du nombre (estimé) de nœuds intermédiaires entre lui et le puits, et chaque nœud est capable de connaître son état de batterie. L'algorithme travaille comme suit : premièrement, le nœud source vérifie une table avec des paramètres de quantification et niveaux d'ondelette pré-calculés pour un réseau déterminé et une qualité d'image requise. Le nœud de capteur compresse l'image capturée en appliquant la compression par ondelettes, puis, il calcul un taux de compression et une dissipation d'énergie. Une nouvelle compression est donc réalisée pour calculer des nouveaux taux de compression et dissipation d'énergie. Ces valeurs sont comparées avec les anciennes. Ces étapes sont répétées tant que les nouveaux taux de compression et de dissipation d'énergie sont meilleures ou jusqu'à un certain nombre d'itérations. Quelques alternatives sont discutées pour obtenir encore plus d'économies d'énergie, comme l'application de tables de recherche pré-calculées ou le calcul des paramètres optimaux dans un centre de traitement externe. On peut quand même voir que cette méthode pousse à compresser l'image plusieurs fois et on a du mal à voir comment la source va économiser l'énergie. En fait, cette méthode vise à optimiser l'énergie dans les nœuds de transit mais elle n'est pas bonne pour la source.

(Yu *et al.*, 2004) ont proposé une autre strategie de transmission d'images basé sur JPEG2000 pour des réseaux de capteurs point-à-point. En partant d'une distortion d'image attendue, une unité de contrôle est capable de déterminer le nombre de niveaux de qualité (couches) à être transmises et de trouver le schéma le plus efficace en énergie.

2.3.2 Compression distribuée

En raison de la nature distribuée des réseaux de capteurs, il était évident que bon nombre de travaux seraient menés sur la compression distribuée des images. Notons que plusieurs propositions ont été publiées sous le qualificatif de *compression distribué*, même s'ils ne realisent pas vraiment du traitement distribué, mais plutôt du traitement collaboratif (même pas dans plusieurs cas).

Dans la bibliographie des réseaux de capteurs de vision, deux approches différentes ont été cataloguées comme méthodes « distribuées ». Le premier groupe (le plus développé) profite de la corrélation existante entre deux (ou plus) images capturées par deux caméras différentes (principalement voisines), en cherchant à diminuer la quantité de données à transmettre depuis chacune. Une deuxième catégorie considère une approche plus exacte à la définition formelle du traitement distribué, en envoyant quelques zones d'une image à des nœuds différents pour qu'ils fassent, chacun, une partie du traitement.

Dans le reste de cette section nous allons décrire et discuter quelques uns des travaux trouvés dans la littérature sur le traitement distribué des images appliqué spécifiquement aux réseaux de capteurs sans fil.

Compression distribuée d'images corrélées

En 1973, Slepian et Wolf ont présenté leur théorème sur le codage de sources corrélées codées indépendamment, mais décodées toutes ensembles (Slepian et Wolf, 1973). Ces propositions de la théorie de l'information classique ont inspiré plusieurs des approches proposés pour le traitement colaboratif-distribué d'images sur réseaux de capteurs sans fil. La principale idée, ici, est de tirer avantage de la corrélation entre deux ou plus images d'une même scène mais originaires de plusieurs caméras. Cette corrélation est plus prononcée quand les caméras capturent des scènes similaires, et plus encore quand elles proviennent de sources voisines.

Plusieurs cas sont possibles. La figure 2.6 illustre le cas d'un ensemble de caméras voisines localisées dans une même ligne orientées perpendulairement à leur rangée, comme les nœuds A et B et C. Un autre cas de chevauchement d'images est obtenu par l'orientation des nœuds D, E et F, dont un objet peut être capturé au même moment par ces trois caméras lors qu'il se trouve à l'intersection de leur zone de couverture.

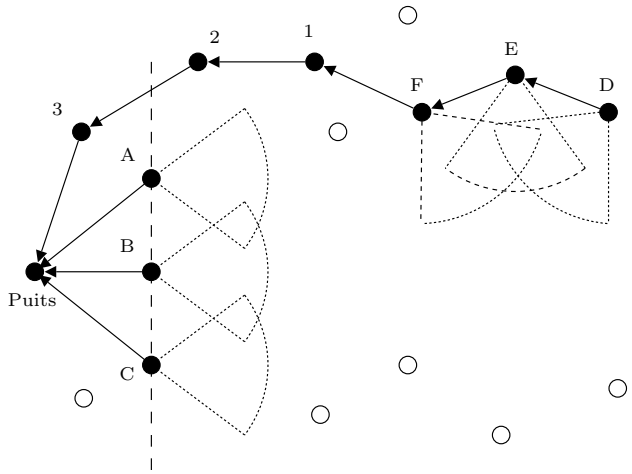


FIG. 2.6: Exemple de plusieurs nœuds caméra capturant des images corrélées.

Même si le théoreme de Slepian-Wolf a inspiré la plupart des travaux publiés sur la compression distribuée dans les réseaux de capteurs de caméras, son application n'est pas exempte de difficultés. En effet, (Wagner *et al.*, 2003) ont proposé un schéma pour des caméras multiples qui capturent différents points de vue de la même scène. Ici, un premier nœud de caméra prend une image d'une scène générale

et un ensemble de caméras secondaires ont chacune une vision chevauchée de la région de la première scène. La méthode est basée sur l'approche de contexte de forme présenté par (Belongie *et al.*, 2002). Elle permet la détermination des similarités entre deux images en représentant les formes des objets capturés comme un ensemble de points échantillonnés depuis ces contours, et par le calcul de la correspondance des points de fonction, en envoyant alors des versions de faible résolution des zones des images chevauchées determinées. Du coté du décodeur, un algorithme de *super-résolution* est utilisé afin de reconstruire une image de plus haute résolution de la region partagée.

Le cas des images corrélées a été largement étudié par Gehrig et Dragotti. 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 (Gehrig et Dragotti, 2004).

Compression distribuée sur une architecture clusterisée

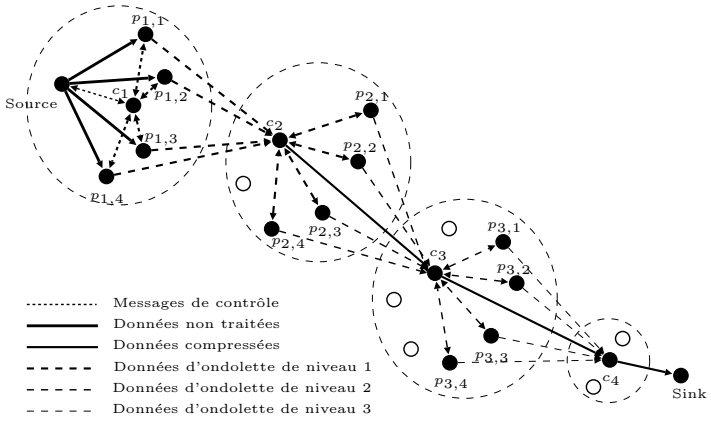
Dans (Wu et Abouzeid, 2004*a*; Wu et Abouzeid, 2005) la compression distribuée d'images en utilisant le standard JPEG2000 est proposée. L'idée de base est de répartir la charge de travail du calcul de la transformée en ondelettes entre les différents nœuds. Deux méthodes pour l'échange de données ont été proposées :

Méthode de transformée d'ondelettes parallèle : Dans la première méthode, l'application d'une transformée d'ondelettes parallèle (Marino *et al.*, 1999) est proposée. L'image capturée est divisée en n blocs de données $R_1, \ldots, R_n$, consistant en une ou plusieurs lignes. Ces blocs sont transmis à certains nœuds voisins de la source. Ces nœuds effectuent une transformée en ondelettes unidimensionelle (1-D) pour chaque bloc de données qui lui a été envoyé, puis transmet le résultat à un nœud aggrégat. Celui-ci divise les données obtenues, cette fois en m blocs $I_1, \ldots, I_m$ composé de colonnes. Puis il distribue ces blocs en les transmettant à des nœuds voisins. Ceux-ci effectuent alors la transformée en ondelettes 1-D et les renvoient au nœud aggrégat. Celui-ci récupère tous les blocs et obtient aussi le résultat de transformation bidimensionelle (2-D) en ondelettes de l'image.

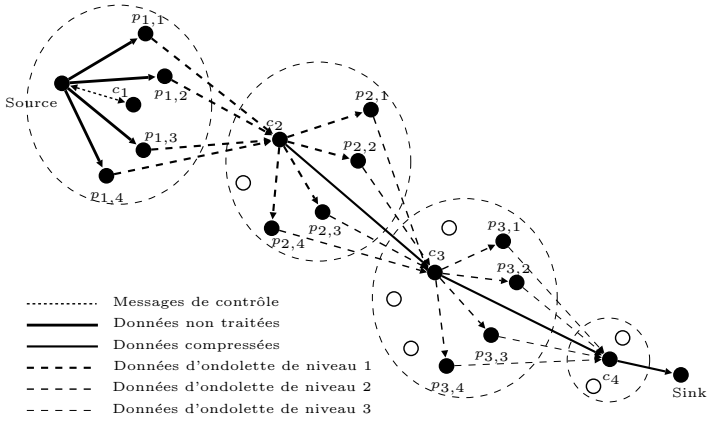
Méthode de carrelage : Dans cette deuxième méthode, l'image capturée est divisée en tuiles (de blocs de données), puis ces tuiles sont distribuées à un certain nombre de nœuds voisins. Ces nœuds effectuent la transformation en ondelettes 2-D de façon indépendante sur ces tuiles. Enfin, les résultats sont transmis à un nœud aggrégat.

Dans les exemples donnés pour JPEG2000, un organisation en clusters est supposée. Pour la première méthode (voir la figure 2.7(a)), la source divise l'image en n blocs de données et les transmet à n noeuds $p_{1,i}$ $(1 \leq i \leq n)$ sélectionnés par le premier nœud tête de cluster (*cluster head*). Les nœuds $p_{1,i}$ effectuent une transformée en ondelettes 1-D et envoient les résultats à la tête du groupe c_1. Les résultats sont agrégés, et répartis à nouveau en n blocs et ces blocs sont envoyés aux nœuds $p_{1,i}$. Ils effectuent une nouvelle transformation en ondelettes 1-D et relaient les résultats à la tête du groupe c_2. Dans ce schéma, chaque cluster effectue un niveau de transformée en ondelettes 2-D. La $l^{\text{ème}}$ tête de cluster c_l décompose les données correspondant à la sous-image LL_{l-1} en n blocs (composé de lignes) pour les n noeuds $p_{l,i}$. Ils effectuent la transformée en ondelettes 1-D et renvoient les données à c_l. Les données sont décomposées en n blocs de nouveau, et transmis aux nœuds $p_{l,i}$. Ils effectuent la deuxième transformation 1-D et retransmettre les paquets vers le prochain chef du cluster c_{l+1}. En même temps, le chef du cluster c_l code

les autres sous-images (LH, HL et HH) et les transmet à c_{l+1}, qui fait l'agrégation. Ce processus est répété pour chaque niveau de décomposition en ondelettes souhaité.



(a) JPEG2000 distribué. Méthode 1 : Division de l'image par lignes/colonnes.



(b) JPEG2000 distribué. Méthode 2 : Carrelage de l'image.

FIG. 2.7: Deux méthodes différentes pour l'application de JPEG2000 distribué.

De une manière similaire, pour la deuxième méthode (voir figure 2.7(b)), le nœud source décompose l'image capturée, mais cette fois-ci en n tuiles ou carreaux. Chaque tuile est envoyée aux nœuds $p_{1,i}$, qui effectuent la transformée en ondelettes 2-D. Ensuite, ils envoient les données à la prochaine tête de cluster. Dans cet exemple, l'unique fonction de c_1 est de sélectionner les nœuds $p_{1,i}$. Dans les autres clusters la procédure est analogue à la première. Chaque nœud exécute la transformée en ondelettes 2-D et envoie les résultats à la prochaine tête de cluster. Les informations correspondant aux sous-images de détail (LH, HL et HH) sont codées par le chef du cluster et transmises directement au suivant.

Cette méthode est motivée par le principe que, même si l'énergie totale nécessaire pour l'ensemble du système est augmentée, l'énergie nécessaire pour chaque nœud est réduite, ce qui allonge la vie utile du réseau (Wu et Abouzeid, 2004*a*). Nous pensons toutefois que la validité de cette affirmation reste à prouver. Les principaux problèmes de leur proposition sont que la source ne réalise aucune économie d'énergie, et donc sa durée de vie ne sera pas allongée. De plus, cette méthode nécessite d'utiliser un

protocole de communication fiable pour fonctionner correctement.

Un autre schéma de compression d'images pour les réseaux de capteurs est présenté dans (Lu *et al.*, 2008). Ce schéma est basé sur la LBT (*Lapped biorthogonal transform*), l'algorithme *zero-tree*, la quantification multiple et le codage de Golomb. La LBT était préférée à la DWT car elle nécessite beaucoup moins de calculs et d'espace mémoire. De même, le codage de Golomb était préféré au codage de Huffman ou au codage arithmétique pour les mêmes raisons. Lu *et al*, se sont inspirés de la méthode clusterisée décrite précédemment pour distribuer la charge de calcul et la consommation d'énergie d'un nœud source individuel. Grosso modo, la distribution de l'algorithme de compression marche comme suit : premièrement, le nœud source envoie un message de requête à l'un de ces voisins (S), qui la renvoie à sa tête de cluster. Le nœud tête de cluster choisit quelques nœuds dans son cluster qui ont une certaine quantité d'énergie minimale et envoie un message d'acceptation vers S, qui le renvoie vers le nœud caméra. Le nœud caméra envoie 8 lignes de données vers S qui les distribue aux nœuds choisis. Ces nœuds réalisent la compression et envoient les données compressées vers la tête du cluster.

2.4 Transmission d'images sur réseaux de capteurs

La plupart des recherches dans le domaine des réseaux de capteurs sans fil concernent les protocoles de communication à tous les niveaux qui doivent être efficaces en termes d'énergie. En dépit de la grande quantité de propositions de protocoles de communication pour les réseaux de capteurs sans fil (Karl et Willig, 2005), il y a encore peu de propositions pour le cas particulier de la transmission d'images. Quelques auteurs ont quand même proposé de nouvelles approches fondées sur l'idée que les nouveaux défis à relever dans la transmission d'images doivent être confrontés avec de nouveaux protocoles adaptés à ces nouvelles contraintes. Même si ces travaux montrent des approches intéressantes, plus de simulations et d'expérimentations sont nécessaires pour valider leur performances. Certains de ces travaux portent sur les FEC (*Forward Error Correction*), la demande de retransmission automatique ou l'application de RCPC/CRC, discuté dans (Wu et Chen, 2003), mais les détails sur leur mise en œuvre sont très difficiles à trouver. Des travaux plus récents, comme (Na *et al.*, 2008; Saxena *et al.*, 2008), commencent à sortir de cette approche purement théorique.

Dans cette section, nous présentons quelques travaux qui représentent l'état de l'art actuel sur les mécanismes de communication spécifiquement conçus pour les réseaux de capteurs sans fil multi-saut. Ils concernent des stratégies de routage, des méthodes de contrôle d'erreurs et des méthodes de contrôle de trafic.

2.4.1 Algorithmes de routage sur les réseaux de capteurs l'image

Les réseaux étendus ont besoin d'algorithmes optimaux et adaptatifs pour faire parvenir les données d'un émetteur à un récepteur qui ne sont pas dans le même voisinage. Comme ces nœuds peuvent être très éloignés géographiquement et les technologies de communication sont souvent limitées en portée de transmission, l'utilisation d'algorithmes de routage, qui rendent possible cette communication d'un point à l'autre à travers plusieurs nœuds intermédiaires, devient impératif. Le routage ad-hoc pour les réseaux sans fil à été dévéloppé pour les réseaux d'ordinateurs ou de véhicules. Nous pouvons trouver quelques exemples proposés spécialement pour la transmission d'images, comme le protocole SPIN-IT (Woodrow

et Heinzelman, 2002). Les réseaux de capteurs sans fil sont un cas très particulier des réseaux ad-hoc, différents en termes d'échelle et en termes de limitations de ressources. De nombreuses approches ont été proposées afin d'assurer une transmission efficace et de prolonger la durée de vie des réseaux (Akkaya et Younis, 2005; Dai *et al.*, 2005).

Nous pouvons classer les approches de routage pour les réseaux de capteurs sans fil en quatre grandes classes, comme le montre la figure 2.8.

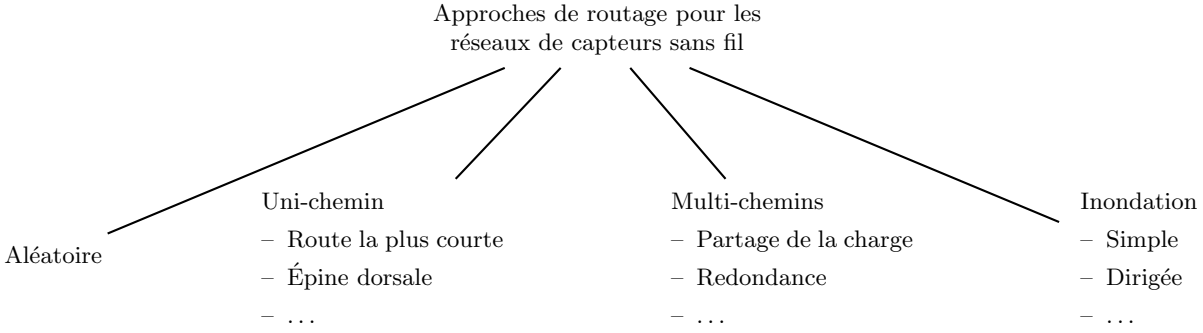
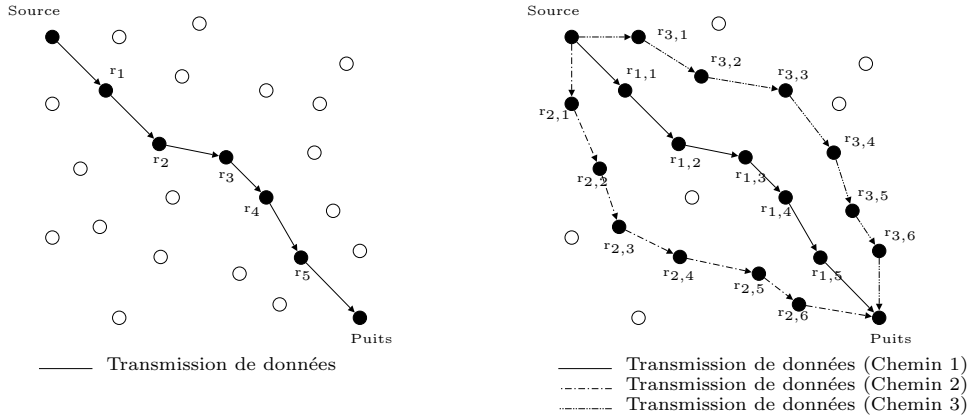


FIG. 2.8: Classification des algorithmes de routage pour les réseaux de capteurs sans fil.

Dans le cas d'une stratégie de routage aléatoire, le paquet de données est envoyé à un nœud du voisinage au hasard (c'est une marche aléatoire). On peut aussi se limiter aux nœuds qui remplissent certaines conditions, par exemple, ceux qui sont plus proche du puits que le nœud source. Le routage uni-chemin (*unipath*) définit une stratégie ou une et une seule route (peut être « optimale ») est sélectionnée avant la transmission, de manière à assurer l'arrivée des paquets par le « meilleur » chemin (voir figure 2.9(a)). Dans une approche multi-chemins (*multipath*), plusieurs chemins sont sélectionnées. À partir de là, soit les paquets sont envoyés de manière répétée par les différents chemins pour augmenter la probabilité d'arrivée au puits, soit les paquets sont envoyés alternativement sur un chemin parmi toutes les routes sélectionnées pour mieux répartir le trafic sur les nœuds du réseau (voir figure 2.9(b)). Finalement, différents types de routage par inondation peuvent être appliqués, en augmentant de la probabilité d'arrivée des paquets, au prix d'une augmentation de la charge du réseau.



(a) Routage uni-chemin.

(b) Routage multi-chemin.

FIG. 2.9: Exemples de différentes méthodes de routage pour les réseaux de capteurs.

(Lui et Lam, 2005) ont proposé un algorithme simple pour transmettre des images dans un réseau de

capteurs de vision. Un groupe de nœuds-caméra prend des photos à partir de perspectives différentes d'un même objet. Les auteurs proposent un système de transmission d'image pour permettre le chevauchement des images compressées et la sélection du chemin à suivre. Chaque nœud de caméra C_i, avec $1 \leq i \leq k$, est en mesure d'exécuter la capture d'image, la compression et le chevauchement. JPEG est utilisé comme algorithme de compression. Il existe un ensemble de P_i nœuds capables de chevaucher deux images (voir la section 2.3.2). Après avoir fait quelques calculs combinatoires exhaustifs, les auteurs proposent un schéma simple dans lequel, pour chaque groupe de trois nœuds caméra (de façon séquentielle C_{i-1}, C_i et C_{i+1}), et après avoir fait quelques compilations des informations (la distance entre les nœuds P_i et le puits et certaines propriétés des images compressées), le nœud central peut choisir le meilleur système de transmission. Une étude complémentaire de ce protocole est en discussion (Chow *et al.*, 2006). Les différents nœuds peuvent effectuer la compression des données avec différents niveaux de qualité. Ce niveau de qualité est en rapport avec le nombre de niveaux de quantification appliqués dans JPEG.

De plus amples détails sur cette idée dans un scénario plus particulier sont donnés dans (Chow *et al.*, 2007). Les auteurs considèrent un scénario avec un puits mobile. Ce puits mobile envoie une requête par innondation et de cette manière, les nœuds peuvent connaître leur distance au puits. En outre, la connaissance de la position est assumée dans chaque nœud (par exemple, par localization GPS). Considérons un cas avec deux nœuds caméra C_i et C_j. S'il y a un nœud k voisin des nœuds C_i et C_j, les nœuds compressent les images capturées et les envoient à k. Pour sélectionner un nœud k, C_i choisit ses voisins parmi ceux qui ont une distance inférieure d'un saut, puis, il peut déterminer si k et C_j sont voisins par le calcul de leur distance (en sachant que C_i connaît les coordonées et la portée de transmission de k et C_j). Un critère supplémentaire pourrait être l'énergie disponible au niveau du nœud. Le nœud k décompresse les deux images et effectue le chevauchement, puis, il envoie l'image résultant au puits. S'il n'y a pas de nœud k avec les caractéristiques décrites ci-dessus, C_i envoie la région chevauchée à C_j et la région non-chevauché au puits à travers un nœud k_i (plus près du puits). Le node C_j fait la « couture » des images capturées et les envoie par C_i et envoie les résultats au puits.

(Savidge *et al.*, 2005; Savidge *et al.*, 2006) proposent un protocole de routage pour les réseaux de capteurs sans fil multi-hop de vision qui considère un positionnement géographique des nœuds. Dans le scénario envisagé dans ce travail, deux types de données peuvent être générées par les capteurs d'image : mesures simples périodiques, à faible bande passante et les données d'images événementielles, à haut débit. Ce régime hétérogène suggère des priorités différentes pour chaque type de données. Pour faire face à cette exigence supplémentaire, les auteurs ajoutent la nécessité de stratégies de contrôle d'accès au medium (MAC) avec support de qualité de service (QoS), permettant l'attribution de priorités aux différents paquets de données, un mécanisme de double file d'attente (un pour chaque type de données), et des capacités d'adopter le prochain nœud intermédiaire dans le plus bref délai prévu. Maintenant, pour les aspects de routage, chaque noeud voisin i a une fonction de coût $c(i)$ associé, ainsi, un nœud peut déterminer entre un ensemble de candidats celui d'entre eux qui sera le prochain relai sur le chemin vers le puits. Trois paramètres sont définis : le coût de la position ($c_p(i)$), le coût des files ($c_q(i)$) et le coût de l'énergie restante ($c_e(i)$). La fonction de coût est définie comme suit :

$$c(i) = c_p(i) + \alpha.c_q(i) + \beta.c_e(i) \tag{2.2}$$

où *alpha* et *beta* sont des variables configurables qui déterminent l'impact de chaque paramètre dans la fonction de coût. Le nœud courant choisit, par conséquent, le nœud avec le coût minimum de $C(i)$.

Comme les nœuds doivent avoir une connaissance de ses voisins, chaque nœud doit diffuser son état actuel à des intervalles réguliers de temps. Le coût de la position peut être calculé avec des stratégies fondées sur la distance et/ou l'angle. Dans la stratégie basée sur la distance, la distance linéaire traditionnelle entre la source et le puits est considérée. Dans la stratégie basée sur l'angle, l'angle formé entre le noeud candidat, le nœud courant et le puits est considéré. En termes généraux, pour un mode périodique de fonctionnement, le coût de la file d'attente peut être défini comme $c_q = Q_{periodic} + 1$, où $Q_{periodic}$ est la longueur de la file d'attente du nœud. Pour une application hybride, où de paquets du type périodique et du type événementiel sont utilisés, le coût de la file d'attente est défini comme suit : $c_q = \frac{Q_{periodic}+1}{1-p}$ pour les paquets périodiques et comme $c_q = \frac{Q_{event}+1}{p}$ pour les paquets événementiels, où Q_{event} est la longueur de la file d'attente périodique du nœud et p est la probabilité d'envoyer un paquet événementiel. Afin de permettre la découverte de la topologie du réseau, dans (Savidge *et al.*, 2005) une méthode basée sur des observations entre les nœuds est proposée.

Les cas du routage multi-chemins pour la transmission de flux vidéo dans les réseaux de capteurs sans fil est examiné dans (Maimour, 2007). Les auteurs proposent SLIM (par *Simple Lifetime-based Multipath*), une protocole de routage multi-chemin conçu pour les couches de transport vidéo sur les réseaux de capteurs de ressources limitées. Avec SLIM, chaque nœud a une table de routage qui contient quatre champs : l'identifiant du chemin (`pathId`), l'identificateur du prochain nœud sur la route vers le puits (`nextNode`), la durée de vie du chemin (`path_lifetime`), et un drapeau qui indique si le chemin est en service (`InUse`). Par l'inondation d'une requête depuis le puits, quelques chemins peuvent être construits. L'algorithme fonctionne comme suit : Tout d'abord, le puits diffuse un paquet avec son identifiant et d'une durée de vie infinie. Au niveau d'un nœud intermédiaire, quand un paquet de requête est reçu du puits, le champ `pathId` sera l'identifiant du nœud courant, c'est-à-dire le nombre de chemins est limitée par le nombre de nœuds voisins au puits. Si la requête provient de tout autre nœud intermédiaire, et si la valeur de `pathId` n'existe pas dans la table du nœud, il insère un nouvel enregistrement dans sa table de routage, en ajoutant le chemin reçu et l'identifiant du nœud comme `pathId` et `nextNode`, respectivement. Le champs `path_lifetime` est calculé comme le minimum entre la durée de vie signalée par le nœud prédécesseur et le temps de vie restant du nœud courant. Enfin, lorsque le nœud source reçoit une demande, il ajoute une nouvelle entrée si le paquet de requête annonce un nouveau `pathId`, c'est-à-dire le nombre de chemins est également limité par le nombre de nœuds voisins à la source. Le résultat pourrait être un système multi-chemins comme celui de la figure 2.9(b). Le protocole a été évalué pour la transmission de flux vidéo en continu, dans des différents scénarios, y compris transport par couches avec des priorités à travers des multiples chemins.

2.4.2 Transmission robuste d'images

Bien que nous pourrions appliquer des protocoles de type ARQ, pour assurer la réception des paquets envoyés par la source, plusieurs propriétés des images pourraient être exploitées pour fournir une transmission d'image résistant aux erreurs de communication. Comme la transmission d'images implique la transmission d'une grande quantité de données, la retransmission pourrait être coûteuse en termes de consommation d'énergie, et, compte tenu que le taux de pertes peut être élevé, des mécanismes de transmission robuste, capable d'assurer que les images reçues auront une qualité acceptable pour l'utilisateur, sont nécessaires.

Nous avons déjà discuté l'approche proposée dans (Wu et Chen, 2003), où des groupes de coefficients sont transmis de façon indépendante afin de réduire l'impact de la perte de données, permettant la reconstruction de certaines versions des images originales. Par ailleurs, (Wu et Abouzeid, 2006) discutent le problème de la probabilité d'erreur sur le transport d'images. Ils proposent une méthode qui intègre une système de routage multi-chemin, et un codage FEC. Dans la stratégie de routage multi-chemin, plusieurs copies du même paquet sont envoyés par des chemins différents. Une organisation en clusters est supposée, donc, une fois que le nœud source capture une image, des copies des paquets sont transmis aux différents nœuds du cluster de la source. Ces nœuds relaient les paquets à la prochaine tête de cluster vers le puits. La tête du cluster sélectionne à son tour un ensemble de nœuds voisins dans son cluster et leur envoie à chacun des copies des paquets, et ainsi de suite. En générant diverses copies du même paquet et en transmettant à travers divers nœuds, si un paquet est perdu, nous avons encore d'autres copies du même paquet à être reçu par le prochain chef du cluster. En outre, chaque nœud tête de cluster peut appliquer un codage FEC pour génerer des paquets de redondance pour avoir une protection encore plus élevé aux pertes de paquets. La probabilité de corriger un paquet de données pour une transmission d'un saut, pour une probabilité d'erreur de bits P_e, est donné par l'équation suivante :

$$P_{cor} = \sum_{i=0}^{t_c} P_s^i (1 - P_s)^{n-i} \tag{2.3}$$

où n est la taille du bloc, $t_c = \left[\frac{n-2}{2}\right]$, et $P_s = 1 - (1 - P_e)^m$. L'application de paquets de redondance. Cette stratégie permet de réduire, en effet, l'énergie et les délais nécessaires pour transmettre l'image par rapport à un schéma fiable basé sur des acquittements et retransmissions, mais l'application de la redondance de paquets et le codage FEC encodage restent coûteux en énergie.

2.5 Conclusion

Les besoins d'applications pour les réseaux de capteurs d'images deviennent de plus en plus nombreux. La demande aujourd'hui concerte notamment les applications environnementales ou le pilotage de robots ou drones. Avec la contrainte de la limitation des ressources des nœuds, des mécanismes de traitement et de transmission d'images efficaces en énergie restent à développer. Dans ce chapitre, nous avons classé les travaux trouvés dans la littérature, qui traitent essentiellement de transmission et de traitement des images sur les réseaux de capteurs sans fil.

Dans le domaine du traitement d'images, on remarque des efforts de plusieurs auteurs pour adapter les algorithmes de compression d'images de l'informatique traditionnelle, comme JPEG ou JPEG2000, aux contraintes particulières des réseaux de capteurs sans fil. Cependant, même si les algorithmes classiques présentent des remarquables performances en termes de ratio débit/distorsion, des expériences sur de vrais capteurs démontrent que ces algorithmes sont beaucoup trop coûteux en énergie, plus coûteux que la transmission d'une image sans compression. Plusieurs adaptations ont été proposées pour rendre ces algorithmes moins gourmands en énergie. Les résultats sont discutables et il y a plusieurs facteurs qui doivent encore être traités, comme la tolérance aux pertes de paquets, par exemple. Les efforts sur la compression distribuée semblent encore peu efficaces. Ils n'apportent aucune économie d'énergie au niveau de la source, et la complexité des méthodes d'échange de données entre les nœuds ne sont pas toujours considerés. Plus de recherche est donc nécessaire dans ce domaine.

De même, la problématique de la transmission d'images sur les réseaux de capteurs a été peu étudiée. Cependant, on trouve des travaux très intéressants en particulier sur le routage des paquets. Ce sont des algorithmes similaires à ceux qu'on trouve dans la bibliographie de réseaux de capteurs en général, mais avec des adaptations qui profitent des caracteristiques de l'image pour améliorer la qualité des images du côté du récepteur ou améliorer la distribution de la charge du réseau. Des travaux récents incorporent maintenant des adaptations des standards de couche 2, comme la norme 802.15.4.

L'histoire de la transmission d'images sur réseaux de capteurs est encore jeune, mais elle commence à attirer la communité cientifique. On peut voir une différence claire entre l'état de l'art disponible lorsque cette thèse a commencé et l'état de l'art disponible aujourd'hui. Néanmoins, le domaine est loin d'être assez developpé. Plus de recherche est nécessaire, surtout il manque de travaux qui considèrent des facteurs du monde réel (comme la perte de paquets, les problèmes liés aux communications, etc.) et de preuves de faisabilité dans des plateformes réelles.

Deuxième partie

Contributions : Vers la transmission efficace d'images sur des réseaux de capteurs sans fil

Chapitre 3

Transmission d'images par un protocole semi-fiable

La consommation d'énergie est un problème fondamental dans les réseaux de capteurs sans fil lorsqu'on prend pour hypothèse que les nœuds du réseau sont dotés d'une batterie dont le renouvellement est impossible. Dans le cas des capteurs d'image, le problème de l'énergie est aggravé par la dimension de l'information mesurée puisqu'il faut de quelques milliers à quelques millions de bits pour représenter une image. Cela est d'un tout autre ordre de grandeur que des mesures scalaires classiques comme la température. Par conséquent, la transmission d'une image va s'effectuer sur (beaucoup) plus qu'un paquet. Tous ces paquets que la source devra envoyer vont lui coûter de l'énergie. Ils vont aussi coûter de l'énergie à tous les nœuds traversés sur le chemin de la source au puits (sachant que les nœuds de transit ne sont pas nécessairement les mêmes d'un paquet à l'autre, cela dépendra bien sûr de la stratégie de routage adoptée). Pour faire des économies d'énergie, il y a deux types de solutions : agir sur l'image à la source ou agir sur le protocole de communication.

Les actions à mener à la source vont viser à réduire le volume des données de l'image, et donc le nombre de paquets à envoyer sur le réseau. C'est d'abord, et tout simplement, prendre des images de petite taille. C'est aussi privilégier les images monochromes. C'est, enfin, effectuer une compression de l'image. Il est attendu de ces actions qu'en ayant moins de paquets à transmettre, le nœud source comme les nœuds de transit feront des économies d'énergie. C'est mécanique. Si le format de l'image adopté dépendra très fortement de l'application, le choix d'un algorithme de compression se fera plutôt sur la base d'un critère de performance (coût d'énergie pour la source, rapport énergie-distorsion, etc..) indépendant de l'application. Nous avons soulevé, à travers l'état de l'art présenté au chapitre 2, que les algorithmes de compression classiques ne sont pas éligibles dans le contexte des réseaux de capteurs dotés de capacités de calcul, de mémoire et d'énergie limitées. En l'absence de techniques de compression dont l'efficacité a été prouvée, les actions pouvant être menées sur l'image à la source pour économiser l'énergie sont assez réduites (nous traiterons de la compression d'image au chapitre 5).

Dans ce chapitre, nous nous intéressons aux actions qui peuvent être menées sur le protocole de communication pour diminuer l'énergie consommée pour transmettre une image d'une source jusqu'au puits. Plusieurs types d'actions sont envisageables, en agissant soit sur le système d'accès au canal de transmission, soit sur le système de routage, soit sur le système de transport de bout en bout. Au niveau

du système d'accès au canal de transmission, beaucoup de travaux sont disponibles dans la littérature. Sans entrer dans les détails, disons simplement que des économies d'énergie peuvent être obtenues en adaptant la puissance du transcepteur radio (Cardei *et al.*, 2008), en limitant son temps d'activité (cela est réalisé en alternant des phases d'activité et des phases de sommeil, comme le fait le protocole S-MAC et ses variantes (Ye *et al.*, 2002)), ou encore en adoptant un protocole sans collision (Rajendran *et al.*, 2003; Stathopoulos *et al.*, 2004). Au niveau du système de routage, la sélection du chemin sur un critère tenant compte du coût d'énergie et du taux d'erreurs résiduel peut amener des économies d'énergie (Chang et Tassiulas, 2000; Shah et Rabaey, 2002; Akkaya et Younis, 2003). Un routage multi-chemin est aussi une option intéressante pour répartir spatialement la dépense d'énergie sur un plus grand nombre de nœuds (Maimour, 2007; Kim *et al.*, 2008). Au niveau du système de transport de bout en bout, dont la fonction a priori est d'assurer une livraison fiable des paquets, des économies d'énergie sont possibles en relâchant la contrainte de fiabilité. Cela veut dire accepter, dans une certaine limite, des pertes de paquets. On parle alors de protocole de communication semi-fiable, la différence avec un protocole non fiable est qu'ici, toutes les pertes de paquets ne sont pas admissibles. Les applications de transmission d'images peuvent s'appuyer sur un protocole de communication semi-fiable puisque les images naturelles ont, par nature, une certaine tolérance aux erreurs. La contrainte de ces applications est exprimée moins en termes de fiabilité qu'en termes de qualité de l'image finale. En d'autres mots, peu importe que des paquets soient perdus dans le réseau pourvu que la qualité de l'image finale n'en souffre pas (trop). La spécification d'un protocole semi-fiable peut être traitée, soit de manière statistique, soit de manière déterministe. Un protocole de type statistique va considérer que tous les paquets ont la même importance, et il va garantir que le nombre de paquets livrés sera au moins égal à un ratio donné (par exemple, 95% minimum des paquets doivent être livrés), quels que soient les paquets effectivement livrés. Un protocole de type déterministe assure un contrôle des pertes de paquets de manière fine, l'importance des paquets étant ici appréciée individuellement. Cela sous-entend que les paquets sont classifiés sur plusieurs niveaux de priorité. Un exemple de ce principe est donné par (Holl *et al.*, 2005) pour les réseaux de bout en bout. Ici, les hautes fréquences dans le domaine des ondelettes ne sont pas retransmises en cas de pertes dans le réseau car elles contribuent moins à l'amélioration de la qualité que les moyennes et basses fréquences.

Dans ce chapitre, nous proposons deux protocoles semi-fiables pour la transmission d'image fixes, sans contrainte de temps. Ils sont dictés par la contrainte de consommation d'énergie, et adaptés aux réseaux de capteurs sans fil. Le premier fonctionne en boucle ouverte, l'autre en boucle fermée. Dans les deux cas, les économies d'énergie sont obtenues en préparant à la source des paquets de différentes priorités, grâce à une transformée en ondelettes de l'image, puis en conditionnant l'acheminement des paquets, saut par saut, à leur priorité et à l'état de charge des batteries. Ces protocoles font un compromis entre la qualité des images obtenues et la quantité d'énergie dissipée pour les transmettre de bout en bout.

Le chapitre est structuré en 4 parties. La première partie présente les principes génériques du protocole de communication semi-fiable proposé. La deuxième partie détaille le fonctionnement du protocole en boucle ouverte. Un modèle analytique de la consommation d'énergie est développé dans la troisième partie pour quantifier le coût de transmission d'une image pour le protocole en boucle ouverte. Les résultats de l'évaluation de performance montrent une réduction importante de l'énergie consommée en moyenne avec ce protocole, en comparaison avec un protocole fiable classique. L'espérance du gain d'énergie avoisine par exemple 70% lorsqu'on considère un chemin de bout en bout long de 30 sauts. La cinquième partie propose des variantes du protocole, en particulier une version auto-adaptative à la longueur du chemin

et une version fonctionnant en boucle fermée.

3.1 Principes techniques

Le protocole de communication semi-fiable que nous proposons s'inscrit dans la classe des protocoles déterministes (Lecuire et Lepage, 1999). Le contrôle de la fiabilité est effectuée paquet par paquet. Les économies d'énergie sont obtenues en préparant à la source des paquets de différentes priorités, grâce à une transformée en ondelettes de l'image, puis en conditionnant l'acheminement des paquets, saut par saut, à leur priorité et à l'état de charge des batteries. Cela est schématisé sur la figure 3.1.

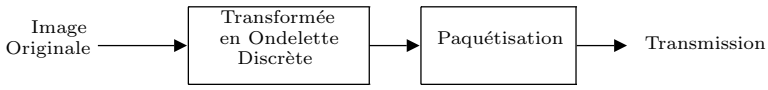


FIG. 3.1: Schéma de transmission semi-fiable.

3.1.1 Transformée en ondelettes d'une image

La transformée en ondelettes discrète (TO) (Mallat, 1999) 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. La partie résultant du filtrage passe-bas représente une approximation du signal d'origine à la nouvelle résolution (ou échelle), celle résultant du filtrage passe-haut représentant les détails perdus entre les deux résolutions.

Puisqu'une image est typiquement un signal en deux dimensions, une TO dyadique est réalisée, en appliquant d'abord les filtres L et H sur les échantillons ligne par ligne, puis en réappliquant les mêmes filtres sur les échantillons résultants, mais colonne par colonne cette fois-ci. Au final, l'image est divisée en 4 parties, les sous-images LL, LH, HL et HH comme présenté sur la figure 3.2(a). La sous-image LL fournit une version à l'échelle $\frac{1}{2}$ de l'image d'origine, LH, HL et HH représentant les détails perdus respectivement dans les directions horizontale, verticale et diagonale. La TO peut être réitérée sur LL pour obtenir plusieurs niveaux de résolution. La figure 3.2(b) présente une image décomposée en trois niveaux de résolution.

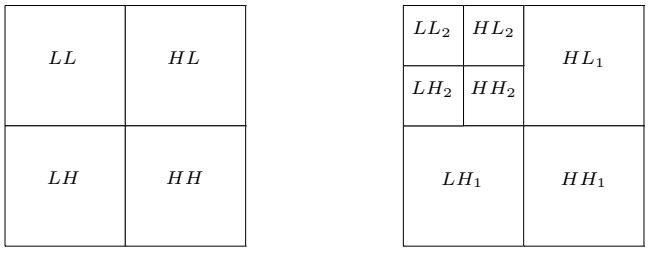


FIG. 3.2: La TO dyadique appliquée une fois (a) ou deux (b).

La transformation de l'image en une représentation multirésolution fournit à la source les moyens de préparer des paquets de données de différentes priorités. En fait, l'image sera divisée en p résolutions si la

TO est appliquée $(p-1)$ fois, ce qui donne au moins p niveaux de priorité (plus si on fait une distinction entre les sous-images ou/et entre les plans de couleur, voire entre les énergies des coefficients). La plus petite résolution (la résolution 0), qui est représentée par la sous-image LL_p, est la plus importante. Cette sous-image doit être transmise jusqu'au puits de manière fiable pour que l'utilisateur puisse reconstruire une version de l'image d'origine avec un niveau de qualité minimum garanti. Quelques informations supplémentaires doivent obligatoirement être connues du décodeur, il s'agit des informations relatives au format de l'image proprement dit (taille de l'image, nombre de plans, nombre de bits par pixels, nombre de niveaux de résolution, etc. ...). Elles sont regroupées dans ce qu'on apelle communément les informations d'entête de l'image. Si la source est programmée pour transmettre des images toujours au même format, le récepteur connaîtra implicitement ce format et il n'aura pas de problèmes pour décoder l'image. Sinon, l'entête de l'image devra aussi être transmis de manière fiable jusqu'au puits. Les données de la résolution 0 de l'image, et si besoin celles de l'entête de l'image, seront donc placées dans des paquets ayant la priorité la plus élevée (la priorité 0). Les données associées aux autres niveaux de résolution ont une importance qui décroit de la résolution 1 à $(p-1)$. Elles seront donc rangées dans des paquets de priorité décroissante. Plusieurs stratégies de priorité sont possibles. Pour une stratégie par résolutions par exemple, la priorité i étant associée à la $i^{ème}$ résolution (représentée par les sous-images HL_{p-i}, LH_{p-i}, et HH_{p-i}). Nous discuterons de ces stratégies plus loin, dans la section 3.1.2.

Pour appliquer la TO, nous avons adopté les filtres 5-3 de *Le Gall*, qui ont des coefficients rationnels. Le filtre passe-bas est donné par $f_L(z) = -\frac{1}{8}.\left(z^2+z^{-2}\right)+\frac{1}{4}.\left(z+z^{-1}\right)+\frac{3}{4}$ et le filtre passe-haut par $f_H(z) = -\frac{1}{2}.\left(z+z^{-1}\right)+1$. Cette ondelette a été adaptée dans (Calderbank *et al.*, 1998) pour opérer spécialement dans l'espace des valeurs entières. Les valeurs obtenues en sortie des filtres sont des arrondis à l'entier le plus proche. La quantité de données de l'image reste donc la même après transformation. Les filtres 5/3 sont particulièrement appréciables dans les applications contraintes par l'énergie car leur implantation fait intervenir des instructions simples, des additions et des décalages de valeurs entières, et non des multiplications et des divisions. Ils sont donc moins gourmand en énergie que des filtres à coefficients non rationnels, c'est la raison pour laquelle ils ont été choisis.

3.1.2 Prioritisation et paquétisation des données

Plusieurs stratégies sont envisagables pour fixer la priorité à attribuer aux paquets en fonction des données qu'ils contiendront. Dans cette section, nous présentons deux stratégies simples : stratégie bassée sur les résolutions de l'image et stratégie bassée la magnitude des coefficients d'ondelette.

Priorités basées sur les résolutions de l'image

Cette stratégie de priorités est la plus banale. Elle associe un niveau de priorité par résolution. Les données d'une image divisée en p niveaux de résolution seront donc différenciées sur p niveaux de priorité, de 0 (la priorité la plus élevée, à $p-1$). La priorité 0 sera aussi assignée aux données de la sous-image LL_p. Les données relatives au $i^{\text{ème}}$ niveau de résolution de l'image, avec $0 < i < p$, et qui sont représentées par les sous-images HL_{p-i}, LH_{p-i} et HH_{p-i}, se verront assigner en priorité i. Cette stratégie est illustrée figure 3.3(a).

Le processus de paquétisation des données va opérer de la manière suivante :

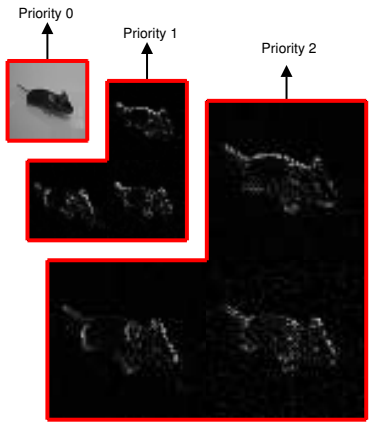
– Lecture des données de la sous-image LL_p et rangement dans les paquets de priorité 0 en s'attachant, bien sûr, à les remplir au maximum.
– Lecture des données des sous-images HL_{p-1}, LH_{p-1} et HH_{p-1} en rangeant dans les paquets de priorité 1, et ainsi de suite pour chaque niveau de résolution.

Priorités basées sur la magnitude des coefficients d'ondelette

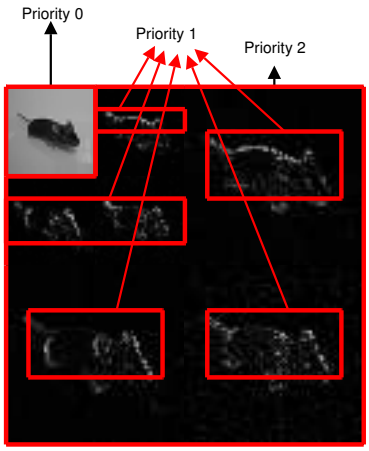
Dans le niveau de résolution le plus petit, tous les coefficients ont la même importance pour la reconstruction de l'image. Ils seront tous assignés à la priorité 0. Mais dans les autres niveaux de résolution, tous les coefficients ne se valent pas. Ceux dont les valeurs sont proches de zéro ayant un plus faible impact pour la qualité des images reconstituées. Cette stratégie de priorités considère l'importance des données de l'image avec un niveau de granularité plus fin, coefficient par coefficient et non plus résolution par résolution. Une stratégie sur p niveaux de priorité, priorité 0 incluse, sera concrétisée en définissant un ensemble de $(p-2)$ seuils, $\{\tau_1, \tau_2, \ldots, \tau_{(p-2)})\}$, avec $\tau_i > \tau_{i+1}$. Un coefficient c_x quelconque se verra assigner le niveau de priorité i selon la règle suivante :

$$\text{Priorité du coefficient } c_x = \begin{cases} 1, & \text{si } |c_x| \geq \tau_1 \\ i, & \text{si } \tau_i \leq |c_x| < \tau_{(i-1)} \\ (p-1), & \text{si } |c_x| < \tau(p-2) \end{cases}$$

En pratique, il faut juste attribuer une priorité à chaque paquet de données en fonction des coefficients qu'il contient. La priorité qui lui sera assignée est égale à celle des coefficients qui a la plus grande valeur absolue dans ce paquet. Cette politique est illustré dans la figure 3.3(b).



(a) Stratégie basée sur les résolutions (exemple pour une image divisée en 3 niveaux de résolution).



(b) Stratégie basée sur la magnitude des coefficients d'ondelette.

FIG. 3.3: Affectation de priorités sur les données d'uné image ayant une représentation multi-résolution bassée sur la transformée en ondelettes.

Le processus de paquétisation des données va opérer de la manière suivante :
– Lecture des données de la sous-image LL_p et rangement dans les paquets de priorité 0 en s'attachant, bien sûr, à les remplir au maximum.

– Lecture des données de toutes les sous-images HL, LH et HH et rangement dans des paquets dont la priorité dépend du coefficient les plus grand stocké dans le paquet. Si sa valeur absolue est supérieure à τ_1, la priorité du paquet sera 1, i si sa valeur absolue est inférieure à $\tau_{(i-1)}$ et supérieure à τ_i, et $(p-1)$ si sa valeur absolue est inférieure à $\tau_{(p-2)}$.

3.1.3 Transmission semi-fiable

Dès qu'un paquet de données a été formé et sa priorité affectée, il est prêt à être envoyé sur le réseau. La source va transmettre les paquets dans l'ordre dans lequel ils ont été préparés. Avec une stratégie de priorité basée sur les résolutions de l'image, la transmission commence pour celles avec la plus grande priorité, puis se poursuit avec ceux de la priorité immédiatement inférieure (1), et ainsi de suite jusqu'à ceux de priorité $(p-1)$. Avec une stratégie basée sur la magnitude des coefficients d'ondelettes, les paquets de priorité 0 sont transmis en premier, puis le reste est fonction de l'image. Cela n'a pas d'importance finalement. Pour relacher la contrainte de fiabilité posée sur le système de communication, nous proposons que l'acheminement des paquets jusqu'au puits soit conditionné, saut par saut, à leur priorité et à l'état de charge des batteries des nœuds traversés. Ce système de transmission d'image est semi-fiable au sens où il n'est pas obligatoire de transmettre tous les paquets jusqu'au bout, excepté bien sûr ceux de la priorité 0. Ce principe est motivé principalement par le souci d'économiser l'énergie des nœuds de transit. On ne vise pas, ici, à minimiser la consommation d'énergie (pour cela, il faudrait transmettre le moins de données possible, c'est-à-dire considérer une image de la plus petite qualité acceptable pour l'utilisateur), il s'agit de faire un compromis entre la qualité des images reçues et la quantité d'énergie dissipée pour les transmettre de bout en bout.

Le protocole de transmission d'image que nous proposons s'appuie sur les principes suivants : dans l'horizon de la communication de proche en proche, les échanges sont traités de manière fiable, c'est-à-dire qu'un paquet envoyé à un nœud voisin doit être acquitté immédiatement, sinon il y a retransmission. Mais dans l'horizon de la communication de bout en bout, un nœud qui reçoit un paquet peut décider de l'écarter s'il estime que la préservation de l'état de sa propre batterie a plus d'importance que la dégradation de la qualité d'image que la perte du paquet entraînera. Lorsque le nœud décide d'écarter un paquet, il économise de fait une quantité d'énergie égale à la transmission de ce paquet (y compris l'attente et la réception d'un acquittement, ainsi que d'éventuelles retransmissions). Il est donc doté d'une sorte d'instinct de survie.

Pour que les nœuds puissent prendre objectivement leurs décisions (relayer ou écarter tels ou tels paquets), nous avons défini un système de décision simple, associant un seuil d'énergie, $\alpha_0, \alpha_1, \ldots \alpha_\ell, \ldots \alpha_{p-1}$, à chaque niveau de priorité des paquets. Formellement, nous avons : $\forall \ell \in \mathbb{N},\ \alpha_\ell \in [0,1[$ and $\alpha_\ell < \alpha_{\ell+1}$. Cela est illustré est présenté sur la figure 3.4.

Il reste encore une question : quelles valeurs utiliser pour ces seuils d'énergie ? Dans la pratique, cela dépendra des exigences de l'application, et ces exigences doivent être identifiées avant la mise en œuvre du protocole.

La loi de distribution des seuils d'énergie peut être quelconque dans le cas général, il suffit que les valeurs des seuils soient programmées dans un nœud pour que celui-ci opère de manière autonome. Il n'y a pas non plus d'obligation pour que tous les noeuds adoptent les mêmes valeurs. Nous n'investiguerons pas plus loin ce point dans le cadre de cette thèse, disons simplement que le choix des seuils aura une

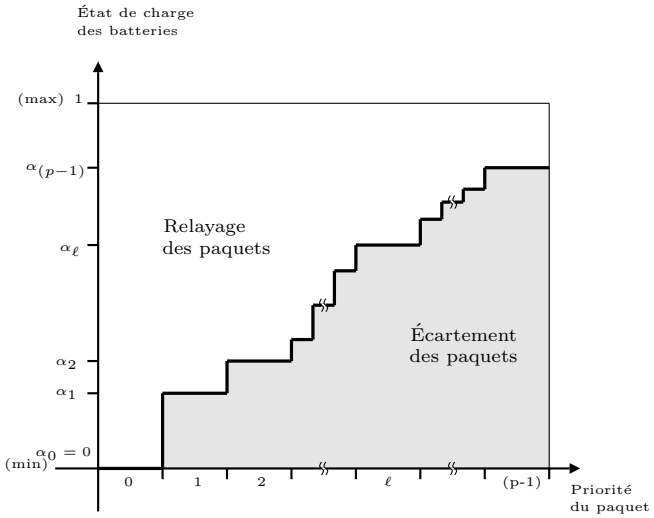


FIG. 3.4: Relayage des paquets en fonction de leur priorité et de l'état de charge des batteries.

influence significative sur les résultats attendus. Par exemple, une distribution logarithmique des seuils α_ℓ favorisera la préservation de l'énergie au détriment de la qualité d'image (elle augmente la probabilité d'écartement des paquets), alors qu'une distribution exponentielle aura tendance à privilégier la qualité d'image. Une distribution uniforme considère à part égal ces deux critères antagonistes.

Sur la base des principes techniques que nous venons d'exposer, plusieurs protocoles de communication semi-fiable peuvent être dérivés. La section suivante décrit un protocole de communication de type boucle ouverte.

3.1.4 Protocole semi-fiable en boucle ouverte

En première proposition, nous considérons qu'un nœud de transit recevant un paquet va décider de relayer ce paquet ou l'écarter en fonction de l'état de sa batterie exclusivement, indépendamment de l'énergie disponible dans les autres nœuds de transit. Comme aucune boucle de retour est utilisée dans ce cas, ce protocole définit un système de transmission en boucle ouverte. C'est le plus simple à mettre en oeuvre. En effet, supposons qu'une loi de distribution des seuils d'énergie α_ℓ a été pré-programmée dans les noeuds du réseau. Quand un paquet arrive à un noeud, celui-ci a besoin de deux informations pour opérer correctement : la priorité assignée au paquet et le nombre de priorités (résolutions) qu'il y a au total. Ces informations peuvent être fournies par le noeud source (c'est lui qui décide du nombre de priorités) en les embarquant dans des champs de l'entête du paquet. En plus de ces informations, l'entête du paquet doit contenir l'identifiant de l'image et l'offset des données. Celles-ci ne servent pas aux nœuds de transit mais sont nécessaires au destinataire final pour qu'il range les données en bon ordre. Si des paquets sont manquants au destinataire, il peut substituer les données manquantes par des zéros si l'image n'a pas été compressée à la source. Autrement, il lui suffit de décoder seulement les données dans les résolutions qui ont été reçues entièrement. Comme indiqué précédemment, la fiabilité des paquets est assurée de proche en proche par un système d'acquittement et retransmission. L'acquittement embarque l'identifiant de l'image et l'offset des données qui étaient contenus dans l'entête du paquet de données.

L'intérêt majeur d'un tel protocole en boucle ouverte, outre sa simplicité, est qu'il est déployable sur

un réseau de capteurs quelque soit le modèle de routage qui est en vigueur. En particulier, les paquets n'ont pas besoin de prendre tous le même chemin. Des protocoles de routage basés sur la diffusion comme Gossiping (Hedetniemi et Liestman, 1988) peuvent donc être utilisés. D'une manière générale, le protocole en boucle ouverte est approprié pour les applications où la transmission des images se fait de manière périodique ou événementielle.

Un analyse de performances de ce protocole de communication est réalisé dans la section suivante.

3.2 Analyse de performances du protocole en boucle ouvert

L'évaluation des performances qui peuvent être obtenues en transmettant les images par un protocole semi-fiable a été réalisée par analyse de la consommation d'énergie. A cette fin, nous avons développé le modèle de consommation d'énergie de la transmission d'une image de bout en bout. Ce modèle est basé sur trois composants élémentaires : un modèle de protocole de communication, un modèle du transcepteur radio, et un modèle du processus de transformée en ondelettes dyadique. Pour faciliter l'analyse et pour simplifier les formules, nous considérons, sans perte de généralité, les hypothèses suivantes :

- Tous les noeuds du réseau ont les mêmes caractéristiques (mêmes composants matériels, même configuration des transcepteurs radio, etc...).
- Le niveau de charge de la batterie dans un nœud ne varie pas de manière significative sur la durée de transmission d'une image.
- La route suivie par les paquets pour aller du noeud source jusqu'au puits est constituée de n nœuds de transit, numérotés de 1 à n dans l'ordre de passage, comme schématisé sur la figure 3.5. Cette route est supposée unique et stable sur la durée de transmission d'une image. De même, les transmissions par liaison radio sont considérées sans erreurs (il est évident que la prise en compte des erreurs de transmission serait à l'avantage des protocoles semi-fiables en comparaison d'un protocole fiable).
- L'image est décomposée en p niveaux de résolutions.
- Les paquets sont différenciés en p niveaux de priorité.

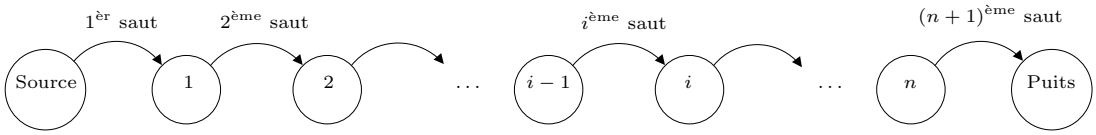


FIG. 3.5: Représentation du chemin entre la source et le puits.

Afin de calculer la consommation d'énergie cumulée de bout en bout pour transmettre une image de la source jusqu'au puits, nous devons déterminer le nombre de sauts qui sera exécuté pour chaque paquet de données. Il dépend du niveau de priorité ℓ du paquet et du niveau de charge des batteries dans les nœuds de transit.

Calculons la probabilité $R(\ell, n)$ que les paquets de priorité ℓ soient transmis jusqu'au puits, c'est-à-dire effectuent $(n+1)$ sauts. Cela correspond à la probabilité que les noeuds de 1 à n ont tous un niveau d'énergie suffisant pour relayer les paquets de cette priorité. L'équation 3.1 fournit le calcul de cette probabilité :

$$R(\ell, n) = (1 - \alpha_\ell)^n \tag{3.1}$$

avec $0 \leq \ell \leq p-1$. Calculons maintenant la probabilité $B(\ell, i)$ que les paquets de priorité ℓ soient écartés en chemin à par le nœud i. Cela correspond à la probabilité que le noeud i soit le premier noeud sur le chemin à avoir un niveau d'énergie insuffisant pour cette priorité. L'équation 3.2 donne cette probabilité :

$$B(\ell, i) = \alpha_\ell . (1 - \alpha_\ell)^{i-1} \tag{3.2}$$

avec $1 \leq i \leq n$ et $1 \leq \ell \leq p-1$.

Les probabilités $R(\ell, n)$ et $B(\ell, i)$ vont servir à évaluer la consommation d'énergie moyenne (c'est l'espérance mathématique) pour transmettre une image, par un protocole semi-fiable. Comme une image constitue *a priori* un gros volume de données, c'est-à-dire plus grand que l'unité maximale de transmission, elle est transmise en plusieurs paquets dans le cas général. Représentons par m_ℓ et t_ℓ respectivement le nombre et la taille moyenne des paquets nécessaires pour transmettre entièrement les données de priorité ℓ, y compris les informations ajoutées dans les en-têtes de protocole. Posons aussi $E(k)$ l'énergie consommée pour transmettre et acquitter un paquet de k octets entre deux noeuds voisins (coût d'énergie pour un saut). Dans ce qui suit, nous allons développer les modèles de consommation d'énergie du protocole de communication, du transcepteur radio et de la transformée en ondelettes dyadique. Nous avons aussi développé un modèle pour estimer la qualité minimale des images reçues en fonction des paquets correctement délivrés au puits.

3.2.1 Modélisation du protocole en boucle ouverte

Rappelons qu'avec le protocole en boucle ouverte, la décision d'un noeud de relayer ou de bloquer les paquets d'un niveau de priorité donné dépendra seulement du niveau d'énergie de sa propre batterie, indépendamment de l'état des batteries des autres noeuds. Considérons d'abord les données de priorité 0 de l'image puisqu'elles sont transmises en premier. Elles sont divisées en m_0 paquets de taille moyenne t_0, qui sont nécessairement transmis jusqu'au puits. Chacun des paquets effectue $(n+1)$ sauts et la consommation d'énergie pour les transmettre de bout en bout équivaut donc à :

$$E_0(m_0, t_0) = (n+1) . m_0 . E(t_0) \tag{3.3}$$

Pour les autres niveaux de priorité par contre, les paquets sont susceptibles d'être écartés en chemin. Considérons les données de priorité ℓ, qui sont divisées en m_ℓ paquets de taille moyenne t_ℓ. Ces paquets effectuent nécessairement le saut du noeud source au noeud 1, comme schématisé sur la figure 3.5, mais les autres sauts sont conditionnés à l'état des nœuds qui forment le chemin de bout en bout. En fait, le nombre de sauts exécutés par les paquets de priorité ℓ sera égal à 1 si cette priorité est bloquée par le noeud 1, égal à 2 si elle est bloquée par le noeud 2, ... égal à i si elle est bloquée par le noeud i, ... ou bien égal à $(n+1)$ si aucun nœud de transit est bloquant pour cette priorité. A partir des équations 3.1 et 3.2, l'énergie consommée de bout en bout pour transmettre les paquets de priorité ℓ est donnée par :

$$E_\ell\left(m_\ell, t_\ell\right) = \underbrace{\sum_{i=1}^{n} B\left(\ell, i\right).i.m_\ell.E\left(t_\ell\right)}_{\text{cas où le noeud } i \text{ est bloquant}} + \underbrace{R\left(\ell, n\right).\left(n+1\right).m_\ell.E\left(t_\ell\right)}_{\text{cas où tous les sauts sont accomplis}} \tag{3.4}$$

Finalement, la consommation d'énergie cumulée de bout en bout pour transmettre l'image dans sa totalité équivaut à :

$$E_T = \left(n+1\right).m_0.E\left(t_0\right) + \sum_{\ell=1}^{p-1}\left[m_\ell.E\left(t_\ell\right).\left(R\left(\ell, n\right).\left(n+1\right) + \sum_{i=1}^{n} B\left(\ell, i\right).i\right)\right] \tag{3.5}$$

3.2.2 Modélisation d'un *transcepteur* radio

La communication d'un message entre deux noeuds voisins par une liaison radio engage un ensemble de procédures chez l'émetteur comme chez le récepteur, qui consomment, chacune, une certaine quantité d'énergie. Un modèle simple de *transcepteur* radio considère trois opérations élémentaires, comme cela est schématisé sur la figure 3.7 : la transmission d'un message lorsque le *transcepteur* est en mode émission, la réception d'un message lorsqu'il est en mode réception, et le basculement d'un mode à l'autre. Pour un noeud i donné, le coût d'énergie pour chacune de ces opérations est noté respectivement $E_{Tx,i}\left(k, P_{out}\right)$, $E_{Rx,i}\left(k\right)$ et $E_{Sw,i}$, où k représente la longueur du message, en octets, et P_{out} la puissance de transmission. Pour simplifier, nous considérerons que tous les noeuds ont exactement les mêmes caractéristiques, donc qu'ils consomment la même quantité d'énergie pour une même opération.

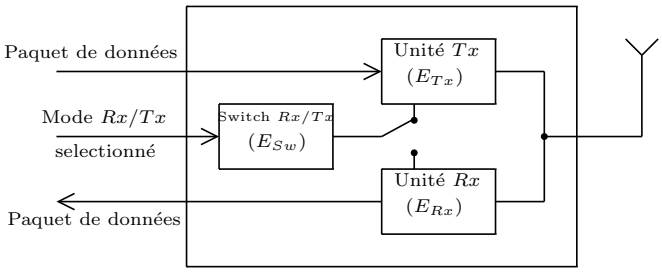


FIG. 3.6: Représentation d'un transcepteur radio.

Si l'énergie consommée est définie en millijoule (mJ), elle peut être exprimée comme le produit de la tension appliquée sur le circuit, en volt (V), de l'intensité du courant qui traverse ce circuit, en milliampère (mA), et du temps écoulé pour exécuter l'opération, en seconde (s). On peut donc écrire $E_{Sw} = C_{Sw}.V_B.T_{Sw}$, $E_{Tx}\left(k, P_{out}\right) = k.C_{Tx}\left(P_{out}\right).V_B.T_{Tx}$ et $E_{Rx}\left(k\right) = k.C_{Rx}.V_B.T_{Tx}$, où V_B représente la tension fournie par la batterie, C_{Sw}, C_{Tx} et C_{Rx} représentent l'intensité des courants pour les trois opérations, T_{Sw} le temps de commutation de mode et T_{Tx} le temps de transmission d'un octet ($T_{Tx} = T_{Rx}$ nécessairement).

Le modèle de consommation d'énergie qui a été présenté dans le paragraphe 3.2.1 exprime par la variable $E\left(k\right)$ le coût d'énergie pour transmettre et acquitter un paquet de k octets entre deux noeuds voisins (coût d'énergie pour un saut). A partir du modèle de *transcepteur* radio, et en considérant qu'un message d'acquittement a une longueur fixe de L_{Ack} octets, nous pouvons écrire :

$$E(k) = (k + L_{Ack}).C_{Tx}(P_{out}).V_B.T_{Tx} + (k + L_{Ack}).C_{Rx}.V_B.T_{Tx} + 4.C_{Sw}.V_B.T_{Sw} \tag{3.6}$$

3.2.3 Modélisation de la transformée en ondelettes dyadique

La transformation de l'image à la source en une représentation multi-résolution est une opération qui coûte de l'énergie. Il faut donc en tenir compte. Un modèle de consommation d'énergie pour la transformée en ondelettes dyadique est défini par *Lee* et *Dey* dans (Lee et Dey, 2002). Ce modèle a été établi en décomposant le processus global en instructions élémentaires, et en déterminant combien de fois ces instructions étaient exécutées lorsque le filtre 5-3 de *Le Gall* est appliqué (c'est justement celui que nous utilisons). En fait, pour chaque pixel de l'image d'origine, l'application du filtre passe-bas nécessite 8 décalages et 8 additions, alors que le filtre passe-haut exige 2 décalages et 4 additions. De plus, chaque pixel doit être lu deux fois en mémoire et écrit deux fois. En considérant que l'image d'origine a une dimension de $M \times N$ pixels, et que l'image est décomposée en p niveaux de résolution, et donc que la transformée en ondelettes dyadique est exécutée itérativement $p-1$ fois, alors le coût d'énergie est approximativement donné par :

$$E_{DWT}(M, N, p) = M.N.(10.\varepsilon_{shift} + 12.\varepsilon_{add} + 2.\varepsilon_{rmem} + 2.\varepsilon_{wmem}).\sum_{i=1}^{p-1}\frac{1}{4^{(i-1)}} \tag{3.7}$$

où ε_{shift}, ε_{add}, ε_{rmem} et ε_{wmem} représentent le coût d'énergie des quatre instructions élémentaires sur octet, respectivement le décalage, l'addition, la lecture et l'écriture.

3.2.4 Modélisation de la qualité des images reçues

Pour analyser l'impact des pertes de paquets sur la dégradation de la qualité d'image reconstituée à l'arrivée, nous utiliserons le PSNR (*Peak Signal to Noise Ratio*) comme indicateur de qualité. Le PSNR fournit une mesure objective de la distorsion entre l'image originale et l'image d'arrivée. Plus la valeur de PSNR est grande, et plus la distorsion est faible. Dans le cas où les deux images sont identiques, le PSNR a une valeur égale à l'infini. Le PSNR est defini comme :

$$PSNR = 10.log_{10}\left(\frac{MAX^2}{MSE}\right) = 20.log_{10}\left(\frac{MAX}{\sqrt{MSE}}\right) \tag{3.8}$$

où MAX est la valeur maximale d'un pixel (traditionnellement 255 dans les images en échelle de gris codées sur 8bpp) et MSE est l'erreur quadratique moyenne (*Mean Squared Error*), calculé comme :

$$MSE = \frac{1}{M.N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}||x_{i,j} - y_{i,j}||^2 \tag{3.9}$$

où $x_{i,j}$ est l'intensité des pixels de l'image originale ayant les coordonnées (i, j) et $y_{i,j}$ est l'intensité du pixel dans l'image reconstruite.

Pour adapter cette notion de qualité d'image à notre modèle probabiliste, nous définissons le PSNR moyen ($\overline{PSNR}$) comme :

$$\overline{PSNR} = R(p-1, n).PSNR(p-1) + \sum_{\ell=0}^{p-2}([R(\ell, n) - R(\ell+1, n)].PSNR(\ell)) \tag{3.10}$$

où $PSNR(\ell)$ est le PSNR calculé avec l'image reconstituée à partir des données de priorité 0 à ℓ seulement. Il s'agit donc du PSNR qui sera obtenu dans le pire des cas (c'est-à-dire quand tous les paquets de priorité supérieur à ℓ ont été perdus).

3.3 Résultats numériques

Dans cette section, nous appliquons les modèles de consommation d'énergie pour évaluer le coût d'énergie pour transmettre une image avec un protocole semi-fiable et comparer ce coût avec celui d'un protocole fiable classique. Plusieurs scénarios de transmission d'image ont été définis, en variant le nombre de résolutions de l'image et la stratégie de prioritisation. Les tests ont été effectués sur une image monochrome de 128×128 pixels, affichée sur la figure 3.7. Elle est codée à l'origine sur 8 bits par pixel. Elle a une taille de 16393 octets, y compris l'en-tête d'image de 4 octets (voir section 3.3.1). Nous donnerons d'abord les valeurs numériques que nous avons affectées aux paramètres des modèles, puis nous présenterons les résultats numériques.

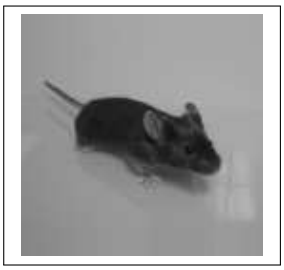


FIG. 3.7: Image originale utilisée pour les tests (128×128 pixels).

3.3.1 Paramètres d'entrée du modèle

Caractéristiques matérielles des capteurs

Nous avons pris comme référence les valeurs caractéristiques des motes MICA2 (Crossbow Technology Inc., n.d.), puisque nous les utilisons au laboratoire. Les valeurs que nous avons adoptées sont tirées des documentations techniques (*ATmega128(L) Summary*, n.d.) et des expériences présentées dans (Polastre *et al.*, 2004; Shnayder *et al.*, 2004; Marhur *et al.*, 2006). Ces valeurs sont données dans le tableau 3.1.

La puissance de transmission de -20 dBm est la plus petite qui peut être configurée sur les motes MICA2. Cela permet en pratique de communiquer dans un rayon de 5 mètres environ. Avec les valeurs indiquées dans le tableau 3.1, nous pouvons calculer l'énergie consommée pour les principales opérations sur les données de l'image (appliquer la TO, transmettre un paquet, le recevoir). Pour donner un ordre de grandeur, disons que la transmission de données coûte $5,6\mu$J par octet, la réception $10,5\mu$J par octet et, le basculement de mode coûte $5,3\mu$J. L'application de la TO sur une image monochrome de 128×128 pixels coûte au total 151mJ lors'elle est exécutée une seule fois (décomposition de l'image en deux niveaux de résolution). Son coût est de 188mJ lorsqu'elle est exécutée récursivement deux fois (trois niveaux de résolution).

TAB. 3.1: Paramétres relatifs aux motes MICA2

Variables	Description	Valeur
V_B	Voltage fourni par la batterie d'un nœud	3V
C_{Tx} (-20)	Intensité du courant dans le *transcepteur* radio pour transmettre 1 octet (à -20 dBm)	3,72mA
C_{Rx}	Intensité du courant dans le *transcepteur* radio pour recevoir 1 octet	7,03mA
C_{Sw}	Intensité du courant dans le *transcepteur* radio pour basculer de mode (rx/tx)	15mA
T_{Tx}	Temps écoulé pour envoyer/recevoir 1 octet	499μs
T_{Sw}	Temps écoulé pour basculer de mode (rx/tx)	250μs
ε_{shift}	Energie dissipée dans le micro-controleur pour exécuter une instruction de décalage sur octet	3,3nJ
ε_{add}	Energie dissipée par le micro-controleur pour exécuter une instruction d'addition sur octet	3,3nJ
ε_{rmem}	Energie dissipée pour lire 1 octet en mémoire Flash	0,26μJ
ε_{wmem}	Energie dissipée pour écrire 1 octet en mémoire Flash	4,3μJ

Caractéristiques des paquets

Les motes MICA2 fonctionnent sous TinyOS/nesC développé par l'UC-Berkeley (UC Berkeley, n.d.). La taille maximale des messages imposée par TinyOS est de 255 octets (Thorn, 2005). Le kit de développement prédéfinit plusieurs formats de messages. Nous avons utilisé des messages `Multihop` pour encapsuler nos paquets de données et d'acquittements. Les messages `Multihop` réservent 17 octets pour les informations d'en-tête et de synchronisation (voir figure 3.8(a)). L'en-tête de nos paquets étant codé lui-même sur 4 octets (voir figure 3.8(b)), il reste 234 octets au maximum pour la charge utile. De la même manière, nos acquittements sont codés sur 3 octets, qui contiennent l'identifiant de l'image (codée sur 1 octet), et l'offset du paquet (codé sur 2 octets) comme montré figure 3.8(c). Nous avons donc fixé $L_{Ack} = 20$.

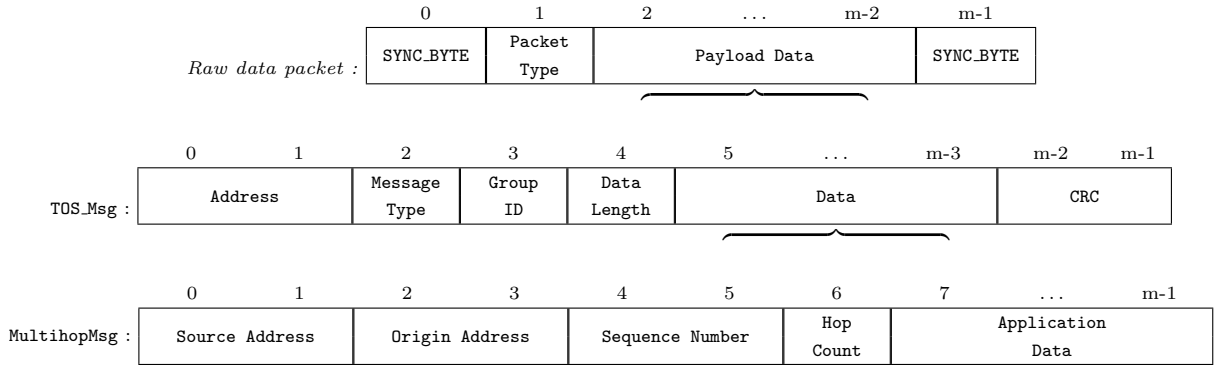
Concernant l'affectation des valeurs des seuils d'énergie pré-programmés dans les motes, nous avons choisi arbitrairement une distribution uniforme, c'est-à-dire $\alpha_\ell = \frac{\ell}{p}, \forall \ell \in \{0, 1.., (p-1)\}$, p étant le nombre de priorités des données.

3.3.2 Coût d'énergie avec un protocole fiable

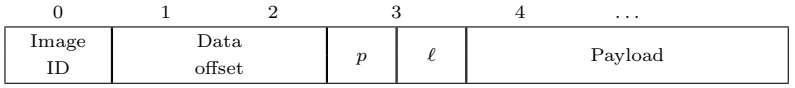
Nous avons d'abord calculé l'énergie consommée de bout en bout pour transmettre l'image test originale, donc sans appliquer de TO. L'image faisant 16393 octets, la source doit donc préparer 71 paquets en moyenne de 231 octets, tous de priorité 0. Dans ce scénario, la transmission est fiable de bout en bout. Dans le cas le plus favorable, c'est-à-dire sans aucun erreur de transmission, l'énergie consommée, normalisée par le nombre de sauts, est constante, égale à 312mJ par saut. Cette valeur va nous servir de référence pour calculer les gains d'énergie que peuvent être obtenus avec un protocole semi-fiable.

3.3.3 Coût d'énergie avec un protocole semi-fiable

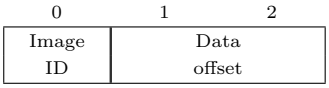
Maintenant, considérons des scénarios où l'image est divisée en plusieurs niveaux de résolutions, puis ses données rangées dans de paquets de différentes priorités.



(a) Format de paquet `Multihop` de TinyOS



(b) Format de paquet d'image



(c) Format de paquet d'acquittement

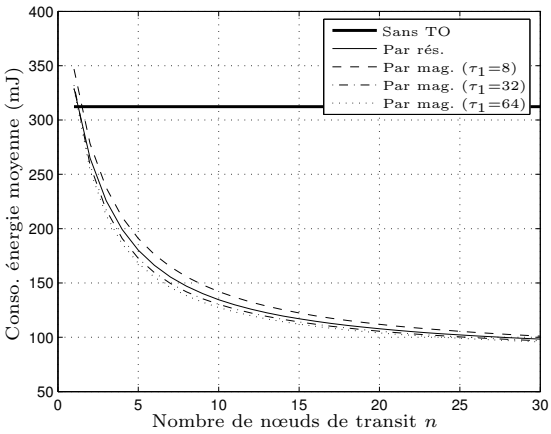
FIG. 3.8: Formats de paquets utilisés comme paramètre dans notre modèle

Un premier scénario consiste à appliquer la TO une seule fois, puis de construire des paquets de données de différentes priorités selon la stratégie basée sur les résolutions de l'image. Il y aura donc deux niveaux de priorités. Dans ce scénario, les données relatives à la résolution 0 occupent 4106 octets (entête d'image compris) et celles relatives à la résolution 1 occupent 12288 octets. La source va donc préparer 18 paquets de priorité 0 ayant une taille moyenne de 228 octets, et 53 paquets de priorité 1 d'une taille moyenne de 232 octets.

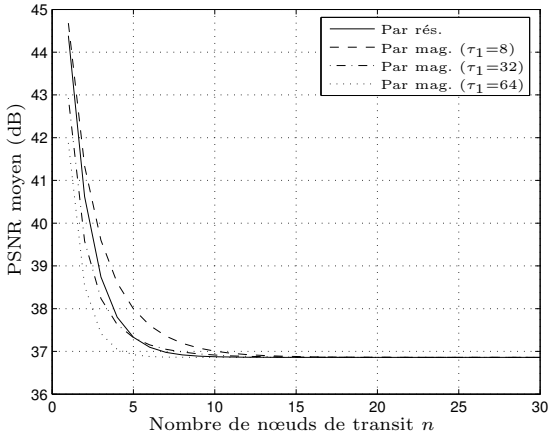
Un deuxième scénario consiste toujours à appliquer la TO une seule fois, mais en adoptant cette fois-ci une stratégie basée sur la magnitude des coefficients d'ondelette. Considérons simplement trois niveaux de priorité, c'est-à-dire un seuil de magnitude τ_1. Prenons arbitrairement $\tau_1 = 8$. Dans ce cas, pour notre image test, la source va préparer 18 paquets de priorité 0 d'une taille moyenne de 228 octets, 25 paquets de priorité 1 d'une taille moyenne de 228 octets, et 29 paquets de priorité 2 d'une taille moyenne de 228 octets. Nous avons aussi considéré des scénarios avec $\tau_1 = 32$ et $\tau_1 = 64$. Les performances de tous ces scénarios sont données figure 3.9(a) pour la consommation d'énergie et figure 3.9(b) pour le PSNR.

Les résultats montrent que quand le chemin est long d'un seul saut, le coût est supérieur à la valeur de référence puisque l'énergie consommée par la TO à la source s'ajoute au coût de la transmission. Mais ensuite, l'espérance de gain augmente vite : sur un chemin de 10 nœuds de transit, en considérant une prioritisation par niveaux de résolution, le coût d'énergie moyen est de 135mJ par saut (soit une économie de 57% par rapport à la valeur de référence), de 108mJ par saut avec 20 noeuds de transit (65% d'économie) et de 98mJ par saut avec 30 noeuds de transit (69% d'économie).

Les différences entre les stratégies de prioritisation par résolution et par magnitudes de coefficients d'ondelette sont, en moyenne, très petites. Ces différences sont plus remarquables dans une première étape, quand la quantité de nœuds de transit est entre 5 et 15 nœuds, et elles deviennent moins significatives quand le nombre de nœuds à traverser augmente. Quand un petit seuil τ_1 est appliqué dans la politique de prioritisation ($\tau_1 = 8$ dans notre exemple) la quantité moyenne d'énergie dépasse celle consommée dans



(a) Consommation d'énergie moyenne avec une TO appliquée.

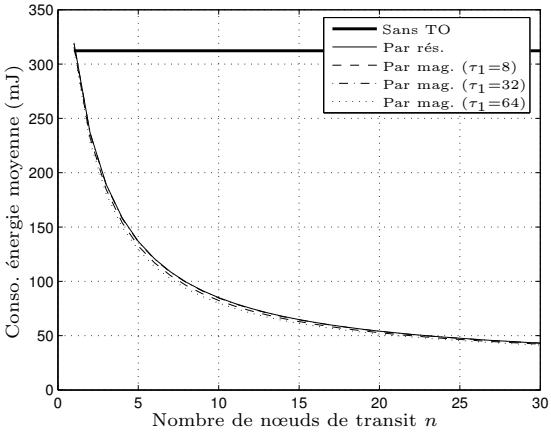


(b) PSNR moyen avec une TO appliquée.

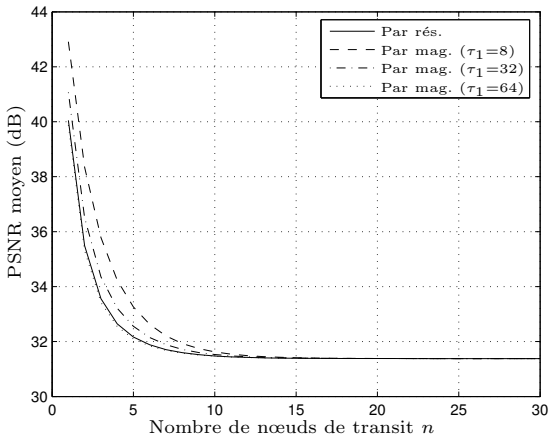
FIG. 3.9: Évaluation de la transmission semi-fiable par niveaux de résolution et par magnitudes de coefficients d'ondelettes avec une TO appliquée sur une image de 128×128 pixels.

le cas par résolution. Des seuils plus grands amènent des économies plus importantes, mais au détriment de la qualité des images (comme noté dans le graphique de la figure 3.9(b)).

Reprenons maintenant tous ces scénarios, mais en appliquant la TO deux fois. Les données relatives à la résolution 0 occupent cette fois-ci 1036 octets, celles relatives à la résolution 1 occupent 3072 octets, et celles relatives à la résolution 2 occupent 12888 octets. Les performances sont données figure 3.10(a) et 3.10(b).



(a) Consommation d'énergie moyenne avec une TO appliquée.



(b) PSNR moyen avec une TO appliquée.

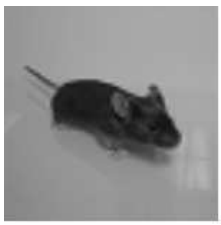
FIG. 3.10: Évaluation de la transmission semi-fiable par niveaux de résolution et par magnitudes de coefficients d'ondelettes avec deux TO appliquée sur une image de 128×128 pixels.

Les résultats confirment que la transmission avec une fiabilité partielle amène des économies d'énergie plus importantes en moyenne quand le nombre de nœuds a parcourir est grand. Le cas avec deux TO implique plus d'économies d'énergie en moyenne que le cas précédent, parce que l'image est décomposée en plus de niveaux de résolution et, en conséquent, la quantité d'information a envoyer de manière fiable

est plus petite. Ces économies d'énergie atteignent environ 86% avec 30 nœuds de transit. Les résultats confirment encore que les économies d'énergie sont en relation directe avec la qualité des images qu'on va pouvoir reconstruire du côté du décodeur. En effet, plus les économies d'énergie sont grandes et plus la dégradation des images sera grande. La figure 3.11 montre des exemples d'images finales. Dans la figure 3.11(b), nous voyons l'image reconstruite dans le meilleur des cas, c'est-à-dire avec le scénario avec 1 TO et tous les paquets de données reçus au niveau du puits. Les figures 3.11(c) et 3.11(d) montrent les images reconstruites dans le pire des cas, pour 1 et 2 TO, respectivement, et avec seulement les paquets de priorité 0 reçus par le puits. Ces dernières images pourraient être acceptables si les exigences de l'application ne sont pas trop strictes.

(a) Image originale (128 × 128).

(b) Image reconstruite avec 1 TO, P_0+P_1 reçues (PSNR = 51.91dB).



(c) Image reconstruite avec 1 TO, P_0 reçue (PSNR = 36.86dB).

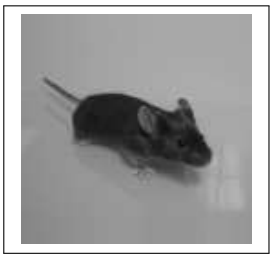
(d) Image reconstruite avec 2 TO, P_0 reçue (PSNR = 31.38dB).

FIG. 3.11: Exemples d'images reconstruites après application de la TO.

Pour conclure, nous remarquons que la plupart du temps, l'approche basée sur la magnitude des coefficients d'ondelette donne de meilleures résultats en termes de rapport qualité/énergie. Pour expliquer cet effet, prenons une décomposition typique avec 2 niveaux de TO de l'image de test. Avec la stratégie basée sur les niveaux de résolution, on obtient un P_1 (subbands HL_2, LH_2 et HH_2) de 3072 octets. Pour transmettre cette quantité de données, une mote Mica2 consomme environ 58.99mJ par saut (selon la formule 3.6). Avec notre image de test, si nous recevons au puits P_0, et que P_1 et P_2 sont perdus, nous obtenons un PSNR de 36,74dB. Avec la même image test et les mêmes niveaux de TO, nous obtenons avec la stratégie basée sur les magnitudes un P_1 de 13 paquets (3042 octets de données) en considerant $\tau = 32$. Dans ce scénario, nous avons calculé une consommation d'énergie de 57,83mJ par saut (1,16mJ de moins que le cas par résolution). En recevant P_0 et P_1 seulement, nous avons obtenu un PSNR de 39,92dB, soit une amélioration de 8.66% de la qualité d'image par rapport à une stratégie par résolution.

Cette amélioration est obtenue parce que dans le cas basé sur les niveaux de résolution, nous pouvons perdre de grandes quantités de données très importantes qui sont en P_2, et nous envoyons plusieurs paquets de données avec de coefficients faibles peu importants. En revanche, l'approche basée sur les magnitudes donne une priorité aux données les plus importantes dans toutes les résolutions. Dans la figure 3.12(c), nous pouvons apercevoir visuellement les différences commentées ci-dessus. Nous pouvons voir qu'avec la stratégie basée sur les magnitudes de coefficients (figure 3.12(c)), on obtient une bien meilleure image que si l'on applique la stratégie basée sur niveaux de résolution (3.12(b)).

Dans le cas général, nous pouvons conclure que la stratégie basée sur les magnitudes des coefficients représente une meilleure stratégie que celle basée sur les niveaux de résolution.



(a) Image originale (128 × 128).

(b) Image reconstruite avec 2 niveaux de TO, priorités par résolution, et P_0+P_1 reçues (PSNR = 36.86dB).

(c) Image reconstruite avec 2 niveaux de TO, priorités par magnitudes de coefficients, et P_0+P_1 reçues (PSNR = 39.92dB).

FIG. 3.12: Comparaison des images reconstruites après l'application de différentes stratégies de priorités et de pertes de paquets.

3.3.4 Impact de la politique de distribution des seuils d'énergie

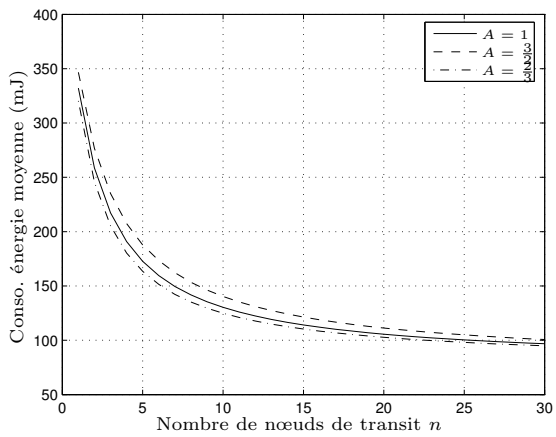
Les résultats qui nous venons de présenter ont montré l'intérêt d'utiliser un protocole semi-fiable pour économiser l'énergie sur les nœuds de transit. Les seuils d'énergie que nous avions adopté avaient été choisis arbitrairement en suivant une distribution uniforme (c'est-à-dire $\alpha_\ell = \frac{\ell}{p}$). Dans cette section, nous discutons de l'impact de la valeur des seuils sur le coût d'énergie et sur la qualité des images reçues.

Rappelons qu'une politique qui fixe les seuils à de valeurs proches de zéro aura tendance à privilégier la qualité de l'image sur les économies d'énergie, alors que des valeurs proches de 1 auront un effet inverse. Nous allons vérifier cela en comparant les performances obtenues avec plusieurs configurations des seuils d'énergie.

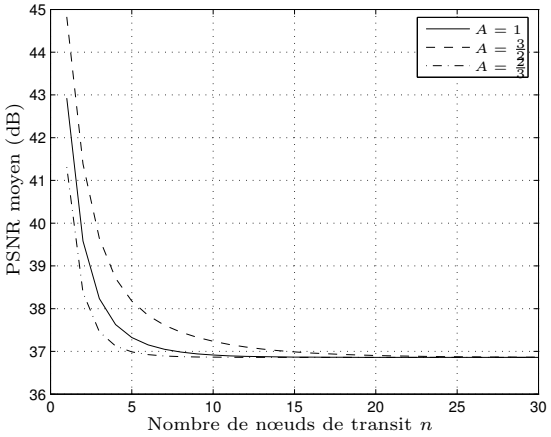
Les graphiques de la figure 3.13 considèrent de valeurs de seuils d'énergie calculées suivant la formule $\alpha_\ell = \left(\frac{\ell}{p}\right)^A$, où A est un facteur défini par l'utilisateur. Les résultats de la section précédente correspondaient au cas où $A = 1$. Si on choisit un facteur $A < 1$, les seuils d'énergie suivront une distribution de type logarithmique, en faveur des économies d'énergie. Si on choisit un facteur $A > 1$, les seuils d'énergie suivront par contre une distribution de type exponentielle favorisant la qualité d'image. Dans les courbes visualisées figure 3.13, trois valeurs de A ($A = 1$, $A = \frac{2}{3}$ et $A = \frac{3}{2}$) sont évaluées pour analyser l'impact de la distribution de seuils d'énergie. La figure 3.13(a) montre la consommation d'énergie par saut en fonction de la longueur du chemin du réseau. Les résultats montrent pour A=1 jusqu'à 68.97% de réduction sur l'énergie consommée par rapport au scénario sans TO. Des baisses de 67.72% et 69.59% sont obtenus en choisissant $A = \frac{3}{2}$ et $A = \frac{2}{3}$, respectivement. La figure 3.13(b) montre la relation entre le PSNR moyen pour 1 et 2 au niveau de TO et la longueur du chemin du réseau. Nous pouvons voir que, avec $A = \frac{3}{2}$, on obtient la meilleure moyenne de qualité d'image, au détriment des économies d'énergie.

3.4 Autres protocoles semi-fiables

Les principes techniques que nous avons exposés section 3.1 sont suffisamment généraux pour qu'ils puissent inspirer toute une classe de protocoles semi-fiables pour la transmission d'images sur réseaux de capteurs. Le protocole en boucle ouverte décrit section 3.1.4 n'est donc pas unique, et il est tout à



(a) Consommation d'énergie moyenne avec une TO appliquée.



(b) PSNR moyen avec une TO appliquée.

FIG. 3.13: Impact de l'application de différentes configurations des seuils d'énergie.

fait possible de définir d'autres protocoles semi-fiables articulés autours des mêmes principes. Dans cette section nous présentons deux nouveaux protocoles :

- un protocole en boucle fermée.
- un protocole tenant compte de la proximité du puits.

3.4.1 Protocole semi-fiable en boucle fermée

Principes du protocole

Contrairement à un protocole en boucle ouverte, nous considérons cette fois-ci que les nœuds de transit vont opérer en fonction, non seulement de l'état de leur propre batterie, mais aussi de la connaissance qu'ils ont de l'énergie disponible dans les nœuds qui les suivent sur le chemin jusqu'au puits. En d'autres mots, un nœud ayant suffisamment d'énergie pour relayer les paquets dans un niveau de priorité mais sachant qu'ils seront bloqués plus loin sur le chemin va anticiper en écartant de lui-même ces paquets. Bien sûr, les noeuds n'ont initialement aucune connaissance de l'état d'énergie des autres nœuds. Cette connaissance va être apprise au fur et à mesure que des acquittements seront reçus du voisin. Pour faire cela, un nœud qui reçoit un paquet de données va en profiter, au moment de l'acquitter, pour indiquer le plus petit état d'énergie disponible sur le chemin dont il a connaissance. Si lui-même n'a encore aucune connaissance de l'état des noeuds qui suivent, il fournit son propre état d'énergie. De cette manière, une boucle de retour est mise en place pour faire remonter l'état d'énergie disponible dans les nœuds entre la source et le puits. Ce protocole définit donc un système de transmission en boucle fermée. Le format des paquets de données reste identique au protocole en boucle ouverte. Par contre, les messages d'acquittement vont posséder un champ supplémentaire pour marquer le niveau d'énergie.

Le protocole en boucle fermée n'est pas beaucoup plus complexe que celui en boucle ouverte, mais il suppose que le chemin suivi par les paquets de bout en bout soit maintenu sur la durée de transmission d'une image complète. Aussi, il est généralement adapté aux applications basées sur un modèle de routage hiérarchique, comme c'est la cas avec protocole LEACH par exemple. Il peut aussi être adapté aux applications basées sur un modèle de type question-réponse (l'envoi d'une image par un capteur est

déclenché sur requête de l'utilisateur), en particulier sur des protocoles de routage tels que Rumor routing ou ACQUIRE.

Modélisation du protocole

Avec le protocole en boucle fermée, chaque noeud rend compte à son voisin en amont, lorsqu'il acquitte un paquet, de l'état d'énergie disponible sur le reste du chemin, tout au moins pour les noeuds dont il a eu le retour. Le retard induit par cette boucle de retour est proportionnel à la distance entre les noeuds considérés.

Etudions le déroulement de ce scénario paquet par paquet dans l'ordre dans lequel ils sont envoyés. Le noeud source transmet d'abord les m_0 paquets relatifs à la résolution 0 de l'image. Ces paquets sont obligatoirement transmis jusqu'au puits, quel que soit le niveau de charge des batteries dans les noeuds intermédiaires. Après transmission de ces paquets, le noeud source connaît par la boucle de retour le niveau de charge le plus petit parmi les noeuds 1 à m_0 au plus. De même, le noeud 1 connaît le niveau de charge le plus petit parmi les noeuds 2 à (m_0+1) au plus, et ainsi de suite. Si nous avons $m_0 \geq n$, alors le noeud source connaît l'état d'énergie disponible sur tout le chemin, et il sait donc, pour tous les niveaux de priorité, si les paquets seront acheminés jusqu'au bout ou non. Dans ce cas, la consommation d'énergie totale pour transmettre l'image dans sa totalité est exprimée par :

$$E_T^{'} = (n+1).m_0.E(t_0) + \sum_{\ell=1}^{p-1} R(\ell, n).(n+1).m_\ell.E(t_\ell) \tag{3.11}$$

avec $m_0 \geq n$. Si nous avons $m_0 < n$, alors une fois que les paquets de priorité 0 ont été envoyés, le noeud source connaît l'état d'énergie jusqu'au noeud m_0 seulement. Si les paquets de priorité 1 peuvent être acheminés jusqu'au noeud m_0, alors la source transmet le 1^{er} paquet de cette priorité au noeud 1, sinon il n'envoie plus rien. Le noeud 1, qui a une connaissance de l'état d'énergie jusqu'au noeud (m_0+1), tient compte de l'état de ce noeud (m_0+1) pour déterminer s'il engage ou pas le saut suivant pour ce paquet, et ainsi de suite. Le nombre de sauts qui sera effectué par le premier paquet de priorité 1 est donc égal à 0 si cette priorité est bloquée par un des nœuds de 1 à m_0, égal à 1 si elle est bloquée par le noeud (m_0+1), égal à 2 si elle est bloquée par le noeud (m_0+2), ... égal à i si elle est bloquée par le noeud (m_0+i), ...égal à $(n-m_0)$ si elle est bloquée par le noeud n, ou égal à $n+1$ si aucun nœud de transit est bloquant. Une fois que ce paquet est envoyé et acquitté, le noeud source connaît alors l'état d'énergie jusqu'au noeud (m_0+1), le noeud 1 celui jusqu'au noeud (m_0+2), et ainsi de suite. Passons au $2^{ème}$ paquet de priorité 1 maintenant. Suivant le même raisonnement, le nombre de sauts qui sera effectué par ce paquet est donc égal à 0 si la priorité 1 est bloquée par un des noeuds de 1 à (m_0+1), égal à 1 si elle est bloquée par le noeud (m_0+2), ... égal à i si elle est bloquée par le noeud (m_0+i+1), ...égal à $n-(m_0+1)$ si elle est bloquée par le noeud n, ou égal à $n+1$ si aucun nœud de transit est bloquant. Après avoir traité les m_1 paquets de priorité 1, le nœud source connaîtra l'état d'énergie jusqu'au noeud (m_0+m_1) si nous avons $m_0+m_1 < n$, et l'état d'énergie entre les noeuds (m_0+m_1+1) et n lui sera encore inconnu. Si on a $m_0+m_1 \geq n$ par contre, l'état d'énergie sur tout le chemin sera connu par la source au plus tard après avoir traité les $(n-m_0)$ premiers paquets de priorité 1. Pour les autres niveaux de priorité, le raisonnement peut être réitéré à l'identique.

Finalement, la généralisation du calcul de la consommation d'énergie est établie en distinguant 3 cas. En notant M_k^* la somme $m_0+m_1+\ldots+m_k$, le premier a lieu quand $M_0^* \geq n$, exprimé par l'équation

3.11. Le deuxième cas est exprimé par l'équation 3.12, où ℓ est donnée telle que $M_\ell^* < n$ et $M_{\ell+1}^* \geq n$:

$$\begin{aligned} E_T^{'} &= (n+1).m_0.E(t_0) + \sum_{k=1}^{p-1} R(k,n).(n+1).m_k.E(t_k) \\ &+ \sum_{k=1}^{\ell} \left[\sum_{j=1}^{m_k} \left(\sum_{i=0}^{n-M_k^*-j} B(k, M_k^*+j+i).(i+1).E(t_k) \right) \right] \\ &+ \sum_{j=1}^{n-M_\ell^*} \left(\sum_{i=0}^{n-M_\ell^*-j} B(\ell+1, M_\ell^*+j+i).(i+1).E(t_{\ell+1}) \right) \end{aligned} \tag{3.12}$$

Enfin, le troisième cas a lieu quand $M_{p-1}^* < n$, exprimé par l'équation 3.13 :

$$\begin{aligned} E_T^{'} &= (n+1).m_0.E(t_0) + \sum_{k=1}^{p-1} R(k,n).(n+1).m_k.E(t_k) \\ &+ \sum_{k=1}^{p-1} \left[\sum_{j=1}^{m_k} \left(\sum_{i=0}^{n-M_k^*-j} B(k, M_k^*+j+i).(i+1).E(t_k) \right) \right] \end{aligned} \tag{3.13}$$

Résultats numériques

Etudions les résultats obtenus avec le protocole en boucle fermée. On observe que l'espérance de gain est plus rapide qu'avec le protocole en boucle ouverte, grâce à l'action anticipée des noeuds qui profitent des informations fournies par la boucle de retour. Pour le scénario avec une TO (avec deux TO), on obtient 93mJ par saut pour 10 noeuds et 70% d'économie (39mJ par saut et 87%), 86mJ par saut pour 20 noeuds et 72% d'économie (29mJ par saut et 90%) et 83mJ par saut pour 30 noeuds et 73% d'économie (26mJ par saut et 91%).

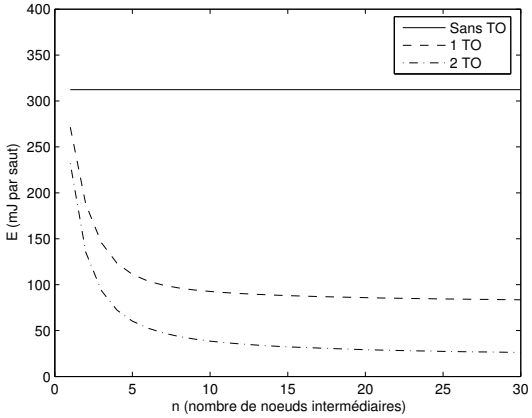


FIG. 3.14: Consommation d'énergie pour la transmission d'une image avec un protocole en boucle fermée

3.4.2 Considération de la proximité au puits

Avec un protocole de transmission semi-fiable, une diminution de la consommation d'énergie peut être obtenue théoriquement en sacrifiant la qualité de l'image finale. Mais remarquons que, lorsque les mêmes seuils d'énergie sont configurés dans tous les nœuds du réseau, un paquet peut être éliminé par un nœud très proche du puits avec la même probabilité qu'un nœud très éloigné comme illustré figure 3.15. Il semble a priori qu'il vaudrait mieux fonder la décision de relayer ou écarter les paquets en considérant l'énergie déjà investie par les nœuds précédents. Pour mettre en oeuvre une telle politique, les seuils d'énergie fixés dans les nœuds de transit doivent varier en fonction de leur proximité au puits ou, de la même manière, de leur distance par rapport à la source. Pour cela, il suffit d'utiliser une fonction de pondération de seuils d'énergie, caractérisée par $f(1) = 1$ et $\lim_{i\to\infty} f(i) = 0$, où i est le nombre de saut accomplis par le paquet depuis la source. En multipliant les seuils d'énergie par la valeur de $f(i)$ dans chaque nœud de transit, la probabilité de rejet d'un paquet de priorité ℓ va diminuer à mesure qu'il s'éloigne de la source, et donc qu'il se rapproche du puits. Pour mettre en œuvre cette proposition, le nœud qui reçoit un paquet doit savoir combien de sauts ont déjà été effectués. Un champ indiquant le nombre de sauts réalisés par le paquet a donc besoin d'être rajouté dans l'en-tête du paquet. Cet compteur de sauts sera utilisé comme paramètre d'entrée pour la fonction $f(i)$.

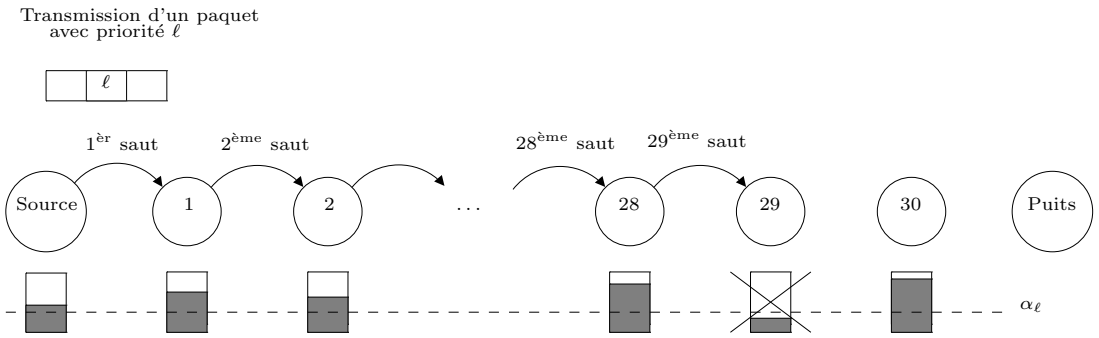


FIG. 3.15: Illustration de la problématique de la proximité au puits.

Adaptation des fonctions de probabilité $R(\ell, n)$ et $B(\ell, i)$

En prenant en compte la proximité au puits, considérée grâce à la fonction de pondération $f(i)$, les équations de probabilité que les paquets de priorité ℓ arrivent jusqu'au puits $R(\ell, n)$, et de probabilité qu'ils soient écartés au nœud i deviennent :

$$R(\ell, n) = \prod_{k=1}^{n} [1 - f(k).\alpha_\ell] \tag{3.14}$$

$$B(\ell, i) = \alpha_\ell . f(i) . \prod_{k=1}^{i-1} [1 - f(k).\alpha_\ell] \tag{3.15}$$

Les équations 3.14 et 3.15 seront utilisées pour l'estimation de la consommation d'énergie par nœud pour les protocoles en boucle ouverte, en utilisant l'équation 3.5, et en boucle fermé, en utilisant les équations 3.11, 3.13 et 3.12.

Proposition d'une fonction $f(i)$

Maintenant, quelle fonction $f(i)$ pouvons-nous utiliser pour faire évoluer les seuils d'énergie alors que nous nous approchons du puits ? Les réponses peuvent être multiples.

Laissez-nous analyser une fonction générique $f(i)$ définie comme suit :

$$f_{a,b}(i) = e^{-\left(\frac{i-1}{b}\right)^a} \tag{3.16}$$

où a et b (avec $a, b > 0$) représentent des facteurs de concavité et de contraction, respectivement. La figure 3.16 illustre l'effet de chaque paramètre sur la fonction $f_{a,b}(i)$ par un chemin constitué de 30 nœuds de transit. L'utilisateur va pouvoir ajuster les paramétres a et b en fonction des besoins de l'application. Plus a augmente, et plus les nœuds en début de chemin vont appliquer des seuils d'énergie proches des valeurs originales (lorsque les paquets ont traversé une « courte distance »). Quand une plus grande distance a été parcourue, les seuils d'énergie vont diminuer de manière plus rapide (les nœuds de transit verront leur probabilité de relayer des paquets s'accroître). Concernant le facteur b, plus il diminue, et plus la fonction $f_{a,b}(i)$ sera contractée (voir sur la figure 3.16 le changement de $f_{4,15}(i)$ à $f_{4,10}(i)$), et plus les seuils d'énergie diminueront rapidement. En revanche, avec de valeurs plus grandes de b, $f_{a,b}(i)$ sera plus allongé (voir sur la figure 3.16 le changement de $f_{4,15}(i)$ à $f_{4,20}(i)$), et les seuils d'énergie diminueront plus lentement. Si les deux facteurs a et b augmentent, la fonction $f_{a,b}(i)$ tendra vers la valeur 1, ce qui signifie que la même politique sera appliquée par chaque nœud sur le long du chemin.

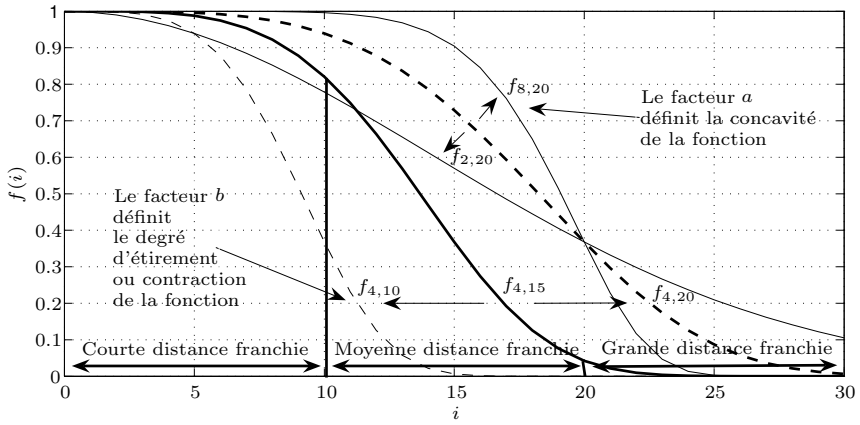
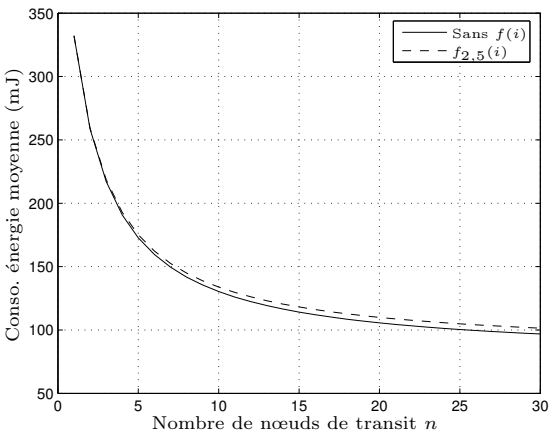


FIG. 3.16: Effet des coefficients de contraction et de concavité.

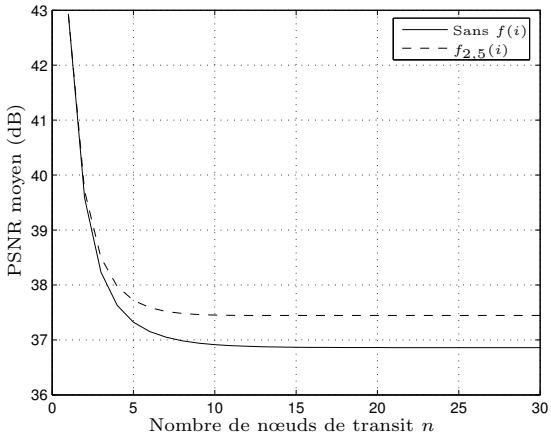
Résultats numeriques

Analysons les résultats exposés figure 3.17. Les courbes évaluent les performances de l'application d'une transmission semi-fiable en boucle ouverte d'une image décomposée en 2 niveaux de résolutions et avec une politique de prioritisation par magnitudes de coefficients avec un seuil $\tau_1 = 32$. Comme il était attendu, la considération de la proximité au puits provoque une montée de l'énergie moyenne consommée par le réseau. Dans notre exemple, on peut voir que quand on applique une fonction de proximité $f(i) = e^{-\left(\frac{i-1}{5}\right)^2}$, l'énergie moyenne consommée avec 30 nœuds de transit augmente de 4.82%, en comparaison avec le cas sans consideration de la proximité au puits. Évidemment, la qualité des images

réçues du côté du décodeur est aussi augmentée. Dans le graphique de la figure 3.17(b) on peut voir que cette augmentation atteint 37.45dB, quand 30 nœuds de transit sont considérés dans notre modèle, par rapport au scénario en absence de fonction de proximité (PSNR moyen = 36.86dB).



(a) Consommation d'énergie moyenne avec une TO appliquée.



(b) PSNR moyen avec une TO appliquée.

FIG. 3.17: Comparaison de performances d'une transmission semi-fiable en boucle ouverte sans et avec considération de proximité au puits.

3.5 Conclusion

Dans ce chapitre, nous avons présenté un système de transmission d'images efficace en énergie pour les réseaux de capteurs sans fil en jouant sur la relaxation de la contrainte de fiabilité. Ce système est basé sur une décomposition de l'image en plusieurs niveaux de priorité, grace à une transformée en ondelettes dyadique, et sur un protocole de transmission semi-fiable. Plusieurs variantes ont été proposées se distinguant par la nature du protocole de communication, en boucle ouverte ou en boucle fermée, par la stratégie de prioritisation des données de l'image, par résolutions de l'image ou par magnitudes des coefficients d'ondelette, et par la politique d'affectation des seuils d'énergie dans les nœuds de transit, avec ou sans adapration dynamique. Les choix se feront à partir des besoins de l'application et de l'organisation du réseau de capteurs. Le choix entre la boucle ouverte et la boucle fermée dépendra du protocole de routage déployé dans le réseau de capteurs. En fait, si les chemins entre les sources et le puits sont persistants sur le temps de transmission d'une image complète, le protocole en boucle fermé est le meilleur choix, sinon la boucle ouverte est plus appropriée. En accord aux contraintes des réseaux de capteurs sans fil, nos propositions sont clairement faciles à mettre en œuvre, et elles permettent aux nœuds d'avoir un comportement autonome et auto-adaptatif pour fournir un compromis entre la qualité de l'image reçue et l'énergie dissipée dans le réseau, avec la garantie que les images finales auront un niveau de qualité avec un minimum borné.

Les protocoles semi-fiables que nous avons proposés sont basés sur une politique tenant compte de la quantité d'énergie disponible dans les nœuds de transit. Bien évidemment, la conséquence est que les nœuds du réseau beaucoup vont, au début de leur cycle de vie, relayer des paquets de basse priorité. Puis au cours du temps, quand les nœuds auront dépensé une certaine quantité de leur énergie, les paquets

de basse priorité seront éliminés. Une autre politique serait d'allouer des crédits d'énergie aux nœuds de transit, par exemple des crédits journaliers. Et les nœuds de transit décideraient de relayer ou non les paquets de basse priorité en fonction de ce crédit et de ce qu'ils auraient déjà consomé sur la période consideré.

Chapitre 4

Amélioration de la résistance aux pertes par entrelacement de pixels

Nous avons présenté dans le chapitre 3 des solutions pour réduire la quantité d'énergie consommée pour transmettre une image en relâchant partiellement la contrainte de fiabilité au niveau du protocole de communication. Ces solutions servaient à contrôler le nombre de paquets et d'acquittements échangés dans le réseau en fonction de l'état des batteries dans les nœuds de transit, selon une logique où plus les nœuds manquaient d'énergie, et plus ils écartaient de paquets. Dans l'horizon de la communication de bout en bout, les protocoles étaient semi-fiables puisqu'ils garantissaient la livraison des paquets de la plus haute priorité seulement. Des paquets de moindre priorité pouvaient donc être perdus, pertes que l'on peut qualifier de volontaires puisqu'elles étaient le fait des nœuds de transit. Il n'y avait pas d'autres causes de pertes de paquets puisque la communication de proche en proche se faisait par un protocole fiable se basant sur un mécanisme d'acquittement et de retransmission. Autrement dit, les pertes dues au fait que les transmissions sans fil sont faillibles (erreurs de transmission dues aux interférences, possibilité de collisions et de congestions) étaient prises en charge par le protocole de communication.

Pour économiser encore plus d'énergie, nous pensons qu'il serait intéressant de s'affranchir totalement de la contrainte de fiabilité, et donc d'éliminer tout le mécanisme d'acquittement et de retransmission des paquets. Dans ce cas évidemment, la qualité minimale des images finales ne peut plus être bornée. Mais les images naturelles ont généralement une tolérance aux pertes assez élevée en raison des corrélations qui existent statistiquement entre pixels voisins. Dans ce chapitre, nous nous proposons d'étudier l'effectivité de la transmission d'image par un protocole non fiable sous la contrainte des pertes de paquets. Nous montrons en particulier que cette approche est viable sous la condition d'appliquer une technique d'entrelacement de pixels à la source, avant transmission. Dans ce cas, les images transmises peuvent tolérer au-delà de 50% de pertes de paquets puisque les dégradations de qualité constatées à l'arrivée restent dans des limites acceptables pour la majorité des applications.

Le chapitre est structuré en 6 parties. La section 4.1 présente le problème des pertes de paquets sur la transmission d'images et ses effets sur la qualité des images finales. Elle présente aussi les méthodes de dissimulation d'erreurs qui peuvent être appliquées au récepteur pour estimer les intensités des pixels perdus. La section 4.2 présente les principes de l'entrelacement de pixels, comme technique de préparation des données au niveau du nœud source pour améliorer la tolérance des images aux pertes de paquets. Dans

la troisième section, nous exposons les automorphismes du Tore, que nous avons adoptés pour réaliser l'entrelacement de pixels. Nous proposons aussi une adaptation de l'algorithme traditionnel pour rendre plus efficace l'implantation des automorphismes toriques dans les dispositifs limités en ressources. Les résultats en termes de qualité d'image et de consommation d'énergie sont donnés respectivement dans les sections 4.4 et 4.5. Des consignes par rapport à la sélection des paramètres de la fonction d'entrelacement sont discutés section 4.6.

4.1 Perte de données et dissimulation d'erreurs

4.1.1 Principes de la dissimilation des erreurs

Au delà du problème de la consommation d'énergie, un autre problème posé par les réseaux de capteurs sans fil concerne les pertes de paquets. Pour mieux comprendre ce problème prenons l'exemple suivant : La figure 4.1(a) montre une image monochrome prise avec un capteur d'image Cyclops et codée sur 8bpp. Les intensités de chaque pixel sont donc des valeurs comprises entre 0 et 255. La table jointe à droite de l'image montre en notation décimale une petite portion des données qui la composent. Imaginons qu'on perde un seul pixel, par exemple celui qui est à la position (70,4) dans l'image comme le montre la figure 4.1(b). En partant de l'hypothèse que dans les images naturelles, les pixels voisins ont une probabilité élevée d'avoir des valeurs proches, on peut calculer une valeur approximative du pixel manquant en s'aidant des valeurs de ses voisins pour dissimuler la perte dans l'image finale. La méthode d'estimation la plus couramment utilisée est surement celle qui prend la valeur moyenne de tous les pixels voisins correctement reçus. Dans notre exemple, l'intensité estimée du pixel manquant $x_{70,4}$ est donc calculée comme $\widetilde{x}_{70,4} = \frac{1}{8}\left(x_{69,3} + x_{69,4} + x_{69,5} + x_{70,3} + x_{70,5} + x_{71,3} + x_{71,4} + x_{71,5}\right)$. Le résultat, 158, a seulement une différence de 4 par rapport a la valeur originale (ce qui est peu si on considère qu'il y a 256 valeurs possibles). L'image finale après dissimulation du pixel manquant est exposée figure 4.1(c). Visuellement, on voit qu'on obtient bien une version de l'image qui est très proche de l'originale. Cela est confirmé objectivement puisqu'on obtient un PSNR de 78.23dB, ce que veut dire que l'image finale a une distortion très faible par rapport à l'originale. Cet exemple est symbolique et en pratique, nous n'aurons pas affaire à des pertes de pixels isolés puisque les pixels seront transmis par paquets, c'est-à-dire par groupes consécutifs.

4.1.2 Effets des pertes de paquets sur la qualité de l'image finale

La figure 4.2 présente un exemple de transmission d'image où un paquet est manquant au récepteur. Comme les paquets sont remplis avec des pixels pris consécutivement dans l'ordre de leur rangement ligne par ligne, la perte d'un paquet est visuellement marquée par un trait noir d'une longueur équivalent au nombre de pixels que contenait le paquet (27 dans l'exemple). La dissimulation des pixels manquants est réalisée en calculant la valeur moyenne des pixels voisins correctement reçus, de manière analogue à l'exemple de la figure 4.1. L'image finale est visualisée figure 4.2(c). Nous voyons que le PSNR a diminué de 20.2 dB par rapport à l'exemple de la perte d'un seul pixel, mais cela est encore un excellent résultat puisque la difference de qualité entre l'image originale et l'image finale est imperceptible à l'œil.

Bien sûr, le nombre de pertes de paquets peut être beaucoup plus élevé en pratique. Nous avons simulé des transmissions d'images avec pertes de paquets sur la base des traces que nous avons collectées (cf.

(a) Image originale « Corridor »

	2	3	4	5	6
68	164	162	164	155	146
69	148	162	162	162	139
70	164	162	**162**	146	146
71	148	162	148	157	139
72	164	162	164	157	146

(b) Image avec un pixel perdu

	2	3	4	5	6
68	164	162	164	155	146
69	148	162	162	162	139
70	164	162	**X**	146	146
71	148	162	148	157	139
72	164	162	164	157	146

(c) Image final après dissimulation du pixel manquant (PSNR = 78.23 dB)

	2	3	4	5	6
68	164	162	164	155	146
69	148	162	162	162	139
70	164	162	**158**	146	146
71	148	162	148	157	139
72	164	162	164	157	146

FIG. 4.1: Illustration de la perte d'un pixel sur une image monochrome et de son estimation à partir des pixels voisins.

(a) Image originale « Corridor »

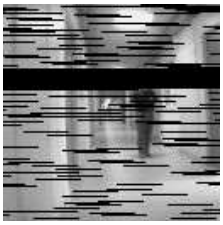
(b) Image reçue

(c) Image finale après dissimulation des pixels manquants (PSNR = 58.03 dB)

FIG. 4.2: Illustration de la perte d'un paquet sur une image monochrome et de la dissimulation des pixels manquants.

section 1.3.4). La figure 4.3 illustre l'effet typique des pertes de paquets pour un pourcentage de pertes de 29%. L'image reçue est montrée figure 4.3(b).

(a) Image originale « Corridor »



(b) Image reçue avec 29% de pertes de paquets

(c) Image finale après dissimulation des pixels manquants (PSNR = 25.63 dB)

FIG. 4.3: Visualisation des pertes du a la transmission non-fiable d'une image sans traitement au niveau de la source.

Comme on peut le constater, les pertes de paquets peuvent produire des pertes de pixels concentrées sur des grandes régions de l'image. Dans ces régions, où beaucoup de pixels voisins sont perdus simultanément, les méthodes d'estimation de pixels deviennent inefficaces. Sur l'image finale visualisée figure 4.3(c) on voit bien que la région inférieure de l'image n'a pas été trop affectée pour la perte de pixels. Par contre, la région supérieure de l'image a subi de pertes impossibles à dissimuler puisqu'il n'y a aucun indice sur la valeur des pixels perdus. Cet exemple montre les limites des méthodes de dissimulation d'erreurs lorsque le nombre de paquets perdus augmente. Cela montre aussi que l'adoption d'un protocole non fiable pour transmettre les images, bien que cela soit intéressant du point de vue de la consommation d'énergie, est discutable du point de vue de la qualité des images, faute de garantie d'un niveau de qualité minimum acceptable.

Faut-il alors réintroduire un mécanisme de correction des pertes de paquets au niveau du protocole de communication ? Des mécanismes de type ARQ ou FEC peuvent en effet être utilisés pour avoir moins de pertes de paquets. Pour un protocole ARQ, le prix à payer en termes d'énergie peut être évalué analytiquement en utilisant le modèle de transcepteur radio décrit dans le chapitre 3.

Il peut aussi être évalué expérimentalement en utilisant notre plateforme de mesure d'énergie. Nous avons comparé le coût d'énergie de deux applications : l'une transmet les paquets de l'image avec un protocole non fiable (sans acquittement ni retransmission) et l'autre avec un protocole ARQ classique. Pour une image monochrome de 128×128 pixels, la consommation d'énergie pour capturer et transmettre l'image sans ARQ était de 2307 mJ et le temps d'exécution était de 25.55 secondes. Pour l'application utilisant ARQ, on obtenait une consommation d'énergie de 3690 mJ et un temps d'exécution de 48.95 secondes lorsqu'il n'y avait aucune erreur de transmission (donc dans le cas le plus favorable). C'est une augmentation de 60% de la consommation d'énergie et de 66% du temps d'exécution. Ces résultats montrent aussi que le protocole ARQ a un coût d'énergie du minimum de 2.28mJ par paquet et ajoute un délai au minimum de 32ms par paquet. C'est beaucoup. En présence d'erreurs de transmission sur le paquet ou sur l'acquittement, le coût va augmenter en proportion.

Nous n'avons pas évalué une protection par FEC qui serait aussi une solution possible.

Pour éviter d'utiliser un protocole ARQ, même limité à un nombre maximum de retransmissions, on peut renforcer la résistance de l'image aux pertes de paquets avec une méthode d'entrelacement de pixels. Une telle méthode vise à décorréler spatialement les pixels voisins dans l'image de manière à les embarquer

dans des paquets distincts. La probabilité de perdre un pixel et (tous) ses voisins va aussi diminuer et par conséquent, cela va augmenter l'efficacité de la méthode de dissimulation des pixels manquants.

4.2 Entrelacement de pixels

Dans des mots simples, l'entrelacement de pixels consiste a changer la séquence initiale des pixels d'une image, c'est-à-dire, chaque pixel x positionné aux coordonnées (i, j) est, après une certaine transformation inversible, déplacé à une nouvelle position (i', j'). L'entrelacement de pixels, appelé aussi mélange de pixels, a été utilisé sur une large gamme d'applications pour la sécurité, en combinaison avec le tatouage d'images (Voyatzis et Pitas, 1996; Chen *et al.*, 2003) ou pour la récupération de pixels, en combinaison avec la dissimulation d'erreurs (Turner et Peterson, 1992). En ce qui concerne la récupération des pixels perdus pendant la transmission d'une image sur un réseau non fiable, l'entrelacement de pixels peut être un mécanisme très utile puisqu'il permet de diminuer la probabilité de perdre à la fois un pixel et ses voisins. Voyons l'exemple de la figure 4.4.

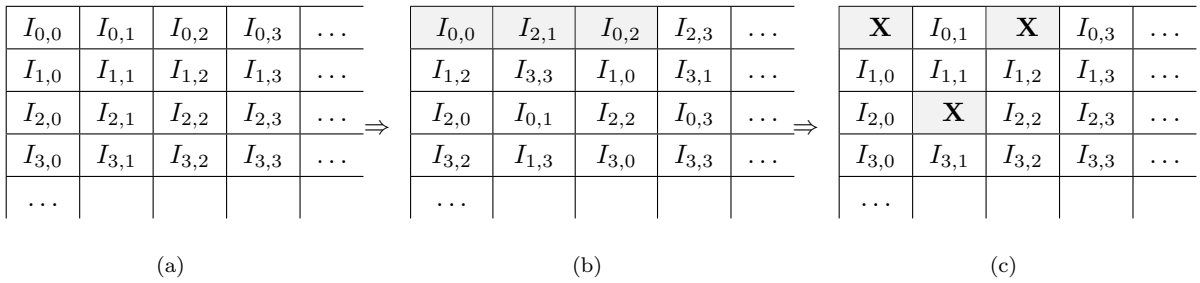


FIG. 4.4: Exemple d'entrelacement de pixels pour la communication d'images sur un réseau sujet à des pertes de paquets.

Soit l'image matricielle $I = \{I_{0,0}, I_{0,1}, I_{0,2}, ...\}$ comme montré figure 4.4(a). Supposons arbitrairement que l'image est transmise par paquets de 3 pixels. Si l'image n'est pas mélangé, le premier paquet contiendra les pixels $I_{0,0}$, $I_{0,1}$ et $I_{0,2}$, le deuxième paquet contiendra $I_{0,3}$, $I_{0,4}$ et $I_{0,5}$, et ainsi de suite. Évidemment, si on perd un paquet, les pixels perdus sont voisins. Par exemple, si on perd le premier paquet, un voisin est déjà perdu pour récupérer $I_{0,0}$ ou $I_{0,2}$, et deux voisins sont déjà perdus pour récupérer $I_{0,1}$. Donc, la méthode de dissimulation d'erreurs sera peu efficace. A l'aide d'une méthode d'entrelacement de pixels, ce problème va s'atténuer. Imaginons une méthode d'entrelacement qui nous donne la configuration exposée sur la figure 4.4(b). Dans ce cas, le premier paquet contiendra les pixels $I_{0,0}$, $I_{2,1}$ et $I_{0,2}$, et ainsi de suite. Si on perd juste le premier paquet, et après avoir réalisé l'entrelacement inverse (figure 4.4(c)) les pixels perdus sont des points isolés dans le bitmap reconstitué. Ils seront plus facilement récupérables à l'aide d'une méthode de dissimulation d'erreurs. L'intérêt théorique est évident, le problème est maintenant de trouver une méthode d'entrelacement efficace pour les réseaux de capteurs, c'est-à-dire qui coûte peu d'énergie et qui permette de reconstruire des images de bonne qualité même avec des taux de perte de paquets élevés.

Pour être efficace, la fonction d'entrelacement des pixels doit assurer que les pixels soit distribués de la meilleure façon possible dans le bitmap, c'est-a-dire, que les pixels que sont voisins doivent être les plus aloignés les uns des autres de façon que si un pixel est perdu on puisse récupérer ses voisins dans

des autres paquets. Par ailleur, pour être applicable sur un nœud de capteur, la fonction d'entrelacement doit être le plus simple et efficace possible, de telle sorte que le mélange de pixels ne rajoute pas une complexité qui fasse dépenser excessivement temps et énergie.

(Turner et Peterson, 1992) ont proposé un mécanisme d'entrelacement de pixels basé sur deux paramètres ajustables, comme montré figure 4.5. Le premier paramètre est appelé *ByteOffset* et contrôle la distance entre deux pixels mis consécutivement dans un paquet, afin d'éviter que deux pixels adjacents soient stockés dans le même paquet. Le deuxième, appelé *PacketOffset*, précise la distance entre le premier pixel stocké dans deux paquets consécutifs. Ce deuxième paramètre permet de prendre en compte des pertes de paquets par raffales. En partant du principe que les pertes de paquets ne sont pas indépendantes, et que si un paquet est perdu, la probabilité de perdre les suivants augmente, le paramétre *PacketOffset* apporte une distance minimale entre pixels adjacents. Le mécanisme proposé est intéressant en soi mais il présente quelques inconvénients pour être appliqué sur un capteur d'image limité en ressources. Le problème principal réside dans la sélection des paramètres *ByteOffset* et *PacketOffset*. En effet, pour être applicable dans les réseaux de capteurs, les paramètres choisis doivent assurer que les cordonnées d'un pixel signalé pour être copié dans le paquet en cours de construction ne soient pas celles d'un pixel déjà copié dans un paquet antérieur. Il faut donc allouer un espace de mémoire pour marquer si les pixels ont déjà été traités ou pas. De plus, avec cette méthode, le nœud source risque de devoir parcourir l'image plusieurs fois pour rechercher les pixels qui n'ont pas été paquetisés. Cela est coûteux en temps d'exécution et en accés mémoire.

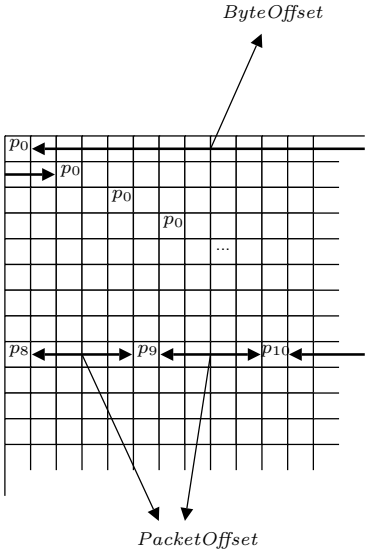


FIG. 4.5: Représentation de l'assignation de pixels selon la méthode de (Turner et Peterson, 1992).

(DeBrunner *et al.*, 1999) étudient trois méthodes d'entrelacement de pixels. Les deux premières méthodes sont les plus simples : elles sont basées sur la paquetisation de lignes (entrelacement horizontal) ou de colonnes (entrelacement vertical) paires et impaires dans des paquets différents. Les méthodes d'entrelacement horizontal et vertical sont illustrées respectivement dans les tables 4.1(a) et 4.1(b). La troisième méthode considère des blocs de 2×2 pixels, où chaque pixel d'un bloc sera stocké dans un paquet différent, comme illustré table 4.1(c). Les auteurs constatent que les méthodes d'entrelacement

vertical et horizontal sont efficaces seulement quand les bordures ou les textures dominantes de l'image sont horizontales ou verticales. Ils suggèrent donc une nouvelle méthode en considérant des blocs de 16×16 pixels et en appliquant les entrelacement verticaux ou horizontaux si les bords du bloc sont principalement horizontaux ou verticaux. Ces méthodes sont simples à implanter, mais présentent des problèmes de performance en raison de la faible distance entre les pixels voisins. De plus, ces méthodes nécessitent de rajouter de l'information suplémentaire dans les paquets pour indiquer avec quelle méthode l'entrelacement a été réalisé pour chaque bloc.

Tab. 4.1: Schémas d'entrelacement de pixels étudiés dans (DeBrunner *et al.*, 1999). Les différents symboles représentent les pixels de l'image qui doivent être transportés dans des différents paquets.

(a) Entrelacement horizontal

o	o	o	o	o	o	o	o
x	x	x	x	x	x	x	x
o	o	o	o	o	o	o	o
x	x	x	x	x	x	x	x

(b) Entrelacement vertical

o	x	o	x	o	x	o	x
o	x	o	x	o	x	o	x
o	x	o	x	o	x	o	x
o	x	o	x	o	x	o	x

(c) Entrelacement horizontal et vertical

o	x	o	x	o	x	o	x
△	•	△	•	△	•	△	•
o	x	o	x	o	x	o	x
△	•	△	•	△	•	△	•

Dans le contexte des réseaux de capteurs sans fil, le problème est de trouver une méthode qui mélange efficacement l'image et qui coûte très peu d'énergie pour la source. Dans (Chen *et al.*, 2003) nous trouvons une méthode qui semble respecter ces deux critéres. Elle est basée sur des automorphismes du Tore. Nous allons expliquer cette méthode dans la section suivante, puis nous ferons son évaluation.

4.3 Entrelacement de pixels par automorphismes du Tore

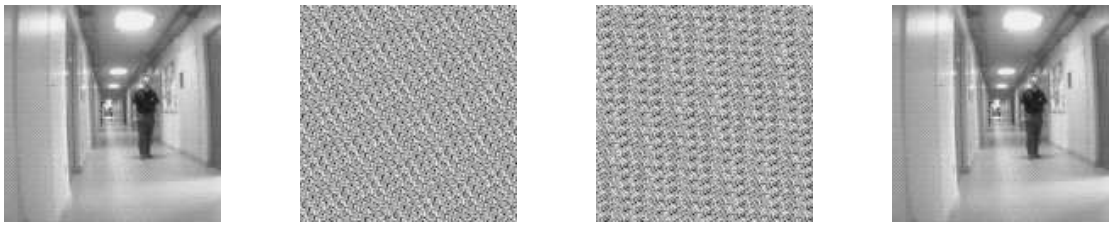
Les automorphismes du Tore (AT) ou *torus automorphisms* dans la littérature anglophone, sont des systèmes fortement chaotiques que peuvent être utilisés comme une transformée de permutation bidimensionnelle (Voyatzis et Pitas, 1996). Nous avons adopté cette méthode pour effectuer l'entrelacement de pixels car elle exige peu de calculs et elle fournit des performances très satisfaisantes (ce qui est vérifié à l'usage, comme nous le verrons plus loin). D'ailleurs, les ATs sont souvent utilisés pour assurer la confidentialité de la transmission d'image parce que son comportement chaotique rend difficile l'extraction de l'information depuis un agent non autorisé (c'est-à-dire sans connaître la clé du mélange). Dans cette section nous expliquerons les principes de la méthode, puis nous proposerons une adaptation pour rendre plus efficace son implantation dans les systèmes limités en ressources.

4.3.1 Principes techniques des ATs

Pour simplifier, nous allons considérer des images de $N \times N$ pixels. Nous avons adopté les AT utilisés dans (Chen *et al.*, 2003) pour le tatouage d'images compressées avec JPEG2000. L'application de l'AT sur un pixel qui se trouve aux coordonnées (i, j) fournit une nouvelle position (i', j') pour ce pixel, calculée ainsi :

$$\begin{pmatrix} i' \\ j' \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ k & k+1 \end{pmatrix}^n \begin{pmatrix} i \\ j \end{pmatrix} \mod N \tag{4.1}$$

où k est une valeur à choisir et n est une clé de diffusion ($k, n \in \mathbb{N}$). En appliquant cette transformée sur tous les pixels d'une image I, on obtient une image melangée I' comme le montre la figure 4.6 sur l'exemple de l'image test « Corridor » (figure 4.6(a)). Pour $k = 1$, et en utilisant les clés de diffusion $n = 8$ et $n = 32$, on obtient les bitmaps melangés visualisés respectivement figures 4.6(b) et 4.6(c). Remarquons



(a) Image originale « Corridor »

(b) Image mélangé avec AT et $n = 8$

(c) Image mélangé avec AT et $n = 32$

(d) Image mélangé avec AT et $n = T = 96$

FIG. 4.6: Example de l'application des Automorphismes du Tore, avec $k = 1$, sur l'image de test 'Couloir' dans une résolution de 128×128 pixels.

qu'avec une clé de diffusion $n = 96$, l'AT donne comme résultat l'image originale I, comme on peut le voir sur l'image 4.6(d). En effet, d'une façon générale, il existe toujours une clé de difusion particulière T, qui donne comme résultat la même image originale. Formellement parlant, $\forall N \in \mathbb{N}, \exists n = T / I'_T = I$, où I'_n est l'image transformée après application de l'AT avec une clé de difusion n. Cette propriété peut être utilisée du coté du décodeur pour restituer l'image originale à partir de l'image mélangé en appliquant la même transformée, mais avec la clé de diffusion $(T - n)$ cette fois-ci. Cette clé, $(T - n)$, assure donc la transformée inverse de la clé n.

4.3.2 Proposition d'adaptation des AT pour les capteurs d'image

Une implantation classique des AT requiert d'allouer un espace de mémoire de taille équivalente à celui de l'image originale pour stocker l'image mélangée sans réécrire les pixels d'entrée. En effet, l'algorithme va démarrer en considérant le premier pixel de l'image, de coordonnées (0,0), et calculer sa projection $(i', j') = \mathrm{AT}(i, j)$. Une fois que la nouvelle coordonnée est connue, l'algorithme va lire la valeur du pixel original et recopier cette valeur à l'emplacement correspondant aux coordonnées (i', j') dans l'espace alloué pour mémoriser l'image résultante. L'algorithme est ensuite réitéré pour le pixel suivant, celui de coordonnées (0,1), et ainsi de suite pour chaque pixel de l'image. Ce processus est illustré figure 4.7.

Concernant les accès mémoire, chaque pixel est donc lu et écrit deux fois, une fois pour la mélange de l'image, puis une fois pour la construction des paquets. Remarquons d'ailleurs que la transformée par AT doit être entièrement réalisée avant que le processus de paquétisation puisse commencer. Les AT vont donc introduire une certain latence entre le moment de la prise d'image et le moment où le premier paquet est envoyé sur le réseau. L'algorithme classique n'est donc pas optimisé, ni en termes d'accès et d'allocation mémoire, ni en termes de délais. Nous proposons ci-dessous une adaptation de l'algorithme qui répond parfaitement à ces deux problèmes, et nous l'appelerons AT* pour le distinguer de l'AT classique. Le principe est le suivant : Plutôt que de chercher la position projetée (i', j') pour un pixel d'entrée (i, j) donné, nous considérons que (i', j') correspond à la position du pixel original à chercher pour une position projetée (x, y). Une fois que la position (i', j') a été calculée, le pixel à cette

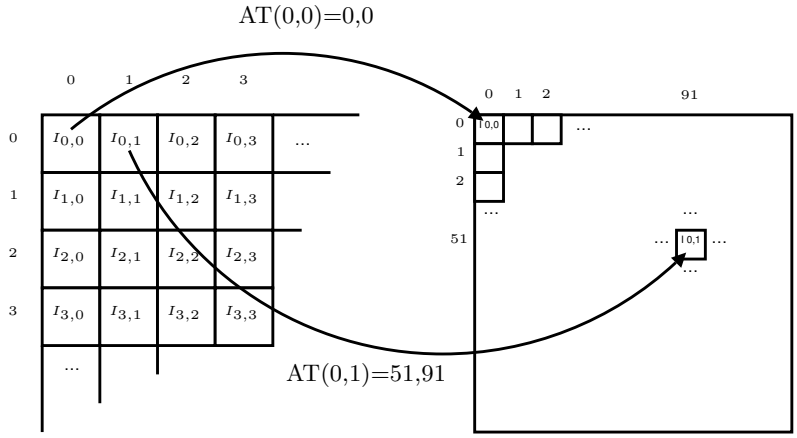


FIG. 4.7: Fonctionnement traditionnel des Automorphismes de Tore

position dans l'image originale et copié directement dans le paquet en construction. Le pseudo-code de l'AT* est montré dans l'algorithme 1.

Algorithme 1 Entrelacement de pixels basé sur l'algorithme AT*

$s \Leftarrow 0$ {position of data in packet}
$H \Leftarrow ImageHeight$, $W \Leftarrow ImageWidth$
for $i = 0$ to $H - 1$ **do**
  for $j = 0$ to $W - 1$ **do**
    Calculate (i', j') of position (i, j) using TA
    $Packet.data[s] \Leftarrow I[i', j']$
    if (packet is full) or $((i, j) = (W - 1, H - 1))$ **then**
      Send *packet*
      $s \Leftarrow 0$
    else
      $s \Leftarrow s + 1$
    end if
  end for
end for

L'algorithme commence en considérant la position projetée $(i = 0, j = 0)$ et calcule les coordonnées (i', j') du pixel correspondant dans l'image originale. Ensuite, le pixel visé $I_{i',j'}$ est copié en première position dans l'espace mémoire réservé au paquet en construction. À chaque itération, l'algorithme répète les mêmes opérations pour les pixels projetés suivants, $I_{0,1}$, $I_{0,2}$, ... et ainsi de suite jusqu'à ce que le paquet soit rempli. Une fois que le paquet en construction est complet, il est envoyé par le radio transcepteur, et l'algorithme reprend pour construire un nouveau paquet.

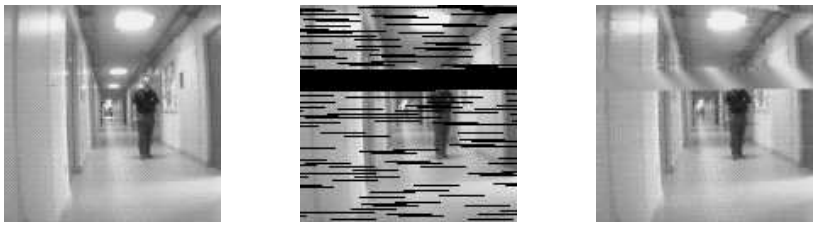
Cet algorithme n'a pas besoin d'allouer un espace mémoire supplémentaire et, en plus, chaque pixel est lu et écrit seulement une fois au lieu de deux comme cela était nécessaire avec l'algorithme standard.

4.4 Expérimentation et analyse de résultats

En exploitant les traces de perte de paquets collectées avec la plateforme décrite dans la section 1.3.4, nous avons évalué des transmissions d'image pour des scénarios avec et sans mélange à la source. Une application developpée en langage `C` et compilatée avec le `GNU gcc` sur Linux a servi à mesurer le PSNR des images finales reconstruites en présence de pertes de paquets. Les pixels manquants étaient estimés en utilisant une méthode de dissimulation d'erreurs basée sur la moyenne des pixels voisins.

Pour que la matrice génératrice de l'AT* n'ait pas de coefficients trop grands, nous suggérons d'adopter des petites valeurs pour k et n. Nous avons choisi arbitrairement $k = 1$ (appelé *cat map* dans la litérature) et $n = 8$, ce qui nous donne des coefficients qui peuvent être codés sur 2 octets. Ces valeurs donnent de bons résultats en pratique. Nous analyserons plus objectivement l'impact de n sur les performances dans la section 4.6.

La figure 4.8 montre une nouvelle fois l'effet des pertes de paquets sur l'image test « Corridor » quand l'image n'a pas été mélangée avant d'être transmise sur le réseau. Dans cet exemple, 29% de paquets avaient été perdus.



(a) Image originale « Corridor »

(b) Image reçue avec 29% de pertes de paquets

(c) Image finale après dissimulation des pixels manquants (PSNR = 25.63 dB)

FIG. 4.8: Visualisation des pertes due a la transmission non-fiable d'une image sans traitement au niveau de la source.

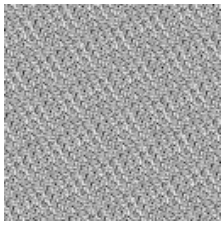
Pour comparaison, la figure 4.9 montre les résultats obtenus avec exactement les mêmes pertes de paquets, mais en mélangeant l'image à la source avant de la transmettre sur le réseau.

Nous pouvons noter que le PSNR de l'image final est amelioré de manière significative quand l'image était mélangée. Visuellement, nous observons (figure 4.9(c)) que les pertes sont réparties sur toute l'image contrairement au scénario sans mélange.

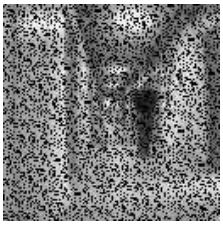
Cela est tout à fait normal puisque les pixels qui sont voisins entre eux ont été dispersés dans différents paquets. Les pixels manquants à l'arrivée sont donc mieux distribués spatiallement. L'application de l'AT* permet une diminution plus harmonieuse et progressive de la qualité de l'image quand le nombre de pertes de paquets augmente, ce qui permet de conserver une perception visuelle acceptable des objets dans le champs de vision de la caméra, même avec des pertes de paquets. Cette baisse progressive de la qualité de l'image peut être observée sur les graphes de la figure 4.10.

Visiblement, l'application des AT* avant transmission de l'image fournit des bien meilleurs résultats que la transmission d'image non préalablement mélangée. La différence de performances devient très marquée avec des taux de pertes élevés, entre 20% et 80% où l'écart dépasse 5 points de PSNR. L'application des AT assure une baisse mieux contrôlée de la qualité des images. En effet, on observe que

(a) Image originale « Corridor »



(b) Image mélangée avant transmission ($n = 8$ et $k = 1$



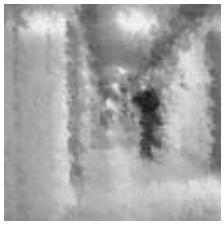
(c) Image reçue et réarrangée avec 29% de perte de paquets

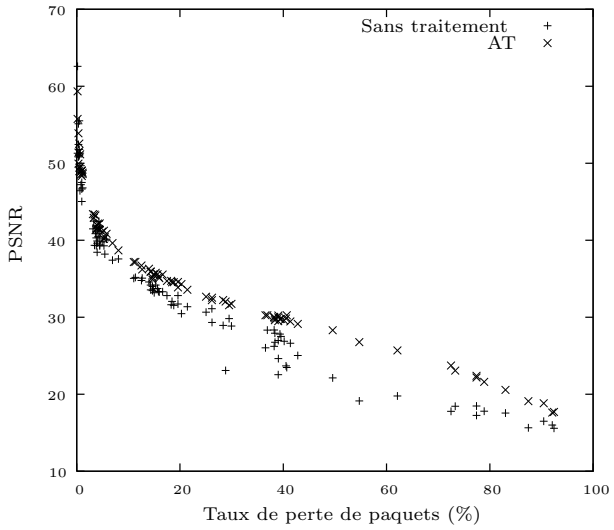
(d) Image finale après dissimulation des pixels manquants (PSNR = 33.36 dB)

FIG. 4.9: AT* appliqués sur l'image « Corridor » de 128×128 pixels.

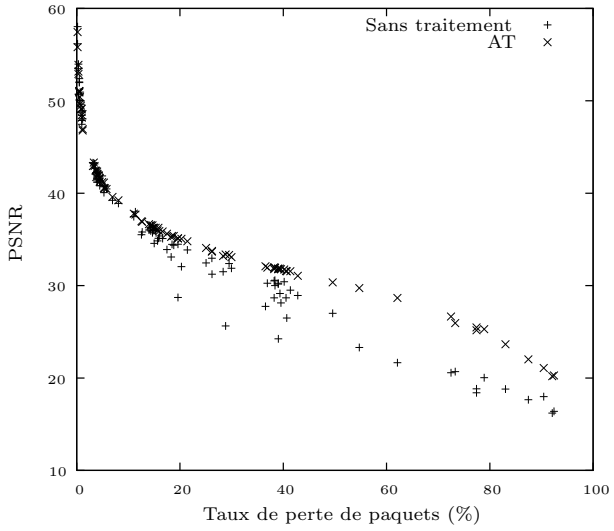
la distribution des points sur les graphiques de la figure 4.10 baisse selon une courbe bien discernable quand l'image est mélangée, contrairement au cas sans mélange, où les points révèlent une plus grande variation de la qualité d'image pour des taux de pertes très proches. Des examples d'images reconstruites pour le deux strategies étudiées, avec et sans entrelacement de pixels, peuvent être visualement comparées dans la table 4.2. On observe qu'avec le mélange de l'image on peut obtenir des image de qualité acceptables, pour des taux de pertes allant jusqu'à 40 à 50%. Par contre, au-delà de 60% de pertes, les images commencent a devenir difficilement reconnaissables.

TAB. 4.2: Exemples d'images reconstruites pour des divers taux de perte

Loss rate	20.27%	40.18%	62.1%	83.03%
Sans mélange	 PSNR = 32.05dB	 PSNR = 30.41dB	 PSNR = 21.65dB	 PSNR = 18.81dB
AT*	 PSNR = 35.06dB	 PSNR = 31.74dB	 PSNR = 28.66dB	 PSNR = 23.66dB



(a) Distortion observée sur l'image « Lenna »



(b) Distortion observée sur l'image « Corridor »

FIG. 4.10: Qualité d'image observée en fonction du taux de pertes de paquets pour les scénarios avec et sans mélange de l'image.

4.5 Évaluation du coût d'énergie des AT*

Nous avons impanté sur nos nœuds Cyclops/Mica2 l'algorithme AT* pour évaluer son coût d'énergie et son coût mémoire. La matrice $A = \begin{pmatrix} 1 & 1 \\ k & k+1 \end{pmatrix}^n$, avec $k = 1$ et $n = 8$, avait été pre-calculée pour limiter le nombre de calculs à faire pour obtenir les coordonnées projetées (i',j'). En termes de mémoire, l'AT* rajoute seulement 148 octets de ROM et 35 octets de RAM par rapport à l'application de référence.Cela nous a permis de vérifier que les AT* peuvent effectivement être implantés dans des systèmes limités en ressources. En ce qui concerne la consommation d'énergie et le temps d'exécution, ces indicateurs

n'augmentent pas énormément. La consommation d'énergie et le temps d'exécution pendant un cycle de travail (capture, mélange et transmission) sont respectivement de 2374mJ et 30.2 secondes. C'est seulement 67mJ et 0.65 secondes de plus que l'application de référence. Le coût de l'AT* est donc d'environ 4μJ and 40μs par pixel.

4.6 Évaluation de la fonction d'entrelacement de pixels

Comme nous l'avons dit précédemment, le mélange de l'image à la source vise à ce que les pixels qui sont voisins dans l'image soient transmis dans des paquets séparés qui soient le plus possible distants les uns des autres.

Le choix des valeurs de k et n aura nécessairement un impact sur la distance des paquets qui embarque des pixels voisins. Nous avions adopté arbitrairement $k = 1$ et $n = 8$. Ces valeurs ont montré de bonnes performances à l'usage. Mais existe-il des valeurs optimales pour k et n ? Cette section étudie la question en faisant une évaluation de AT* en fonction de ces paramètres.

Pour évaluer la fonction d'entrelacement de pixels, nous avons adopté comme fonction objective la distance moyenne entre les pixels voisins, notée $\overline{D_v}$ et exprimée en nombre de paquets, que nous définissons comme suit : Considérons la matrice initiale I $(H \times W)$, où $I = \{x_{i,j}\}$ avec $0 \leq i < H$ et $0 \leq j < W$. Après l'application d'une méthode de mélange, chaque pixel $x_{i,j}$ est projeté sur une nouvelle position (i', j'), formant la matrice $I' = \{x_{i',j'}\}$ avec $0 \leq i' < H$ et $0 \leq j' < W$. Maintenant, considérons le vecteur de positions $N[\imath]$, avec $0 \leq \imath < H - 1 \times W - 1$ correspondant a l'index de la position séquentielle de chaque pixel $x_{i,j}$ d'une image organisée ligne par ligne, c'est-à-dire, $\imath = W * i + j$. Dans chaque position du vecteur N, nous stockons la ligne et la colonne correspondant à l'application de la méthode de mélange sur le pixel $x_{i,j}$, i' et j', dénotés respectivement $N[\imath]_{row}$ et $N[\imath]_{col}$. Nous calculons la distance en paquets entre deux pixels x_{i_1,j_1} et x_{i_2,j_2}, comme :

$$d(x_{i_1,j_1}, x_{i_2,j_2}) = \left| \left\lfloor \frac{W * i_1 + j_1}{PPP} \right\rfloor - \left\lfloor \frac{W * i_2 + j_2}{PPP} \right\rfloor \right| \tag{4.2}$$

ou PPP est le nombre de pixels qui peuvent être embarqués par paquet. Avec celà, nous pouvons définir la distance moyenne des pixels voisins du pixel $x_{i,j}$ comme :

$$\overline{\delta v}_{i,j} = \frac{1}{NPV_{i,j}} \cdot \sum_{r=\max(i-1,0)}^{\min(i+1,W-1)} \sum_{s=\max(j-1,0)}^{\min(j+1,H-1)} d(x_{N[W*i+j]_{row},N[W*i+j]_{col}}, x_{N[W*r+s]_{row},N[W*r+s]_{col}}) \tag{4.3}$$

ou $NPV_{i,j}$ est le nombre de pixels voisins au pixel $x_{i,j}$, qui peut être calculé comme :

$$NPV_{i,j} = (|\max(i-1,0) - \min(i+1,H-1)| + 1).(|\max(j-1,0) - \min(j+1,W-1)| + 1) - 1 \tag{4.4}$$

Finalement, la distance moyenne ramenée sur l'echelle de tous les pixels de l'image est calculée comme :

$$\overline{D_v} = \frac{1}{H \times W} \cdot \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} \overline{\delta v}_{i,j} \tag{4.5}$$

Les résultats de l'évaluation des automorphismes toriques, en faisant varier la clé diffusion n et pour $k = 1$, sont présentés sur le graphique de la figure 4.11.

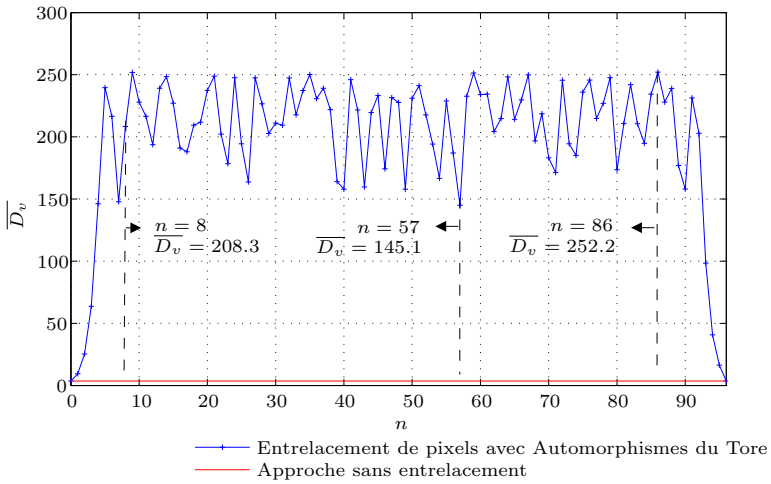


FIG. 4.11: Evaluation de la distance moyenne entre un pixel et ces voisins dans une image mélangée par AT*.

La figure montre que la distance moyenne entre les pixels voisins $\overline{D_v}$ augmente rapidement lorsque on augmente n de 0 à ~ 4 et diminue aussi rapidement entre $n = 92$ à 96, valeur qui correspond à l'inverse de la fonction, donc à l'évaluation de la distance moyenne $\overline{D_v}$ sur une image non mélangée (dans ces cas là, $\overline{D_v}$ est approximativement égale à 3.56). Entre $n = 4$ et $n = 92$ la distance moyenne oscille entre les valeurs de $\overline{D_v} = 145$ et $\overline{D_v} = 252$ approximativement. La valeur maximale $\overline{D_v} = 252$ est obtenue pour $n = 86$. La valeur minimale $\overline{D_v} = 145$ est obtenue par $n = 57$. Nous avons donc comparé les qualités d'image observées sous différents taux de perte de paquets pour $n = 8$ (notre choix arbitraire), $n = 86$ (cas le plus favorable), et $n = 57$ (cas le plus défavorable). Les résultats sont visualisés figure 4.12.

Nous constatons que les fluctuations observées pour la valeur de $\overline{D_v}$ quand n varie entre 4 et 92 ne sont pas significatives dans les résultats de PSNR, quelque soit le taux de perte du réseau. Dans la figure 4.12 en effet, nous pouvons voir que les valeurs du PSNR pour l'image « Corridor » sont très proches, difficile à distinguer. Comme la variation de n ne rapporte pas de différences significatives dans le PSNR, alors il y a une certaine liberté dans le choix de la clé de diffusion n. Nous conservons la valeur de $n = 8$ car elle permet de manipuler des valeurs entières sur 16 bits, contrairement à $n = 86$ qui sous-tend un coût d'énergie plus élevé pour les calculs.

4.7 Conclusion

Dans ce chapitre, nous avons discuté l'effet des pertes de paquets sur la qualité des images reconstituées au nœud puits. Les pertes peuvent être élevées dans les réseaux de capteurs, de plusieurs pourcents à plusieurs dizaines de pourcents. La correction des pertes de paquets au niveau du protocole de communication aura un coût d'énergie proportionel au taux de pertes constaté. Nous avons donc étudié s'il est envisageable de transmettre des images avec un protocole non-fiable. Mais les expériences ont montré que la qualité des images reçues se dégrade très rapidement avec l'augmentation des pertes de paquets. Pour améliorer la résistance des images aux pertes de paquets, nous avons étudié l'application d'un algorithme d'entrelacement de pixels de faible complexité, basé sur les automorphismes du Tore.

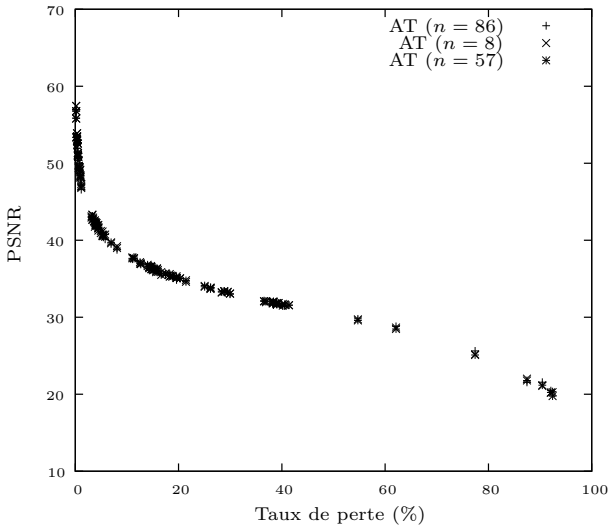


FIG. 4.12: Qualités observées sur l'image « Corridor » en fonction des taux de perte de paquets avec différents clés de diffusion n.

Cette opération de mélange de pixels de l'image est effectuée à la source, avant de construire les paquets de données. Cela coûte nécessairement de l'énergie pour la source, mais les résultats de notre évaluation de performance montrent que la consommation d'énergie est raisonnable, environ 4μJ par pixel, et que le temps d'exécution est correct, environ 40μs par pixel.

Les résultats montrent aussi une très grande amélioration de la résistance de l'image aux pertes de paquets. Avec le mélange de l'image à la source, la qualité de l'image finale se dégrade beaucoup plus lentement avec l'augmentation des pertes de paquets. Les images sont visuellement acceptables pour des pourcentages de pertes qui dépassent les 50%. De tels résultats nous amènent à conclure qu'il est préférable, dans la recherche du meilleur compromis énergie-distortion, d'utiliser un protocole de communication non fiable pour transmettre des images plutôt qu'un protocole semi-fiable ou un protocole totallement fiable.

Chapitre 5

Algorithme de compression d'images de faible complexité et résistant aux pertes de paquets

Nous avons présenté dans les chapitres précédents des solutions pour réduire la quantité d'énergie consommée dans le réseau pour *transmettre une image non compressée* jusqu'au puits. Ces solutions étaient toutes fondées sur l'acceptation *a priori* de perdre des paquets, et donc de la dégradation potentielle de la qualité des images reçues. Ce relâchement de la contrainte de fiabilité au niveau du protocole de communication, qu'il soit partiel (cf. chapitre 3) ou total (cf. chapitre 4), tend en effet à diminuer le nombre de paquets et d'acquittements échangés dans le réseau lorsque les conditions sont défavorables, ce qui entraîne mécaniquement des économies d'énergie dans les nœuds de transit (même si les économies d'énergie sont inégalement réparties, puisque sur l'horizon d'un chemin de bout en bout, plus un nœud est éloigné de la source, et plus il sera favorisé). Réduire la dépense d'énergie dans les nœuds de transit peut être bon pour le réseau dans son ensemble, mais en fin de compte, cela est insuffisant pour augmenter l'espérance de vie du réseau si les nœuds source n'économisent pas, eux-mêmes, leur énergie. Nous avons déjà discuté de l'impact qu'avait l'adoption d'un protocole semi-fiable ou non fiable sur la dépense d'énergie de la source en comparaison avec un protocole fiable. Les limites de ces protocoles ont été clairement évaluées. Maintenant, la compression de l'image à la source est la voie naturelle pour dépasser ces limites puisque cette opération va permettre au nœud source, et par incidence aux nœuds de transit, de traiter (beaucoup) moins de paquets. Mais compresser une image engage des calculs et des accès mémoire, donc une dépense d'énergie. Pour que cela soit rentable pour la source, il faut évidemment que le coût d'énergie de la compression de l'image soit inférieur à l'économie d'énergie qu'elle amène sur la transmission des paquets. Et c'est là que le bât blesse ! En effet, les méthodes de compression bien connues comme SPIHT, JPEG et JPEG2000, ont beau fournir d'excellentes performances en termes de rapport débit-distorsion, elles ne sont pas éligibles dans le contexte des réseaux de capteurs. Deux raisons à cela :

– D'abord, leur complexité est telle que cela coûte plus d'énergie de compresser l'image que de la transmettre sans compression. Cette assertion est validée par les travaux de (Ferrigno *et al.*, 2005)

et de (Mammeri *et al.*, 2008), comme nous l'avons précisé dans l'état de l'art (cf. chapitre 2, section compression). Elle est aussi confirmée par nos propres résultats expérimentaux pour la méthode JPEG implanté sur un mote MICA2 et un capteur d'image Cyclops (cf. la section 5.3 de ce chapitre).

– Ensuite, ces méthodes résistent très mal aux erreurs (perte de paquets) apportées par le canal de transmission (l'absence de quelques bits d'information au récepteur suffit pour entraîner une dégradation brutale de la qualité de l'image décodée, voire même l'impossibilité de décoder l'image). Elles vont donc amener à renforcer la contrainte de fiabilité au niveau du protocole de communication pour se protéger des pertes de paquets (alors que nous plaidons pour faire le contraire !), et cela a un coût d'énergie. Précisons que dans une certaine mesure, cette contrepartie est tout à fait normale puisque compresser une image consiste à éliminer les redondances spatiales et/ou fréquentielles qu'elle contient. Une fois compressée, l'image perd donc sa faculté d'être « résistante » aux erreurs, le récepteur ne pouvant plus compter sur les redondances originelles pour appliquer avec succès une méthode de dissimulation d'erreurs. Mais il va de soit qu'une méthode de compression qui contournerait le problème serait beaucoup plus attractive dans notre contexte.

Dans ce chapitre, nous proposons un algorithme de compression d'image original qui, d'une part, est très peu calculatoire donc peu gourmand en énergie, et qui, d'autre part, assure que l'image compressée a un maximum de résistance aux erreurs de transmission. Notre objectif est clairement de transmettre des images compressées en se basant sur un protocole de communication non fiable, donc en acceptant, sans exclusive, toutes les pertes de paquets. Evidemment, en procédant comme cela, nous ne pourrons pas garantir le niveau de qualité des images reconstituées au récepteur. Et il faudra que la qualité des images reste « acceptable » même pour des taux de pertes élevés. Pour que cela soit possible, nous nous sommes orientés naturellement vers un algorithme de compression par blocs étant donné que :

– le fait de découper l'image originale en blocs de pixels et de compresser un à un les blocs de manière indépendante assure qu'au décodeur, tout bloc correctement reçu est décodable à tous les coups ;
– les blocs de l'image peuvent être mélangés sans que cela interfère sur l'algorithme de compression, sachant qu'un tel mélange est utile pour atténuer les effets des pertes de paquets sur la qualité des images reconstituées.

La taille des blocs aura une importance cruciale sur les performances attendues : Plus les blocs seront petits, et plus il sera facile de dissimuler leur perte au décodeur. Mais en retour, plus les taux de compression seront faibles. C'est donc affaire de compromis. Au vu des taux de pertes de paquets couramment rencontrés dans les réseaux de capteurs sans fil, et en accord avec notre volonté de ne pas les corriger du tout, nous privilégierons la « résistance aux pertes » au détriment du « taux de compression », et en conséquence, nous choisirons des blocs de toute petite taille, 2×2 pixels. Cela est d'autant plus justifié qu'en pratique, la majorité des applications de réseau de capteurs d'images manipuleront des images de petite taille, surement plus proche de 32×32 ou 64×64 pixels que de 512×512 pixels.

Le reste du chapitre de découpé en trois sections. La première section présente les principes techniques de l'algorithme de compression que nous proposons, et que nous avons appelé ICES (Image Compression for Energy-constrained Sensors). ICES opère sur des blocs de 2×2 pixels et provoque une suppression de pixel astucieuse pour réduire le volume d'information à encoder. La section suivante détaille les résultats de l'évaluation de performances, d'abord en termes de rapport débit-distorsion, ensuite en termes de consommation de ressources (énergie, temps d'exécution et quantité de mémoire). Les performances de ICES sont comparées avec des algorithmes de complexité similaire, ainsi qu'avec l'algorithme JPEG. La

dernière section présente le couplage de ICES avec TA*, la technique de mélange de (blocs de) pixels basée sur les automorphismes du Tore qui a fait l'objet du chapitre 4.

5.1 Principes techniques de ICES

Il existe plusieurs méthodes pour la compression d'images sans perte (*lossless*) et avec perte (*lossy*). Les méthodes sans perte permettent de reconstituer l'image originale à partir des données compressées, tandis que les méthodes avec perte permettent seulement de reconstituer une version approchée de l'image originale. Les méthodes avec perte permettent d'atteindre des taux de compression plus élevés que les méthodes sans perte, mais elles sont aussi plus complexes, c'est-à-dire, qu'elles demandent plus de calculs, et donc coûtent plus cher en énergie.

La plupart des méthodes de compression avec pertes sont basées, soit sur la transformée en cosinus discrète (DCT), soit sur la transformée en ondelettes discrète (DWT). La DWT s'applique sur l'image entière ou sur des très gros blocs d'image. La DCT peut travailler sur des petits blocs (il est usuel de prendre des blocs 8×8 comme le fait JPEG) mais elle perd de son intérêt quand on opére sur des blocs aussi petits que 2×2 pixels.

ICES tourne le dos aux approches de compression classiques, c'est une méthode de compression avec pertes qui est spécifiquement conçue pour les systèmes embarqués limités en ressources de calcul, de mémoire et d'énergie. ICES exécute juste quelques instructions élémentaires sur des valeurs entières (décalages, additions, soustractions, comparaisons ...). Cette méthode est basée sur la supression de pixel, un par bloc. Elle fournit donc typiquement un taux de compression de 4 : 3, mais ce taux de compression peut être augmenté si on réalise une quantification scalaire des pixels d'abord. La chaîne de compression de ICES est schématisée sur la figure 5.1.

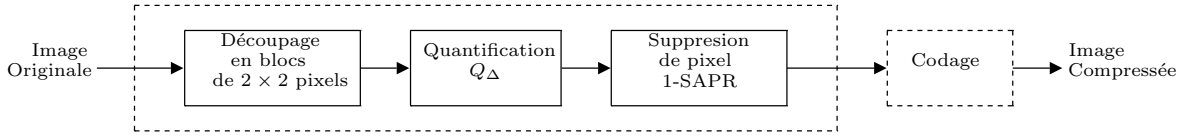


FIG. 5.1: Schéma de compression ICES.

Remarquons qu'il est possible de rajouter une étape de codage pour utiliser des codes de longueur variable, en vue d'atteindre des taux de compression plus élevés. Mais nous ne travaillons pas cette étape dans ce mémoire, et cela sera considéré comme une continuation possible de nos travaux.

Nous avons adopté la notation suivante : L'image originale, notée I, est constituée de $W \times H$ pixels, où W est la largeur de l'image et H sa hauteur (W et H étant de préférence des entiers pairs). Considérons un plan quelconque dans l'espace de couleurs, et considérons que dans ce plan, les valeurs des intensités des pixels sont représentées sur m bits, et donc qu'elles sont comprises dans l'intervalle $[0, M]$, où $M = 2^m - 1$. Notre algorithme divise initialement le plan de couleurs de l'image localisé en blocs non-chevauchants de 2×2 pixels. Notons $B_{p,q}$ localisé à la $p^{\text{ème}}$ ligne et $q^{\text{ème}}$ colonne, avec $0 \leqslant p < \frac{H}{2}$ et $0 \leqslant q < \frac{W}{2}$, le bloc $B_{0,0}$ étant positionné sur le coin supérieur gauche du plan de couleurs. Les quatre pixels contenus dans le bloc $B_{p,q}$ seront notés $x_{0|p,q}$, $x_{1|p,q}$, $x_{2|p,q}$ et $x_{3|p,q}$, comme montré sur la figure 5.2. $B_{p,q}$ est representé sur $4 \times m$ bits. Notons respectivement $B'_{p,q}$ la version compressée de $B_{p,q}$ et $\widehat{B}_{p,q}$ la version decompressée de $B'_{p,q}$.

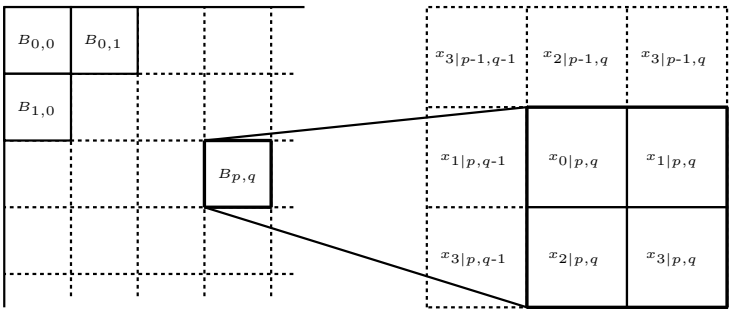


FIG. 5.2: Représentation du bloc $B_{p,q}$ de 2×2 pixels.

5.1.1 Suppression de pixels

La suppression de pixel (PR, *pixel removal*) est certainement la méthode la plus triviale pour réduire le nombre de bits nécessaires pour représenter un bloc de l'image. Cela consiste à supprimer un certain nombre de pixels du bloc, puis, du côté du décodeur, exploiter les corrélations spatiales qui existent entre les pixels restants pour estimer l'intensité des pixels manquants. Comme nous travaillons sur des tous petits blocs, nous avons choisi raisonnablement de supprimer un et un seul pixel parmi les quatre que contient le bloc de 2×2 pixels (1-PR). Par conséquent, le bloc résultant sera répresenté sur $3 \times m$ bits, ce qui donne typiquement un taux de compression de 4 : 3. Du côté du décodeur, le pixel supprimé peut être estimé à partir des trois autres en appliquant une méthode de dissimulation d'erreurs et exclusivement sur ces trois là si on veut que chaque bloc puisse être décodé indépendamment des autres. Remarquons qu'il n'y aura pas de problème d'alignement des pixels au décodeur si le pixel supprimé est toujours le même d'un bloc à l'autre (par exemple $x_{0|p,q}$).

Même si 1-PR est une méthode de compression très rudimentaire, voire grossière, et qu'on ne doit pas s'attendre à obtenir des rapports débit/distorsion du même ordre de grandeur que les algorithmes classiques, notre intérêt pour cette méthode vient du fait qu'elle n'implique aucun calcul particulier, et donc que son coût d'énergie est nul. Chaque pixel est simplement lu est écrit une fois pour remplir les paquets, comme pour la transmission d'une image non compressée. Mais l'inconvénient, quand le pixel supprimé est choisi de manière arbitraire, c'est qu'on ne contrôle pas du tout le niveau de distorsion qui sera observé entre le bloc original $B_{p,q}$ et le bloc $\widehat{B}_{p,q}$ restitué. Cette distorsion peut varier de manière significative d'un bloc à l'autre sur toute l'image de manière aléatoire. C'est pourquoi nous proposons un algorithme de suppression de pixels auto-adaptatif (nommé 1-SAPR) pour minimiser la distorsion.

5.1.2 1-SAPR

Notons $D_{k|p,q}$ la distorsion entre $B_{p,q}$ et $\widehat{B}_{p,q}$ quand le pixel $x_{k|i,j}$ est supprimé de $B_{p,q}$. Plus la distorsion est grande, moins bonne sera la qualité de l'image reconstruite. $D_{k|p,q}$ est calculé comme :

$$D_{k|p,q} = \frac{1}{4}\left(\widetilde{x}_{k|p,q} - x_{k|p,q}\right)^2 \tag{5.1}$$

où $\widetilde{x}_{k|p,q}$ est la valeur estimée de l'intensité manquant du pixel. La valeur de $\widetilde{x}_{k|p,q}$, et donc de $D_{k|p,q}$, dépend de la méthode de dissimulation d'erreurs adoptée au récepteur. Si la méthode de dissimulation d'erreurs est connue côté codeur, il est possible de trouver le pixel du bloc qui, une fois supprimé, induit la plus petite distorsion pour le bloc décompressé. Ceci correspond a une méthode de suppression de pixel

qui serait réalisée de manière auto-adaptative. Évidemment, cette approche conduira à une diminution de la distorsion globale de l'image, mais le problème est que le pixel supprimé ne sera pas toujours à la même position d'un bloc à l'autre. Comme l'emplacement du pixel manquant doit être connu du décodeur pour qu'il puisse travailler correctement, il est nécessaire d'embarquer cette information dans $B'_{p,q}$, la version compressée de $B_{p,q}$.

Comme il y a quatre positions possibles, nous avons besoin théoriquement d'au moins 2 bits pour représenter cette information. Mais plutôt que de représenter le bloc codé $B'_{p,q}$ sur $(3 \times m) + 2$ bits, nous allons insérer cette information dans les bits les moins significatifs (le LSB, ou *Least Significant Bit*) des trois pixels restants, afin de préserver un taux de compression de 4 : 3. Nous allons procéder comme suit : Nous définissons ω=[ω_a ω_b ω_c] le mot composé par les trois bits de poids faible des trois pixels restants dans le bloc dans l'ordre de leur alignement, et $k \in \{0, 1, 2, 3\}$ l'information à insérer sur ω. Nous avons défini la transformée $\tau : \omega \longrightarrow \omega' = \tau(\omega, k)$ pour insérer k dans ω de manière à ce que dans le pire des cas, seulement un bit de ω sera changé. Le tableau 5.1 donne le détail de cette transformée.

TAB. 5.1: Résultats de l'insertion de k sur ω, en utilisant la transformée τ. Les valeurs ont été écrites en notation binaire (la conversion décimale est donnée entre parenthèse).

ω	$\omega'=\tau(\omega,0)$	$\omega'=\tau(\omega,1)$	$\omega'=\tau(\omega,2)$	$\omega'=\tau(\omega,3)$
000 (0)	000 (0)	001 (1)	010 (2)	100 (4)
001 (1)	000 (0)	001 (1)	101 (5)	011 (3)
010 (2)	000 (0)	110 (6)	010 (2)	011 (3)
011 (3)	111 (7)	001 (1)	010 (2)	011 (3)
100 (4)	000 (0)	110 (6)	101 (5)	100 (4)
101 (5)	111 (7)	001 (1)	101 (5)	100 (4)
110 (6)	111 (7)	110 (6)	010 (2)	100 (4)
111 (7)	111 (7)	110 (6)	101 (5)	011 (3)

L'algorithme 2 donne la séquence des opérations réalisées pour coder le bloc $B_{p,q}$.

Algorithme 2 Codage auto-adaptative de blocs pour la compression avec ICES

1: $k = 0$
2: **for** $l = 0$ to 3 **do**
3: Calculer $\widetilde{x}_{l|p,q}$
4: Calculer $D_{l|p,q}$ en supposant que le pixel a supprimer est $x_{l|p,q}$
(Evidemment, l'alteration des LSBs doit être considerée dans le calcul de la distorsion des blocs)
5: **if** $D_{l|p,q} > D_{k|p,q}$ **then**
6: $k = l$
7: **end if**
8: **end for**
9: Supprimer le pixel $x_{k|p,q}$
10: Trouver $\omega' = \tau(\omega, k)$
11: Insérer ω' sur les LSBs des trois pixels retenus

Pour faire l'estimation du pixel manquant, plusieurs méthodes de dissimulation d'erreurs sont possibles, et par conséquent, il existe autant de variantes de ICES qu'il existe de méthodes de dissimulation d'erreurs. Dans la cadre de cette thèse, nous nous sommes limités à l'évaluation et la comparaison des performances des deux méthodes suivantes :

1. Estimation par la moyenne des pixels voisins : Cette méthode consiste à calculer la moyenne des trois pixels retenus, c'est-à-dire, $\widetilde{x}_{0|p,q} = \frac{1}{3}\left(x_{1|p,q} + x_{2|p,q} + x_{3|p,q}\right)$, $\widetilde{x}_{1|p,q} = \frac{1}{3}\left(x_{0|p,q} + x_{2|p,q} + x_{3|p,q}\right)$, $\widetilde{x}_{2|p,q} = \frac{1}{3}\left(x_{0|p,q} + x_{1|p,q} + x_{3|p,q}\right)$ et $\widetilde{x}_{3|p,q} = \frac{1}{3}\left(x_{0|p,q} + x_{1|p,q} + x_{2|p,q}\right)$.
2. Estimation par duplication d'un des pixels voisins : Cette méthode consiste à copier simplement l'un des trois pixels retenus à la place du pixel manquant, par exemple, $\widetilde{x}_{0|p,q} = x_{1|p,q}$, $\widetilde{x}_{1|p,q} = x_{3|p,q}$, $\widetilde{x}_{2|p,q} = x_{0|p,q}$, et $\widetilde{x}_{3|p,q} = x_{2|p,q}$.

Les résultats de l'évaluation de performance, donnés plus loin dans ce chapitre, ont montré que dans presque tous les cas, c'est la méthode d'estimation par duplication de pixels qui est la meilleure.

5.1.3 Quantification scalaire uniforme

Comme 1-SAPR supprime un et un seul pixel par bloc de 2×2 pixels, la méthode de compression proposée fournit typiquement un taux de compression de 4:3. C'est évidemment très peu. Toutefois, des taux de compression plus importants peuvent être obtenus en appliquant en amont un quantificateur scalaire. Ici, la quantification peut être vue comme un arrondi des valeurs d'entrée (les intensités des pixels à l'origine répresentés par m bits) pour réduire le nombre de valeurs possibles en sortie à un ensemble plus petit tel que m' bits ($m' < m$) soient suffisant pour les représenter. Nous avons adopté un simple quantificateur scalaire uniforme, Q_Δ, qui définit les niveaux de quantification par :

$$Q_\Delta(x) = \left\lfloor \frac{x}{2^\Delta} \right\rfloor \tag{5.2}$$

où 2^Δ est le pas de quantification. Ainsi, nous avons : $m' = m - \Delta$. Un tel quantificateur est approprié pour les capteurs d'image ayant des contraintes d'énergie car il requiert une opération de décalage binaire (un *shift*) par pixel seulement.

5.1.4 Paquétisation

Le schéma de transmission adopté est séquentiel, d'une manière analogue à la transmission des images non compressées (et non mélangées), à la différence qu'au lieu d'une transmission ligne par ligne de pixels, la paquétisation est effectuée par lignes de blocs compressés. Chaque bloc compressé est stocké entièrement dans un seul paquet, c'est-à-dire, que les données correspondant à un même bloc compressé seront toujours transmises dans un même paquet et non pas dispersées sur plusieurs paquets. Cela assure, que tous les blocs embarqués dans un même paquet sont immédiatement décodables au récepteur et simplifie le processus de décodage.

Avec des calculs simples on peut calculer la quantité de blocs compressés à envoyer, qui est égale à $QBC = \frac{H \times W}{BlockHeight \times BlockWidth}$, où $BlockHeight$ et $BlockWidth$ représentent respectivement la hauteur et la largeur des blocs. Alors, et sans perte de généralité, la quantité de paquets à envoyér est calculée comme :

$$QP = \left\lceil \frac{QBC}{\left\lfloor \frac{8 \times DataSize}{TC.(BlockHeight \times BlockWidth)} \right\rfloor} \right\rceil \tag{5.3}$$

ou $DataSize$ est la quantité d'octets disponibles pour l'enregistrement de données dans un paquet et TC est le taux de compression en bpp. Par exemple, en considérant un format de paquet qui réserve $DataSize = 27$ octets pour la transmission de blocs compressés, pour transmettre une image de 128×128 pixels compressée avec ICES à 3.75 bpp, on transmet une quantité de $QBC = \frac{128 \times 128}{2 \times 2} = 4096$ blocs, et on transmet une quantité de $QP = \left\lceil \frac{4096}{\left\lfloor \frac{8 \times 27}{3.75 \times (2 \times 2)} \right\rfloor} \right\rceil = 293$ paquets.

5.2 Évaluation de la qualité des images

Pour évaluer les performances de notre algorithme de compression, nous avons sélectioné un ensemble d'images test monochromes, de taille 512×512 pixels, encodées sur 8 bpp : les célèbres « Lenna », « Baboon », « Barbara », « Goldhill » et « Peppers ». Nous avons aussi sélectionné quelques images que nous avons capturé avec un capteur Cyclops, en particulier l'image monochrome « Corridor », encodée sur 8bpp, et disponible en trois tailles, 128×128 pixels, 64×64 pixels, et 32×32 pixels.

Pour les évaluations portant sur la qualité d'image, nous avons programmé notre algorithme ICES et les algorithmes avec lesquels nous le comparons en utilisant le langage `C++` et le compilateur `GNU g++`. Les programmes ont été exécutés sur une station de travail de type PC ayant le système d'exploitation Linux. Le PSNR (*Peak Signal to Noise Ratio*) était utilisé comme métrique objective de la qualité d'image (ou plus précisément de la distorsion).

5.2.1 Comparaison de deux variantes de ICES

Une première série de tests a été réalisée pour comparer les performances de ICES selon qu'il utilise l'une ou l'autre des deux méthodes de dissimulation d'erreurs présentées en section 5.1.2 pour estimer le pixel manquant : estimation par moyenne des pixels voisins ou estimation par duplication d'un des pixels voisins. Nous distinguerons ces deux variants de ICES en les appelant respectivement ICESv1 et ICESv2.

Le tableau 5.2 montre les valeurs du PSNR obtenues avec ICESv1 et ICESv2 pour chacune des images de test selectionnées.

Tab. 5.2: Comparaison des variantes de ICES (3.75bpp) en termes de PSNR

Image	(Résolution)	ICESv1	ICESv2
Lenna	(512×512)	33.03dB	33.37dB
Barbara	(512×512)	30.80dB	32.33dB
Goldhill	(512×512)	32.47dB	33.16dB
Peppers	(512×512)	32.02dB	33.23dB
Corridor	(128×128)	32.70dB	33.44dB

Les résultats montrent sans ambiguité que ICESv2 est le meilleur puisqu'il surpasse ICESv1 avec presque toutes les images test. En conséquence, nous conservons dans la suite de nos évaluations cette variante seulement, que nos appelerons tout simplement ICES.

5.2.2 Comparaison de ICES avec des algorithmes de complexité similaire

Dans une deuxième série de tests, nous avons comparé ICES avec les algorithmes de compression d'image comparable (c'est-à-dire des algorithmes dont la complexité est du même ordre de grandeur), y compris la *suppression de pixels* (PR, par *Pixel Removal*) et la *Quantification Uniforme* (UQ, *Uniform Quantization*). Rappelons que PR et ICES fournissent typiquement un taux de compression de 4:3, soit un débit binaire[1] de 6bpp pour les images codées originalement à 8bpp, mais ils peuvent atteindre des taux de compression plus grands par l'application d'un quantificateur scalaire (par exemple, Q_Δ comme décrit dans la section 5.1.3). Ainsi, PR et ICES peuvent atteindre des taux de compression de 5.25bpp avec Q_1, de 4.5bpp avec Q_2 et de 3.75bpp avec Q_3. Dans la figure 5.3, nous montrons la variation du PSNR en fonction du taux de compression, pour les images « Lenna », « Baboon », et « Corridor ».

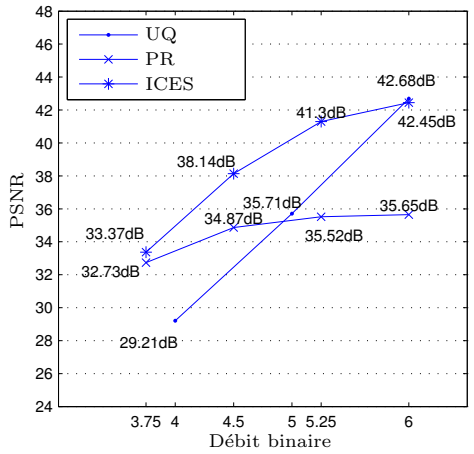
Les résultats montrent que, dans tous les cas, ICES fournit des rapports débit/distorsion plus élevés que PR. Ceci est dû au fait que PR choisit arbitrairement d'enlever les pixels du bloc, sans aucun égard sur la distorsion que cela peut porter sur la reconstruction, alors que ICES sélectionne toujours le moins gênant.

La diminution de la qualité de l'image devient plus importante avec des images de haut détails (voir, par exemple, le PSNR pour Baboon dans la figure 5.3(b)). Dans tous les tests, UQ compressant à un taux de 6bpp, offre une grande qualité dans les images obtenues, beaucoup plus grande que notre algorithme. En effet, la perte seulement des 2 bits les moins significatifs n'engendre pas de répercussions importantes dans la qualité de l'image, puisque les valeurs des pixels résultants sont extrêmement proches. Cette perte porte au maximum une distance de 3 points de détail à l'égard du pixel original. Dans une échelle de valeurs de 0 à 255 (échelle normalement utilisée pour coder une image monochrome à 8bpp), cette différence n'est pas significative.

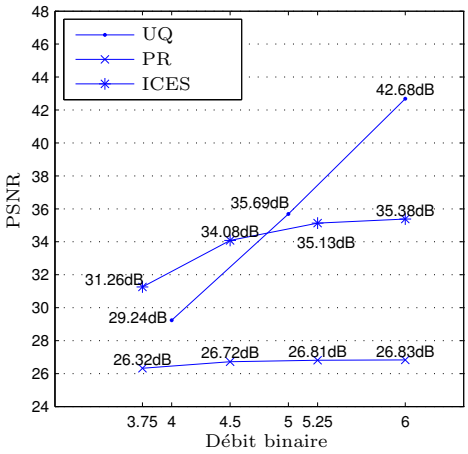
ICES devient plus intéressant avec des taux de compression plus élevé, lorsque l'on combine avec le processus de quantification.

D'après les résultats, on observe la continue et non-récupérable dégradation qui se produit lorsque l'on applique UQ, chaque fois que nous avons gagné un bit de compression (c'est bien sûr prévisible pour le principe même de la mesure de distortion par PSNR). Nous pouvons également observer les différents effets que l'application de PR à différents débits cause sur la qualité de l'image reconstruite. D'une part, dans des images à forte corrélation spatiale entre les pixels, c'est-à-dire, des images de peu de détails avec une grande ou de nombreux zones qui contiennent peu de variation dans les valeurs de leurs pixels, et surtout dans les grandes images, PR donne des résultats très acceptables. Par exemple, dans les images telles que Lenna ou Peppers, qui ont de grands espaces de la même tonalité, PR donne de bons résultats, ce qui est plus notable avec de forts degrés de compression. D'autre part, dans les images très détaillées (Baboon, par exemple), PR montre les plus mauvais résultats. Ces résultats sont généralisés

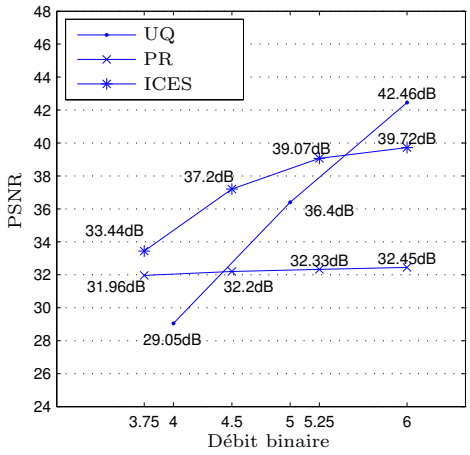
[1] Dans la communauté compression d'images, le terme débit binaire est utilisé pour désigner la répresentation en bits par pixel (bpp) d'une image compressée, et non pas la quantité de données transmises par unité du temps comme dans la communauté des réseaux.



(a) Image « Lenna » (512 × 512 pixels).



(b) Image « Baboon » (512 × 512 pixels).



(c) Image « Corridor » (128 × 128 pixels).

FIG. 5.3: Débit/distorsion d'ICES vs. des algorithmes de complexité similaire.

pour la plupart des images testées dans diverses résolutions. ICES donne de résultats très acceptables en comparaison aux autres méthodes simples étudiées dans presque tous les tests.

La figure 5.4 montre l'image Lenna (512 × 512), avant et après compression avec ICES à 3.75bpp. On peut remarquer une certaine détérioration de la qualité de l'image (PSNR = 33.37dB), ce qui était prévu en raison de la simplicité de la méthode. Cependant, nous pensons que cette dégradation est largement acceptable pour la plupart des applications visant des réseaux de capteurs d'image, et certainement justifiable.

Pour illustrer encore plus les effets des différentes méthodes de compression dans le champ d'application de ce papier, nous avons extrait et mis à l'échelle une partie de diverses images reconstruites que nous avons obtenu. Dans la figure 5.4(d), on peut clairement appercevoir la perte de la qualité de l'image visualisée en raison de la réduction de la quantité de tonalités causés par l'application de UQ avec un pas de quantification élevé ($\Delta = 4$). La perte de qualité est également perceptible pour la deuxième méthode (PR à 3.75bpp), où l'on peut observer un effet pixelation principalement dans les bords de haut contraste (voir, par exemple, dans la figure 5.4(e), le sourcil et le bord supérieur de l'œil).

(a) (b)

(c) (d) (e) (f)

FIG. 5.4: Première rangée : visualisation de (a) l'image de test originale « Lenna » et (b) de la version reconstruite compressée par ICES à 3.75bpp (PSNR = 33.37dB).
Deuxième rangée : comparaison de (c) « l'œil de Lenna » mis a l'échelle et reconstruit après sa compression par (d) UQ à 4bpp (PSNR = 29.21dB), (e) PR à 3.75bpp (PSNR = 32.73dB) et (f) ICES à 3.75bpp (PSNR = 33.37dB).

Nous avons également comparé les algorithmes de compression sur les images capturées avec notre nœud caméra. Dans la figure 5.5, nous illustrons la dégradation de l'image « Corridor », capturée dans une résolution de 128×128 pixels (image originale mis a l'échelle dans la figure 5.5(a)). De la même façon que sur l'image Lenna, nous pouvons voir la baisse de qualité due à la réduction de la gamme de tonalités causée par l'algorithme UQ. La perte de qualité devient de plus en plus perceptible lorsque on compresse avec PR. En effet, dans la figure 5.5(c), on peut voir clairement la mauvaise qualité de l'image obtenue après l'application de PR à 3.75bpp.

(a) (b) (c) (d)

FIG. 5.5: Visualisation de partie mise l'échelle de (a) l'image originale « Corridor » (8bpp) et des images reconstruites compressées par (b) UQ à 4bpp (PSNR = 29.05dB), (c) PR à 3.75bpp (PSNR = 31.96dB) et (d) par ICES à 3.75bpp (PSNR = 33.44dB).

5.2.3 Comparaison de ICES avec JPEG

Enfin, dans une troisième série de test, nous comparons également ICES à l'algorithme de la norme JPEG. Nous comparons également avec une modification du JPEG en considérant une compression de blocs indépendants, c'est-à-dire, les coefficients DC ne sont pas corrélés entre eux comme dans la norme JPEG. Nous appellerons cette approche, JPEG*, et elle est appliquée afin de rendre l'algorithme JPEG robuste, car la perte d'un ou de plusieurs blocs sur la séquence de transmission ne perturbera pas la reconstruction des blocs bien reçus. Les graphiques de la figure 5.6 comparent ICES avec JPEG, illustrant le ratio de distorsion pour les images « Lenna », « Baboon », et « Corridor ».

Comme prévu, nous constatons que JPEG rapporte des meilleurs PSNR que ICES même si on compresse avec des facteurs de qualité de près de 100%, atteignant aussi de plus fort taux de compression.

Les images des figures 5.7(a) et 5.7(b) montrent l'image de test « Lenna » comprimé par JPEG avec des facteurs de qualité Q=30 et par ICES à 3.75bpp, respectivement. En regardant l'œil de Lenna mis à l'échelle (figure 5.7(c)), on constate que la perte de la qualité est presque imperceptible quand on applique JPEG avec des taux de compression de 0.47bpp (voir les figures 5.7(d) et 5.7(e)). Une certaine dégradation peut être aperçue quand on applique l'algorithme de compression ICES à 3.75bpp (voir la figure 5.7(f)).

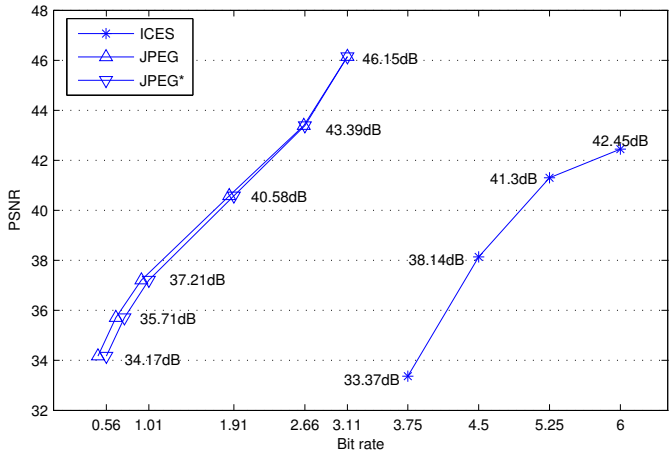
En résumé, la comparaison visuelle, ainsi que les PSNR obtenus, montrent les très hautes performances de JPEG, l'indiscutable algorithme de compression d'images de nos jours, donnant de taux de compression et des qualités d'image très élevés à différents débits. Dans tous les cas, les performances du JPEG sont incontestables. La complexité et l'évolution de cette méthode de compression lui permettent de réaliser des grands taux de compression et de rapporter des hautes qualités d'image dans presque tous les cas. Dans les prochaines sections de ce papier, nous discutons l'applicabilité de JPEG pour être integré sur des dispositifs de capture d'image actuels.

5.3 Évaluation des ressources consommées sur un capteur d'image réel

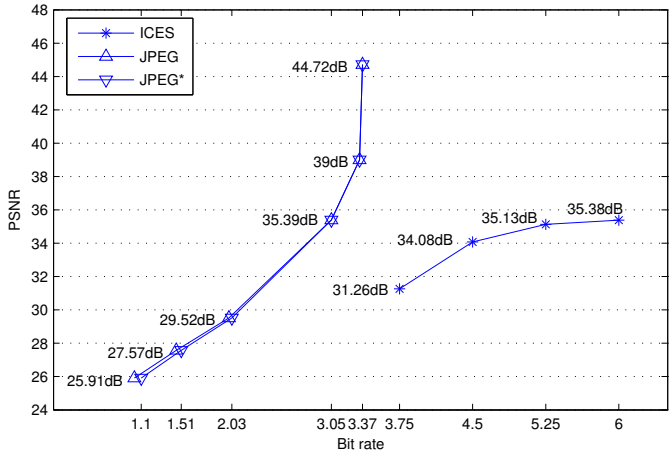
Dans cette section nous évaluons les performances de ICES, ainsi que de PR, UQ et JPEG, dans la cadre de leur implantation sur un vrai capteur d'image sans fil, constitué d'une caméra Cyclops montée sur un mote Mica2. Les détails de la plateforme matérielle et de la programmation des capteurs, ainsi que la méthode utilisée pour mesurer l'energie consommée et le temps d'exécution des applications sont donnés dans la section 1.3.3. Comme pour l'algorithme d'entrelacement de pixels proposé dans le chapitre 4, les applications de compression d'images ont été évaluées en considérant trois critères de performance : la quantité de memoire requise, le temps d'exécution d'un cycle de travail (incluant la capture d'image, la compression des données et la transmission des paquets), et bien sûr, la consommation d'énergie.

5.3.1 Quantité de mémoire requise

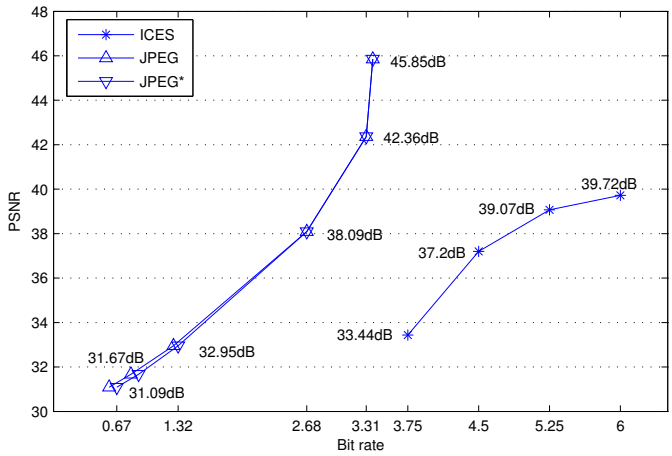
A partir du processus de compilation des programmes développés en `NesC` qui génère le code exécutable sur la caméra Cyclops, nous pouvons recueillir les informations sur la quantité de mémoire ROM et RAM qui ont été allouées.



(a) Image de test « Lenna » (512 × 512 pixels).



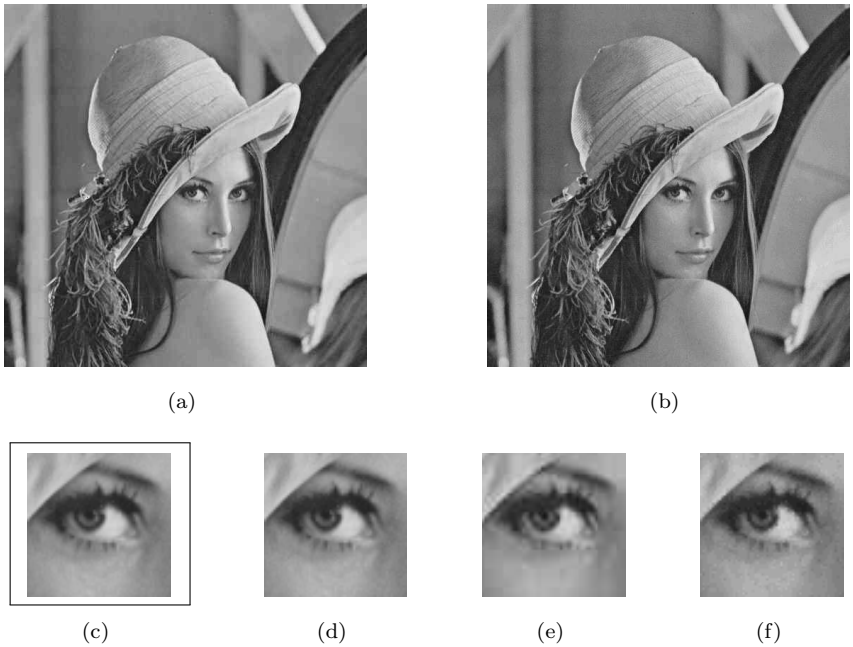
(b) Image de test « Baboon » (512 × 512 pixels).



(c) Image de test « Corridor » (128 × 128 pixels).

FIG. 5.6: Débit/Distorsion de ICES vs. JPEG.

Comme son nom l'indique, la *Read Only Memory* (ROM), connue aussi sous le nom de *Mémoire*



(a) (b)

(c) (d) (e) (f)

FIG. 5.7: Première rangée : visualisation de l'image « Lenna » reconstruite après compression par (1) JPEG avec facteur de qualité Q=30 (0.47bpp ; PSNR = 34.17dB) et (b) ICES à 3.75bpp (PSNR = 33.37dB).
Deuxième rangée : comparaison de « l'œil Lenna » mis a l'échelle et reconstruite après compression par (c) JPEG avec facteur de qualité (d) Q=97 (3.11bpp ; PSNR = 46.15dB) et (e) Q=30 (0.47bpp ; PSNR = 34.17dB), et (f) par ICES à 3.75bpp (PSNR = 33.37dB).

morte, est une mémoire dans laquelle toutes les données qu'on enregistre restent sans modification lorsque l'unité qui la contient n'est plus alimentée. Dans le processus de compilation la mémoire RAM se réfère principalement à la mémoire de programme, c'est-à-dire, à la mémoire destinée à enregistrer les codes binaires de l'application que nous enregistrons sur la mote. D'autre part, la *Random Access Memory* (RAM), ou *Mémoire vive* est celle qui est utilisée pour stocker les données lors de leur utilisation. Cette mémoire est effacée quand l'unité n'est plus alimentée électriquement.

Les applications ont des besoins de mémoire différents, selon la complexité des algorithmes de compression et le processus d'allocation de bits. Pour l'application `captureRadioTest` originale, basée sur le composant `RadioDump` du firmware de la caméra Cyclops, c'est-à-dire pour l'application de transmission d'image non compressée et non mélangée (NC), les besoins de mémoire ROM et RAM sont respectivement de 17332 et 987 octets (le composant logiciel associé aux `Leds` a été enlevé car l'activation des leds a un coût d'énergie significatif). Toutes les applications assurant la compression de l'image avant la transmission ont des besoins mémoire supérieurs à l'application NC, ce qui est tout à fait logique. Les résultats sont présentés sur le tableau 5.3.

En ce qui concerne les algorithmes de compression de faible complexité, incluant donc UQ, PR et ICES, nous pouvons observer que ICES a besoin de légèrement plus de mémoire ROM que les autres.

Tab. 5.3: Comparaison des allocations de mémoire

Méthode	ROM (bytes)	RAM (bytes)
NC	17332	987
UQ	17362	1024
PR	18046	1026
ICES	18356	1029

Dans le pire de cas (ICES à 3.75bpp), ICES nécessite 18356 octets de mémoire ROM et 1029 octets de mémoire RAM. Cela s'explique par le fait que ICES réalise un peu plus de calculs que UQ et PR pour compresser chaque bloc. Au final, l'application ICES a juste besoin de 1024 octets de ROM et 42 octets de RAM de plus que l'application de référence NC. C'est donc le coût propre de l'algorithme de compression. C'est peu quand on sait qu'un capteur Cyclops dispose au total de 128Ko de mémoire de programme et de 4 Ko de RAM. On peut donc affirmer que ICES est un algorithme tout à fait adapté à une implantation dans des systèmes embarqués dotés de très peu de ressources mémoire, comme cela est la norme dans le contexte des réseaux de capteurs.

Evidemment, la plus grande complexité de l'algorithme JPEG conduit a des besoins de mémoire plus importants que ICES. Mais JPEG peut être implanté sur une caméra Cyclops sans problèmes. JPEG requiert de la mémoire pour stocker non seulement le code nécessaire aux calculs sur les données (DCT, quantification scalaire, codage RLE et codage Huffman) mais aussi pour stocker la matrice de quantification et les tables contenant les codes de Huffman. L'application JPEG exige au final 26082 octets de ROM et 2419 de RAM, soit 8750 octets de ROM et 1432 octets de RAM de plus que l'application de référence NC. C'est donc le coût propre de l'algorithme JPEG. C'est huit fois plus de ROM et deux fois plus de RAM que ICES. Nous verrons dans les sections suivantes que, même si JPEG peut être implanté dans des capteurs d'image comme Cyclops, il a un temps d'éxécution et un coût d'énergie trop élévé pour qu'il soit rentable.

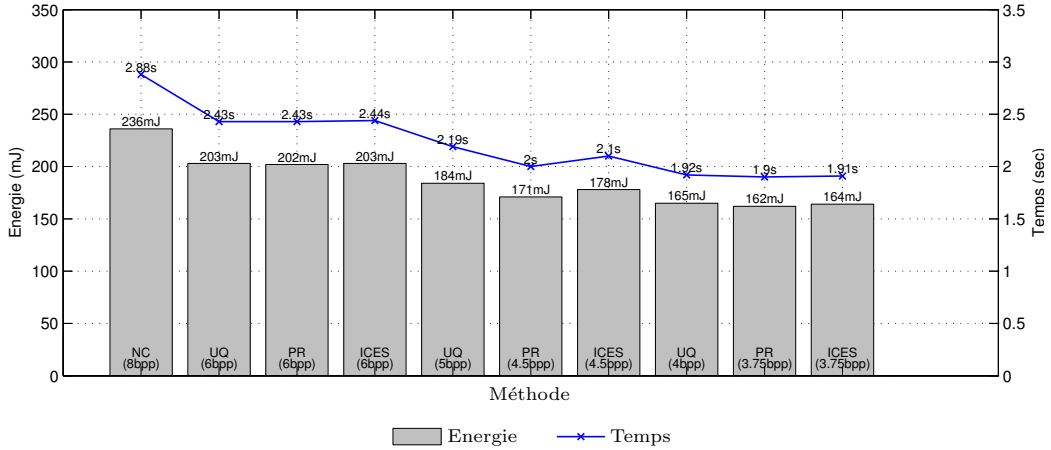
5.3.2 Temps d'exécution et consommation d'énergie

Nous avons mesuré la consommation d'énergie et le temps d'éxecution d'un cycle de travail (capture de l'image, compression de données et transmission par paquets) des applications ICES, PR, UQ, JPEG et NC (cette dernière servant de référence et bien sûr, ne compressant pas les données). Les mesures ont été réalisées avec notre banc d'éssai décrit section 1.3.3.

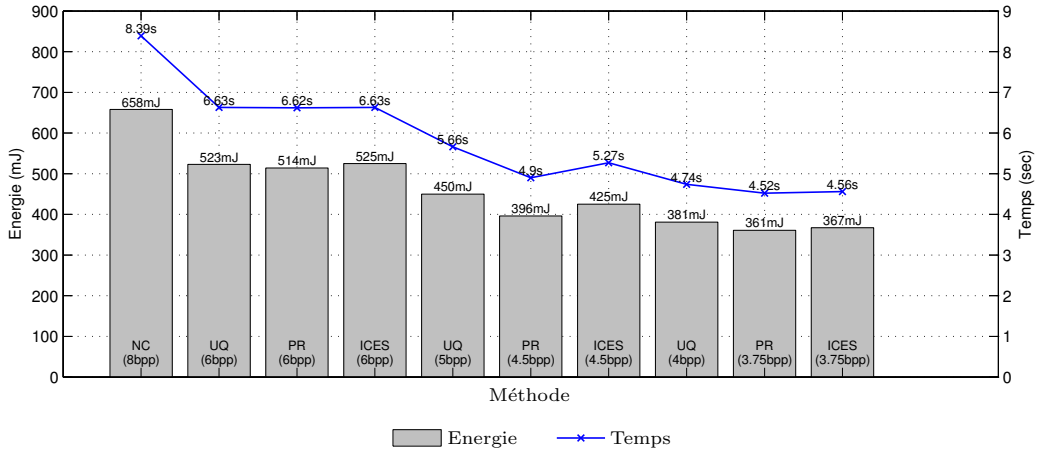
Afin de rendre le coût de la compression d'image le plus significatif possible, nous avons configuré la puissance de transmission du transcepteur radio à -20dBm, qui est la puissance minimale des motes Mica2.

La figure 5.8 présente les résultats pour les applications NC, PR, UQ et ICES en considérant trois tailles d'image différentes 32×32 (figure 5.8(a)), 64×64 (figure 5.8(b)) et 128×128 (figure 5.8(c)). Précisons que les résultats de l'application JPEG ne sont pas du même ordre de grandeur, et pour cette raison, ils sont reportés sur une autre figure.

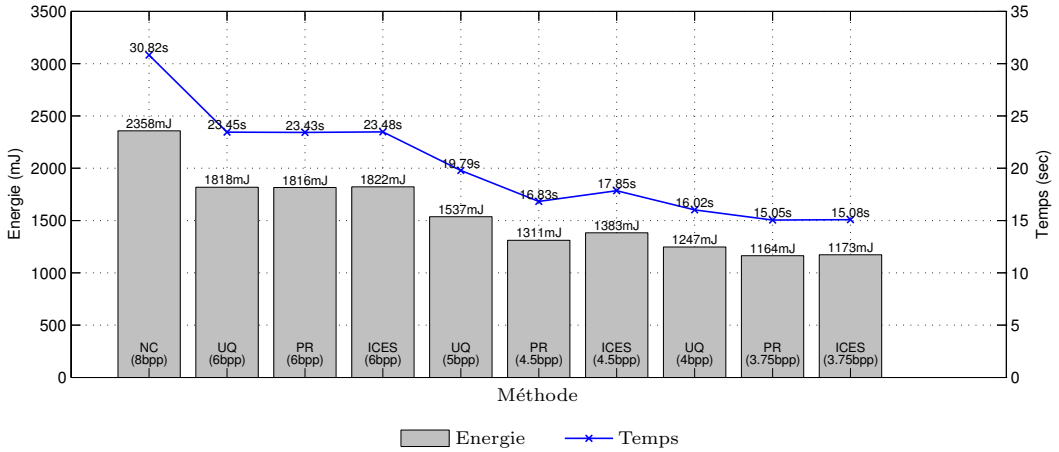
Nous pouvons observer une nette amélioration des performances pour les deux critères d'évaluation quand une compression simple de l'image est effectuée avec UQ, PR ou ICES. Par exemple, pour les applications qui fournissent des taux de compression 4 : 3, soit un débit binaire de 6bpp, les économies



(a) Consommation d'énergie et du temps d'exécution pour un cycle de travail avec une image de 32×32 pixels.



(b) Consommation d'énergie et du temps d'exécution pour un cycle de travail avec une image de 64×64 pixels.



(c) Consommation d'énergie et du temps d'exécution pour un cycle de travail avec une image de 128×128 pixels.

FIG. 5.8: Consommation d'énergie et du temps d'exécution pour les algorithmes de compression simples étudiés avec des débits binaires différentes.

d'énergie sont d'environ 14%, 20% et 22.6% respectivement, quand on capture des images de 32×32, 64×64 et 128×128. Des gains de temps similaires sont observés pour ces applications. Nous constatons logiquement que ICES est un peu plus gourmand en énergie et en temps que PR et UQ, mais la différence n'est jamais considérablement prononcée. Pour les images de 32×32 par exemple, ICES presente un excès de temps d'exécution de 25ms et 22.4ms en comparaison respectivement avec PR et UQ. De même, le surcoût d'énergie est seulement de 2mJ et 1mJ. Ce léger surcoût est acceptable puisque ICES fournit un meilleur rapport débit-distorsion que UQ et PR (voir figure 5.3). Les économies d'énergie et de temps deviennent de plus en plus intéressants lorsque l'on augmente le taux de compression. Pour des images codées au débit binaire de 3.75bpp, on atteint jusqu'à environ 50% d'économie d'énergie et de temps d'exécution avec ICES en comparaison avec l'application de référence NC.

ICES est donc un algorithme de compression efficace en énergie puisque les nœuds source, les capteurs d'images donc, vont pouvoir réduire leur consommation d'énergie pour un cycle de travail d'un facteur de 2. Cela va entraîner nécessairement une augmentation significative de leur durée de vie. Pour prendre un exemple, imaginons un scénario où les capteurs d'image devraient capturer et transmettre au puits une image monochrome 128×128 par minute. Ils s'activeront donc a chaque minute, exécuteront un cycle de travail, et retourneront à l'état endormi jusqu'au prochain réveil. Avec l'application NC, c'est-à-dire sans compression, la durée de vie des capteurs d'image serait précisément de 6 jours, 6 heures et 45 minutes, équivalent à la transmission de 9045 images. Avec l'application ICES compressant les images au débit de 3.75bpp, la durée de vie des capteurs serait de 12 jours, 7 heures et 35 minutes, et 17735 images auraient été transmises.

Maintenant, comparons ICES avec JPEG. Dans la section 5.2.3, nous avons vu que JPEG offrait des rapports débit-distorsion très largement supérieurs à ICES, ce qui était attendu vu la faible complexité de ce dernier. Dans une dernière série d'expériences, nous avons évalué l'application JPEG* avec notre caméra Cyclops. Les résultats sont donnés figure 5.9. Ils révélent immédiatement que JPEG* n'est pas rentable puisqu'il consomme (beaucoup) plus d'énergie que l'application NC. Le coût d'énergie est principalement dû à la DCT. Cette opération coûte en effet de l'ordre de grandeur de 44mJ et 545ms par bloc de 8×8 pixels. Soit pour une image entière de 128×128, un coût d'énergie de 11264mJ, et un temps d'exécution d'environ 139.52s, juste pour la DCT. Le coût de la quantification est d'environ 0.6mJ d'énergie et 10ms du temps, et cel du codage d'environ 0.8mJ et 10ms d'énergie et du temps, respectivement. Le coût de JPEG* (y compris la quantification et le codage) est d'environ 13727mJ et 168.56s lors de la compression avec un facteur de qualité Q=97% et d'environ 12163mJ et 150.74s lors de la compression avec Q=30%. Ces résultats dépassent largement ceux obtenus même par une transmission sans compression avec un transcepteur radio à pleine puissance. Avec une puissance de 5dBm (c'est le maximum autorisé par notre mote Mica2), la consommation d'énergie pour l'application NC augmente jusqu'à 2830mJ.

Dans tous les cas, l'application de JPEG* a des performances négatives. De tous les résultats, nous concluons que ICES est un algorithme efficace pour être appliqué sur ces nœuds caméras de ressources limitées, car il réalise de la compression d'image à très faible consommation d'énergie avec un rapport débit-distorsion pas si mauvais que cela.

Un atout supplémentaire de ICES, que nous allons développer dans la section suivante, est que l'image compressée est robuste aux pertes de paquets puisque les blocs sont codés indépendamment les uns des autres. ICES peut à juste titre être couplé avec une méthode d'entrelacement de (blocs de) pixels pour

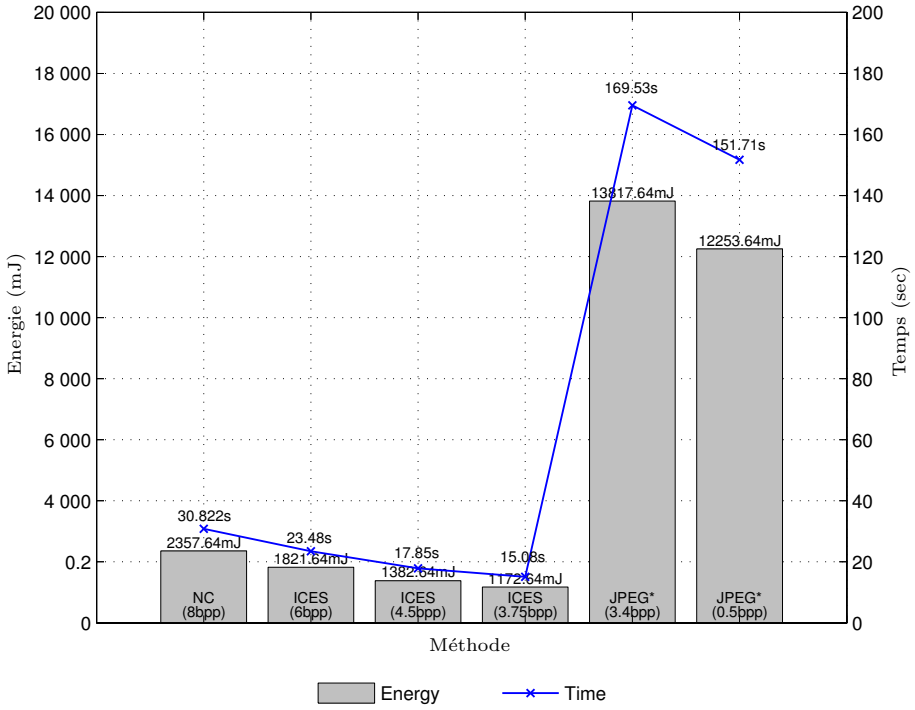


FIG. 5.9: Consommation d'énergie et du temps d'exécution pour les algorithmes de compression simples étudiés avec des débits différents.

optimiser cette résistance aux pertes de paquets.

5.4 Couplage ICES et AT

Lorsque nous avons conçu ICES, nous avons choisi volontairement de travailler avec des tous petits blocs, en sachant que, forcément, cela limiterait les taux de compression que l'on pourrait atteindre. Notre intention était de disposer d'un système de compression qui ne nous obligerait pas, derrière, à devoir contrôler les pertes de paquets au niveau du système de communication. Autrement dit, on ne voulait pas que les gains d'énergie obtenus grâce à la compression de l'image soient perdus ensuite par l'ajout d'un mécanisme d'acquittement et de retransmission des paquets.

ICES a donc été conçu pour que les images, une fois compressées, puissent être transmises sur le réseau en utilisant un protocole de communication non fiable. Pour réduire les effets des pertes de paquets sur qualité des images reconstruites au récepteur, ICES peut être avantageusement couplé avec une méthode d'entrelacement de pixels comme celle utilisée dans le chapitre 4. Dans cette section nous décrivons comment nous avons réalisé un tel couplage, puis nous évaluons les performances par des expérimentations.

5.4.1 Principes techniques

Le mélange des pixels de l'image se fait ici, non plus pixel par pixel, mais bloc de pixels par bloc de pixels. En principe, les blocs de l'image sont mélangés avec AT*, et ensuite les blocs sont compressés avec ICES. La chaîne de compression est donc légèrement modifiée, elle est schématisée figure 5.10.

En pratique, les deux opérations sont réalisées à la volée pour chaque bloc à la construction des paquets

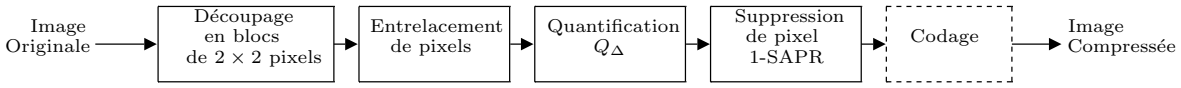


FIG. 5.10: Schéma de compression ICES.

de données, selon la séquence : calcul des coordonnées d'un bloc - compression du bloc - rangement des données dans le paquet en cours. Si le bloc compressé est plus grand que l'espace disponible dans le paquet en construction, il est stocké en memoire et, une fois que le paquet actuel est envoyé, il est mis dans un nouveau paquet. Cet algorithme suppose que la taille d'un bloc compressé est toujours inférieure ou égale à la quantité d'espace réservé dans le paquet pour la transmission de données.

Le pseudo-code est donné par l'algorithme 3.

Algorithme 3 Adapted TA-based compressed-block interleaving

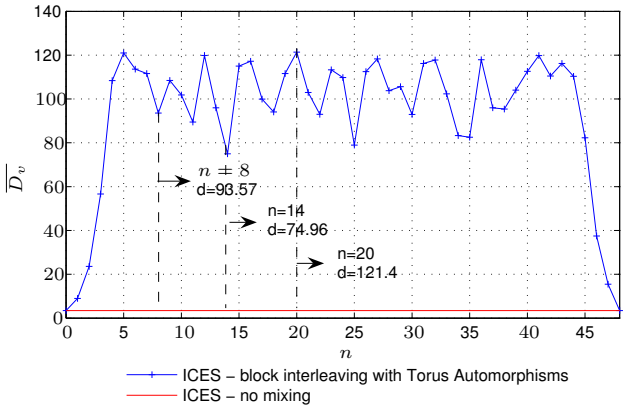
$H \Leftarrow ImageHeight$, $W \Leftarrow ImageWidth$
for $p = 0$ to $\left(\frac{H}{BlockHeight} - 1\right)$ **do**
  for $q = 0$ to $\left(\frac{W}{BlockWidth} - 1\right)$ **do**
    Calculate (p', q') of position (p, q) using TA
    Compress the block in (p', q'), $B_{p',q'}$
    if sizeof(Compressed Block) > available space on $Packet.data$ **then**
      Send $Packet$
    end if
    Packetize Compressed Buffer $B'_{p',q'}$
    if ($Packet$ is full) or $((p, q) = \left(\left(\frac{W}{BlockHeight} - 1\right), \left(\frac{H}{BlockWidth} - 1\right)\right))$ **then**
      Send $Packet$
    end if
  end for
end for

Notons à partir de l'algorithme que le système proposé de paquétisation enregistre chaque bloc compressé entièrement dans un seul paquet, c'est-à-dire, l'information codée représentant les intensités de tous les pixels qui forment un bloc doit être enregistré entièrement dans un seul paquet et non pas divisé entre deux ou plusieurs paquets. Ainsi, si le paquet est bien reçu du côté du décodeur, tous les blocs qu'il contient seront immédiatement décodables.

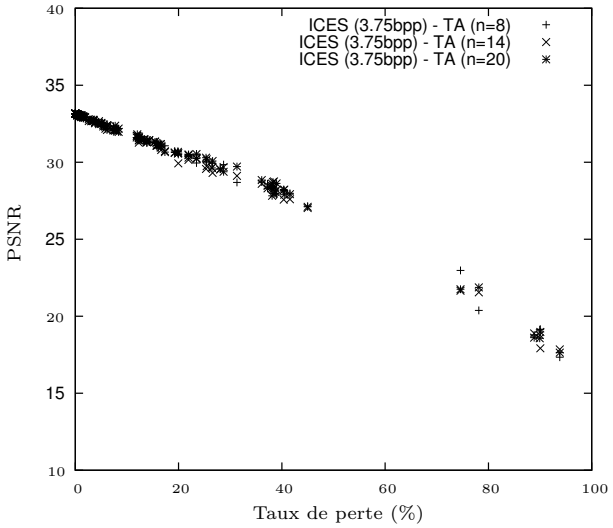
5.4.2 Évaluation de la fonction d'entrelacement de blocs compressés avec ICES

De la même manière que nous l'avons fait dans la section 4.6 (cf. chapitre 4), lorsque nous avons proposé la mise en œuvre des automorphismes pour l'entrelacement de pixels, nous allons évaluer l'influence du choix de la clé de diffusion des AT, n, dans la qualité des images reconstruites après sa compression avec l'algorithme ICES et le mélange par blocs. Le graphique de la figure 5.11(b), montre la distance moyenne entre blocs voisins (notons que cette fois-ci la fonction $\overline{D_v}$ représente la distance entre blocs et non pas entre pixels) par rapport à la clé de diffusion n. Nous constatons que, bien évidemment, les dis-

tances maximales atteignables descendent à environ 100 paquets. De la figure 5.11(b) nous pouvons aussi constater que malgré l'utilisation de clés de diffusion de différentes valeurs (dans la zone d'oscillation), de la même manière que dans le cas étudié dans le chapitre précédent, la distance entre blocs voisins reste non significative.



(a) Évaluation de la distance moyenne des pixels voisins pour les automorphismes toriques adaptés.



(b) Qualités observés sur l'image « Couloir » pour les schèmas sans et avec mélange par automorphismes toriques avec différentes clés de diffusion n.

FIG. 5.11: Évaluation de l'influence de la clé de difusion n des automorphismes toriques dans la qualité d'images compressées et mélanges par blocs.

5.4.3 Évaluation des performances

Nous avons refait des expérimentations similaires à celles du chapitre 4, où nous avions évalué l'efficacité des AT pour des images non compressées, mais cette fois-ci, AT était couplé avec ICES. Les expérimentations ont portées sur l'image test « Corridor ». Les perte de paquets étaient produites sur la base des traces que nous avions enregistrées (cf section 1.3.4).

La figure 5.12 montre l'évolution du PSNR de l'image en fonction du pourcentage de pertes de paquets pour deux scénarios : transmission d'une image compressée par ICES avec et sans couplage avec AT, c'est-à-dire avec et sans mélange.

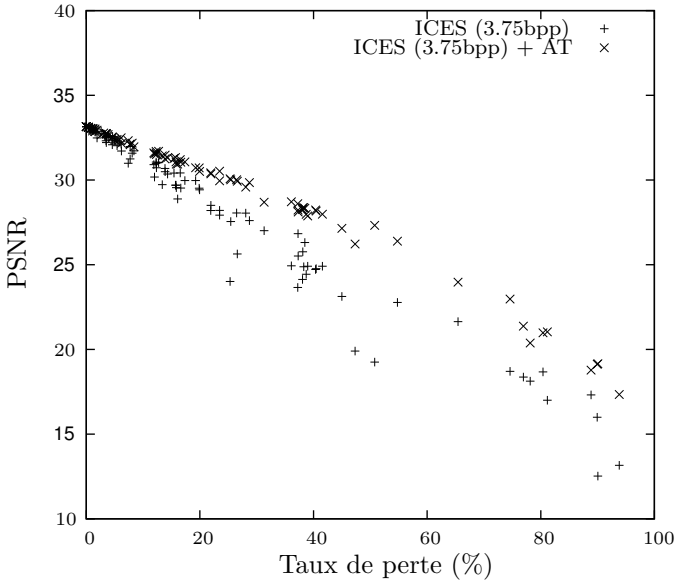


FIG. 5.12: Comparaison de la qualité pour l'image « Corridor » après compression et transmission par ICES avec et sans mélange.

Les résultats montrent, comme attendu, une nette amélioration de la qualité des images lorsque ICES est couplé avec AT. Pour un pourcentage de pertes de 40% par exemple, ce qui est déjà considérable, le gain de PSNR est de 3.5dB lorsque l'image est mélangée. En contrepartie, le coût d'énergie supplémentaire à la source est modeste : 1240 mJ pour ICES+AT vs. 1173 mJ pour ICES. De même, le coût en besoin de mémoire est, lui aussi, très faible : 18504 de ROM et 1064 de RAM vs 18356 de ROM et 1029 de RAM. Il est donc tout à fait bénéfique de coupler ICES avec AT.

Bien sûr, le choix de tous petits blocs n'est pas étranger à ces bons résultats. L'efficacité de la stratégie de mélange de l'image est intimement liée à la taille des blocs. Plus les blocs sont grands, et plus la dissimulation des blocs manquants au récepteur sera dure à réaliser. Pour s'en convaincre, nous avons comparé nos résultats avec un scénario où l'image serait mélangée par bloc de 8×8 pixels, puis chaque bloc serait compressé avec JPEG*. Les résultats sont donnés sur la figure 5.13.

Des exemples d'images reconstruites en présence de pertes de paquets sont aussi présentés sur le tableau 5.4 pour ICES et pour JPEG*. Les résultats confirment que notre stratégie de compression par petit blocs est efficace. Regardons par exemple les images reconstruites après avoir perdu environ 40% des paquets : Avec JPEG*, nous voyons que les pertes subies par l'image ont conduit à la perte d'informations très importantes (la tête de l'individu observé, par exemple). Comme les blocs de pixels sont gros, la perte de l'un d'eux est difficile à récupérer. Notons que l'application de notre technique de mélange des blocs ne conduit pas à une amélioration de la qualité de l'image affichée (même le PSNR est affecté). Comme nous pouvons le remarquer dans le graphique de la figure 5.13, l'entrelacement de blocs ne produit pas des améliorations substantielles. Les résultats sont assez vagues et, dans de nombreux cas, l'application de la technique de mélange n'améliore pas et même empire les résultats.

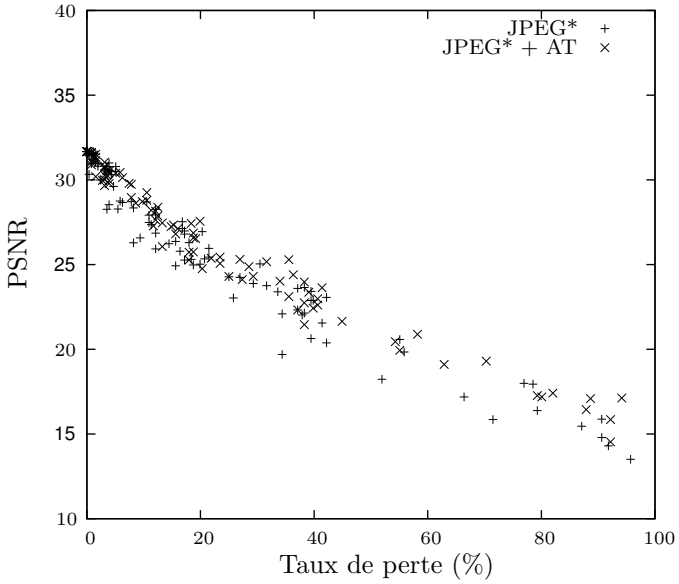


FIG. 5.13: Comparaison de la qualité pour l'image « Corridor » après compression et transmission par JPEG* avec et sans mélange.

5.5 Conclusion

Dans ce chapitre, nous avons proposé un algorithme de compression d'images par bloc de très faible complexité, pour être peu gourmand en énergie, couplé à un algorithme de mélange de blocs de pixels, pour être résistant aux pertes potentielles de paquets. Les images compressées de cette manière, si elles ne peuvent bien sûr pas avoir des taux de compresion du même ordre de grandeur que ceux obtenus avec des algorithmes comme JPEG par exemple, ont l'immense avantage de pouvoir être transmises en utilisant un protocole de communication non fiable, donc lui même pas gourmand en énergie, tout en conservant une qualité correcte au récepteur, même pour des taux de perte de paquets élevés, de l'ordre de 40% à 50%.

Les taux de compression atteignables restant modestes puisqu'ils sont légèrement supérieurs à 2, mais ils permettent justement aux nœuds sources de réduire leur consommation d'énergie dans les mêmes proportions. Ce n'est pas négligeable et cela entraînera une augmentation de la durée de vie du réseau de capteurs.

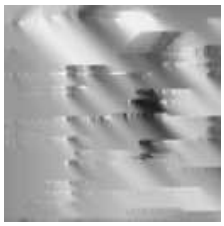
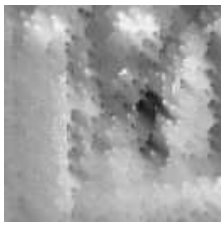
Le couple (ICES+AT) est aussi l'un des tous premiers algorithmes de compression d'image implantés sur une plateforme réelle, ce qui prouve son efficacité pratique. C'est aussi l'un des tous premiers algorithmes de compression d'image qui est évalué, au delà des critères classiques qui sont le rapport débit-distorsion et la complexité (qui est directement correlé au coût d'énergie), en considérant sa résistance aux pertes de paquets par le réseau.

TAB. 5.4: Visualisation de la qualité des images compressées par JPEG* avec facteur de qualité du Q=95%, transmises avec plusieurs taux de pertes

(a) Exemples d'images reconstruites dans le cas d'une compression avec JPEG*

Loss rate	19.53%	39.45%	55.49%	79.30%
JPEG*	 PSNR = 29.02dB	 PSNR = 23.98dB	 PSNR = 20.47dB	 PSNR = 18.66dB
JPEG* + AT	 PSNR = 27.95dB	 PSNR = 22.16dB	 PSNR = 20.48dB	 PSNR = 17.30dB

(b) Exemples d'images reconstruites dans le cas d'une compression avec ICES à 3.75bpp

Loss rate	21.90%	40.36%	65.43%	80.40%
ICES	 PSNR = 29.20dB	 PSNR = 24.72dB	 PSNR = 21.64dB	 PSNR = 18.68dB
ICES + AT	 PSNR = 30.43dB	 PSNR = 28.23dB	 PSNR = 23.98dB	 PSNR = 20.98dB

Conclusions

Les travaux menés dans cette thèse se situent à l'intersection des domaine des réseaux de capteurs sans fil et du traitement d'images. Nous avons étudié les principaux travaux de recherche dans le domaine des réseaux de capteurs, en regard plus particulièrement des applications des réseaux de capteurs d'image. Cette étude faisant l'objet des deux premiers chapitres du document. Nous avons mis en avant que le principal problème à résoudre est la durée de vie des batteries embarquées sur les capteurs. Il y a deux grandes approches pour diminuer la consommation d'énergie des capteurs. La première vise la compression de l'image au niveau de la source. En effet, des énormes économies d'énergie soit envisageables si on applique des algorithmes de compression locaux de faible complexité (donc de faible consommation d'énergie). Ces économies toucheront le nœud source ainsi bien que les nœuds de transit puisque la compression amène mécaniquement une réduction du nombre de paquets à transmettre jusqu'au puits. La deuxième approche vise le protocole de communication, en optimisant la sélection des routes, la répartition du trafic entre les nœuds, les cycles d'endormissement des nœuds, ... Après une vaste revue de la littérature existante, nous avons pointé le fait qu'un grand nombre de propositions, bien que très attractives d'un point de vue théorique, n'ont pas été validées par des expérimentations sur de vrais capteurs d'images. En effet, les méthodes d'évaluation sont parfois peu satisfaisantes car les modèles mathématiques et de simulation ne considèrent pas toujours les contraintes du monde réel comme la limitation des ressources des capteurs et les pertes de paquets du réseau. La complexité des protocoles et des mécanismes necessaires pour mettre en œuvre ces propositions est souvent laissée de côté. L'objectif de cette thèse était de développer des procédures de traitement et de transmission d'images prenant en considération non seulement l'énergie consommée mais aussi la qualité des images finales en présence de pertes de paquets.

Avec les images naturelles, des économies d'énergie peuvent être obtenues en relachant la contrainte de fiabilité du protocole de communication si on profite des redondances spatiales existantes entre les pixels voisins dans l'image. Une première solution sur laquelle nous avons travaillé a été presentée dans le troisième chapitre. Sachant que le coût des acquittements et des retransmissions dans un scénario avec un protocole fiable peut être très coûteux en énergie, même dans un scénario hypothétique sans pertes de paquets, nous avons étudié l'impact d'une rupture avec le schéma de transmission fiable traditionnel et l'application d'un schéma de transmission semi-fiable. Avec ce schéma, une partie des données seulement (les plus importantes pour la reconstitution de l'image finale) étaient transmises de manière fiable jusqu'au puits. Pour le reste, les nœuds de transit entre la source et le puits décidaient de relayer ou pas les paquets selon une politique qui considèrait l'état énergetique des nœuds. L'analyse mathématique de la consommation d'énergie a révélé que notre protocole semi-fiable entraînait des économies d'énergie pouvant aller jusqu'au 70% quand on considère un chemin avec 30 nœuds de transit sans pertes de paquets.

Cependant, nous sommes conscients que cette évaluation mathématique fournit une approximation peu précise de la réalité. Mais ce premier résultat montre que l'application d'une politique de transmission semi-fiable peut mener à des économies d'énergie significatives. Logiquement, cela serait encore plus intèressant dans un scénario avec beaucoup de pertes de données. En situation réelle toutefois, nous pensons que le système de communication devrait idéalement s'appuyer sur un protocole non-fiable, qui représente la solution la moins coûteuse en énergie, à la condition de renforcer la tolérance des images aux pertes de paquets.

Dans le quatrième chapitre, nous avons travaillé sur ce point, le renforcement de la tolérance des images aux pertes de paquets étant obtenu par une méthode d'entrelacement de pixels. Autrement dit, il s'agissait de transmettre l'image en plaçant dans les paquets des pixels qui étaient distants les uns des autres dans le bitmap de l'image, de façon que, si des paquets étaient perdus, on ait une forte probabilité de trouver des pixels voisins à ceux qui manquaient pour pouvoir les estimer avec une bonne précision en utilisant une technique de dissimulation d'erreurs. Nous avons démontré qu'une méthode d'entrelacement de pixels basée sur les *Automorphismes Toriques* amène un surcoût d'énergie pour la source qui est très faible tout en entraînant une amélioration considérable de la qualité des images finales en présence de pertes de paquets. Nous avons évalué cela en programmant les automorphismes toriques sur une vraie plateforme de réseau de capteurs sans fil composée de motes Mica2 dont l'un était équipée d'une caméra Cyclops.

Jusqu'à ce point, nos propositions portaient sur des images non compressées. Dans le cinquième chapitre, nous avons proposé un algorithme de compression de très faible complexité, ICES, qui opère par blocs indépendants de 2×2 pixels. L'idée était de conserver l'approche du chapitre 4, transmettre des images mélangées par un protocole de communication non fiable, mais en réduisant le volume de données à transmettre pour économiser plus d'énergie. En implantant notre proposition sur nos capteurs Cyclops, nous avons démontré sa faisabilité pour des systèmes limités en ressources. Les performances obtenues sont probantes puisqu'on arrive à des économies d'énergie et du temps d'exécution qui peuvent atteindre jusqu'à 50%. Nous avons aussi démontré que l'application d'algorithmes standards comme JPEG, programmés de façon traditionnelle, peuvent amener à des consommations d'énergie qui dépassent celles d'une transmission sans aucune compression. En outre, le découpage en très petits blocs adopté dans notre algorithme de compression fournit une meilleure résistance aux pertes de paquets que des méthodes opérant classiquement sur des blocs de 8×8 pixels. Cette résistance est encore améliorée en appliquant une stratégie d'entrelacement de blocs analogue à celle présentée dans le chapitre 4.

En résumé, cette thèse est l'une des premières à traiter des réseaux de capteurs d'image en France. Les résultats exposés dans les deux derniers chapitres sont particulièrement significatifs puisque les performances ont été validées sur une vraie plateforme de réseaux de capteurs sans fil. Nous avons obtenu des économies importantes en temps d'exécution et d'énergie. Même si dans ces deux derniers chapitres, nous n'avons pas fait d'évaluation sur un réseau à grande échelle, il semble évident que le fait de n'avoir ni à acquitter les paquets, ni à les retransmettre devrait être bon pour tous les nœuds du réseau globalement. Une approche de transmission d'image basée sur un protocole non fiable présente beaucoup d'avantages. Les temps de transmission sont diminués (rappelons que la seule transmission d'une image de 128×128 pixels avec un capteur Cyclops peut prendre environ 30 seconds avec un seul saut), ainsi que la charge du réseau (donc la probabilité de collision et de congestion devrait diminuer). Toutefois, nous sommes conscients que d'autres schémas de compression apparaîtront bientôt et qu'ils fourniseront de meilleures

performances. Cette thèse représente une des premières contributions vers un système de compression et de transmission d'images pour des réseaux de capteurs sans fil qui considère des contraintes réalistes comme la limitation de l'énergie et la limitation des ressources de calcul, de mémoire et de communication, mais aussi des contraintes applicatives comme l'indépendance à des topologies de réseau particulières, ou des besoins de mécanismes d'échange d'information trop complexes (comme pourrait être le cas d'un approche de compression distribuée).

Perspectives

Les travaux présentés tout au long de cette thèse ont traité de la compression et de la transmission d'images sur des réseaux de capteurs sans fil sous la contrainte de l'énergie et de pertes de paquets. Plusieurs perspectives peuvent être envisagées sur la base de ces travaux.

Du point de vue du traitement des images, il faut continuer à chercher et a comprarer des algorithmes sur la base d'un ratio énergie-distortion, plutôt que du ratio débit-distortion qui ne considère pas l'impact des pertes de paquets. Dans l'approche d'entrelacement présentée dans le chapitre 4, seule la technique basée sur les automorphismes toriques a été testée. Même si cette technique a plusieurs avantages, comme la difficulté de déchiffrement, la vitesse de calcul et la bonne distance entre pixels/blocs obtenue, il serait intéressant de rechercher d'autres algorithmes qui présentent des performances comparables. Pour aller plus loin, la recherche d'un schéma optimal pourrait être envisagé. Pour ceci il faudrait modéliser premièrement une fonction d'optimisation plus complète, c'est-à-dire qui considère plus de facteurs que la seule distance entre pixels ou blocs, par exemple des facteurs comme un modèle de pertes, des facteurs liés à l'utilisation de ressources pour le calcul, ou la corrélation entre les résultats pour les pixels/blocs qui sont voisins entre eux.

Les résultats obtenus dans les deux derniers chapitres montrent en definitive que l'entrelacement de pixels a une forte incidence dans la robustesse de la transmission sur un milieu perturbé en appliquant un protocole non-fiable. Cependant, il existe d'autres techniques que pourraient être évaluées comme le tatouage d'images par exemple, utilisé généralement pour la protection des données informatiques (copies illegales, modification, etc.) (Chen *et al.*, 2003; Parisis *et al.*, 2004), mais qui peut être aussi appliqué afin d'embarquer dans les données de l'image des informations redondantes utiles à sa reconstruction. Ce sujet fait l'objet de la thèse de Leila Mekkaoui, démarrée en septembre 2008 sous la direction de J-M. Moureaux, au sein du Centre de Recherche en Automatique de Nancy. Son objectif est de trouver de nouvelles méthodes conjointes de compression et de tatouage d'image satisfaisant aux contraintes posées par les réseaux de capteurs sans fil.

Dans la méthode de compression proposée dans le chapitre 5, deux méthodes de suppression de pixels auto-adaptatifs ont été évaluées : une basée sur la moyenne des pixels du bloc et une autre basée sur la duplication de l'un des pixels. Des expérimentations sur des images de test classiques nous ont permis de déduire intuitivement que la deuxième méthode donnait les meilleurs résultats. D'autres variantes pourraient être étudiées. D'autre part, ICES fournit jusqu'à maintenant pour des images codées à 8bpp un débit binaire de 3.75bpp au maximum, pas négligable mais qui peut être amélioré en ajoutant une étape de codage par codes à longueur variable pour réduire encore plus la quantité de données à transmettre. Une approche « *cross layer* » tenant compte des caractéristiques du canal physique pour le codage pourrait aussi être développée, suivant le principe de la méthode WTSOM (Boeglen *et al.*, 2007) mais en plus

simple.

À la fin du cinquième chapitre, nous avons démontré qu'il été possible de réaliser la compression d'image avec une très faible consommation de ressources. Nous avons comparé ICES avec une implantation classique de JPEG. Ce dernier algorithme est aujourd'hui l'un des standards de référence, mais nos experimentations ont démontré que son application sur des nœuds de capteurs est bien trop coûteuse pour le nœud source. Ces résultats ne sont pas catégoriques. L'applicabilité de JPEG et d'autres algorithmes complexes dépend beaucoup de la plateforme utilisée. Dans le cas de JPEG, la DCT est l'étape la plus gourmande en énergie, et de très loin. Plusieurs méthodes pour calculer la DCT avec des ressources minimales ont été proposées dans la littérature, comme dans (Loeffler *et al.*, 1989), où les auteurs proposent une méthode rapide pour faire une DCT 1-D avec seulement 11 multiplications. De telles méthodes sont sûrement viables pour les réseaux de capteurs et devront être évaluées. Remarquons toutefois que, même si l'utilisation de JPEG peut fournir des économies d'énergie, l'utilisation de gros blocs (8×8 pixels) implique une faible capacité à résister contre les pertes de paquets. Ceci a été demontré dans l'étude menée au dernier chapitre de cette thèse. Notre choix d'opérer sur de très petits blocs reste justifié.

Du point de vue du réseau, des expérimentations sur des réseaux multi-sauts doivent être réalisées, aussi que des simulations à grande échelle. Des études de différents protocoles de routage (mono et multi-chemins) et de couche MAC doivent être réalisées pour identifier quelles sont les meilleures stratégies pour les réseaux de capteurs d'image.

Finalement, une autre voie pour fournir des économies d'énergie concerne l'implantation des algorithmes sur circuits. Une collaboration avec le Laboratoire d'Electronique et de Micro-électronique de l'Université de Monastir en Tunisie est actuellement en œuvre pour évaluer les coûts d'implantation sur circuits FPGA et ASIC. Outre le coût d'énergie, le temps d'exécution et la surface du circuit sont les principaux critères de performance.

* * *

Liste des publications

Revues internationales avec comité de lecture

V. Lecuire, C. Duran-Faundez, and N. Krommenacker, *Energy-efficient image transmission in sensor networks*. International Journal of Sensor Networks (IJSNet), 4(1-2), 37-47.

V. Lecuire, C. Duran-Faundez, and N. Krommenacker, *Energy-efficient transmission of wavelet-based images in wireless sensor networks*. EURASIP Journal on Image and Video Processing 2007, Article ID 47345, 11 pages. doi :10.1155/2007/47345.

Conférences internationales avec comité de lecture

C. Duran-Faundez and V. Lecuire, *Error Resilient Image Communication with Chaotic Pixel Interleaving for Wireless Camera Sensors*. Proceedings of the 2008 Workshop on Real-World Wireless Sensor Networks (REALWSN'08), Glasgow, Scotland, 2008.

V. Lecuire, C. Duran-Faundez, T. Holl, N. Krommenacker, M. Maimour, and M. David, *POSTER : Transmission d'images avec préservation de l'énergie dans les réseaux de capteurs sans fil*. 12ème Colloque Francophone sur l'Ingénierie des Protocoles, Tozeur, Tunisie, octobre 2006.

V. Lecuire, C. Duran-Faundez, T. Holl, N. Krommenacker, M. Maimour, and M. David, *Energy consumption analysis of a simple image transmission protocol in wireless sensor networks*. 6th IEEE International Workshop on Factory Communication Systems (WFCS'2006), pages 215-218, Torino, Italy, 2006.

Groupes de travail nationaux

C. Duran-Faundez and V. Lecuire, *Compression et transmission d'images sur réseau de capteurs sans fil sous la contrainte de l'énergie*, Journées non thématiques ResCom, Strasbourg, octobre, 2008.

C. Duran-Faundez and V. Lecuire, *Transmission adaptative d'images avec contrôle de l'énergie pour les réseaux de capteurs sans fil*, 2èmes Journées Doctorales / Journées Nationales MACS, JD-JN-MACS, Reims, août, 2007.

C. Duran-Faundez and V. Lecuire, *Transmission adaptative d'images avec contrôle de l'énergie pour les réseaux de capteurs sans fil*, groupe de travail Réseaux Grand Est, RGE, Montbéliard, juin 2007.

C. Duran-Faundez and V. Lecuire, *Protocole de transmission d'images avec contrôle de l'énergie pour les réseaux de capteurs sans fil*, 8ème Journées Doctorales en Informatique et Réseaux, JDIR'2007, Marne-la-Vallée, janvier 2007.

Bibliographie

Aboelaze, Mokhtar et Fadi Aloul (2005). Current and future trends in sensor networks : A survey. In : *Second IFIP International Conference on Wireless and Optical Communications Networks (WOCN 2005)*. pp. 551–555.

Abramson, Norman (1970). THE ALOHA SYSTEM - another alternative for computer communications. In : *Proceedings of Fall Joint Computer Conference, AFIPS Conference*. pp. 281–285.

Akkaya, Kemal et Mohamed Younis (2003). An energy-aware qos routing protocol for wireless sensor networks. In : *Proceedings of the IEEE Workshop on Mobile and Wireless Networks (WMN 2003)*. Providence, Rhode Island.

Akkaya, Kemal et Mohamed Younis (2005). A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, **3**(3), 325–349.

Akyildiz, Ian F., Tommaso Melodia et Kaushik R. Chowdhury (2007). A survey on wireless multimedia sensor networks. *Computer Networks*, **51**(4), 921–960.

Akyildiz, I.F., W. Su, Y. Sankarasubramaniam et E. Cayirci (2002). Wireless sensor networks : A survey. *Computer Networks*, **38**(4), 393–422.

Arampatzis, Th., J. Lygeros et S. Manesis (2005). A survey of applications of wireless sensors and wireless sensor networks. In : *Proceedings of the 2005 IEEE International Symposium on Intelligent Control, Mediterrean Conference on Control and Automation*. pp. 719–724.

Arici, Tarik, Bugra Gedik, Yucel Altunbasak et Ling Liu (2003). PINCO : A pipelined in-network compression scheme for data collection in wireless sensor networks. In : *Proceedings of 12th International Conference on Computer Communications and Networks*.

Arora, A., P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora et M. Miyashita (2005). A line in the sand : A wireless sensor network for targer detection, classification, and tracking. *IEEE Computer Networks*, **46**(5), 605–634.

ATmega128(L) Summary (n.d.). Datasheet. Atmel Corporation. http ://www.atmel.com/.

Bae, Jaewook et Richard M. Voyles (2006). Wireless video sensor networks over bluetooth for a team of urban search and rescue robots. In : *Proceedings of the 2006 International Conference on Wireless Networks*. Las Vegas, NV.

Belongie, Serge, Jitendra Malik et Jan Puzicha (2002). Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(24), 509–522.

Biagioni, Edoardo S. et K. W. Bridges (2002). The application of remote sensor technology to assist the recovery of rare and endangered species. *International Journal of High Performance Computing Applications*.

Boeglen, Hervé, Christian Chatellier, Christian Olivier et Olivier Haeberle (2007). Un système de transmission d'images fixes robuste pour canaux radiomobiles sélectifs en temps et en fréquence. In : *12èmes journées d'étude et d'échange COmpression et REprésentation des Signaux Audiovisuels, CORESA 2007*. Montpellier, France.

Calderbank, A. R., Ingrid Daubechies, Wim Sweldens et Boon-Lock Yeo (1998). Wavelet transforms that map integers to integers. *Applied and Computational Harmonic Analysis (ACHA)*, **5**(3), 332–369.

Cao, Zhi-Yan, Zheng-Zhou Ji et Ming-Zeng Hu (2005). An image sensor node for wireless sensor networks. In : *International Conference on Information Technology : Coding and Computing (ITCC'05)*. Vol. 2. pp. 740–745.

Cardei, Mihaela, Shuhui Yang et Jie Wu (2008). Algorithms for fault-tolerant topology in heterogeneous wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, **19**(4), 545–558.

Center of Embedded Network Sensing (2004). CENS - CVS Repository. `http://cvs.cens.ucla.edu/`.

Chang, Jae-Hwan et Leandros Tassiulas (2000). Maximum lifetime routing in wireless sensor networks. In : *Proceedings of the Advanced Telecommunications and Information Distribution Research Program (ATIRP 2000)*. College Park, MD.

Chen, Tung-Shou, Jeanne Chen et Jian-Guo Chen (2003). A simple and efficient watermarking technique based on jpeg2000 codec. In : *Proceedings of the Fifth International Symposium on Multimedia Software Engineering*. pp. 80–87.

Cho, K., A. Krymski et E. Fossum (2003). A 1.5-v 550-μw 176×144 autonomous cmos active pixel image sensor. *IEEE Transactions on Electron Devices*, **50**(1), 96–105.

Choi, Kyung Jun et Jong-In Song (2008). A miniaturized mote for wireless sensor networks. In : *10th International Conference on Advanced Communication Technology (ICACT'08)*. Gangwon-do, South Korea.

Chow, Kit-Yee, King-Shan Lui et Edmund Y. Lam (2006). Balancing image quality and energy consumption in visual sensor networks. In : *International Symposium on Wireless Pervasive Computing (ISWPC'06)*. Phuket, Thailand. pp. 1–5.

Chow, Kit-Yee, King-Shan Lui et Edmund Y. Lam (2007). Efficient on-demand image transmission in visual sensor networks. *EURASIP Journal on Advances in Signal Processing*, **2007**, Article ID 95076, 11 pages. doi :10.1155/2007/95076.

Christopoulos, C. A., T. Ebrahimi et A. N. Skodras (2000). JPEG2000 : the new still picture compression standard. In : *Proceedings of the 2000 ACM workshops on multimedia*. ACM Press. Los Angeles, California, United States. pp. 45–49.

Chuang, Shun-Yu et Chien Chen (2007). Smartbone : An energy-efficient smart backbone construction in wireless sensor networks. *Journal of Information Schence and Engineering*, **23**, 1023–1039.

Crossbow Technology Inc. (n.d.). http ://www.xbow.com/.

Culurciello, Eugenio et Andreas G. Andreou (2004). ALOHA CMOS imager. In : *IEEE International Symposium on Circuits and Systems (ISCAS)*. Vancouver, Canada. pp. IV–956–9.

Culurciello, Eugenio et Andreas G. Andreou (2006). CMOS image sensors for sensor networks. *Analog Integrated Circuits and Signal Processing*, **49**(1), 39–51.

Culurciello, Eugenio, Ralph Etienne-Cummings et Kwabena Boahen (2003). A biomorphic digital image sensor. *IEEE Journal of Solid-State Circuits*, **38**(2), 281–294.

Dai, Shijin, Xiaorong Jing et Lemin Li (2005). Research and analysis on routing protocols for wireless sensor networks. In : *Proceedings. 2005 International Conference on Communications, Circuits and Systems*. Vol. 1. pp. 407–411.

DeBrunner, Victor, Linda DeBrunner et Longji Wang (1999). Recovery of lost blocks by dynamic pixel interleaving. In : *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems (ISCAS'99)*. Vol. 4. IEEE. pp. 131–134.

Doolin, David M. et Nicholas Sitar (2005). Wireless sensors for wildfire monitoring. In : *Proceedings of SPIE Symposium on Smart Structures & Materials/ NDE 2005*. San Diego, California.

Downes, Ian, Leili Baghaei Rad et Hamid Aghajan (2006). Development of a mote for wireless image sensor networks. In : *Proceedings of Cognitive Systems and Interactive Sensors (COGIS 2006)*.

Feng, Wu-Chi, Ed Kaiser, Wu-Chang Feng et Mickael Le Baillif (2005*a*). Panoptes : Scalable low-power video sensor networking technologies. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, **1**(2), 151–167.

Feng, Wuchi, Brian Code, Ed Kaiser, Mike Shea et Wuchang Feng (2005*b*). Panoptes : A scalable architecture for video sensor networking applications. *ACM Transactions on Multimedia Computing, Communications and Applications*.

Ferrigno, L., S. Marano, V. Paciello et A. Pietrosanto (2005). Balancing computational and transmission power consumption in wireless image sensor networks. In : *IEEE International Conference on Virtual Environments, Human-Computer Interfaces, and Measures Systems (VECIMS 2005)*. Giardini Naxos, Italy.

Fischer, T.R. (1986). A pyramid vector quantizer. *IEEE Transactions on Information Theory*, **32**, 568–583.

Gaudeau, Y., L. Guillemot et JM. Moureaux (2008). Fast dead zone lattice vector quantization. *IET Electronics Letters*, **44**(3), 191–192.

Gauger, Matthias, Daniel Minder, Pedro Jose Marron, Arno Wacker et Andreas Lachenmann (2008). Prototyping sensor-actuator networks for home automation. In : *Workshop on Real-World Wireless Sensor Networks, REALWSN'08*. ACM. Glasgow, Scotland.

Gehrig, Nicolas et Pier Luigi Dragotti (2004). Distributed compression in camera sensor networks. In : *Proceedings of IEEE International Workshop on Multimedia Signal Processing (MMSP)*. Siena, Italy.

Gortz, Manuel, Ralf Ackermann, Johannes Schmitt et Ralf Steinmetz (2004). Context-aware communication services : A framework for building enhanced IP telephony services. In : *International Conference on Computer Communications and Networks (ICCCN) 2004*. pp. 535–540.

Guillemot, L., Y. Gaudeau, S. Moussaoui et JM. Moureaux (2008). Entropy-coded lattice vector quantization dedicated to the block mixture densities. *IEEE Transactions on Image Processing*, **17**(9), 1574–1586.

Harvard Sensor Networks Lab (2004 - 2008). Volcano monitoring. `http://fiji.eecs.harvard.edu/Volcano`.

He, Tian, Pascal A. Vicaire, Ting Yan, Liqian Luo, Lin Gu, Gang Zhou, Radu Stoleru, Qing Cao, John A. Stankovic et Tarek Abdelzaher (2007). Achieving real-time target tracking using wireless sensor networks. *ACM Transaction on Embedded Computing System (TECS)*.

Hedetniemi, S. et A. Liestman (1988). A survey of gossiping and broadcasting in communication networks. *Networks*, **18**(4), 319–349.

Hengstler, Stephan, Daniel Prashanth, Sufen Fong et Hamid Aghajan (2007). MeshEye : A hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. In : *Proceedings of the 6th international conference on Information processing in sensor networks (IPSN-SPOTS)*. ACM Press. Cambridge, Massachusetts, USA. pp. 360–369.

Hengstler, Stephan et Hamid Aghajan (2006). A smart camera mote architecture for distributed intelligent surveillance. In : *ACM SenSys Workshop on Distributed Smart Cameras (DSC)*.

Hill, Jason et David Culler (2002*a*). A wireless embedded sensor architecture for system-level optimization. Technical report. UC Berkeley.

Hill, Jason L. et David E. Culler (2002*b*). Mica : A wireless platform for deeply embedded networks. *IEEE Micro*, **22**(6), 12–24.

Holl, T., V. Lecuire et J-M. Moureaux (2005). Transport à fiabilité partielle d'images compressées sur internet. In : *11ème Colloque Francophone sur l'Ingénierie des Protocoles, CFIP'2005*. Bordeaux, France.

Horn, Berthold Klaus Paul (1986). *Robot Vision*. The MIT Press. Cambridge, Massachusetts.

Industrial Technologies Program - U.S. Department of Energy (2002). Industrial wireless technology for the 21st century. `http://www1.eere.energy.gov/industry/sensors_automation/pdfs/wireless_technology.pdf`. Last access to pdf file : november 20th, 2008.

Intanagonwiwat, Chalermek, Ramesh Govindan et Deborah Estrin (2000). Directed diffusion : A scalable and robust communication paradigm for sensor networks. In : *Sixth ACM/IEEE International Conference on Mobile Computing and Networks*. Boston (MA). pp. 56–67.

Jeong, D.G. et J.D. Gibson (1993). Uniform and piecewise uniform lattice vector quantization for memoryless gaussian and laplacian sources. *IEEE Transactions on Information Theory*, **39**, 786–804.

Kahn, J. M., R. H. Katz et K. S. J. Pister (1999). Next century challenges : Mobile networking for "smart dust". In : *Proceedings of the ACM MobiCom'99*.

Kahn, Joseph M., Randy Howard Katz et Kristofer S. J. Pister (2000). Emerging challenges : Mobile networking for "smart dust". *Journal of Communications and Networks*, **2**(3), 188–196.

Karl, Holger et Andreas Willig (2005). *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, Ltd.

Karlsson, Johannes, Tim Wark, Philip Valencia, Michael Ung et Peter Corke (2007). Demonstration of image compression in a low-bandwidth wireless camera network. In : *Proceedings of the 6th international conference on Information processing in sensor networks (IPSN 2007)*. ACM Press. Cambridge, Massachusetts, USA. pp. 557–558.

Kim, Moonseong, Euihoon Jeong, Young-Cheol Bang, Soyoung Hwang, Changsub Shin, Gwang-Ja Jin et Bongsoo Kim (2008). An energy-aware multipath routing algorithm in wireless sensor networks. *IEICE Transactions on Information and Systems*, **E91-D**(10), 2419–2427.

Kimura, Naoto et Shahram Latifi (2005). A survey on data compression in wireless sensor networks. In : *International Conference on Information Technology : Coding and Computing (ITCC 2005)*. Vol. 2. pp. 8–13.

Kleihorst, Richard, Ben Schueler, Alexander Danilin et Marc Heijligers (2006). Smart camera mote with high performance vision system. In : *ACM SenSys 2006 Workshop on Distributed Smart Cameras (DSC 2006)*.

Köppe, Enrico, Achim Liers, Hartmut Ritter et Jochen Schiller (2004). Low-power image transmission in wireless sensor networks using ScatterWeb technology. In : *First Workshop on Broadband Advanced Sensor Networks (BaseNets 2004)*. San Jose, CA.

Kotay, Keith, Ron Peterson et Daniela Rus (2005). Experiments with robots and sensor networks for mapping and navigation. In : *International Conference on Field and Service Robotics*. Port Douglas, Australia.

Krishnamurthy, Lakshman, Robert Adler, Phil Buonadonna, Jasmeet Chhabra, Mick Flanigan, Nandakishore Kushalnagar, Lama Nachman et Mark Yarvis (2005). Design and deployment of industrial sensor networks : Experiences from a semiconductor plant and the north sea. In : *3rd international Conference on Embedded Networked Sensor Systems*. pp. 64–75.

Kulkarni, Purushottam, Deepak Ganesan et Prashant Shenoy (2005*a*). The case for multi-tier camera sensor networks. In : *Proceedings of the Fifteenth International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*. Stevenson, WA, USA.

Kulkarni, Purushottam, Deepak Ganesan, Prashant Shenoy et Qifeng Lu (2005*b*). Senseye : A multi-tier camera sensor network. In : *Proceedings of ACM Multimedia*. Singapore.

Kusuma, J., L. Doherty et K. Ramchandran (2001). Distributed compression for sensor networks. In : *Proceedings of 2001 International Conference on Image Proceesing*.

Lecuire, Vincent et Francis Lepage (1999). Proposition d'un protocole de transport à fiabilité partielle déterministe. In : *Colloque Francophone sur Ingénierie des Protocoles (CFIP'99)*. Nancy, France. pp. 183–198.

Lee, Dong-Gi et Sujit Dey (2002). Adaptive and energy efficient wavelet image compression for mobile multimedia data services. In : *IEEE International Conference on Communications (ICC 2002)*. Vol. 4. pp. 2484–2490.

Lee, Dong-U, Hyungjin Kim, Steven Tu, Mohammad Rahimi, Deborah Estrin et John D. Villasenor (2007*a*). Energy-optimized image communication on resource-constrained sensor platforms. In : *IPSN'07*. Cambridge, Massachusetts, USA.

Lee, Seungjoon, Dave Levin, Vijay Gopalakrishnan et Bobby Bhattacharjee (2007*b*). Backbone construction in selfish wireless networks. In : *SIGMETRICS'07*. San Diego, California, USA.

Lifton, Joshua, Deva Seetharam, Michael Broxton et Joseph Paradiso (2002). Pushpin computing system overview : A platform for distributed, embedded, ubiquitous sensor networks. In : *Proceedings of the Pervasive Computing Conference*. Zurich, Switzerland.

Loeffler, Christoph, Adriaan Lieenberg et George S. Moschytz (1989). Practical fast 1-D DCT algorithms with 11 multiplications. In : *International Conference on Acoustics, Speech, and Signal Processing (ICASSP'89)*. Vol. 2. Glasgow, UK. pp. 988–991.

Louberry, Christine, Philippe Roose et Marc Dalmau (2007). Towards sensor integration into multimedia applications. In : *4th European Conference on Universal Multiservice Networks ECUMN'2007*. Toulouse, France. pp. 355–363.

Lu, Qin, Wusheng Luo, Jidong Wang et Bo Chen (2008). Low-complexity and energy efficient image compression scheme for wireless sensor networks. *Computer Networks*, **52**, 2594–2603.

Lui, King-Shan et Edmund Y. Lam (2005). Image transmission in sensor networks. In : *IEEE Workshop on Signal Processing Systems Design and Implementation*. pp. 726–730.

Lymberopoulos, Dimitrios et Andreas Savvides (2005). XYZ : A motion-enabled, power aware sensor node platform for distributed sensor network applications. In : *Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*. pp. 449–454.

Maimour, Moufida (2007). Multipath routing protocol for layered video transport in wireless sensor networks. In : *7th International Conference on New Technologies of Distributed Systems (NOTERE 2007)*. Marrakesh, Marocco.

Maimour, Moufida (2008). Maximally radio-disjoint multipath routing for wireless multimedia sensor networks. In : *Fourth ACM International Workshop on Wireless Multimedia Networking and Performance Modeling, WMuNeP'08*. Vancouver, Canada.

Maimour, Moufida, Congduc Pham et Julien Amelot (2008). Load repartition for congestion control in multimedia wireless sensor networks with multipath routing. In : *Proceeding of the IEEE International Symposium on Wireless Pervasive Computing*. Santorini, Greece.

Mainwaring, Alan, Joseph Polastre, Robert Szewczyk, David Culler et John Anderson (2002). Wireless sensor networks for habitat monitoring. In : *2002 ACM International Workshop on Wireless Sensor Networks and Applications. WSNA '02*. Atlanta GA.

Mallat, Stephane (1999). *A Wavelet Tour of Signal Processing*. 2nd ed.. Academic Press.

Mammeri, Abdelhamid, , Ahmed Khoumsi, Djemel Ziou et Brahim Hadjou (2008). Modeling and adapting JPEG to the energy requirements of VSN. In : *Proceedings of 17th International Conference on Computer Communications and Networks ICCCN '08*. US Virgin Islands.

Maniezzo, D., K. Yao et G. Mazzini (2002). Energetic trade-off between computing and communication resource in multimedia surveillance sensor network. In : *4th IEEE International Workshop on Mobile and Wireless Communications Network (MWCN 2002)*. Stockholm, Sweden.

Margi, Cintia B., Vladislav Petkov, Katia Obraczka et Roberto Manduchi (2006). Characterizing energy consumption in a visual sensor network testbed. In : *Proceedings of the 2nd International IEEE/Create-Net Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom 2006)*.

Marhur, Gaurav, Peter Desnoyers, Deepak Ganesan et Prashant Shenoy (2006). Ultra-low power data storage for sensor networks. In : *Proceedings of IEEE/ACM Conference on Information Processing in Sensor Networks*. Nashville, TN.

Marino, Francescomaria, Vincenzo Piuri et Jr. Earl E. Swartzlander (1999). A parallel implementation of the 2-D discrete wavelet transform without interprocessor communications. *IEEE Transactions on Signal Processing*, **47**(11), 3179–3184.

McCormick, Chris, Pierre-Yves Laligand, Huang Lee et Hamid Aghajan (2006). Distributed agent control with self-localizing wireless image sensor networks. In : *Proceedings of Cognitive Systems and Interactive Sensors (COGIS 2006)*.

McCulloch, John, Siddeswara Mayura Guru et Daniel Hugo (2008). Wireless sensor network deployment for water use efficiency in irrigation. In : *Workshop on Real-World Wireless Sensor Networks, REALWSN'08*. Glasgow, Scotland.

McIlrath, L. (2001). A low-power low-noise ultrawide-dynamic-range cmos imager with pixel-parallel a/d conversion. *IEEE Journal of Solid-State Circuits*, **36**(5), 846–853.

Misra, Satyajayant, Martin Reisslein et Guoliang Xue (n.d.). A survey of multimedia streaming in wireless sensor networks. accepted for publication in IEEE Communications Surveys and Tutorials.

Na, Chewoo, Yaling Yang et Amitabh Mishra (2008). An optimal gts scheduling algorithm for time-sensitive transactions in ieee 802.15.4 networks. *Computer Networks*, **52**, 2543–2557.

Nachman, L., R. Kling, R. Adler, J. Huang et V. Hummel (2005). The intel mote platform : a bluetooth-based sensor network for industrial monitoring. In : *Fourth International Symposium on Information Processing in Sensor Networks, 2005. IPSN 2005.*. number 15. pp. 437–442.

Naumowicz, Tomasz, Robin Freeman, Andreas Heil, Martin Calsyn, Eric Hellmich, Alexander Brandle, Tim Guilford et Jochen Schiller (2008). Autonomous monitoring of vulnerable habitats using a wireless sensor network. In : *Workshop on Real-World Wireless Sensor Networks, REALWSN'08*. Glasgow, Scotland.

Oh, Songhwai, Phoebus Chen, Michael Manzo et Shankar Sastry (2006). Instrumenting wireless sensor networks for real-time surveillance. In : *Proc. of the International Conference on Robotics and Automation*.

Parisis, A., P. Carré, C. Fernandez-Maloigne et N. Laurent (2004). Tatouage d'images couleur avec adaptation locale des forces de marquage. In : *Compression et Représentation des Signaux Audiovisuels (CORESA 2004)*. Lille, France.

Petrovic, Dragan, Rahul C. Shah, Kannan Ramchandran et Jan Rabaey (2003). Data funneling : Routing with aggregation and compression for wireless sensor networks. In : *Proceedings of First IEEE International Workshop on Sensor Network Protocols and Applications*.

Polastre, Joseph, Jason Hill et David Culler (2004). Versatile low power media access for wireless sensor networks. In : *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys)*.

Polastre, Joseph, Robert Szewczyk et David Culler (2005). Telos : Enabling ultra-low power wireless research. In : *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks : Special track on Platform Tools and Design Methods for Network Embedded Sensors (IPSN/SPOTS)*.

Pradhan, S. Sandeep, Julius Kusuma et Kannan Ramchandran (2002). Distributed compression in a dense microsensor network. *IEEE Signal Proceesing Magazine*, **19**(2), 51–60.

Rahimi, Mohammad, Deborah Estrin, Rick Baer, Henry Uyeno et Jay Warrior (2004). Cyclops : Image sensing and interpretation in wireless networks. In : *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. ACM Press.

Rahimi, Mohammad, Rick Baer, Obimdinachi I. Iroezi, Juan C. Garcia, Jay Warrior, Deborah Estrin et Mani Srivastava (2005). Cyclops : In situ image sensing and interpretation in wireless sensor networks. In : *Proceedings of the 3rd ACM Conference on Embedded Networked Sensor Systems (SenSys 2005)*. San Diego, CA. pp. 192–204.

Rajendran, Venkatesh, Katia Obraczka et J. J. Garcia luna aceves (2003). Energy-efficient, collision-free medium access control for wireless sensor networks. In : *Proceedings of ACM Sensys 03*. ACM Press. Los Angeles, CA. pp. 181–192.

Ramamurthy, Harish, B. S. Prabhu, Rajit Gadh et Asad M. Madni (2005). Smart sensor platform for industrial monitoring and control. In : *4th IEEE SENSORS*. Irvine, California.

Ramamurthy, Harish, B. S. Prabhu, Rajit Gadh et Asad M. Madni (2007). Wireless industrial monitoring and control using a smart sensor platform. *IEEE Sensors Journal*, **7**(5), 611–618.

Rowe, Anthony, Adam Goode, Dhiraj Goel et Illah Nourbakhsh (2007). CMUcam3 : An open programmable embedded vision sensor. Technical Report RI-TR-07-13. Carnegie Mellon Robotics Institute. Pittsburgh, Pennsylvania 15213.

Sadler, Christopher M. et Margaret Martonosi (2006). Data compression algorithms for energy-constrained devices in delay tolerant networks. In : *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys) 2006*.

Said, Amir et William A. Pearlman (1996). A new, fast and efficient image-codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and System for Video Technology*, **6**(3), 243–250.

Salles, Nicolas, Nicolas Krommenacker et Vincent Lecuire (2008). Performance study of IEEE 802.15.4 for industrial maintenance applications. In : *IEEE International Conference on Industrial Technology, ICIT2008*. Chengdu, China.

Salomon, David (2004). *Data Compression : The Complete Reference*. 3rd edition ed.. Springer Verlag New York, Inc.

Savidge, Laura, Huang Lee, Hamid Aghajan et Andrea Goldsmith (2005). QoS-based geographic routing for event-driven image sensor networks. In : *Proceedings of the International Conference on Broadband Networks*. Boston, MA, USA.

Savidge, Laura, Huang Lee, Hamid Aghajan et Andrea Goldsmith (2006). Event-driven geographic routing for wireless image sensor networks. In : *Proceedings of Cognitive Systems and Interactive Sensors (COGIS 2006)*. Paris.

Savvides, Andreas et Mani B. Srivastava (2002). A distributed computation platform for wireless embedded sensing. In : *20th International Conference on Computer Design (ICCD'02)*. Freiburg, Germany.

Saxena, Navrati, Abhishek Roy et Jitae Shin (2008). Dynamic duty cycle and adaptive contention window based qos-mac protocol for wireless multimedia sensor networks. *Computer Networks*, **52**, 2532–2542.

Schettini, R, C. Fernandez-Maloigne et S. Susstrunck (2003). Color image processing. *Pattern Recognition Letter*.

Schulz, Jens, Frank Reichenbach, Jan Blumenthal et Dirk Timmermann (2008). Low cost system for detecting leakages along artificial dikes with wireless sensor networks. In : *Workshop on Real-World Wireless Sensor Networks, REALWSN'08*. Glasgow, Scotland.

Shah, Rahul C. et Jan M. Rabaey (2002). Energy aware routing for low energy ad hoc sensor networks. In : *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*. Orlando, FL.

Shnayder, Victor, Mark Hempstead, Bor-Rong Chen, Geoff Werner Allen et Matt Welsh (2004). Simulating the power consumption of large-scale sensor network applications. In : *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*. Baltimore, MD.

Slepian, David et Jack K. Wolf (1973). Noiseless coding of correlated information sources. *IEEE Transactions on Information Theory*, **IT-19**(4), 471–480.

Srivastava, Mani, John Villasenor, Deborah Estrin et Mohammad Rahimi (n.d.). Cyclops. `http://research.cens.ucla.edu/projects/2007/Multiscaled_Actuated_Sensing/Cyclops/`. CENS, Research Project.

Stankovic, John A. (2004). Research challenges for wireless sensor networks. *SIGBED Rev.*, **1**(2), 9–12.

Stathopoulos, Thanos, Rahul Kapur, Deborah Estrin, John Heidemann et Lixia Zhang (2004). Application-based collision avoidance in wireless sensor networks. In : *Proceedings of the 29th IEEE International Conference on Local Computer Networks*. IEEE. Tampa, Flordia, USA. pp. 506–514.

Strohbach, M. (2004). The smart-its platform for embedded context-aware systems. In : *Proceedings of the First International Workshop on Wearable and Implantable Body Sensor Networks*. London, UK.

Swarm-Intelligent Systems Group (2004). Low power sensor network for collective vision. Project Summary SWIS-SP6. École Polytechnique Fédérale de Lausanne. Lausanne.

Teixeira, Thiago, Andreas G. Andreou et Eugenio Culurciello (2005). Event-based imaging with active illumination in sensor networks. In : *IEEE International Symposium on Circuits and Systems (ISCAS 2005)*. Vol. 1. pp. 644–647.

Teixeira, Thiago, Eugenio Culurciello, Joon Hyuk Park, Dimitrios Lymberopoulos, Andrew Barton-Sweeney et Andreas Savvides (2006). Address-event imagers for sensor networks : Evaluation and modeling. In : *Proceedings of Information Processing in Sensor Networks (IPSN)*.

Tezcan, Nurcan et Wenye Wang (2008). Self-orienting wireless multimedia sensor networks for occlusion-free viewpoints. *Computer Networks*, **52**, 2558–2567.

Thorn, Jeff (2005). Deciphering tinyos serial packets. Technical Report Octave Tech Brief 5-01. Octave Technology.

Turner, Charles J. et Larry L. Peterson (1992). Image transfer : an end-to-end design. In : *Proceedings of the SIGCOMM '92 Symposium*. ACM Press. Baltimore, Maryland. pp. 258–268.

UC Berkeley (n.d.). TinyOS : An operating system for networked sensors. `http://www.tinyos.net/`.

UVA Department of Computer Science (2005-2007). Assisted-living and residential monitoring network – a wireless sensor network for smart healthcare. `http://www.cs.virginia.edu/wsn/medical/`.

van Dam, Tijs et Koen Langendoen (2003). An adaptive energy-efficient mac protocol for wireless sensor networks. In : *1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*. Los Angeles, Calif, USA. pp. 171–180.

Vieira, Marcos Augusto M., Claudionor N. Coelho Jr., Diógenes Cecílio Da Silva Junior et José M. Da Mata (2003). Survey on wireless sensor network devices. In : *IEEE Conference Emerging Technologies and Factory Automation (ETFA '03)*. Vol. 1. pp. 537–544.

Voyatzis, G. et I. Pitas (1996). Chaotic mixing of digital images and applications to watermarking. In : *Proceedings of European Conference on Multimedia Applications, Services and Techniques (ECMAST'96)*. Vol. 2. Louvain-la-Neuve, Belgium. pp. 687–695.

Wagner, Raymond, Robert Nowak et Richard Baraniuk (2003). Distributed image compression for sensor networks using correspondence analysis and super-resolution. In : *Proceedings of 2003 International Conference on Image Processing (ICIP)*. Vol. 1. pp. 597–600.

Wawerla, Jens, Shelley Marshall, Greg Mori, Kristina Rothley et Payam Sabzmeydani (2008). Bearcam : Automated wildlife monitoring at the arctic circle. *Journal of Machine Vision Applications*. (to appear).

Willig, A., M. Kubisch, C. Hoene et A. Wolisz (2002). Measurements of a wireless link in an industrial environment using an ieee 802.11-compliant physical layer. *IEEE Transaction on Industrial Electronics*, **49**(6), 1265–1282.

Woodrow, Edward et Wendi Heinzelman (2002). SPIN-IT : a data centric routing protocol for image retrieval in wireless networks. In : *International Conference on Image Processing (ICIP '02)*.

Wu, Huaming et Alhussein A. Abouzeid (2004*a*). Energy efficient distributed JPEG2000 image compression in multihop wireless networks. In : *4th Workshop on Applications and Services in Wireless Networks (ASWN 2004)*. pp. 152–160.

Wu, Huaming et Alhussein A. Abouzeid (2004*b*). Power aware image transmission in energy constrained wireless networks. In : *Proceedings of the 9th IEEE Symposium on Computers and Communications (ISCC'2004)*. Alexandria, Egypt.

Wu, Huaming et Alhussein A. Abouzeid (2005). Energy efficient distributed image compression in resource-constrained multihop wireless networks. *Computer Communications*, **28**(14), 1658–1668.

Wu, Huaming et Alhussein A. Abouzeid (2006). Error resilient image transport in wireless sensor networks. *Computer Networks*, **50**(15), 2873–2887.

Wu, Min et Chang Wen Chen (2003). Multiple bitstream image transmission over wireless sensor networks. In : *Proceedings of IEEE Sensors*. Vol. 2. pp. 727–731.

Xu, Ning, Sumit Rangwala, Krishna Kant Chintalapudi, Deepak Ganesan, Alan Broad, Ramesh Govindan et Deborah Estrin (2004). A wireless sensor network for structural monitoring. In : *Proceedings of the 2nd International Conference on Embedded Networked Systems*.

Ye, Wei, John Heidemann et Deborah Estrin (2002). An energy-efficient MAC protocol for wireless sensor networks. In : *Proceedings of the IEEE Infocom*. USC/Information Sciences Institute. IEEE. New York, NY, USA. pp. 1567–1576.

Yu, Wei, Zafer Sahinoglu et Anthony Vetro (2004). Energy efficient JPEG 2000 image transmission over wireless sensor networks. In : *IEEE Global Telecommunications Conference (GLOBECOM '04)*. Vol. 5. pp. 2738–2743.

Résumé

Parmi les nombreuses applications potentielles des réseaux de capteurs sans fil, celles utilisant des capteurs d'image sont appréciables pour tout ce qui concerne la détection, la reconnaissance et la localisation d'objets par la vision. Des capteurs de petite taille, peu gourmands en énergie et dotés d'une caméra existent déjà au stade de prototype, mais des algorithmes de traitement et de compression de données, ainsi que des protocoles de communication de faible complexité et peu coûteux en énergie doivent être développés pour que ces applications puissent être envisagées en pratique.
La contribution de cette thèse porte principalement sur deux aspects. Premièrement, nous avons proposé un protocole de transmission d'images semi-fiable pour réduire la consommation d'énergie des noeuds relayant les paquets jusqu'au collecteur. Les économies d'énergie sont obtenues en préparant à la source des paquets de différences priorités, grâce à une transformée en ondelettes de l'image, puis en conditionnant l'acheminement des paquets, saut par saut, suivant leur priorité et l'état de charge des batteries. Deuxièmement, nous avons étudié plus profondément les aspects de traitement et codage d'images à la source, et nous avons proposé un nouvel algorithme de compression d'images de faible complexité, combiné avec une technique d'entrelacement de pixels basée sur les automorphismes toriques. Des expérimentations sur une plate-forme réelle de réseau de capteurs d'images ont été réalisées afin de démontrer la validité de nos propositions, en mesurant des aspects telles que la quantité de mémoire requise pour l'implantation logicielle de nos algorithmes, leur consommation d'énergie et leur temps d'exécution, ainsi que la qualité des images reconstituées au récepteur en présence de pertes de paquets.

Mots clés

Réseaux de capteurs sans fil, communication d'images, conservation de l'énergie

Abstract

Among the many potential applications of wireless sensor networks, those using image sensors are valuable for everything concerning the detection, recognition and locating objects by sight. Sensors small, less energy and with a camera already exist in prototype stage, but processing algorithms and data compression and communication protocols of low complexity and low expensive energy should be developed for these applications can be envisaged in practice.
The contribution of this thesis focuses on two aspects. First, we have proposed a protocol for transmitting images semi-reliable to reduce energy consumption nodes relaying packets until collector. Energy savings are achieved in preparing for the source of differences in priorities, with a wavelet transform of the image, then a condition for delivery of packages, jump by jump, according to their priority and status charging batteries. Secondly, we looked deeper aspects of processing and image coding to the source, and we have proposed a new compression algorithm of images of low complexity, combined with a technique of interlacing of pixels based on automorphism rings . Experiments on a platform of real network image sensors have been conducted to demonstrate the validity of our proposals, by measuring aspects such as the amount of memory required for the implementation of our software algorithms, their consumption energy and their execution time, and the quality of reconstructed images to a receiver in the presence of lost packages.

Keywords

Wireless sensor networks, image communication, energy conservation

Université Henri Poincaré, Centre de Recherche en Automatique de Nancy

Faculté des Sciences, B.P. 70239, 54506 Vandœuvre lès Nancy