PROYECTO FIN DE GRADO

TÍTULO: Applications for wireless sensor networks: tracking with binary proximity sensors

AUTOR: Andoni Ruiz Fernández

TUTOR (o Director en su caso): Rafael Herradón Díez

DEPARTAMENTO: DIAC

TITULACIÓN: Grado en Ingeniería de Sistemas de Telecomunicación

V°B°

Miembros del Tribunal Calificador:

PRESIDENTE: Jesús Moreno Blázquez

TUTOR: Rafael Herradón Díez

SECRETARIO: Florentino Jiménez Muñoz

Fecha de lectura: **de** **Septiembre** **de** **2012**

Calificación:

El Secretario,

Agradecimientos

En primer lugar quisiera agradecer a Rafael Herradón la ayuda que me ha brindado para realizar este proyecto a pesar del poco tiempo del que dispone.
A mi familia, que en lo bueno y en lo malo siempre están ahí, especialmente cuando más enfermo he estado.
Al grupo #graciasAndoni por aguantar mis ausencias y por tener que aguantarme a partir de ahora.
A mis compañer@s desadaptad@s, a Ana, Belén y Sonia por ese relajante regalo y a mi Johnny por toda la ayuda aportada, sin la cual me habría sido muy difícil acabar este proyecto a tiempo.
Y a Pauli, porque tu apoyo durante todo el curso ha sido muy importante para mí y espero que lo siga siendo el resto de mi vida.
Gracias.

Resumen

El interés cada vez mayor por las redes de sensores inalámbricos pueden ser entendido simplemente pensando en lo que esencialmente son: un gran número de pequeños nodos sensores autoalimentados que recogen información o detectan eventos especiales y se comunican de manera inalámbrica, con el objetivo final de entregar sus datos procesados a una estación base. Los nodos sensores están densamente desplegados dentro del área de interés, se pueden desplegar al azar y tienen capacidad de cooperación. Por lo general, estos dispositivos son pequeños y de bajo costo, de modo que pueden ser producidos y desplegados en gran número aunque sus recursos en términos de energía, memoria, velocidad de cálculo y ancho de banda están enormemente limitados. Detección, tratamiento y comunicación son tres elementos clave cuya combinación en un pequeño dispositivo permite lograr un gran número de aplicaciones. Las redes de sensores proporcionan oportunidades sin fin, pero al mismo tiempo plantean retos formidables, tales como lograr el máximo rendimiento de una energía que es escasa y por lo general un recurso no renovable. Sin embargo, los recientes avances en la integración a gran escala, integrado de hardware de computación, comunicaciones, y en general, la convergencia de la informática y las comunicaciones, están haciendo de esta tecnología emergente una realidad. Del mismo modo, los avances en la nanotecnología están empezando a hacer que todo gire entorno a las redes de pequeños sensores y actuadores distribuidos.

Hay diferentes tipos de sensores tales como sensores de presión, acelerómetros, cámaras, sensores térmicos o un simple micrófono. Supervisan las condiciones presentes en diferentes lugares tales como la temperatura, humedad, el movimiento, la luminosidad, presión, composición del suelo, los niveles de ruido, la presencia o ausencia de ciertos tipos de objetos, los niveles de tensión mecánica sobre objetos adheridos y las características momentáneas tales como la velocidad , la dirección y el tamaño de un objeto, etc. Se comprobará el estado de las Redes Inalámbricas de Sensores y se revisarán los protocolos más famosos. Así mismo, se examinará la identificación por radiofrecuencia (RFID) ya que se está convirtiendo en algo actual y su presencia importante. La RFID tiene un papel crucial que desempeñar en el futuro en el mundo de los negocios y los individuos por igual. El impacto mundial que ha tenido la identificación sin cables está ejerciendo fuertes presiones en la tecnología RFID, los servicios de investigación y desarrollo, desarrollo de normas, el cumplimiento de la seguridad y la privacidad y muchos más. Su potencial económico se ha demostrado en algunos países mientras que otros están simplemente en etapas de planificación o en etapas piloto, pero aún tiene que afianzarse o desarrollarse a través de la modernización de los modelos de negocio y aplicaciones para poder tener un mayor impacto en la sociedad.

Las posibles aplicaciones de redes de sensores son de interés para la mayoría de campos. La monitorización ambiental, la guerra, la educación infantil, la vigilancia, la micro-cirugía y la agricultura son sólo unos pocos ejemplos de los muchísimos campos en los que tienen cabida las redes mencionadas anteriormente. Estados Unidos de América es probablemente el país que más ha investigado en esta área por lo que veremos muchas soluciones propuestas provenientes de ese país. Universidades como Berkeley, UCLA (Universidad de California, Los Ángeles) Harvard y empresas como Intel lideran dichas investigaciones. Pero no sólo EE.UU. usa e investiga las redes de sensores inalámbricos. La Universidad de Southampton, por ejemplo, está desarrollando una tecnología para monitorear el comportamiento de los glaciares mediante redes de sensores que contribuyen a la investigación fundamental en

glaciología y de las redes de sensores inalámbricos. Así mismo, Coalesenses GmbH (Alemania) y Zurich ETH están trabajando en diversas aplicaciones para redes de sensores inalámbricos en numerosas áreas. Una solución española será la elegida para ser examinada más a fondo por ser innovadora, adaptable y polivalente. Este estudio del sensor se ha centrado principalmente en aplicaciones de tráfico, pero no se puede olvidar la lista de más de 50 aplicaciones diferentes que ha sido publicada por la firma creadora de este sensor específico.

En la actualidad hay muchas tecnologías de vigilancia de vehículos, incluidos los sensores de bucle, cámaras de video, sensores de imagen, sensores infrarrojos, radares de microondas, GPS, etc. El rendimiento es aceptable, pero no suficiente, debido a su limitada cobertura y caros costos de implementación y mantenimiento, especialmente este último. Tienen defectos tales como: línea de visión, baja exactitud, dependen mucho del ambiente y del clima, no se puede realizar trabajos de mantenimiento sin interrumpir las mediciones, la noche puede condicionar muchos de ellos, tienen altos costos de instalación y mantenimiento, etc. Por consiguiente, en las aplicaciones reales de circulación, los datos recibidos son insuficientes o malos en términos de tiempo real debido al escaso número de detectores y su costo. Con el aumento de vehículos en las redes viales urbanas las tecnologías de detección de vehículos se enfrentan a nuevas exigencias. Las redes de sensores inalámbricos son actualmente una de las tecnologías más avanzadas y una revolución en la detección de información remota y en las aplicaciones de recogida. Las perspectivas de aplicación en el sistema inteligente de transporte son muy amplias. Con este fin se ha desarrollado un programa de localización de objetivos y recuento utilizando una red de sensores binarios. Esto permite que el sensor necesite mucha menos energía durante la transmisión de información y que los dispositivos sean más independientes con el fin de tener un mejor control de tráfico. La aplicación se centra en la eficacia de la colaboración de los sensores en el seguimiento más que en los protocolos de comunicación utilizados por los nodos sensores.

Las operaciones de salida y retorno en las vacaciones son un buen ejemplo de por qué es necesario llevar la cuenta de los coches en las carreteras. Para ello se ha desarrollado una simulación en Matlab con el objetivo localizar objetivos y contarlos con una red de sensores binarios. Dicho programa se podría implementar en el sensor que Libelium, la empresa creadora del sensor que se examinará concienzudamente, ha desarrollado. Esto permitiría que el aparato necesitase mucha menos energía durante la transmisión de información y los dispositivos sean más independientes. Los prometedores resultados obtenidos indican que los sensores de proximidad binarios pueden formar la base de una arquitectura robusta para la vigilancia de áreas amplias y para el seguimiento de objetivos. Cuando el movimiento de dichos objetivos es suficientemente suave, no tiene cambios bruscos de trayectoria, el algoritmo ClusterTrack proporciona un rendimiento excelente en términos de identificación y seguimiento de trayectorias los objetos designados como blancos. Este algoritmo podría, por supuesto, ser utilizado para numerosas aplicaciones y se podría seguir esta línea de trabajo para futuras investigaciones. No es sorprendente que las redes de sensores de binarios de proximidad hayan atraído mucha atención últimamente ya que, a pesar de la información mínima de un sensor de proximidad binario proporciona, las redes de este tipo pueden realizar un seguimiento de todo tipo de objetivos con la precisión suficiente.

Abstract

The increasing interest in wireless sensor networks can be promptly understood simply by thinking about what they essentially are: a large number of small sensing self-powered nodes which gather information or detect special events and communicate in a wireless fashion, with the end goal of handing their processed data to a base station. The sensor nodes are densely deployed inside the phenomenon, they deploy random and have cooperative capabilities. Usually these devices are small and inexpensive, so that they can be produced and deployed in large numbers, and so their resources in terms of energy, memory, computational speed and bandwidth are severely constrained. Sensing, processing and communication are three key elements whose combination in one tiny device gives rise to a vast number of applications. Sensor networks provide endless opportunities, but at the same time pose formidable challenges, such as the fact that energy is a scarce and usually non-renewable resource. However, recent advances in low power Very Large Scale Integration, embedded computing, communication hardware, and in general, the convergence of computing and communications, are making this emerging technology a reality. Likewise, advances in nanotechnology and Micro Electro-Mechanical Systems are pushing toward networks of tiny distributed sensors and actuators.

There are different sensors such as pressure, accelerometer, camera, thermal, and microphone. They monitor conditions at different locations, such as temperature, humidity, vehicular movement, lightning condition, pressure, soil makeup, noise levels, the presence or absence of certain kinds of objects, mechanical stress levels on attached objects, the current characteristics such as speed, direction and size of an object, etc. The state of Wireless Sensor Networks will be checked and the most famous protocols reviewed. As Radio Frequency Identification (RFID) is becoming extremely present and important nowadays, it will be examined as well. RFID has a crucial role to play in business and for individuals alike going forward. The impact of 'wireless' identification is exerting strong pressures in RFID technology and services research and development, standards development, security compliance and privacy, and many more. The economic value is proven in some countries while others are just on the verge of planning or in pilot stages, but the wider spread of usage has yet to take hold or unfold through the modernisation of business models and applications.

Possible applications of sensor networks are of interest to the most diverse fields. Environmental monitoring, warfare, child education, surveillance, micro-surgery, and agriculture are only a few examples. Some real hardware applications in the United States of America will be checked as it is probably the country that has investigated most in this area. Universities like Berkeley, UCLA (University of California, Los Angeles) Harvard and enterprises such as Intel are leading those investigations. But not just USA has been using and investigating wireless sensor networks. University of Southampton e.g. is to develop technology to monitor glacier behaviour using sensor networks contributing to fundamental research in glaciology and wireless sensor networks. Coalesenses GmbH (Germany) and ETH Zurich are working in applying wireless sensor networks in many different areas too. A Spanish solution will be the one examined more thoroughly for being innovative, adaptable and multipurpose. This study of the sensor has been focused mainly to traffic applications but it cannot be forgotten the more than 50 different application compilation that has been published by this specific sensor's firm.

Currently there are many vehicle surveillance technologies including loop sensors, video cameras, image sensors, infrared sensors, microwave radar, GPS, etc. The performance is acceptable but not sufficient because of their limited coverage and expensive costs of implementation and maintenance, specially the last one. They have defects such as: line-of-sight, low exactness, depending on environment and weather, cannot perform no-stop work whether daytime or night, high costs for installation and maintenance, etc. Consequently, in actual traffic applications the received data is insufficient or bad in terms of real-time owed to detector quantity and cost. With the increase of vehicle in urban road networks, the vehicle detection technologies are confronted with new requirements. Wireless sensor network is the state of the art technology and a revolution in remote information sensing and collection applications. It has broad prospect of application in intelligent transportation system. An application for target tracking and counting using a network of binary sensors has been developed. This would allow the appliance to spend much less energy when transmitting information and to make more independent devices in order to have a better traffic control. The application is focused on the efficacy of collaborative tracking rather than on the communication protocols used by the sensor nodes.

Holiday crowds are a good case in which it is necessary to keep count of the cars on the roads. To this end a Matlab simulation has been produced for target tracking and counting using a network of binary sensors that e.g. could be implemented in Libelium's solution. Libelium is the enterprise that has developed the sensor that will be deeply examined. This would allow the appliance to spend much less energy when transmitting information and to make more independent devices. The promising results obtained indicate that binary proximity sensors can form the basis for a robust architecture for wide area surveillance and tracking. When the target paths are smooth enough ClusterTrack particle filter algorithm gives excellent performance in terms of identifying and tracking different target trajectories. This algorithm could, of course, be used for different applications and that could be done in future researches. It is not surprising that binary proximity sensor networks have attracted a lot of attention lately. Despite the minimal information a binary proximity sensor provides, networks of these sensing modalities can track all kinds of different targets classes accurate enough.

Index

1. Introduction

The increasing interest in wireless sensor networks (WSN) can be promptly understood simply by thinking about what they essentially are: a large number of small sensing self-powered nodes which gather information or detect special events and communicate in a wireless fashion, with the end goal of handing their processed data to a base station. The sensor nodes are densely deployed inside the phenomenon, they deploy random and have cooperative capabilities. Usually these devices are small and inexpensive, so that they can be produced and deployed in large numbers, and so their resources in terms of energy, memory, computational speed and bandwidth are severely constrained. Sensing, processing and communication are three key elements whose combination in one tiny device gives rise to a vast number of applications. Sensor networks provide endless opportunities, but at the same time pose formidable challenges, such as the fact that energy is a scarce and usually non-renewable resource. However, recent advances in low power Very Large Scale Integration (VLSI), embedded computing, communication hardware, and in general, the convergence of computing and communications, are making this emerging technology a reality. Likewise, advances in nanotechnology and Micro Electro-Mechanical Systems (MEMS) are pushing toward networks of tiny distributed sensors and actuators.

There are different sensors such as pressure, accelerometer, camera, thermal, and microphone. They monitor conditions at different locations, such as temperature, humidity, vehicular movement, lightning condition, pressure, soil makeup, noise levels, the presence or absence of certain kinds of objects, mechanical stress levels on attached objects, the current characteristics such as speed, direction and size of an object, etc. The state of Wireless Sensor Networks will be checked and the most famous protocols reviewed. As RFID is becoming extremely present and important nowadays, it will be examined as well.

Possible applications of sensor networks are of interest to the most diverse fields. Environmental monitoring, warfare, child education, surveillance, micro-surgery, and agriculture are only a few examples. Firstly, we are going to check some real hardware applications in the United States of America (USA) as it is probably the country that has investigated most in this area. Universities like Berkeley, UCLA (University of California, Los Angeles) Harvard and enterprises such as Intel are leading those investigations. But not just USA has been using and investigating wireless sensor networks. University of Southampton e.g. is to develop technology to monitor glacier behaviour using sensor networks contributing to fundamental research in glaciology and wireless sensor networks. Coalesenses GmbH (Germany) and ETH Zurich are working in applying WSN in many different areas too. A Spanish solution will be the one examined more thoroughly for being innovative, adaptable and multipurpose.

Currently there are many vehicle surveillance technologies including loop sensors, video cameras, image sensors, infrared sensors, microwave radar, GPS, etc. The performance is acceptable but not sufficient because of their limited coverage and expensive costs of implementation and maintenance, specially the last one. They have defects such as: line-of-sight, low exactness, depending on environment and weather, cannot perform no-stop work whether daytime or night, high costs for installation and maintenance, etc. Consequently, in actual traffic applications the received data is insufficient or bad in terms of real-time owed to detector quantity and cost. With the increase of vehicle in urban road networks, the vehicle

detection technologies are confronted with new requirements. Wireless sensor network is the state of the art technology and a revolution in remote information sensing and collection applications. It has broad prospect of application in intelligent transportation system. An application for target tracking and counting using a network of binary sensors has been developed. This would allow the appliance to spend much less energy when transmitting information and to make more independent devices in order to have a better traffic control. The application is focused on the efficacy of collaborative tracking rather than on the communication protocols used by the sensor nodes.

Summarizing, in section 2 the WSN background issue will be addressed. The most common wireless protocol standards will be reviewed and compared. As RFID (Radio Frequency Identification) is starting to be more and more present in real life, it will be analysed as well. In section 3, some possible WSN applications will be checked, how is this technology being developed all over the world and a specific WSN solution will be examined more thoroughly. In section 4 a study and simulation of and application for target tracking using a binary proximity sensor network will be seen. This could useful and applied in a real WSN solution as the one presented in section 3. Finally section 5 will be for conclusions and future investigation lines that could be followed.

2. Wireless Sensor Network Background

In this section, we are going to check the state of Wireless Sensor Networks and which technology could be more appropriate depending on the required application. As RFID is becoming extremely present and important nowadays, we will take a look on that technology as well. Before starting, a brief note of "The Internet of Things" which will be used during this project and a very important term to know.

'Internet of Things(IOT)' is defined by Commission of Europeans Communities under European Commission as that "in which the Internet does not only link computers and communications terminals, but potentially any of our daily surrounding objects – be clothes, consumer goods, etc.". In detail, the European Commission report says, 'The Internet of Things' means the fusion of the physical and digital worlds: physical entities have digital counterpart; objects become context-aware – they can sense, communicate and interact; immediate responses can be given to physical phenomena; instant information can be collected about physical entities; intelligent real-time decision making becomes possible, thus opening up new opportunities to handle incidents, meet business requirements, create new services based on real-time physical world data, gain insights into complex processes and relationships, address environmental degradation (pollution, disaster, global warming), monitor human activities (work, criminal, health, military), improve infrastructure integrity (civil, energy, water, transport), and so on as can be seen in figure 1.

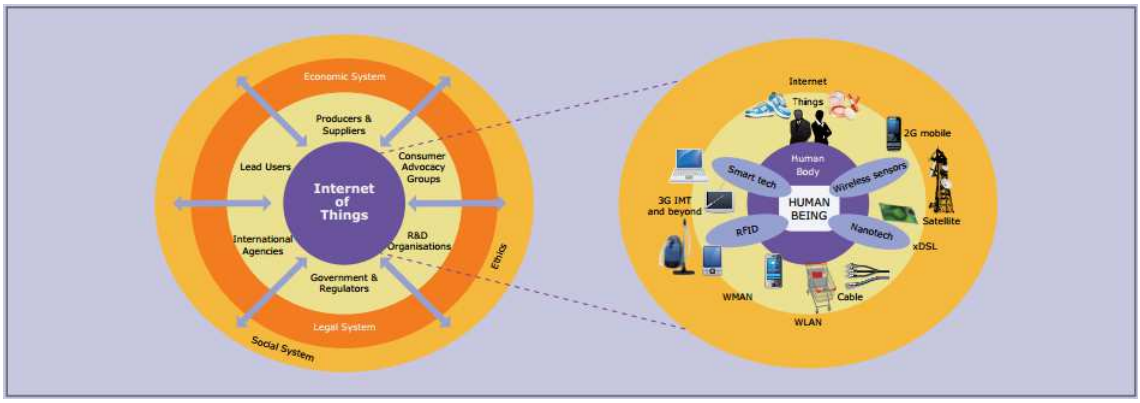

Figure 1.Everything is interconnected

It is about a network of billions or trillions of machines interacting with one another – a major theme in the evolution of information and communications over the next few decades, and in its simplest form it is already here. The idea has grown from advanced concepts from the last twenty years such as ubiquitous communications, pervasive computing, and ambient intelligence.

From the technological perspective, the Internet of Things will enable computing to meld into the fabric of our business, personal and social environments. Here are three simple examples:

• Asset management. Electronic tagging and remote sensing tracks the location of baggage in an airport, or of goods in a factory production process.

• Healthcare. Blood pressure and heart rate sensors relay regular readings from a patient's home to a monitoring centre. A computer detects adverse movements and signals a doctor to check patient.

• Environment monitoring. A network of sensors monitors river heights and rainfall, predicting floods and supporting water management measures for flood relief.

Potential applications of pervasive networking are limitless. Some proposals appear essential, for example people tracking in a disaster such as a tunnel fire. Others may at first seem unrealistic. This then will drive the needed 'generic' technology developments. The Internet of Things receives attention worldwide and from diverse research communities. It would benefit from a strong industrial drive and a clear commitment to application perspective that goes beyond the classical RFID devices. The applications have interconnected 'smart devices', though groups of devices need not be equally smart. For example, a gateway device may filter, aggregate and format data from a group of simple sensors before sending it somewhere else.

The reason for Sensor Networks being radio communicated is because it supports ubiquity and mobility and it supports flexibility not available in wired communications. For example, a device can move into a group and then move out of it again. Focus of the network will shift from human interaction to machine-machine connectivity. When machines communicate directly with each other, people can focus on major issues, not routine. Indeed, new networks and service infrastructures are expected to emerge replacing the current Internet.

2.1 Wireless Protocol Standards

Bluetooth (over IEEE 802.15.1), ultra-wideband (UWB, over IEEE 802.15.3), ZigBee (over IEEE 802.15.4), and Wi-Fi (over IEEE 802.11) are four protocol standards for short-range wireless communications with low power consumption.

From an application point of view, Bluetooth is intended for low bandwidth applications where higher USB bandwidth is not required and cable-free connection desired, UWB is oriented to high-bandwidth multimedia links, ZigBee is designed for reliable wirelessly networked monitoring and control networks, while Wi-Fi is directed at computer-to-computer connections as an extension or substitution of cabled networks.

The aim is to provide a study of these popular wireless communication standards, evaluating their main features and behaviours in terms of various metrics, including the transmission time, data coding efficiency, complexity, and power consumption and to compare them. This comparison is not to draw any conclusion regarding which one is superior since the suitability of network protocols is greatly influenced by practical applications, of which many other factors such as the network reliability, roaming capability, recovery mechanism, chipset price, and installation cost need to be considered in the future. Still, it will be helpful once the application part of the project is reached in order to select a suitable network standard.

2.1.1 Bluetooth

Bluetooth, also known as the IEEE 802.15.1 standard is based on a wireless radio system designed for short-range and cheap devices to replace cables for computer peripherals, such as mice, keyboards, joysticks and printers and other low bandwidth applications cable-free connection is desired. This range of applications is known as wireless personal area network (WPAN).

Two connectivity topologies are defined in Bluetooth: the piconet and scatternet. A piconet is a WPAN formed by a Bluetooth device serving as a master in the piconet and one or more Bluetooth devices serving as slaves. A frequency-hopping channel based on the address of the master defines each piconet. All devices participating in communications in a given piconet are synchronized using the clock of the master. Slaves communicate only with their master in a point-to-point fashion under the control of the master. The master's transmissions may be either point-to-point or point-to-multipoint. Also, besides in an active mode, a slave device can be in the parked or standby modes so as to reduce power consumptions. A scatternet is a collection of operational Bluetooth piconets overlapping in time and space. Two piconets can be connected to form a scatternet. A Bluetooth device may participate in several piconets at the same time, thus allowing for the possibility that information could flow beyond the coverage area of the single piconet. A device in a scatternet could be a slave in several piconets, but master in only one of them.

2.1.2 UWB

UWB has recently attracted much attention as an indoor short-range high-speed wireless communication. One of the most exciting characteristics of UWB is that its bandwidth is over 110 Mbps (up to 480 Mbps) which can satisfy most of the multimedia applications such as audio and video delivery in home networking and it can also act as a wireless cable replacement of high speed serial bus such as USB 2.0 and IEEE 1394. UWB communications transmit in a way which does not interfere with conventional narrowband and carrier wave uses in the same frequency band.

Following the United States and the Federal Communications Commission (FCC) frequency allocation for UWB in February 2002, the Electronic Communications Committee (ECC TG3) is progressing in the elaboration of a regulation for the UWB technology in Europe. From an implementation point of view, several solutions have been developed in order to use the UWB technology in compliance with the FCC's regulatory requirements. Among the existing PHY solutions, in IEEE 802.15 Task Group 3a (TG3a), multiband orthogonal frequency-division multiplexing (MB-OFDM), a carrier-based system dividing UWB bandwidth to sub-bands, and direct-sequence UWB (DS-UWB), an impulse-based system that multiplies an input bit with the spreading code and transmits the data by modulating the element of the symbol with a short pulse have been proposed by the WiMedia Alliance and the UWB Forum, respectively.

The TG3a was established in January 2003 to define an alternative PHY layer of 802.15.3. However, after three years of a jammed process in IEEE 802.15.3a, supporters of both proposals, MB-OFDM and DS-UWB, supported the shutdown of the IEEE 802.15.3a task

group without conclusion in January 2006. On the other hand, IEEE 802.15.3b, the amendment to the 802.15.3 MAC sub-layer was approved and released in March 2006.

2.1.3 Zigbee

ZigBee over IEEE 802.15.4 defines specifications for low rate WPAN (LR-WPAN) for supporting simple devices that consume minimal power and typically operate in the personal operating space (POS) of 10m. ZigBee provides self-organized, multi-hop, and reliable mesh networking with long battery lifetime.

Two different device types can participate in an LR-WPAN network: a full-function device (FFD) and a reduced-function device (RFD). The FFD can operate in three modes serving as a PAN coordinator, a coordinator, or a device. An FFD can talk to RFDs or other FFDs, while an RFD can talk only to an FFD. An RFD is intended for applications that are extremely simple, such as a light switch or a passive infrared sensor. They do not have the need to send large amounts of data and may only associate with a single FFD at a time. Consequently, the RFD can be implemented using minimal resources and memory capacity.

After an FFD is activated for the first time, it may establish its own network and become the PAN coordinator. All the star networks operate independently from all other star networks currently in operation. This is achieved by choosing a PAN identifier, which is not currently used by any other network within the radio sphere of influence. Once the PAN identifier is chosen, the PAN coordinator can allow other devices to join its network. An RFD may connect to a cluster tree network as a leave node at the end of a branch, because it may only associate with one FFD at a time. Any of the FFDs may act as a coordinator and provide synchronization services to other devices or other coordinators. Only one of these coordinators can be the overall PAN coordinator, which may have greater computational resources than any other device in the PAN.

2.1.4 Wi-Fi

Wireless fidelity (Wi-Fi) includes IEEE 802.11a/b/g standards for wireless local area networks (WLAN). It allows users to surf the Internet at broadband speeds when connected to an access point (AP) or in ad hoc mode.

The IEEE 802.11 architecture consists of several components that interact to provide a wireless LAN that supports station mobility transparently to upper layers. The basic cell of an IEEE 802.11 LAN is called a basic service set (BSS), which is a set of mobile or fixed stations. If a station moves out of its BSS, it can no longer directly communicate with other members of the BSS. Based on the BSS, IEEE 802.11 employs the independent basic service set (IBSS) and extended service set (ESS) network configurations. As shown in figure 2, the IBSS operation is possible when IEEE 802.11 stations are able to communicate directly without any AP. Because this type of IEEE 802.11 LAN is often formed without pre-planning, for only as long as the LAN is needed, this type of operation is often referred to as an ad hoc network. Instead of existing independently, a BSS may also form a component of an extended form of network that is built with multiple BSSs. The architectural component used to interconnect BSSs is the distribution system (DS). The DS with APs allow IEEE 802.11 to create an ESS network of arbitrary size and complexity. This type of operation is often referred to as an infrastructure network.

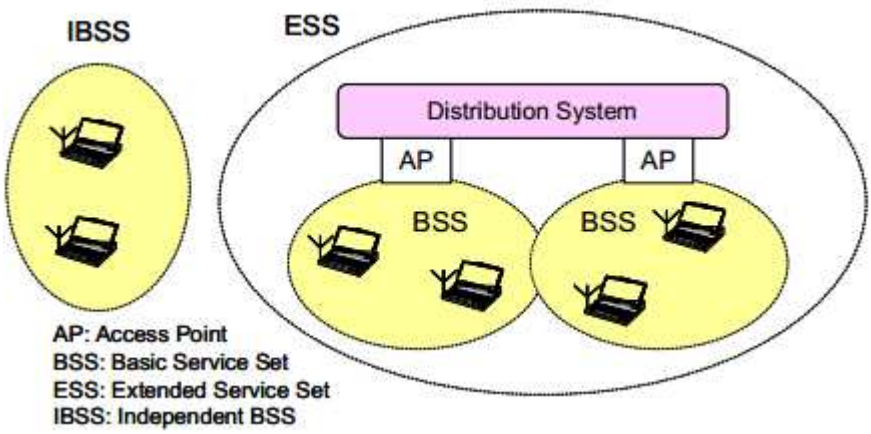

Figure 2.IBSS and ESS configurations of Wi-Fi networks

2.2 Comparative Study

Table 1 summarizes the main differences among the four protocols. Each protocol is based on an IEEE standard. Obviously, UWB and Wi-Fi provide a higher data rate, while Bluetooth and ZigBee give a lower one. In general, the Bluetooth, UWB, and ZigBee are intended for WPAN communication (about 10m), while Wi-Fi is oriented to WLAN (about 100m). However, ZigBee can also reach 100m in some applications.

Table 1.Comparison of Bluetooth, UWB, Zigbee and Wi-Fi protocols

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
IEEE spec.	802.15.1	802.15.3a *	802.15.4	802.11a/b/g
Frequency band	2.4 GHz	3.1-10.6 GHz	868/915 MHz; 2.4 GHz	2.4 GHz; 5 GHz
Max signal rate	1 Mb/s	110 Mb/s	250 Kb/s	54 Mb/s
Nominal range	10 m	10 m	10 - 100 m	100 m
Nominal TX power	0 - 10 dBm	-41.3 dBm/MHz	(-25) - 0 dBm	15 - 20 dBm
Number of RF channels	79	(1-15)	1/10; 16	14 (2.4 GHz)
Channel bandwidth	1 MHz	500 MHz - 7.5 GHz	0.3/0.6 MHz; 2 MHz	22 MHz
Modulation type	GFSK	BPSK, QPSK	BPSK (+ ASK), O-QPSK	BPSK, QPSK COFDM, CCK, M-QAM
Spreading	FHSS	DS-UWB, MB-OFDM	DSSS	DSSS, CCK, OFDM
Coexistence mechanism	Adaptive freq. hopping	Adaptive freq. hopping	Dynamic freq. selection	Dynamic freq. selection, transmit power control (802.11h)
Basic cell	Piconet	Piconet	Star	BSS
Extension of the basic cell	Scatternet	Peer-to-peer	Cluster tree, Mesh	ESS
Max number of cell nodes	8	8	> 65000	2007
Encryption	E0 stream cipher	AES block cipher (CTR, counter mode)	AES block cipher (CTR, counter mode)	RC4 stream cipher (WEP), AES block cipher
Authentication	Shared secret	CBC-MAC (CCM)	CBC-MAC (ext. of CCM)	WPA2 (802.11i)
Data protection	16-bit CRC	32-bit CRC	16-bit CRC	32-bit CRC

* Unapproved draft.
• Acronyms: ASK (amplitude shift keying), GFSK (Gaussian frequency SK), BPSK/QPSK (binary/quardrature phase SK), O-QPSK (offset-QPSK), OFDM (orthogonal frequency division multiplexing), COFDM (coded OFDM), MB-OFDM (multiband OFDM), M-QAM (M-ary quadrature amplitude modulation), CCK (complementary code keying), FHSS/DSSS (frequency hopping/direct sequence spread spectrum), BSS/ESS (basic/extended service set), AES (advanced encryption standard), WEP (wired equivalent privacy), WPA (Wi-Fi protected access), CBC-MAC (cipher block chaining message authentication code), CCM (CTR with CBC-MAC), CRC (cyclic redundancy check).

FCC power spectral density emission limit for UWB emitters operating in the UWB band is -41.3 dBm/Mhz. This is the same limit that applies to unintentional emitters in the UWB band, the so called Part 15 limit. The nominal transmission power is 0 dBm for both Bluetooth and ZigBee, and 20 dBm for Wi-Fi.

2.2.1 Radio Channels

Bluetooth, ZigBee and Wi-Fi protocols have spread spectrum techniques in the 2.4 GHz band, which is unlicensed in most countries and known as the industrial, scientific, and medical (ISM) band. Bluetooth uses frequency hopping (FHSS) with 79 channels and 1 MHz bandwidth, while ZigBee uses direct sequence spread spectrum (DSSS) with 16 channels and 2 MHz bandwidth. Wi-Fi uses DSSS (802.11), complementary code keying (CCK, 802.11b), or OFDM modulation (802.11a/g) with 14 RF channels (11 available in US, 13 in Europe, and just 1 in Japan) and 22 MHz bandwidth. UWB uses the 3.1-10.6 GHz, with an unapproved and jammed 802.15.3a standard, of which two spreading techniques, DSUWB and MB-OFDM, are available.

2.2.2 Coexistence Mechanism

Since Bluetooth, ZigBee and Wi-Fi use the 2.4 GHz band, the coexistence issue must be dealt with. Basically, Bluetooth and UWB provide adaptive frequency hopping to avoid channel collision, while ZigBee and Wi-Fi use dynamic frequency selection and transmission power control.

2.2.3 Network Size

The maximum number of devices belonging to the network's building cell is 8 (7 slaves plus one master) for a Bluetooth and UWB piconet, over 65000 for a ZigBee star network, and 2007 for a structured Wi-Fi BSS. All the protocols have a provision for more complex network structures built from the respective basic cells: the scatternet for Bluetooth, peer-to-peer for UWB, cluster tree or mesh networks for Zigbee, and the ESS for Wi-Fi.

2.2.4 Security

All the four protocols have the encryption and authentication mechanisms. Bluetooth uses the E0 stream cipher and shared secret with 16-bit cyclic redundancy check (CRC), while UWB and ZigBee adopt the advanced encryption standard (AES) block cipher with counter mode (CTR) and cipher block chaining message authentication code (CBC-MAC), also known as CTR with CBC-MAC (CCM), with 32-bit and 16-bit CRC, respectively.

In 802.11, Wi-Fi uses the RC4 stream cipher for encryption and the CRC-32 checksum for integrity. However, several serious weaknesses were identified by cryptanalysts, any wired equivalent privacy (WEP) key can be cracked with readily available software in two minutes or less, and thus WEP was superseded by Wi-Fi protected access 2 (WPA2), i.e. IEEE 802.11i standard, of which the AES block cipher and CCM are also employed.

2.2.5 Transmission Time

The transmission time depends on the data rate, the message size, and the distance between two nodes. The formula for transmission time (μs) can be described as:

$$T_{tx} = \left(N_{data} + \left(\frac{N_{data}}{N_{maxPld}} * N_{ovhd} \right) \right) * T_{bit} + T_{prop} \quad (1)$$

where N_{data} is the data size, N_{maxPld} is the maximum payload size, N_{ovhd} is the overhead size, T_{bit} is the bit time, and T_{prop} is the propagation time between any two devices. For simplicity, the propagation time is negligible in this paper. The typical parameters of the four wireless protocols used for transmission time evaluation are listed in Table 2. Note that the maximum data rate 110 Mbit/s of UWB is adopted from an unapproved 802.15.3a standard. As shown in Figure 3, the transmission time for the ZigBee is longer than the others because of the lower data rate (250 Kbit/s), while UWB requires less transmission time compared with the others. Obviously, the result also shows the required transmission time is proportional to the data payload size and disproportional to the maximum data rate.

Table 2.Typical system parameters of the wireless protocols

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
IEEE Spec.	802.15.1	802.15.3	802.15.4	802.11a/b/g
Max data rate (Mbit/s)	0.72	110*	0.25	54
Bit time (μ s)	1.39	0.009	4	0.0185
Max data payload (bytes)	339 (DH5)	2044	102	2312
Max overhead (bytes)	158/8	42	31	58
Coding efficiency* (%)	94.41	97.94	76.52	97.18
* Unapproved 802.15.3a.	* Where the data is 10K bytes.			

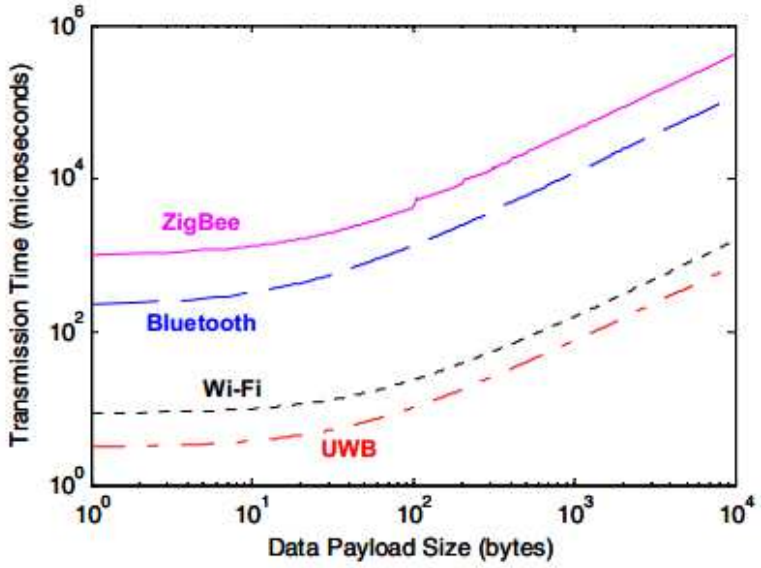

Figure 3.Comparison of the transmission time versus the data size.

2.2.6 Data Efficiency

Data coding efficiency is defined by the ratio of the data size and the message size (i.e. the total number of bytes used to transmit the data). The formula for data coding efficiency (%) can be described as:

$$P_{codEff} = \frac{N_{data}}{N_{data} + \left(\frac{N_{data}}{N_{maxPld}} * N_{ovhd}\right)} \quad (2)$$

The parameters listed in Table 2 are also used for the coding efficiency comparison. Figure 4 shows the data coding efficiency of the four wireless networks versus the data size. For small data sizes (around smaller than 339 bytes), Bluetooth is the best solution. Also, ZigBee has good efficiency for data size smaller than 102 bytes. For large data sizes, Bluetooth, UWB, and Wi-Fi have much better efficiency of over 94%, as compared to the 76.52% of ZigBee (where the data is 10K bytes as listed in Table 2). The discontinuities in Figure 3 and 4 are caused by data fragmentation, i.e. the maximum data payload, which is 339, 2044, 102, and 2312 bytes for Bluetooth, UWB, ZigBee, and Wi-Fi, respectively. In a Wi-Fi infrastructure mode, note that most APs connect to existing networks with Ethernet, and therefore limit the payload size to the maximum Ethernet payload size as 1500 bytes. However, for a general comparison, an ad-hoc mode is assumed and the 2312 bytes is adopted. For a wireless sensor network in factory automation systems, since most data size of industrial monitoring and control are generally small, (e.g. the temperature data in an environmental monitoring may require less than 4 bytes only), Bluetooth and ZigBee protocols may be a good selection (from a data coding efficiency point of view) in spite of their slow data rate.

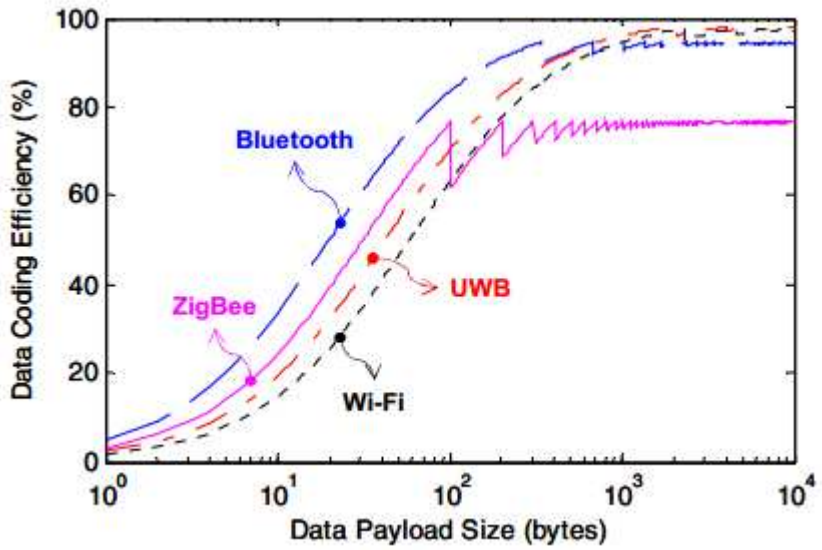

Figure 4.Comparison of the data coding efficiency versus the data size.

It is important to notice that several slight differences exist in the available sources. For example, in the IEEE 802.15.4 standard, the action range is about 10m, while it is 70-300m in the released documents from ZigBee Alliance. Thus, other factors such as receiver sensitivity and interference play a major role in affecting the performance in realistic implementations.

2.2.7 Protocol Complexity

The complexity of each protocol is compared based on the numbers of primitives and events. Table 3 shows the number of primitives and host controller interface (HCI) events for Bluetooth, and the numbers of MAC/PHY primitives for UWB, ZigBee, and Wi-Fi protocols. In the MAC/PHY layers, the Bluetooth primitives include client service access point (SAP), HCI SAP, synchronous connection-oriented (SCO) SAP, and logical link control and adaptation protocol (L2CAP) primitives. As shown in Figure 5, Bluetooth is the most complicated protocol with 188 primitives and events in total. On the other hand, ZigBee is the simplest one with only 48 primitives defined in 802.15.4. This total number of primitives is only about one fourth the number of primitives and events defined in Bluetooth. As compared with the Bluetooth, UWB, and Wi-Fi, the simplicity makes ZigBee very suitable for sensor networking applications due to their limited memory and computational capacity.

Table 3.Number of primitives and events for each protocol

Standard	Bluetooth	UWB	ZigBee	Wi-Fi	Standard
IEEE Spec.	802.15.1	802.15.3	802.15.4	802.11a/b/g	IEEE Spec.
Primitives	151	77*	35	32	MAC primitives
HCI events	37	29	13	43	PHY primitives
					* Approved 802.15.3b.

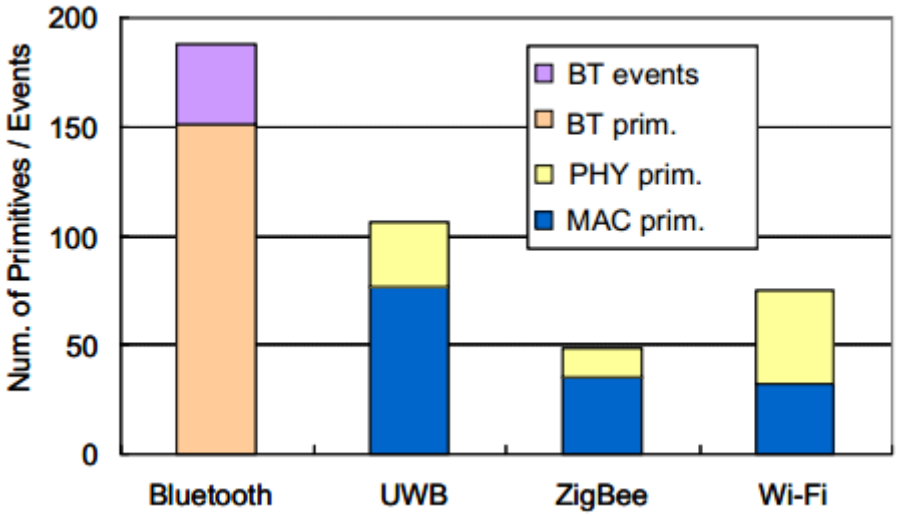

Figure 5.Comparison of the complexity for each protocol.

2.2.8 Power Consumption

Bluetooth and ZigBee are intended for portable products, short ranges, and limited battery power. Consequently, it offers very low power consumption and, in some cases, will not measurably affect battery life. UWB is proposed for short-range and high data rate applications. On the other hand, Wi-Fi is designed for a longer connection and supports devices with a substantial power supply. In order to practically compare the power consumption, four wireless products are briefly presented as an example, including BlueCore2 from Cambridge Silicon Radio (CSR), XS110 from Freescale, CC2430 from

Chipcon of Texas Instruments (TI), and CX53111 from Conexant (previous Intersil's Prism). The current consumptions of the transmit (TX) and receive (RX) conditions for each protocol are shown in Table 4. The data shown are for particular products, although are broadly representative for examples of the same type. Figure 6 indicates the power consumption in mW unit for each protocol. Obviously, the Bluetooth and ZigBee protocols consume less power as compared with UWB and Wi-Fi. Based on the bit rate, a comparison of normalized energy consumption is provided in figure 7. From the mJ/Mb unit point of view, the UWB and Wi-Fi have better efficiency in energy consumption.

Table 4.Current consumption of chipsets for each protocol

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
Chipset	BlueCore2	XS110	CC2430	CX53111
VDD (volt)	1.8	3.3	3.0	3.3
TX (mA)	57	~227.3	24.7	219
RX (mA)	47	~227.3	27	215
Bit rate (Mb/s)	0.72	114	0.25	54

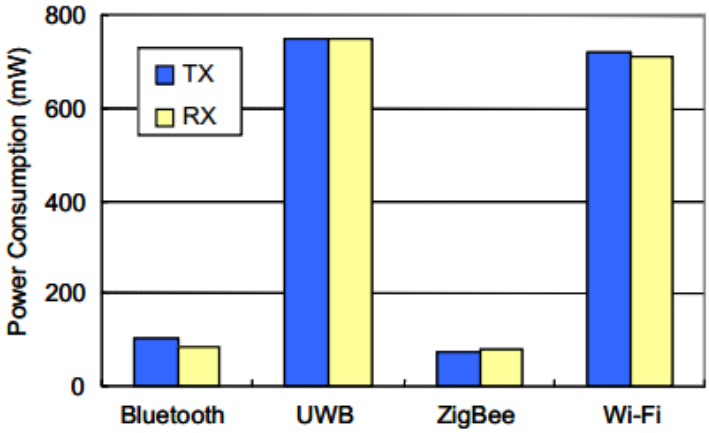

Figure 6.Comparison of the power consumption for each protocol

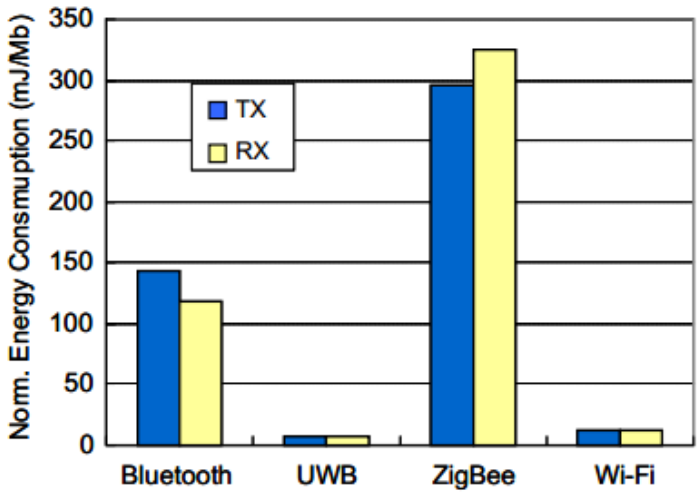

Figure 7.Comparison of the normalized energy consumption for each protocol

In summary, Bluetooth and ZigBee are suitable for low data rate applications with limited battery power (such as mobile devices and battery-operated sensor networks), due to their low power consumption leading to a long lifetime. On the other hand, for high data rate implementations (such as audio/video surveillance systems), UWB and Wi-Fi would be better solutions because of their low normalized energy consumption.

2.3 Radio Frequency Identification (RFID)

Radio Frequency Identification (RFID) is a system that transmits the identity of an object (in the form of a unique serial number) wirelessly, that is, using radio waves. RFID is similar to bar code identification systems. However, RFID does not rely on the line-of-sight reading that bar code scanning requires. The RFID system incorporates the use of electromagnetic or electrostatic feature in the Radio Frequency (RF) portion of the electromagnetic spectrum to uniquely identify, track, sort or detect a wide variety of tagged objects or assets.

2.3.1 RFID Overview

Most are well-acquainted with the emerging Radio Frequency (RF) discipline of short range communications. These include the services from the use of Near Field Communications (NFC), Radio Frequency Identification (RFID), and the combination of RFID and WSN to provide services in Real-time Location Systems (RTLS).

Short range communications has undeniably evolved along with technological advances. Today, its applications operate at ranges of low and high frequencies to microwave. The promise of wireless data capture and management of the data linked to a central server and the Internet is to eventually have industries dealing its business processes in real-time, or at almost near real-time with, for example its suppliers upstream and its distributors downstream in end-to-end connected network.

This real-time sort of communications will substantially increase the efficiency, speed, productivity, and security in, for example, supply chain management. Added value is expected to be further generated after undergoing re-engineering process in accordance to the business applications.

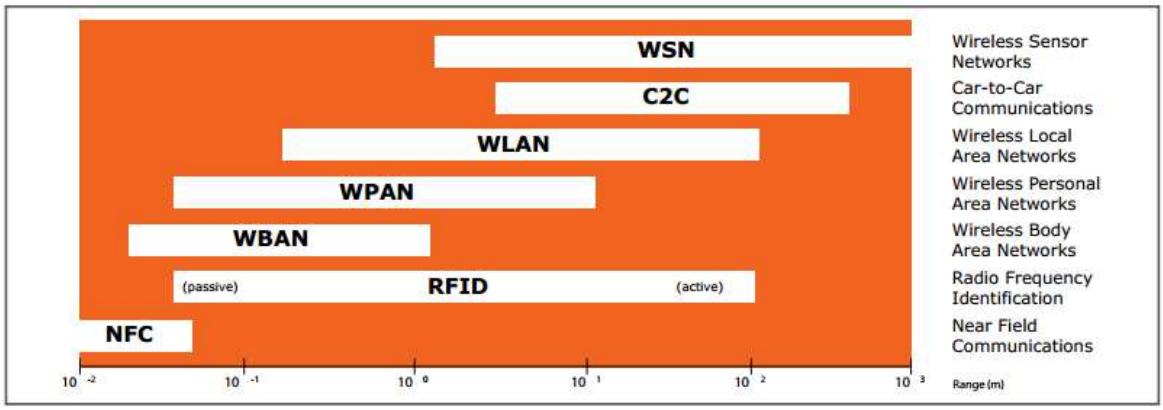

Figure 8.RFID in the world of short range communications

Type	Key Primary Characteristics	Examples of applications adopted
NFC	13.56MHz (frequency); 410Kbps data transmission rate; and more than three meters reading range	Hands-free earpiece and headphones; secure verification and validation; and contactless payments
RFID	Low frequency; High frequency and Ultra High frequency. Ranging from centimetres to metres (system dependent)	Toll collection and traffic control management, inventory control and supply chain management
WSN/RFID (e.g., RTLS)	Low frequency, simple installation Low Range/Low infrastructure, high accuracy	Indoor tracking

Figure 9.Short range communications

Year	RFID Development	RFID Application	Countries	Inventor/ Vendors	Evolution of Major Uses
1935 (during World War II)	First passive RFID-like method	Using radar which signal on approaching planes (lacking a primary feature to differentiate between enemy or alliance aircraft).	Germany, Japan, US and UK	Discovered by Sir Robert Alexander Watson-Watt, Scottish physicist	Vehicle tracking
1950-1960's	First active friend or foe (IFF) system, long-range transponder systems	Using a range transponder systems installed on aircraft that signal a wave to be read by fire-control crews	UK	Sir Robert Alexander Watson-Watt, Scottish physicist	Better tracking of aircraft
	"One-bit" tags, using Electronic Article Surveillance (EAS) systems	First commercial and most widespread use of RFID - using inexpensive tag with microwave or inductive technology	US	Checkpoint Systems Inc., Sensormatic and Knogo	Inventory tracking/ Anti-theft/ Anti-shoplifting measures
1970s	First Patented Active RFID tag with rewritable memory	-	US	Mario W.Cardullo	-
	Patented Passive Transponder	To unlock a door without a key – using a card with embedded transponder waves a signal to a reader near the door.	US	Charles Walton	Access to building
	Invented concept of putting a transponder in the truck and readers at the gates of secure facilities	Tracking nuclear materials Leading solution for real-time data communications, automatic vehicle tracking and satellite positioning for the transportation industry	US	Los Alamos National Laboratory	Tracking item movement
1980s	Wireless data solutions using OmniTRACS® satellite mobile communications system (Fleet-management systems)	Leading solution for real-time data communications, automatic vehicle tracking and satellite positioning for the transportation industry	US	Qualcomm	Transportation systems (railroads/ trucks
	System consisting of transponder (an ID) in the vehicle would respond to gate antenna	Commercial usage for toll collection	US	Los Alamos National Laboratory	Automated toll payment systems (widely deployed globally for roads, bridges and tunnels)

Figure 10.Key milestones for RFID technology (1930-1980s)

In much longer range platforms, RFID technology has been commercially available since two decades ago in the 1980s.

The 1990s and the first half decade of 2000s saw the accelerated adoption of RFID to mainstream use upon the adoption of global standards, and the US government and global retail stores mandate for use of RFID tags in their supply chain. Today, we see the potential of the use of RFID towards eventual machine to machine communications in the connection of a network of things in many environments of the home, workplace and even in wearable or personal accessories.

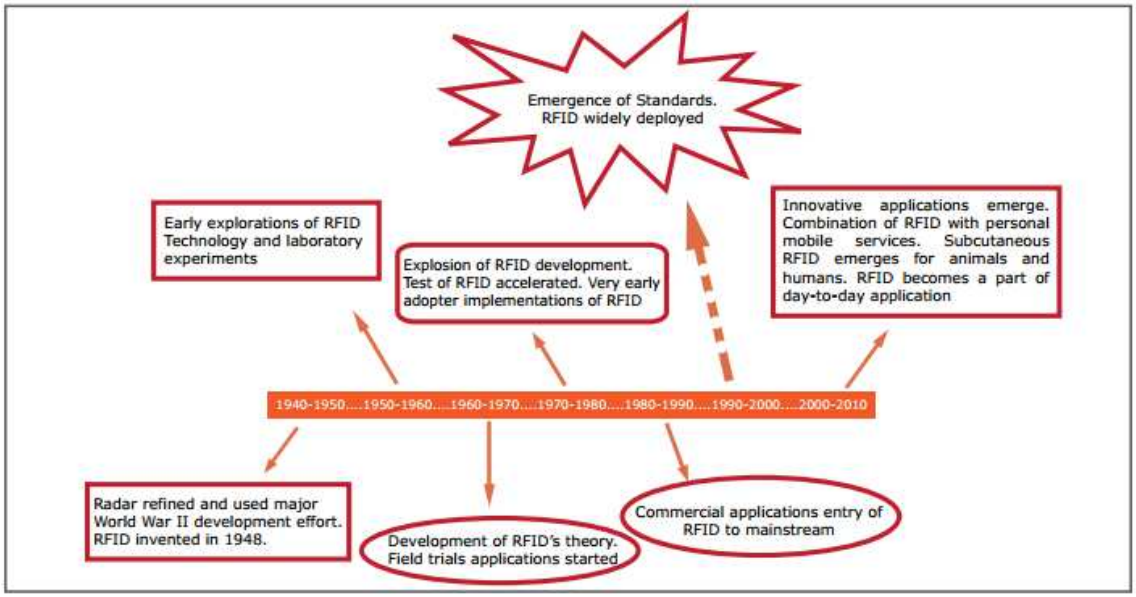

Figure 11.Decades of RFID frontier

Initially, RFID was perceived by many as a bar code replacement. At this time, the technology adoption was purely compliance-driven in most cases in closed-loop environment to automate business processes. Such is the case of Wal-Mart/Sam's Club mandates where many Consumer Packaged Goods (CPG) manufacturers were among the most significant in adopting RFID for satisfying these compliance requirements. A comparison between RFID and bar code can be observed in figure 12.

In others, it complements bar codes usage such as, e.g., all packages containing iPhone 4S mobile phones have the same bar code. However, with RFID tags, each packet of these blades would have its own unique identifier that can be transmitted to suitably located readers for monitoring purposes.

RFID is considered a mature technology in that its evolution can be traced back to technologies developed during World War II. Essentially, it is today used to provide various solutions such as in commercial supply chain management with its relatively simple, unobtrusive and cost-effective system beyond its original applications in the military. RFID is also making its way for broader adoption in various industries such as building access control, animal tracking, postal item tracking, anti-counterfeiting in the pharmaceutical industry and other applications where for example, control, speed, and data capture for monitoring and management control are prime requirements.

System	Bar Code	RFID
Data Transmission	Optical	Electromagnetic
Memory/Data Size	Up to 100 bytes	Up to 128Kbps
Tag Writable	No	Possible
Position of Scan/Reader	Line-of-sight	Non-line-of-sight
Read Range	Up to several metres (line-of sight)	Centimetres to metres (system dependent)
Access Security	Low	High
Environmental Susceptibility	Dirt	Low
Anti-collision	Not possible	Possible

Figure 12.Comparison of bar code and RFID

The RFID adoption and application scope figure 13 provides insight into the next levels of RFID usage in re-engineered workflows to new process models. In these levels of adoption, the adopters could enjoy cheaper price of tags through scale economies as tags are more widely applied to improve business process and minimise human intervention. The lower prices such as for passive tags and also for readers generally influence (like in a virtuous cycle) wider innovation in RFID adoption and its applications. Such combinations can impact positively on the bottom-line.

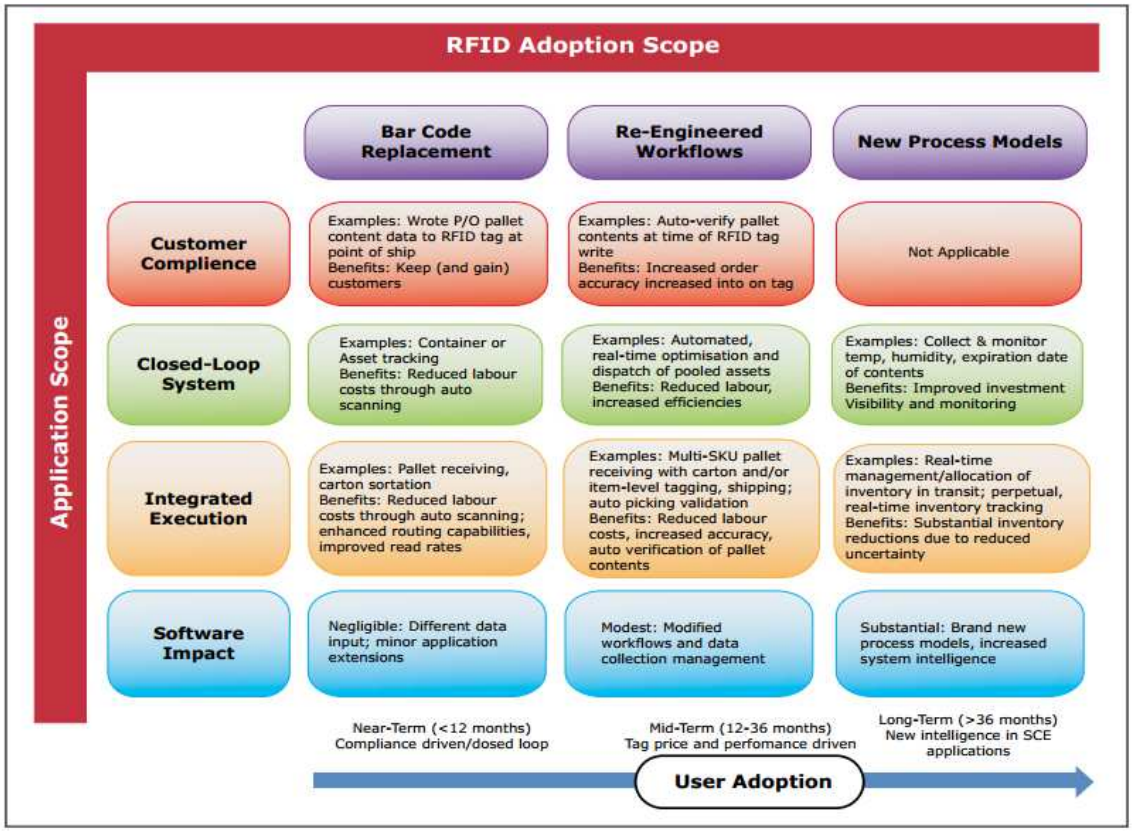

Figure 13.RFID Adoption and application scope

Besides that, the automated business processes are often associated with real-time optimisation of deriving information from the tags. This has proven to contribute to significant increase in efficiencies as well. For example, Marks & Spencer rolled out RFID usage in 53 stores across the UK, tagging men's and women's suits. The monitoring capability enabled through the RFID based information system provides advantages to the extent of ensuring the shelves are not left empty. That is, the ready availability of products a customer wants to buy reduces the opportunity loss of the same customer buying an alternative product, thus making the sale. The use of RFID also allows scalability and system integration which can be customised in accordance to individual context of use requirements.

Therefore, the application scope becomes more sophisticated as business process improvement and efficiency are derived. This is from the improved (including real-time) inventory system in terms of visibility and monitoring. With new process models, the user can experience more benefits and value propositions including system enhancement along the supply chain.

2.3.2 RFID System Components

Effectively, the RFID system components comprise the RFID Subsystem and the Enterprise Subsystem which essentially deals with the function of readers reading the information from the tags. The RFID Enterprise subsystem deals with the information processing aspect, that is, the information collected and its subsequent use in analytical system for purposes of monitoring and management control. Figure 14 visually shows the components and provides a conceptual view of the dynamic relationship of one component of the system to another.

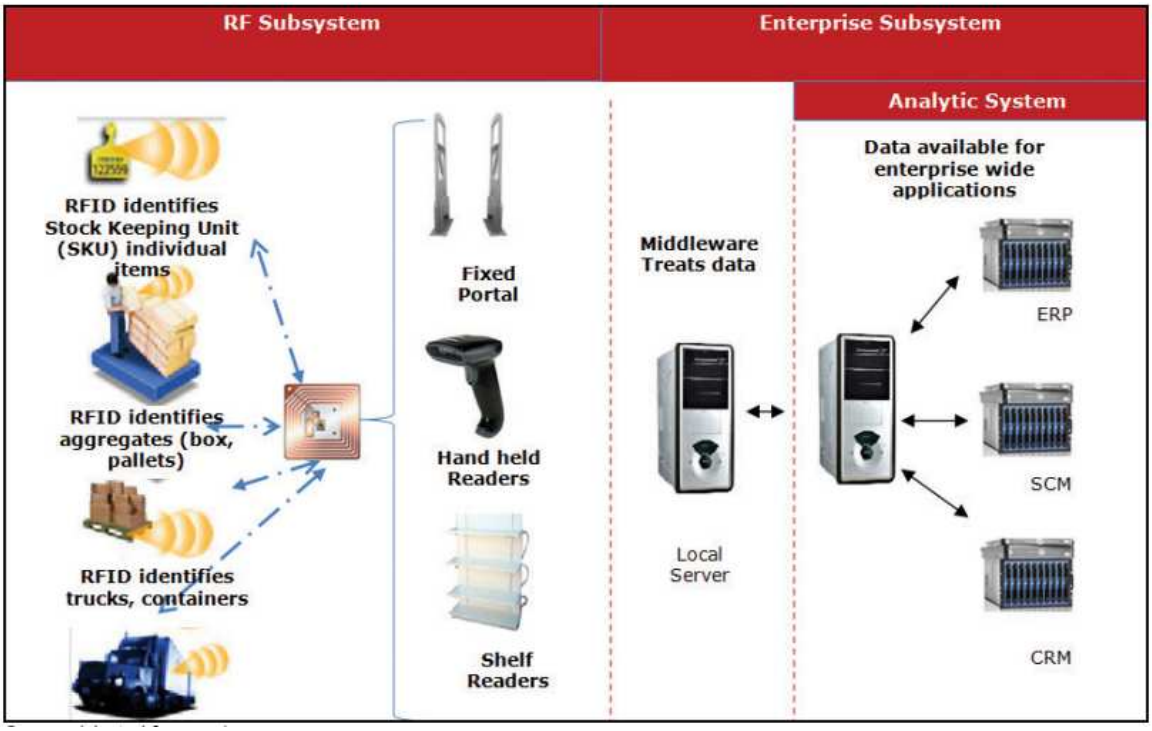

Figure 14.RFID system components

RFID Subsystem

In the RFID subsystem, the communication is between an interrogator (reader) and a transponder (silicon chip connected to an antenna) often called a tag. In an RFID setup, the RFID tags are interrogated by an RFID reader, that is, the tag reader generates a radio frequency interrogation signal that communicates with the tags. When a tag fixed to, for example, an asset passes through the field of the scanning reader, it detects the activation signal from the reader and wakes up the RFID microchip in the tag containing the unique ID. The chip then transmits this information to the scanning reader.

Figure 15.Examples of RFID subsystem components

Tags/Transponders

RFID tags have two basic elements: a chip and an antenna. The chip and antenna are mounted to form an inlay, which means the tag chip itself is connected to the antenna; using techniques such as wire bonding or flip chip.

Reader/Decoder/interrogator

An RFID interrogator is a transmitter/receiver that reads the contents of RFID tags. It acts as a conduit or bridge between the RFID tag and the controller or middleware. The maximum distance by which the reader antenna and the tag can communicate varies depending on the application.

RFID interrogator is composed of roughly three parts which are an antenna, a RF electronics module (responsible for communicating with the RFID tag) and a controller electronics module (responsible for communicating with the controller).

Sensor

Sensor is a device that responds to a physical stimulus and produces an electronic signal. Sensors are increasingly combined with RFID tags to detect the presence of a stimulus at an identifiable location. Ultimately, RFID are evolving to intelligent devices that have networking capabilities allowing for wider applications using WSN. In addition, sensors

combined with RFID tags open new possibilities to monitor and transmit various parameters like temperature, humidity, pressure, acceleration, position, or sound level.

An example of use of sensors is taking the temperature readings of a medicine that has to be kept at a certain temperature in transport or storage. This information is kept in the (active) RFID tag which can transmit the temperature readings to a central server in the RFID system for purposes of monitoring storage quality control.

Development of RFID Chips and Tags

There are two basic types of chips available on RFID tags: read-only and read-write. According to the majority of sources found in the internet, read-only tags cost less than read/write tags and the infrastructure required to support is also less expensive. The read-only tags still deliver on one of the main promises of RFID, that is, reduced operator involvement. This translates into lower operating costs and has minimal impact on employee work processes or job functions.

The read-write tags are dynamic data carrier, which means that users can add or change the data on the tag. Therefore, a read-write tag can respond quickly to changing conditions in the supply chain. Users can have more information stored locally on the tag. This enables faster processing, reduces data latency and may not require a database lookup or any contact with an external system.

Characteristics of the RFID chips are described in more detail in table 5.

Table 5.Basic types of RFID chips

Type	Description
Read-Only	• Programmed with unique information stored on them during the manufacturing process. • The information that can be read only and cannot be changed.
Read-Write	• User can add information to the tag or write over existing information when the tag is within range of the reader. • More expensive than read-only chips. • The application may include field service maintenance or item attendant data – where a maintenance record associated with a mechanical component is stored and updated on a tag attached to the component. • Another method used is WORM chips (Write Once Read Many) – can be written once and then becomes Read-only afterwards.

2.3.3 Primary Frequencies for RFID

Generally, there are four primary frequency bands allocated for RFID use which is Low Frequency (LF), High Frequency (HF), Ultra High Frequency (UHF) and microwave. Its specifications can be observed in table 6. Nevertheless, different countries have respectively specific frequency bands stipulated for RFID usage.

Table 6.Commonly used RFID frequencies for passive tags

	LF	HF	UHF	Microwave
Frequency Range	< 135KHz	13.56MHz	860 – 930MHz	2.45GHz
Standard Specifications	ISO/IEC 18000-2	ISO/IEC 18000-3 AutoID HF class 1 ISO 15693, ISO 14443 (A/B)	ISO/IEC 18000-6 AutoID class 0, class 1	ISO/IEC 18000-4
Typical Read Range	Less than 0.5metre	About 1metre	About 4 to 5metre	About 1metre
Tag Power Source	Mainly passive using inductive coupling (near field)	Mainly passive using inductive coupling (near field)	Active and passive tags using electromagnetic field back scatter	Active and passive tags using electromagnetic field back scatter
Typical Applications	Access control, animal tagging, vehicle immobiliser	Smart card, access control, payment, ID, item level tagging, baggage control, library, biometric, laundry, transport, apparel	Supply chain pallet and box tagging, baggage handling, electronic toll collection	Electronic toll collection, real time location of goods
Multiple Tag Read Rate	Slower ⟷			Faster
Ability to read near metal or wet surface	Better ⟷			Worse
Passive Tag Size	Larger ⟷			Smaller

2.3.4 Passive and Active Tags

Tags in the RFID systems are also classified either as active (powered by battery) or passive (powered by the reader's electromagnetic field) and come in various forms including smart cards, tags, labels, watches and even embedded in mobile phones. The basic passive tags are the most widely used in RFID applications. An example of use is in food traceability. RFID can deliver significant benefits to businesses requiring tight deadlines to deliver goods to shelves during the retail operation, such as in the perishable goods industry. It can record ambient data, such as temperature or humidity to indicate the remaining product shelf life.

Nevertheless, as the networked environment further unfolds to support real-time tracking systems within a more connected ecosystem of users, there is expected wider usage of active RFID tags. An example of use of active tags is in warehouse asset tracking applications. These tags are programmed with certain information and assigned locations and then placed on containers and pallets stored in a warehouse. The systems can even alert management and security when unscheduled movements occur; also reduce costs and time for check-in and check-out as containers and pallets enter and leave the warehouse.

Semi-passive RFID tags use a process to generate a tag response similar to that of passive tags. Semi-passive tags differ from passive ones in that semi-passive tags possess an internal power source (battery) for the tag circuitry which allows the tag to complete other functions such as monitoring of environmental conditions (temperature, shock) or extending the tag signal range.

Table 7.Comparison of passive and active tags

Type	Advantages	Disadvantages	Remarks
Passive	• Longer life time with the lowest cost • Wider range of form factors • Tags are more mechanically flexible	• Communication distance is limited with the range from 4-5metres using the UHF frequency band • Strictly controlled by local regulations	• Most widely used in RFID applications. • Tags are LF, HF or UHF
Semi-Passive	• Tags have built-in batteries and do not require energy from the reader to power the chip. • Greater communication distance of up to 100 metres • Can be used to manage other devices like sensors (Temperature, pressure, others) • Do not fall under the same strict power regulations imposed on passive devices	• Expensive – due to battery, and tag packaging • Reliability – impossible to determine whether a battery is good or bad, particularly is multiple transponder environments	• Used mainly in real-time systems to track high value materials or equipment throughout a factory • Tags are UHF
Active			• Greatest communication distance • Used in logistics for tracking of containers on trains, trucks and others. • Tags are UHF or microwave

2.3.5 Evolution of RFID Active Tags

There are observations of overall, three generations of active RFID usage. The first generation or traditional active RFID has been around for the last 60 years, followed by those used in Real-time Location Systems (RTLS) over the last 12 years. The third generation called Wireless Sensor Networks is now becoming more widely available. The timeline where each generation will gain more market share is appreciated in figure 16.

With each generation, there is seen expansion in RFID application scope from providing information on tags in close-loop system in remote use, such as in a factory, to more integrated execution across a network (through a Virtual Private Network or Internet) across geographically separated locations. Parallel to this development, the volume of tag use also increases.

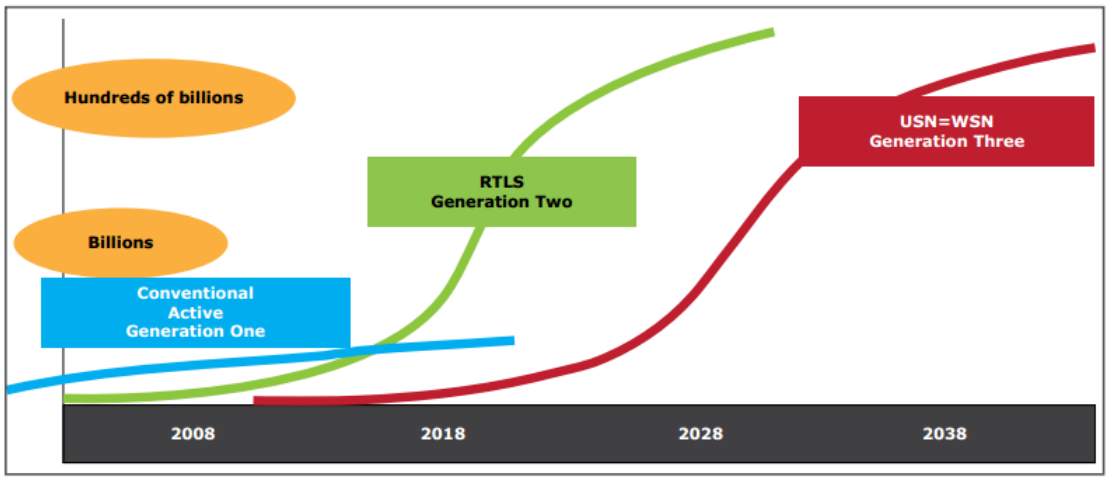

Figure 16.High volume active RFID lies in the future

First generation of RFID (Conventional RFID)

Conventional RFID is about a battery in a tag that permits a signal to be initiated, provide longer range or distance, manage a sensor or otherwise improve on the capability of a passive RFID tag. For example, an application of conventional RFID that locks and unlocks car at a distance of 30 metres or tags placed into letter post envelopes to monitor the performance of postal services.

Second generation of RFID/Real Time Location System (RTLS)

RTLS is a system that is able to seek continuously and in real-time determines the position of a person or object from a distance within a physical space of 30 to 300 metres away. For example, to monitor assets within an entire hospital, the intensive care unit or a patient's room.

Third Generation: Wireless sensor network

WSN is a third generation form of RFID that involves an unprecedented amount of memory and processing power at the node to perform tasks that earlier forms of RFID cannot address.

WSN is about attaching sensors to each RFID tag that then doubles in capability as a reader with mesh network. It is limited to 30 metres range and the tag may be rather like active RFID tags from ten years ago and even use AAA batteries.

This capability is notable because it can make systems scalable, self-healing, affordable and extraordinarily "capable".

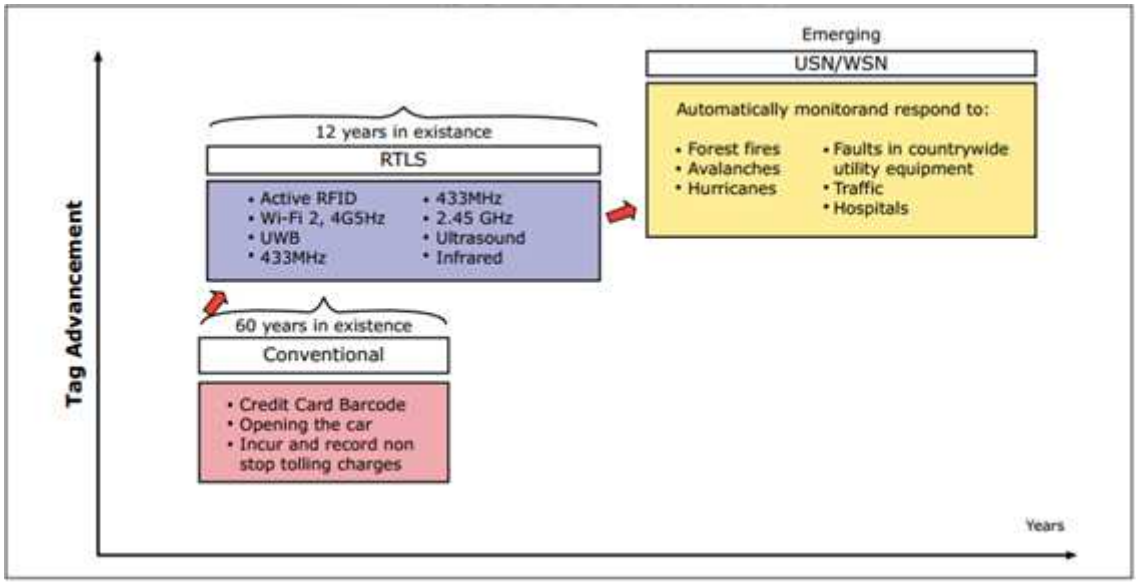

Figure 17.Three generations of active RFID

2.3.6 RFID Security and Privacy

With the adoption of RFID globally so far, there is now not only greater awareness of privacy concerns but also a heightened focus on the development of privacy standards. Privacy is one of the main concerns of RFID faced today. Consumers and citizens are concerned that they

could be 'read' from a distance and this being done without their knowledge. People want to know when they are being 'read' such as when their belongings are being scanned.

Privacy and security are two different things and are addressed separately in the context of RFID. 'Privacy' deals with keeping the meaning of the information in transition while 'security' is about the information in transition only.

Both privacy and security issues have different repercussions and solutions. For example, in a situation when a tag exposes its unique identification number in an unencrypted manner, it would enable any reader to process the signal. This deals with a security issue where the unique identifier can only mean a number as long as the user does not have access to the backend database that shows the relationship between the unique identifier of the tag and the object.

However, the issue on traceability and inventory may still remain.

Four of the most RFID Security Threats are the following:

• Eavesdropping: Unauthorized users take opportunity to eavesdrop while data is in transmission between a reader and a tag.

• Spoofing: Revealed security protocol in the RFID channel could allow a dishonest person to replace the information stored on the tag with a much lower price.

• Relay attack/Cloning: This type of threat is about cloning or imitating genuine data.

• Industrial Sabotage: Someone who has something against a company and decides to corrupt, modify and edit data in tags using hand held device.

And the RFID Security Methods to counter strike the previous threads are:

• Faraday Cage or Shield: This provides natural barriers to radio waves. This requires manual action required by a user to cover and uncover the tag every time the user wants the tag to function.

• Limited Range Transmission: Products can only be scanned if in a close proximity. Assuming that unintended reader in close proximity would be easily identified making it difficult for stealing the information.

• KILL Command: The KILL command makes the tag unreadable provided that the PIN is successfully transmitted to the tag. The tag is made unreadable at the point of sale. This command prevents the use of the tag that was KILLED from being used again for future applications.

• SLEEP Command: This tag in contrast to the KILL command tag temporarily deactivated at the point of sale.

• Clip Tags: Reduces the readable range from a few centimetres to one or two centimetres

2.3.7 Some Key Criteria to Be Considered When Selecting RFID and Conclusions

While many supply and demand determinants drive the market for RFID technologies, existing users and potential users continuously explore the utilisation of RFID at optimum level in order to either search for problem-solving solutions at pilot phase or gain greater business value at post-pilot phase. Based on solid business cases for leveraging the use of RFID and its successful case-studies previously mentioned, these elements may spark interest and influence users to consider in their pursuit of RFID start-ups.

However, RFID adoption is not as simple as it sounds. Implementation requires some level of specialized technical expertise in order to capture the dramatic effects from the business value chain. In additions obstacles ranging from the initial selection of hardware and software, its setup, fine-tuning the system processes as well as maintenance needs may arise.

All said then, RFID has a crucial role to play in business and for individuals alike going forward. The impact of 'wireless' identification is exerting strong pressures in RFID technology and services research and development, standards development, security compliance and privacy, and many more. The economic value is proven in some countries while others are just on the verge of planning or in pilot stages, but the wider spread of usage has yet to take hold or unfold through the modernisation of business models and applications. Therein lies the potential and opportunities to capture the benefits from RFID solution for local enterprise productivity gains and competitive innovation.

RFID technology is also perceived as an open door to a 'new phase of development of the Information Society, often referred to as the Internet of things' in ubiquitous connectivity. With a strategic and proper alignment towards the identified value that an RFID solution can add to the business, RFID could explode beyond its initial applications as an emerging technology arm in the supply chain to the world of 'Internet of things' at large.

Despite the fact that its promising technological capabilities into the mainstream of business and society, RFID does pose challenges in terms of legal transparency, standards, product information services and compatibility issues, privacy and security and other security threats such as spoofing, industrial sabotage and espionage.

The answer, nevertheless, invariably resides in the fundamentals of proper understanding, holistic view in planning, and perfect execution in collaborative mode. In summary, there is some pain to be mitigated in RFID adoption, but its service does promise much gain.

3. Wireless Sensor Network Applications

Possible applications of sensor networks are of interest to the most diverse fields. Environmental monitoring, warfare, child education, surveillance, micro-surgery, and agriculture are only a few examples. Firstly, we are going to check some real hardware applications in the United States of America (USA) as it is probably the country that has investigated most in this area.

Through joint efforts of the University of California at Berkeley and the College of the Atlantic, environmental monitoring is carried out off the coast of Maine on Great Duck Island by means of a network of Berkeley motes equipped with various sensors. The nodes send their data to a base station which makes them available on the Internet. Since habitat monitoring is rather sensitive to human presence, the deployment of a sensor network provides a non-invasive approach and a remarkable degree of granularity in data acquisition. The same idea lies behind the Pods project at the University of Hawaii at Manoa, where environmental data (air temperature, light, wind, relative humidity and rainfall) are gathered by a network of weather sensors embedded in the communication units deployed in the South-West Rift Zone in Volcanoes National Park on the Big Island of Hawaii. A major concern of the researchers was in this case camouflaging the sensors to make them invisible to curious tourists. In Princeton's Zebranet Project, a dynamic sensor network has been created by attaching special collars equipped with a low-power GPS system to the necks of zebras to monitor their moves and their behaviour. Since the network is designed to operate in an infrastructure-free environment, peer-to-peer swaps of information are used to produce redundant databases so that researchers only have to encounter a few zebras in order to collect the data. Sensor networks can also be used to monitor and study natural phenomena which intrinsically discourage human presence, such as hurricanes and forest fires. Joint efforts between Harvard University, the University of New Hampshire, and the University of North Carolina have recently led to the deployment of a wireless sensor network to monitor eruptions at Volcán Tungurahua, an active volcano in central Ecuador. A network of Berkeley motes monitored infrasonic signals during eruptions, and data were transmitted over a 9 km wireless link to a base station at the volcano observatory.

Intel's Wireless Vineyard is an example of using ubiquitous computing for agricultural monitoring. In this application, the network is expected not only to collect and interpret data, but also to use such data to make decisions aimed at detecting the presence of parasites and enabling the use of the appropriate kind of insecticide. Data collection relies on data mules, small devices carried by people (or dogs) that communicate with the nodes and collect data. In this project, the attention is shifted from reliable information collection to active decision-making based on acquired data.

Just as they can be used to monitor nature, sensor networks can likewise be used to monitor human behaviour. In the Smart Kindergarten project at UCLA, wirelessly-networked, sensor-enhanced toys and other classroom objects supervise the learning process of children and allow unobtrusive monitoring by the teacher.

Medical research and healthcare can greatly benefit from sensor networks: vital sign monitoring and accident recognition are the most natural applications. An important issue is the care of the elderly, especially if they are affected by cognitive decline: a network of

sensors and actuators could monitor them and even assist them in their daily routine. Smart appliances could help them organize their lives by reminding them of their meals and medications. Sensors can be used to capture vital signs from patients in real-time and relay the data to handheld computers carried by medical personnel, and wearable sensor nodes can store patient data such as identification, history, and treatments. With these ideas in mind, Harvard University is cooperating with the School of Medicine at Boston University to develop CodeBlue, an infrastructure designed to support wireless medical sensors, PDAs, PCs, and other devices that may be used to monitor and treat patients in various medical scenarios. On the hardware side, the research team has created Vital Dust, a set of devices based on the MICA2 1 sensor node platform (one of the most popular members of the Berkeley motes family), which collect heart rate, oxygen saturation, and EKG data and relay them over a medium-range (100 m) wireless network to a PDA. Interactions between sensor networks and humans are already judged controversial. The US has recently approved the use of a radio-frequency implantable device (VeriChip) on humans, whose intended application is accessing the medical records of a patient in an emergency. Potential future repercussions of this decision have been discussed in the media.

An interesting application to civil engineering is the idea of Smart Buildings: wireless sensor and actuator networks integrated within buildings could allow distributed monitoring and control, improving living conditions and reducing the energy consumption, for instance by controlling temperature and air flow.

Military applications are plentiful. An intriguing example is DARPA's self-healing minefield, a self-organizing sensor network where peer-to-peer communication between anti-tank mines is used to respond to attacks and redistribute the mines in order to heal breaches, complicating the progress of enemy troops. Urban warfare is another application that distributed sensing lends itself to. An ensemble of nodes could be deployed in an urban landscape to detect chemical attacks, or track enemy movements. PinPtr is an ad hoc acoustic sensor network for sniper localization developed at Vanderbilt University. The network detects the muzzle blast and the acoustic shock wave that originate from the sound of gunfire. The arrival times of the acoustic events at different sensor nodes are used to estimate the position of the sniper and send it to the base station with a special data aggregation and routing service.

Going back to peaceful applications, efforts are underway at Carnegie Mellon University and Intel for the design of IrisNet (Internet-scale Resource-Intensive Sensor Network Services), an architecture for a worldwide sensor web based on common computing hardware such as Internet-connected PCs and low-cost sensing hardware such as webcams. The network interface of a PC indeed senses the virtual environment of a LAN or the Internet rather than a physical environment; with an architecture based on the concept of a distributed database, this hardware can be orchestrated into a global sensor system that responds to queries from users.

But not just USA has been using and investigating with wireless sensor networks. University of Southampton is to develop technology to monitor glacier behaviour using sensor networks contributing to fundamental research in glaciology and wireless sensor networks. Custom sensor probes are placed in, on and under glaciers and data collected from them by a base station on the surface. Measurements include temperature, pressure, stress, weather and sub glacial movement. The information gathered is important in understanding the dynamics of glaciers as well as global warming. Coalesenses GmbH, in Germany, is working in an iSense hardware platform; ETH Zurich has developed the BTnode, an autonomous wireless

communication and computing platform based on a Bluetooth radio and a microcontroller; and finally Libelium, the whose sensor mote called *Waspmote* will be examined more thoroughly for being innovative, adaptable multipurpose and Spanish, specifically from Zaragoza.

3.1 *Waspmote*

More than 50 billion devices will be connected to the Internet by 2020, but this new connectivity revolution has already started. Libelium (www.libelium.com) has published a compilation of 50 cutting edge Internet of Things applications grouped by vertical markets. For this end, Libelium has developed the so called *Waspmote* sensor which contains a large number of wireless standards and even larger sensor boards, depending on the final application. It is even possible to customize your own sensing board.

Figure 18.Libelium product called *Waspmote*

The exponential growing number of objects connected to the Internet is changing completely our world. What new business models will appear? Which processes can be optimized? How many vertical markets are benefited? Libelium has released the document *"50 Sensor Applications for a Smarter World. Get Inspired!"* covering the most disruptive sensor and Internet of Things applications. The list is grouped in 12 different verticals, showing how the Internet of Things is becoming the next technological revolution. It includes the trendiest scenarios, like Smart Cities where sensors can offer us services like managing the intensity of the luminosity in street lights to save energy. Climate change, environmental protection, water quality or CO2 emissions are also addressed by sensor networks and are just some of the examples included in the Smart Water and Smart Environment sections included in the document.

Other sections such as Industrial Control, Logistics or Retail cover applications more focused in process efficiency like providing information for restocking the shelves and even product placement for marketing purposes. The list is completed with applications in the verticals of Smart Metering, Security and Emergencies, Smart Agriculture, Animal Farming, Domotic, Home Automation and eHealth.

3.1.1 *Waspmote* application areas

Regarding Smart Environment applications *Waspmote* can be used for:

- Forest Fire Detection: Monitoring of combustion gases and pre-emptive fire conditions to define alert zones.
- Air Pollution: Control of CO2 emissions of factories, pollution emitted by cars and toxic gases generated in farms.
- Landslide and Avalanche Prevention: Monitoring of soil moisture, vibrations and earth density to detect dangerous patterns in land conditions.
- Earthquake Early Detection: Distributed control in specific places of tremors.

There are applications in the Smart Water Area too:

- Water Quality: Study of water suitability in rivers and the sea for fauna and eligibility for drinkable use.
- Water Leakages: Detection of liquid presence outside tanks and pressure variations along pipes.
- River Floods: Monitoring of water level variations in rivers, dams and reservoirs.

There are plenty of applications for *Waspmote* in the Urban Areas such as:

- Smart Parking: Monitoring of parking spaces availability in the city.(This one will be explained more thoroughly in section 3 of this chapter)
- Structural health: Monitoring of vibrations and material conditions in buildings, bridges and historical monuments.
- Noise Urban Maps: Sound monitoring in bar areas and centric zones in real time.
- Traffic Congestion: Monitoring of vehicles and pedestrian levels to optimize driving and walking routes.
- Smart Lightning: Intelligent and weather adaptive lighting in street lights.
- Waste Management: Detection of rubbish levels in containers to optimize the trash collection routes.
- Intelligent Transportation Systems: Smart Roads and Intelligent Highways with warning messages and diversions according to climate conditions and unexpected events like accidents or traffic jams.

There are a large number of situations in which measures must be made, and *Waspmote* is helpful in that as well.

- Smart Grid: Energy consumption monitoring and management.
- Tank level: Monitoring of water, oil and gas levels in storage tanks and cisterns.
- Photovoltaic Installations: Monitoring and optimization of performance in solar energy plants.
- Water Flow: Measurement of water pressure in water transportation systems.
- Silos Stock Calculation: Measurement of emptiness level and weight of the goods.

In the Security & Emergency area, there are several things that Libelium's sensor can do.

- Perimeter Access Control: Access control to restricted areas and detection of people in non-authorized areas.
- Liquid Presence: Liquid detection in data centres, warehouses and sensitive building grounds to prevent break downs and corrosion.
- Radiation Levels: Distributed measurement of radiation levels in nuclear power stations surroundings to generate leakage alerts.
- Explosive and Hazardous Gases: Detection of gas levels and leakages in industrial environments, surroundings of chemical factories and inside mines.

The sensor responsible of this application list has a lot to do in the retail business as well.

- Supply Chain Control: Monitoring of storage conditions along the supply chain and product tracking for traceability purposes.
- NFC Payment: Payment processing based in location or activity duration for public transport, gyms, theme parks, etc.
- Intelligent Shopping Applications: Getting advices in the point of sale according to customer habits, preferences, presence of allergic components for them or expiring dates.
- Smart Product Management: Control of rotation of products in shelves and warehouses to automate restocking processes.

Waspmote can be used for logistics too:

- Quality of Shipment Conditions: Monitoring of vibrations, strokes, container openings or cold chain maintenance for insurance purposes.
- Item Location: Search of individual items in big surfaces like warehouses or harbours.

- Storage Incompatibility Detection: Warning emission on containers storing inflammable goods closed to others containing explosive material.
- Fleet Tracking: Control of routes followed for delicate goods like medical drugs, jewels or dangerous merchandises.

The earlier mentioned sensor has applications for the industrial control:

- M2M Applications. Machine auto-diagnosis and assets control.
- Indoor Air Quality: Monitoring of toxic gas and oxygen levels inside chemical plants to ensure workers and goods safety.
- Temperature Monitoring: Control of temperature inside industrial and medical fridges with sensitive merchandise.
- Ozone Presence: Monitoring of ozone levels during the drying meat process in food factories.
- Indoor Location: Asset indoor location by using active (ZigBee) and passive tags (RFID/NFC).
- Vehicle Auto-diagnosis: Information collection from CanBus to send real time alarms to emergencies or provide advice to drivers.

As to agriculture related the following list are some of the many things that can be done:

- Wine Quality Enhancing: Monitoring soil moisture and trunk diameter in vineyards to control the amount of sugar in grapes and grapevine health.
- Green Houses: Control micro-climate conditions to maximize the production of fruits and vegetables and its quality.
- Golf Courses: Selective irrigation in dry zones to reduce the water resources required in the green.
- Meteorological Station Network. Study of weather conditions in fields to forecast ice formation, rain, drought, snow or wind changes.
- Compost: Control of humidity and temperature levels in alfalfa, hay, straw, etc. to prevent fungus and other microbial contaminants.

Continuing with rural applications, *Waspmote* can be used for:

- Offspring Care: Control of growing conditions of the offspring in animal farms to ensure its survival and health.
- Animal Tracking: Location and identification of animals grazing in open pastures or location in big stables.
- Toxic Gas Levels: Study of ventilation and air quality in farms and detection of harmful gases from excrements.

For domestic and home automation:

- Energy and Water Use: Energy and water supply consumption monitoring to obtain advice on how to save cost and resources.
- Remote Control Appliances: Switching on and off remotely appliances to avoid accidents and save energy.
- Intrusion Detection Systems: Detection of windows and doors openings and violations to prevent intruders.
- Art and Goods Preservation: Monitoring of conditions inside museums and art warehouses.

And finally, in what is becoming most important for humans nowadays, healthcare.

- Fall Detection: Assistance for elderly or disabled people living independent.
- Medical Fridges: Control of conditions inside freezers storing vaccines, medicines and organic elements.
- Sportsmen Care: Vital signs monitoring in high performance centres and fields.
- Patients Surveillance: Monitoring of conditions of patients inside hospitals and in old people's home.
- Ultraviolet Radiation: Measurement of UV sun rays to warn people not to be exposed in certain hours.

3.1.2 *Waspmote* characteristics

A list of some of the many possible applications areas was provided in the previous section. From this moment on, our goal will be to overview the main specifications of the hardware and some different sensors.

General characteristics of *Waspmote*

- Microcontroller: ATmega1281.
- Frequency: 8MHz.
- SRAM: 8KB.
- EEPROM: 4KB.
- FLASH: 128KB.
- SD Card: 2GB.
- Weight: 20gr.
- Dimensions: 73.5 x 51 x 13 mm.

- Temperature range: [-20ºC, +65ºC].
- Clock: RTC (32 KHz).

Consumption

- On: 9 mA.
- Sleep: 62 uA.
- Deep Sleep: 62 uA.
- Hibernate: 0.7 uA.

Operation:

- 1 year using Hibernate as saving energy mode.

Input / Output

- 7 Analog, 8 Digital (I / O), 1 PWM, 2 UARTs, 1 I2C, 1USB.

Electrical characteristics

- Battery voltage: 3.3 V - 4.2V.
- USB charging: 5 V - 100mA.
- Solar panel load: 6 - 12 V - 240mA.
- Auxiliary battery voltage: 3V.

Sensors embedded on board

- Temperature (+/-): -40ºC, +85ºC. Accuracy: 0.25ºC.
- Accelerometer: ±2g (1024 LSb/g) / ±6g (340LSb/g). 40Hz/160Hz/640Hz/ 2560Hz.

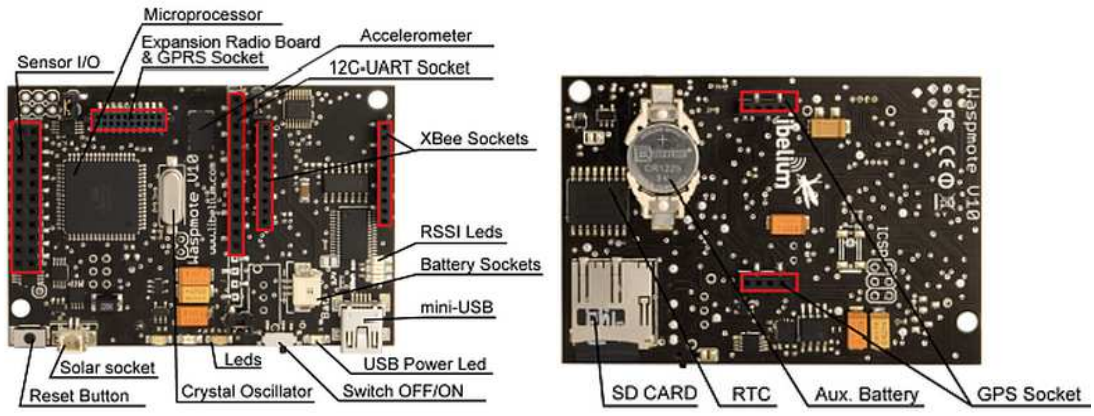

Figure 19.*Waspmote* sensor hardware

Waspmote supports ZigBee with different protocols, frequencies, transmission power, sensitivity and ranges, depending on the purpose of our application. It supports Wi-Fi with secure web connections, file transfers and connections with iPhone and Android. It connects with any commercial Wi-Fi router as well. Bluetooth is a standard contained as well as GSM/GPRS are. Bluetooth radio for device discovery is another of the features and lastly the RFID, really important for business such as retail.

3.1.3 *Waspmote* sensor example

Libelium offers a wide variety of sensors packs differing on the final purpose such as agriculture and health areas. Just one will be reviewed as there are too many and it is not the main goal here to spend a lot of time on the specifics of each one. The one proposed for Smart Cities, will be explained below and can be appreciated in figure 20.

Figure 20.Hardware used for Cities' applications

Applications

- Noise maps: monitor in real time the acoustic levels in the streets of a city.
- Structural health monitoring: crack detection and propagation.
- Air quality: Detect the level of particulates and dust in the air.
- Waste management: Measure the garbage levels in bins to optimize the trash collection routes.

Sensors

- Microphone (dBSPLA).

- Crack detection gauge.

- Crack propagation gauge.

- Dust - PM-10.

- Ultrasound (distance measurement).

- Temperature.

- Humidity.

- Luminosity.

3.2 Traffic monitoring with *Waspmote*

The Vehicle Traffic Monitoring Platform from Libelium allows system integrators to create real time systems for monitoring vehicular and pedestrian traffic in cities by using the Bluetooth-ZigBee double radio feature available in the *Waspmote* sensor board. The platform is capable of sensing the flow of Bluetooth devices in a given street, roadway or passageway differentiating hands-free car kits from pedestrian phones. Sensor data is then transferred by a multi-hop ZigBee radio, via an internet gateway, to a server. The traffic measurements can then be analysed to address congestion of either vehicle or pedestrian traffic.

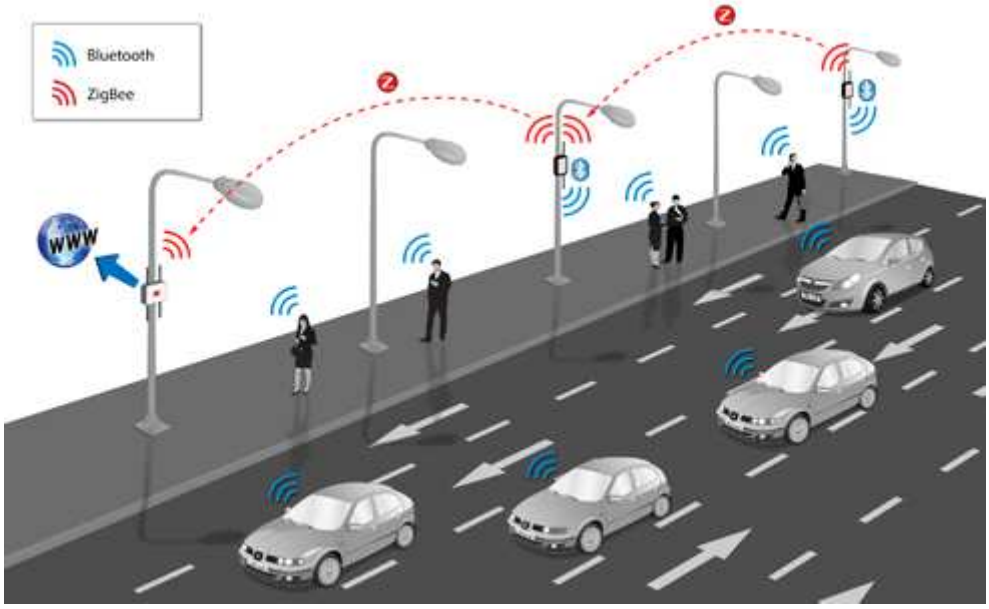

Figure 21.One of the possible connectivity scenarios

Understanding the flow and congestion of vehicular traffic is essential for efficient road systems in cities. Smooth vehicle flows reduce journey times, reduce emissions and save energy. Similarly the efficient flow of pedestrians in an airport, stadium or shopping centre saves time and can make the difference between a good and a bad visit. Monitoring traffic (whether road vehicles or people) is useful for operators of roads, attractions and transport hubs.

The monitoring system can also be used to calculate the average speed of the vehicles which transit over a roadway by taking the time mark at two different points.

Libelium's Vehicle Traffic Monitoring Platform enables system integrators to create intelligent monitoring systems for the urban environment. With widespread use of Bluetooth

devices both vehicular and pedestrian traffic can be monitored anonymously by detecting and tracking the MAC addresses of such devices. The platform can help drivers avoid congested roads through provision of real time warnings on electronic displays or via Smartphone applications. Similarly, pedestrian monitoring enables improvements to be made in the operation of airports, shopping centres, tourist attractions and sports stadiums. Such data can even be used to assess the suitability of emergency evacuation plans.

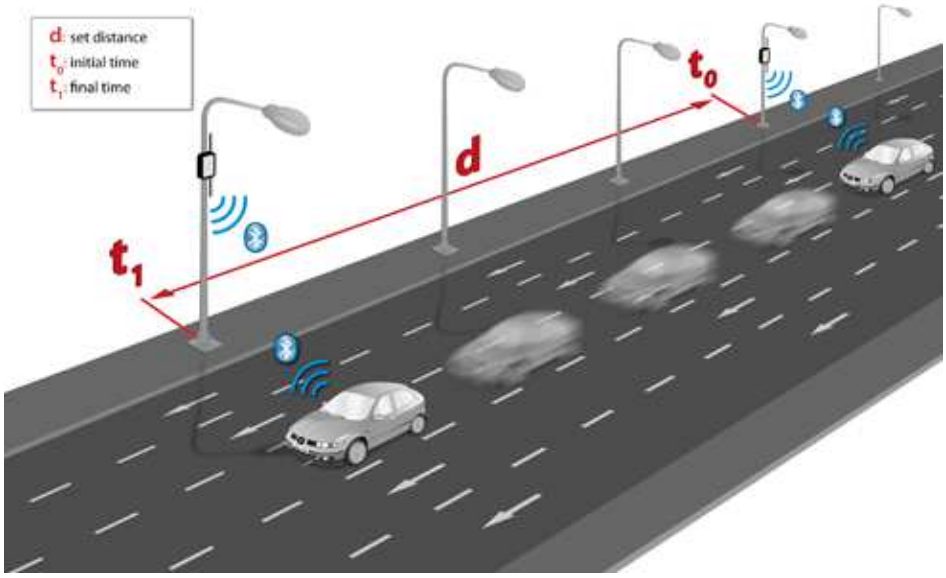

Figure 22.A way of measuring distances

The Platform uses the Expansion Radio Board for *Waspmote* which allows two different types of radio to be connected at the same time. In this case a Bluetooth radio is used as a sensor to make inquiries and detect nearby devices, while the ZigBee radio sends the information collected using its multi-hop capabilities. The new Bluetooth radio allows the possibility to scan up to 250 devices in a single inquiry and set six different power levels allowing sensor operators to set an "inquiry zone" from between 10 and 50 metres.

Figure 23.Expansion Radio Board

Although Bluetooth, ZigBee and Wi-Fi all operate in the 2.4 GHz ISM band, *Waspmote* uses Adaptive Frequency Hopping (AFH) to enable the Bluetooth radio to identify channels already in use by ZigBee and Wi-Fi devices and thus avoid interference. The way the sensor works will be explained in order to have a better understanding.

The way of controlling the inquiry area: There are seven different power levels which go from -27dBm to +3dBm in order to set different inquiry zones from 10 to 50m. These zones can also be increased or decreased by using a different antenna for the module as it counts with a standard SMA connector. The default antenna which comes with the module has a gain of 2dBi.

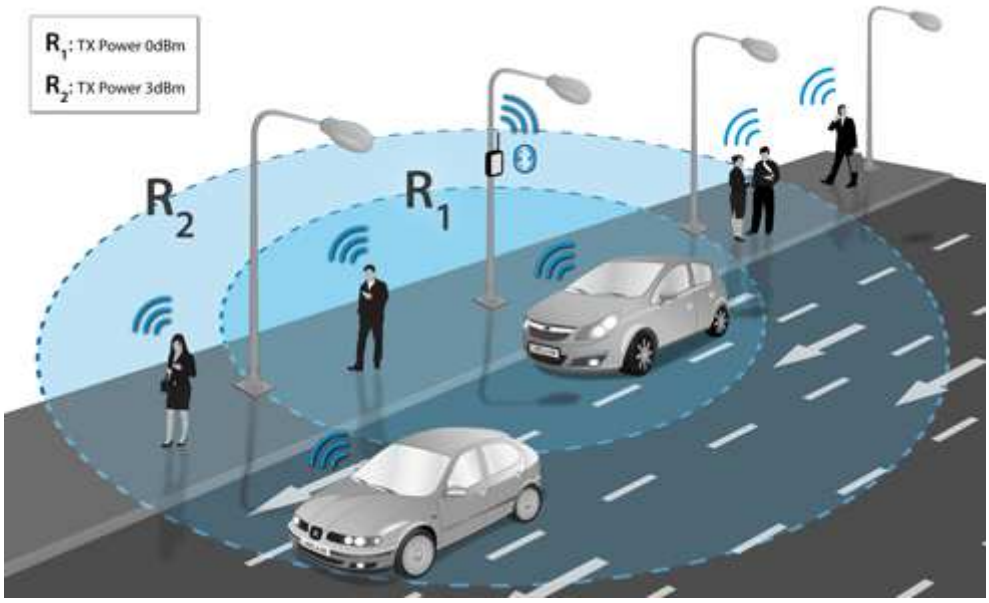

Figure 24.Possible scenario of area controlling

To calculate the distance of any of the devices detected the following is done: In the inquiry process we receive the MAC address of the Bluetooth device, its CoD and the Received Signal Strength Indicator (RSSI) which gives us the quality of the transmission with each device. RSSI values usually go from -40dBm (nearest nodes) to -90dBm (farthest ones). In the tests performed Bluetooth devices at a distance of 10m reported -50dBm as average, while the ones situated at 50m gave us an average of -75dBm.

ZigBee and Bluetooth work in the 2.4GHz frequency band (2.400 - 2.480MHz), however, the Bluetooth radio integrated in *Waspmote* uses an algorithm called Adaptive Frequency Hopping (AFH) which improves the common algorithm used by Bluetooth (FHSS) and enables the Bluetooth radio to dynamically identify channels already in use by ZigBee and Wi-Fi devices and to avoid them so interferences do not happen. In the figure below we can see how AFH avoids transmitting in frequency bands which are being used by DSSS technologies such as ZigBee and Wi-Fi networks, allowing a complete coexistence between all them as shows figure 25.

The anonymous nature of this technique is due to the use of MAC addresses as identifiers. MAC addresses are not associated with any specific user account or mobile phone number not

even to any specific vehicle. Additionally, the "inquiry mode" (visibility) can be turned off so people have always chosen if their device will or will not be detectable.

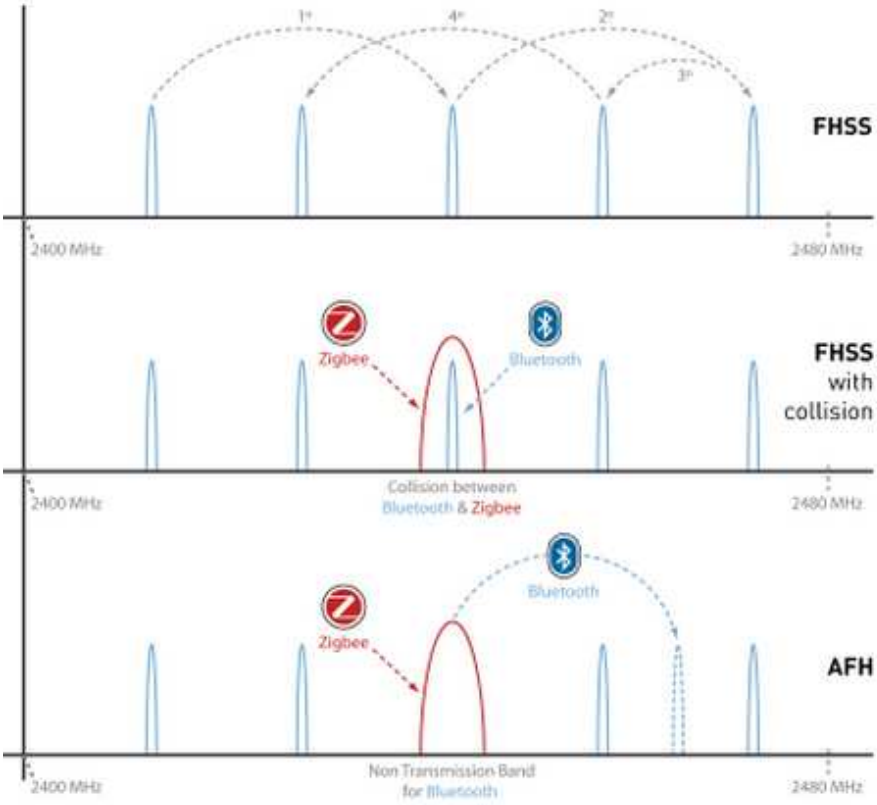

Figure 25.Avoiding Bluetooth and Zigbee collision

3.3 Smart parking with *Waspmote*

Another application for a Smart City is the Smart Parking Sensor Platform which allows city motorists save time and fuel. For this end, it will be necessary to use Libelium's Smart Parking sensor, in figure 26, designed to be buried in parking spaces and to detect the arrival and departure of vehicles. The Smart Parking platform will allow system integrators to offer comprehensive parking management solutions to city councils. By providing accurate information on available parking spaces, motorists save time and fuel and cities reduce atmospheric pollution and congestion.

The quality of city life across the world is negatively impacted by atmospheric pollution and congested roads. Road congestion results in lost time for motorists, wasted fuel and is a major cause of air pollution. A significant contribution to congestion arises from motorists searching for available parking spaces (often requiring a considerable time before they are successful) and is a major cause of driver frustration. Providing accurate information to drivers on where to find available parking spaces helps traffic flow better and allows the deployment of applications to book parking spaces directly from the car.

Systems based on Libelium's new Smart Parking sensor platform enable drivers to find free parking spaces quickly and efficiently. Efficient parking not only means happier motorists,

but also reduces CO2 emissions, saves fuel and helps minimise traffic jams. Smart Parking sensors can be buried in parking spaces and communicate with the rest of the sensor network using *Waspmote*'s ZigBee radio.

Figure 26.Smart Parking sensor

Waspmote's outstanding power management and over the air programming (OTA) mean that, once installed, parking sensors do not need to be accessed for years. Motes only need to transmit when a parking event (a vehicle arriving or leaving a space) takes place. With suitable batteries a sensor can operate for five years before it needs to be physically accessed for battery replacement. OTA programming enables the software for entire networks to be upgraded efficiently over the radio network without digging up the parking spaces. The low maintenance involved in smart parking sensor networks means that networks with hundreds of nodes can readily be deployed.

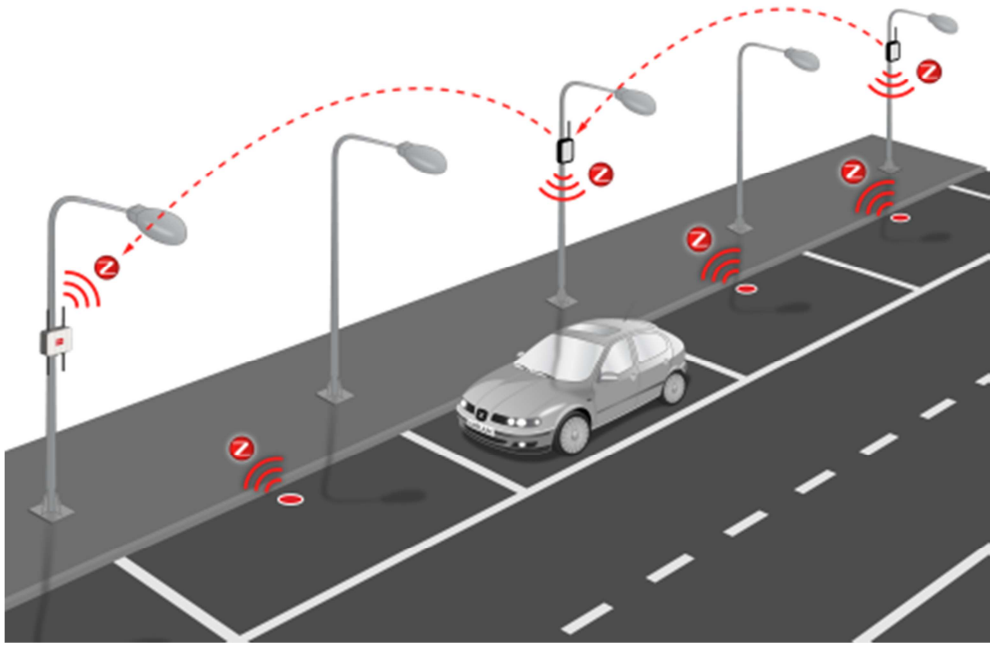

Figure 27.Interconnecting the Smart Parking sensors

Smart parking sensors communicate with their gateway via radios at either 2.4GHz or 868/900MHz. For 2.4 GHz ZigBee connections, mesh networks are implemented with routing motes located in street lights. For the lower frequency radios, it is possible for parking sensors to communicate directly with the gateway as the propagation distance is longer.

Figure 28.Burying the Smart Parking sensor

Parking sensors must be robust enough to be buried under parking spaces. The sensor nodes must be supplied in a PVC casing capable of providing mechanical impact protection and for ingress protection. The use of PVC ensures that radio communication is not hindered.

3.4 Traffic, RFID and binary proximity sensor networks

It has been many decades since urban traffic became a big problem with the rapid increase of vehicle quantity, and it disturbed and still disturbs normal life of urban residents and travellers. Especially traffic jams are a difficult problem to confront as generates great financial losses every year, basically because people are trapped in their cars instead of working. Intelligent traffic control system is proved the most effective approach to resolve this problem. Vehicle surveillance, including detection and classification, that provides real-time traffic data for traffic light control system with the needs to optimize the spatial and temporal allocation of traffic resource. And consequently, the performance of vehicle surveillance is significant to traffic light control, optimal traffic resource allocation and maintenance of the pavement system.

Currently there are many vehicle surveillance technologies including loop sensors, video cameras, image sensors, infrared sensors, microwave radar, GPS, etc. The performance is acceptable but not sufficient because of their limited coverage and expensive costs of implementation and maintenance, specially the last one. They have defects such as: line-of-sight, low exactness, depending on environment and weather, cannot perform no-stop work whether daytime or night, high costs for installation and maintenance, etc. Consequently, in actual traffic applications the received data is insufficient or bad in terms of real-time owed to detector quantity and cost. Thus the actual performance of traffic control system such as SCOOT (Split, Cycle, & Offset Optimization Technique)/SCATS (Sydney Coordinated Adaptive Traffic System) is influenced. With the increase of vehicle in urban road networks, the vehicle detection technologies are confronted with new requirements.

Wireless sensor network is the-state-of-art technology and a revolution in remote information sensing and collection applications. Sensor node has advantages like low costs, small size, wireless communication, high sensing accuracy, and can be deployed with great quantity as they have high configuration flexibility. It has broad prospect of application in intelligent transportation system. This is the case of the before-seen *Waspmote* sensor as it has the possibility of reading RFID tags as well. RFID tags can be used e.g. to identify vehicles and to know where you parked yours if you are in an unknown city. Still, this could lead to security problems and the possibility of having your car identified by hackers. It may also arouse fears in what current society is so afraid of: an excessive government control.

Holiday crowds are a good case in which it is necessary to keep count of the cars on the roads. What I have worked on is in target tracking and counting using a binary sensor network that could be implemented, e.g., in the sensor Libelium has developed. This would allow the appliance to spend much less energy whenever transmitting information and to make more independent devices. Dalian University of Technology published a paper in which this topic is studied. They developed a new algorithm called Magnetic Sensors based Vehicle Classification Algorithm (MSVCA). In this algorithm, magnetic sensors are deployed as Binary Proximity Sensor Networks (BPSN) to detect the magnetic field distortion with a distributed threshold, and estimate the length of vehicle via the geometric characteristics of the topology. Their algorithm was able to enhance vehicle classification with good performance and solid robustness.

The subject of tracking targets using BPSN is thoroughly discussed in the next chapter in order to have greater knowledge of this kind of networks. It will just be simulated though and not implemented in a real life scenario.

4. Tracking targets using a binary proximity sensor network

I investigate the problem of tracking targets using a network of binary proximity sensors. Each sensor produces a single bit of output, which is 1 when one or more targets are in its sensing range and 0 otherwise. These sensors are not able to distinguish individual targets, decide how many distinct targets are in the range, or provide any location-specific information. Despite the minimal information provided by a single binary sensor, I investigate the problem of tracking multiple targets, without *a priori* knowledge of the number of targets.

I have chosen to focus on the simple and minimalistic setting of binary sensors because the cost and power consumption of sensor nodes is a severe constraint in large-scale deployments, and both can be significantly reduced by restricting the nodes to provide binary detection. In fact, by constraining to a binary sensing model, we can work with low-power, low-cost sensor nodes that can form the basis for a highly scalable architecture for wide area surveillance. This information can, of course, be augmented by a small number of more capable sensors (e.g., cameras), although I have not explored such enhancements in this project.

Examples of sensor modalities that are suitable for low-cost nodes are Seismic, Acoustic, Passive infrared (PIR), Active infrared, Ultra wide band radar imaging, Millimetre wave radar, Magnetometer and Ultrasonic. For multiple targets, I have encountered significant additional difficulties than just with one or two targets, since I cannot tell how many targets are within a sensor's range when it outputs a 1.

The first task in this chapter, therefore, is to understand the fundamental of the particle filtering. Next one would be to know how well can be counted the number of targets, given a snapshot of the sensor readings. I employ geometric arguments to characterize when an accurate count is possible, and provide a lower bound on the number of targets, based on a greedy algorithm for explaining the sensor's observations with the minimum number of targets. While these arguments bring out the difficulty of target counting based on a snapshot, they do not preclude the possibility of accurate counting and tracking when we account for the evolution of the sensor readings in time, using a model for the targets' behaviour.

To this end, I develop a particle filtering algorithm which employs a cost function penalizing changes in velocity. It is shown by simulations that the particle filter algorithm is effective in tracking targets even when their trajectories have significant overlap. I restrict attention to one-dimensional systems and this enables to gain fundamental insight, as well as to easily display multiple trajectories on two-dimensional space-time plots.

I will focus on the efficacy of collaborative tracking rather on the communication protocols used by the sensor nodes. Thus, I assume that all of the sensor readings are available at a centralized processor, which can then estimate the targets' locations and trajectories. I note that the binary sensing model has minimal communication requirements, hence this assumption of centralized processing is quite practical: a sensor need only convey the intervals at which it switches “on” and “off” (assuming that the readings are averaged so as to remain reasonably steady, this is far more efficient than sending a sample of the sensor's readings at regular intervals).

In Section 4.2 the fundamentals of particle filtering are seen, in Section 4.3 discusses the problem of target counting based on a snap shot of the sensor readings. In Section 4.4, the target tracking problem is revised. In Section 4.5, it is described the particle filtering algorithm and in Section 4.6 provides simulation results and it ends with the conclusions in Section 4.7.

4.1 Fundamentals of Particle Filtering

Consider a system/signal with a state-space representation given by.

$$\begin{aligned} x_t &= f_t(x_{t-1}, u_t) \\ y_t &= g_t(x_t, , v_t,) \end{aligned} \tag{3}$$

As already pointed out, the main task of sequential signal processing is the estimation of the state x_t recursively from the observations y_t. In general, there are three probability distribution functions of interest, and they are the filtering distribution $p(x_t|x_{0:t})$ the predictive distribution; and the smoothing distribution $p(x_t|y_{0:t})$, where $T > t$. All the information about x_t regarding filtering, prediction, or smoothing is captured by these distributions, respectively, and so the main goal is their tracking, which is obtaining $p(x_t|x_{0:t})$ from $p(x_t|y_{0:t-1})$, $p(x_{t+l}|y_{0:t})$ from $p(x_{t+l-1}|y_{0:t})$, or $p(x_t|y_{0:T})$ from $p(x_{t+1}|y_{0:T})$. The algorithms that exactly track these distributions are known as optimal algorithms. In many practical situations, however, the optimal algorithms are impossible to implement, primarily because the distribution updates require integrations that cannot be performed analytically or summations that are impossible to carry out due to the number of terms in the summations.

For the joint a posteriori distribution of $x_0, x_1, \dots, x_t$, in case of independent noise samples we can write

$$p(x_t|x_{0:t}) \alpha\, p(x_0|x_0) \prod_{k=1}^{t} p(y_k|x_k)\, p(x_k|x_{k-1}) \tag{4}$$

It is straightforward to show that a recursive formula for obtaining $p(x_{0:t}|y_{0:t})$ from $p(x_{0:t-1}|y_{0:t-1})$ is given by

$$p(x_{0:t}|y_{0:t}) = \frac{p(y_t|x_t)\, p(x_t|x_{t-1})}{p(y_t|y_{0:t-1})}\, p(x_{0:t-1}|y_{0:t-1}) \tag{5}$$

Since the transition from $p(x_{0:t-1}|y_{0:t-1})$ to $p(x_{0:t}|y_{0:t})$ is often analytically intractable, I resort to methods that are based on approximations.

In particle filtering, the distributions are approximated by discrete random measures defined by particles and weights assigned to the particles. If the distribution of interest is $p(x)$ and its approximating random measure is

$$\chi = \{x^{(m)}, w^{(m)}\}_{m=1}^{M} \tag{6}$$

where $x(m)$ are the particles, $w(m)$ are their weights, and M is the number of particles used in the approximation, χ approximates the distribution $p(x)$ by

$$p(x) \approx \sum_{m=1}^{M} w^{(m)} \delta(x - x^{(m)}) \tag{7}$$

where δ(-) is the Dirac delta function. With this approximation, computations of expectations (which involve complicated integrations) are simplified to summations, that is, for example,

$$E\big(g(X)\big) = \int g(x)p(x)dx \tag{8}$$

is approximated by

$$E\big(g(X)\big) \approx \sum_{m=1}^{M} w^{(m)} g(x^{(m)}) \tag{9}$$

The next important concept used in particle filtering is the principle of importance sampling. Suppose we want to approximate a distribution $p(x)$ with a discrete random measure. If we can generate the particles from $p(x)$, each of them will be assigned a weight equal to $1/M$. When direct sampling from $p(x)$ is intractable, one can generate particles $x(m)$ from a distribution $\pi(x)$, known also as importance function, and assign (non-normalized) weights according to

$$w^{*(m)} = \frac{p(x)}{\pi(x)} \tag{10}$$

which upon normalization become

$$w^{(m)} = \frac{w^{*(m)}}{\sum_{m=1}^{M} w^{*(m)}} \tag{11}$$

Suppose now that the posterior distribution $p(x_{0:t-1}|y_{0:t-1})$ is approximated by the discrete random measure $\chi_{t-1} = \left\{X_{0:t-1}^{(m)}, W_{t-1}^{(m)}\right\}_{m=1}^{M}$. Note that the trajectories or streams of particles $X_{0:t-1}^{(m)}$ can be considered particles of $p(x_{0:t-1}|y_{0:t-1})$. Given the discrete random measure χ_{t-1} and the observation y_t, the objective is to exploit χ_{t-1} in obtaining χ_t. Sequential importance sampling methods achieve this by generating particles $X_t^{(m)}$ and appending them to $X_{0:t-1}^{(m)}$ to form $X_{0:t}^{(m)}$, and updating the weights $W_t^{(m)}$ so that χ_t allows for accurate estimates of the unknowns of interest at time t. If we use an importance function that can be factored as

$$\pi(x_{0:t}|y_{0:t}) = \pi\, p(x_t|x_{0:t-1}, y_{0:t})\pi\,(x_{0:t-1}|y_{0:t-1}) \tag{12}$$

and if

$$X_{0:t-1}^{(m)} \sim \pi\left(x_{0:t-1} \middle| y_{0:t-1}\right) \tag{13}$$

And

$$W_{t-1}^{(m)} \propto \frac{p(X_{0:t-1}^{(m)} \middle| y_{0:t-1})}{\pi(X_{0:t-1}^{(m)} \middle| y_{0:t-1})} \tag{14}$$

we can augment the trajectory $X_{0:t-1}^{(m)}$ with $X_t^{(m)}$, where

$$X_t^{(m)} \sim \pi\left(x_t \middle| X_{0:t-1}^{(m)}, y_{0:t}\right) \tag{15}$$

and easily associate with it an updated weight $W_t^{(m)}$ obtained according to

$$W_{t-1}^{(m)} \propto \frac{p(y_t \middle| X_t^{(m)}) p(X_t^{(m)} \middle| X_{t-1}^{(m)})}{\pi(X_t^{(m)} \middle| X_{0:t-1}^{(m)}, y_{0:t})} W_t^{(m)} \tag{16}$$

The sequential importance sampling algorithm can thus be implemented by performing the following two steps for every *t*:

1. Draw particles $X_t^{(m)} \sim \pi\left(X_t \middle| X_{0:t-1}^{(m)}, y_{0:t}\right)$ where m=1, 2,…, M.
2. Compute the weights of $W_t^{(m)}$ according to (14).

The importance function plays a very important role in the performance of the particle filter. This function must have the same support as the probability distribution that is being approximated. In general, the closer the importance function to that distribution, the better the approximation is. In the literature, the two most frequently used importance functions are the prior and the optimal importance function. The prior importance function is given by $p(x_t \middle| X_{t-1}^{(m)})$ and it implies particle weight updates by

$$W_t^{(m)} \propto W_{t-1}^{(m)} p(y_t \middle| X_t^{(m)}) \tag{17}$$

The optimal importance function minimizes the variance of the importance weights conditional on the trajectory $X_{t-1}^{(m)}$ and the observations $y_{0:t}$ and is given by $p(x_t \middle| X_{t-1}^{\prime(m)}, y_{0:t})$. When the optimal function is used, the update of the weights is carried out according to

$$W_t^{(m)} \propto W_{t-1}^{(m)} p(y_t \middle| X_t^{(m)}) \tag{18}$$

Note that implementations of particle filters with prior importance functions are much easier than those with optimal importance functions. The reason is that the computation of $p(y_t \middle| X_{t-1}^{(m)})$ requires integration.

A major problem with particle filtering is that the discrete random measure degenerates quickly. In other words, all the particles except for a very few are assigned negligible weights. The degeneracy implies that the performance of the particle filter will deteriorate. Degeneracy, however, can be reduced by using good importance sampling functions and *resampling*.

Resampling is a scheme that eliminates particles with small weights and replicates particles with large weights. In principle, it is implemented as follows:

1. Draw M particles, $X_t^{*(m)}$ from the discrete distribution χ_t .

2. Let $X_t^{(m)} = X_t^{*(m)}$, and assign equal weights (1/ M) to the particles.

The idea of resampling is depicted in Figure 29 with M=10 particles. There, the left column of circles represents particles before resampling, where the diameters of the circles are proportional to the weights of the particles. The right columns of circles are the particles after resampling. In general the large particles are replicated and the small particles are removed. For example, the "blue" particle with the largest weight is replicated three times and the "yellow" particle, two times, whereas the green particles, which have small weights, are removed. Also, after resampling all the circles have equal diameters, that is, all the weights are set to1/ M.

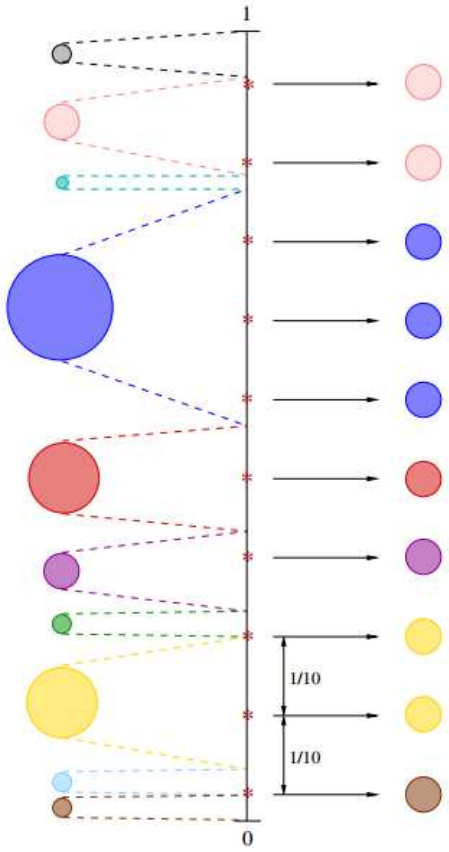

Figure 29.Schematic description of resample

In Figure 30, we represent pictorially the random measures and the actual probability distributions of interest as well as the three steps of particle filtering: particle generation, weight update, and re-sampling. In the figure, the solid curves represent the distributions of interest, which are approximated by the discrete measures. The sizes of the particles reflect the weights that are assigned to them.

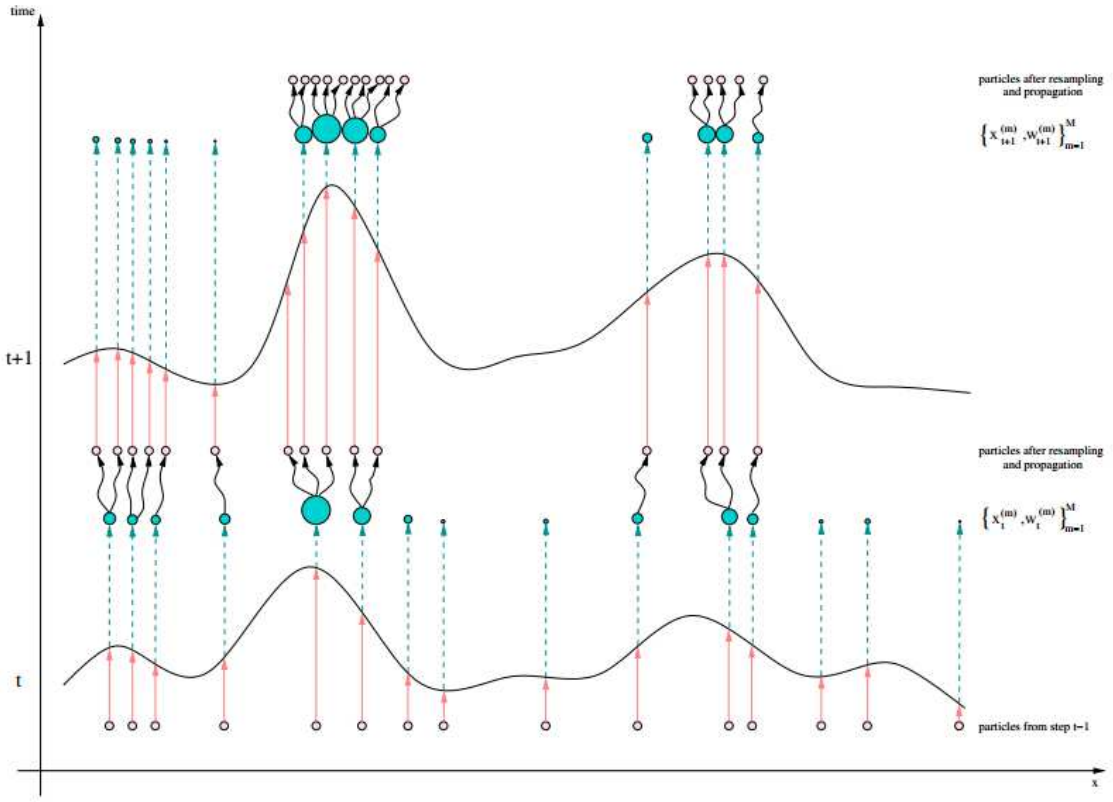

Figure 30.A pictorial description of particle filtering

Finally in Figure 31, we display a flowchart that summarizes the particle filtering algorithm. At time t, a new set of particles is generated and their weights are computed. Thereby we obtain the random measurement χ_t which can be used for estimation of the desired unknowns. Before we proceed with the generation of the set of particles for time instant $t+1$, we estimate the *effective* particle size (a metric that measures the degeneracy of the particles). If the effective particle size is below a predefined threshold, re-sampling takes place; otherwise we proceed with the regular steps of new particle generation and weight computation.

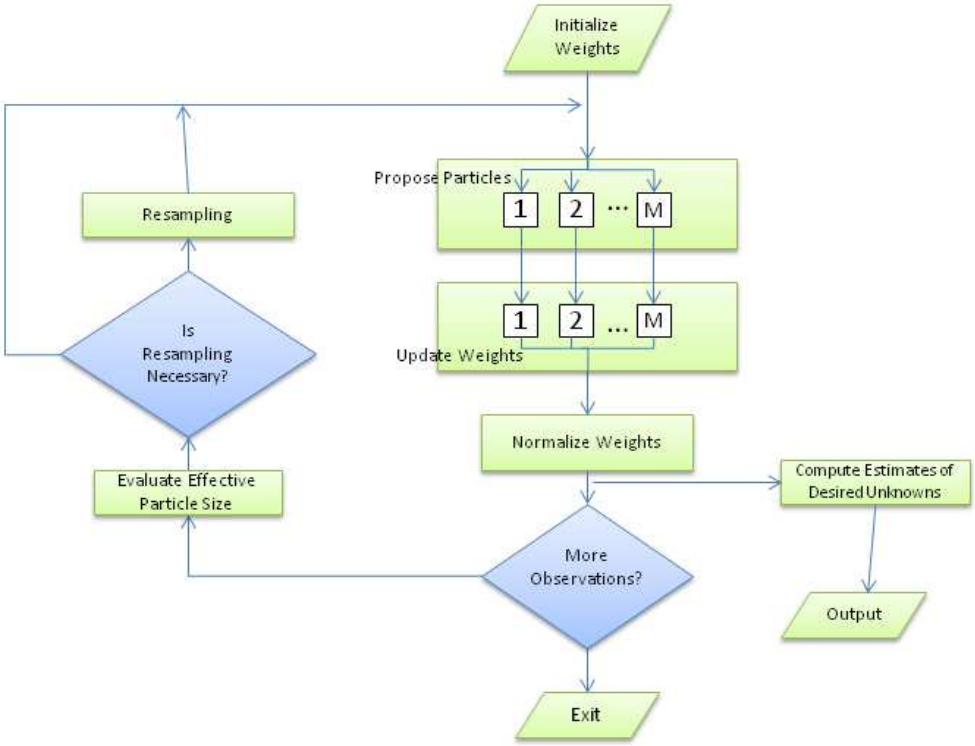

Figure 31.A block diagram of particle filtering

4.2 Target countability and fundamental limit

4.2.1 Easy case

In order to solve the problem of counting targets within a field of binary proximity sensors it is clear that some spatial separation among the targets must hold. When such condition exists then we settle with the so called easy case. While investigating the problem one can distinguish two different types of spatial separation. If for instance each target moves sufficiently far from the remaining targets, we have the first so called individual separation condition. Although, at a first glance it seems sufficient, after gaining more insight on the problem it becomes obvious that this is not the case. As shown and by figure 32 even arbitrarily large individual separation is not sufficient to reliably count a set of moving targets using binary sensors.

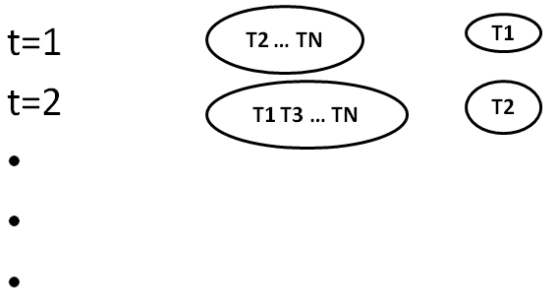

Figure 32

As illustrated by the previous figure, and due to the nature of binary sensors it is clear that the intermixture of targets, under the individual separation condition, will form in the worst case a partition of only two groups thus making the task of distinguishing targets not plausible. On the other hand if the group of targets has pair wise instead of individual separation more than 4R then binary sensing does permit precise counting of targets. Having solved the counting problem when such spatial separation serves as a precondition the question is now reformed as if we can solve the problem of target counting and without the fundamental separation limit. As shown and in the following sections under the right circumstances with no respect to spatial separation *two targets no matter how close can always be disambiguated if two sensors with non-overlapping sensing ranges detect them.*

4.2.2 Geometric preliminaries and fundamental counting resolution

Before stating the conditions under which an accurate count would be feasible and proving their adequacy, the project first cites some basic geometric preliminaries that will serve as building blocks. As mentioned earlier the project focuses on a field covered by binary proximity sensors organized in a one dimensional line. Suppose the total number of sensors is N. Each sensor's range is then an interval of length 2R. Denoting byC_i, the interval that is covered by an individual sensor i, the domain of interest is then the set $\{C_i\}$ which consists the union of all sensor rangers. It has to be stated that for simplicity reasons, there are no gaps in the coverage. Otherwise conditions at which targets will appear and disappeared must be taken in mind. Our modelled system consisting of sensor nodes and targets, changes as time instances pass leading to a vector of binary outputs from the sensors. This concludes to the partition of the sensors into two groups the first of which consists of contiguous 'on' sensors separated by groups of "off" sensors. If I denote the set of sensors whose binary output is 1 and Z the set of sensors whose binary output is 0, then it is easily deduced that all targets must lay in the region specified by F and called the feasible target space: F = $[C_i: i\in I] - [C_j: j\in Z]$, where [ ] notes union

The region F as shown and by the blue regions of figure 33 is a subset of the entire domain of interest and informs us about all the possible locations a target might be. For sensor i, the contributing area to the feasible target space is $g_i = C_i - [C_j: j\in Z]$, where [ ] notes union. It is easily proved that any two connected components of the feasible target space F are separated by at least distance 2R.

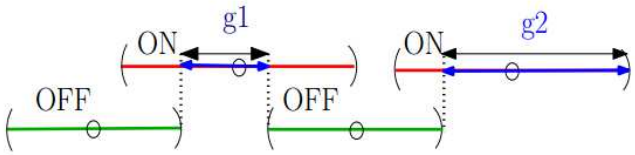

Figure 33

However although for localization purposes the primary interest is in this so called feasible target space, it is really the regions that consist of the off sensors that give us richer information content. From a sensor that outputs '0' one can deduce that there are no targets anywhere in that region in opposition to an 'on' sensor that informs us that there is a least one

target somewhere in the sensors range. It is clear that using more sensors partitions the domain of interest into more regions therefore accomplishing more accurate target tracking.

Establishing a theorem on the fundamental limit of target counting requires one last geometric framework to be cited. Two binary proximity sensors as stated in the paper are said to be positively independent if 1) they both have binary output 1, and 2) either their sensing ranges are disjoint or they belong to different connected components of the feasible target space F. Figure 34 makes these two different cases more clear:

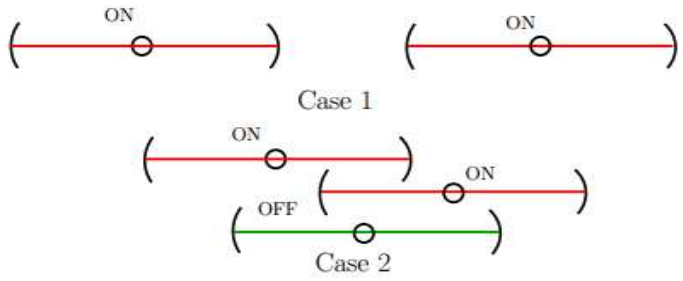

Figure 34

See it in another way, as illustrated by the previous figure, two sensors are positively independent if they are both detecting targets and are either sufficient apart as in case 1 or are separated by an off sensor as in case 2. As an answer to the counting problem and with the necessary geometrical background being beforehand it is proven that a *set of k targets on a line can be counted correctly if and only if there exist k (pair wise) positively independent sensors.*

An aspect that should be commented and is a consequence of the previous remark is that sensor density does affect the counting resolution. In a space of length 2LR, at most L+1 targets can be at best resolved (about one target per distance 2R), due to the fact that each sensor is either 2R away from its neighbour or it is preceded by a sensor with binary output 0, which in turn has a coverage area of 2R. So in order to disambiguate two closely spaced targets one would have to increase density.

4.2.3 Finding the minimum target number consistent to sensor readings

Having provided the necessary conditions via the previously stated theorem it is understandable what must hold in order to count a number of targets within a binary sensor network field. As a supplement to the afore-mentioned observations, it is of interest to ask which is the simplest explanation for a given snapshot of sensor readings. In other words, it is desired to specify what is the minimum number of targets consistent with the sensor readings at any given time t. Interestingly, in the assumption of 1D-Euclidean space *the minimum number of targets matches precisely the maximum number of independent sensor.* In order to prove the preceding theorem, and moreover find the minimum number of targets in respect to sensor readings, one could use a greedy algorithm to specify the maximum possible non-overlapping intervals. Following is the particular algorithm:

1. Let $s_1 \ldots s_m$ be the sensors and $g_1 \ldots$gm the intervals each sensor contributes to F

2. Let S donate the maximum number of positively independent sensors

3. Sort the intervals g_1…gm in the increasing order of their right endpoints.

4. Pick the first interval and add it to S.

5. Delete all overlapping intervals.

6. Pick the next available interval and repeat until no more intervals are left.

Figure 35 illustrates a trivial example of applying the algorithm to a set of five sensors. The maximum possible non-overlapping intervals are provided by the greedy scheme which in the preceding case consists of the set $S = \{g_1, g_4\}$. Figure 36 summarizes the overall correlation between the minimum number of targets, the maximum number of positively independent sensors and the greedy algorithm shown previously.

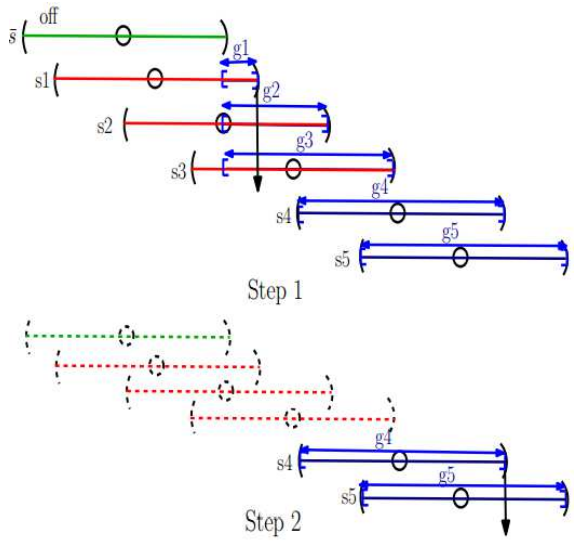

Figure 35

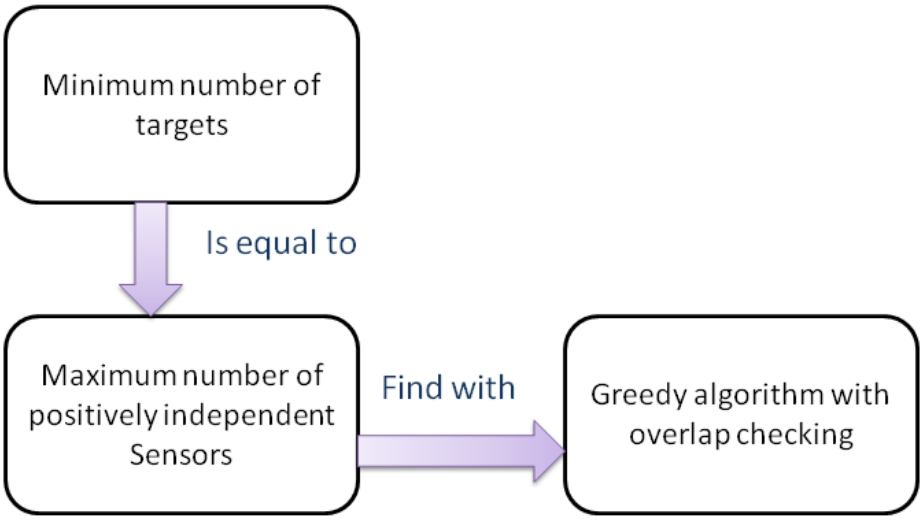

Figure 36

4.2.4 Geometric versus probabilistic techniques

As shown by previously, using the binary sensing model can solve the problem of target counting accurate enough under certain circumstances. This kind of approach to the general problem of target tracking is the best someone could achieve, in a worst case model. This worst case model applies for example when the rate at which the sensors report their readings is low compared the rates at which the targets cross the boundaries of the sensors coverage area. Likewise, when targets move along arbitrary trajectories, thus forming high frequency paths we cannot hope to do better than the previous lower bound. When this more general scenario applies, the conclusions derived from the pre-mentioned geometric foundations are the only way to address the counting problem in the context of binary proximity sensors. However probabilistic, model based, techniques for tracking are more effective in binary sensing model, attempt to estimate only low pass versions of the targets' trajectories. To this end the paper tries to develop a particle filtering algorithm which employs a cost function that penalizes rapid changes in velocity for target tracking using the geometrical framework established for its sampling.

4.3 Addressing the target tracking problem

After studying the geometrical nature of the binary sensing field, in respect to the 1D-Eucledean assumption without loss of generality, and employing theorems that specify the conditions under which the counting problem could be solved, the paper follows by implying a more realistic model for target movement for the purpose of tracking multiple targets. In addition it is considered that sensor readings are available at higher rates than that of the movement of all targets in the field. Towards this end a particle filtering algorithm is developed. However, to get a better understanding of the solution using the particle filtering scheme, prior work on tracking a single object with binary sensors is revised. Thus, the upgrade to tracking multiple objects will be somehow more natural having gained an insight on the limitations and problems that the particle filtering algorithm for one target has.

4.3.1 Tracking a single target

In the context of tracking only one moving target the particle filtering algorithm works as follows. The algorithm begins at t=1 and proceeds step by step to t=T. At every time instant, a large number of K candidate trajectories are hold. These candidate trajectories are called particles and each one of them at any intermediate time t is a candidate for the trajectory till time t. Let the k^{th} particle at time t be denoted as $P_k^t = \{x_k\ [t]: t \in \{1,\ldots, T\}\ \}$ with 1<= k <= K.

As an example a particle at time instant t=3 is a vector that contains three position estimates. The following notation is used:

At t=3 the 20^{th} particle $P_{20}^3 = \{x_{20}[0], x_{20}[1], x_{20}[2]\}$

The first two time instants of the algorithm belong to the initialization phase and are used as a starting point for the algorithm. Thus for t=1, K points are picked uniformly over the feasible

target space $f[1]$. Each of the particles from the previous step is extended to t=2 by picking a point randomly (in a uniform manner) from the set $f[2]$, hence generating the set $\{P_k^2\}$.

After finishing the initialization phase the algorithm proceeds to its main procedure. Given K particles at time t >2 the K particles at the time t+1 are obtained as follows:

Initially every P_k^t particle is extended to time t+1 by choosing m>1 candidates for x_k[t+1] using uniform sampling over the feasible set$f[t+1]$. This produces a total of m*k particles each of length (t+1). So, having a total of m*k particles for the time instant t+1, the best (lowest) K particles are picked, based on a cost function that will be specified in a later on. These selected particles are designated as the K surviving particles for time t+1. The procedure is repeated until T. The algorithm finishes at time instant t=T and from all the K surviving particles the one with the lower cost is selected and designated as the trajectory estimation. Figure 37 illustrates the initialization phase of the previously proposed algorithm where for simplicity matters a 2D-Eukledean space is considered with k=3.

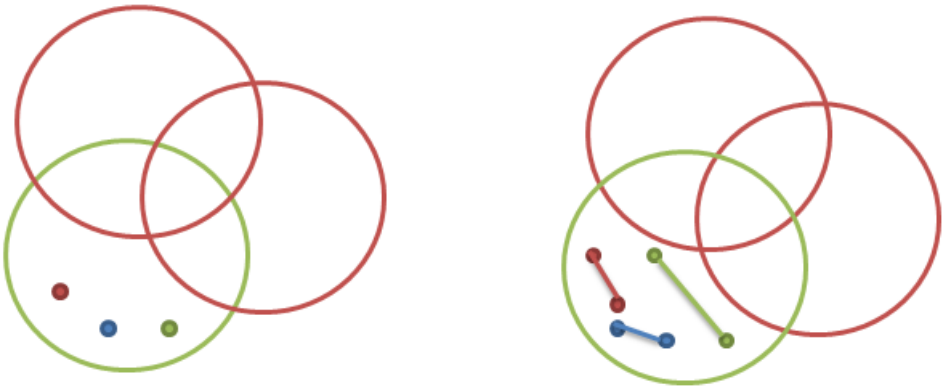

Figure 37.Initialization Step 1(left) and Initialization Step 2(right)

It is clear that the effectiveness of the previously explained algorithm relies mainly on the cost function that will be chosen. There are two characteristics any cost function must comply with. Firstly it must be general enough so as to be able to estimate a variety of target movements. Secondly it must be picked in accordance to an anticipated model of target motion. Then particles that do not conform to this anticipated motion will eventually drop out due to large cost functions, leaving particles that would be good estimates of the true trajectory. With respect to these previously stated observations and in addition keeping with the notion that because of the nature of binary sensing high frequency variations in a trajectory cannot be captured, a cost function that will be an additive metric that will penalize changes in velocity is chosen.

To become more specific let $P(t) = (x[1], \dots x[T])$ denote a particle. Assuming that the rate at which sensors readings are reported is high enough, then the instantaneous estimate of the particles velocity at time t is simply the increment in position x[t+1] - x[t]. Likewise the corresponding increment in the cost function for a target in going from t to t+1 is taken to be the norm of change of the velocity, therefore:

$$c(t) = \big|\big|(x[t+1] - x[t]) - (x[t] - x[t-1])\big|\big| = \big|\big|x[t+1] + x[t-1] - 2 * x[t]\big|\big| \qquad (19)$$

By studying the previously cited cost function it is deduced that the cost is inversely related to the probability that a target moves from the location x[t] at time t, to x [t+1] at time (t+1) given that it had moved from x[t-1] to x[t] from t-1 to t. The net cost function associated a particle P is simply the sum of the incremental costs.

4.3.2 Observations and the naïve approach to multiple target tracking

When the above particle filtering algorithm finishes its execution the best out of the K remaining particles will be picked as an estimate for the target's trajectory. It is also true that a large fraction of the K particles will not differ appreciably from each other thus forming a cluster of so called 'good' particles around the 'best' one. This observation leads to the motivation of looking for clusters of particles instead of individual ones at the end of the algorithm. Using this different approach the pre stated algorithm will hopefully accomplish the task of distinguishing and tracking multiple targets. However, despite initial belief, this naive approach taken for tracking multiple targets by simple looking for clusters does not actually work very well. Unfortunately this intuition does not quite serve the purpose of solving the tracking problem.

First of all, the naive scheme fails to distinguish between targets whose trajectories have significant overlap since the classification and detection of the corresponding clusters of particles would not be possible. Clusters would not be very distinct. In addition, even when all trajectories are distinguishable the naive approach could still miss some targets. As an example suppose there were four targets with very distinguishable movements. Due to how the particle filtering algorithm works by retaining at every time instant the best K particles, there might be a case where if one of the targets moves sufficiently smoother than the others it would gather all the particles. Smooth movement corresponds to small cost functions. It is obvious that the specific target will monopolize the particles which will lock on to it. It is clear that a different approach must be taken from the naive in order to achieve good tracking performance in the context of multiple targets.

4.4 Tracking multiple targets using Cluster Track

As stated and in the previous paragraph although the particle filtering algorithm for one target performs tracking accurate enough, upgrading it to a naive approach where the search of clusters is of primary concern in order to specify distinct trajectories, does not work at a satisfactory level. The most significant reason for that incapability is the fact that most of the cases a single target turns out to monopolize all the surviving particles. So in order to bypass this particular problem the goal would be to prevent a single target from monopolizing all the available particles. Towards this end, the idea realized was that instead of looking for clusters at the end of the algorithm, monitoring their formation throughout the procedure, and limiting the number of particles per cluster would prove sufficient and solve the problem of distinguishing distinct targets thus accomplishing an accurate and low error overall tracking. Upon extending the particle filtering algorithm proposed for only one target in the binary sensing field, the K best particles will still be picked as normally. The fundamental difference that will not permit the monopolizing problem to be encountered is that the task of picking the K best particles will be subject to the constraint that the number of particles per cluster does

not exceed a threshold H, increasing the likelihood of the new particle filtering algorithm to compute successfully all of the targets trajectories.

At the end in order for the particle filter to return the estimated trajectories, it could either consider the best particle of each cluster or chose a "consensus path" (e.g. based on a median filter at each time instant) for each cluster. Moreover, in order to ensure that at every iteration the algorithm does not end up scanning the whole sequence, an upper limit L could be specified. So with the last little augmentation the particle filter stops when: 1) K particles have been found or 2) k>L particles have been scanned. To prune the m*k particles and retain the K particles at each time instant, all of the surviving particles are sorted in increasing order with respect to their cost function. A particle is then retained only if the number of similar particles (same cluster) retained thus far is less than H. The similarity property which is a distance metric and a function of time is to be specified. The following algorithm outlines the new extended particle filtering algorithm used for tracking multiple objects in a pseudo code description of Cluster Track (F) at time (t).

1: Get the set $\{P_k^{t-1}\}$ of K_{t-1} surviving particles from time $t-1$.
2: Extend this set to time t, generating a total of $m_t K_{t-1}$ particles.
3: Sort the $m_t K_{t-1}$ particles in ascending order of cost to get the set $\{\widehat{\boldsymbol{P}}_1, \ldots, \widehat{\boldsymbol{P}}_{m_t k_{t-1}}\}$
4: Put $\widehat{\boldsymbol{P}}_1$ in $Cluster_1, \boldsymbol{P}_1^t = \widehat{\boldsymbol{P}}_1$, Nc = 1, $Count_1 = 1, i = 2, k = 1$
5: **while** $(\boldsymbol{i \le L \; and \; k \le K})$ **do**
6: **if** ($\widehat{\boldsymbol{P}}_i \in Cluster_j$ for some j **then**
7: **if** $Count_j < H$ **then**
8: $Count_j \leftarrow Count_j + 1, \; k \leftarrow k + 1$
9: Retain $\widehat{\boldsymbol{P}}_i$ and $P_k^t = \widehat{\boldsymbol{P}}_i$
10: **else**
11: Abandon $\widehat{\boldsymbol{P}}_i$
12: **end if**
13: **else**
14: Make new cluster for $\widehat{\boldsymbol{P}}_i$, $N_c \leftarrow N_c + 1, k \leftarrow k + 1$
15: $P_k^t = \widehat{\boldsymbol{P}}_i$
16: **end if**
17: $i \leftarrow i + 1$
18: **end while**

The previous pseudo code describes the Cluster Track algorithm. In order to clarify the above outlined code, some points of the algorithm are described in the following section in detail. To begin with, the semantics of the variables used are stated. $Cluster_j$ represents the j^{th} cluster and is a sequence containing particles, $count_j$ is the number of particles that participate in the j^{th} cluster, N_c denotes the current number of formed clusters, H is the maximum number of particles to be retained from a particular cluster and lastly L corresponds to the maximum number of particles to be inspected at each iteration in order to find the surviving particles at time t.

As for the step 2 of the algorithm, where the generation of samples from the feasible target space $f(t)$ is done, the following policy is considered. Initially suppose that $F(t)$ consists of N_t disjoint intervals. As mentioned and in previous chapters, when the geometrical features of the binary sensing field where exploited, these disjoint intervals are obtained by repeatedly

merging overlapping subintervals. Then m_0 samples are picked randomly, in uniform manner, for each of these N_t disjoint intervals thus generating a total of $m(t) = m_0 * N_t$ samples. This must be done for each of the $k(t-1)$ surviving particles.

In addition, the decision in step 6, whether a particle P under consideration belongs to any of the existing clusters is made as follows. Every cluster is assigned to a so called cluster head which is denoted as CH_j. This assignment is done during the creation of a cluster. The particle under consideration that was responsible for the birth and was the first particle to join the specific cluster is the cluster head. It has to be considered that for any time instant t, both P_i and CH_j are vectors of same length t. If the distance between them is defined as D (P_i, CH_j) = Σ | P_i [t] – CH_j[t] |, which is the sum of the absolute difference of all of their components, then, after computing D (P_i, CH_j) for every j and keeping the minimum value, the particle i is said to belong to cluster j if and only if that particular distance D is smaller than a threshold D_o(t), known before hand.

Simulation results show that Cluster Track algorithm performance depends on the choice of the threshold sequence D_o(t). It must be noted that the threshold is proportional to the length t of the particles being compared because of its additive nature.

4.5 Simulation results

In order to prove the correctness of the Cluster Track, several simulations were run and lot of different scenarios was simulated. Before stating the values that were used for the different parameters of the algorithm and how its performance is related to these specific parameter-value associations, first the layout of the binary sensor network as well as the general sensing capabilities of the sensor modules are outlined. In consideration to the initial 1D-Eucledean space assumption, where the previous geometrical framework was established, that the field of concern is reduced to a straight line, an ideal model of binary sensing is applied throughout the simulation experiments. In accordance each sensor has a fixed range, and the possibility of detection of targets which fall within their range is one, hence sensing without any misses or false alarms. The sensing area is therefore normal forming a disk with centre the location of the sensor and radius R. This of course is not the case that holds in real time situation experiments where these kinds of error types are inevitable. If a more realistic setting is to be used then considering modelling the binary sensor modalities as exposed by figure 38 is far more appropriate.

While target detection within the nominal range specified by Re approximates a probability detection of one, therefore almost always detecting correctly, if a target falls in the area of the ring which is the case when R_i <x<R then the detection probability decreases dramatically hence producing missed detections. In addition, and as illustrated by the above picture, because of the non-zero detecting probability for distances bigger than R, a target clearly outside the range of detection of the sensing modality can also fire the sensor and produce false alarms.

In addition to the network and sensor modelling used as described previously, a varying number of targets were employed in different simulation runs for tracking purposes. Although the robustness of the algorithm does not depend on the type of movement (fixed velocity or

with acceleration) if it relates to the pre-mentioned anticipated motion model, throughout all of the following experiments the first case, where the velocity remains unchanged, is applied.

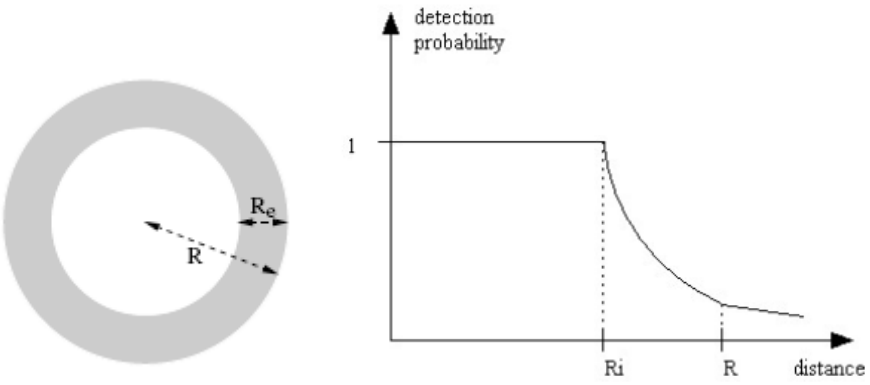

Figure 38

The task of simulating the cluster track algorithm and modelling all the needed components, such as sensors and targets, was written in Matlab language. The data visualization part of the simulation runs was made using the same program, MATLAB, making use of its powerful scripting interface. In order to evaluate the performance of the proposed cluster track algorithm under different circumstances, we'll see some simulation runs with the most significant aspects of each one being discussed and commented.

For the following simulation runs unless other values are used, the following parameter values hold. A one dimensional system as pre-said is considered, with 30 sensors placed uniformly along a straight line on the X axis starting from position 1. Every binary sensor is separated by a distance of 60 units from its neighbour, hence forming a total surveillance domain of 1820 units, and the ideal sensing range what a radius of 40 units is considered, thus implying an overlap of 20 units. Moreover, in the simulation runs, the total number of particles used for the purpose of target tracking was K = 500 while the number of new samples obtained at every time instant for each of the disjoint intervals formed by the sensor readings was and as in the paper m_0 = 12. The threshold sequence was taken to be D_o(t) = 60*t, which is within the range the paper proposes. It must be indicated that throughout the whole simulation process the total number of targets was fixed. In other words cases where missed detections hold where not taken into account. In all the graphs, the real trajectories will be painted in red colour and the estimated ones in blue.

The first four simulation runs concern two overlapping targets. The algorithm is executed once, changing one parameter in order to observe the circumstances under which appreciable tracking is achieved. The information and initial state for the two specific targets, that will put under test the particle filtering algorithm proposed, for the next four simulations is as follows.

Id: 1 pos: 500 velocity: 5 acceleration: 0

Id: 2 pos: 670 velocity: -10 acceleration: 0

Simulation Run 1:

In the first simulation we can see that it works quite well (Cluster Track) and that it follows all the time the targets. Here we can see the values and after the graph:

{508 513 518 521 524 528 531 534 537 541 546 551 557
563 567 572 577 582 587 592 598 604 610 615 620
625 630 634 638 641} Cos → 14

{668 657 647 638 629 620 611 603 596 585 574 564 555
543 531 520 510 500 489 478 468 458 448 438 427
415 405 393 381 370} Cos → 24

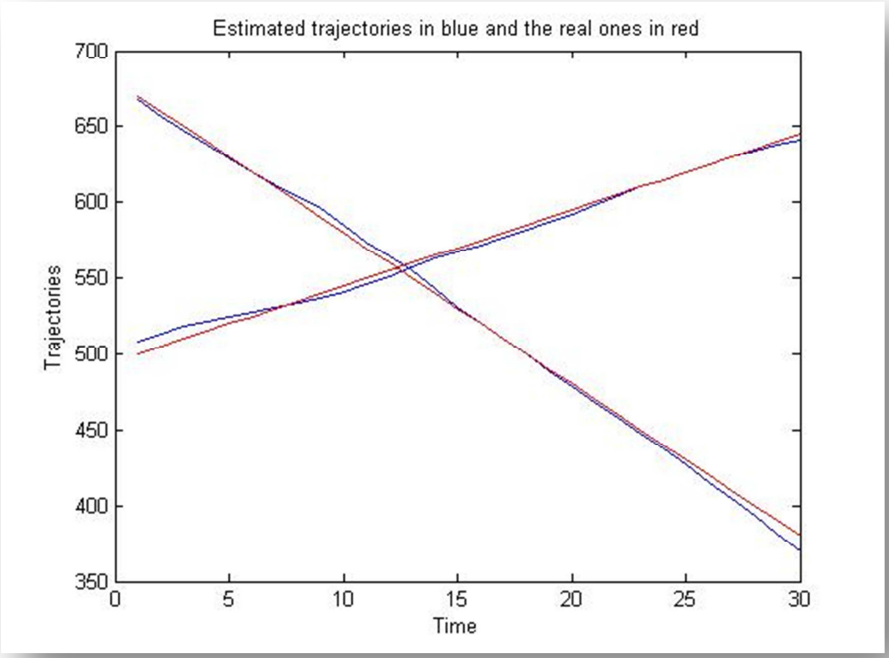

Figure 39

Simulation Run 2:

In this specific simulation the parameter that was changed was the number of total particles. It was set to 100 in opposition to the paper's recommendation of 500 in addition to leaving the parameter of how many particles can belong to a particular cluster unchanged. The algorithm estimated the two trajectories in figure 40.

It is obvious that cluster track performed quite well estimating two trajectories that where good approximates of the true paths despite the limitation of the total number of particles. However when considering more complex problems where more targets are to be tracked and their trajectories have significant overlap, limiting the total number of particles even more might conclude to a poor tracking estimation due to the fact that eventually will die if the cost isn't small enough. It is thus deduced that the parameter K must be much bigger than H.

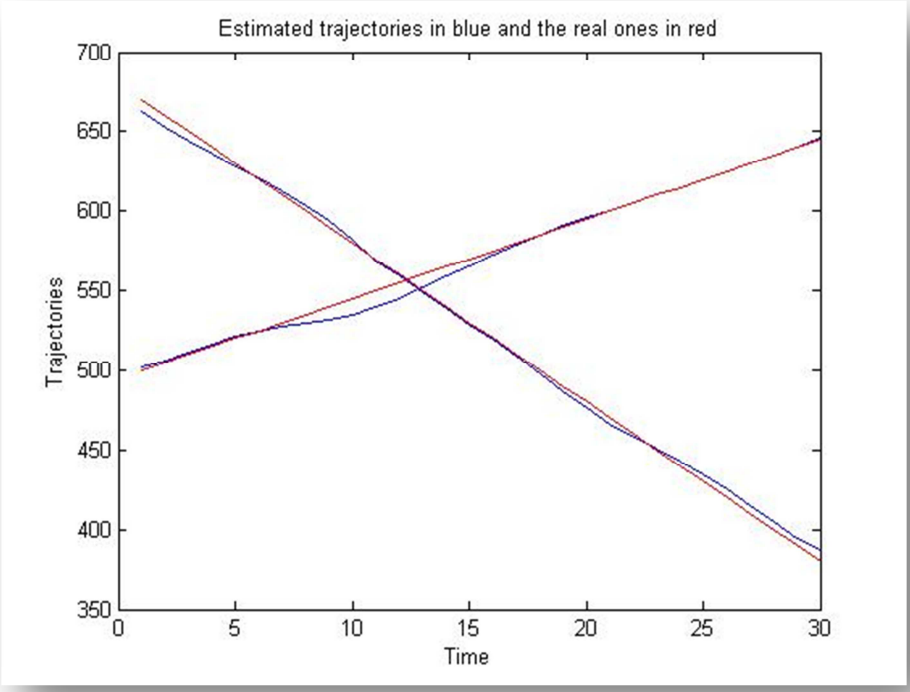

Figure 40

Simulation Run 3:

For this simulation run the parameter that was changed in order to see the response of the algorithm was the threshold sequence Do (t). It was set to 100 which is two and a half times more than the sensors' sensing range radius. The following estimates where produced:

{501 505 508 511 516 522 529 536 543 548 553 558 563
569 572 575 577 582 586 590 596 603 609 615 620
625 629 634 639 644} Cost → 23

It is obvious that the tracking performance is low as only one of the two targets was estimated as intuited. As the paper suggested threshold sequences that are bigger than two times the sensing range radius is not suggested when targets have overlapped paths.

Simulation Run 4:

In addition to the previous example, this simulation again changes the threshold sequence employed. In opposition to the previous simulation run though, instead of increasing the threshold it is decreased and set to 20, which is two times lower compared to the sensing range radius. The following trajectory estimates where produced:

{494 502 510 516 522 530 536 541 546 549 554 559 563
567 571 576 580 585 590 595 600 604 610 615 620
626 632 638 644 651} Cost → 21

{684 670 656 642 628 617 609 601 593 584 576 568 559
548 536 525 514 502 489 478 468 458 449 439 429
419 410 401 392 384} Cost → 22

{493 504 515 525 609 629 615 602 590 578 567 555 545
537 529 520 511 501 490 479 467 457 447 438 429
420 410 400 390 380} Cost→ 189

{663 655 646 639 631 543 537 534 534 536 539 544 549
554 563 571 578 585 590 595 600 606 612 619 626
629 632 636 641 646} Cost→ 193

As expected the algorithm again did not perform very well. Instead of producing the actual two trajectories, it estimated four. Since the threshold was lowered it is expected for more clusters to arise. But this might not always be the case since the algorithm in an extreme situation might approach the naïve scheme therefore generating one big cluster.

Simulation Run 5:

Having exploited the case where two targets overlap, it is time now to check a more realistic tracking case where overlapping occurs for a longer period of time. Towards this end, a more complicated tracking situation is considered where the algorithm tries to track three overlapping targets instead of two in the field of the binary sensing modalities. The targets' information and initial states are as followed:

Id: 1 pos: 500 velocity: 5 acceleration: 0

Id: 2 pos: 670 velocity: -10 acceleration: 0

Id: 3 pos: 550 velocity: 10 acceleration: 0

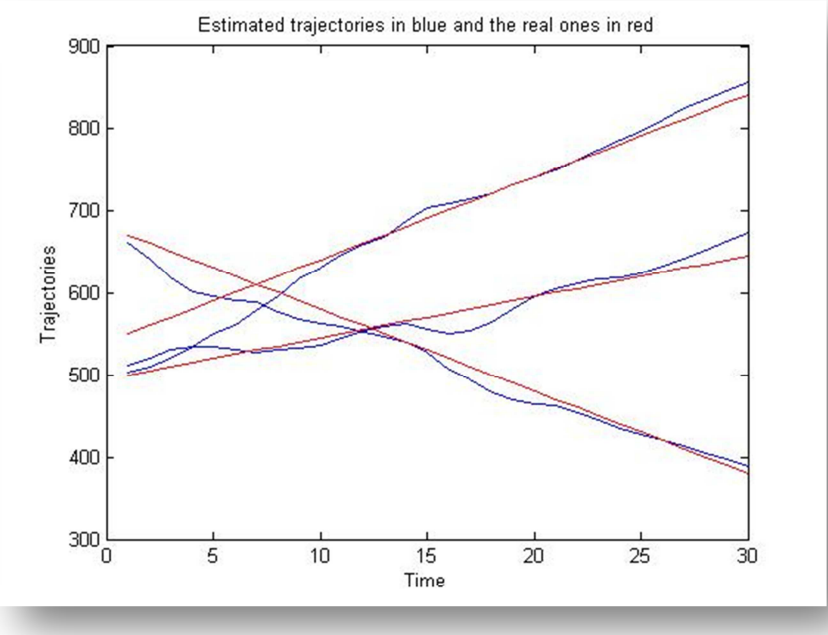

Figure 41

As we can see in the figure 41, although in the beginning they're not so close to their real paths they do converge to the actual trajectories. This holds because, although there are 3 distinct targets the fact that they are not separated enough initially generates a feasible target

space at which they mix their trajectories. Even though overlapping was occurred for a large portion of the execution time, the true trajectories actually become distinct quite soon, approximating them with estimates produced by the particle filtering algorithm works quite well.

Simulation Run 6:

For this next simulation we will have 4 targets to track. The parameters remain the same. The position, velocity and acceleration of each target initially are:

Id: 1 pos: 500 velocity: 5 acceleration: 0

Id: 2 pos: 670 velocity: -10 acceleration: 0

Id: 3 pos: 550 velocity: 10 acceleration: 0

Id: 4 pos: 750 velocity: 5 acceleration: 0

The result obtained is not satisfactory. As shown in the figure 42, the particle filtering algorithm does estimate four trajectories but not accurately enough. There is a 'switch' between all the estimated trajectories. This occurs because of the smooth transition between the specific trajectories resulting to a wrong penalizing in the cost function. One way to handle this problem is by increasing resolution as demonstrated in the last two simulations represented.

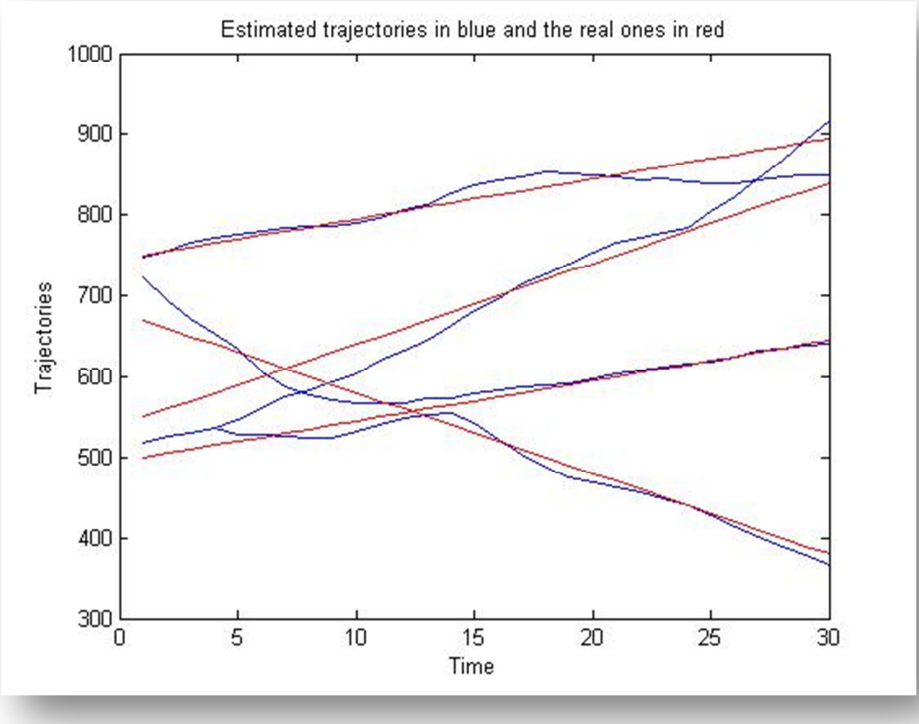

Figure 42

Simulation Run 7:

It is noticeable from the simulation before that the parameters stated are insufficient. A series of simulation runs exhibit that, the total number of estimated trajectories calculated from the cluster track algorithm in most cases when such conditions hold are more than the actual true paths. Moreover these spurious trajectories which are obtained are typically seen to be built out of portions of the actual true paths the targets have. Having already seen how the variety of parameters affects the performance of the particle filtering algorithm proposed, a change in the topology of the field is made in order to demonstrate the impact of the geometrical properties of the network in the performance of cluster track. Towards this end, the previously examined tracking situation is employed with the difference spotted in the layout of the binary proximity sensors. A total of 100 sensors are deployed with a sensing range of 15 units each. The separation between each sensor is 27 units. Figure 43 shows the results:

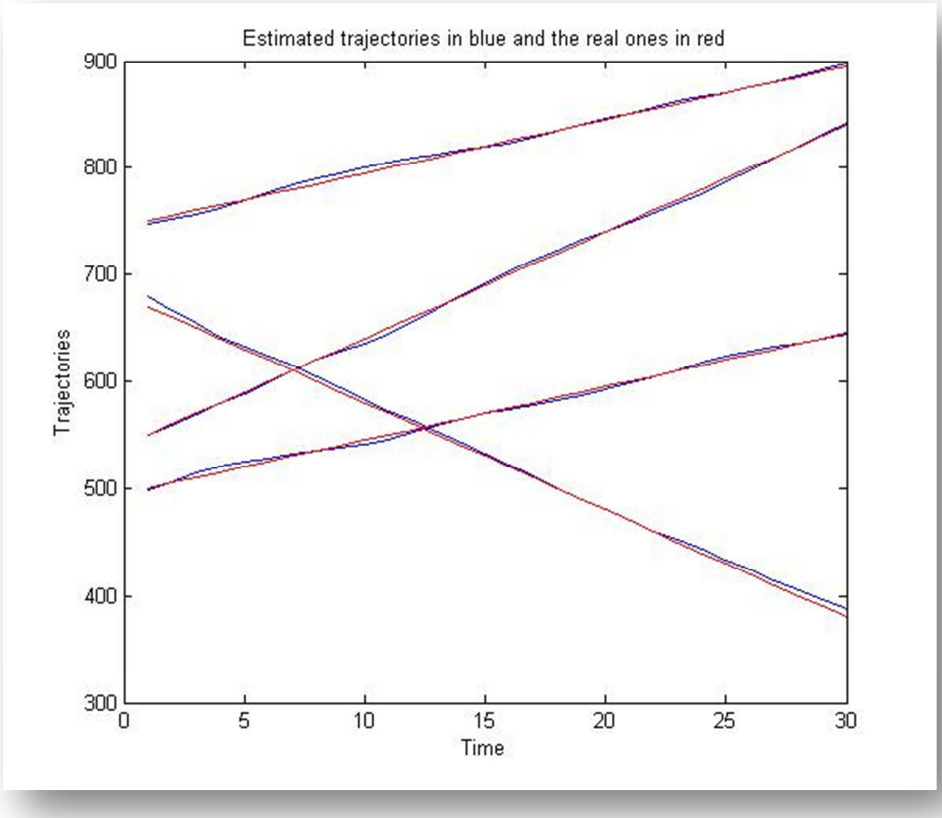

Figure 43

It is obvious that increasing the sensor density, in contrast to the previous example, gives us respectable better tracking. This is due to the fact that two targets which are close enough could now be distinguished, in opposition to the previous case, since their sensing range has been lowered resulting in increasing the resolution. Another fact that is associated with the topological features of a sensor network and has an impact on the tracking performance the cluster track algorithm could achieve, is the overlap of the sensing ranges the binary proximity sensor modalities have. More overlap leads to a larger partition of the domain of interest thus resulting again to a better resolution and a more satisfying tracking performance.

Simulation Run 8:

As the last simulation, we're going to try moving targets which have acceleration too and see how the algorithm works. The settings for the targets are the next ones:

Id: 1 pos: 400 velocity: 5 acceleration: 1

Id: 2 pos: 850 velocity: 0 acceleration: -0.5

Id: 3 pos: 550 velocity: 10 acceleration: 0

Id: 4 pos: 750 velocity: 5 acceleration: 0

We can observe in the figure 44 that the algorithm examined works very well and just has some problems with one path, but there's nearly no difference with the real one ending that the algorithm works quite well even in the most adverse scenarios.

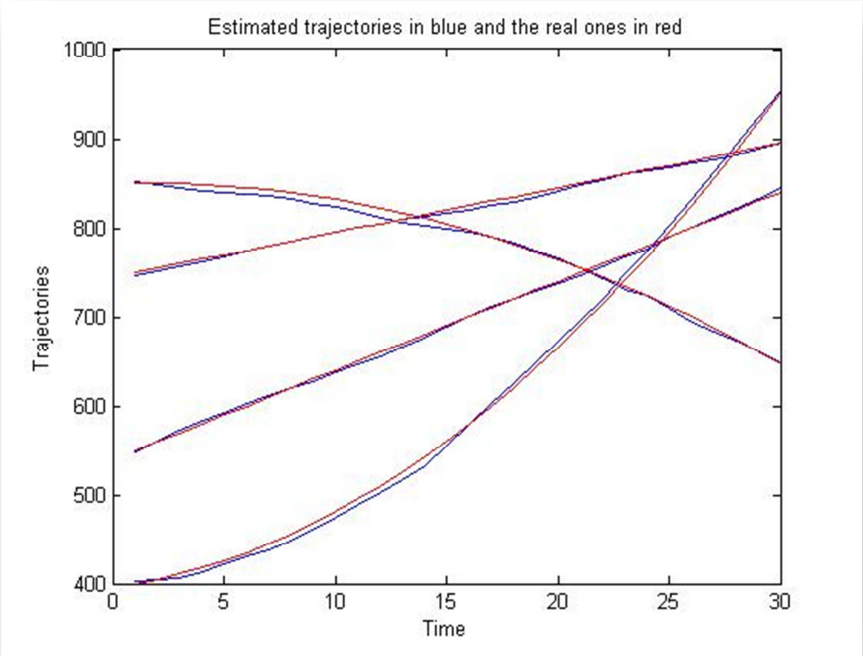

Figure 44

4.6 Matlab code

The following Matlab code is the necessary one for the main program to work. It is a language translation from the pseudo code that can be found in the chapter 4.3. All the specifics are described in that same section and will not be reproduced again in order not to be redundant.

```
function [best] = clusterTrack
%Main program

totalParticles=500;%Define the number of particles for this example
l=5*totalParticles;
m0=12;
H=30;
threshold=60;%Set the threshold
totalTime=30;%Time elapsed time
currTime=1;%initialize the time
[alltargets] = set_targets;%%Will be explained later(WBEL)%%
[sensor,numb_sens] = set_sensors;%%WBEL%%
[sensor] = is_enabled(alltargets,sensor,numb_sens,currTime);%%WBEL%%

if currTime==1
     [particles]=initializationStep1(sensor,numb_sens,totalParticles);%%WBE
     L%%
     currTime=currTime+1;%Once the initialization has been done move to the
     next time step
end
[sensor] = erase_enabled(sensor,numb_sens);%%WBEL%%
[sensor] = is_enabled(alltargets,sensor,numb_sens,currTime);%%WBEL%%

if currTime==2
     [particles]=initializationStep2(sensor,numb_sens,totalParticles,partic
     les);%%WBEL%%
     currTime=currTime+1;%Once the second initialization has been done add
     another time step
end
k=totalParticles+1;

for m=3:totalTime
     [sensor] = erase_enabled(sensor,numb_sens);%%WBEL%%
     [sensor]=is_enabled(alltargets,sensor,numb_sens,currTime);%%WBEL%%
     [particles]=extendParticles(m0,particles,currTime,k,totalParticles,sen
     sor,numb_sens);%extend the particles
     [interval,i] = feasible_space(sensor,numb_sens);%%WBEL%%
     numbParticles=(k-1)*m0*i;
     [particles]=calculatecost(particles,currTime,k,i,m0);%calculate the
     cost and sort according to ascending order of it
     [allclusters] = createClusters(H,k);%preallocate and create the new
     clusters ( I don't need to erase the information if I'm creating them
     all the time)
     allclusters(1).cluster(1)=particles(1);%register the first particle in
     the first cluster
     allclusters(1).addedNumber=1;
     [survivingParticles] = createSurvivingParticles;%%WBEL%%
     survivingParticles(1)=particles(1);%keep the particle to surviving
     particle list
```

```
    j=2;
    k=2;
    numbClust=1;

    while (j<=l && k<=totalParticles && j<=numbParticles)
        [belongs,belongingNumber]=belongsToCluster(threshold,j,particles,c
        urrTime,numbClust,allclusters);%%WBEL%%
        if belongs ~=0
          if (allclusters(belongingNumber).addedNumber)<H
            allclusters(belongingNumber).cluster((allclusters(belongingNu
            mber).addedNumber)+1)=particles(j);%add the particle to the
            returned cluster
            survivingParticles(k)=particles(j);%add it to the surviving
            particles
            k=k+1;
            allclusters(belongingNumber).addedNumber=(allclusters(belongi
            ngNumber).addedNumber)+1;%Add one to the particle number
            added already in that cluster
          else %just ignore the particle and go to the next particle
          end
        else
          numbClust=numbClust+1;
          allclusters(numbClust).cluster(1)=particles(j);%register
          the particle in the numbClust cluster
          allclusters(numbClust).addedNumber=1;
          survivingParticles(k)=particles(j);%register the particle into
          surviving particles list
          k=k+1;
        end
        j=j+1;
    end
    particles=survivingParticles;%delete the old list with the surviving
    particles and assign the new list
    currTime=currTime+1;
end
[best]=estimateTrajectories(allclusters,numbClust,threshold);%calculates
the final cost and estimates the trajectories
print_trajectories(best);
%print the true trajectories and the estimated ones
end
```

For a better understanding of the main function named *clusterTrack*, the list of created sub-functions and their definitions is shown. A flow chart of their mutual interaction is given as well. They are listed in order of appearance.

1. *Set Targets*: places all the targets in the space-time axes. In this case, the values were directly inserted in the function instead of being asked on screen every time the program was being run. Features such as position, speed and acceleration are given to each target and defined for all the timeline.

2. *Set Sensors*: the sensors are placed in the scenario and the number of sensors, its separation and sensing range are defined. Again, it could have been asked to the user in the time of running but it was made this way as it is faster when lots of repetitions are needed to check its stable response.

3. *Is Enabled*: shows whether the sensors have targets nearby or not. It just checks if the four targets are inside the influence area of each sensor.
4. *Initialization Step 1*: the first step to initialize the particles that will be necessary for the target tracking is described.
5. *Erase Enabled*: The purpose is to reset the enabled sensors.
6. *Initialization Step 2*: sets the particles in the second time slot so from this moment on the particle filters can develop themselves with their previous time steps data.
7. *Extend Particles*: The goal is to extend the particles so they can accept new data and be sorted later on.
8. *Feasible Space*: is valid for getting the feasible space intervals where the tracked targets can be. What we will get at the end is all the possible intervals and the total amount of them.

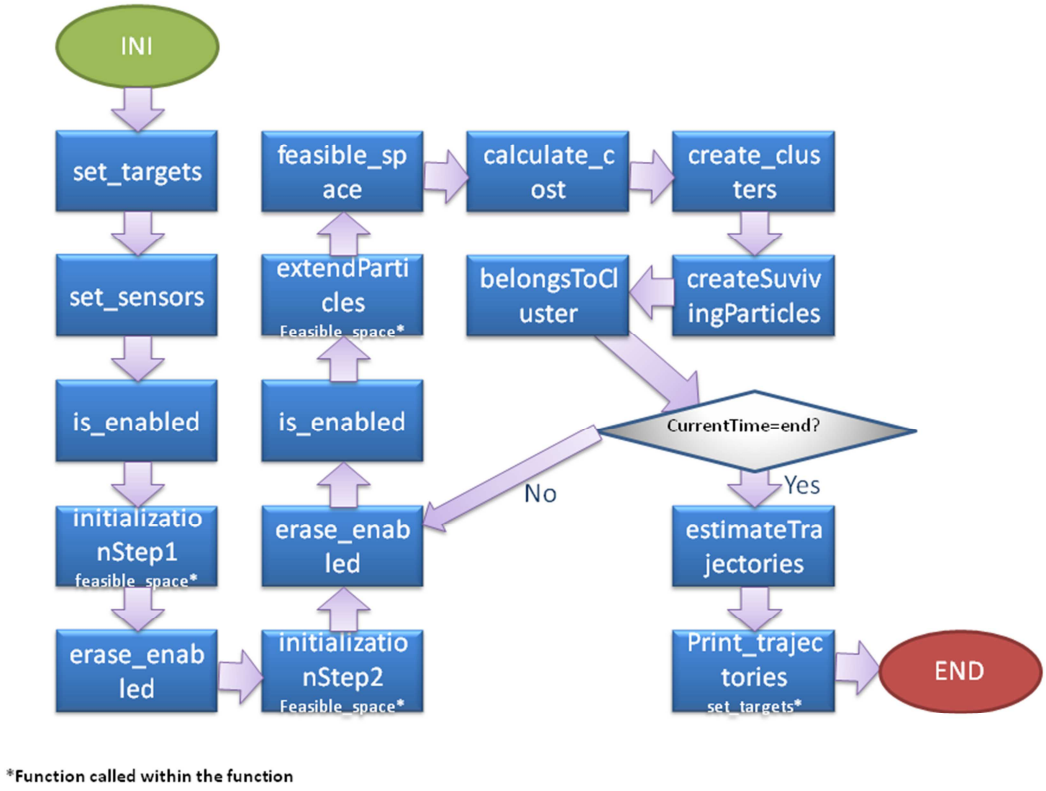

Figure 45

9. *Calculate Cost*: to estimate how probable it is for a trajectory to belong to a cluster (the more cost, the less probability of belonging) and sorts the particles from lower to higher cost.

10. *Create Clusters*: The main purpose is to create the clusters that will be used and needed in the main program.

11. *Create Surviving Particles*: creates the matrix of the still surviving particle.

12. *Belongs to Cluster*: The purpose is to realize if the particle belongs to any of the formed clusters. If the counting is greater than a threshold selected previously the particle will be discarded.

13. *Estimate Trajectories*: The main goal is to calculate the cost in the end of the timeline and return the best particle/trajectory in each cluster.

14. *Print Trajectories*: to print the real and estimated trajectories, both in the same figure, so the differences can be appreciated.

The Matlab code for the 14 functions included in *clusterTrack* will be described in the following sections.

4.6.1 Set targets

```
function [alltargets] = set_targets
%For setting the target's values
%the result is a matrix of targets containing all of them in every time
%to change the values of the targets, here
p1=400;
v1=5;
a1=1;
p2=850;
v2=0;
a2=-0.5;
p3=550;
v3=10;
a3=0;
p4=750;
v4=5;
a4=0;

target1(30)=struct('position',0, 'velocity',0,'acceleration',0);
target2(30)=struct('position',0, 'velocity',0,'acceleration',0);
target3(30)=struct('position',0, 'velocity',0,'acceleration',0);
target4(30)=struct('position',0, 'velocity',0,'acceleration',0);
target1(1).position=p1;
target1(1).velocity=v1;
target1(1).acceleration=a1;
target2(1).position=p2;
target2(1).velocity=v2;
target2(1).acceleration=a2;
target3(1).position=p3;
target3(1).velocity=v3;
target3(1).acceleration=a3;
target4(1).position=p4;
target4(1).velocity=v4;
target4(1).acceleration=a4;

for m=2:30
```

```
    if target1(1).acceleration~=0
         target1(m).velocity=target1(m-1).velocity + target1(m-
         1).acceleration;
    else
         target1(m).velocity=target1(m-1).velocity;
    end
    if target2(1).acceleration~=0
         target2(m).velocity=target2(m-1).velocity + target2(m-
         1).acceleration;
    else
         target2(m).velocity=target2(m-1).velocity;
    end
    if target3(1).acceleration~=0
         target3(m).velocity=target3(m-1).velocity + target3(m-
         1).acceleration;
    else
         target3(m).velocity=target3(m-1).velocity;
    end
    if target4(1).acceleration~=0
         target4(m).velocity=target4(m-1).velocity + target4(m-
         1).acceleration;
    else
         target4(m).velocity=target4(m-1).velocity;
    end
    target1(m).position=target1(m-1).position + target1(m-1).velocity;
    target1(m).acceleration=target1(1).acceleration;
    target2(m).position=target2(m-1).position + target2(m-1).velocity;
    target2(m).acceleration=target2(1).acceleration;
    target3(m).position=target3(m-1).position + target3(m-1).velocity;
    target3(m).acceleration=target3(1).acceleration;
    target4(m).position=target4(m-1).position + target4(m-1).velocity;
    target4(m).acceleration=target4(1).acceleration;
end%tctn: add another if and position and acceleration
alltargets=struct('target1',target1,'target2',target2,'target3',target3,'ta
rget4',target4);
end
```

4.6.2 Set sensors

```
function [sensor,numb_sens] = set_sensors
%For setting the sensors all over the x axis
%The result is a matrix of sensors that contains all of them with position,
%radius, enabled and Id
%to change the value of the sensors, here
numb_sens=70;
separation=27;%separation=60;
sensing_range=15;%sensing_range=40;
sensor(100) = struct('position',0, 'radius',0,'enabled',0,'Id',0);
for m=1:numb_sens
    sensor(m) = struct('position',separation*(m-1),'radius',
    sensing_range,'enabled',0,'Id',m);
end
end
```

4.6.3 Is_enabled

```
function [sensor] = is_enabled(alltargets,sensor,numb_sens,time)
%for checking which sensors must be ON because there is a target in its
limits
%returns the proper sensors enabled
for x=1:numb_sens
     if ((sensor(x).position - sensor(x).radius)<=
     alltargets.target1(time).position)&&
     (alltargets.target1(time).position<=(sensor(x).position +
     sensor(x).radius))
           sensor(x).enabled=1;
     elseif ((sensor(x).position - sensor(x).radius)<=
     alltargets.target2(time).position)&&
     (alltargets.target2(time).position <= (sensor(x).position +
     sensor(x).radius))
          sensor(x).enabled=1;
     elseif ((sensor(x).position - sensor(x).radius)<=
     alltargets.target3(time).position)&&
     (alltargets.target3(time).position <= (sensor(x).position +
     sensor(x).radius))
          sensor(x).enabled=1;
     elseif ((sensor(x).position - sensor(x).radius)<=
     alltargets.target4(time).position)&&
     (alltargets.target4(time).position <= (sensor(x).position +
     sensor(x).radius))
          sensor(x).enabled=1;
     else
          sensor(x).enabled=0;
     end
end
end
```

4.6.4 InitiaizationStep1

```
function[particles]=initializationStep1(sensor,numb_sens,totalParticles)
%For spreading the particles all over the feasible space
%I put space for 24.000 particles because I will not have more than 4
targets
%so: totalParticles(500)*m0(12)*max targets(4)=24.000.

[interval,i]=feasible_space(sensor,numb_sens);%calculate de feasible space
particles(24000)=struct('position',0,'cost',0);%I create the particles
for m=1:24000%I give them the space for the 30 time slots
     particles(m).position(30)=0;
     particles(m).cost(30)=0;
end

totalgap=0;
for m=1:i
     sum=(interval(m).end - interval(m).beginning);
     totalgap=totalgap + sum;
end%I calculate the total gap of the intervals

for n=1:totalParticles
     point=round(rand*totalgap);
```

```
     for m=1:i
          if point<(interval(m).end - interval(m).beginning)
            break
          else
            point=(point - interval(m).end + interval(m).beginning);
          end
     end
     particles(n).position(1)=interval(m).beginning + point;
end
end
```

4.6.5 Erase_enabled

```
function [sensor] = erase_enabled(sensor,numb_sens)
%for erasing the enabled sensors before sampling again
%returns all sensors disabled
for x=1:numb_sens
   sensor(x).enabled=0;
end
end
```

4.6.6 InitializationStep2

```
function[particles]=initializationStep2(sensor,numb_sens,totalParticles,par
ticles)
%For extending the particles in the second time slot

[interval,i]=feasible_space(sensor,numb_sens);%calculate de feasible space
totalgap=0;
for m=1:i
     sum=(interval(m).end - interval(m).beginning);
     totalgap=totalgap + sum;
end%I calculate the total gap of the intervals

for n=1:totalParticles
     point=round(rand*totalgap);
     for m=1:i
          if point<(interval(m).end - interval(m).beginning)
            break
          else
            point=(point - interval(m).end + interval(m).beginning);
          end
     end
     particles(n).position(2)=interval(m).beginning + point;
end
end
```

4.6.7 ExtendParticles

```
function[newParticles]=extendParticles(m0,particles,currTime,k,totalPaticl
es,sensor,numb_sens)
%This function is for extending the particles.
%The output is the extended matrix of particles
%I put space for 24.000 particles cos i will not have more than 4 targets
%so: totalParticles(500)*m0(12)*max targets(4)=24.000.
```

```
newParticles(24000)= struct('position',0,'cost',0);%I create the particles
for m=1:24000%I give them the space for the 30 time slots
     newParticles(m).position(30)=0;
     newParticles(m).cost(30)=0;
end

[interval,i]=feasible_space(sensor,numb_sens);%calculate de feasible space
if (k-1)~=totalParticles
     totalParticles=(k-1);
end
a=1;
for m=1:totalParticles%because in the clusterTrack, totalParticles is the
maximun number that survives
     for n=1:i
          for x=1:m0
            newParticles(a)=particles(m);%I copy the information of one
            particle to m0*i particles
            point=round(interval(n).beginning + (interval(n).end -
            interval(n).beginning)*rand);
            newParticles(a).position(currTime)=point;
             a=a+1;
          end
     end
end
end
```

4.6.8 FeasibleSpace

```
function [interval,i] = feasible_space(sensor,numb_sens)
%For getting the feasible space intervals where the targets can be
%returns the intervals where the targets can be and the number of them

i=1;
interval(10).beginning=0;%no more intervals than 10%
interval(10).end=0;
for x=1:numb_sens
     if sensor(x).enabled == 1
          if x==1
            interval(i).beginning =0;
          else if sensor(x-1).enabled == 0  %previous one is a 0
               if (sensor(x).position - sensor(x).radius) < (sensor(x-
               1).position + sensor(x-1).radius)
                 interval(i).beginning = (sensor(x-1).position + sensor(x-
                 1).radius);
               else
                 interval(i).beginning = (sensor(x).position -
                 sensor(x).radius);
               end
               if sensor(x+1).enabled == 0  %next one is a 0
                 if (sensor(x).position + sensor(x).radius) >
                 (sensor(x+1).position - sensor(x+1).radius)
                   interval(i).end = (sensor(x+1).position -
                   sensor(x+1).radius);
                 else
                   interval(i).end = (sensor(x).position +
                   sensor(x).radius);
```

```
                end
                i=i+1;
              end%if next one is a 1 just don't finish the interval
            else %previous one is a 1
              if sensor(x+1).enabled == 0  %next one is a 0
                if (sensor(x).position + sensor(x).radius) >
                (sensor(x+1).position - sensor(x+1).radius)
                  interval(i).end = (sensor(x+1).position -
                  sensor(x+1).radius);

                else
                  interval(i).end = (sensor(x).position + sensor(x).radius);
                end
                i=i+1;
              %if next one is a 1, just don't finish the interval
              end
            end
            end
      end
end
i=i-1;
end
```

4.6.9 CalculateCost

```
function [newParticles] = calculatecost(particles,currTime,k,i,m0)
%This function calculates cost of the particle and sorts the particles in
order of cost

control=(k-1)*i*m0;
cost(control)=0;
for m=1:control
     particles(m).cost(currTime)=abs(particles(m).position(currTime) +
     particles(m).position(currTime-2) - 2*particles(m).position(currTime-
     1))+particles(m).cost(currTime-1);
     cost(m)=particles(m).cost(currTime);
end

[y,i]=sort(cost);%i is the vector with the previous position of the sorted
elements in y
newParticles(control)= struct('position',0,'cost',0);%I create the
particles
for m=1:control%I give them the space for the 30 time slots
     newParticles(m).position(30)=0;
     newParticles(m).cost(30)=0;
end
for m=1:control%I sort the elements from the one with less cost to the one
with more cost.
     aux=i(m);
     newParticles(m)=particles(aux);
end
end
```

4.6.10 CreatClusters

```
function [allclusters] = createClusters(H,k)
%This function is for creating the clusters for the clusterTrack
%In the worst case, with threshold=1, there can be totalParticles clusters
because this is the maximum number of particles that survive after the
execution of clusterTrack

cluster(H)= struct('position',0,'cost',0);%I create one cluster

for m=1:H%I give it the space for the 30 time slots
     cluster(m).position(30)=0;
     cluster(m).position(30)=0;
end

allclusters(k-1) = struct('cluster',cluster,'addedNumber',0);
%I preallocate all the clusters

for m=1:(k-1)
     allclusters(m).cluster=cluster;
     allclusters(m).addedNumber=0;
end
end
```

4.6.11 CreateSurvivingParticles

```
function [particles] = createSurvivingParticles
%Create the matrix of the surviving particles
%Even if I need it just the size of totalParticles I will give it a
%size of 24.000 to avoid programming problems

particles(24000)= struct('position',0,'cost',0);%I create the particles
for i=1:24000%I give them the space for the 30 time slots
    particles(i).position(30)=0;
end
```

4.6.12 BelongsToCluster

```
function[belongs,n]=belongsToCluster(threshold,i,particles,currTime,numbClu
st,allclusters)
%This function is to know if the particle belongs to any of the already
formed clusters

thres=threshold*currTime;

for n=1:numbClust
     counting=0;
     for m=1:currTime
          counting=abs(allclusters(n).cluster(1).position(m)-
          particles(i).position(m))+counting;
     end
     if counting<=thres
          break;
     end
```

```
end
if (n==numbClust && counting>thres)%This means that the for loop ended
without the break
     belongs=0;
else
     belongs=1;
end
end
```

4.6.13 EstimateTrajectories

```
function [best]=estimateTrajectories(allclusters,numbClust,threshold)
%This function is to calculate the cost in the final time (30) and to take
the best particle of each cluster
for m=1:numbClust
     for n=1:allclusters(m).addedNumber
          allclusters(m).cluster(n).cost(30)=(allclusters(m).cluster(n).cost
          (29))+ (abs(allclusters(m).cluster(n).position(30) +
          allclusters(m).cluster(n).position(28) -
          2*allclusters(m).cluster(n).position(29)));
     end
end

for m=1:numbClust
     if allclusters(m).cluster(1).cost(30)<(9*threshold)
          best(m)=allclusters(m).cluster(1);
          for n=2:allclusters(m).addedNumber
            if best(m).cost(30)>allclusters(m).cluster(n).cost(30)
               best(m)=allclusters(m).cluster(n);
            end
          end
     end
end
end
```

4.6.14 Print_Trajectories

```
function print_trajectories(best)
%This function prints the true and the estimated trajectories

[alltargets] = set_targets;

close
y=best(1).position;
plot(y)
hold on
y=best(2).position;
plot(y)
y=best(3).position;
plot(y)
y=best(4).position;
plot(y)
z1(30)=0;
z2(30)=0;
z3(30)=0;
z4(30)=0;
```

```
for i=1:30
    z1(i)=alltargets.target1(i).position;
    z2(i)=alltargets.target2(i).position;
    z3(i)=alltargets.target3(i).position;
    z4(i)=alltargets.target4(i).position;
end
plot(z1,'r')
plot(z2,'r')
plot(z3,'r')
plot(z4,'r')
title('Estimated trajectories in blue and the real ones in red')
xlabel('Time')
ylabel('Trajectories')
hold off
%figure

end
```

4.7 Conclusions

The promising results obtained here indicate that binary proximity sensors can form the basis for a robust architecture for wide area surveillance and tracking. When the target paths are smooth enough, although I have tested it with acceleration targets with quite satisfactory results, ClusterTrack particle filter algorithm gives excellent performance in terms of identifying and tracking different target trajectories.

It is not surprising that this kind of sensor network has attracted a lot of attention lately. Despite the minimal information a binary proximity sensor provides, networks of these sensing modalities can track all kinds of different targets classes accurate enough. However, although a lot of work has been done for the case of tracking in one and two dimensions a lot of aspects of the preceding proposed algorithm and theorems must be further studied in order to gain a better insight to the more general problem of tracking with binary sensors and extend it to the three dimensions scenario.

5 Conclusions

Different WSN protocols were reviewed and, in summary, Bluetooth and ZigBee are the most suitable ones for low data rate applications with limited battery power (such as mobile devices and battery-operated sensor networks), due to their low power consumption leading to a long lifetime. On the other hand, for high data rate implementations (such as audio/video surveillance systems), UWB and Wi-Fi would be better solutions because of their low normalized energy consumption. RFID has a crucial role to play in business and for individuals alike going forward. The impact of 'wireless' identification is exerting strong pressures in RFID technology and services research and development, standards development, security compliance and privacy, and many more. The economic value is proven in some countries while others are just on the verge of planning or in pilot stages, but the wider spread of usage has yet to take hold or unfold through the modernisation of business models and applications. Despite the fact that its promising technological capabilities into the mainstream of business and society, RFID does pose challenges in terms of legal transparency, standards, product information services and compatibility issues, privacy and security and other security threats such as spoofing, industrial sabotage and espionage.

As it has been seen throughout the document, WSNs have created themselves a promising future as the areas that can be applied are really numerous. A great number of possible application has been reviewed and plenty of universities and enterprises worldwide which are contributing to the WSN development. Libelium, a Spanish company, has developed an adaptable and innovative sensor and it has been deeply examined. This study of the sensor has been focused mainly to traffic applications but it cannot be forgotten the more than 50 different application compilation that Libelium has published.

Holiday crowds are a good case in which it is necessary to keep count of the cars on the roads. To this end a Matlab simulation has been produced for target tracking and counting using a network of binary sensors that e.g. could be implemented in the sensor Libelium has developed. This would allow the appliance to spend much less energy when transmitting information and to make more independent devices. The promising results obtained indicate that binary proximity sensors can form the basis for a robust architecture for wide area surveillance and tracking. When the target paths are smooth enough ClusterTrack particle filter algorithm gives excellent performance in terms of identifying and tracking different target trajectories. This algorithm could, of course, be used for different applications and that could be done in future researches.

It is not surprising that binary proximity sensor networks have attracted a lot of attention lately. Despite the minimal information a binary proximity sensor provides, networks of these sensing modalities can track all kinds of different targets classes accurate enough. However, although a lot of work has been done for the case of tracking in one and two dimensions a lot of aspects of the preceding proposed algorithm and theorems must be further studied in order to gain a better insight to the more general problem of tracking targets with binary sensors and, for example, extend it to the three dimensions scenario.

6 Bibliography

[1] Petar M.Djuric, Jayesh H. Kotecha, Jianqui Zhang, Yufei Huang, Tadesse Ghirmai, Mónica F. Bugallo and Joaquín Míguez.”Particle Filtering”. IEEE Signal Processing Magazine. September 2003.

[2] Jaspreet Singh, Upamanyu Madhow, Rajesh Kumar, Subhash Suri, Richard Cagley. “Tracking Multiple Targets Using Binary Proximity Sensors”. 2007.

[3] Charalambos Kourdounakis. “Tracking Multiple Targets Using Binary Proximity Sensors”. Undergraduate Case Study. March 2008.

[4] Javier Atencia, Raúl Nestar. “Aprenda Matlab 6.0 como si estuviera en primero”.July 2001.

[5] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen. “A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi”

[6] “Short-Range Wireless Communications”, John Wiley & Sons Ltd, 2009

[7] “Near-Field Technology - An Emerging RF Discipline”, Q-Track Corporation, 2006

[8] “Shrouds of Time - The History of RFID, AIM Publication, 2001 and Ubiquitous Network Societies: The Case of Radio Frequency Identification”, International Telecommunication Union, 2005

[9] “A Guide to Radio Frequency Identification”, WILEY, 2007

[10] “RFID: An Introduction, Microsoft EMEA Manufacturing Industry Solutions Architect”, Simon Holloway, June 2006

[11] “Third Generation Active RFID Bursts onto The Scene”, IDTechEx, October 2008

[12] “Wireless Sensor Network 2009-2019: The New Market for Ubiquitous Sensor Network (USN)”, IDTechEx, 2008

[13] “Radio Frequency identification (RFID) Wireless Network- Some Pain, Much To Gain” Malaysian Communications and Multimedia Comission, 2010

[14] “ITU Internet Reports: The Internet of Things”, International Telecommunication Union (ITU), 2005

[15] www.libelium.com

[16] Jayesh H. Kotecha and Petar M. Djuric “Gaussian Particle Filtering” IEEE Transactions On Signal Processing, Vol. 51, NO.10, October 2003

[17] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus. Tracking a moving object with a binary sensor network. In *Proc. ACM SenSys*, 2003.

[18] J. Shin, L. Guibas, and F. Zhao. A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks. In *Proc. of IPSN*, April 2003.

[19] N. Shrivastava, R. Mudumbai, U. Madhow, and S. Suri. Target tracking with binary proximity sensors: Fundamental limits, minimal descriptions and Algorithms

[20] E. Ferro and F. Potorti, "Bluetooth and Wi-Fi wireless protocols: A survey and a comparison," IEEE Wireless Commun., vol. 12, no. 1, pp. 12-16, Feb. 2005

[21] Baker, N. "ZigBee and Bluetooth: Strengths and weaknesses for industrial applications," IEE Computing & Control Engineering, vol. 16, no. 2, pp 20-25, April/May 2005

[22] A. Sikora and V. F. Groza, "Coexistence of IEEE802.15.4 with other systems in the 2.4 GHz-ISM-Band," in Proc. IEEE Instrumentation & Measurement Technology Conference, Ottawa, May 2005, pp.1786-1791.

[23] www.wikipedia.com

[24] http://glacsweb.org/

[25] http://www.btnode.ethz.ch/

[26] http://www.coalesenses.com/

[27] Daniele Puccinelli and Martin Haenggi "Wireless Sensor Networks: Applications and Challenges of Ubiquitous Sensing" Third Quarter 2005, IEEE Circuits and Systems Magazine

[28] Wei Zhang, Guo-Zhen Tan, Hui-Min Shi and Ming-Wen Lin "A Distributed Threshold Algorithm for Vehicle Classification Based on Binary Proximity Sensors and Intelligent Neuron Classifier" Journal of Information Science and engineering 26, 769-783 (2010)

[29] Nan Ding Guozhen Tan, Wei Zhang and Hongwei Ge "Distributed Algorithm for Traffic Data Collection and Data Quality Analysis Based on Wireless Sensor Networks" International Journal of Distributed Sensor Networks Volume 2011 (2011), Article ID 717208, 9 pages doi:10.1155/2011/717208