وزارة التعليم العالي والبحث العلمي

BADJI MOKHTAR- ANNABA UNIVERSITY
UNIVERSITE BADJI MOKHTAR ANNABA

جامعة باجي مختار- عنابة

Année : 2019

Faculté: Sciences de l'Ingéniorat
Département: Electronique

MEMOIRE

Intitulé:

Compression d'images faible coût et transmission robuste aux erreurs avec un Raspberry PI3

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

Domaine : Sciences et Technologie

Filière : Electronique

Spécialité: Electronique embarquée

Par :

AZARNIA Fatma Zohra

DEVANT Le JURY

Président : M. BOUGHAZI Mohamed	**Professeur**	**UBM Annaba**
Directeur de mémoire : M. DOGHMANE Noureddine	**Professeur**	**UBM Annaba**
Examinateurs1: M. HAFS Toufik	**MCB**	**UBM Annaba**
Examinateurs2 : Mme ZERMI Narima	**MCB**	**UBM Annaba**

ملخص

في هذا المشروع ، تخصص الإلكترونيات المدمجة ، نحن مهتمون بدراسة وتنفيذ منصة تمثل عقدين من أجهزة استشعار الفيديو اللاسلكية لالتقاط ونقل الصور بتكلفة منخفضة الطاقة. الغرض من هذا الإنجاز هو تطبيق طريقة بسيطة وفعالة لمراقبة بعض الطيور المائية المهددة بالانقراض في منطقة عنابة. لقد اخترنا استخدام وحدة Raspberry PI3 القائمة على الحلول والتي يكون أدائها متعددًا. تم اختبار النظام الأساسي الذي تم إنتاجه من وجهة نظر كفاءة الانضغاط وقوة النقل والتكلفة الحسابية.

Résumé

Dans ce projet de fin d'études master, option électronique embarquée, nous nous sommes intéressés à l'étude et la mise en œuvre d'une plateforme représentant deux nœuds capteurs visuels sans fil pour le captage et la transmission d'images avec faible coût énergétique. La finalité de cette réalisation est de mettre en œuvre un moyen simple et efficace permettant de surveiller certains oiseaux d'eau menacés d'extinction dans la région d'Annaba. Nous avons choisi d'utiliser une solution à base de module Raspberry PI3 dont les performances sont nombreuses. La plateforme ainsi réalisée a été testée de point de vue efficacité de compression, robustesse de transmission et coût calculatoire.

Mots clés : WVSN, compression d'image, Rapberry PI3, WiFi, Nœud capteur, PSNR

Abstract

In this master thesis, embedded electronics option, we are interested in the study and the implementation of a platform representing two wireless visual sensor nodes for the capture and transmission of images with low cost energy. The purpose of this achievement is to implement a simple and effective way to monitor some endangered waterbirds in the Annaba region. We chose to use a solution based Raspberry PI3 module whose performance is numerous. The testbed thus produced was tested from the point of view of compression efficiency, transmission robustness and computational cost.

Keywords: WVSN, image coding, Raspberry PI3, WiFi, Sensor Node, PSNR

LISTE DES FIGURES

Figure **page**

Figure 1 : Système de reconnaissance automatique d'images d'oiseaux à l'aide de WSN 09
Figure 2 : Exemple de structure d'un nœud de capteur sans fil 10
Figure 3 : Solutions existantes pour l'implémentation d'un capteur scalaire/vidéo 15
Figure 4 : Raspberry Pi 2 model B 15
Figure 5 : Raspicam V2.1 16
Figure 6 : camera pi V2.1 8Mp 18
Figure 7 : Diagramme représentant les interactions entre les composants du Raspberry 19
Figure 8 : Certaines des nouveautés incluses dans le Raspberry Pi 3 20
Figure 9: Menu de démarrage du Raspberry Pi avec une carte MicroSD avec NOOBS 21
Figure 10 : Schéma du fonctionnement de la compilation croisée. 22
Figure 11: Téléchargement de Raspbian, pour commencer l'installation sur le Raspberry Pi 23
Figure 12 : Etapes pour décompresser l'image sur la carte SD 24
Figure 13 : Attendre la fin de l'écriture sur la carte SD 24
Figure 14: Bureau du Raspberry Pi 3 25
Figure 15 : Contrôle à distance d'un raspberry 26
Figure 16 : Raspberry et caméra HD 26
Figure 17: Exemple d'un script de prises de vue par Python 27
Figure 18 : Script pour le codage d'image avec le logiciel open cv (compression) 27
Figure 19 : exemple d'image acquise par rapsberry 28
Figure 20 : Architecture MQTT «publish/subscribe » 29
Figure 21 : Envoi de l'Image avec le protocole MQTT 31
Figure 22 : Aperçu sur les deux rapsberry en mode émission/réception 32
Figure 23 : Principe de fonctionnement du MQTT en mode transmission 32
Figure 24 : Schéma de principe d'un codec JPEG
Figure 25 : Exemples d'images acquises et transmises par le raspberry avec caméra
Figure 26 : Résultats de la compression des trois images en termes de PSNR vs bitrate

LISTE DES TABLEAUX

Tableau	page
Tableau 1 : Nombre d'espèces d'oiseaux par continent	03
Tableau 2 : Comparaison de différentes techniques de collecte de données	05
Tableau 3 : Raspberry vs Arduino	13
Tableau 4 : Caractéristiques de certains nœuds de capteurs sans fil	14

LISTE DES ABREVIATIONS

ADC : Analog to Digital Converter
AGC : Automatic Gain Control
CPU : Central Processing Unit
CSI : Camera Serial Interface
DCT : *Discrete Cosine Transform*
DSSS : *Direct Sequence Spread Spectrum*
FFT : Fast Fourier Transform
EIRP : Effective Isotropically Radiated Power
FPGA : Field-Programmable Gate Array
GPS : Global Position System
HD : Haute Définition
HDMI : High-Definition Multimedia Interface
HEVC : High Efficiency Video Coding
HF : Haute Fréquence
I2C : Inter-Integrated Circuit
IoT : Internet of Things
IP : Internet Protocol
ISM : Industrielles, scientifiques et médicales
IUCN: International Union for Conservation of Nature
JPEG : Joint Photographic Expert Group
QoS : Quality of Service
QoE : Quality of Experience
LoRa : Long Range
MAC : Media Access Control
MQTT : Message Queuing Telemetry Transport
MPEG : Moving Pictures Experts Group
MSE : Mean Squared Error
NiCd : Nickel-cadmium
NiMH : Nickel-hydrure métallique
NiZn : Nickel-zinc
NOx : Oxydes d'azote
OTAP : Over the Air Programming
PC : Personal Computer
PHY : Physical
PSNR : Peak Signal to Noise Ratio
RAM : Random Access Memory
RCSF : Réseaux de capteurs sans fil
RF : Radio-fréquence

RISC : Reduced instruction set computer
ROI : Region of Interest

SCCB : Serial Camera Control Bus

SD : Secure Digital
SFG : Signal Flow Graph

SRAM : Static Random Access Memory

UART: Universal asynchronous receiver-transmitter

UICN : Union internationale pour la conservation de la nature

USB: Universal Serial Bus

VGA : Video Graphics Array

VPN : Virtual Private Network

WiFi : Wireless Fidelity

WMSN : Wireless Multimedia Sensor Network

WSN : Wireless Sensor Network

WVSN: Wireless Visual Sensor Network

Sommaire

Titre — Page

Introduction générale 01

Chapitre 1 : Position du problème 02

1. Introduction 02
2. Etat de l'art 03
3. Réseaux de capteurs sans fil 06
4. Conclusion 08

Chapitre 2 : Solution adoptée 09

1. Introduction 09
2. RCSF comme solution 09
3. Equipements adoptés 11
   - 3.1 Module Raspberry Pi 15
   - 3.2 Raspberry camera v2 16
   - 3.3 Software 17
     - 3.3.1 L'OS Raspbian Jessie Avec Pixel 17
     - 3.3.2 Python 2.7 et 3.0 17
4. Conclusion 17

Chapitre 3 : La réalisation du nœud capteur 18

1. Introduction 18
2. Présentation du matériels 18
   - 2.1 Liste du matériel utilisé 18
   - 2.2 Spécification du raspberry pi 19
3. Installation/Configuration du deux Raspberries 20
   - 3.1 Choix de la distribution 20
   - 3.2 Première installation et configuration de Raspbian 22
4. Installation et configuration de Raspbian 23
5. Connexion réseau avec Putty et VNC 25
6. Utilisation de la Caméra 26
7. Transmission de données entre deux raspberry 28
   - 7.1 Présentation du protocole MQTT 28
   - 7.2 Sécurité 29
   - 7.3 Qualité de Service (QoS) 30
8. Résultats de connexion entre deux raspberry avec le MQTT 30
9. Conclusion 33

Chapitre 4 : Tests et résultats 34

1. Introduction 34
2. Compression d'image 34

3. Exemples d'expérimentations réalisées 36
4. Conclusion 38
Conclusion générale et perspectives 39
Références bibliographiques 40

Introduction générale

Introduction générale

Les oiseaux réagissent aux éventuels changements de l'environnement et constituent de bons indicateurs de la biodiversité. Si certaines espèces aviaires commencent à décliner, cela peut mettre en lumière de graves problèmes à la fois dans la chaîne alimentaire et dans la sécurité environnementale. À cette fin, la surveillance de la faune et de la flore en général et des oiseaux en particulier peut fournir de nombreuses informations précieuses sur l'environnement, la santé, la productivité des terres, etc.
La surveillance des oiseaux est pertinente pour identifier l'espèce et pour évaluer les progrès accomplis dans les efforts de sauvegarde de la biodiversité. Cela a notamment permis de définir des indicateurs, tels que la liste rouge de l'UICN [ref]. Cette liste d'espèces menacées, établie en 1964, est largement utilisée comme inventaire de l'état de conservation des espèces biologiques et permet de suivre l'évolution de ces espèces et de surveiller leur habitat naturel.

Dans ce projet, nous proposons une surveillance basée sur des réseaux de capteurs visuels sans fil (WVSN) de certains oiseaux jugés menacés d'extinction dans leur habitat naturel. L'idée est d'identifier et de reconnaître instantanément différentes espèces d'oiseaux à partir de la détection, du traitement et de la transmission de leurs images par des nœuds de capteurs sans fil. Les réseaux de capteurs se révèlent être des plates-formes idéales pour l'enregistrement et le traitement de telles données en raison de leurs caractéristiques conformes aux exigences du projet telles que l'indépendance énergétique, le coût financier moyen, la couverture géographique étendue et la préservation de l'environnement.

Une des sources de consommation les plus gourmandes concerne l'énergie de transmission entre le capteur et la passerelle. Pour cela des attentions particulières doivent être mises en œuvre afin de réduire au maximum la taille des données à transmettre (efficacité de la compression) mais aussi un choix judicieux du moyen de transmission s'impose. Nous nous sommes donc intéressés à la mise en œuvre d'un nœud capteur visuel sans fil (WVSN) à partir d'équipements simples, non onéreux, flexibles et précis. Le choix est vite fait et nous avons donc opté pour un module Raspbery PI 3 muni d'une caméra HD CSI et une transmission WiFi.

Le manuscrit est donc divisé en quatre chapitres. Le premier chapitre pose la problématique liée à ce projet et d'une manière succincte présente un état de l'art sur la surveillance des animaux dans la vie sauvage à partir de réseaux de capteurs sans fil (visuels et/ou scalaires). Le second chapitre expose la solution adoptée aussi bien de point de vue matérielle que logicielle. Le troisième chapitre est consacré à la réalisation du nœud capteur. Enfin, le dernier chapitre présente les différents tests menés, les résultats obtenus et une discussion. Pour terminer une conclusion générale résume les tout ce qui a été effectué durant ce mémoire de fin d'études et propose certaines perspectives jugées pertinentes.

Chapitre 1

Position du problème

Chapitre 1 : Position du problème

1. Introduction

Dans le monde entier, il existe environ 10 000 espèces d'oiseaux connues, avec en moyenne plusieurs dizaines à quelques centaines par pays et plusieurs milliers par continent (Tableau 1) [1]. Les oiseaux réagissent aux changements possibles de l'environnement et constituent de bons indicateurs de la biodiversité. Ils jouent également un rôle majeur dans la sécurité environnementale, par exemple:

- la consommation d'insectes nuisibles à l'agriculture,
- dispersion des graines pour le reboisement,
- pollinisation des plantes,
- …etc

Principalement, les zones humides abritent un nombre impressionnant de faune et de flore, dont certaines sont menacées d'extinction. Il existe un grand nombre de zones humides à travers le monde, parmi lesquelles le Pantanal est certainement le principal. Il couvre une superficie totale de 150000 km^2 dans trois pays d'Amérique du Sud (Brésil, Bolivie et Paraguay). Il abrite une faune et une flore très riche et nombreuse comprenant plus de 650 espèces d'oiseaux. Il existe d'autres zones humides importantes dans le monde, telles que la Camargue (France), les Everglades (États-Unis) ou l'Okavango (Botswana). L'Algérie compte plus de 250 zones humides, dont 50 sont classées au niveau international pour leur importance et leur rôle écologique. Ils sont également des lieux privilégiés pour que des dizaines de milliers d'oiseaux d'eau de différentes espèces puissent hiverner ou faire une halte temporaire. Certaines de ces espèces sont menacées d'extinction selon la dernière classification de l'Union internationale pour la conservation de la nature (UICN), telles que le canard à tête blanche (Oxyura leucocephala), le canard ferrugineux (Aythya nyroca), la sarcelle marbrée, le goéland d'Audouin, le faucon Eleonora et la sittelle Kabyle (Sitta ledanti, unique espèce endémique d'Algérie). Cela donne une grande importance aux zones humides algériennes et requiert une attention particulière.

Dans ce projet, nous proposons une surveillance basée sur WSN de ces oiseaux dans leur habitat naturel. L'idée est d'identifier et de reconnaître instantanément différentes espèces d'oiseaux à partir de la détection, du traitement et de la transmission de leurs images par des nœuds de capteurs sans fil. Les réseaux de capteurs se révèlent être des plates-formes idéales pour l'enregistrement et le traitement de telles données en raison de leurs caractéristiques conformes aux exigences du projet telles que l'indépendance énergétique, le coût financier moyen, la couverture géographique étendue et la préservation de l'environnement. Cependant, la méthode de surveillance proposée, basée sur le WSN, doit relever deux défis majeurs. D'une part, une immunité aux bruits et les interférences inévitablement présents dans les canaux de transmission utilisés par les WSN et d'autre part, une complexité informatique

réduite qui conduit à une économie d'énergie des nœuds de capteurs sans fil, augmentant ainsi leur durée de vie.

Continent	Nombre d'espèce d'oiseau
Amérique du sud	Plus de 3400
Asie	Plus de 2900
Afrique	Plus de 2300
Amérique du nord	Plus de 900
Australie	Plus de 820
Europe	Plus de 700
Antarctique	Plus de 65

Tableau 1: Nombre d'espèces d'oiseaux par continent [ ]

2. Etat de l'art

Les premières méthodes de surveillance des populations d'une espèce animale ont été sous forme de collectes manuelles d'informations sur le comportement d'individus par des observateurs sur le terrain. Les objectifs étaient souvent limités à l'étude de l'évolution du nombre de ces oiseaux et leur distribution spatiale sur le site, en observant les individus de manière discontinue (en moyenne 3 fois par mois). Cependant, dans certains cas, cette technique est sujette au problème de la différenciation des individus de la même espèce. Il devient difficile de garantir que deux individus observés à des moments différents sont identiques ou non.

La recherche manuelle d'informations sur le comportement d'individus par le biais d'observateurs de terrain a été la première méthode de collecte de données sur les populations animales. Cette technique est encore largement utilisée. Cependant, dans le cas des oiseaux, l'identification visuelle d'un individu est difficile pour différentes raisons :

- La végétation dense, caractéristique de leur habitat, rend difficile l'accès au site et à la localisation des oiseaux.
- La présence humaine peut perturber les animaux et modifier leur comportement, biaisant ainsi les informations collectées.
- La durée des campagnes d'observation est limitée et dépend de la disponibilité des équipes. Cependant, il est important de disposer de suffisamment de données réparties sur de longues périodes.

Les récentes améliorations apportées aux micro-technologies sont à l'origine de nouveaux types de dispositifs capables de fournir des données spatio-temporelles de manière continue et en temps réel. À partir du milieu des années 2000, la miniaturisation croissante de ces systèmes a permis leur utilisation dans la collecte automatique d'informations sur la nature. En

effet, des projets de recherche ont exploité les nouvelles capacités proposées par ce que l'on appelle maintenant "les capteurs portables. Par exemple, la technologie GPS est utilisée depuis plusieurs années pour suivre et identifier la position des animaux. Le problème d'identification des individus ne se pose pas puisque le capteur est porté par un individu et par un seul. Ce dispositif pourrait être utilisé dans des situations où une observation à grande échelle était impossible. Cependant, bien que le système GPS installé sur les animaux soit actuellement la méthode la plus largement utilisée, cela n'était pas faisable dans des contextes tels que celui de certains oiseaux pour les raisons suivantes:

- Il ne fonctionne efficacement que dans les zones ouvertes et ne contient aucun objet pouvant gêner les champs de vision du spectateur. Une opération sous un feuillage dense, qui constitue l'environnement de la sauvagine, n'est pas possible.
- Il est particulièrement coûteux car il est nécessaire de capturer et d'équiper tous les individus.
- Cela consomme beaucoup d'énergie
- Il a parfois un poids non négligeable, ce qui peut handicaper les mouvements des petits oiseaux.

Ces inconvénients majeurs pour l'étude des oiseaux forestiers et/ou d'eau ont conduit à d'autres dispositifs de collecte qui sont des capteurs fixes. Des capteurs fixes, tels que des nœuds de capteurs sans fil, ont déjà été utilisés dans diverses applications aussi différentes les unes que les autres. En fonction des périphériques associés aux capteurs, ce type de dispositif permet de: détecter un nombre varié d'interactions à condition qu'elles puissent être identifiées. Contrairement au GPS, ces appareils sont fixes et ne doivent pas être installés sur les individus étudiés. Ils sont généralement installés au sol ou sur des équipements fixés au sol.

Cependant, l'utilisation de ce type de dispositifs nécessite la capacité d'identifier les individus et les interactions avec précision. Typiquement, des algorithmes de traitement du signal et/ou d'analyse de séquence d'images doivent être mis en place pour permettre la reconnaissance des individus ainsi que le type d'interaction. De plus, les données collectées sur les individus ne sont pas régulières dans le temps. Les individus sont détectés seulement quand ils sont dans le champ d'action des capteurs. Par exemple, il est nécessaire de placer un grand nombre de capteurs afin de détecter le maximum de sons et de vocalisations ou encore d'images d'oiseaux et d'animaux à surveiller. Ces nœuds de capteurs sans fil doivent également pouvoir identifier, filtrer et communiquer efficacement les informations qu'ils enregistrent aux systèmes centralisés permettant de collecter et traiter d'une manière plus avancées les données.

Dans le contexte de la collecte d'informations et de la reconnaissance des individus d'une espèce animale, une méthode de collecte idéale doit pouvoir garantir certaines contraintes fonctionnelles majeures telles que le respect du comportement de l'individu et de son habitat et donc la capacité de fonctionner sous un feuillage dense. Il doit également pouvoir couvrir une zone de taille relativement grande, de l'ordre de plusieurs centaines de mètres carrés à plusieurs hectares, et pouvoir détecter efficacement les individus et leurs interactions. Une comparaison des trois familles de techniques (méthodes manuelles, équipements mobiles et capteurs fixes) par rapport à un certain nombre de critères fonctionnels définis pour l'étude. Parmi ces critères nous pouvons cités : durée de l'étude, détérioration de l'habitat, obligation de capture du spécimen, coût financier, changement du comportement, fonction sous un feuillage dense …etc. Ces critères, incontournables pour un tel projet, exclus l'utilisation des appareils mobiles. En effet, bien que les appareils mobiles permettent d'effectuer la collecte sur une longue période et ne contribuent pas à la détérioration de l'habitat, ils nécessitent la capture et l'équipement de ces animaux, ce qui peut entraîner des coûts de mise en œuvre élevés et handicaper également les animaux dans leurs déplacements. De plus, le fonctionnement en feuillage dense n'est pas garanti, ce qui peut affecter la qualité des données collectées dans le cadre d'études menées sur la sauvagine. Le tableau suivant présente une comparaison concise entre ces trois méthodes de collecte de données selon différents critères.

Critères fonctionnels	Méthode manuelle	Equipements mobiles	Capteurs fixes
Durée de l'étude	courte	**longue**	**longue**
Détérioration de l'habitat	oui	**non**	**non**
Obligation de capture	**non**	oui	**non**
Coût financier	**moyen**	élevé	**moyen**
Changement de comportement (modification)	oui	oui	**non**
Fonction sous un feuillage dense	non	non	**oui**
Identification des individus	difficile	**facile**	moyen
Type d'interactions	**multiple**	limité	**multiple**
Détection d'interactions	difficile	**facile**	moyen
Mesures en continue	non	**oui**	non
Zone d'étude	limitée	**large**	**variable**

Tableau 2 : Comparaison de différentes techniques de collecte de données par rapport à un certain nombre de contraintes fonctionnelles [ ]

3. Réseaux de capteurs sans fil RCSF (WSN)

Les réseaux de capteurs sans fil WSN sont des réseaux spontanés constitués de nœuds déployés en grand nombre en vue de rassembler et de transmettre des données vers un ou plusieurs points de collecte (sink). L'un des principaux buts des WSN est de surveiller un environnement limité en termes de ressources. Pour y parvenir plusieurs objectifs doivent être mis en œuvre. Ces objectifs comprennent la synchronisation entre les nœuds, le choix de protocoles de communication, la localisation, la gestion de la topologie, l'agrégation et le stockage des données, l'optimisation de la consommation énergétique. Les capteurs sont conçus pour fonctionner d'une manière autonome durant des mois voire des années. Ainsi, la capacité énergétique des capteurs doit être utilisée efficacement afin de maximiser la durée de vie du réseau.

Parmi les exigences de ce projet la gestion de la consommation de l'énergie. Rappelons que l'énergie consommée par un nœud capteur est due essentiellement à ses différentes opérations suivantes en l'occurrence :

- **Énergie de détection d'évènements :** C'est l'énergie consommée par un nœud capteur lors de l'activation de son unité d'acquisition et de collecte de données. Le coût de cette énergie dépend du type spécifique du capteur (image, son, température, humidité, vitesse du vent, etc.) et des tâches (mesures, conditionnement des signaux et conversion analogique-numérique, etc.) qui lui sont assignées.

- **Energie de traitement :** C'est l'énergie consommée par un nœud lors de l'activation de son unité de traitement de données (opérations, lecture/écriture en mémoire). L'énergie de traitement se divise en deux parties : l'énergie de commutation et l'énergie de fuite. L'énergie de commutation est déterminée par la tension d'alimentation et la capacité totale commutée au niveau logiciel (en exécutant un logiciel). Par contre l'énergie de fuite correspond à l'énergie consommée lorsque l'unité de calcul n'effectue aucun traitement. En général, l'énergie de traitement est faible par rapport à celle nécessaire pour la communication.

- **Energie de communication :** C'est l'énergie consommée par un nœud lors de l'activation de son unité de transmission. Des expérimentations ont montré que c'est l'étape la plus consommatrice en énergie. Il a été démontré que la transmission d'un bit d'information peut consommer autant que l'exécution de quelques milliers d'instructions.

Cette contrainte énergétique sera un des critères les plus pertinents dans le choix du protocole de transmission à adopter et aussi dans l'algorithme de traitement qui doit être implémenté

dans chaque nœud. Ceci est d'autant plus important lorsqu'il s'agit de capteurs visuels. En effet, l'acquisition d'images nécessite généralement une compression basée souvent sur des transformations linéaires orthogonales qui consomment beaucoup d'énergie due à leurs complexités calculatoires. Ceci nous oblige donc à réduire le coût calculatoire de ces algorithmes de compression en utilisant entre autres des transformations entières ou même binaires. D'autre part, l'image contient souvent des objets d'un certain intérêt (exemple l'animal à surveiller) au détriment d'autres détails et objets de moindre intérêt (exemple l'arrière plan). Il est donc intéressant de faire appel à des techniques de compression basées sur des régions d'intérêt.

Il y a également d'autres exigences à satisfaire lors du déploiement d'un WSN, parmi lesquelles :

4. la distance moyenne entre les nœuds de capteur voisins :
5. le modèle de déploiement (modèle aléatoire, régulier …etc).

La distance moyenne entre les nœuds de capteur voisins est importante pour les performances d'un réseau appliqué à la surveillance d'animaux dans leurs habitats naturels (à l'aide du captage des sons et/ou images d'une espèce animale), à la mesure de données climatiques et à la détection des feux de forêt. Cette distance inter-nœuds est d'une importance cruciale et elle joue un rôle pertinent dans :

- la localisation et la reconnaissance de l'animal à surveiller
- le délai de détection d'un départ de feu par exemple,
- …etc

A cet effet, la distance moyenne entre les nœuds de capteurs voisins doit être raccourcie. Toutefois, cela nécessite le déploiement de davantage de nœuds de capteurs dans une zone donnée et peut entraîner des problèmes de collision de canaux. Il s'agit donc d'un compromis entre la réduction du temps de détection et la probabilité de collision.

La reconnaissance automatique des d'oiseaux dans leurs habitats naturels en utilisant des WSN est soumise à de nombreuses contraintes et difficultés, par exemple:

- Les enregistrements sur le terrain sont fortement corrompus en raison de bruits environnementaux (si on utilise les sons d'oiseaux), très volumineux (s'il s'agit d'images ou vidéo)
- Les oiseaux sont généralement très éloignés de l'enregistreur (ou du nœud du capteur), de sorte que données acquises (sons ou images) sont relativement faibles ou petits.

- Interférences avec d'autres données appartenant à d'autres espèces d'oiseaux ou animaux. Un nœud capteur peut capter simultanément des données de plusieurs oiseaux de différentes espèces.
- Les canaux utilisés par les réseaux de capteurs sans fil sont souvent hostiles d'où le problème de la qualité de service qui doit être assurée.

Les WSN comme beaucoup d'autres systèmes communicants embarqués peuvent être amenés à interagir avec d'autres équipements et appareils connectés voire même des opérateurs de télécommunications afin d'atteindre des objectifs communs.

4. Conclusion

Dans ce chapitre nous avons essayé de poser la problématique qui nous a motivé pour réaliser ce projet de fin d'études. L'idée de base est de mettre en œuvre un système embarqué à faible consommation énergétique dans le but de surveiller des oiseaux menacés d'extinction dans leurs habitats naturels au niveau des zones humides d'Algérie notamment les zones humides d'Annaba et d'El taref

Chapitre 2

Solution adoptée

Chapitre 2 : Solution adoptée

1. Introduction

Il s'agit de proposer une solution matérielle sous forme d'un prototype ou plateforme représentant un ou des nœuds de capteurs sans fil sur lesquels une solution globale sera implémentée, optimisée et évaluée. Cette solution est de mettre en œuvre des nœuds de capteurs sans fil ou ''mote'' (également appelé en anglais ''node'') éventuellement hétérogènes capables de fonctionner d'une manière hiérarchique, de supporter un algorithme de compression d'images et/ou vidéo, d'effectuer éventuellement un traitement préliminaire et bien évidemment les transmettre à travers un réseau WMSN (Wireless Multimedia Sensor Network)

2. RCSF comme solution

Au cours des deux dernières décennies, les réseaux de capteurs sans fil (WSN) ont de plus en plus intéressé les communautés scientifiques et professionnelles. En effet, les perspectives d'applications et les domaines d'utilisation sont en perpétuelle augmentation: militaire, environnement, santé et sécurité. Un réseau WSN est un réseau de nœuds de capteurs qui détectent, contrôlent et transmettent informations pertinentes sur un environnement donné.

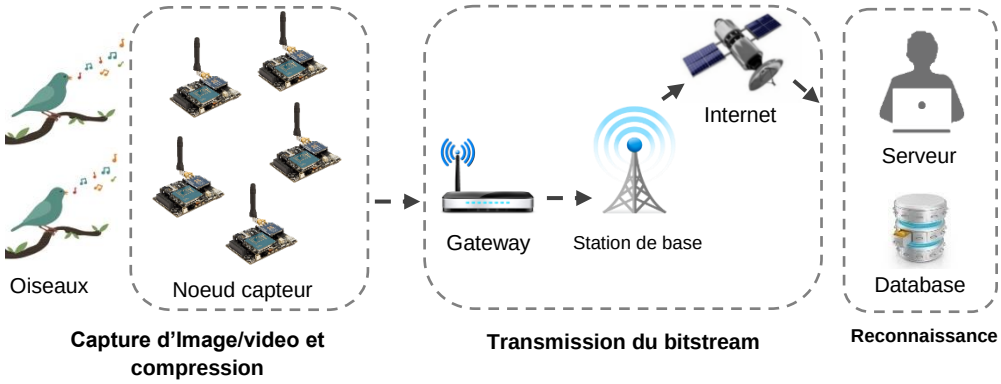

Figure 1: Système de reconnaissance automatique d'images d'oiseaux à l'aide de WSN

Les réseaux de capteurs sans fil offrent de nombreux avantages, par exemple:

- Leur facilité de déploiement,
- Ils sont utilisés dans des environnements hostiles et difficiles d'accès,
- Ils sont évolutifs,
- Ils utilisent plusieurs types de technologies de communication sans fil

Les WSN présentent également des inconvénients, notamment:

- Leur durée de vie est limitée en raison de leurs autonomies énergétiques,
- Ils ont des capacités de stockage et de calcul limitées,
- Ils ont une bande passante limitée,
- Leurs canaux de transmission sont souvent hostiles
- Peut être sujet à des attaques de sécurité

Malgré les inconvénients ci-dessus, les WSN sont largement utilisés dans de nombreux domaines et représentent aujourd'hui, entre autres, le fer de lance de l'Internet des objets (IoT : Internet of Things). Un nœud de capteur, également appelé mote, est un nœud d'un réseau de capteurs sans fil capable d'effectuer certains traitements, d'acquérir des informations sensorielles et de communiquer avec les autres nœuds du même réseau.

Généralement, un nœud de capteur est composé d'une unité de traitement, d'une unité de communication (émetteur-récepteur), d'une alimentation et d'un ou plusieurs capteurs.

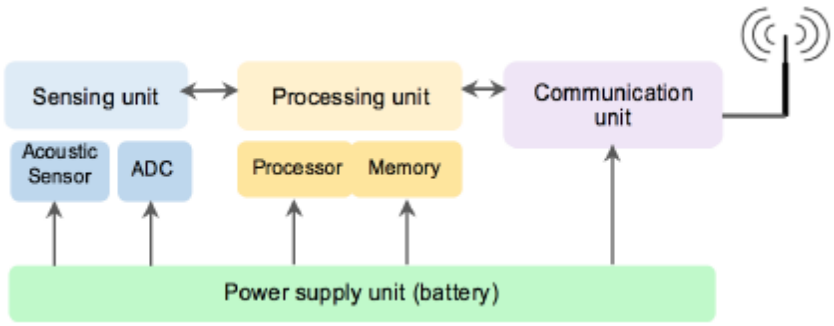

Figure 2 : Exemple de structure d'un nœud de capteur sans fil

Unité de traitement: L'unité de traitement, qui est souvent un microcontrôleur, assure le traitement du signal numérique et gère les autres unités de nœud. Pour certains nœuds, l'unité de traitement peut être une technologie développée telle qu'un processeur de signal numérique ou même une cible FPGA. Le contrôleur est généralement accompagné d'une mémoire de

stockage. La mémoire flash est la plus utilisée en raison de sa capacité de stockage et de son coût.

Unité de communication (émetteur-récepteur): La transmission utilisée est souvent une radiofréquence (RF). Cependant, certains nœuds de capteurs possèdent des unités de communication optique (laser) ou infrarouge. La technologie infrarouge, tout comme les lasers, a une capacité de diffusion limitée. En RF, le WSN utilise souvent les bandes radio industrielles, scientifiques et médicales (ISM) telles que les fréquences 173, 433, 868, 915 MHz; et 2,4 GHz [1].

Bloc d'alimentation: l'alimentation d'un nœud de capteur sans fil, souvent placé dans un endroit difficile à atteindre, ne peut qu'être assurée par une batterie. Néanmoins, le remplacement fréquent de la batterie peut être difficile et coûteux. Cet aspect représente l'inconvénient majeur du WSN. Parmi les évolutions connues du réseau WSN, on peut citer l'extension de leur durée de vie en utilisant plusieurs stratégies:

- Réduire la consommation d'énergie du contrôleur et de l'unité de transmission,
- Fermeture des parties du nœud du capteur qui ne sont pas utilisées actuellement,
- Adapter les niveaux d'alimentation dans le nœud du capteur en fonction de la charge de travail,
- Améliorer la technologie de stockage électrique des batteries en utilisant d'autres matériaux électrochimiques. Aujourd'hui, le matériel électrochimique sont utilisés NiCd (nickel-cadmium), NiZn (nickel-zinc), NiMH (nickel-hydrure métallique) et lithium-ion ... etc [4].

Unité de détection: elle peut généralement comprendre plusieurs unités de captage et chaque capteur produit une réponse mesurable à un changement d'état physique tel que la température, l'humidité ou la lumière. Cette unité est généralement composée de deux sous-unités: les capteurs et les convertisseurs analogiques-numériques (ADC) [5].

3. Equipements adoptés

Il existe de nombreux petits modules embarqués, à l'instar du MICAz, Telos, Waspmote, cyclops, imote2, arduino due, raspberry PI3 …etc, dotés de capteurs scalaires (température, humidité, polluants atmosphériques de type NOx, ….etc) et aussi de capteurs d'images (comme par exemple uCamIII, Caméra IMB400, Camera Module V2-8, CMUcam, ARducam …etc) nous permettant de construire la solution que l'on adoptera pour des applications de surveillance de l'environnement.

L'une des plus simples de point de vue conception et programmation tout en assurant une grande flexibilité est la solution utilisant un arduino DUE, basé sur un Atmel ARM

Cortex SAM3X8E cadencé à 84 MHz sous 32 bits, surtout si nous nous limitons à une application basée sur l'acquisition d'images et non de vidéo. En effet, ce type de module dispose d'une technologie largement suffisante, en particulier la taille de sa mémoire SRAM (Static Random Access Memory) de 96koctets pour manipuler des images en format brut (RAW) d'une certaine résolution, par exemple de 128x128 pixels codés sur 8 bits chacun. Maintenant, si les exigences de notre application en terme de technologie (vitesse d'exécution, taille mémoire …etc) sont plus pertinentes spécialement si on doit travailler avec des images et/ou des vidéos de tailles importantes nous pouvons nous orienter vers des Raspberry PI 3, basés sur des processeurs ARM (ARM Cortex-A53 64 bits à 1,2 GHz) avec des mémoires RAM pouvant atteindre 1 Goctet, avec leur caméras HD fixées sur un port CSI (Camera Serial Interface) et un décodage et encodage H.264 1080p30 intégré.

Le choix entre un arduino (en l'occurrence due) et un raspberry s'avère donc assez difficile compte tenu que chacun des deux modules possède des avantages mais aussi des limites et inconvénients. En effet, ces deux modules sont très différents. À partir du noyau. Arduino est livré avec un microcontrôleur 8 bits. Le Raspberry Pi est livré avec un microprocesseur 64 bits.
Arduino a 2 kilo-octets de RAM. Raspberry Pi a 1 Go de RAM. (500x)
En ce qui concerne les E / S, Arduino possède un port USB-B qui peut être utilisé par un ordinateur pour transférer de nouveaux programmes, une entrée d'alimentation et un ensemble de broches d'E / S. Un Raspberry Pi est beaucoup plus sophistiqué à cet égard, ayant une sortie vidéo, un port HDMI, un port de carte SD, une prise audio, un port de caméra CSI, un port d'affichage DSI, 4 ports USB 2.0 que vous pouvez utiliser pour connecter des périphériques USB, une prise Ethernet Gigabit, un réseau local sans fil, Bluetooth 4.2 et des broches d'E / S (GPIO). Arduino n'a pas de système d'exploitation. Il ne peut exécuter que des programmes compilés pour la plate-forme Arduino, ce qui signifie principalement des programmes écrits en C ++. Raspberry Pi exécute un système d'exploitation, qui est généralement Linux. C'est un mini-ordinateur, alors qu'Arduino est beaucoup plus simple. Lequel devrions-nous utiliser?

Compte tenu de ces différences, on peut pensez logiquement qu'un Raspberry Pi est tellement plus puissant et capable qu'un Arduino. Oui, mais un Arduino consomme beaucoup moins d'énergie (~ 50 mA en veille) qu'un Raspberry Pi (700+ mA). De même, un Arduino a 20 broches d'E / S. Raspberry Pi en a 8. Les broches d'E / S individuelles d'Arduino peuvent gérer 40 mA tandis que les broches RIO de Raspberry Pi peuvent chacune gérer un maximum de 16 mA.

	Raspberry Pi	Arduino
1	C'est un mini ordinateur avec Raspbian OS.Il peut exécuter plusieurs programmes à la fois.	Arduino est un microcontrôleur. Il exécute un seul programme encore et encore.
2	Il est difficile d'alimenter avec une batterie.	Arduino peut être alimenté par une batterie.
3	Cela nécessite des tâches complexes telles que l'installation de bibliothèques et de logiciels pour l'interfaçage de capteurs et d'autres composants.	Il est très simple d'interfacer des capteurs et d'autres composants électroniques avec Arduino.
4	Il est assez coûteux	Il est disponible à faible coût.
5	Raspberry Pi peut être facilement connecté à Internet en utilisant un port Ethernet et des clés de sécurité USB Wi-Fi ou natifs.	Arduino nécessite du matériel externe pour se connecter à Internet et ce matériel est adressé correctement à l'aide d'un code.
6	Raspberry Pi n'avait pas de stockage à bord. Il fournit un port de carte SD.	Arduino peut fournir un stockage embarqué.
7	Raspberry Pi dispose de 4 ports USB pour connecter différents périphériques.	Arduino ne dispose que d'un seul port USB pour se connecter à l'ordinateur.
8	Le processeur utilisé est de la famille ARM.	Le processeur utilisé dans Arduino appartient à la famille AVR Atmega328P
9	Il doit être correctement arrêté sinon il y a un risque de corruption de fichiers et de problèmes logiciels.	Ceci est un appareil juste plug and play. Si le courant est connecté, il lance le programme et s'il est déconnecté, il s'arrête simplement.
10	Le langage de programmation recommandé est Python, mais C, C ++, Python et Ruby sont préinstallés.	Arduino utilise C / C ++.

Tableau 3 : Raspberry vs Arduino

Dans ce projet nous sommes donc intéressés à la mise en œuvre d'un nœud capteurs visuels sans fil basé sur un Raspberry PI3 doté d'une caméra HD. Ceci malgré son handicap vis-à-vis de la consommation énergétique. Quant au module de transmission c'est un dongle wifi (802.11).

Noeud	Fabricant	Contrôleur	RF	RAM (Bytes)	système d'exploitation
MICA2	MEMSIC, USA	ATmega128L – 16bit 8MHz	433/315 or868/916 MHz	4 k	TinyOS
Telos	University of California, Berkeley/Sentilla (Moteiv)	TI MSP430 – 16bit 8MHz	2.4GHz IEEE 802.15.4	10 k	TinyOS
MICAz	MEMSIC, USA	ATmega128L – 16bit 8MHz	2.4 GHz IEEE 802.15.4	4 k	TinyOS
ez430-RF2500	Texas Instrument, USA	TI MSP430 – 16bit 16MHz	2.4 GHz SimpliciTI	1 k	TinyOS
WSN430	FIT IoT-LAB	TI MSP430 – 16bit 8MHz	ISM band (315 to 915 MHz) IEEE 802.15.4 2.4 GHz	10 k	FreeRTOS, Contiki, Riot, TinyOS, OpenWSN
VirtualSense	/	TI MSP430 – 16-bit 25MHz	2.4 GHz IEEE 802.15.4	16	JAVA
Waspmote	Libenium	AT Atmega 1281	868/900/2400 MHz	8 k	Libelium OTAP[1]
LOTUS	MEMSIC, USA	NXP LPC1758 32-bit ARM Cortex-M3	2.4 GHz IEEE 802.15.4	64 k	RTOS, MoteRunner and TinyOS
Shimmer	TI MSP430F1611	TI MSP430F1611	2.4 GHz IEEE 802.15.4	10 k	TinyOS

Tableau 4: Caractéristiques de certains nœuds de capteurs sans fil

[1]Over the Air Programming

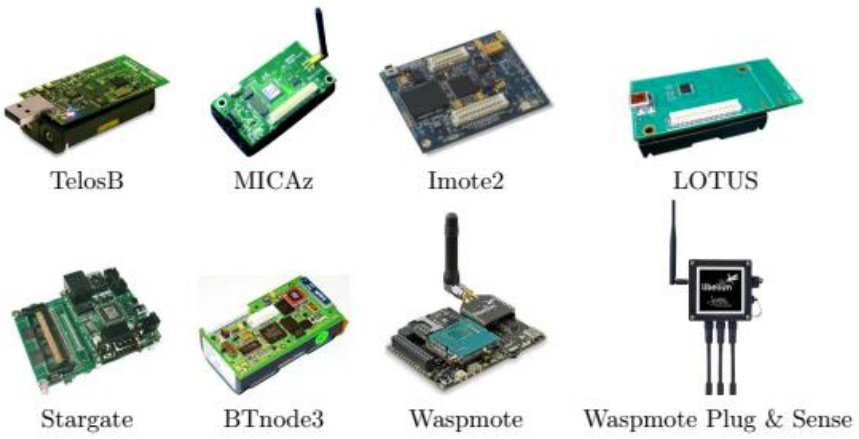

Figure 3 : Solutions existantes pour l'implémentation d'un capteur scalaire/vidéo communiquant.

3.1 Module Raspberry Pi :

Plusieurs types de systèmes embarqués existent. Les systèmes fortement contraints nécessitant une grande puissance de calcul requièrent des cycles de développement lourds et coûteux. Les électroniques mettent en œuvre des technologies onéreuses et en faibles quantités.
Les évolutions technologiques ont permis :

- D'avoir des processeurs de plus en plus puissants, consommant de moins en moins, des coûts de plus en plus réduits,
- La venue du logiciel libre (avec ses avantages et ses inconvénients) a permis une démocratisation du développement logiciel.

On doit trouver un système qui regroupe un bon nombre d'éléments habituellement disponibles que sur des plateformes lourdes (puissance de calcul, périphériques entrée/sortie, environnement logiciel libre) sur un format facilement embarquant, à un prix défiant toute concurrence et facile a programmé à savoir le Raspberry PI3

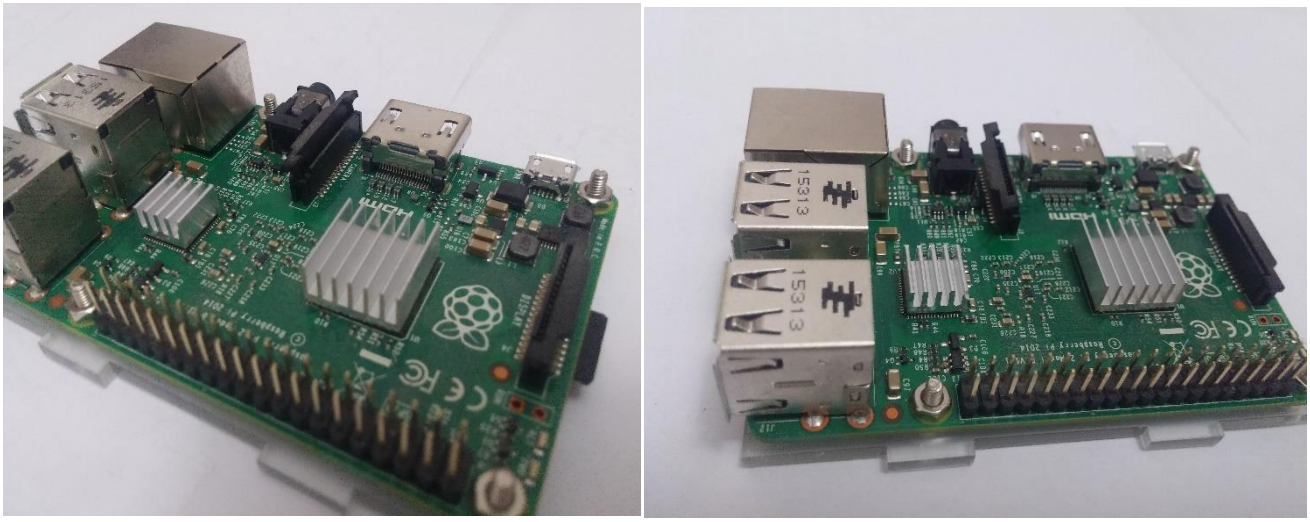

Figure 4: Raspberry Pi 2 model B

Les premiers prototypes du Raspberry Pi sont développés sur des microcontrôleurs Atmel ATmega 644. Ce Microordinateur s'inspire du BBC Micro d'Acorn Computer (1981) et est destiné à encourager la jeunesse à la programmation. Le model B a été élancé en février 2015

Avec 900MHz Quad-Core ARM-cotre-A7 CPU :

- 1GB de RAM
- 4 ports USB
- Full HDMI port
- Port Ethernet
- Prise jack 3.5mm
- Interface d'une smart camera
- Interface d'un écran
- Lecteur de carte mémoire
- VidéoCore IV 3D graphique

Il s'agit tout simplement un microordinateur qui a des caractéristiques d'un smartphone de milieu de gamme et critiquable au niveau de :

- utilisation carte SD pas suffisamment fiable.
- tenue en température.

Toutefois elle offre des avantages non négligeables :

- puissance de calcul importante
- interfaçage facile : UART, USB, Ethernet, Picamera
- Faible coût

3.2 Raspberry Pi Camera V2

On parle d'une smart-Camera connectée à travers l'interface camera de la Raspberry Pi. La V2 a remplacé l'ancienne version dotée d'un capteur Sony IMX219 8MP. Vidéo HD avec ces qualités elle très efficace pour beaucoup d'applications en traitement d'images et vidéos.

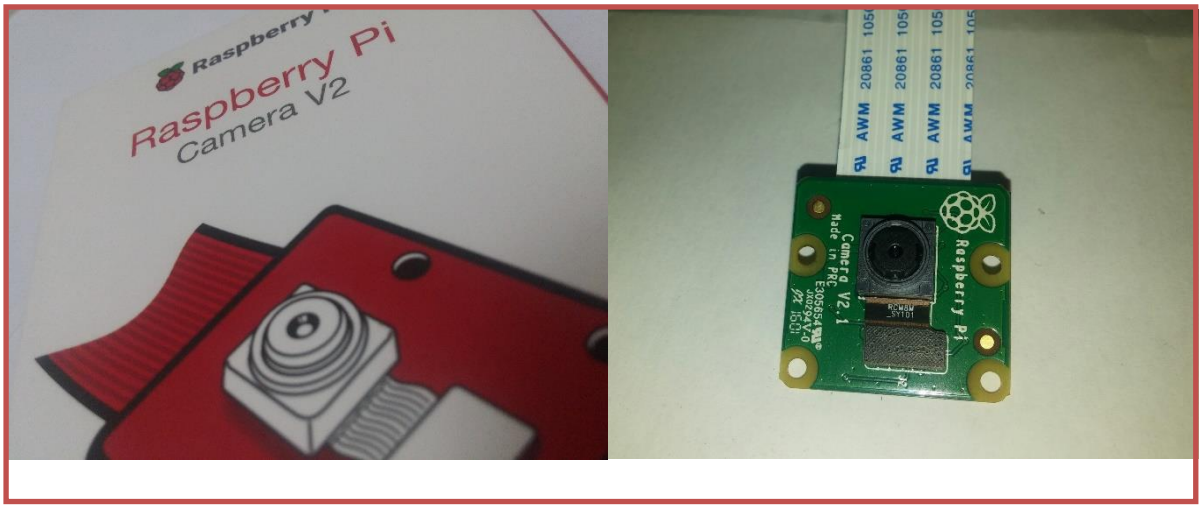

Figure 5: Raspicam V2.1

3.3 Software:

3.3.1 L'OS Raspbian Jessie Avec Pixel

La Fondation Raspberry a mis en ligne une version gratuite de raspbian jessie basée sur Linux Debian avec la technologie d'affichage de bureau appelée Pixel. (Pi Improved Xwindows environment Light weight), installé sur notre carte mémoire 16Gb classe 10 Toshiba.

3.3.2 Python 2.7 et 3.0

Un langage de programmation riche et facile très utilisé pour les débutants est trop flexible et relativement court. Un code avec python est plus court jusqu'à 10 fois que le C++. Par contre il est un langage de programmation lent lors des calculs.

4. Conclusion

Dans ce chapitre nous avons présenté la solution adoptée pour réaliser nos nœuds capteurs sans fil. Nous avons alors opté pour une solution à base d'un module raspberry PI3. Dès lors nous avons exposé les différents types de raspberry qu'on peut trouver sur le marché et ses principales caractéristiques. Comme Nous avons aussi rappelé quelques autres technologies utilisées pour la mise en œuvre de nœuds capteurs sans fil.

Chapitre 3
La réalisation du nœud capteur

Chapitre 3 : La réalisation du nœud capteur

1. Introduction :

Le projet consiste à développer et étudier la conception, la mise en œuvre et l'implémentation d'une chaine de compression image de faible coût et robuste vis-à-vis des erreurs de transmission. Cette chaine de compression JPEG doit être implémentée dans un nœud capteur à base d'un Raspberry PI3. Il s'agit donc d'utiliser un petit système embarqué (un raspberry PI3) avec une caméra HD (Haute définition) qui doit :

- ✓ Capter des images avec une certaine cadence (exemple 1 image par seconde)
- ✓ Les compresser avec JPEG approximé à faible coût calculatoire
- ✓ La transmettre en wifi vers un autre système embarqué ou bien directement à un PC
- ✓ Evaluer les résultats en termes de qualité d'images reçues et consommation énergétique.

2. Présentation du matériel :

2.1 Liste du matériel utilisé :

- ✓ Deux cartes raspberry pi B+
- ✓ Caméra PI v2.1 8Mp
- ✓ Une carte SD de 32 GB avec Raspbian préinstaller dessus
- ✓ Le boitier pour la carte Raspberry Pi B+
- ✓ Le chargeur EU/US

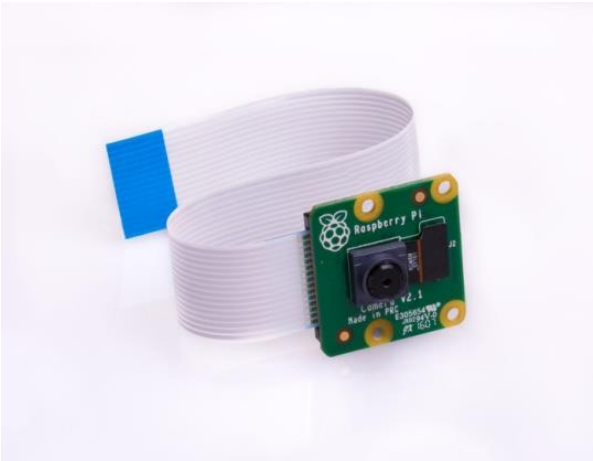

Figure 6 : camera pi V2.1 8Mp

En effet le Pi 3 intègre nativement le Wi-Fi et le Bluetooth ce qui facilite les manipulations et apporte certains avantages. Moins de drivers à installer car on n'utilise plus de dongles WiFi et Bluetooth et on libère ainsi 2 ports USB ce qui va réduire un peu notre consommation électrique. En plus des modules raspberry PI3, notre réalisation a besoin d'un matériel additionnel:

- Lecteur de carte MicroSD pour formater les cartes
- Un clavier, une souris, un écran qui sont utiles pour les premières utilisations du Raspberry Pi, c'est-à-dire l'installation du système d'exploitation et sa configuration
- Le câblage nécessaire, c'est-à-dire un câble d'alimentation, un câble HDMI pour brancher le Raspberry à l'écran,
- Eventuellement un câble Ethernet pour accéder au réseau et à internet si le WiFi n'est pas opérationnelle.

2.2 Spécification du Raspberry :

Le Raspberry Pi est une série de nano-ordinateurs mono-carte (avec un processeur ARM) développé en Angleterre par David Braben (un créateur de jeux-vidéos) dans le cadre de la fondation « Raspberry Pi Foundation » afin de promouvoir l'apprentissage des sciences de l'informatique dans les écoles des pays en voie de développement. Cet ordinateur ayant la taille d'une carte de crédit permet l'utilisation de nombreux systèmes d'exploitation, en particulier GNU/Linux. Le premier modèle s'est répandu plus vite que ce qui était prévu avec des utilisations comme la robotique. Les périphériques comme les claviers, souris et boitiers ne sont pas inclus avec le Raspberry Pi dans l'optique de réduire les coûts et réutiliser du matériel mais des accessoires ont été inclus dans des packs officiels ou non.

Du coté hardware, la plupart des modèles suivent la représentation suivante, à l'exception des modèles A, A+ et Zero qui ne possèdent pas de connexion Ethernet et de ports USB.

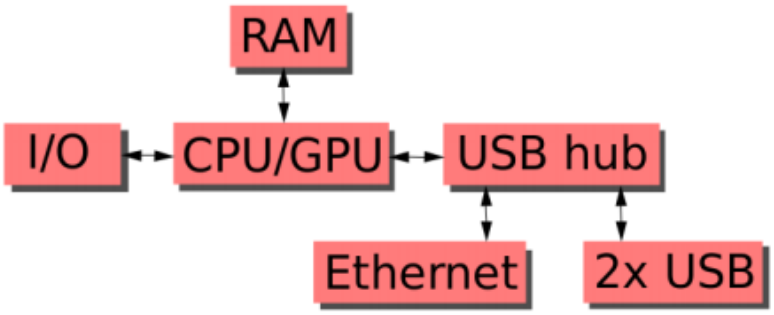

Figure 7 : Diagramme représentant les interactions entre les composants du Raspberry

Le Raspberry PI3 contient certaines nouveautés par rapport à ses prédécesseurs. Nous pouvons sur la figure suivante les différentes constitutions d'un raspberry PI3.

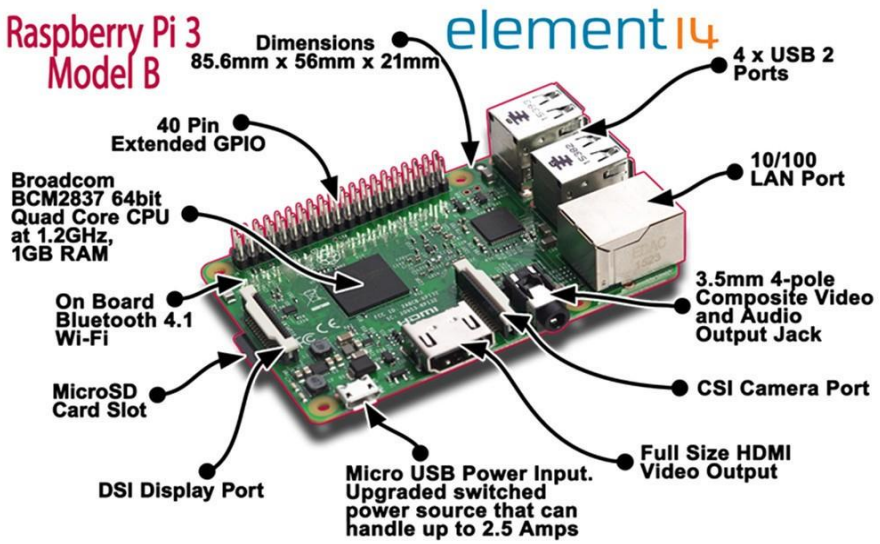

Figure 8 : Certaines des nouveautés incluses dans le Raspberry Pi 3

3. Installation/configuration du deux Raspberries :

3.1 Choix de la distribution :

Le Raspberry Pi peut fonctionner sous de nombreux systèmes d'exploitation qui sont compatibles avec l'architecture ARM :

- Debian : qui a une version dédiée appelée Raspbian (avant avec le kernel Wheezy et puis maintenant avec Jessie), ce système est recommandé par la fondation Raspberry PI
- Ubuntu Mate et Snappy Ubuntu : basé sur Ubuntu, Snappy Ubuntu est plus réservé aux développeurs.
- Fedora : un autre système d'exploitation libre et une distribution GNU/Linux. Arch Linux : encore une distribution Linux qui met en avant la simplicité.
- Gentoo Linux, Slackware et Suse qui sont des distributions Linux.
- RISC OS : un système d'exploitation spécialement conçu pour les architecture ARM.
- NetBSD : un système d'exploitation libre de type Unix BSD.
- Kali Linux : anciennement Backtrack, ce système d'exploitation possède tous les outils nécessaires aux tests de sécurité d'un système d'information.
- Windows 10 IOT : Windows 10 spécialement conçu pour les objets connectés.
- OSMC : Media center gratuit et libre, basé sur du Linux.
- LibreElec : évolution de OpenElec, un autre media center.

Il existe aussi NOOBS (New Out Of the Box Software) qui est un assistant et va permettre l'installation d'un ou plusieurs système d'exploitation sur la carte MicroSD sans avoir à se soucier du partitionnement. NOOBS possède des avantages certains : tout d'abord il est simple à mettre en œuvre sur la plupart des systèmes d'exploitation et tous est simplifié au maximum grâce à une interface graphique épurée, on peut installer plusieurs systèmes d'exploitation sur une même carte SD, s'il y un problème on peut réinitialiser ou désinstaller le système d'exploitation source du problème. Nous avons choisi de ne pas passer par NOOBS et d'installer manuellement Raspbian sur le Raspberry Pi. En effet les avantages proposés par NOOBS ne sont pas utiles dans notre situation car nous savons déjà installer Debian en mode expert sur une machine normale et nous comptons utiliser seulement Raspbian. De plus NOOBS prend sensiblement plus de place sur la carte mémoire et nous semble moins stable de façon générale que Raspbian.

Au démarrage du Raspberry Pi avec une carte MicroSD ayant NOOBS chargé dessus, c'est le menu que l'on obtient (figure ci-dessous).

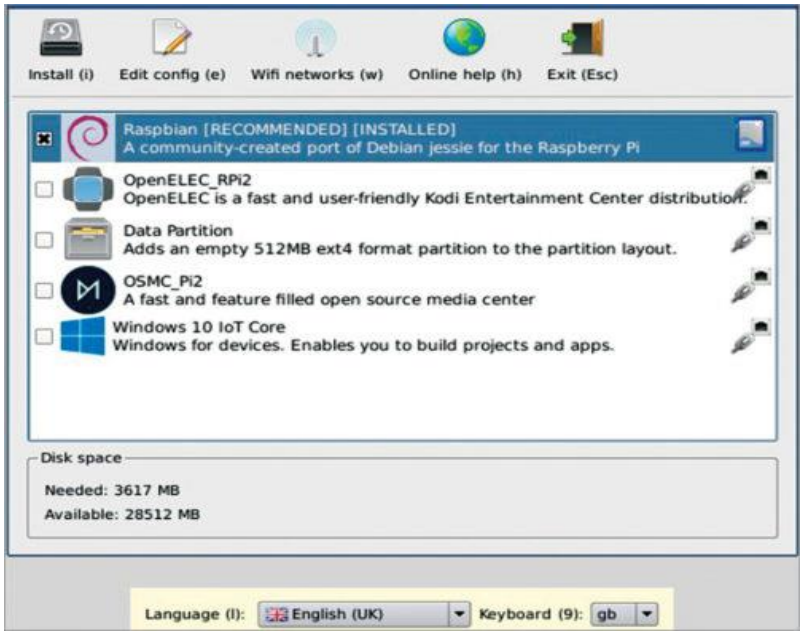

Figure 9 : Menu de démarrage du Raspberry Pi avec une carte MicroSD avec NOOBS

A titre de comparaisons entre les systèmes d'exploitation Raspbian et Debian il faut savoir que Raspbian est un portage non officiel de Debian Wheezy ARMhf (Arm Hard Float Port). Les architectures ARM, sont des architectures matérielles RISC (Reduced Instruction set computer, traduit littéralement par processeur à jeu d'instructions réduit). On ne peut donc pas installer les paquets de la même manière sur une architecture ARMhf et AMD64/i386 (Debian). Il va donc falloir utiliser ce qu'on appelle de la compilation croisée.

Une chaîne de compilation croisée est une chaîne compilée pour fonctionner sur l'architecture de processeurs de la machine hôte, mais qui va compiler des logiciels pour une architecture

cible différente. La compilation croisée fait donc référence aux chaînes de compilation capables de traduire un code source en code objet dont l'architecture processeur diffère de celle où la compilation est effectuée. Ces chaînes sont principalement utilisées en informatique industrielle et dans les systèmes embarqués.

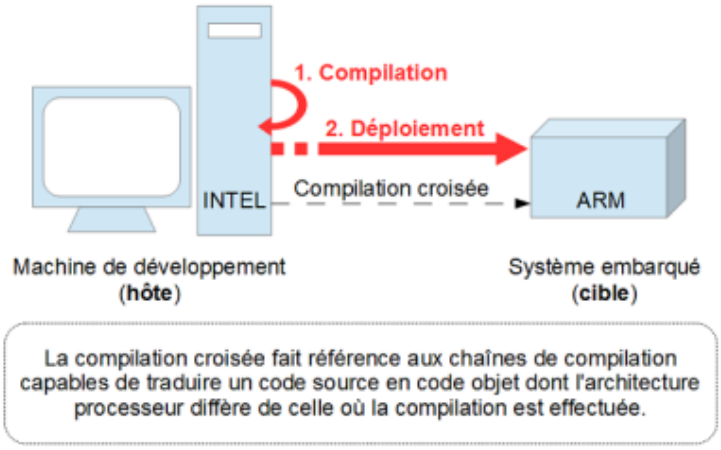

Figure 10 : Schéma du fonctionnement de la compilation croisée.

3.2 Première installation et configuration de Raspbian :

Les principales étapes pour installer et configurer Raspbian :

- Préparation de la carte MicroSD, pour cela il faut télécharger l'image de Raspbian, puis il faut utiliser un logiciel comme Win32DiskImager pour rendre la carte bootable avec le bon système d'exploitation dessus.
- On met la carte MicroSD dans le Raspberry et on effectue les branchements nécessaires à une première installation (souris, clavier, câble Ethernet, câble DVI avec un adaptateur DVI-HDMI) puis on alimente le Raspberry.
- On attend que le système démarre et lorsque nous avons accès à l'interface graphique, on lance un terminal pour commencer la configuration de Raspbian avec la commande sudo raspi-config.
- Faire en sorte que Raspbian utilise bien toute la carte MicroSD avec Expand Filesystem.
- De base le clavier est en QWERTY, il faut donc le passer en AZERTY grâce au menu Change Keyboard Layout.
- Passer Raspbian en Français grâce au menu Change Locale.
- Modifier le mot de passe pour l'utilisateur par défaut pi et mettre un mot de passe pour root.
- Activation du protocole SSH afin de ne plus avoir besoin de tous les accessoires lors des prochaines utilisations.
- Mettre à jour les sources et faire un upgrade du système puis redémarrer

- on installe VNCserver qui s'occupe d'exporter l'environnement graphique au lieu d'utiliser a chaque fois un écran, souries et clavier donc avec ce logiciel on peut accéder a la carte avec une connexion SSH.
- On active la camera PI

4. Installation et configuration de Raspbian

Il faut commencer par préparer la carte SD depuis Windows afin que celle-ci intègre Raspbian et non NOOBS. On peut utiliser le logiciel Win32DiskImager pour ce faire. Les performances du Raspberry Pi sont fortement influencées par la qualité de la carte SD on a donc choisi une carte SD de 32Go. Il faut ensuite télécharger la dernière version de Raspbian disponible sur le site officiel de Raspberry Pi : https://www.raspberrypi.org/downloads/raspbian/

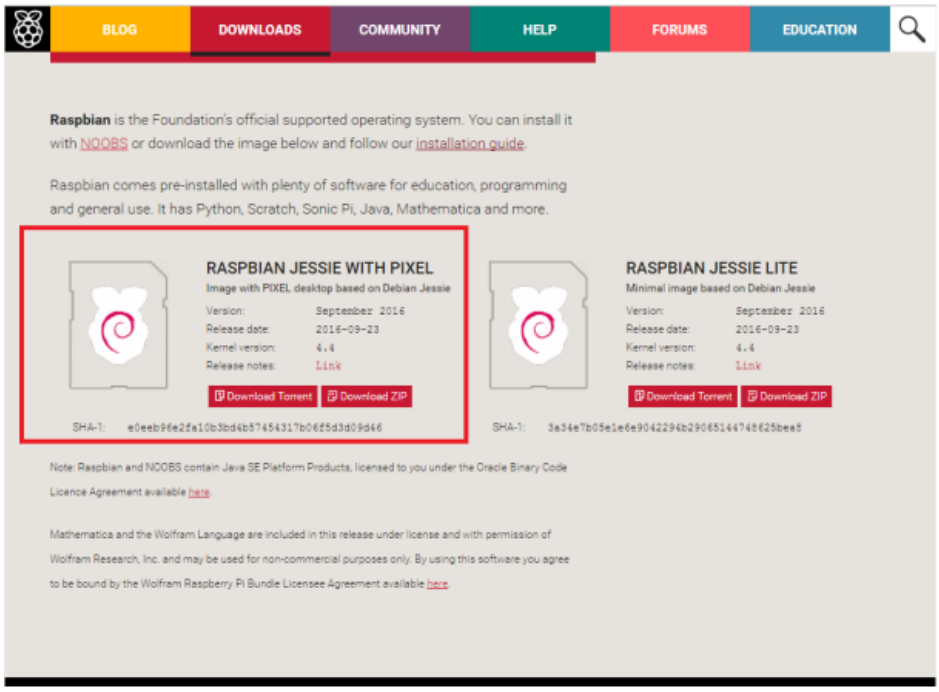

Figure 11 : Téléchargement de Raspbian, pour commencer l'installation sur le Raspberry Pi

Win32DiskImager est téléchargeable à partir de l'adresse suivante :
https://sourceforge.net/projects/win32diskimager/files/latest/download

Après avoir installé et lancer Win32DiskImager, on décompresse le fichier .zip de l'image de Raspbian pour obtenir un fichier .img. On insère la carte SD dans le lecteur que l'on connecte à l'ordinateur, celle-ci doit bien être reconnue.

1. On indique ou se trouve l'image que l'on veut écrire sur la carte SD (Raspbian-jessie.img).
2. On choisit la carte SD comme périphérique [J:\].
3. On appuie sur Write.

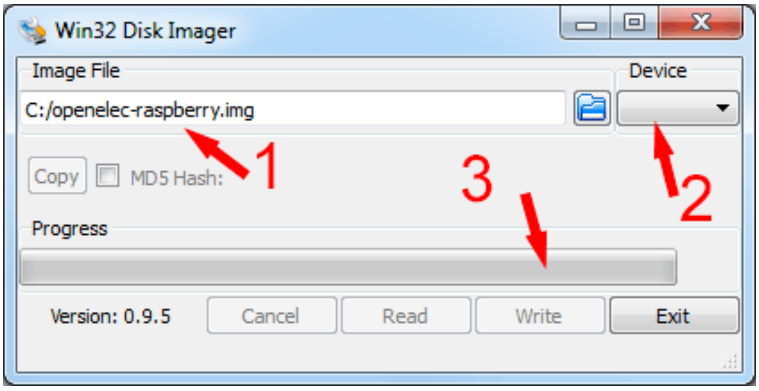

Figure 12: Etapes pour décompresser l'image sur la carte SD.

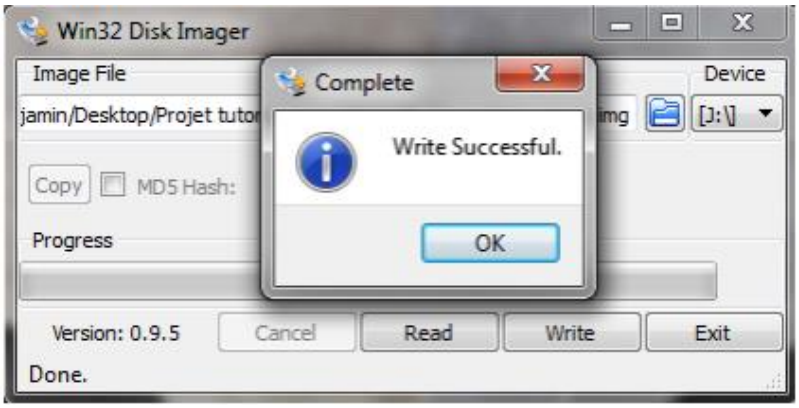

Figure 13 : Attendre la fin de l'écriture sur la carte SD.

Nous allons maintenant insérer la carte SD et effectuer les branchements nécessaires pour démarrer le Raspberry Pi. Le Raspberry s'occupe lui-même de l'installation de Raspbian. Après le démarrage on arrive à voir sur l'écran connecté au raspberry une Interface Graphique avec déjà pas mal de fonctionnalités installées : Python 2 et 3, Java IDE, Geany, Wolfram, la suite LibreOffice, Chromium et d'autres moins utiles.

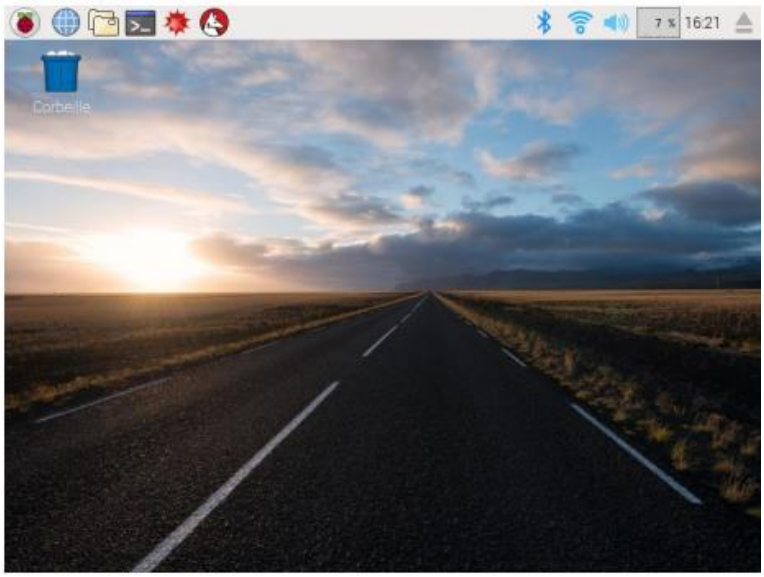

Figure 14 : Bureau du Raspberry Pi 3.

Pour toutes configuration (réglage heure, clavier ..) il faut juste taper la commande :
sudo raspi-config

5. Connexion réseau avec Putty et VNC :

VNC, ou Virtual Network Computing, est un système permettant la prise de contrôle à distance d'un ordinateur par un autre. Lors de l'utilisation de VNC, deux parties différentes du logiciel sont utilisées. La première partie est le serveur VNC. Celui-ci est installé sur la machine dont l'on souhaite prendre le contrôle (ici la raspberry pi), et il va permettre la connexion et le contrôle par la partie client. La seconde partie est donc le client VNC. Celui-ci est installé sur la machine depuis laquelle on souhaite contrôler le serveur, et il va permettre de traduire vos actions en opérations compréhensibles par le serveur qui va alors contrôler la machine distante depuis votre ordinateur.

PuTTY est un émulateur de terminal doublé d'un client pour les protocoles SSH, Telnet, rlogin, et TCP brut. Il permet également des connexions directes par liaison série RS-232. À l'origine disponible uniquement pour Windows, il est à présent porté sur diverses plates-formes Unix.

Pour pouvoir utiliser ces deux software et contrôler à distance le rapsberry il faut activer les ports SSH et VNC a partir de la commande raspi-config dans raspberry et installer les deux logiciels sur Windows.

Nous avons maintenant la main à distance sur la carte Raspberry

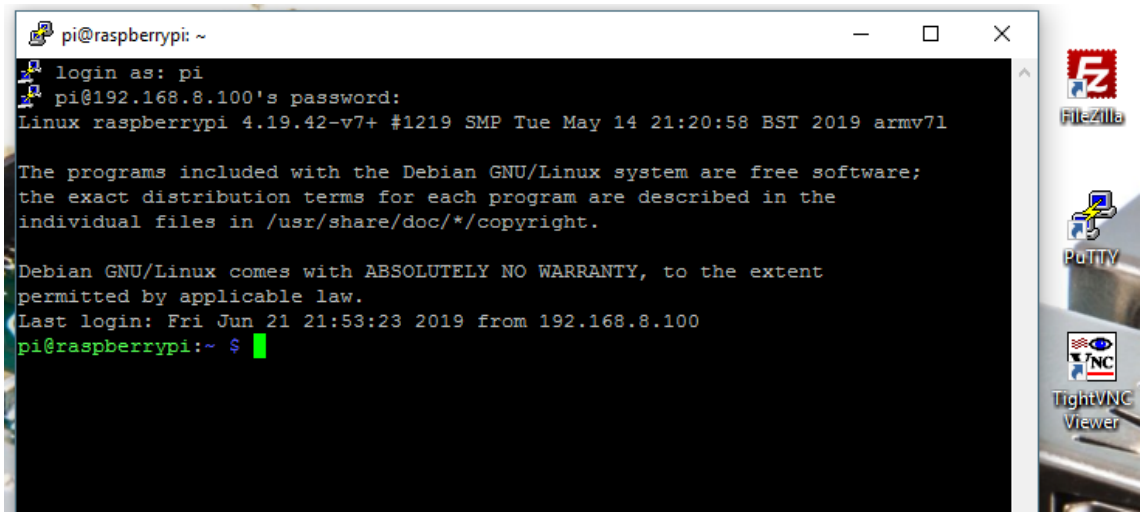

Figure 15 : Contrôle à distance d'un raspberry

6. Utilisation de la Caméra :

Avec un capteur d'une résolution native de 8 mégapixels (8M) nous aurons la possibilité d'effectuer des prises d'images et vidéo de haute définition. Ainsi, plusieurs applications en traitement d'images, de vidéo et en robotique peuvent être réalisées. Le port de la caméra HD au niveau du raspberry est le CSI (Camera Serial Interface).

Figure 16 : Raspberry et caméra HD

Pour utiliser cette caméra et prendre des prises de vue et/ou des séquences vidéo, plusieurs possibilités s'offrent à nous. Nous pouvons utiliser entre autres le langage de programmation python :

```
from time import sleep
from picamera import PiCamera

camera = PiCamera()
camera.resolution = (1024, 768)
camera.start_preview()
# Camera warm-up time
sleep(2)
camera.capture('foo.jpg')
```

Figure 17 : Exemple d'un script de prises de vue par Python

Ou encore l'opencv et python 3 :

```
50
51  def decode_zip(orig, shape):
52      return np.frombuffer(zlib.decompress(orig), dtype=np.int8).astype(float).reshape(shape)
53
54
55  if __name__ == '__main__':
56
57      im = cv2.imread("IMG_0108.JPG")
58      quants = [1, 10] #[0.5, 1, 2, 5, 10]
59      blocks = [(16,16)] #[(2, 8), (8, 8), (16, 16), (32, 32), (200, 200)]
60      for qscale in quants:
61          for bx, by in blocks:
62
63              quant = (
64                  (np.ones((bx, by)) * (qscale * qscale))
65                  .clip(-100, 100)  # to prevent clipping
66                  .reshape((1, bx, 1, by, 1))
67              )
68              enc = encode_dct(im, bx, by)
69              encq = encode_quant(enc, quant)
70              encz = encode_zip(encq)
71              decz = decode_zip(encz, encq.shape)
72              decq = decode_quant(encq, quant)
73              dec = decode_dct(decq, bx, by)
74              cv2.imwrite("IMG_0108_recompressed_quant_{}_block_{}x{}.png".format(qscale, bx, by), dec.astype(np.uint8))
75
```

Figure 18 : Script pour le codage d'image avec le logiciel open cv (compression)

Figure 19 : exemple d'image acquise par rapsberry

7. Transmission de données entre deux raspberry

7.1 Présentation du protocole MQTT

L'objectif final de notre projet est de pouvoir mettre en œuvre un réseau de capteurs visuels sans fil pouvant capter des images (voire même des séquences vidéo) et les transmettre d'un nœud capteur à autre. Pour cela nous allons utiliser comme moyen de transmission le WiFi. Mais côté software plusieurs solutions s'offrent à nous. Nous avons opté pour un protocole bien connu à savoir le protocole MQTT (Message Queuing Telemetry Transport)

MQTT est un protocole simple, léger et facile à mettre en œuvre. Ce protocole est idéal pour répondre aux besoins suivants :

- Utilisation d'une très faible bande passante
- L'utilisation sur les réseaux sans fils
- Consommation en énergie
- Rapidité avec un temps de réponse supérieur aux autres standards du web actuel,
- Fiabilité
- Usage de faible ressources processeurs et de mémoire

MQTT est un standard ISO (ISO/IEC PRF 20922). Au départ, TT (premier nom du protocole) a été développé par Andy Stanford-Clark (IBM) et Arlen Nipper (Eurotech, actuellement Cirrus Link) en 1999 pour le contrôle d'un pipeline dans le désert. L'objectif était d'avoir un protocole de bande passante efficace utilisant peu d'énergie à un moment où les périphériques ne pouvaient être connectés qu'au travers de satellites. Le protocole MQTT utilise une architecture « publish/subscribe » en contraste avec le protocole HTTP et son architecture « request/response ». Le point central de la communication est le broker MQTT en charge de relayer les messages des émetteurs vers les clients. Chaque client s'abonne via un message vers le broker : le « topic » (sorte d'information de routage pour le broker) qui permettra au broker de réémettre les messages reçus des producteurs de données vers les clients. Les clients et les producteurs n'ont ainsi pas à se connaître, ne communiquant qu'au travers des topics. Cette architecture permet des solutions multi-échelles.

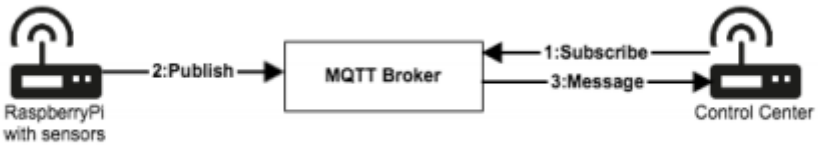

Figure 20 : Architecture MQTT « publish/subscribe »

Chaque client MQTT a une connexion ouverte en permanence avec le broker. Si la connexion s'arrête, le broker bufférise les messages et les émet dès que la reconnexion avec le client est effectuée. Un « topic MQTT » est une chaîne de caractères qui peut posséder une hiérarchie de niveaux séparés par le caractère « / ». Par exemple, une information de température du salon pourrait être envoyée sur le topic « maison/salon/temperature » et la température de la cuisine sur « maison/cuisine/temperature ». Le signe « + » est un caractère « wildcard » qui permet des valeurs arbitraires pour une hiérarchie particulière et le signe « # » pour plus d'un niveau. Les messages envoyés peuvent être de toutes sortes mais ne peuvent excéder une taille de 256 Mo.

Par exemple, si un publisher s'abonne au topic « maison/salon/# », il recevra toutes les données du salon. De même, s'il s'abonne au topic « maison/# », il collectera toutes les données des sondes de la maison.

7.2 Sécurité

Les données IoT échangées peuvent s'avérer très critiques, c'est pourquoi il est aussi possible de sécuriser les échanges à plusieurs niveaux :

- Transport en SSL/TLS
- Authentification par certificats SSL/TLS
- Authentification par login/mot de passe

7.3 Qualité de Service (QoS)

MQTT intègre en natif la notion de QoS. En effet le publisher à la possibilité de définir la qualité de son message. Trois niveaux sont possibles :

- Un message de QoS niveau 0 « At most once » sera délivré tout au plus une fois. Ce qui signifie que le message est envoyé sans garantie de réception, (le broker n'informe pas l'expéditeur qu'il l'a reçu et le message)
- Un message de QoS niveau 1 « At least once » sera livré au moins une fois. Le client transmettra plusieurs fois s'il le faut jusqu'à ce que le Broker lui confirme qu'il a était transmis sur le réseau.
- Un message de QoS niveau 2 « exactly once » sera obligatoirement sauvegardé par l'émetteur et le transmettra toujours tant que le récepteur ne confirme pas son envoi sur le réseau. La principale différence étant que l'émetteur utilise une phase de reconnaissance plus sophistiquée avec le broker pour éviter une duplication des messages (plus lent mais plus sûr).

8. Résultats de connexion entre deux raspberry avec le protocole MQTT

Pour tester le bon fonctionnement de la transmission entre deux raspberry en se basant sur MQTT nous avons effectué plusieurs tests sur des données scalaires et aussi sur des images. Les résultats de transmission sont présentés sous forme d'images et figures (voir ci-dessous).

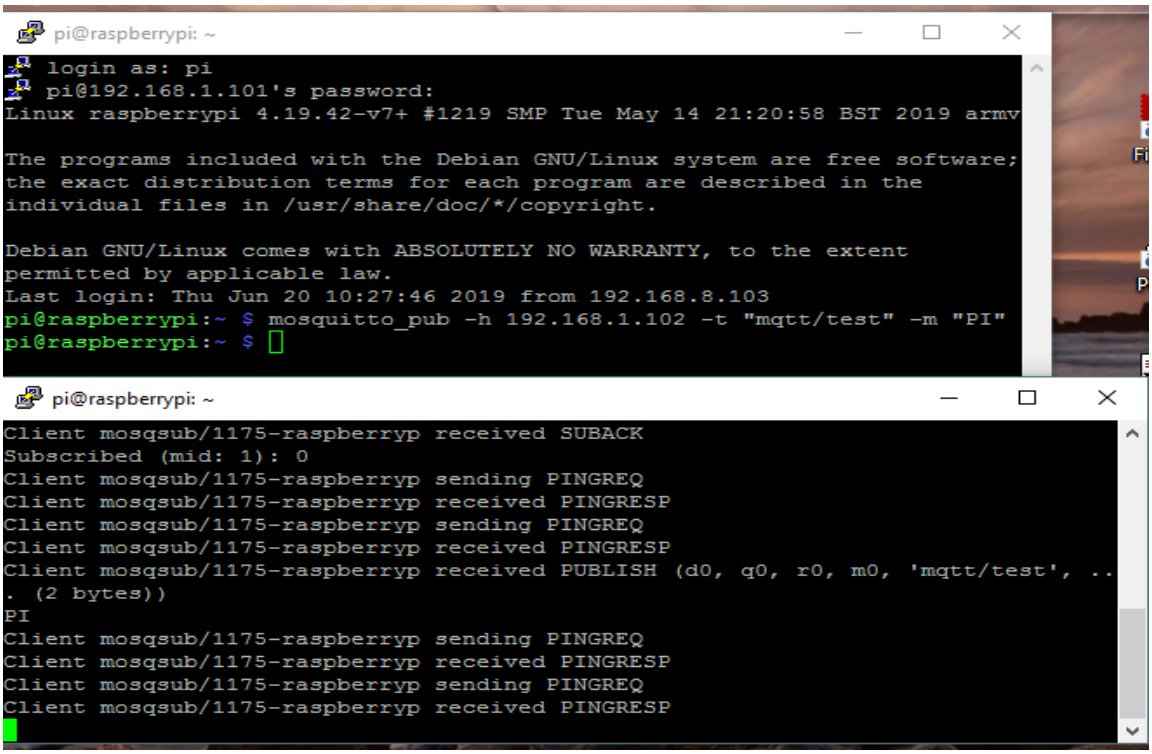

Figure 21 : Envoi de l'Image avec le protocole MQTT

On configure les deus raspberry, l'un en émetteur et l'autre en tant que récepteur :

Émetteur :

- Un Raspberry Pi 3 sous Raspbian Stretch

Récepteur :

- Un Raspberry Pi 3 sous Raspbian Stretch version 04-2018
- Pour traiter les données MQTT le pack Telegraf / Influxdb / Chronograf

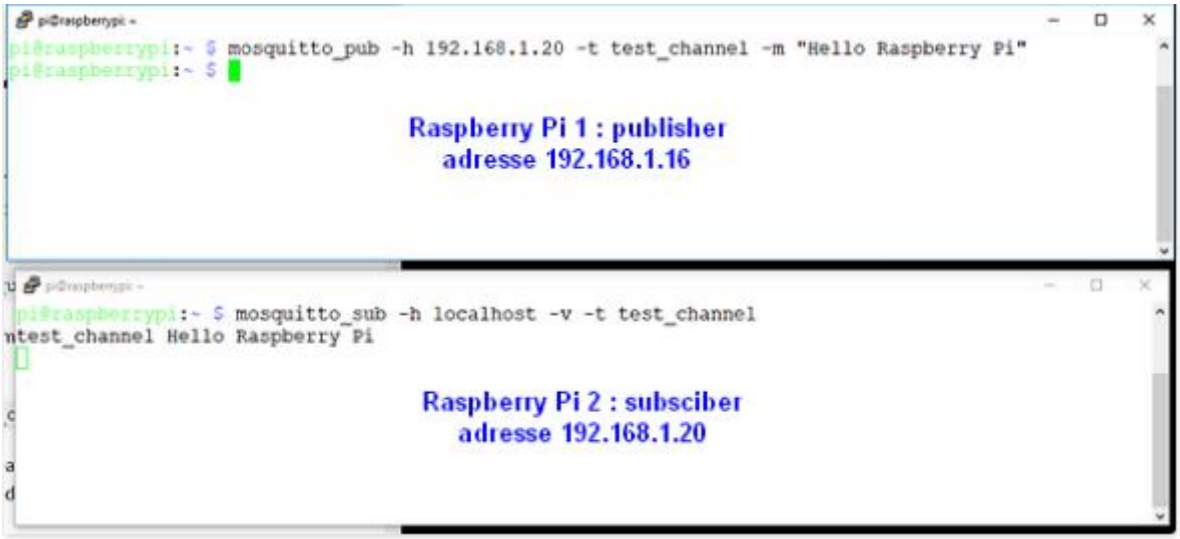

Figure 22 : aperçu sur les deux rapsberry en mode émission/réception

Il faut d'abord convertir l'image en utilisant le script suivant :

```
import base64

def convertImageToBase64():
 with open("image_test.jpg", "rb") as image_file:
 encoded = base64.b64encode(image_file.read())
 return encoded
```

Après la conversion on envoie l'image avec le protocole MQTT :

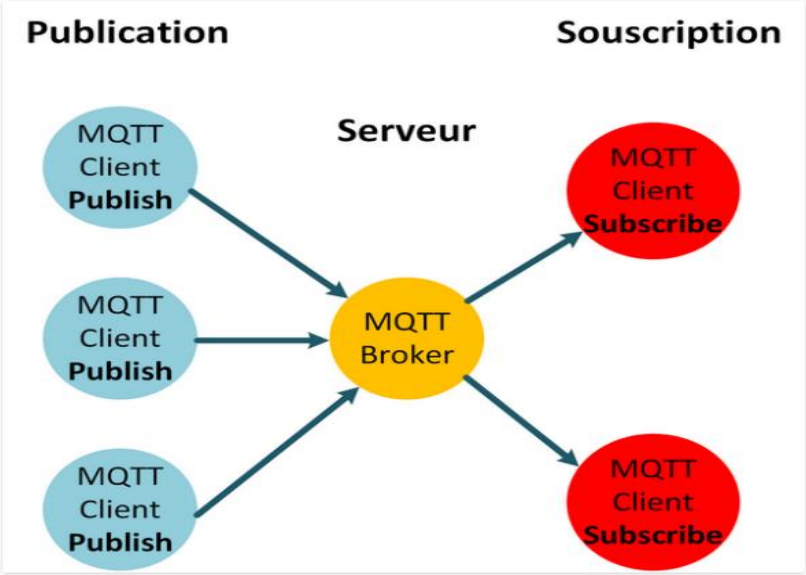

Figure 23 : Principe de fonctionnement du MQTT en mode transmission

En utilisant les logiciels suivants : Open Cv et Python 3

Le code pour l'émission :

```
import paho.mqtt.client as mqtt

def on_publish(mosq, userdata, mid):
    mosq.disconnect()

client = mqtt.Client()
client.connect("test.mosquitto.org", 1883, 60)
client.on_publish = on_publish

f=open("b.jpg", "rb") #3.7kiB in same folder
fileContent = f.read()
byteArr = bytearray(fileContent)
client.publish("image",byteArr,0)

client.loop_forever()
```

Le code pour la réception :

```
import paho.mqtt.client as mqtt

def on_connect(client, userdata, rc):
    print("Connect" + str(rc))
    client.subscribe("image")

def on_message(client, userdata, msg):
    print "Topic : ", msg.topic
    f = open("/tmp/output.jpg", "w")  #there is a output.j
    f.write(msg.payload)
    f.close()

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect("test.mosquitto.org", 1883, 60)

client.loop_forever()
```

9. Conclusion

Dans ce chapitre nous avons pu exposer l'essentiel de notre réalisation de point de vue matériel et logiciel. Ainsi, nos nœuds capteurs sont fins près pour capter, émettre et recevoir des images de différentes résolutions à la cadence souhaitée. Il nous reste simplement d'implémenter le jpeg à faible coût calculatoire dans nos nœuds capteurs afin de réduire la consommation énergétique. Il faut noter que dans le raspberry le jpeg classique est déjà implanté et que nous pouvons utiliser. Mais ce jpeg classique pose le problème d'une relative forte consommation énergétique ce qui handicap l'autonomie énergétique et par conséquence la durée de vie du nœud capteur.

Chapitre 4

Tests et résultats

Chapitre 4 : Tests et résultats

1. Introduction

Après avoir réalisé et mis en œuvre les solutions matérielle et logicielle de notre prototype à base de raspberry nous nous sommes intéressés à l'implémentation d'un jpeg faible consommation, aux tests de ses fonctionnalités et à l'évaluation de ces performances.

2. Compression d'images

L'un des plus grands chalenges à relever pour les réseaux de capteurs sans fil est la contrainte énergétique. En effet, la durée de vie de ces capteurs sans fil est liée directement à leur autonomie énergétique. Pour prolonger la durée de vie des batteries alimentant ces nœuds de capteurs sans fil il faut réduire essentiellement la quantité de données à transmettre. Cette réduction ne peut se faire que grâce à un algorithme de compression. L'algorithme JPEG est la méthode de compression par référence. En effet, la méthode JPEG est la solution la plus efficace pour ce genre d'applications. Dans l'algorithme JPEG (figure 24), norme de référence, la transformée en cosinus discrète (DCT) (1), découverte en 1974 [17-19] et appartenant à la classe des transformées unitaires sinusoïdales, consomme à elle seule plus de 60% de l'énergie totale du codeur [20 – 24].

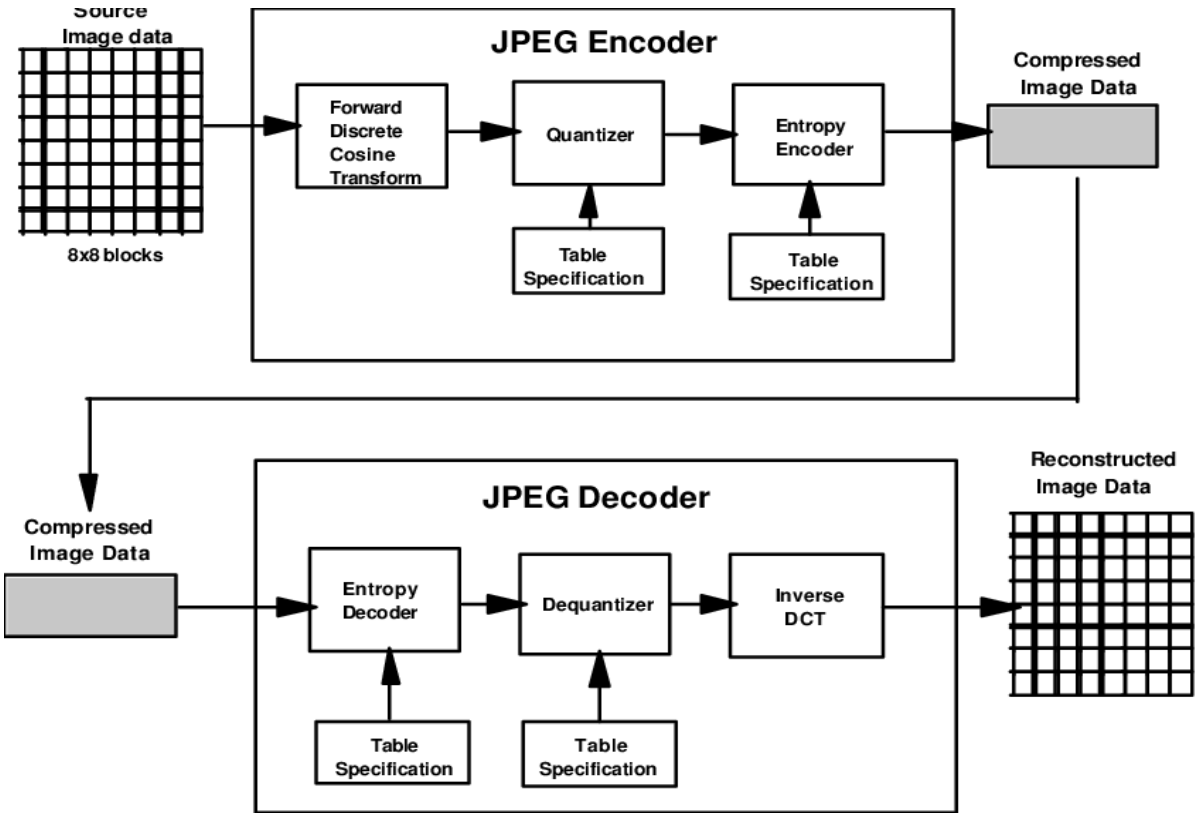

Figure 24 : Schéma de principe d'un codec JPEG

Depuis trois décennies, de nombreux travaux ont pu proposer des algorithmes rapides de la DCT exacte, permettant de gagner du temps, de la vitesse et de l'énergie, en diminuant sa complexité algorithmique.

$$X_k = A_k \times \sum_{n=0}^{7} x_n \cos\left(\frac{\pi}{8} \times \left(n + \frac{1}{2}\right) \times k\right) \text{ for } k = 0,1,\ldots N-1 \quad (1)$$

où $A_0 = \sqrt{\frac{1}{8}}$ et $A_k = \sqrt{\frac{2}{8}}$ ailleurs

avec les N sorties Xk de la transformée 1D réelle DCT-II pour N entrées x_n sont données.

Sous forme matricielle, la DCT-II 1D s'écrit :

$$[X_{8\times1}] = [C^{II}_{8\times8}] \times [x_{8\times1}] \quad (2)$$

Où $[X_{8\times1}]$ et $[x_{8\times1}]$ sont les vecteurs sorties et entrées de taille 8×1. $[C^{II}_{8\times8}]$ est la matrice transformée 1D DCT-II donnée par :

$$[C^{II}_{8\times8}]_{n,k} = A_k \times \cos\left(\frac{\pi}{8} \times \left(n + \frac{1}{2}\right) \times k\right) \text{ pour } k, n = 0,1,\ldots 7 \quad (3)$$

Compte tenu de la propriété de séparabilité, la DCT-II 2D est obtenue en effectuant la DCT-II 1D le long des N lignes de ce bloc, puis le long de ses N colonnes. Cette DCT-II 2D, sous forme de matrice, s'écrit:

$$[X_{8\times8}] = [C^{II}_{8\times8}] \times [x_{8\times8}] \times [C^{II}_{8\times8}]^t \quad (4)$$

La 2D DCT-II inverse est donnée par :

$$[x_{8\times8}] = [C^{II}_{8\times8}]^t \times [X_{8\times8}] \times [C^{II}_{8\times8}] \quad (5)$$

Mais l'utilisation directe de la transformation DCT-II exacte dans une chaine de compression d'images nécessite un grand nombre d'opérations, ce qui est contradictoire avec les applications temps réel et à faible coût de calcul. Ainsi la DCT réelle a été remplacée des approximations de DCT, utilisant que des 1 et des 0, au détriment d'une certaine perte de qualité [17 – 19].

3. Exemples d'expérimentations réalisées

Nous avons réalisé différents test pour trois (03) images différentes. En effet, nous avons commencé par capter trois (03) images différentes avec notre module Raspberry/CaméraHD représentant trois vues différentes que l'on a noté Image1, Image2 et Image3. Chacune de ces trois images est d'une résolution de 1280 × 720.

Pour chaque cas d'image captée nous avons effectuée plusieurs tests différents représentant différents taux de compressions différents (ou différents débits binaires) (Tableaux 1, 2 et 3). Dans chacun de ces 05 cas nous avons effectué la transmission et ensuite mesuré la qualité de l'image reconstruite en termes de PSNR (Peak Signal to Noise Ratio. Le PSNR est une métrique objective permettant d'évaluer la qualité d'une image reconstruite par comparaison avec une image originale.

- ⇨ $Mean\ Square\ error: MSE = \frac{1}{N}\sum_{n-1}^{N} x_n - \hat{x}_n$ (6)
- ⇨ $Peak\ Signal\ to\ Noise\ Ratio: PSNR = 10 * log_{10}(\frac{\max x^2}{MSE})$ (7)

RQ : max x ici c'est le max du niveau de gris en général on le pose 255. Car chaque pixel est généralement codé sur 8 bits.

Pour une image en couleur nous calculons la MSE pour chaque composante R, V et B séparément. Ensuite, on calcule la MSE moyenne. A partir de la MSE moyenne on calcule le PSNR de l'image en couleur :

$$Pour\ une\ image\ couleur: PSNRmoy = 10 \times log_{10}(3 \times \frac{\max x^2}{MSE_R + MSE_V + MSE_B}) \quad (7)$$

Nous remarquons dès lors que les PSNR obtenus (figure 26) pour des débits binaires relativement bas sont élevés ce qui signifie que la qualité des images compressées/transmises et décompressées sont bons. Ceci est bien évidemment dû à effet de canal de transmission négligeable. En effet, le WiFi utilisé est dans un réseau local (WLAN) avec des pertes de paquets négligeables voire inexistantes. Donc les PSNR obtenus sont dus uniquement à l'effet de la compression décompression. Afin de pouvoir évaluer correctement l'effet des pertes de paquets dû au canal de transmission il faut se mettre dans un environnement plus réaliste.

Figure 25 : exemples d’images acquises et transmises par le raspberry avec caméra

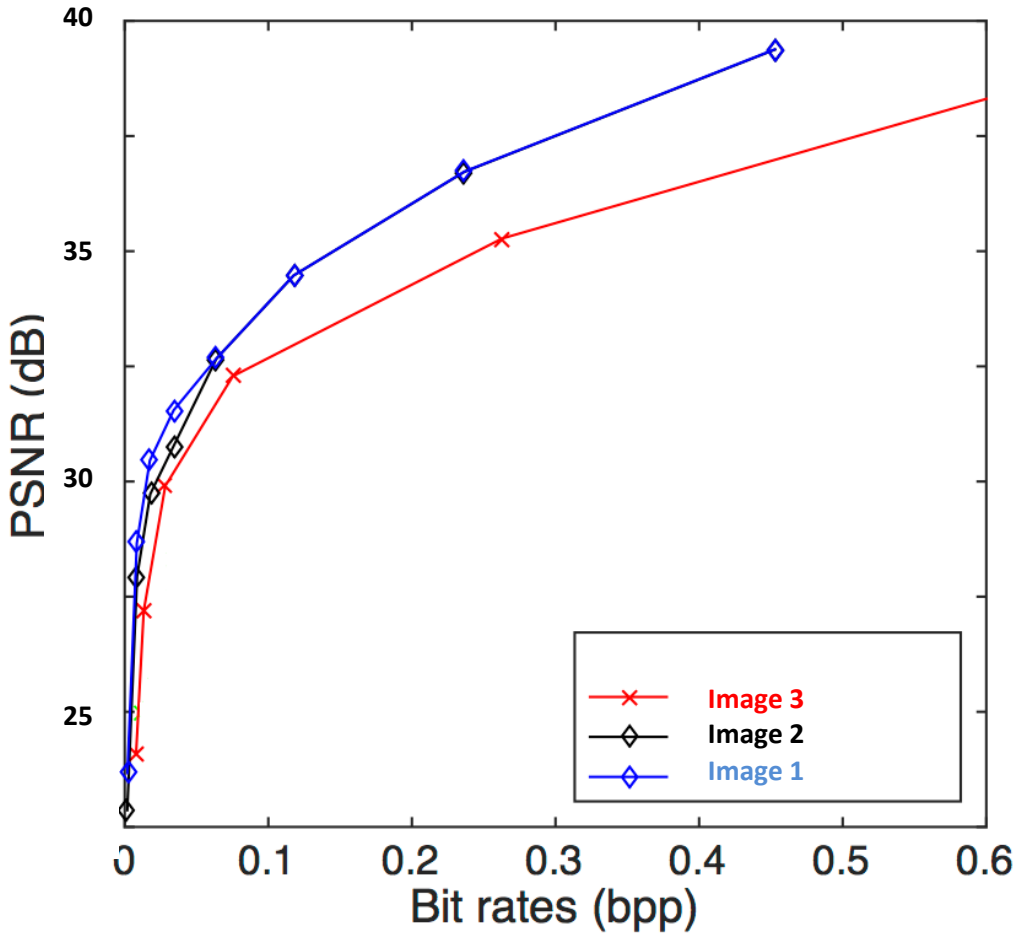

Figure 26 : Résultats de la compression des trois images en termes de PSNR vs bitrate

4. Conclusion

Dans ce chapitre nous avons effectué quelques tests et évaluations de notre réalisation. A cet effet, nous avons commencé par vérifier le bon fonctionnement de la partie captage d'images à l'aide du raspberry muni de sa caméra HD. Ensuite à l'aide d'un script en python nous avons pis en œuvre un programme simplifié de compression décompression d'images utilisant une DCT approximée. Le bitstream ainsi obtenu est transmis d'un raspberry à un autre en utilisant le protocole MQTT. Ceci nous a permis aussi d'évaluer l'effet de la compression/décompression et à moindre échelle l'effet de la transmission en utilisant des métriques objectives telles que la MSE et le PSNR.

Conclusion générale

Conclusion générale et perspectives

Dans ce projet de fin d'études nous sommes intéressés à la réalisation d'une plateforme représentant deux nœuds de capteur sans fil à faible consommation énergétique basés sur des modules raspberry. Ces modules fonctionnant en émetteur récepteur peuvent capter, compresser et transmettre des images avec une résolution assez élevée et une bonne qualité de reconstruction. L'objectif final de ce projet est la réalisation d'un système de surveillance de certains oiseaux migrateurs des zones humides d'Annaba en utilisant des réseaux de capteurs visuels sans fil. A cet effet, notre réalisation doit vérifier plusieurs conditions pour satisfaire cet objectif final tracé:

- Les nœuds capteurs ainsi réalisés doivent être efficace et à faible coût énergétique,
- Le réseau doit être flexible,
- Le prix total doit être relativement modeste.

La réduction de la consommation énergétique est assuré grâce à l'utilisation du WiFi comme moyen de transmission et au jpeg basé sur une DCT approximée comme outil de compression. Cependant, le choix d'un nœud capteur basé sur un raspberry pose toujours le problème de la consommation électrique. En effet, les modules raspberry sont relativement gourmands en consommation électrique même s'ils sont en veille. Ce qui rend difficile de les alimenter à partir de simples batteries. Un défi est donc de surmonter ce handicap dans le but de pouvoir les utiliser comme solutions finales à notre projet.

Les perspectives pour ce travail sont nombreuses. D'abord nos tests doivent prendre en considération des environnements réalistes. En effet, les pertes de paquets sont le plus gros problème que nous pouvons rencontrer et qui sont dus aux canaux sans fil hostiles à la transmission. Il est donc pertinent de protéger la transmission dans notre système en utilisant par exemple un codage canal contre les erreurs dues au canal de transmission.

Une autre perspective qui nous semble importante est de ne pas se restreindre aux images et utiliser plutôt de la vidéo. En effet, les raspberry sont des petits systèmes très puissants en comparaison avec les arduino. Leurs processeurs sont en mesure de gérer de la vidéo au lieu des simples images. De plus, une surveillance de la vie sauvage basée sur des séquences vidéo est beaucoup plus intéressante et plus significative.

Références bibliographiques

Références bibliographiques

[1] JF Clements, TS Schulenberg, MJ Iliff, D Roberson, TA Fredericks, BL Sullivan et CL Wood. The eBird/Clements checklist of birds of the world: v2015. URL: http://www. birds. cornell. edu/clementschecklist/download/IOC, 2015.

[2] https://www.iucnredlist.org/

[3] https://www.birdlife.org/news/tag/iucn-red-list

[4] A. Boulmaiz, N. Doghmane, S. Harize, N. Kouadria et D. Messadeg. The use of WSN (Wireless Sensor Network) in the surveillance of endangered bird species. A chapter of the book ‘’Ubiquitous Computing: Cyber-Physical Systems, Smart Cites, and Ecological Monitoring’’. will be published in Elsevier in 2019

[5] David Lusseau. Evidence for social role in a dolphin social network. Evolutionary ecology, vol. 21, no. 3, pages 357–366, 2007.

[6] Erick Stattner, Philippe Hunel, Nicolas Vidot et Martine Collard. Acoustic scheme to count bird songs with wireless sensor networks.In World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a, pages 1–3. IEEE, 2011.

[7] Li Da Xu, Wu He et Shancang Li. Internet of things in industries: A survey. IEEE Transactions on industrial informatics, vol. 10, no. 4, pages 2233–2243, 2014.

[8] Ladislav Ptacek, Lukas Machlica, Pavel Linhart, Pavel Jaska et Ludek Muller. Automatic recognition of bird individuals on an open set using as-is recordings. Bioacoustics, vol. 25, no. 1, pages 55–73, 2016.

[9] **Yang**, Shuang-Hua. Wireless sensor networks. Principles, Design and Applications. Signals and Communication Technology Springer 2014.

[10] Ke Xu, Yi Qu, Kun Yang A tutorial on the internet of things: From a heterogeneous network integration perspective IEEE Network 2016, pages 102-108

[11] Varsha Bapat, Prasad Kale, Vijaykumar Shinde, Neha Deshpande et Arvind Shaligram. WSN application for crop protection to divertanimal intrusions in the agricultural land. Computers and Electronics in Agriculture, vol. 133, pages 88–96, 2017.

[12] Amira Boulmaiz, Djemil Messadeg, Noureddine Doghmane et Abdelmalik Taleb-Ahmed. Robust acoustic bird recognition for habitat monitoring with wireless sensor networks. International Journal of Speech Technology, vol. 19, no. 3, pages 631–645, 2016.

[13] Amira Boulmaiz, Djemil Messadeg, Noureddine Doghmane et Abdelmalik Taleb-Ahmed. *Design and implementation of a robustacoustic recognition system for waterbird species using TMS320C6713 DSK*. International Journal of Ambient Computing and Intelligence (IJACI), vol. 8, no. 1, pages 98–118, 2017.

[14] Li Da Xu, Wu He et Shancang Li. Internet of things in industries: A survey. IEEE Transactions on industrial informatics, vol. 10, no. 4, pages 2233–2243, 2014.

[15] XBee/XBee-PRO S1 802.15.4 (Legacy), RF Modules. User guide DIGI https://www.digi.com/.../documentation/digidocs/PDFs/90000982....

[16] OV7670 datasheet. *web.mit.edu/6.111/www/f2016/tools/OV7670_2006.pdf*

[17] *Rao K, Yip P, Discrete Cosine Transform: Algorithms, Advantages, Applications, Boston: Academic Press,1990.*

[18] Ahmed N, Natarajan T, Rao K. Discrete Cosine transform.IEEE Trans. On Computers 1974; C-23(1):90-93.

[19] Britanak V, Yip PC, Rao K.Discrete Cosine and Sine Transforms. 1st Edition, Sept 2006 Elsevier.

[20] Cintra RJ, Bayer FM. A DCT approximation for image compression. IEEE Signal Process.Lett2011;18(10):579–582

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

[22] K. Mechouek, N. Kouadria, N. Doghmane, N. Kaddeche. Low Complexity DCT Approximation for Image Compression in Wireless Image Sensor Networks. Journal of Circuits, Systems and Computers 25(8) ·August 2016

[23] N. Kouadria, K. Mechouek, D. Messadeg, N. Doghmane. Pruned discrete Tchebichef transform for image coding in wireless multimedia sensor networks. AEU - International Journal of Electronics and Communications 74 ·February 2017

[24] N. Kouadria, K. Mechouek, S. . Harize, N. Doghmane. Region-of-interest based image compression using the discrete Tchebichef transform in wireless visual sensor networks. Computers & Electrical Engineering 73:194-208 ·November 2018

[25]https://www.digi.com/resources/documentation/digidocs/90001526/tasks/t_download_and_install_xctu.htm