Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid–Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option: *Réseaux et Systèmes Distribués (R.S.D)*

Thème

Implémentation et réalisation d'un algorithme de localisation « Centroid » dans les RCSF

Réalisé par :

- **Melle Djiral Amina**

Présenté le 22 juin 2015 devant le jury composé de MM.

- *Mr. BENAMAR Abdelkrim* *(Président)*
- *Mme. LABRAOUI Nabila* *(Encadreur)*
- *Mr. BEKARA Chakib* *(Examinateur)*
- *Mr. BELHOUCINE Amine* *(Examinateur)*

Année universitaire: 2014-2015

Remerciements

Nous remercions notre Dieu le tout puissant qui nous a accordé la volonté, la patience, et surtout la santé durant tout notre cursus.

*Je tiens à remercier en premier lieu à «**Mme LABRAOUI** », Mon encadreur pour sa sympathie, sa disponibilité, ses idées, ses précieux conseils ainsi que sa*
prise en charge des dépenses du matériel utilisé et ses encouragements qui m'ont permis de mener à bien de ce projet de fin d'études.

*Je remercie tout particulièrement « **Melle Mesmoudi Asma »** qui m'a beaucoup aidé et soutenue pour réaliser ce laborieux travail, merci pour sa patience, ses conseils et ses encouragements prodigués tout au long de ce mémoire.*

Nos remerciements vont aussi aux membres du jury pour l'honneur d'avoir voulu examiner et évaluer cette modeste contribution.

Nos sincères remerciements vont également à tous les enseignants qui nous ont formées durant ces cinq dernières années.

*Pour finir. Un grand **MERCI** à tous les membres de ma famille et spécialement mes parents qui m'ont soutenu tout au long de mes études et qui ont fait en sorte, par leur amour, leur affection et leur soutien financier, que je puisse avoir les meilleures conditions possibles pour continuer mes études et aller de l'avant. Qu'ils trouvent ici ma gratitude et mon amour pour eux.*

Dédicaces

Merci Allah (mon dieu) de m'avoir donné la capacité d'écrire et de réfléchir, la force d'y croire, la patience d'aller jusqu'au bout de mes rêves.

Je dédie ce modeste travail, *à tous ceux qui me sont chers,* A m*es parents* pour leur patience infinie, leur soutien sans faille et leur amour sans limite. Ce mémoire est autant le fruit de vos efforts que de mon travail.

A ma chère sœur Djawida et A mon frère Mohammed Amine pour tous leur encouragements.

A mon futur mari Tahraoui Mohammed pour son soutien, sa compréhension, son attention, sa patience et ses encouragements.

A toute ma famille Djiral et Hariat.

A mon encadreur " Mme Nabila LABRAOUI".

A toutes mes amies chacun par son nom et à toute personne qui me connait. A toute personne qui m'a aidé un jour à réussir jusque là, en espérant être toujours à la hauteur de leurs attentes et de leurs espérances.

Amina

Résumé

Dans ce mémoire, nous sommes focalisées sur le problème de l'absence d'information sur la position des noeuds d'un réseau de capteurs sans fil, au sein de l'environnement où ils sont déployés, les données récoltées peuvent s'avérer d'une utilité limitée. Alors le problème traité concerne la localisation de chacun de ces noeuds à partir de mesures de portée inter-capteurs telles que quelques capteurs dits ancres dont la position est connue. Nous avons proposé un algorithme nommé **Centroid** pour détermine l'emplacement des nœuds. L'objectif de ce travail est de créer une application qui détermine l'emplacement d'un noeud capteur ordinaire nommé « Blind » qui ne connait pas sa position, à partir d'autres nœuds statique qui sont référencé nommé « ancres » et de ces derniers doit être calculer la position du « Blind » selon l'algorithme de localisation Centroid.

La mise en place de notre algorithme exige des outils matériels et logiciels bien spécifiques puisque les capteurs sont des composants à ressources limitées. Pour cela nous avons utilisé le langage NesC qui minimise l'utilisation de la mémoire et TinyOs qui est un système d'exploitation léger.

Mots clés : Réseaux de capteurs sans fil, Problème de la localisation, Blind, Ancres.

Abstract

In this thesis, we focused on the problem in the lack of the position information of the nodes of a wireless sensor network, with in the environment where they are deployed; the collected data may become obsolete. Then the localization problem has been processed of each of these nodes from inter-sensor measurements such as few sensors called anchors whose position is known. We have proposed an algorithm called" **Centroid** "to determine the location of nodes.

The objective of this work is to create an application that determines the location of a static sensor node named "Blind" which does not know its position from other static nodes that are referenced named "anchors" and these last must be calculate the position of "Blind" according to the location algorithm Centroid.

The implementation of our algorithm requires very specific hardware and software tools since the sensors are components with limited resources. For this we used the NESC language that minimizes the use of memory and TinyOS is a lightweight exploratory system.

Keywords: Wireless sensor networks, Localization problem, Blind, Anchors.

ملخص

لقد ثم التركيز في مشروعنا النهائي على مشكل عدم توفر معلومات حول مكان وجود اللاقط أو(الحساس) في شبكات الاستشعار اللاسلكية، ضمن المنطقة التي يتم التوزيع فيها اللواقط، قد تكون البيانات التي تم جمعها من للاستخدام ذات أهمية محدودة بدون تحديد الموقع.في الواقع، معرفة موقف حدث التقاطها بواسطة جهاز استشعار هو من أهمية قصوى خاصة بالنسبة للتوجيه أو مراقبة التطبيقات الجغرافية.ادن المشكل المطروح للمعالجة مختص بتحد يد موقع كل لاقط عادي في الشبكة و بعض اللواقط المعلوم مكان تموقعها ولهدا يمكن القول عنها أنها مراجع.و لهدا تم اقترح خوارزمية المسماة " سنترويد " لتحديد موق ع اللواقط.الهدف من هدا العمل هو إنشاء تطبيق بحيث من خلاله يتم تحديد موقع لاقط استشعار عادي الذي لا يعرف مكان وجوده و يطلق عليه اسم"أعمى أو مكفوف". يتم تحديد موقع هدا الأخير من خلال وجود لاواقط أخرى ثابتة التي لنا علم بمكان تموقعها و تسمى " مراجع" و باستخدام هده المراجع و خوارزمية تحديد الموقع الجغرافي يتم حساب موقع اللاقط "الأعمى".

تنفيذ أنظمتنا تتطلب أجهزة معينة جدا و أدوات البرمجيات لان اجهزة الاستشعار مكونات ذات مواد محدودة. لهدا ثم استخدام لغة" ناس س " تقلل من استخدام الذاكرة و " تنيواس" و هو نظام تشغيل خفيف.

كلمات البحث: شبكات الاستشعار اللاسلكية، مشكلة تحديد موقع ، مكفوف، الواقط التي تم تحديد مواقعها.

.

Table de matières :

Introduction générale

Introduction Générale

Les réseaux de capteurs provoquent un intérêt croissant au sein des communautés scientifiques et industrielles depuis plusieurs années. Concrètement, un réseau de capteurs sans fil est composé d'un ensemble de capteurs intelligents qui sont capables de récolter et traiter des informations ayant aussi une capacité de communiquer entre eux. Le déploiement rapide, le coût réduit et la tolérance aux pannes des réseaux de capteurs sont des caractéristiques qui les rendent un outil appréciable dans plusieurs domaines d'applications d'activités militaires et civiles telles que la surveillance de l'environnement, l'habitat intelligent, la télémédecine, le suivi de population et la gestion des catastrophes.

Si leurs perspectives d'utilisation sont claires et attrayantes, les problématiques qu'engendrent ces réseaux n'en sont pas moins nombreuses. A priori, ils ne dépendent d'aucune infrastructure et les capteurs n'ont aucune information relative au réseau auquel ils appartiennent. De plus, étant construits de façon ad hoc, ces réseaux sont caractérisés par une auto-organisation.

Parmi ces problèmes cruciaux celui de la localisation: il s'agit d'attribuer une position géographique (exacte ou estimée) aux capteurs; une application telle que la surveillance des feux de forêt, par exemple, n'aurait pas de sens et d'importance. Donc, la localisation est un problème primordial dans de nombreuses applications où la connaissance de l'emplacement du capteur apporte une information très utile. La localisation est devenue une information à grande valeur ajoutée, au même titre que la sécurité ou la logistique, en plus, la connaissance des positions des capteurs dans l'environnement est souvent souhaitable, afin de pouvoir déterminer l'origine des flux de mesures collectées.

La localisation des capteurs est une parmi les principaux problèmes dans ce type de réseaux. En plus elle est une parmi les travaux de recherche antérieure dans le domaine des réseaux de capteurs puisque les capteurs peuvent être déployés arbitrairement, leur emplacement doit être connu afin de fournir des données significatives à l'utilisateur.

Pour les résoudre, une première solution est proposée : elle consiste à équiper tous les capteurs par un système de positionnement ou GPS (Global Positionning System).

Cependant, cette méthode s'avère être trop coûteuse sur le plan énergétique en raison de la consommation excessive en énergie des GPS. Parfois, cette solution est inadaptée à cause d'obstacles naturels ou autres qui rendent les connexions aux satellites impossibles. D'autres solutions ont été proposées les algorithmes de localisation (auto_localisation).

Le but de la localisation dans les WSNs est d'attribuer une position exacte ou estimée à chacun des capteurs du réseau nous utilisons la position de certain nombre d'entre eux s'appellent ancres dont la position est déjà connue.

La localisation dans les réseaux de capteurs s'établit en deux étapes, l'estimation des distances (avec mesure) les technologies (TDoA ,ToA ,RSSI) et la dérivation des positions (sans mesure)des noeuds qui respectent au mieux les distances inter-noeuds estimées .

Dans cette mémoire, nous présentons une étude détaillée sur la localisation dans les réseaux de capteurs et parmi les différentes solutions proposées pour résoudre ce problème nous proposons un algorithme de géo localisation Centroid dans le but d'apporter une estimation de positionnement.

Afin d'aborder l'ensemble de ces étapes, nous avons organisé ce rapport en 3 chapitres:

Le chapitre 1 : nous présentons une description générale des réseaux de capteurs sans fil ainsi que leurs caractéristiques, contraintes et spécificités.

Le chapitre 2 : est consacré à la problématique de la localisation et les classifications des algorithmes de localisation.

Le chapitre 3: constitue le cœur de notre travail, dans ce chapitre nous présentons le

La plateforme TinyOS et langage NesC avec une brève description de ces composants, l'implémentation et évaluation d'un algorithme Centroid. Par la suite nous donnons les résultats sous forme de graphes, effectuées pour obtenir des mesures pour voir la précision de méthode.

Nous terminerons ce mémoire par une conclusion générale conclusion est donnée pour résumer les apports essentiels de notre travail.

Chapitre 1

Généralités sur les RCSF

1. Introduction

Les réseaux de capteurs ont connu un grand essor ces dix dernières années. Ils interviennent dans tous les domaines de notre vie quotidienne et la rendent plus aisée. Sont généralement des composants de petite taille capable de récolter les données relatives à son environnement, de les traiter, puis, si nécessaire, de les transmettre et communiquer à des voisins. Ces composants sont appelés des nœuds capteurs et ils ont la capacité de s'auto-organiser pour former un réseau de capteurs sans fil(RCSF)

Dans un cas particulier les (RCSF) sont des réseaux sans infrastructure (réseaux ad-hoc), ils peuvent aussi être déployés pour exploiter diverses applications (environnementales, militaires, médicales, etc.).En effet, un réseau de capteurs peut être mis en place dans le but de surveiller une zone géographique plus ou moins étendue pour détecter un événement (par exemple: un changement de températures, des mouvements, des vibrations,…).

2. C'est quoi un capteur sans fil

2.1. Définition d'un capteur

Un capteur est un dispositif de taille réduit parmi les taches de capteurs de transformer une mesure physique observée en une mesure généralement électrique qui sera à son tour traduite en une donnée binaire exploitable et compréhensible par un système d'information.

Parmi les différents types d'evenement enregistrées par les capteurs, on peut citer entre autres : « la température, l'humidité, la luminosité, l'accélération, la distance, les mouvements, la position,la pression, la présence d'un gaz, la vision (capture d'image), le son, etc. . . »

Autremant dit, La notion de capteur s'est évoluée ces dernières années puisque leur domaine d'application s'est élargi. Les premiers capteurs n'étaient pas dédiés qu'à un unique type de mesure, les capteurs contemporains sont la combinaison de plusieurs dispositifs capables de mesurer différentes mesures physiques. En outre, à ces possibilités de mesures multiples, les capteurs actuels ont vu se gèrer des fonctionnalités qui leur permettent, en plus de l'enregistrement et de la détection d'événements mesurables, le traitement de ces données et leur communication vers un autre dispositif.

On parle alors de capteur intelligent,capable à la fois de mesurer des données ,stocker et les communiquer avec d'autres capteurs au sein d'un réseau, tel qu'il est caractérisé par sa capacité à effectuer une collecte des mesures, les traiter et à les communiquer au monde extérieur [1].

2.2. Architecture d'un capteur

Dans cette unité, nous distinguons les deux parties qui composent un capteur :

2.2.1. Architecture matérielle

La figure 1.1 est l'illustration la plus générale de l'architecture d'un capteur dit intelligent.

Figure 1. 1 : Architecture d'un capteur sans fil [1].

Cette architecture s'articule autour de quatre unités :

- ***l'unité de traitement*** : (Processeur, RAM et Flash) : c'est l'unité principale du capteur. Elle est généralement représentée par un processeur couplé à une mémoire vive. Son rôle est de contrôler le bon fonctionnement des autres unités. Sur certains capteurs elle peut embarquer un système d'exploitation pour faire fonctionner le capteur. Elle peut aussi être couplée à une unité de stockage, qui servira par exemple à y enregistrer les informations transmises par l'unité d'acquisition de données.
- ***l'unité d'acquisition ou de captage*** :(Capteur, ADC) elle permet la mesure des grandeurs physiques ou analogiques et leur conversion en données numériques. Elle est composée du capteur lui-même et de l'ADC1qui permet la conversion des données. Le capteur est chargé de récupérer les signaux analogiques qu'il transmet à

[1] ADC : Analog-to-Digital Converters

l'ADC qui a pour rôle de transformer et de communiquer les données analogiques en données numériques compréhensibles pour l'unité de traitement.

- ***l'unité de communication*** : (Radio, Antenne) elle a pour fonction de transmettre et recevoir l'information. Elle est équipée d'un couple émetteur/récepteur pour communiquer au sein du réseau. Il existe cependant d'autres possibilités de transmission (optique, infrarouge, etc. . .).
- ***l'unité d'alimentation :*** (Batterie ou une pile alimentant le capteur) c'est un élément fondamental de l'architecture du capteur, c'est elle qui fournit en énergie toutes les autres unités. Elle correspond le plus souvent à une batterie ou une pile alimentant le capteur, dont les ressources limitées en font une problématique propre à ce type de réseau puisque ces derniers sont généralement déployés dans des zones non accessibles. La réalisation récente d'unité d'alimentation à base de panneaux solaires tente d'apporter une solution pour prolonger sa durée de vie [3].

Par ailleurs, un capteur peut être associé par d'autres unités additionnelles. Citons, parmi ces unités la possibilité d'ajouter une unité de localisation, tel qu'un GPS (Global Positioning System), une unité de mobilité pour assurer la mobilité du capteur, ou une unité spécifique de capture comme une caméra pour de l'acquisition vidéo, ou un générateur d'énergie (exemple cellules solaires).

Dans le cas de l'utilisation d'un GPS ou d'une caméra de surveillance, il est intéressant de noter que leur utilisation dans les capteurs actuels a un coût non négligeable en termes de consommation énergétique, qui peut réduire grandement la durée de vie d'un capteur équipé de ce type de dispositifs.

2.2.2. Architecture logicielle

La contrainte énergétique des capteurs exige l'utilisation de systèmes d'exploitation légers tels que TinyOS [22] ou Contiki [4]. Cependant, TinyOS reste toujours le plus utilisé et le plus populaire dans le domaine des RCSF. Il est libre et utilisé par une large communauté de scientifiques dans des Simulations pour le développement et le test des algorithmes et protocoles réseau.

2.3. Les types de capteurs

Il existe actuellement un grand nombre de capteurs, avec des fonctionnalités diverses et variées. Tous ces différents capteurs ne pourraient être décrits ici, cependant une liste exhaustive peut être trouvée sur le site The Sensor Network Museum [23].

La plupart des capteurs dépendent de l'application pour lesquels ils ont été conçus (capteur aquatique, sous-terrain, etc. . .). Il est plus intéressant de décrire les capteurs les plus utilisés et leur évolution au cours du temps.

En l'occurrence, la figure 1.2 illustre l'évolution des capteurs au cours de ces 20 dernières années. Cette représentation met en avant l'importance des travaux de recherche de l'université de Berkeley dans l'essor des réseaux de capteurs, surtout sachant que l'entreprise Xbow (aussi appelé Crossbow) qui fait jusqu'à aujourd'hui office de référence dans la fabrication de capteurs est née au sein de la célèbre université californienne.

Les capteurs fabriqués par Xbow au cours des dix dernières années (famille de capteurs Mica et Telos) sont sans aucun doute les plus utilisés dans les expériences et travaux de recherche. Ces capteurs sont capables de mesurer plusieurs métriques (température, humidité, luminosité, etc. . .) et s'articulent pour la plupart d'entre eux autour du Chipcon CC2420 qui est devenu le standard au niveau des modules de transmission utilisant le protocole de communication IEEE 802.15.4.

Figure 1. 2: Quelques exemples de capteurs [1].

2.4. Les caractéristiques de capteurs

- **Etendue de mesure** : Valeurs extrêmes pouvant être mesurée par le Capteur.
- **Résolution :** Plus petite variation de grandeur mesurable par le capteur.
- **Sensibilité** : Variation du signal de sortie par rapport à la variation du Signal d'entrée.
- **Précision :** Aptitude du capteur à donner une mesure proche de la valeur Vraie.
- **Rapidité :** Temps de réaction du capteur. La rapidité est liée à la bande Passante.
- **Linéarité :** Représente l'écart de sensibilité sur l'étendue de mesure [24]

3. Les Réseaux de capteurs sans fil (RCSF)

3.1. Description des réseaux de capteurs

Le réseau de capteur sans fil (Wireless Sensor Network ; WSN, ou RCSF) est un type spécial de réseaux ad hoc .Se composent généralement d'un grand nombre de petits dispositifs, qui communiquent entre eux via des liens radio pour le partage d'information et le traitement coopératif. Ces dispositifs sont déployés aléatoirement dans une zone d'intérêt pour superviser ou surveiller des phénomènes divers. Après le déploiement initial, les capteurs peuvent s'auto-organiser en une infrastructure réseau appropriée, souvent en mode multi-sauts. Les données collectées par ces capteurs sont acheminées directement ou via un routage multi-sauts à un nœud considéré comme "point de collecte", appelé station de base (sink). Cette dernière peut être connectée à une machine puissante via internet ou par satellite. En outre, l'usager peut adresser ses requêtes aux capteurs en précisant l'information d'intérêt.

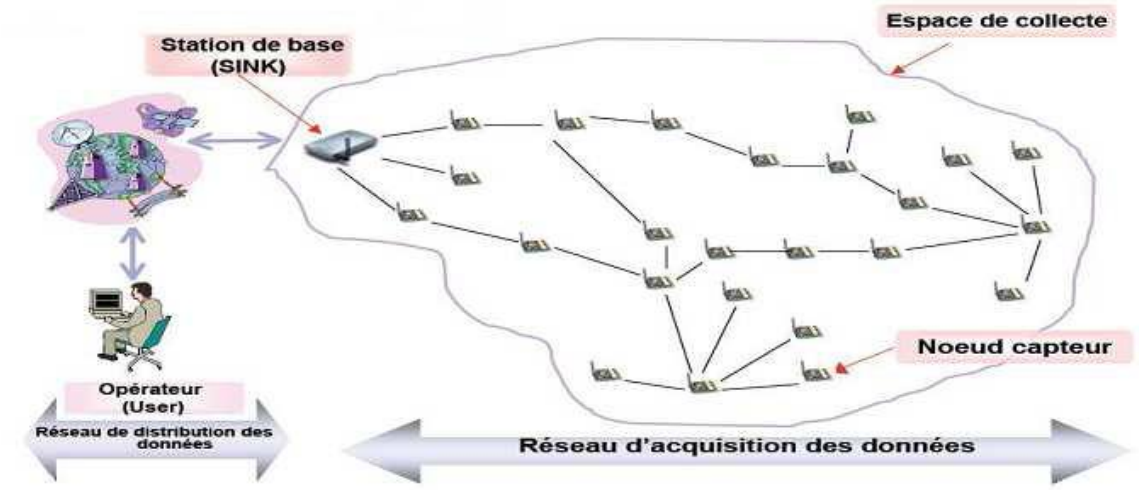


Figure 1. 3: Architecture de communication d'un RCSF [2].

3.2. Similarité et différence entre (RCSF) et réseaux Ad-hoc (MANETs)

Les réseaux de capteurs sont considérés comme un type particulier des réseaux ad-hoc et sont significativement différents des réseaux MANET (Mobile Ad-hoc NETwork) traditionnels. Ce tableau résume les similarités et les différences entre les réseaux de capteurs sans fil et les MANETs:

Réseau de capteurs sans fils	**Réseau MANET (Ad-Hoc)**
1 .Utilisation d'un médium sans fil.	1. Utilisation d'un médium sans fil.
2. Réseau auto-configurable	2. Réseau auto-configurable
3. Topologie dynamique	3. Topologie dynamique
4.Mode de transmission: many to one	4.Mode de transmission: one to one (any to any)
5. Utilisation du broadcast	5. Communication point à point
6. La mobilité des noeuds est restreinte.	6. Mobilité des noeuds.
7. Grand nombre de noeuds (de l'ordre de mille) nœuds n'ayant pas tous un ID	7. Nombre de noeuds moyen (de l'ordre de cents) Notion d'ID
8. Des petits noeuds plus susceptibles aux pannes, avec moins de capacité de	8. Des noeuds ayant plus de capacité de traitement et de stockage.

traitement et de stockage.	
9. Noeuds collaborent pour remplir un objectif Commun	9. Chaque noeud a son propre objectif
10. Portée radio est de l'ordre de 40 m	10. Portée radio est dans l'environs de 250 m
11. Energie (facteur déterminant)	11. Débit majeur

Tableau 1. 1: Similarités et les différences entre (RCSF) et réseaux Ad-hoc.

3.3. Les principales caractéristique des réseaux de capteurs sans fil

- **absence d'infrastructure** : les réseaux Ad-hoc en général, et les réseaux de capteurs en particulier se distinguent des autres réseaux par la propriété d'absence d'infrastructure préexistante et de tout genre d'administration centralisée. Les hôtes mobiles sont responsables d'établir et de maintenir la connectivité du réseau d'une manière continue.
- **taille importante** : un réseau de capteurs peut contenir des milliers de nœuds.
- **interférences :** les liens radio ne sont pas isolés, deux transmissions simultanées sur une même fréquence, ou utilisant des fréquences proches, peuvent interférer.
- **topologie dynamique** : les capteurs peuvent être attachés à des objets mobiles qui se déplacent d'une façon libre et arbitraire rendant ainsi la topologie du réseau fréquemment changeante.
- **sécurité physique limitée** : les réseaux de capteurs sans fil sont plus touchés par le paramètre de sécurité que les réseaux filaires classiques. Cela se justifie par les contraintes et limitations physiques qui font que le contrôle des données transférées doit être minimisé.
- **bande passante limitée** : une des caractéristiques primordiales des réseaux basés sur la communication sans fil est l'utilisation d'un médium de communication partagé. Ce partage fait que la bande passante réservée à un nœud est limitée.
- **contrainte d'énergie, de stockage et de calcul** : la caractéristique la plus critique dans les réseaux de capteurs est la modestie de ses ressources énergétiques car chaque capteur du réseau possède de faibles ressources en

termes d'énergie (batterie). Afin de prolonger la durée de vie du réseau, une minimisation des dépenses énergétiques est exigée chez chaque nœud. Ainsi, la capacité de stockage et la puissance de calcul sont limitées dans un capteur [25].

3.4. Applications des RCSF

Les RCSF mobiles peuvent être constitués des différents types de capteurs capables de surveiller une variété de paramètres, tels que, la température, l'humidité, la pression, le mouvement des véhicules, le niveau de bruit, la présence ou l'absence d'objets, etc. …,et le domaine d'applications des RCSF est très varié. Ces réseaux sont présents dans le domaine militaire, sécurité civile, médical, transport, environnemental, commerciales,…etc

- **Applications militaires :** Les RCSF permettent la détection des mouvements ennemis sur un champ de bataille ou bien de tracer leurs mouvements. la détection d'agents chimiques ou bactériologiques, De façon analogue, ils peuvent permettre la détection d'intrusion ou de cambriolage dans le domaine de la sécurité civile.
- **Applications environnementales :** Les réseaux de capteurs permettent la détection des incendies, la surveillance des catastrophes naturelles, la détection des pollutions et le suivi des écosystèmes, le suivi des mouvements des oiseaux, des animaux et des insectes
- **Applications agricoles :** Les capteurs peuvent être semés avec les graines. Ainsi, les zones sèches seront facilement identifiées et l'irrigation sera donc plus efficace et économique.
- **Applications médicales :** Les RCSF permettent par exemple la surveillance de l'état de santé des patients à distance (rythme cardiaque, pression du sang, etc), l'identification des allergies et des médicaments administrés aux patients, la localisation des docteurs et des patients dans l'hôpital, etc.
- **Applications de transport :** Les RCSF permettent la surveillance des réseaux ferroviaires ou routiers ainsi que d'éventuelles intrusions comme exemple la gestion de trafic, la déformation de structure, les capteurs de pression des pneus… .

- **Applications domestiques :** En plaçant, sur le plafond ou dans le mur, des capteurs, on peut économiser l'énergie en gérant l'éclairage ou le chauffage en fonction de la localisation des personnes.
- **Applications commerciales :** Parmi les domaines dans lesquels les réseaux de capteurs ont aussi prouvé leur utilité, on trouve le domaine commercial. Dans ce secteur on peut énumérer plusieurs applications comme : la surveillance de l'état du matériel, le contrôle et l'automatisation des processus d'usinage, etc.

Figure 1. 4 : Quelques domaines d'applications des RCSF [6].

3.5. Facteurs et contraintes conceptuelles des RCSF

La conception et la mise en place des RCSF sont influencés par plusieurs contraintes qui peuvent être des contraintes conceptuelles ou matérielles .Ces facteurs importants servent comme directives pour le développement des algorithmes et protocoles utilisés dans les réseaux de capteurs ;ils sont considérés également comme métrique de comparaison de performances entre déférents travaux dans le domaine parmi ces problématiques on cite :

3.5.1. Contraintes conceptuelles

La conception des RCSF, leurs protocoles et algorithmes sont guidés par plusieurs facteurs :

- **Tolérance aux fautes :**

Le réseau doit être capable des maintenir ses fonctionnalités sans interruption en cas de défaillance d'un de ses capteurs. Cette défaillance peut être causée par une perte d'énergie, par un dommage physique ou par interférence de l'environnement. Le degré de tolérance dépend du degré de criticité de l'application et des données échangées.

- **Scalabilité (passage à l'échelle) :**

  Une des caractéristiques des RCSF est qu'ils peuvent contenir des centaines voir des milliers de nœuds capteurs. Le réseau doit être capable de fonctionner avec ce nombre de capteurs tout en permettant l'augmentation de ce nombre et la concentration (densité) des nœuds dans une région.

- **Coût de production :**

  Le coût des RCSF, qui sont constitués d'un grand nombre de nœuds, dépend de celui d'un seul nœud qui ne doit pas, par conséquent, être cher.

- **Environnement :**

  Les nœuds capteurs doivent être conçus d'une manière à résister aux différentes et sévères conditions de l'environnement : forte chaleur, pluie, humidité…

- **Média de transmission :**

  Les nœuds communicants sont reliés sans fil. Ce lien peut être réalisé par radio, signal infrarouge ou un média optique [3].

- **Contrainte d'énergie, de stockage et de calcul** :

  La caractéristique la plus critique dans les réseaux de capteurs est la modestie de ses ressources énergétiques car chaque capteur du réseau possède de faibles ressources en termes d'énergie, de calcul et de stockage. Afin de prolonger la durée de vie du réseau, une minimisation des dépenses énergétiques est exigée chez chaque nœud [5].

- **Agrégation de données :**

  Dans les RCSF, les données produites par les noeuds capteurs voisins sont corrélées spatialement et temporellement. Ceci peut engendrer la réception par la station de base d'information redondante. Réduire la quantité d'informations redondantes transmises par les capteurs permet de réduire la consommation d'énergie dans le réseau ainsi d'améliorer sa durée de vie. L'une des techniques utilisée pour réduire la transmission d'informations redondantes est l'agrégation des données, appelée aussi fusion des données.

3.5.2. Contraintes matérielles:

Un nœud capteur est constitué des composantes principales suivantes [25]:

- **Unité de détection :**
  Est composée de deux sous unités, capteurs (sensors) et ADC (Analog to Digital Converters) qui permet de convertir le signal produit par le capteur, sur la base du phénomène observé, en un signal digital qui est par la suite fournit a l'unité de calcul.

- **Unité de calcul :** gère les procédures permettant au nœud de collaborer avec les autres nœuds pour accomplir la requête assignée.
- **Transceiver unit:** connecte le nœud au réseau.
- **Unité d'énergie :** est une composante importante du nœud capteur.
- **Location finding system:** fournit des informations sur la localisation requise par les techniques de routage.
- **Mobilizer :** est nécessaire si le nœud capteur doit être déplacé pour accomplir la requête assignée.

Ces composantes nécessitent d'être regroupées dans un module convenable avec la contrainte de la taille qui ne doit pas être très petite, et ne dépasse pas quelques centimètres.

4. Conclusion

Nous avons présenté dans ce chapitre quelques définitions de base sur les RCSF, leurs spécificités et les concepts nécessaires à la compréhension des réseaux de capteurs. Dans le chapitre qui suit, nous présentons la localisation conçue pour les RCSF, et ainsi les classifications des algorithmes de localisation.

Chapitre 2

La localisation dans les RCSF

1. Introduction

Le rôle des réseaux de capteurs est souvent la surveillance d'un espace donné. La localisation est une tache importante des futurs services informatiques, particulièrement adaptées aux réseaux de capteurs sans fil. Dans le but d'attribuer une position exacte ou estimée à chacun des capteurs du réseau.

La localisation dans les réseaux de capteurs déployés de manière aléatoire consiste à déterminer les coordonnées géographiques des différents capteurs. La localisation des nœuds est très nécessaire car elle est la première tache exécutée après le déploiement des nœuds et après la détection d'un événement la question qui se pose et où se trouve cet événement !!!, donc c'est ca l'utilité de la localisation. Les systèmes de localisation existants sont nombreux et utilisent plusieurs voies technologiques, ils sont très différents les uns des autres et répondent en général à des besoins différents (Localisation dans une pièce ou à l'échelle de la planète, localisation précise ou imprécise, localisation d'un seul ou plusieurs objets, localisation statique ou suivi de cible, ...).

2. Les principaux systèmes de localisation

- ✓ **Le System GPS (Global Positioning System)**

Est issu d'un programme militaire Américain débuté en 1958 : ce programme visait a obtenir la position ou la localisation d'un mobile n'import ou dans le monde a partir d'émissions radio en provenance d'un satellite [7]

Le système de géo localisation GPS comprend au moins 24 satellites artificiels répartis sur six orbites situés à 20 000 km d'altitude qui effectuent une rotation terrestre en douze heures. Quatre satellites suffisent pour connaître sa position dans les trois dimensions

- ✓ **Le System Galileo : (CNES et ESA)**

Galileo est un projet européen de système de positionnement par satellites (radionavigation), Ce système est en phase de test depuis 2004, commencera à être utilisable en 2010 et pleinement en 2013.destiné à supprimer la dépendance de l'Europe en matière spatiale, et notamment vis-à-vis du système américain GPS (Global Positioning System) [8].

- ✓ **Le System Glonass**

Signifie en russe GLObal'naya NAvigatsionnaya Sputnikovaya Sistema soit système mondial de navigation par satellite a débuté en 1976 avec pour but une couverture mondiale en 1991. C'est un système de positionnement développé par l'actuelle Union Soviétique et contrôlé pour le gouvernement russe par l'agence spatiale russe. Il est une alternative et complémentaire du GPS américain et du futur Galileo européen [8].

- ✓ **Le système Beidou :**

Le système de navigation et de positionnement par satellite chinois Beidou (BDS), signifiant « le Grande Ourse », consiste en deux constellations de satellites et a pour objectif de permettre l'indépendance de la Chine vis-à-vis des systèmes de navigations étrangers (GPS américain, Galileo européen et GLONASS russe).L'évolution future qui rendra le système mondiale.il sera totalement opérationnel en 2020[26].

- ✓ **Le system IRNSS (Indian Regional Navigation Satellite System)**

Est une proposition de système de positionnement par satellites qui serait construit et contrôlé par le gouvernement Indien. Un but de contrôle complet de la part du gouvernement Indien a été cité, ce qui implique que toutes les parties du projet soit construites en Inde. Le système proposé consisterait en 7 satellites et une partie au sol (grounds segment). Trois des satellites seront placés en orbite géostationnaire et les quatre autres en orbite géosynchrone inclinée de 29 degrés par rapport au plan de l'équateur

3. La localisation dans les RCSF

Dans bon nombre d'applications, un événement détecté par un capteur n'est utile que si une information relative à sa localisation géographique est fournie. C'est le cas de la surveillance des feux de forêt ou de troupes ennemies dans un contexte militaire. Sans cette information, ces applications n'auraient aucun sens. Il s'agit donc de déterminer pour chacun des capteurs sa position. La localisation des capteurs est une des principaux problèmes dans ce type de réseaux et nombreuses sont les solutions qui ont été proposées pour le résoudre, et chaque méthode a ses avantages et ses inconvénients. Parmi ces approches nous citons :

3.1. 1ere approche: GPS (Global Positioning System)

Cette solution consiste à accorder chaque nœud par un GPS pour spécifier la localisation des nœuds. Mais il est disponible seulement en extérieur « outdoor », et encore si aucun obstacle ne vient obstruer le champ de vue des récepteurs : le fonctionnement sous un feuillage dense, ou dans des villes aux rues étroites, n'est pas possible, où seulement dans de très mauvaises conditions. De plus il est particulièrement coûteux, tant en ce qui concerne le matériel qui est dupliqué en nombreux exemplaires dans un réseau à forte densité de capteurs. De plus, la réception du signal est très gourmande en énergie, ce qui n'est pas compatible avec les problématiques de gestion de durée de vie des batteries.

3.2. 2éme approche : utilisation des algorithmes de localisation (auto-localisation)

La plupart des algorithmes de localisation existants sont constitués de deux phases :

3.2.1. Estimation de la distance

Plusieurs technologies permettent à un capteur de mesurer la distance qui le sépare d'un capteur voisin (ToA, TDoA, RSSI) ou bien de mesurer l'angle qu'il forme avec celui-ci (AoA).

Certaines sont des mesures réalisées par le système (puissance du signal RSSI, temps de vol TOA , AOA, nombre de sauts …) et autres technologies afin de mesurer les distances ou les angles entre deux capteurs voisins. Grâce à cette capacité de mesure, un capteur pourra, sous certaines conditions, obtenir sa position exacte. Tandis que d'autres sont du domaine des hypothèses relatives au réseau (connectivité…).nous citons quelque technologies :

- ✓ **Temps d'arrivée**

La technologie ToA (Time of Arrival) suppose que les noeuds du réseau sont synchrones.

La distance qui sépare deux capteurs se déduit de la vitesse de propagation du signal et de la différence entre les dates d'émission et de réception du message. Cette technologie est celle utilisée par le système GPS (Global Positioning System) [9].

- ✓ **Différence des temps d'arrivée**

La technologie TDoA (Time Difference of Arrival)[10] se base sur la différence des

dates d'arrivée d'un ou plusieurs signaux et suppose également que la vitesse de propagation des signaux est connue. Cette technologie s'applique dans les cas suivants :
– un émetteur envoie des signaux de natures différentes (par exemple, l'ultrason, l'onde radio,...) à un récepteur ;
– un récepteur reçoit des signaux d'une même nature d'au moins trois émetteurs ;
– un émetteur envoie un signal reçu par au moins trois récepteurs (dans ce dernier cas une vue globale des signaux sera connue).

- ✓ **Puissance du signal**

La technologie RSSI (Received Signal Strength Indicator) [11] propose une solution élégante pour l'estimation des distances dans les réseaux de capteurs. Elle considère la perte de puissance d'un signal entre son émission et sa réception. Cette perte varie en fonction de la distance entre les deux capteurs : plus les capteurs sont éloignés (resp. proches), plus la perte est importante (resp. faible). Cette perte sera alors traduite en une distance.

- ✓ **Angle d'arrivée**

La technologie AoA (Angle of Arrival) [12] calcule l'angle formé entre deux capteurs. Chaque capteur est doté d'antennes orientées de sorte à déduire l'angle qu'il forme avec un voisin lorsque ce dernier lui envoie un signal. Cet angle est reporté par rapport à un axe propre au capteur. Toutefois, un capteur peut être équipé d'une boussole et, dans ce cas, l'angle sera reporté sur un des axes nord, sud, est ou ouest.

3.2.2. Dérivation des positions

La dérivation des positions consiste à calculer les positions finales de chaque noeud capteur en utilisant un des algorithmes de localisation. Chaque algorithme utilise une méthode de calcul qui dépend de la technique d'estimation de distance utilisée. Nous classifions ces méthodes en trois catégories :

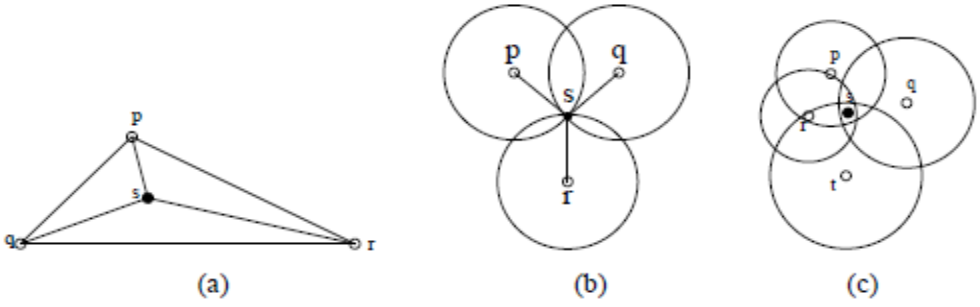


Figure 2. 1: (a) triangulation, (b) trilatération, (c) multilatération[13].

- ✓ **Trilatération**

La trilatération [13] est la méthode la plus simple pour déterminer les positions des capteurs en utilisant la géométrie des triangles. Elle est fondée sur le même principe qu'un système GPS. Elle consiste à s'appuyer sur trois points de référence, c'est à dire des noeuds dont on connaît la position (les ancres), et sur les distances qui les séparent du noeud dont on cherche à estimer la position. Cette dernière correspond alors au point d'intersection des trois cercles. Voir la Figure 2.1.

- ✓ **Triangulation**

La triangulation est une technique permettant de déterminer la position d'un point en mesurant les angles entre ce point et d'autres points de référence dont la position est connue (les ancres), et ceci plutôt que de mesurer directement la distance entre les points. Ce point peut être considéré comme étant le troisième sommet d'un triangle dont on connaît deux angles et la longueur d'un côté. Comme présentée la Figure 2.1.

- ✓ **Multilatération**

La multilatération [13] a le même principe que la trilatération, en utilisant plus que trois points de référence (ancres). La position d'un noeud est calculée en résolvant l'intersection de plusieurs hyperboles basées sur la différence des temps d'arrivée TDOA. Soit une cible *i*, connaissant les positions(x_a , y_a) de *m* ancres (1 ≤ a≤ *m*) ainsi que les distances d_{ia} , ou d_{ia} représente la distance euclidienne entre *i* et l'ancre *a*. Ayant ces informations et pour calculer la position(x_i , y_i) [1] de la cible *i* nous formons le système linéaires suivant :

$$(\boldsymbol{x_{1^+}\, x_i})^2 + (\boldsymbol{y_{1^+}\, y_i})^2 = \boldsymbol{d_{i1}}^2$$
$$(\boldsymbol{x_{2^+}\, x_i})^2 + (\boldsymbol{y_{2^+}\, y_i})^2 = \boldsymbol{d_{i2}}^2$$
$$(\boldsymbol{x_{3^+}\, x_i})^2 + (\boldsymbol{y_{3^+}\, y_i})^2 = \boldsymbol{d_{i3}}^2$$
$$\cdot$$
$$\cdot$$
$$(\boldsymbol{x_{m^+}\, x_i})^2 + (\boldsymbol{y_{m^+}\, y}\ _{\prime})^2 = \boldsymbol{d_{im}}^2$$

4. Classifications des algorithmes de localisation (self-localization)

Les algorithmes de localisation peuvent être classifiés suivant plusieurs critères. Selon le critère de la dépendance des techniques de mesures, ces algorithmes peuvent être classés en deux catégories: Les algorithmes range-based et les algorithmes range –free.

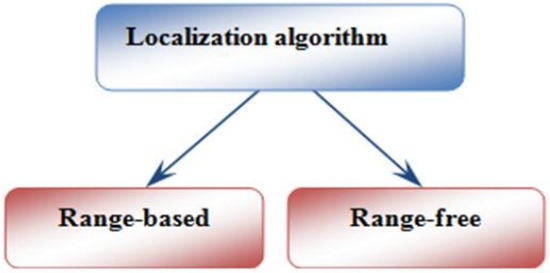


Figure 2. 2: Classification des algorithmes de localisation

4.1. Les algorithmes de localisation range-based (avec mesure)

Les méthodes range-based utilisent les technologies ToA, RSSI, AoA afin de mesurer les distances ou les angles entre deux capteurs voisins. Grâce à cette capacité de mesure, ces méthodes sont plus précises. Mais parmi ces inconvénients que certaine technologies comme (ToA, DoA, AoA) sont très couteux ou sensible a l'environnement comme RSSI .Parmi les algorithmes de localisation range-based, nous citons :

Dynamic fine-grained [10], **APS** using AoA [12] et **MDS** [14].

4.2. Les méthodes de localisation range-free (sans mesure)

Ces méthodes ne calculent jamais la distance entre voisins. Elles utilisent d'autres informations telles que la connectivité pour identifier la position des noeuds, elles supposent que le déploiement des noeuds respecte certaines contraintes et proposent des calculs plus ou moins complexes pour évaluer la position. Dans cette famille des méthodes, la position n'est pas trop précise mais elle est adapté pour certaine application nous citons dans les cas de feux de forêt , Dans un objectif de simplicité et de réduction du coût ,et cette catégorie vise d'amélioré la localisation d'après ces

observations on se concerne dans notre travaille par cette famille de range-free parmi les algorithmes utilisé dans cette famille nous citons : **APS** [15] avec ses trois méthodes (Dv-Hop, Dv-Distance et Euclidian distance), **Convex position** [16] et **APIT** [17] et notre algorithme implémenté dans le chapitre suivant **Centroid** [18] .

5. Critères des algorithmes de localisation

Un algorithme de localisation est évalué selon une liste de critères dont nous citons :

• **Précision de la localisation** : Erreur entre la position réelle et la position estimée.
• **Cout énergétique de la localisation** : Dans les WSN, une gestion de l'énergie très économique est nécessaire et comme le facteur dominant de la consommation d'énergie est la communication radio, il faut trouver un algorithme qui communique le moins possible via la radio.
• **Robustesse au bruit** : Il faut analyser comment un algorithme se comporte face au bruit rencontre dans les mesures de distances avec les voisins.
• **Passage à l'échelle :** L'algorithme est-il optimiser pour un réseau à grande échelle (algorithme implémentable de façon distribuée ou non).
• **Tolérance à la basse connectivite** : Dans le cas d'un réseau ou chaque noeud ne sait communiquer qu'avec un petit nombre de ses voisins, les performances de l'algorithme.
• **Réactivité du système :** rapidité du processus de localisation des noeuds. Ce paramètre est important si on veut gérer la mobilité des noeuds dans le réseau [19].

6. Conclusion

Dans ce chapitre, nous avons présenté des méthodes de localisations et les étapes pour construire un système de localisation conçu essentiellement pour les réseaux de capteurs sans fil afin de permettre aux capteurs de calculer leurs positions. Certaines méthodes s'appuient uniquement sur les informations des ancres, d'autres sur des mesures captés par des antennes. Les méthodes conçues doivent veiller à prolonger la durée de vie des capteurs tout en fournissant une bonne estimation.
Nous avons présenté les algorithmes de localisation range-based qui ont des avantages comme la précision et qui ont aussi des inconvénients comme le calcule complexe, le cout …..Range-free comme l'autre catégorie a des inconvénients la position qui n'est pas trop précise mais a aussi des avantages comme sa simplicité et cette catégorie vise d'améliorer la localisation. Nous focalisons notre travail par l'implémentation d'un

algorithme de cette famille de range-free. Dans le chapitre suivant nous allons focus et traiter un algorithme de localisation Centroid.

Chapitre 3

Implémentation et évaluation de l'algorithme de localisation Centroid

1. Introduction

Parmi les principaux problèmes dans les réseaux de capteurs sans fil la localisation. Puisque dans la majorité des applications déployées des nœuds de manière aléatoire dans des zones hostiles là où l'accès humain est difficile voir impossible, lorsque un événement capté par un capteur n'est utile que si une information relative à sa localisation géographique est fournie. C'est le cas de la surveillance des feux de forêt ou de troupes emmenais dans domaine militaire. Désormais, sans l'information de la localisation, ces applications n'auraient aucune importance.

L'algorithme de localisation Centroid est l'un des algorithmes d'estimation de position proposé pour les réseaux de capteurs sans fil.

Dans ce chapitre, nous avons développé une application sur les captures réelles pour illustrer l'apport de la localisation des nœuds selon l'algorithme de Centroid. Pour concrétiser cet apport, nous avons évalué un réseau de capteurs dans plusieurs parties. Dans ce qui suit, nous détaillons les éléments matériels et logiciels qui ont une relation avec notre application.

2. Présentation de l'algorithme Centroid

Dans ce chapitre, nous présentons et implémentons l'algorithme proposé par Bulusu et Heidemann, J., Estrin [18].Les hypothèses de départ de Centroid .La méthode se base sur un quadrillage de noeuds de référence (ancres) émettant des nœuds référencé son déploiement peut être manuelle ou aléatoire pour calculer la position du nœud ordinaire « Blind » qui ne connais pas sa position. Selon une métrique de connectivité comme montré (équation 1). Les facteurs sont caractérisés par : $\sum_{i=1}^{N} \mathbf{x}_i$: la somme des x des ancres émis par le capteur i, $\sum_{i=1}^{N} \mathbf{y}_i$: la somme des y émis par le capteur i et $\boldsymbol{N}$ **:** le nombre des ancres émis la valeur (x, y).

$$\boldsymbol{PE} = \left[\frac{\sum_{i=1}^{N} \mathbf{x}_i}{\boldsymbol{N}}, \frac{\sum_{i=1}^{N} \mathbf{y}_i}{\boldsymbol{N}}\right] \qquad \textbf{(1)}$$

Les coordonnées des ancres retenues sont utilisées pour calculer le centre de gravité de la zone et la position que le noeud considèrera comme la sienne. Sur la figure 3.1, une réalisation de Centroid est illustrée avec uniquement les balises retenues représentées par un point. La position estimée est représentée par un plus.

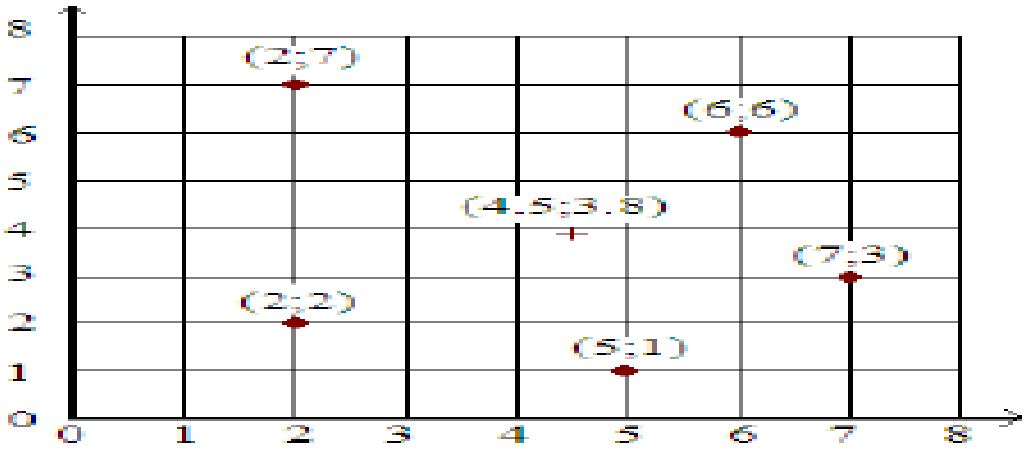


Figure 3. 1 : Exemple de localisation avec Centroid

La validation par un prototype en environnement extérieur a impliqué les cinq capteurs placées aux coins d'un carré de 10 m X 10 m et a mené à une erreur moyenne de 2m.

3. Environnement de travail

L'implémentation de l'algorithme nécessite l'utilisation des différents outils matériels qui sont les capteurs telosb, ainsi que les outils logiciels bien spécifiques aux réseaux de capteurs sans fil, tels que NesC qui est un langage orienté composant, TinyOS qui est un système d'exploitation dédié aux équipements à ressources limitées et Java pour une meilleure présentation des résultats recueillis par les capteurs.

3.1. Les outils matériels :

Dans cette section, nous présentons la plateforme des capteurs sur laquelle nous avons développé notre application on utilisant :

- **Capteur TelosB**
  - **Microcontrôleur Msp430** fabriqué par Texas Instrument, cadencé à 8MHz. Il dispose de 10 KOctets de RAM et de 48 KOctets de mémoire flash. Ce microcontrôleur est optimisé pour répondre aux contraintes d'économie

d'énergie des réseaux de capteurs, d'où son cadencement faible, et de ce fait, il a une capacité de calcul assez limitée [20].

- **Port USB** permettant de programmer (flasher) les capteurs, ou de faire remonter des informations et de les récupérer ensuite à l'aide d'un terminal à partir d'un ordinateur [20].
- **LED** outil visuel indispensable de vérification du fonctionnement du code intégré. Chaque capteur dispos de trois LED (rouge, vert et bleu), elles permettent de suivre le changement d'état de fonctionnement du capteur (réception, envoi, etc.) avec une programmation adéquate [20].

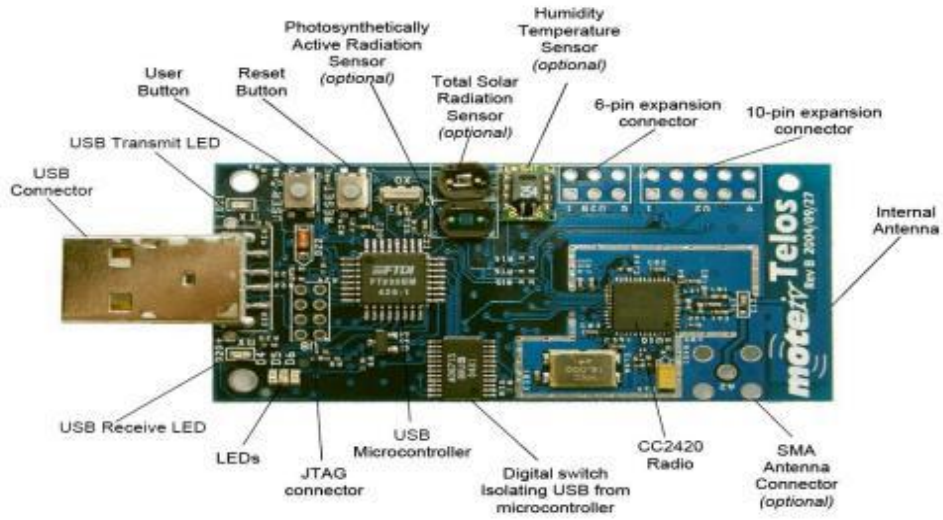


Figure 3. 2 : Capteur TelosB [20]

Ces capteurs de type TelosB ont été élaboré et publié à la communauté scientifique par l'université Berkeley. Cette plate-forme offre une faible consommation d'énergie, elle est compatible avec la distribution open-source de TinyOs. Alimenté par deux piles AA (1.5 V). Un PC portable de type Acer, La figure 3.3 présente la plateforme matérielle utilisée.

Figure 3. 3 : Plateforme matérielle

Le schéma d'architecture est un ensemble des senders qui doivent être capté l'information concerné (la position) et doivent être reliés à un RecBlind qui via un lien radio. Le RecBlind qui jouer le rôle d'un noeud Blind qui ne connait pas sa position et doit être calculer d'après la réception de la position (x,y) des ancres (capteur référencé) selon l'algorithme Centroid . Ce noeud est relié directement à un centre de contrôle via un port série comme montré la figure 3.4.

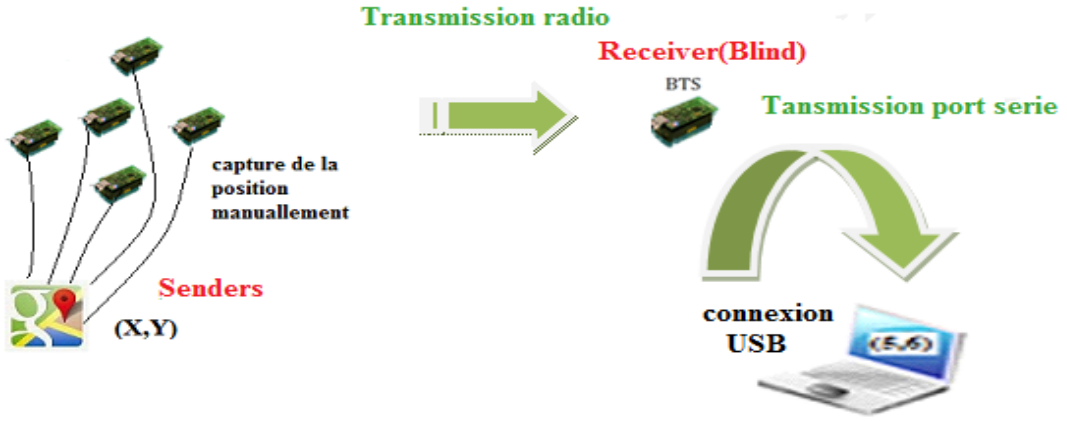


Figure 3. 4 : Schéma de l'architecture

3.2. Les outils logiciels

Dans cette partie, nous présentons les outils logiciels nécessaires pour le développement de notre application. Pour réaliser cette application nous avons utilisé Ubuntu14.04, langage NesC (TinyOS2.1.2), et langage java (NetBeans IDE 6.9.1).

- **Le système d'exploitation TinyOS2.1.2**

TinyOS est un système d'exploitation en open source installé sous linux Ubuntu14.04 développé et suivi par l'université de Berkeley. Ce système d'exploitation a été conçu pour les réseaux de capteurs sans fil car un capteur n'a pas assez de mémoire pour supporter un système d'exploitation comme Linux ou Windows qui prennent beaucoup de place. Il respecte une architecture basée sur les associations de composants, réduisant ainsi la taille du code nécessaire à sa mise en place, respectant ainsi la contrainte de mémoire des capteurs.

TinyOS est basé sur un fonctionnement événementiel et la gestion de taches et d'événements. Les événements sont prioritaires sur les taches et seul un événement peut arrêter une tache. Il y a donc deux niveaux de priorités : basse pour les taches et haute pour les événements.

L'outil de développement de TinyOS est compatible avec Linux et Windows, sa programmation est simple et portable sur de nombreuses plateformes (Mica, Telos...),

cependant il n'est pas possible de faire des allocations dynamiques, des pointeurs sur fonctions et la mémoire n'est pas protégée ce qui implique la possibilité de crash et de corruption de la mémoire.

- **Langage de programmation NesC**

Le langage NesC un est langage de programmation dérivé du langage C utilise une architecture basée sur des composants, celle-ci vise à réduire la taille mémoire du système et les applications développées. Une application est considérée comme un ensemble de composants ayant un but précis. Chaque composant correspond à un élément matériel (LEDs, timer, ADC, etc) et peut être réutilisé dans différentes applications.

Un composant est constitué de trois éléments essentiels :

Les modules, contenant le code d'un composant.

Les configurations qui permettent de faire le lien entre différents modules.

Les interfaces décrivent les commandes et événements proposés par les composants. Il est possible de les redéfinir.

L'extension d'un fichier **NesC** est "**.nc ".**

- **IDE NetBeans (Integrated Development Environment)**

L'environnement de développement, nous avons utilisé NetBeans qui est un IDE (Integrated Development Environment) modulable et open-source, et qui offre de nombreuses fonctionnalités:

- Il contient un éditeur avec coloration syntaxique suivant le langage choisi.
- Il permet la navigation et la gestion des différents projets et de leurs classes.
- Il contient un debugger et un compilateur.
- Il renomme automatiquement les fonctions/classes/variables dans l'ensemble du code.
- Il permet la complétion lors de la saisie du code.
- Il est compatible avec Windows, Unix et Mac.

Etant simple d'utilisation, très complète et gratuite, le choix de ce logiciel s'est donc fait naturellement.

- **Langage java**

JAVA a été développé par Sun au début des années 90 dans une filiation avec le langage C et surtout le langage objet C++ mais dans une optique de plus grande portabilité d'une machine à une autre et d'une plus grande fiabilité. Les programmes JAVA sont compilés en "bytecode", un langage intermédiaire indépendant de la plateforme. Dans notre implémentation, nous avons choisi JAVA car il permet la création des interfaces graphiques grâce à sa librairie Swing et aussi parce que c'est le seul langage dont le code exécutable est portable.

L'implémentation de notre l'application nous avons adopté deux langages de programmations NesC et Java. Le premier est un **langage orienté composant** pour programmer les dispositifs à ressources limitées tels que les capteurs,
et le deuxième est un **langage orienté objet** utilisé pour interpréter et bien exploiter ce qui se passe au centre de contrôle et pour communiquer les données envoyées par les capteurs. D'autre part, pour réaliser l'interface graphique notre choix s'est porté sur JAVA car il permet la création des interfaces graphiques grâce à sa librairie Swing et aussi parce que c'est le seul langage dont le code exécutable est portable.

4. Mise en place de la plateforme

La mise en place de la plateforme nécessite deux étapes: l'installation logicielle et l'installation matérielle.

4.1. Installation logicielle

Cette étape nous a permis de se familiariser avec les capteurs telosb et le langage NesC ainsi que le système d'exploitation TinyOS. L'installation du système d'exploitation TinyOS 2.x sous Ubuntu a été la phase la plus délicate. En effet, nous avons commencé par installer TinyOS 2.1.2, NesC et le microcontrôleur MSP430 pour les capteurs de type TelosB. La procédure d'installation de TinyOS se déroule en plusieurs étapes et elle est décrite dans l'annexe. Puis, nous avons installé NetBeans 6.9.1 au niveau du poste de contrôle pour communiquer avec les capteurs et faire visualiser les valeurs des grandeurs remontées à la station de base.

4.2. Installation matérielle

Lorsque l'installation logicielle terminée, consiste a la réalisation de la plateforme implémentant notre algorithme (voir la figure 3.5), Nous utilisons pour cela six capteurs TelosB, Un capteur relié à l'ordinateur via un port USB qui jouer le rôle d'un station de base, cinq capteurs TelosB. Chacun des TelosB communique avec la station de base via une liaison sans fil et la station de base communique avec l'ordinateur via le port USB. Chacun représentant un rôle bien spécifique comme décrit ci-dessus :

Figure 3. 5: Plateforme d'exécution du l'algorithme

A. Les cinq capteurs sont flashés avec le programme «SenderS ». La Figure 3.6 illustre la documentation de ce programme.

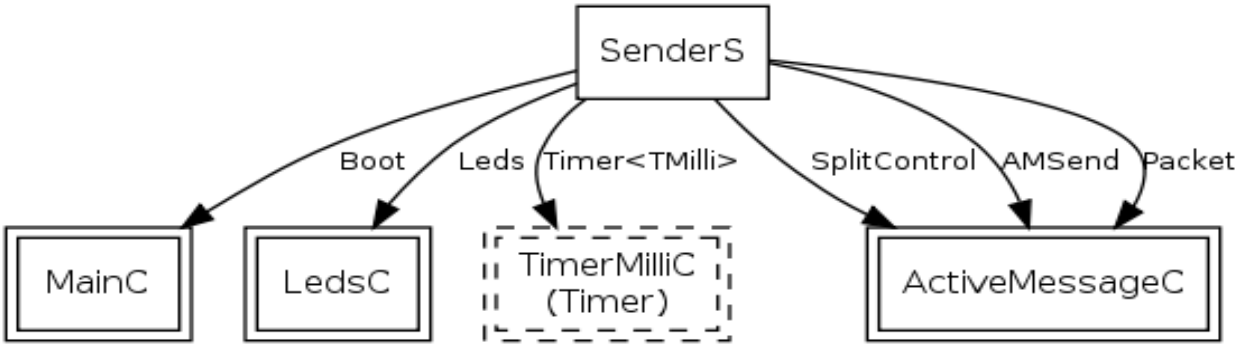


Figure 3. 6 : Représentation graphique du programme « SenderS »

Ce programme utilise les interfaces:

- Boot : permet d'initialiser tous les composants au démarrage, elle est fournie par la configuration MainC qui est le cœur de l'application

- Leds : utilisée pour la manipulation des leds, fournie par LedsC

- Timer<TMilli> : c'est une interface de synchronisation qui permet de gérer le timer d'émission, de round et d'allumage des leds

- SplitControl : utilisée pour l'émission radio fournie par la configuration

- AMSend : pour l'envoi du paquet

- Packet : pour accéder aux données du message

B. Un capteur station de base est flashé avec le programme « RecBlind » qui joue le rôle de Blind qui ne connais pas sa position,qui affiche sur le PC par serial. La Figure 3.7 illustre la documentation de ce programme.

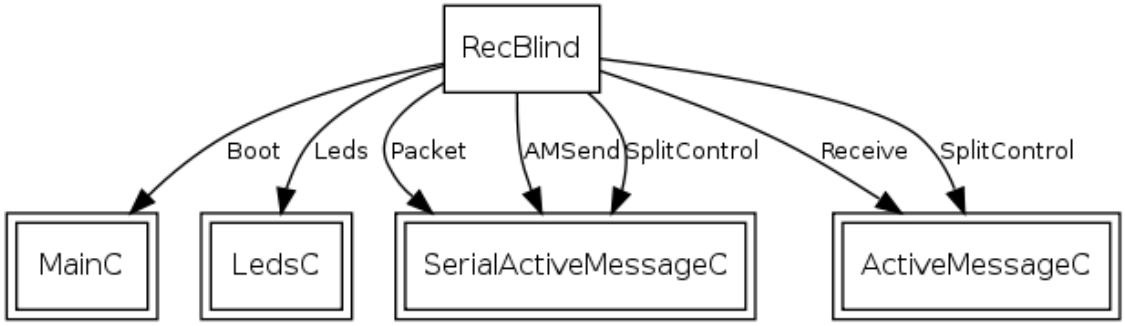


Figure 3. 7 : Représentation graphique du programme « RecBlind »

-Receive : pour la réception des messages envoyer par le programme SenderS

-AMSend : pour envoyer le moyenne calculer vers station de base

Et aussi des composants (MainC, LedC, SerialActiveMessageC, ActivemessageC) parmi ces composants deux composant essentielles :

- ActivemessageC : permet l'accès à la liaison sans fil et l'encapsulation des messages qui pourront être ensuite envoyés via la liaison sans fil.
-SerialActiveMessageC: permet l'accès à la liaison filaire et l'encapsulation des messages qui pourront être ensuite envoyés via la liaison série.

5. Les éléments de l'application

Dans cette section, nous détaillons les principales fonctionnalités offertes par l'application que nous avons développée. Notre application est composée de deux parties: une partie embarquée sur des noeuds capteurs (SenderS) du réseau, une deuxième partie embarquée sur le noeud Blind (RecBlind), et aussi reliée par un PC.

5.1. Partie SenderS

C'est une partie de l'application qui est embarquée sur des noeuds capteurs et qui contient les fonctionnalités suivant :
- Flashé la position qui est donnée manuellement à chaque capteur et envoie au noeud RecBlind.
-L'envoi de cette lecture et son identifiant à RecBlind via un lien radio.
Le code de cette partie est divisé en trois fichiers distincts: le premier c'est la configuration qui contient la définition des composants qui seront utilisés par l'application déployée sur le capteur nommé **SenderAppC.nc** alors que le second

fichier le module **SenderS.nc** et le dernier fichier d'en-tête que nous avons nommé **Message.h** qui contient la structure du message à envoyé.

5.2. Partie RecBlind

C'est une deuxième partie de l'application le RecBlind joue le rôle d'un nœud « blind » qui ne connait pas sa position, ce capteur permet la collection des informations captées depuis les Senders.

- Il reçoit des paquets via la liaison radio les positions des Senders, selon l'algorithme Centroid qui défini précédemment (c'est-à-dire il fait la moyenne des valeurs qu'il a reçue) pour avoir sa position.
- Dans cette partie de l'application embarquée sur un noeud capteur permettant la collection des informations envoyées par les SenderS et ensuite les envoyer via la liaison série. Et aussi permettant de récolter les messages envoyés par les noeuds capteurs de type RecBlind du réseau et les acheminer vers le PC à travers la liaison série. Comme la partie noeud capteur SenderS, la partie noeud capteur RecBlind est constituée aussi de trois fichiers distincts: configuration **RecBlindAppC.nc**, et le module **RecBlind.nc** et fichier d'entête **Message.h.** Dans cet élément exécute une application développée en Java au niveau du centre de contrôle. Nous avons un programme Java qui reçoit et affiche les paquets reçus, mais pour que l'application puisse interpréter les informations reçues, il faut qu'il sache la structure du contenu dans le fichier "**Message.h**".

TinyOS fournit l'outil MIG (Message Interface Generator), qui est capable d'analyser un **"fichier.h"** et de convertir toutes les structures en classes Java disposant des méthodes nécessaires qui nous permettent d'accéder aux valeurs des attributs. Nous faisons appel à cet outil depuis le fichier Makefile, dans notre application coté RecBlind**.** Son appel prend la syntaxe suivante:

```
mig java –target=telosb $ (CFLAGS) –java-classname=LocalisationMsg
opt/tinyos-2.1.2/apps/PFE/RecBlind/Message.h Localisation–Msg-o $@
```

Maintenant pour afficher les valeurs reçues, nous lançons le programme Java en exécutant la commande suivante en mode console:

```
Java MsgReader –comm serial@/dev/ttyUSB0: telosb
```

6. Exemples d'exécution

Dans ce qui suit, nous effectuons une étude de réalisation pour démontrer l'aspect pratique et l'efficacité de l'algorithme de localisation Cendroid. L'algorithme est réalisé sur une plateforme TinyOS qui rajoute l'efficacité et l'analyse des différentes applications des WSN. Dans le cadre de notre projet nous présentons les résultats obtenus en illustrant d'une part la position des ancres et la position estimée ou calculée au Blind et d'autre part l'évaluation des performances.

6.1. Première section : illustration de la localisation par Centroid

Au début, Nous allons utiliser 4 scénarios de notre application qui permet d'estimer la position de nœud ordinaire qui ne connaît pas sa position. Dans notre étude nous avons un réseau de capteur statique constitué des capteurs qui sont déployés dans une zone de 100 mètres sur 100 mètres (distribués d'une manière uniforme) dans un champ d'intérêt. Dans cette section l'application consiste à afficher la position de chaque ancre reçue qui connait sa position absolue par une configuration manuelle à partir d'un programme SenderS et la position estimée du nœud ordinaire (Blind) avec l'ajout des ancres successivement.

La démonstration est présentée dans les scénarios ci-dessous :

6.1.1. Scénario 1 : La position estimée avec deux ancres

Après l'exécution notre application consiste à afficher la position de chaque ancre reçu. Alors nous avons obtenu l'affichage de la position de l'ancre id=1 et la position de deuxième ancre id=2. Nous affectons aux ancres les coordonnées suivantes : $C_1(5,5)$; $C_2(30,5)$ qui sont illustrés dans la figure par un cercle **jaune** et un cercle **rouge** pour illustrer « la position réelle » cette position l'utilisateur devra choisir ou positionner le Blind. Nous affectons les cordonnées(17,10) pour calculer l'erreur de localisation entre « La Position Réelle » et «La Position Estimée » nous assistons à l'affichage d'une position résultante avec un cercle en pointillé de la localisation par Centroid a partir des deux ancres. La Position Estimée = ((5+30)/2, (5+5)/2) qui donne (17.5, 5) et une autre

valeur l'erreur de localisation =5.02 qu'il faut la détaillée dans la section suivante. comme le montre la figure 3.8.

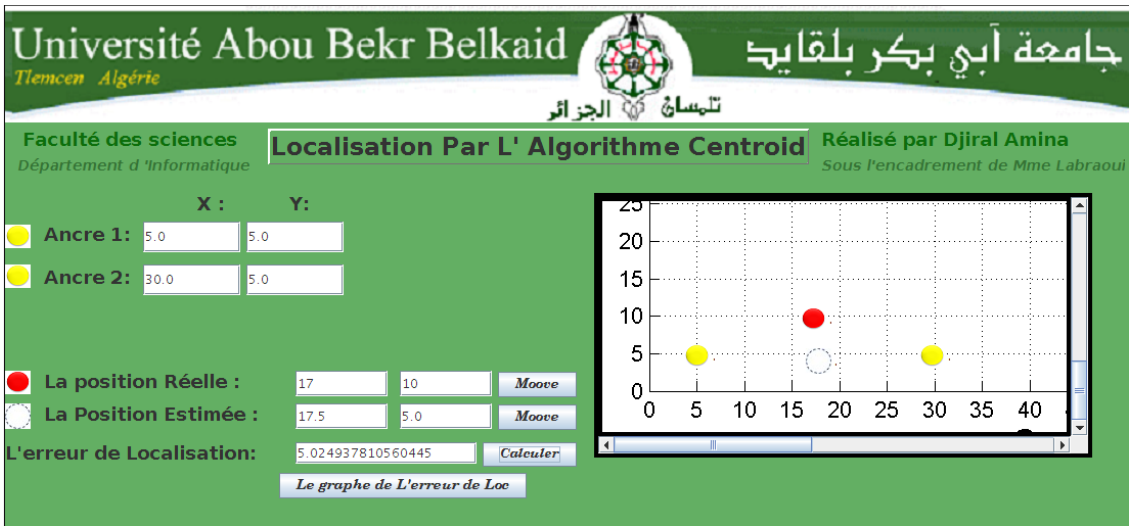


Figure 3. 8: La position estimée des deux ancres

6.1.2. Scénario 2 : La position estimée avec trois ancres

Dans la figure 3.9 un troisième ancre est devenu actif son id =4. Ce dernier a une position(30,15), par la suite nous assistons à une nouvelle valeur résultante de la localisation Centroid avec les trois ancres.

Position Estimée = ((5+30+30) /3, (5+5+15)/3) qui donne (21.66, 8.33).L'erreur de localisation =4.95 comme présentée la figure 3.9.

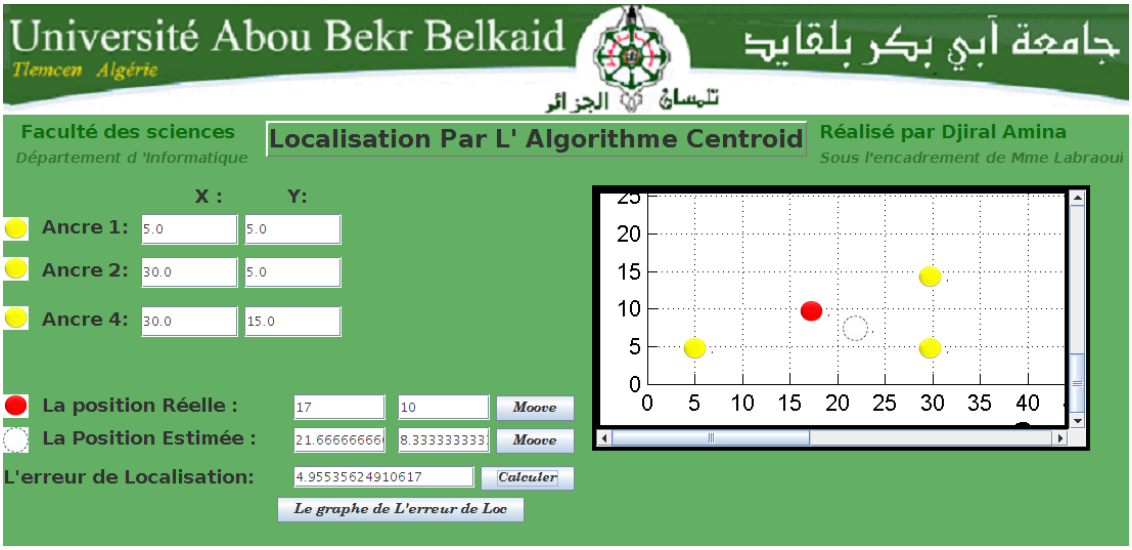


Figure 3. 9 : La position estimée des trois ancres

6.1.3. Scénario 3 : la position estimée avec quatre ancres

Dans cette figure un quatrième ancre est devenu actif son id =3. Ce dernier a une position(5,15), par la suite nous assistons à une nouvelle valeur estimée avec les quatre ancres.

Position Estimée = ((5+30+30+5) /4, (5+5+15+15)/4) qui donne (17.5, 10).L'erreur de localisation =0.5 comme présentée la figure 3.10.

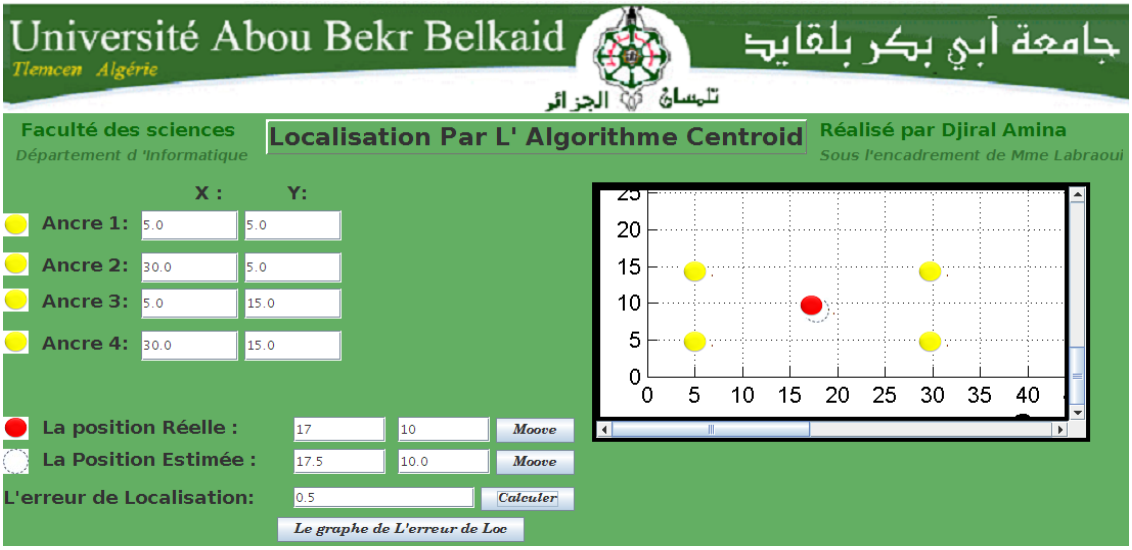


Figure 3. 10: La position estimée des quatre ancres

6.1.4. Scénario 5 : La position estimée avec cinq ancres

Dans la figure 3.11 un cinquième ancre avec un id=5 avec une position (15,10).Par la suite nous assistons à une nouvelle valeur résultante estimée des cinq valeurs de position captées par les ancres.

La Position Estimée = ((5+30+30+5+15) /5, (5+5+15+15+10)/3) qui donne (17,10) L'erreur de localisation =0.0.

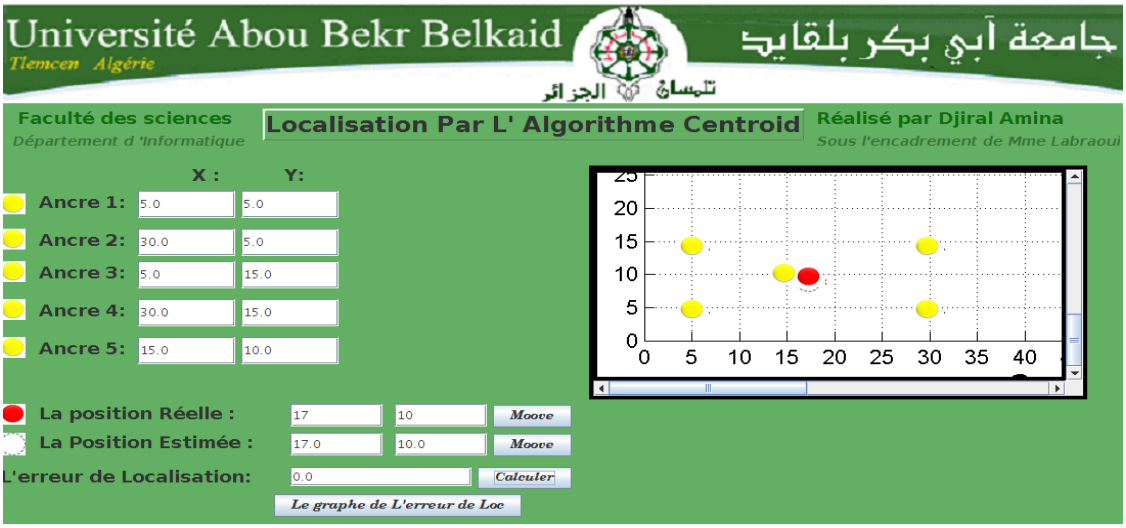


Figure 3. 11 : La position estimée des cinq ancres

D'après le résultat obtenu de l'erreur de localisation nous remarquons clairement que l'erreur de localisation a une relation avec les nombres des ancres, a chaque fois les nombres des capteurs augmentent l'erreur de localisation diminue et la position estimée va être plus précise comme monter la figure 3.12.

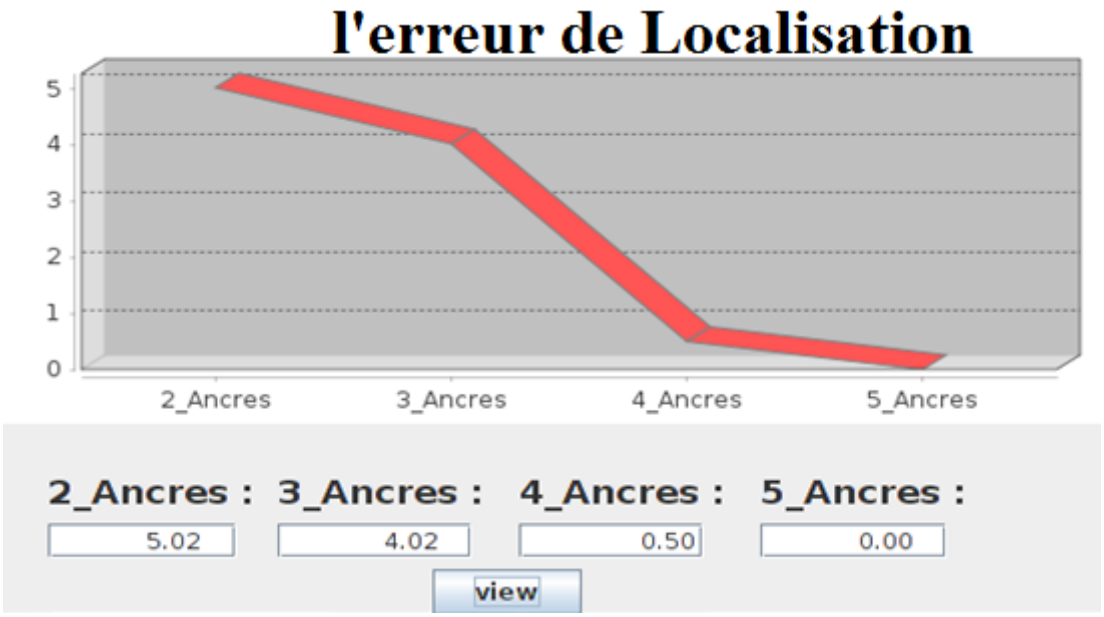


Figure 3. 12 : l'erreur de localisation

6.2. Deuxième section : évaluation des performances

Dans cette sous-section, nous étudions les performances de l'algorithme de localisation Centroid en termes d'erreur de localisation. Afin de faciliter la compréhension des résultats de nos études.

- **Erreur de localisation**

Un des facteurs d'évaluation le plus important dans la technologie de localisation est l'exactitude de localisation, qui se réfère au degré de précision de l'information

Calculée par un nœud capteur ordinaire obtenue par l'algorithme de localisation ou le système. Dans les WSN. L'erreur de localisation est généralement utilisée comme une description quantitative de l'exactitude de la localisation.

Cette erreur de localisation est calculée comme la moyenne de la différence absolue entre la position réelle et la position estimée par chaque nœud du réseau. Supposons que le réseau est constitué de N nœuds, que la position réelle de chaque nœud est$\{(\mathrm{x},\mathrm{y})_a | a \in N\}$, et que la position estimée par chaque nœud est [26] :

$$\boldsymbol{LocError = \frac{1}{N}\sum_{\partial \epsilon \mathbf{N}} \sqrt{(\mathbf{x},\hat{\mathbf{x}})^2 + (\mathbf{y},\hat{\mathbf{y}})^2}} \quad \textbf{(2)}$$

Dans notre expérience, nous étudions l'impact de nombre et de dépoilement des ancres sur les erreurs de localisation. Par conséquent, nous utilisons cinq ancres, un nœud ordinaire. Les nœuds sont déployés dans une zone de 100 mètres sur 100 mètres. Et nous nous basons sur une technique de pavage en grille, où les ancres vont être déployées selon une topologie virtuelle pour propager une partie de la zone initiale sous forme d'une grille. Cette grille virtuelle est formée par conséquent d'un nombre de lignes i et d'un nombre de colonnes j et le nombre des cellules carrés qui compose la grille est calculé comme suit : k=i×j. La hauteur et la largeur de chaque cellule de grille sont égales à 15 m

Nous décrivons ensuite deux stratégie de déploiement : un déploiement des nœuds avec distribution uniforme et un autre déploiement avec distribution non uniforme.

- **Stratégie 1 : déploiement des nœuds avec distribution uniforme :**

  La première stratégie consiste à placer les nœuds ancres au centre de chaque cellule carrée. Dans notre cas nous avons un seul nœud ordinaire et cinq ancres .On affecte aux capteurs ancres les coordonnées suivantes : $C_1(7,7)$; $C_2(22,7)$, $C_3(7,22)$, $C_4(22,22)$, $C_5(15,15)$. Ces derniers qui sont déployés suivant une distribution uniforme selon une configuration

manuelle dans une zone 100 mètres sur 100 mètres comme illustre la figure 3.13.

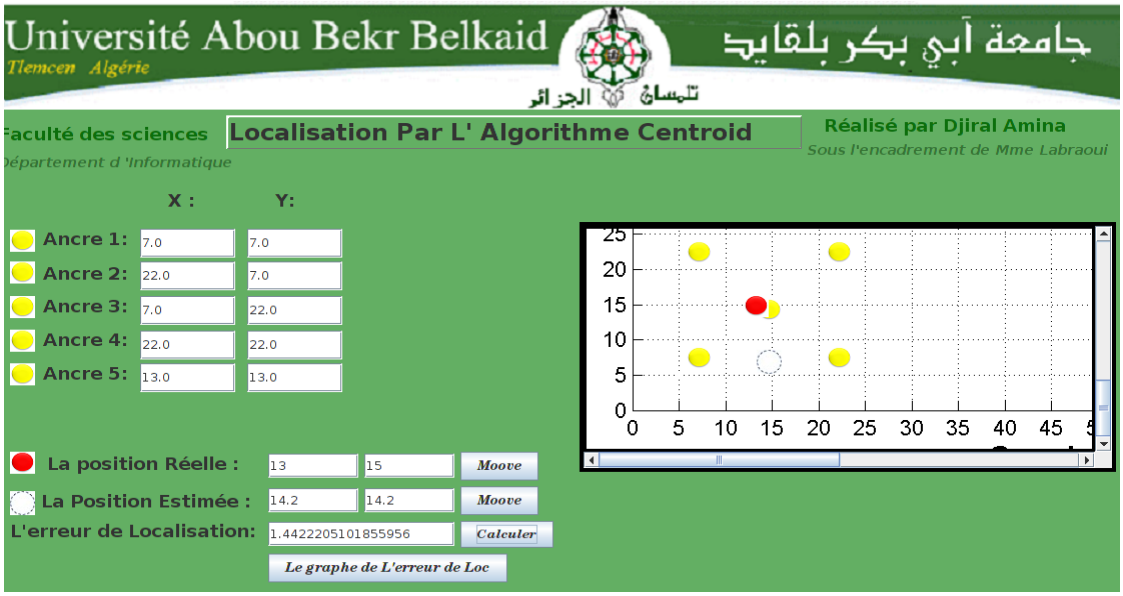


Figure 3. 13 : Déploiement des nœuds avec distribution uniforme

- **Stratégie 2 : déploiement avec distribution non uniforme**

La deuxième stratégie consiste à placer les ancres dans les cellules carrées avec densité différentes. Dans cette expérience nous utilisons aussi la même topologie un nœud ordinaire et cinq ancres .On affecte aux capteurs ancres les coordonnées suivantes : $C_1(7,7)$; $C_2(1,7)$, $C_3(15,2)$, $C_4(22,7)$, $C_5(1,1)$. Ces derniers qui sont déployées suivant une distribution aléatoire comme illustrée la figure 3. 14.

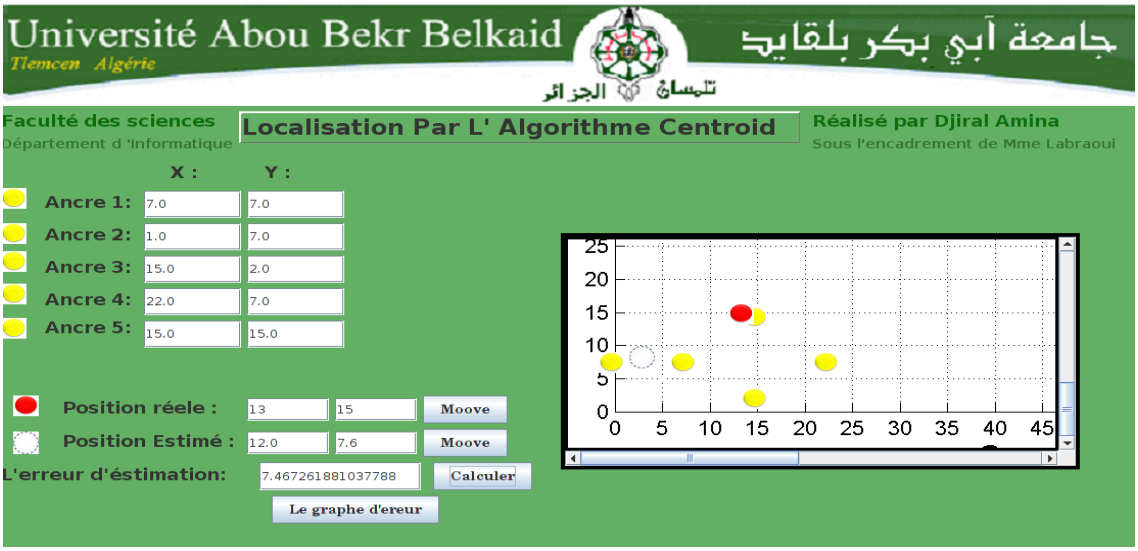
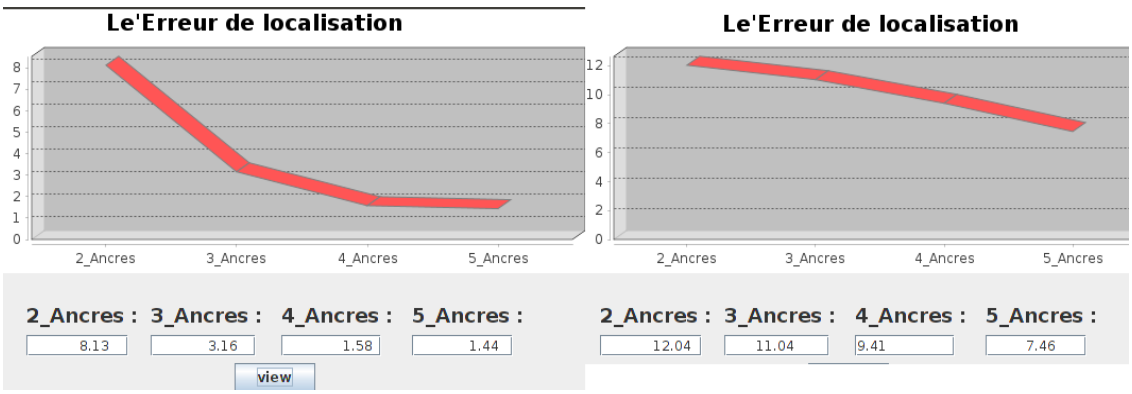


Figure 3. 14 : Déploiement des nœuds avec distribution non uniforme

❖ **Comparaison entre les deux stratégies de déploiement des ancres :**

Afin d'étudier l'impact du déploiement des ancres sur la précision de localisation. La figure 3.15 illustre La relation de LocError et le nombre des nœuds ancres pour deux stratégies de déploiement (déploiement avec distribution uniforme et non uniformes) comme présentée la figure ci-dessus :



(a) **(b)**

Figure 3. 15 : L'erreur de localisation : (a) déploiement avec distribution uniforme, (b) avec distribution non uniforme.

Interprétation des résultats: nous remarquons clairement que l'erreur de localisation dans la figure 3.15(a) est très inférieur par rapport au l'erreur de localisation dans la figure 3.15 (b).et Ceci s'explique par le fait que le déploiement avec distribution uniforme est beaucoup mieux que le déploiement avec distribution non uniforme.

7. Conclusion

Dans ce chapitre, nous avons présenté l'environnement de TinyOS, langage NesC avec lequel nous avons travaillé aussi IDE NetBeans et langage java. Ensuite, nous avons décrit en détail le protocole de localisation Centroid dans le but d'étudier et d'évaluer leurs performances sous deux distributions (uniforme et non uniforme) et ainsi avoir L'erreur de localisation.

En fin nous clôturons notre manuscrit par la conclusion suivante la distribution uniforme donne des meilleures localisations et une meilleure précision des positions par rapport à la distribution non uniforme.

Conclusion générale

Conclusion générale

Ce projet nous a permis de découvrir un nouveau monde les réseaux de capteurs sans fil. Ces derniers constituent un axe de recherche très vaste et peuvent être appliqués dans plusieurs domaines : militaire, médical, environnemental, surveillance, etc… . Cependant, il reste encore de nombreux problèmes à résoudre dans ce domaine afin de pouvoir les utiliser dans les conditions réelles. L'un des problèmes qu'on peut rencontré dans ce genre de réseaux nous citons la problématique de la localisation.

La localisation des capteurs est un problème crucial dans les réseaux de capteurs. Pour le résoudre, on utilise la meilleure méthode qui nous permet d'estimer les positions inconnues des capteurs, à partir des mesures de portée inter-capteurs et de la position connue d'une fraction de capteurs.

L'étude s'est principalement focalisée sur la précision des positions attribuées aux capteurs. Plusieurs recherches ont été faites pour la conception de l'algorithme qui tiennent compte de cette contrainte et en plus qui minimisent la consommation d'énergie. En effet, c'est dans le cadre de ce thème que s'oriente l'objectif de notre projet de fin d'études.

Ce projet nous a permis d'acquérir des connaissances en programmation évènementielle. Il nous a aussi fait découvrir un nouveau langage de programmation, le NesC ainsi que la plateforme de programmation adéquate qui est TinyOs.

Dans notre projet, nous nous sommes intéressées à mettre en place une application de l'algorithme de localisation dans lesRCSFs (*Centroid*) dans le but d'estimer la position d'un nœud « blind » qui ne connait pas sa position exacte et aussi de faire une comparaison d'erreur d'estimation en fonction du nombre des nœuds ancres. Centroid est un algorithme range-free, qui se base uniquement sur la connectivité des nœuds capteurs qui est adaptée dans certaine application comme les feux de forêt. Cet algorithme est caractérisé par sa simplicité par rapport à d'autres algorithmes existants demandant beaucoup de calcul et une grande consommation d'énergie.

Comme perspective il serait intéressant de comparer l'algorithme « Centroid » à un autre algorithme de localisation de type range-free, voir même d'apporter une amélioration dans les calculs afin de minimiser encore plus l'erreur d'estimation.

Bibliographie & Web graphie

Bibliographie :

[1] :David, M. :Sécurité dans les réseaux de capteurs sans fil Stéganographie et réseaux de confiance, L'U.F.R. des Sciences et Techniques de l'université de Franche-Comté, 2010.

[2] : Bouabdellah K., « Problématique de la consommation de l'énergie dans les réseaux de capteurs sans fil », Séminaire LIUPPA, Université de Pau et des Pays de l'Adour, 14 Octobre 2007.

[3] :Yacine, Y. : Minimisation d'énergie dans un réseau de capteurs, Mémoire de Master, Département d'Informatique, Université Mouloud Mammeri de Tizi-Ouzou, Septembre 2012.

[4] :Dunkels, A., Grönvall, B. et Voigt. T. Contiki: a Lightweight and Flexible Operating System for Tiny Networked Sensors. In Proceedings of the First IEEE Workshop on Embedded NetworkedSensors, pages 455-462, Tampa, Florida, USA, 2004.

[5] :Nadjib, B. « les réseaux de capteurs », rapport de recherche dans les systèmes informatiques, février 2004.

[6] :Xue, Y., Gonzalez A., Aguilar A., Barroux M. :Agrégation de données dans les réseaux de capteurs, Rapport de Projet SR04, Université de Technologie Compiègne, Automne 2010.

[7] :Clément S., Quelques contributions dans les réseaux de capteurs sans fil : Localisation et routage, Ecole doctoral 166 I2S, Mathématiques et informatique, Université d'Avignon et pays de Vaucluse, doctorat 2008.

[8]: Arondel O., Ponpardin T. « Systeme de positionnement Galileo/Glonass ». Rapport, Ecole supérieur d'ingénieurs 01/09.

[9]:Parkinson, B. et al: *Global positionning system: Theory and application*. Progrress in Astronomics and Aeronotics, volume1, 1996.

[10]:Savvides, A., Han C C. and Strivastava M B.:Dynamic fine-grained localization in ad-hoc networks of sensors, ACM SIGMOBILE, 2001.

[11]:Bahl, P. , Padmanabhan, V N.**,** RADAR :An In-building RFbased User Location and Tracking System, IEEE INFOCOM, 2000.

[12]:Niculescu, D. and Nath, B.: Ad Hoc Positioning System (APS) using AoA, IEEE INFOCOM '03,2003.

[13]:Langendoen, K. and Reijers N.: Distributed localization in wireless sensor networks : a quantitative comparison, Computer Networks, 2003, Vol. 43(4), pp. 499–518.

[14]:Ji X. and Zha H.: Sensor positioning in wireless ad-hoc sensor, IEEE INFOCOM'04, 2004.

[15]: Niculescu D. and Nath B. :Ad Hoc Positioning System (APS), IEEE GLOBECOM, 2001, pp.2926–2931.

[16]:Doherty L., Pister K. and El Ghaoui L.: Convex position estimation in wireless sensor networks,IEEE Conference on Computer (INFOCOM), 2001.

[17]:He T., Huang C. and Blum B M.:Range-free localization schemes for large scale sensor networks,9th annual international conference on Mobile computing and networking, 2003, pp. 81-95.

[18] :Bulusu, N., Heidemann, J., Estrin, D.:GPS-less low-cost outdoor localization for very small devices. IEEE Personal Communications, vol. 7, no. 5, Oct. 2000, pp. 28-34

[19] : Yannick D. S. : Localisation des nœuds dans les Réseaux de capteurs sans fil Master 2 RTM 2007/2008.

[20] : Gérard C., « Les réseaux de capteurs sans fil », Complexe scientifique des Cézeaux , Clermont Université, Vol. 6, pp. 4.

[21] : Mme LABRAOUI N., :La sécurité dans les réseaux sans fil ad hoc a L'université de Tlemcen Faculté des sciences Pour l'obtention du diplôme de DOCTORAT Spécialité : " Informatique" Soutenue en 2012.

Web graphie :

[22]: http://www.tinyos.net/,2010.

[23]: http://www.btnode.ethz.ch/Projects/SensorNetworkMuseum.

[24]:http://gte.univ-littoral.fr/sections/documents-pdagogiques/chapitre-8-mesure/downloadFile/file/Les_capteurs.pdf?nocache=1289041293.82

[25] :http://www.tafats.fr/Techniques/Reseaux_de_capteurs/Reseaux_capteur_text.html

[26]:http://www.bulletins-electroniques.com/actualites/77208.htm

Annexe

INSTALLATION DE TINYOS 2.1.2 SOUS LINUX 14.04

1. ouvrez le terminal et exécutez cette commande :

$ sudogedit /etc/apt/sources.list

2. Ajoutez « **deb http://tinyos.stanford.edu/tinyos/dists/ubuntu natty main** »à la fin du fichier

3. Maintenant installez tinyos-2.1.2 en tapant :

$sudo apt-get update

$ sudo apt-get install tinyos-2.1.2

Il vous faudra environ 5 - 6 minutes, en fonction de votre vitesse d'Internet.

4. Maintenant, vous devez configurer l'environnement pour TinyOS. Ouvrez le fichier :

$ sudogedit ~/.bashrc

Ajouter les lignes indiquées ci-dessous à la fin de ce fichier :

#Sourcing the tinyos environment variable setup script
source /opt/tinyos-2.1.2/tinyos.sh
export CLASSPATH=$CLASSPATH:.

5. Après tapez:

$ sudogedit /opt/tinyos-2.1.2/tinyos.sh

Ajouter les lignes suivantes, sauvegardez et quittez.

#!/usr/bin/envbash
 # Here we setup the environment
 # variables needed by the tinyos
 # make system
echo "Setting up for TinyOS 2.1.2 Repository Version"
export TOSROOT=
export TOSDIR=
export MAKERULES=
TOSROOT="/opt/tinyos-2.1.2"
TOSDIR="$TOSROOT/tos"
CLASSPATH=$CLASSPATH:$TOSROOT/support/sdk/java:.:$TOSROOT/support/sdk/java/tinyos.jar

MAKERULES="$TOSROOT/support/make/Makerules"
export TOSROOT
exportTOSDIR
export CLASSPATH
export MAKERULES

6. à la fin changez la permission en utilisant cette commande :

$ sudochown -R <username> /opt/tinyos-2.1.2/

Maintenant pour tester, vous compilez le programme **Blink** :

$ cd /opt/tinyos-2.1.2

$ make telosb

LISTE DES FIGURES

LISTE DES TABLEAUX

Liste des abréviations

RCSF : Réseaux de Capteurs Sans Fil.

WSN : Wireless Sensor Networks.

GPS : Global Positioning System.

TinyOS : Tiny Operating System.

CPU : Central Processing Unit .

RAM : Random Access Memory.

NESC : National Electrical Safety Code.

SE : Système d'Exploitation.

LED : Light-Emitting Diode.

MIG : Message Interface Generator.