UCGE Reports

Number 20302

Department of Geomatics Engineering

Wireless Sensor Network-Based Distributed GNSS Receiver Architecture for Infrastructure Monitoring

(URL: http://www.geomatics.ucalgary.ca/graduatetheses)

by

Lionel Garin

January 2010

THE UNIVERSITY OF CALGARY

Wireless Sensor Network-Based Distributed GNSS Receiver Architecture for Infrastructure Monitoring

by

Lionel Garin

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF GEOMATICS ENGINEERING

CALGARY, ALBERTA

JANUARY, 2010

ABSTRACT

Wireless Sensor Networks (WSN) have received a marked interest in the recent years. They are constituted of low cost and completely autonomous nodes with extremely limited power and processing capabilities that wirelessly communicate between each other. The majority of field deployments is executed in an ad-hoc manner in an area with no power or communication infrastructure, and in unpredictable locations. Their essential use is to provide spatio-temporal information about their environment. They can collect physical information such as temperature, or boundaries of a diffuse phenomenon over time and space such as expanding chemical plumes. They can be also used in collaborative data collection such as seismologic monitoring, or construction monitoring. Their optimal use presupposes knowledge of their relative location, either in an approximate manner, or in a very accurate manner. Earlier solutions to provide the position or location function were not using Global Navigation Satellite Systems, usually resulting in low accuracy. Later, existing GPS receivers were attached to a subset of the nodes, without trying to rethink their architecture in the context of WSN. In order to achieve optimality, the GPS function should become an integral part of the node and the network architectures in such a way that the redundancy is eliminated across nodes, and the energy consumption is reduced as much as possible and distributed among nodes as evenly as possible.

The main goal of this work is to propose a practical solution for accurate relative positioning between nodes of a Wireless Sensor Network, outlining their fundamental

limitations, and developing a GPS distributed architecture aligned with these limitations. In particular a new concept, namely the WA-GPS or Wireless Sensor Network A-GPS concept, will be introduced, where the per node support to the location function is extremely limited, and the effort to collect GPS necessary information is distributed among all nodes. The problem of synchronization between nodes and its unacceptable impact on carrier phase relative positioning accuracy will be exposed and a novel technique, the Ambiguity Resolution of Time Integer (ARTI), will be proposed as mitigation. Three WSN architectures apt to support relative positioning will be proposed and assessed in terms of energy consumption and robustness.

"I do not think there is any thrill that can go through the human heart like that felt by the inventor as he sees some creation of the brain unfolding to success... such emotions make a man forget food, sleep, friends, love, everything." Nikola Tesla

DEDICATION

My deepest thanks go to my dear wife, Elizabeth, who supported me tremendously throughout my study, by constantly reminding me of my ultimate goal when the pressure and stress from the daily work were pulling me away, and accepting the sacrifice of precious family time for this work to come to fruition.

ACKNOWLEDGEMENTS

I express sincere appreciation to Professor Gérard Lachapelle and Professor Elizabeth Cannon for their help in making me understanding the unique requirements and level of excellence called for by such an academic work, so different for me having for so long pursued a career in an industrial environment.

Lastly, but in no sense the least, I am thankful to all students from the University of Calgary I have interacted with, who gave me the thrill of going back to this unique learning experience, and kept me young by unknowingly helping me to renew my life-long learning commitment by their refreshing questions and points of view.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF UNITS

dBm	Decibel referenced to 1mW level.
J	Joule.
kB	KiloByte.
kbps	kilobit per second.
mA	MilliAmpere.
mA.h	MilliAmpere.hour.
MB	MegaByte.
pJ	picoJoule.

LIST OF SYMBOLS

c	Velocity of light(m/s).
c_{air}	Velocity of sound in the air (m/s).
$C_{float\|\tilde{N}}$	Conditional covariance matrix for the float ambiguities.
$C^1_{r,Master}$	Sub millisecond code offset for the nth satellite received at the master in fraction of one millisecond (no units).
$C^1_{r,second}$	Sub millisecond PRN code offset of the Reference satellite at the secondary in fraction of one millisecond (no units).
$C^2_{r,second}$	Sub millisecond code offset of the first non reference satellite at the secondary in fraction of one millisecond (no units).
$C^{(M-1)}_{r,second}$	Sub millisecond code offset of the (M-1) th non reference satellite at the secondary in fraction of one millisecond (no units).
$C^n_{r,Master}$	Sub millisecond code offset for the nth satellite received at the master in fraction of millisecond (no units).
$\delta^p_{k,1,\varphi}(t_k)$	Hardware delays and multipath effects on the L1 carrier phase (cycles).

$\delta^{p}_{master,\varphi}(t_k)$	Hardware delays and multipath delays on the L1 carrier phase (in L1 cycles).
$\delta^{p}_{master,\Phi}(t_{r,master})$	Hardware delay and multipath effects on the L1 carrier phase in master receiver (in metres).
$\delta^{p}_{second,l,\Phi}(t^{p}_{s,second})$	Hardware delay and multipath effects on the L1 carrier phase (in metres).
$d^{p}_{k,1,\varphi}(t_k)$	Multipath delay between satellite p and receiver k at L1 (cycles).
$d^{p}_{1,\varphi}(t_k)$	Satellite p Hardware delay at L1 (cycles).
Δ	Inter node clock offset.
d	Propagation delay.
$d_{k,1,\varphi}(t_k)$	Receiver k hardware delay at L1 (cycles).
dt^{p}	pth satellite clock error vs. GPS system time. It is found in the broadcast ephemeris (in seconds).
dt_1	REF satellite clock error vs. GPS system time. It is found in the broadcast ephemeris (in seconds).
$dt_2, \dots, dt_{M-1}$	Second to M-1 satellite clock error vs. GPS system time. It is found in the broadcast ephemeris (in seconds).
dt_{master}	Master receiver clock time error (in seconds).

dt_{second}	Receiver clock error vs. GPS system time at receive time (in seconds).
DD_{12}^{12}	Distance double difference.
$\varepsilon_{1,\varphi}$	L1 phase measurement noise (cycles).
$\varepsilon_{l,\Phi}$	L1 carrier phase measurement noise (in metres).
η	Path loss exponent.
f_1	Nominal frequency L1(Hz).
$I_{k,1,\varphi}^{p}(t_k)$	Ionospheric C/A phase delay at L1 frequency between satellite p and receiver k.
$I_{master,\varphi}^{p}(t_k)$	pth satellite Ionospheric L1 carrier phase advance at master (negative value) (in L1 cycles).
$I_{master,\Phi}^{p}\left(t_{r,master}\right)$	Ionospheric delay at receive time between satellite p and master receiver (in metres).
$I_{second,l,\Phi}^{p}\left(t_{s,second}^{p}\right)$	Ionospheric delay at transmit time t^p of the distance between satellite p and secondary receiver (in metres).
λ_1	L1 Frequency Wavelength (in metres).
N_{master}^{p}	Integer Ambiguity of pth satellite at master. It remains constant during the period of observation (no units).

$\widetilde{\mathrm{N}}^1$	Best integer candidate.
$\widetilde{\mathrm{N}}^2$	Second best integer candidate.
$N^1_{s,second}$	Unknown integer number of milliseconds of correction to the transmit time of Reference satellite (no units).
$N^{1,n}_{s,Master}$	Integer number of milliseconds between nth satellite code offset reference point minus first (i.e. reference) satellite code offset point (no units).
$N^{1,2}_{second}$	Integer number of milliseconds of correction of second satellite transmit time at secondary minus reference satellite transmit time (no units).
$N^p_{second}(l)$	Number of cycle integer ambiguities between p satellite and secondary receiver at l=L1 frequency. Fixed number, unless cycle slips (no units).
$\varphi^p_{master}(t_{r,master})$	Carrier phase on L1 of pth satellite received at master receiver (in L1 cycles).
$\Phi^p_{master.l}(t_{r,master})$	Cumulated carrier phase measurement at $t_{r,master}$ at the master receiver at frequency L1 from the satellite p (in metres).

$\Phi^{p}_{second}(t^{p}_{s,second})$	Cumulated carrier phase measurement at the secondary receiver at frequency L1 of a signal that was transmitted at $t^{p}_{s,second}$ from the satellite p (in metres).
$\Omega(\tilde{N})$	Weighted sum squared of ambiguity residuals.
$P_{r,dB}(d_0)$	Received power in dB at reference distance d_0.
θ	Air temperature in degrees Celsius.
$\rho_k^p(\hat{t}^p)$	Geometric distance between satellite p and receiver k at true time at epoch of transmission $\hat{t}^p$(m).
$\rho^p_{master}(t^p)$	Geometric distance traveled by the code form transmission of satellite p to reception at master receiver antenna (in metres).
$\rho^p_{master}(t_{r,master})$	Geometric distance traveled by the signal from transmission at satellite p and received at master receiver antenna at $t_{r,master}$ (in metres).
$\rho^p_{second}(t^p_{s,second})$	Geometric distance traveled by the signal from transmission at satellite p at time t^p to reception at secondary receiver antenna (in metres).
$t^{p,second}$	Transmit time at the satellite p of the signal received at the secondary receiver.

$t^{1}_{s,Master}$	Transmit time of the first satellite received at the Master (in seconds).
$t^{1}_{s,second}$	Accurate Reference Satellite transmit time of the signal received at $t_{sync,second}$ at secondary receiver (in seconds).
$t^{approx}_{s,second}$	Approximate GPS transmit time for a signal received at secondary receiver rounded as an integer number of milliseconds (in seconds).
$t^{n}_{s,Master}$	Transmit time of the nth satellite received at the Master (in seconds).
$t_{r,master}$	Receive time at the master receiver from pth satellite.
$t^{second}_{s,(M-1)}$	(M-1) th satellite transmit time received at the secondary receiver (in seconds).
t_{flight}	Satellite-receiver average time of flight ($t_{flight} \approx 75 \cdot 10^{-3} s$econds).
t_k	Time at receiver k at the epoch the code entered the antenna.
$\hat{t}_k$	True time at receiver k at the epoch the code entered the antenna.
$t_{r,master}$	True Receive time at master at measure of φ^{p}_{master} (in seconds).

$t_{sync,second}$	Reception time at secondary receiver synchronized from master receiver (in seconds).
$\hat{t}^p$	True time at satellite p at epoch of transmission.
$T_k^p(t_k)$	Tropospheric delay in distance (m).
T_{master}^p	pth satellite Tropospheric delay (always positive) (in metres).
$T_{master}^p(t_{r,master})$	Tropospheric delay at receive time $t_{r,master}$ between satellite p and master receiver (in metres).
$T_{second}^p(t_{s,second}^p)$	Tropospheric delay at transmit time t^p of the distance between satellite p and secondary receiver (in metres).
$X_{\sigma,dB}$	Normal variable in dB units.

LIST OF ABBREVIATIONS

1-PPS	One Pulse Per Second.
A-GPS	Assisted GPS.
ALU	Arithmetic and Logic Unit.
AoA	Angle of Arrival.
ARTI	Ambiguity Resolution of Time Integer.
BBD	Bucket Brigade Device.
C/A-L1	Coarse Acquisition GPS signal transmitted on L1 frequency.
CDMA	Code Division Multiple Access.
CMOS	Complementary metal–oxide–semiconductor.
CPU	Central Processing Unit.
CRC	Cyclic Redundancy Check.
CSMA/CA	Carrier Sensing Multiple Access/Collision Avoidance.
CSMA/CD	Carrier Sensing Multiple Access/Collision Detection.
DOP	Dilution of Precision.
DSSS	Direct Spread Spectrum Sequence.

FND	First Node Dies.
GCC	GNU Compiler Collection.
GNSS	Global Navigation Satellite System.
GPS	Global Positioning System.
GSM	Global System for Mobile communications.
GUI	Graphical User Interface.
HNA	Half Nodes Alive.
LEACH	Low Energy Adaptive Clustering Hierarchy.
LND	Last Node Dies.
MAC	Medium Access Control.
P(Y)-L1	Precise code GPS signal transmitted on L1 frequency.
P(Y)-L2	Precise code GPS signal transmitted on L2 frequency.
RAM	Random Access Memory.
RBS	Reference Broadcast Synchronization.
RINEX	Receiver Independent Exchange Format.
RSSI	Receiver Signal Strength Indicator.
RTS	Request To Send.

SDP	Semi-Definite Positive.
SHM	Structural Health Monitoring.
SOC	Systems-On-a-Chip.
SPS	Standard Positioning Service.
TDMA	Time Division Multiple Access.
TDoA	Time Difference of Arrival.
ToA	Time of Arrival.
TPSN	Timing-sync Protocol for Sensor Networks.
TTA	Time To Ambiguities fixing.
WA-GPS	Wireless sensor networks A-GPS.
WCDMA	Wide Band Code Division Multiple Access.
WSN	Wireless Sensor Network.
WWAN	Wireless Wide Area Network.

CHAPTER 1

INTRODUCTION

1.1 Motivations for Wireless Sensor Network Accurate Location

The importance of accurate location information has been recognized quite early in the Wireless Sensor Networks (WSN) field. GPS technology advantages are also widely acknowledged for accuracy, and convenience. However, all WSN location solutions proposed in the literature, either make traditional use of GPS by hooking up a complete GPS box to a node, or avoid altogether its usage by proposing alternate solutions that, ironically, recreate the fundamental GPS functions using similar technologies with varying successes (Marotti et al. 2005). The Cricket technology (Priyantha 2000) that implements ranging measurements by synchronously sending an RF pulse and an ultrasonic pulse is a good example. The time of transmission tagging problem, elegantly resolved in GPS, is avoided by assuming that the RF signals propagate quasi instantaneously compared to the ultrasonic signals; the RF pulse reception time is assumed to be the transmission time of the ultrasonic pulse. The ranging measurement is then accomplished on the ultrasonic signal only, with limited resolution. A closer examination of this state of affairs suggests that the WSN community has a severely limited knowledge of GPS, and of its potential and limitations to try to seamlessly integrate it into a WSN architecture. An opinion that has probably been expressed by the

pioneers in the field and that is handed over without further examination is that GPS is a black-box, expensive and power hungry, with limited interactivity, that delivers position, if its antenna is correctly exposed to the sky. The present work tries to dispel this misconception by drawing new ideas from Assisted GPS (A-GPS) (Moeglein & Krasner 2009) and to apply them to this massively parallel number of nodes that constitute a WSN. The premises are similar insofar as the acquisition functions and measurement functions are distributed among multiple physical entities, with widely different processing capabilities. The A-GPS server collects all satellite broadcast information, at the nominal rate of transmission from the satellites, and delivers it to the clients or sensors, which perform only measurement collection. The position can be computed back at the server. A distributed version of Relative Carrier Positioning whose accuracy is more appropriate for short distances usually found between nodes will be applied.

1.2 Technical Challenges

1.2.1 Low Power

In spite of the fast chipset technology progress that translates into even lower power consumption at each generation, the current claim for full processing power consumption for the best implementations is still at about 25 mW, with the current crop of chipsets at 65 nm including CMOS baseband and RF CMOS on the same die. This is still a few orders of magnitude larger than what is practically acceptable for a node architecture. One answer to this challenge is that a very large amount of GPS processing is absolutely the same for every node, a large amount is very similar, and a very limited amount is

unique to each node. At first glance, it looks obvious to confine high power common tasks to entities benefiting the most resources, and to relegate to the simplest node only the measurement collection, which is the only function that cannot be distributed.

This simplified view has to be somewhat corrected when one realizes that, in order to collect measurements, a single node still needs to acquire the satellites locally and therefore, as a minimum, needs to know the list of satellites in view at a given time, and their code offsets and Doppler as accurate as possible in terms relative to its own internal clock. The former calls for transmission and local reception of these data, that should be factored in the power budget, the latter imposing a relationship between the delivered data and the internal node clock. In a multi hop configuration, both problems are compounded (Ganeriwal et al 2003). The data source can be several hops away from the consuming entity, and requires reception-retransmission of data at each intermediate node, further impacting the power budget at every node. If implemented as a chain of individual synchronization operations between pairs of neighboring nodes, the synchronization across several hops accuracy will degrade with the number of hops. A successful solution will have to balance the saving of concentrating common processes with the cost of wirelessly exchanging the results. It will also hinge on the capability to achieve synchronization with bounded uncertainty from the gateway to any node at the periphery of the network.

1.2.2 Low Computing Capabilities

The processors or microcontrollers suited for low power also lack sophisticated computing capabilities. The most common limitation is a lack of "double" or even "float"

representation mathematical hardware support. It is customary to implement all satellite position computing functions in "double" representation, and very often the positioning algorithms are also implemented in "double" representation. One remedy would be to use a software "float or double" library, that would seriously impact the power and memory footprint, where a simple double mathematical operation has to be implemented in several hundreds of fixed accuracy arithmetic. A more radical solution, that has been done in the past is the implementation of positioning algorithms either in single precision float (Islam et al 2009) or even in 16 bits fixed arithmetic.

Another issue is the computation with sufficient accuracy of harmonic trigonometric functions.

1.2.3 Synchronization to GPS time

To elaborate on the need for accurate synchronization introduced in the low power section, carrier phase relative positioning requires relative data collection which is very precisely timed at both ends of the baseline, about a few microseconds in relative timing errors, and a few milliseconds in common timing error compared to absolute GPS time. In this situation where no code-based positioning is done at each node, the local clock error cannot be corrected as a byproduct of a preliminary code based positioning operation as it is customary in carrier phase relative positioning. The published WSN synchronization techniques are borderline in absolute accuracy (10 to 50 μs), and are not bounded after an unspecified number of hops. The success depends on the development of a global synchronization method that can be applied regardless of the number of hops. One has to keep in mind that, due to the global nature of the satellite transmission, any

couple of nodes in the WSN will receive a common set of satellites. This is the basis for a clock synchronization technique that is effective regardless of the distance between nodes and of the number of hops. Ambiguity Resolution of Time Integer (ARTI), will be introduced for this purpose. ARTI, as it will be demonstrated, can resolve a number of integer milliseconds of timing uncertainty between nodes, and can significantly relax the synchronicity of the measurements between nodes, up to a few 100 ms, or about two orders of magnitude beyond the traditional technique. The cumulative synchronization error resulting from a chain of individual synchronization operations then becomes acceptable.

1.3 Scope Of Research

This thesis introduces the Wireless Sensor Networks in the context of position, and explores the limitations of the earlier attempts at adding positioning capabilities to the architecture. These techniques use non-GNSS positioning solutions. After focusing on the shortcomings and describing how they compromise the overall mission of the WSN, GPS-based solutions that have been deployed in the field will be explored. Albeit they resolve the issue of precision and accuracy, they still fall short on power consumption, complexity and cost. The desirable characteristics of a more appropriate positioning method will be recalled, and then a new architecture combining WSN and GNSS architectures will be described. The main aspects of this new architecture will be explored one by one in depth.

1.4 Literature Review

The problem of relative positioning in wireless sensor networks is traditionally handled without GPS (termed GPS-free in the literature). Following the subdivision proposed by Boukerche (2007), the solutions can be characterized by the three components of distance/angle measurements, position computation, and localization algorithm. The problem, as formulated by Boukerche and a great number of researchers in the field, assumes few anchor nodes, knowing their own position by external means such as GPS, plus secondary nodes, of unknown location, collecting relative measurements between themselves and towards the anchor nodes. The problem is to locate the largest number of secondary nodes, from anchor nodes first, then from other secondary nodes that have been located relative to the first located anchor nodes. Savvides (2001) provides a similar view with more details on the localization algorithms. The distance measurement can be done by a direct Time of Arrival (ToA) or a Time Difference of Arrival (TDoA). The first method measures the time of flight of a signal in the medium, and converts it into distance by multiplying by the velocity in the medium. A popular solution for TDoA measurement is to use the very large propagation velocity difference between two types of signals, usually RF and acoustic waves (Priyantha 2000). Also Direct Spread Spectrum Sequences (DSSS) similar to GPS have been experimented with (Fu et al 2006). The Receiver Signal Strength Indicator (RSSI) can be exploited as a substitute for distance measurement as demonstrated by Whitehouse & Culler (2002). The measurement can be a carrier phase similar to GPS RTK techniques such as Radio Interferometric Geolocation (Marotti et al 2005). Distances are also linked with direct connectivity, where the

measurement is a binary information, below or above a distance threshold (Shang et al 2003, Shang et al 2004); the lack of accuracy is compensated by the multiple number of measurements. The Angle of Arrival (AoA) technique has been also proposed, for RF signals, but also for ultrasonic signals. The RF technique calls for either an antenna array, or for a rotating high gain antenna, and an absolute orientation measurement usually obtained with a magnetic compass. Niculescu & Nath (2003) claim that ultrasonic sensors can provide an angle of arrival accuracy in the order of 5 degrees, in a range of plus or minus 40 degrees. It is claimed in this dissertation that these techniques are difficult to deploy in the field, as their inherent high cost is compounded with an orientation measurement issue when the nodes are randomly dropped in the field. The second component to localization is the position computation method. It can use trilateration or multilateration (Fu 2006), when relative distances are only available, this being the GPS technique of choice. Triangulation is associated with AoA measurements (Niculescu & Nath 2003). Instead of using a circle of radius d, the Bounding Box technique (Simic & Sastry 2001) represents the distance to a neighboring node as a square of sides 2d, with the node at the center. Although less accurate than the circle model, it eliminates the need for floating operations, hence the need for float libraries. Ramadurai & Sichitiu (2003) proposed the probabilistic technique, where the probability density functions from multiple measurements are combined to deliver the most probable location of each node. In this approach, the inter-node distance of d is represented as a ring circling the known node at a distance d, with a Gaussian probability density distribution along the radius to represent the measurement uncertainty. The central position is a centroid of the positions of all nodes simultaneously heard by a given node

of unknown location. Doherty (2001) makes use only of the connectivity or reachability information and resolves the multiple position problem as a convex optimization. The localization component deals with the way to maximize the number of nodes finally located, with a given amount of data.

As a concluding comment, in multiple instances, techniques essential to GPS have been borrowed with mixed success, but no researcher has tried to use the GPS concept in its entirety to resolve the WSN localization problem. The opinion of the author is that researchers resisted using GPS, either due to lack of knowledge of its principles, or by fear of complexity overwhelming the limited node capabilities.

The basic GPS techniques used here are quite well known in general, and there is no need to delve too long on this subject. The expected accuracy is sub-centimetre relative positioning, which necessitates carrier phase processing. The usual approach is Carrier Phase Double Differencing. Among multiple possibilities, one popular method for Integer Ambiguity Resolution is the Least-squares AMBiguity Decorrelation Adjustment (LAMBDA) method (de Jonge & Tiberius 1995, de Jonge & Tiberius 1996a, de Jonge et al 1996b, Teunissen 2000) . For unit cost reasons and for a high number of nodes, the GPS receivers are usually single frequency in the present WSN applications considered. Since the geographic area is limited (less than 3 km), the ionospheric and tropospheric spatial dispersion should not be a problem. Still, ambiguity resolution can take up to 20 minutes in the conventional approach. Luo (2000) , and Luo and Lachapelle (2003) describe an interesting method about speeding-up the integer ambiguity fixing using ambiguity closure properties on a multi-platform configuration: the sum of the ambiguities on a closed geometrical path is always zero. The reported improvements on a

relatively moving platform case are in the order of 50-70%. Beyond ambiguity resolution verification, the closure properties can be also used to reduce the ambiguity search domain (Sun et al 1999). Another technique to speed ambiguity resolution is to use known coordinates or an average of the measurements before ambiguity resolution (only if the vibration amplitude is less than 5cm, a fraction of the L1 wavelength) (Cosser 2004). Even if the early solutions were using exclusively Dual Frequency receivers (Ashkenazi 1997), Cosser (2003) analyses the performance difference between single and double frequency receivers in Structural Health Monitoring (SHM) applications, with the conclusion that, when the ambiguities have been resolved, no significant performance difference can be demonstrated. Liu (2003) gives a good overview of strategies used for ambiguity resolution.

Cycle slips also need to be mentioned; this problem, which impacts the ability to provide centimetre accuracy continuously, as it destroys the resolved integer ambiguities, should be benign in the case at hand. The structures are usually outdoors, with high signal levels. Obstruction of some satellites in view is still a problem that has to be dealt with. The cycle slip problem will not be considered in this dissertation, first because the solutions are known, and second because the problem will be the same for any of the architectures that will be analyzed. The processing of the cycle slips will not be a differentiating parameter and it can be ignored in this context.

1.5 Thesis Outline

After the introduction in this chapter, a general description of the WSN architecture will be presented in Chapter 2. Its limitation regarding its suitability to positioning is explored, and some early conclusions are drawn. In particular, the claim that the traditional relative positioning architecture is highly inefficient for WSN will be demonstrated. The traditional positioning functions of Acquisition, Measurement Collection and Position computation are highly redundant among nodes, and it does not make sense to multiply the same functions again and again in all nodes, with the related waste in on-time or equivalently consumed power.

Chapter 3 poses the problem of synchronization between nodes, and how it affects the accuracy of carrier phase positioning. The adopted carrier positioning technique will be quickly introduced, with the appropriate references. The inter node synchronization realized with WSN time synchronization protocols will be shown insufficient for ambiguity resolution.

Chapter 4 proposes a solution to the synchronization problem developed in Chapter 3, the Ambiguity Resolution Time Integer (ARTI). It develops all the mathematical tools necessary for its application.

Chapter 5 focuses on the positioning aspects of the WA-GPS, with more details on the algorithmic section, with several variants on how to partition the position function between nodes. It also introduces the solutions that are considered for ambiguity resolution acceleration, taking advantage of the multiple cycles in the dense network, to

reduce the time to ambiguity resolution, reduce the powered on time, and overall improve the operational lifetime of the network.

Chapter 6 puts all pieces together, and compares the relative lifetime duration of the architectural variants proposed in Chapter 5 by simulating a network, its message exchanges, its computing tasks, while keeping track of the per node consumed energy.

A general conclusion will be offered in Chapter 7, with principal conclusions, and suggestions on how to expand on the material presented in this dissertation.

Appendices develop the details of the formulas used in ARTI.

CHAPTER 2

WIRELESS SENSOR NETWORKS

ARCHITECTURE

2.1 Wireless Sensor Network Architecture

Before delving into the improvements proposed for the location problem in wireless sensor networks, the architectures will be introduced first and the implementation challenges will be described in general terms, then in terms of position and location (Ilyas 2005a).

Compared to traditional networks, the sensor networks have very serious constraints in energy, in computation capabilities, in local storage, and wireless link interconnection (size of packets, and number of packets per second). The technologies deployed have significantly evolved over the years. The microcontrollers WSN applications use are usually based on Complementary metal–oxide–semiconductor (CMOS) technology that currently delivers the lowest power consumption. None of the nodes architectures are using the System on a Chip (SOC) concept, where all functions, including RF, are integrated into a single die or package. At the risk of rapid obsolescence, two architectures at both ends of the spectrum will be succinctly described. At the high end, a good current example is the SUN SPOT (SUNSPOT 2008) which has a 180 MHz, 32 bit ARM 920T CPU, 4 MB of Flash memory, 512 kB of RAM, and a 2.4 GHz IEEE

802.15.4 radio, and operates on a single 3.7 V rechargeable 750 mA.h Lithium-Ion Battery; the consumption is at 80 mA in active mode, calculating, but without radio transmission (to compare to 98 mA when transmitting), 33 µA in deep sleep mode. The programming is done on "Squawk", an embedded JAVA machine. At the low end, the MICA II from UC Berkeley (MICA 2008) will be mentioned, with a 7.37 MHz, 8 bit ATMEL CPU, with 128 kB Flash program, 4 kB of RAM, and 38.4 kbps, 868/916 MHz, Chipcon radio; it operates for more than 1 year on two AA batteries using sleep modes, at about 8 mA of total current in active mode and no radio (to compare to 27 mA when transmitting at highest power, and 10 mA when receiving only) and 15 µA in sleep mode. The software development platform is "Moteworks", based on the open source operating system "TinyOS" (TinyOS 2008). The low power wireless physical layer technology IEEE 802.15.4 running at 2.4 GHz with the CHIPCON radio chipset technology, is becoming mainstream for these applications, due to its widespread availability and low power consumption. For WSN applications, the ad-hoc mesh "Zigbee" protocol (ZigBee 2008) is usually not deployed on top of the IEEE 802.14.5 physical layer; its complexity and extremely large number of nodes it can support is beyond the capabilities of the WSN.

2.1.1 General architecture

In its most comprehensive implementation, a WSN functional architecture is composed of three layers, a mote or node layer, comprising all the low cost nodes, the server layer, and the client layer (see Figure 1). Data collection and wireless forwarding take place in

the node layer. Data aggregation and storage are implemented in the server layer. The client layer deals with the data extraction as reports and presentation to the end user.

The node layer is the one actually deployed in the field, and it is the most critical as all the architecture limitations mentioned earlier are actually relevant only to this layer. There are "regular" nodes that wirelessly interface between themselves and the gateway and the "gateway" node that assures the interface with the Server Layer.

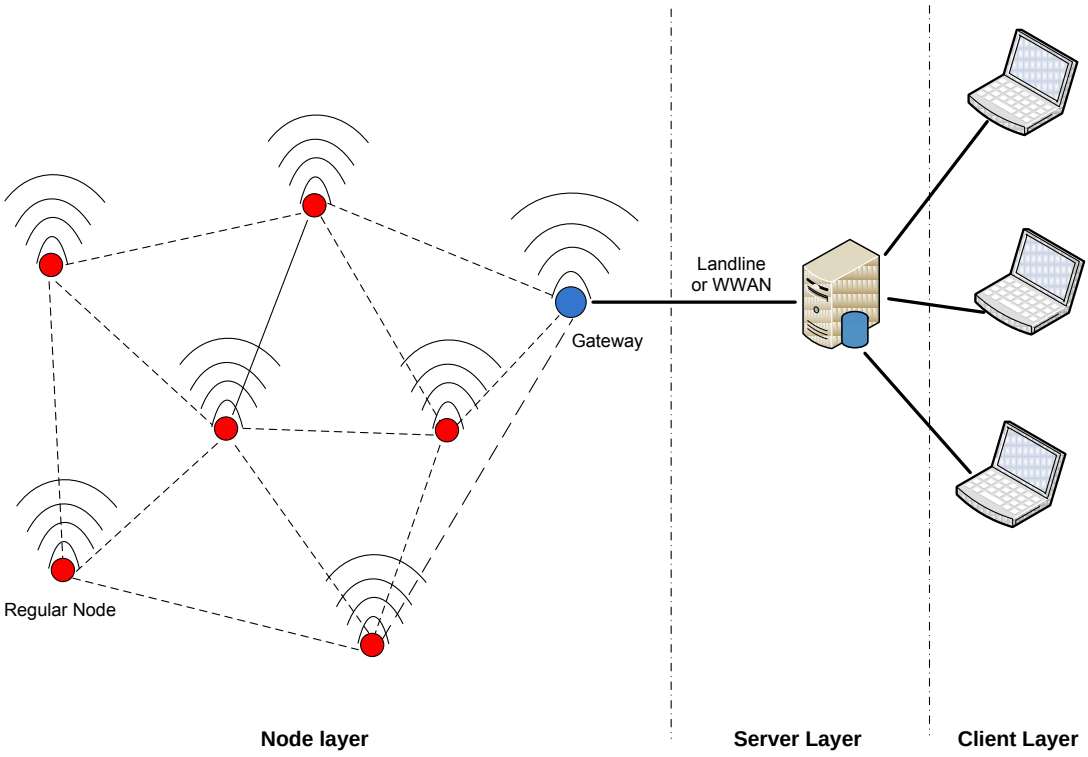


Figure 1 - Wireless Sensor Network Functional Architecture

The server layer is usually implemented in the "cloud", has no limitations of energy and processing power as it is directly connected to the power grid, and comprises a networked cluster of desktops or laptops continuously powered on, with very large storage capabilities and processing power. The server layer does the last part of data fusion

processing that did not take place inside the node layer because the energy budget was less impacted by forwarding the data in their raw form or semi-fused form, rather than doing the full data fusion "in situ". It will also store massive amounts of data collected over weeks, months or years, and is a large data repository. The final element of the chain is the client layer, which is usually a thin software layer residing in a laptop or workstation, that presents the data in a graphical or tabular form desired by the end user, coming from a report generated by a query to the server database. With all functional architectures, these functions can be differently aggregated in different physical units or entities, for convenience. For example, the gateway node, the concentration point of all data in the node layer, can be connected to the power grid, can be housed in a weatherproof shed, and can possibly support the storage and processing function.

In its traditional approach, WSN terminology only conjures up the node layer, but this concept in meaningless without the other layers.

2.1.2 Gateway or Sink Node Architecture

There is a privileged node called the gateway. It is the interface of the WSN with the external world. On one side, it implements the WSN wireless radio protocol as the root node, and on the other side, it is connected to the "cloud", either by phone landline, or by WWAN (CDMA, WCDMA, or GSM network). Because it concentrates all data traffic coming out of the network, and is connected to multiple nodes, it has to process a much larger data flow. Most queries to the network are generated from the gateway, or forwarded by the gateway. It has possibly some local processing and data fusion capabilities, as well as storing capabilities. It also needs even more power to forward the

data by radio transmission through a cellular network or a phone landline. For all these reasons, the gateway is usually a more powerful node, connected to a large power supply, or even connected to the power grid. Its reliability is also of concern. Any failure of this concentration node will render the WSN unreachable. This issue is mitigated either by internal redundancy (parallel function redundancy in the node), or by better protection of the node (shed), or even by external redundancy (multiple gateways), or by a combination of all. The SUNSPOT node, provided it is compatible with the radio in the other nodes, is a good example of this implementation.

On top of the special functions, the physical gateway can play the role of a regular node, collecting data as any other regular node.

2.1.3 Regular node Architecture

The node or mote (or even dust) is the lowest level in the WSN architecture. It is the one deployed in the largest numbers, and the most constrained in resources. The computing power is so limited that usually, no attempt to embed a full GPS or GNSS function in it is ever made. This is where the data collection takes place. It can be temperature, sound information (for localization of intruders, or for gunshot detection). It is usually designed for no maintenance. The embedded energy available at the deployment is usually not renewed for the whole lifetime of the node, and is not serviced in case of failure. The wireless protocols need to be very robust to the loss of any of the nodes, either by power depletion, or by failure mechanism. It is usually an ad-hoc protocol, where any transmission between nodes needs to be preceded by a discovery phase where all nodes identify the nearest neighbors with which they have direct connection, and a routing

phase, where the multi hop optimum route from one node to another is determined. It is important to note this optimum routing is not necessarily fixed over the total lifetime of the network. Every time a node in the route gets low in energy, the routing algorithm is run another time to avoid the weak node in the route.

Some other techniques are also used to extend the duration of the network, for example by introducing a clustered or hierarchical architecture. One node in the neighborhood is designated as the "cluster head", and usually aggregates all messages from all neighbors, and forwards them to the next level. It can also support some local extra functions that are not found in the "leaf" nodes (i.e. the lowest level). The cluster head is practically the same hardware and firmware than all the other nodes, but is elected for this function. The cluster head drains more power than the neighbors because of more transmission and processing activity, and, starting from the same energy budget, it will be depleted earlier. In this case, another cluster head is designated to distribute as evenly as possible the total processing burden. Figures 2 and 3 illustrate an example of routing reconfigurations after role swapping between an old cluster head becoming a regular node and a regular node morphing into a new cluster head to service the traffic.

2.1.4 Routing Protocols

One of the classical techniques for network discovery is the "flooding" algorithm, quite suitable for single gateway, multiple sensors configuration dealt with here. The method consists of broadcasting a probe routing message tagged with the identification of the sender node. Each node in the vicinity that receives the probe routing message will first update their routing table memorizing that they have direct connectivity with node one,

then it rebroadcasts the same message by appending their own identification to the message. The following wave of nodes receiving this second level message will memorize their connectivity with first node in two hops, then direct connectivity with the second wave of nodes in direct connection. The initial single message triggers a flow of

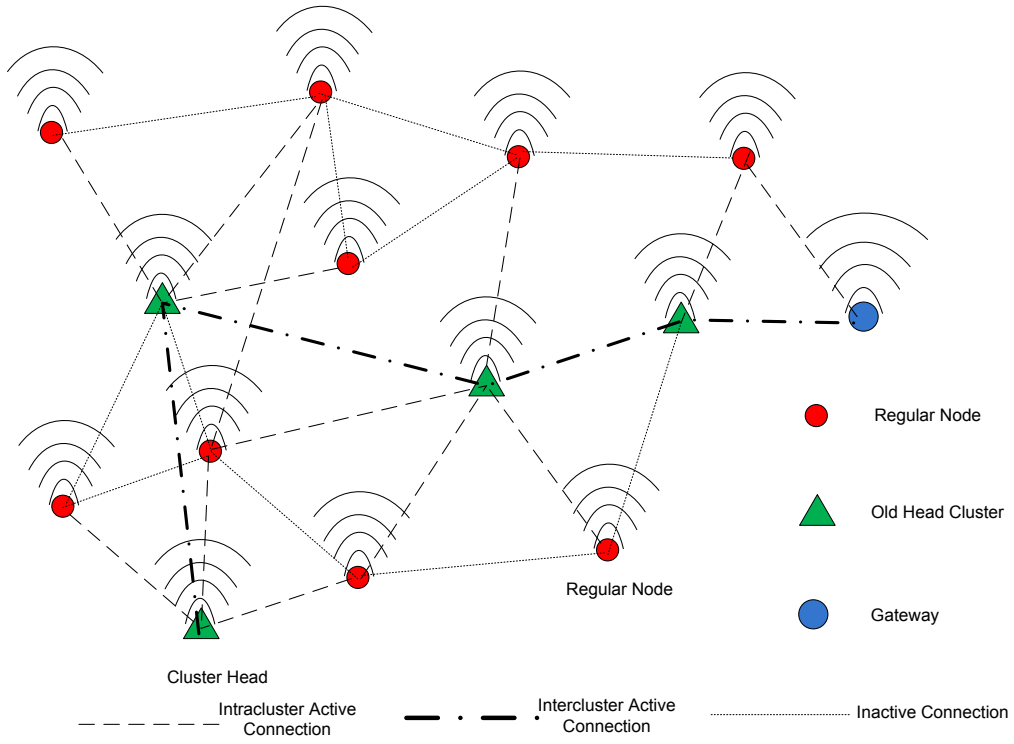


Figure 2 - Clustered WSN before reconfiguration

messages that reverberate throughout the whole network, hence the name of the technique. The usual protocol is to generate the first probe message from the gateway, and each node eventually will know that they have a connection with the gateway (or first node), to which immediate node to send any message addressed to the gateway, and even which is the optimal route (the one with the minimum number of hops). It even can contribute to the load balancing goal by varying routes from message to message.

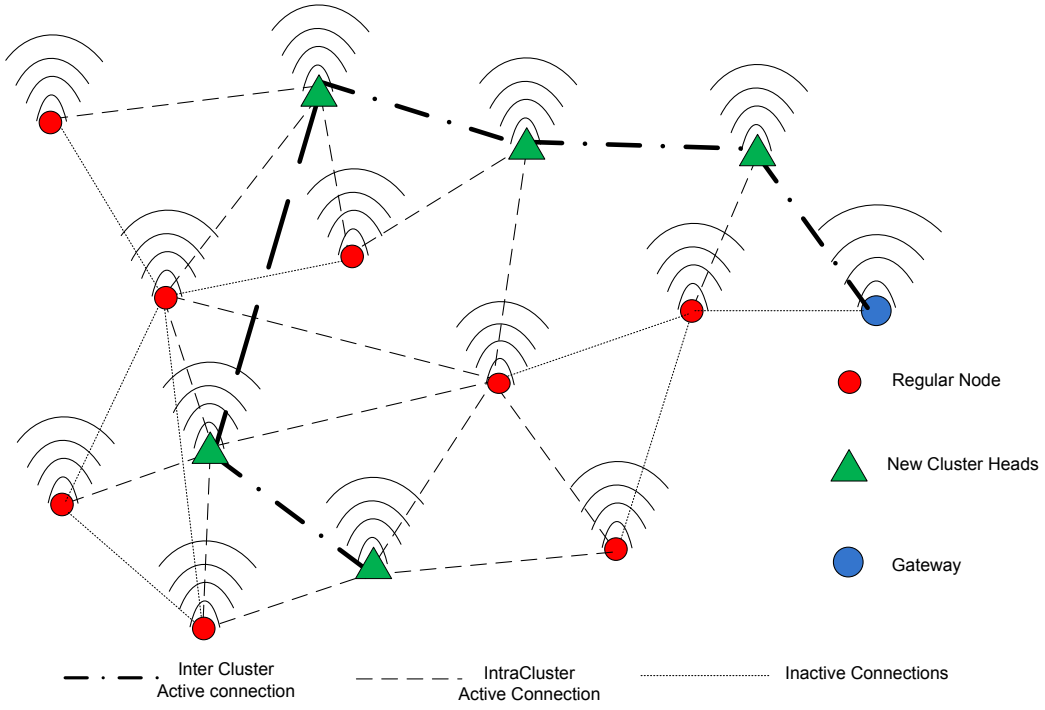


Figure 3 - Clustered WSN after reconfiguration

2.1.5 Power consumption Considerations

From the advertised lifetime of a MICA II of about one year without changing the batteries, and total capacity of AA alkaline batteries at 2890 mA.h, one can infer than the expected average current consumption is about 0.33 mA at 3 V or equivalently, the active/sleep duty cycle is 4 % or about 1 s of activity every 25 s.

Assuming the same duty cycle of 4 % for the high end SUNSPOT architecture, one ends up with an average consumption of 3.2 mA at 3.7 V, or a total lifetime of 227 hours or about 10 days.

It should be obvious now that both architectures fulfill radically different needs, and that only the MICA II or a similar architecture is suitable for structural monitoring, given its lifetime duration without having to replace the batteries.

Before leaving this critical topic, the high energy cost of wireless transmission will be emphasized. Focusing again on the MICA II, the extra power consumption in reception mode only is about 0.25 times the fully active consumption without a radio, and about 2.25 times in radio transmitting mode.

The cost of wireless transmission per bit is:

$$\frac{19.10^{-3} \times 3}{38.4\ 10^{3}} = 1.5\mu J/\text{bit} \tag{2.4}$$

These numbers deal with the actual energy drawn from the battery by the radio chip, and not with the radiated transmitted power that will be always significantly less. The announced transmitted power of +5 dBm is equivalent to a linear power of 3.2 mW or 80 pJ/bit.

2.2 Review of Positioning methods

Why Position is so important in WSN

The fundamental intent of a WSN is to provide a triplet of information, namely a measurement, such as temperature, a location, where the measurement was taken, and finally the time of the measurement.

A method to derive position of each node and a way to synchronize their internal clock for further synchronized time sampling is thus necessary for the optimal use of the measurements. The position accuracy requirements can vary greatly from 100 m down to few centimetres, absolute in geographic coordinates or relative compared to a reference node, oriented or not vs. the local North direction. The timing requirements can vary between hours down to microseconds.

Besides its use to fulfill the goals of the WSN, position can be used internally to the WSN to improve the routing of multi hop messages, in particular by directing messages towards the known position of the receiving node, and avoiding to waste energy on the nodes which do not need to participate in the routing. It can be also used to do load balancing, to even the resource drains when the message traffic volume is too large for a single string of nodes, and a second route needs to be opened to expedite the traffic; the knowledge of the relative location of the source and sink node, and of the nodes in between facilitates the identification of the second shortest path. The same information can also be used to identify more alternate routes, for distributing the power drain on the largest number of nodes, and avoid the premature loss of connectivity when some nodes at the centre of the network acting as a hub are energy depleted.

The knowledge of timing, at least relative between nodes allows further energy savings by cycling through wake and sleep periods, and making sure that the neighbor nodes are waking up simultaneously, allowing two-way communications.

General approach to positioning

As they are mainly for reference comparison with the proposed new solution, some of the high level ideas in this section will be paraphrased from the reference (Ilyas 2005a). First of all, the location can be a-priori known at the deployment phase of the WSN. The study will be limited to the ad-hoc deployment where the nodes are randomly located, with a density sufficient to allow some redundancy in the communications. The discussion will be limited to the static or pseudo-static cases believed to cover all structural monitoring use cases.

One can also imagine to have only few well located reference nodes in the network. They could be on the periphery or randomly distributed throughout the whole network. Such a situation would arise, even because only a few points are fitted with dedicated location capabilities for cost reasons, or because some of the nodes are not in the right environment, for example poor GPS reception. Usually the other nodes are located relative to these reference nodes. The location accuracy of the secondary nodes will heavily depend on the number of hops to the closest reference node, and the ratio of reference nodes vs. total number of nodes.

The localization algorithm can be centralized, where all individual measurements are forwarded to a central computing entity, where global minimization can be attempted for better overall accuracy, using Maximum Likelihood techniques. However, it has some serious drawbacks, such as the necessity to traverse the network with large amounts of measurement data, and then to redistribute the found locations to each node for local processing such as routing. The search for a solution will be limited to local methods, involving the minimum number of nodes, with a location algorithm also locally implemented. This decision has the obvious advantage of eliminating the large amounts

of transmitted data that are one of the most energy costly functions in the WSN, and will facilitate the overall robustness, i.e. the possibility to survive to the loss of some nodes.

Some measurement techniques that have been deployed or intended for WSN will be reviewed, followed by the algorithms designed for local positioning, then for global positioning.

Measurements

Proximity

The only information between two nodes is that they are in radio visibility, therefore, their distance cannot exceed a maximum, depending on the antennas gain (transmitting and receiving), transmitted power, propagation attenuation, and sensitivity of the receiver. Several radio technologies will fall into this category, WiFi, ZigBee or RFID tags.

Distance

A whole class of measurements is related to the direct or indirect distance measurement between nodes that are in radio communication.

Signal Strength

There is a relationship between, transmitted power, received power and relative node distance (Krish 2005):

$$P_{r,dB}(d) = P_{r,dB}(d_0) - \eta \cdot 10 \cdot log\left(\frac{d}{d_0}\right) + X_{\sigma,dB} \qquad (2.5)$$

with $P_{r,dB}(d_0)$ being the received power in dB at reference distance d_0, η the path loss exponent, and $X_{\sigma,dB}$ a normal variable in dB units (or equivalently log normal in power units), or standard deviation σ, that represents the short term signal fluctuations around

the average value given by the rest of the formula. It has to be mentioned that η, the path loss exponent, can be anywhere between 2 (free space propagation) to 4 (heavy indoor environment). This formula implicitly includes the transmitter and receiver antenna gains.

If some estimate of the path loss exponent is made, one can convert the received power in an equivalent distance. This indirect distance measurement is quite imprecise, but can be sufficient to provide min-max bounds for the distance. This can be sufficient for applying a constraint based position algorithm (discussed later).

Time of arrival

The distance between nodes is directly proportional to the time the signal takes to propagate from the transmitting node to the receiving node. This proportionality constant is obviously the velocity of the signal in the propagation medium. The signal can be Radio Frequency, with a velocity close to 300,000 km/s or ultrasonic or acoustic with a velocity of about 340 m/s in the air.

For reference, the velocity in air depends on the temperature according to the formula:

$$c_{air} = 331.3\sqrt{1 + \frac{\theta}{273.15}} \approx 331.3 + 0.6064\,.\,\theta \qquad (2.6)$$

where

c_{air} velocity of sound in the air (m/s)

θ air temperature in degrees Celsius

The variation of velocity is about 10% between -25ºC and +25ºC, that should be corrected with a measurement of the ambient temperature (Wikipedia 2010).

The Time of Arrival requires a measurement of the transmit time at the transmitter node and of the receive time at the receiver node using synchronized clocks. For a distance of 10 metres, the propagation time is 33 ns for RF and 30 ms for acoustic waves. Mutual synchronization is straightforward for the acoustic case, but difficult for the RF case.

Time-difference of arrival

To get around the synchronization problem, the time difference of arrival has been proposed, measuring the difference of arrival between two signals transmitted at the same time, but received at widely different times. The first signal is usually RF, the second one is acoustic.

Position computation techniques

Based on proximity information

Centroid based

If there are enough nodes with known position in the immediate neighbourhood, the first possibility is to compute the centroid of the locations of the received stations, in x and y coordinates, under the loose assumption that the node to locate is somewhere "in the middle" of the received stations. If the 2D location of the i^{th} node out of the known n nodes is (x_i, y_i), the location of the unknown node is:

$$x_u = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i \, , y_u = \frac{1}{n} \cdot \sum_{i=1}^{n} y_i \tag{2.7}$$

From Krish (2005), if the ratio $\frac{R}{d}$ with R maximum radio range, and d average distance between nodes is increased from 1 to 4, the RMS location error goes from $0.5 \cdot d$ to $0.25 \cdot d$.

Intersection of coverage areas

If the coverage areas of each received node are not identical, one can estimate the location of the unknown node to be in the intersection of all coverage areas. The most probable is the centroid of this overlapping area (see Figure 4).

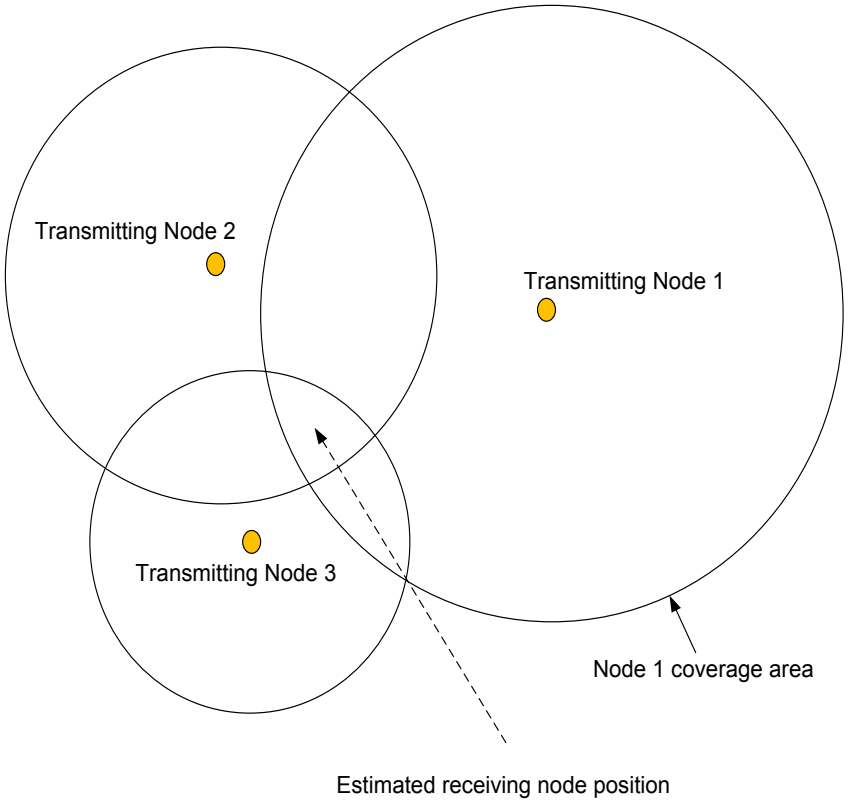


Figure 4 - Position computation by intersection of coverage areas

The drawback of such techniques is that the coverage area shape and size is not so well defined, varies depending on external factors, such as environment, geometric shadowing, that cannot be taken into account in the simple algorithm, if the topography of the location is not known, and the location of the unknown node is not known either.

Based on relative distances

Another related technique, proposed by Intel Research group mainly for WiFi 802.11 nodes but also applicable to the WSN (LaMarca 2005), is based on collaborative local computations. First, the nodes are assigned the location of the nearest reference node (in the minimum number of hops sense). The algorithm goes iteratively in a random fashion around all non reference nodes, and minimizes the node error by modifying its location that delivers the lowest error. The node error is the sum of the square of all distances of this node with all other nodes with radio connectivity.

Based on Geometric Constraints

If some nodes are in radio contact with too few nodes with known positions for a direct position computation, or equivalently, several nodes are connected to the nearest reference node (in number of radio hops) through more than one hop, the earlier solutions do not work. The relative location of each node in between can be still found by imposing node to node distance constraints (Biswas 2004). This can work for proximity as well as direct distance measurement cases. A first known method is the MDS-MAP (Multidimensional scaling), (Shang 2003), where only the proximity (i.e. connectivity) is

used. It starts by estimating the shortest distance between any pair of nodes, expressed in the shortest number of hops. Then a method borrowed from mathematical psychology does derive node locations from node distances. Finally, the node locations are adjusted to take into account the reference node positions. An accuracy of about 0.5 the average node distance is achieved if 0.2 % of nodes have known positions, and the connectivity is about 12 (average number of nodes directly heard from any node). Intuitively, one expects the accuracy expressed in radio range units to degrade when the connectivity decreases, thus providing less geometric constraints. If no reference node were available, the method would suffer from a rotation and flipping indetermination. In two dimensions, with only three nodes with known locations, these indeterminations are removed. Also, if a more accurate distance between some nodes is known, the method can take this into account. It is also worth mentioning that this method works well when the known nodes are located at the periphery of the convex hull represented by all the nodes. If the reference nodes are within the cloud, the location degradation becomes quite severe, and the location of the outer nodes has tendency to collapse towards the center of the cloud. This effect is similar to DOP degradation in GPS. This method requires $\mathcal{O}(n^3)$ time for a network of n nodes. It is also a centralized technique to the extent that all measurements need to be available at a single site to run the algorithm.

Another concept uses the Semi-Definite Positive algorithm (Doherty 2001), and the node connectivity. Known peer-to-peer communication in the network is modeled as a set of geometric constraints on the node positions, i.e. the distance between communicating nodes cannot be larger than the maximum radio range. Provided that the constraints are

tight enough, a simulation illustrates that this estimate becomes close to the actual node positions.

2.3 GNSS positioning Methods

Rather than describing a very traditional view of the GPS technology, this section is aimed at pinpointing the challenges that GPS raises when integrated in a WSN during all phases of the positioning process. All the surveyed WSN literature that mentions GPS is describing an all-in-one black-box solution that usually collocates commercial standalone GPS receivers to a few strategically chosen nodes, usually the root node, and some nodes on the boundary of the network. The availability of the accurate GPS time is mentioned quite often in this configuration, along with its ability to provide a primary absolute synchronization means by ways of 1-PPS hardware line down to an accuracy of 200 ns. The GPS receiver is qualified as costly and very power hungry (Ganeriwal 2003). Its cited accuracy is in the range of a few metres, that implies a single code based positioning technique. Given the expected high cost of the GPS solution, quite a few papers explore ways to determine the positions of the non-GPS equipped nodes from the known positions of the GPS equipped ones. Some non-GPS positioning solutions claim relative positioning accuracies in the order of few centimetres. Apart from a very few cases, these techniques merely measure Euclidian distance between nodes, and not an oriented vector baseline. All these measurements need to be converted into a 2D or 3D network as a distinct operation. The vertical accuracy is very poor, as all nodes as roughly located in a plane. Even more interesting, an author reinvents from scratch the

concept of the Double Differencing between beat frequency phases using radio interferometry and beat frequency between two senders and two receivers (Maroti 2005), with a claimed accuracy of 3 cm at distances up to 160 metres. All these partially integrated GPS architectures are highly inefficient. They do not take into account the close proximity of these GPS receivers that is conducive of a massively aided architecture on all phases of the positioning elaboration. Obviously, no serious attempt has been made to tightly integrate the GPS function with the WSN.

2.3.1 Acquisition

The energy cost of searching for satellites is very high as every code offset and every carrier frequency bin needs to be searched. For a maximum straight distance of 5 km, the most extreme code offset difference between nodes is no more than 17 C/A code chips.

If the node local clocks are synchronized in offset and drift, the successful code offset and frequency bin can be transmitted from one node to any other node. The search per node is then limited to the range in frequency and code offset to cover the positioning uncertainty.

2.3.2 Navigation messages capture

Obviously, given the small size of the network, the satellite visibility will be virtually the same in all points of the WSN. With the assumptions of an ephemeris update every 2 hours and of an automatic receiver wakeup at update time, the ephemeris collection effort is estimated at about 30 seconds at full GPS receiver power every 2 hours, and hence imposes a 0.42 % power duty cycle. With an assumption of 25 mW at full power for the

best available low cost GPS receivers, the consumption is about the same as the full MICA II node without radio data transmission. If the GPS section was the only one to draw energy during data collection, at the rate of 30 seconds every 2 hours, and would share the MICA II battery, this mode would deplete the standard MICA II battery of a single node in 3468 days.

$$t_{life}(days) = \frac{2.890(A.h)\times 3(V)}{30(s)/3600(s/h)\times 25.10^{-3}(W)} \times \frac{2(h)}{24(h/d)} = \quad (2.8)$$

$$3468\ days$$

From a WSN marginal only energy perspective, the two extremes will be now examined, either each node has its own GPS receiver and collects its own ephemeris locally, or there is a central GPS receiver at the root node that collect all of them and retransmits wirelessly the data to all nodes.

To draw a first conclusion, the energy cost per node required for transmission still has to be estimated. To begin the analysis, one notes that the same data will be forwarded over and over from each node to each other node. There is no data aggregation taking place, and the exact same volume of data will be transmitted by the first and the last node; it can be safely assumed that the energy cost per node per ephemeris is independent of the distance of its node to the root node . If the assumption of 200 Bytes per satellite and per ephemeris is made (about 10% larger than the raw compacted data found in the navigation message), or 2000 Bytes for 10 visible satellites, the transmission duration of a single message including all collected ephemeris is:

$$MsgDuration = \frac{2000\ (Bytes) \times 8(bits/Byte)}{38.4.10^3(bits/s)} = 0.4167\ (s) \tag{2.9}$$

With a conservative assumption of a new set of ephemeris every 2 hours, every node would be theoretically depleted after 260100 days:

$$t_{life}(days) = \frac{2.890(A.h) \times 3(V)}{MsgDuration(s)/3600(s/h) \times 8.10^{-3}(A) \times 3(V)} \times \frac{2(h)}{24\left(\frac{h}{d}\right)} = 260100(days) \tag{2.10}$$

In a traditional architecture, with every node doing their own ephemeris data collection, the theoretical per-node marginal lifetime (or lifetime without any other energy draw) will be 3468 days. With a root node ephemeris data collection with no power consumption limitation, and dissemination by wireless messages the per-node marginal lifetime is 260100 days. This lifetime duration ratio of 72 between 30 seconds of GPS data collection (GPS section only powered on) and 0.42 seconds of data reception (node processor and wireless section only powered on) every two hours strongly favors the ephemeris distribution through wireless.

2.3.3 Measurement/Tracking

It is not the intent to propose a detailed analysis of the methods that would be suitable for code and carrier phase tracking in a node. This would certainly require a full study.

However, some ideas on what are the implementation hurdles to overcome and how this could be done, will be developed. The first remark is about the periodicity of the measurements. The increase of the interval between carrier phase measurements is beneficial to the ambiguity resolution, by the mere fact that the satellite geometry will have changed more. It is not really a viable option as carrier phase measurements have to be collected continuously over the period to avoid cycle slips. The GPS measurement functions will need to stay powered on for a longer period of time, thus impacting the energy budget. The position computation and the measurement aggregation can be done at larger intervals, as each computation attempt also impacts the energy budget, by wireless measurement exchanges and by numerical processing. These two design parameters, the total duration of data collection per fix, and the measurement reporting interval (the same as the ambiguity resolution attempt interval) are somewhat decorrelated, and should be tuned independently. The first parameter depends on the geometry of the network and the average inter node distance. The second is an energy budget optimization process, where the chance to fix the ambiguities earlier by accelerating the frequency of the attempts (and going back to sleep mode earlier) is balanced by the energy spent to try that is directly proportional to the frequency of the attempts.

The requirement of measurement continuity also prevents the implementation of raw data buffering strategies, so called "Store and Process", at least for non-contiguous or overlapping buffering periods. FFT based tracking techniques will exhibit low accuracy and are unsuitable for carrier phase. A traditional simple loop per satellite tracking structure is probably the most efficient in terms of power consumption. This is true only

if the long search in the code and Doppler domains is drastically reduced by very accurate acquisition aiding. This aiding should come from the gateway, and this again advocates for a very accurate synchronization of each node to the GPS clock. The nodes do not have the complexity for resolving the bit transitions, collecting and decoding the ephemeris, so the reported measurements are only L1 cumulated carrier phase.

Synchronization to GPS time

The well known technique of GPS receiver clock calibration by usage of minimum four measurements, and the introduction of an extra clock bias parameter in the position solution is not an option in this architecture. No code pseudorange is measured at the nodes so another synchronization technique is needed.

Synchronization between nodes

At least for the setup of the network, and in order to acquire satellites for the first time using the gateway aiding information, each node clock needs to be roughly synchronized to the GPS time. The only solution is to propagate the GPS time from node to node starting at the gateway that is assumed synchronized in the way described in the previous section. The existing synchronization solutions that are candidates for this important function will be reviewed now.

2.3.4 Position Computation

Given the sub centimetre expected accuracy, a carrier phase technique is needed. As it will be outlined in Section 2.4, none of the WSN synchronization techniques are accurate enough for inter node relative synchronization at the needed level of accuracy for correct carrier phase operation as will be demonstrated in Chapter 3. The category of solutions

envisioned to evade this apparent impossibility was to reintroduce, in a carrier phase relative positioning algorithm, a form of receiver clock bias error determination akin to the four satellite solution so pervasive in the code phase single positioning algorithms. This solution had to be robust enough to converge at the first invocation time tolerating a very loose synchronization accuracy as achievable with WSN standard techniques, and yet to improve the overall synchronization for a tight "sleep" and "powered on" duty cycle synchronization.

2.4 Review of the WSN synchronization methods

To operate at expected performance, the traditional Relative Carrier Phase solution, as will be shown in Chapter 4, requires a good absolute synchronization vs. GPS, and with a synchronization error less than 5 microseconds at any other end of each baseline, even with multi hop connectivity. This section will show than none of the known WSN synchronization techniques is suitable regarding these two criteria. They only provide relative or local synchronization. Their achievable worst case accuracy is 10 μs to 100 μs per hop. This important fact has been the motivation behind the search for a viable solution, and led to the introduction of the main contribution of this dissertation, the WA-GPS. This contribution is at once a relative position technique, and a very accurate inter node time synchronization between neighboring (shortest distance in hop number is only one hop) and distant nodes (shortest distance at more than one hop).

Three of the most popular WSN synchronization methods will be introduced in their original formulation, which is believed to span a good sampling of all methods available

and will provide an overview of their absolute and compared performance. The motivation is multiple. First, WA-GPS still requires some relaxed synchronization method for initial coarse synchronization, which will be chosen among these; a good understanding of the logic and concepts is needed here. Second, this description will provide a benchmark performance for comparison with WA-GPS. All methods introduced here, namely the "Reference Broadcast Synchronization" (RBS) (Elson 2003), the "Timing-sync Protocol for Sensor Networks" (TPSN) (Ganeriwal 2003) and the “Flooding Time Synchronization Protocol” (FTSP) (Maroti 2004), are local pair-wise synchronization schemes. They differ in the fact that the first is a receiver-receiver technique, the other ones being a transmitter-receiver technique. This distinction alludes to the fact that RBS synchronizes two receiving nodes listening to the same sender under the assumption that reception at both nodes is simultaneous (true if one makes abstraction of the propagation time difference, that will be usually negligible). TPSN executes a round trip time measurement, where both involved nodes are taking turns at sending and receiving. FTSP is based on a single forward message.

To support the upcoming descriptions, some terminology is first introduced that will be used throughout this section. According to Ganeriwal (2003), the delays experienced by the packet during a transmission reception sequence can be generically decomposed as follows,

Send time, when the decision to transmit a packet is scheduled as a task, the packet is constructed at the application layer, and passed to the Medium Access Control (MAC) layer,

Access Time, when the packet waits for the channel to be available for transmission.,

Transmission Time, when the packet is transmitted bit by bit on the wireless medium,

Propagation time, taken by the packet to travel from sender to receiver,

Reception Time, or time to receive bit by bit and pass them to the MAC layer,

Receive time, where the packet is reconstructed from the bits, the packet is passed to the application layer, and the packet is decoded. This time is highly variable, because it depends on the task scheduling of the receiving node.

The following sections have been adapted from Savaranos (2006a), especially the advertised relative performance.

2.4.1 Reference Broadcast Synchronization (RBS)

The following description refers to Figure 5. A sender broadcasts a beacon message that does not include any time tag. This beacon defines a common implicit timing reference at the sender antenna that will be simultaneously received at both receiver antennas, if one abstracts the propagation time. Each receiver time tags this reception with its own local clock. Both receivers then exchange their local time tags within messages. Each receiver then combines its observed time tag and the received one, and calculates its own local clock offset compared to the other local clock.

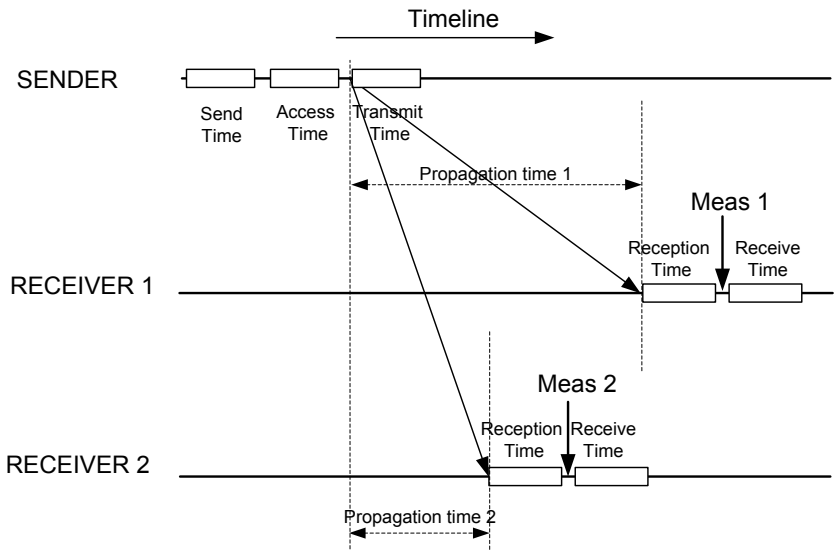


Figure 5 - RBS Timing Sequence

This technique claims to eliminate two of the largest sources of indetermination, “send time” and “access time”, as well as “receive time”. The remaining reception time is the time the radio chipset takes to receive the message, and to send an interrupt to the host computer. The reception time from a particular bit in the message to the interrupt event is usually very predictable, because this sequence is implemented in hardware in the radio chip. If the interrupts have not been masked, the extra time for the host computer to acknowledge the interrupt, to save the current process context, to start the interrupt processing code, and to read the local clock is variable but is usually below 2 μs. If the radio chipsets in receivers 1 and 2 are from the same manufacturer, they will respond very similarly, and all the implementation peculiarities will be eliminated in the time difference. According to the authors, the timing noise is very close to Gaussian, with 0 s mean and 11 μs standard deviation, with maximum error in the order of one transmission bit duration or 53.4 μs. An accuracy improvement suggestion is to send multiple messages and do compute the mean value of all time differences. This well-behaved

statistics allows reduction of the timing error as the square root of the number of messages, a number of reduction of standard deviation from 11 μs to 1.6 μs after 30 broadcast messages, or a full power sequence of up to 60 seconds. The clock skew (only the difference of R1 and R2 clock skews to be correct) can be estimated by curve fitting the sequence of time differences to a first order correction formula. All these improvements will not be an option for this application, as they require too much energy. The temporary conclusion is that the performance using strictly standard or commodity hardware in a "black box" fashion is remarkable, and that the authors are fully aware of the performance improvement they would achieve with time tagging or "application collapsed into the MAC" as the paper refers to it.

2.4.2 Timing-sync Protocol for Sensor Networks (TPSN)

This is an example of the round time delay technique applied to the WSN. The algorithm starts with a discovery phase where each node gets a rank. The root node is arbitrarily assigned the level 0, and sends a "level_discovery" packet, including the level of the transmitter (in this case 0); it finally becomes insensitive to any received "level_discovery" packet. Every node receiving this packet assigns to itself a level that is the level received in the "level_discovery" packet plus one. Then they retransmit another "level_discovery" packet, but with their own level, then become insensitive to any other "level_discovery" packet they might receive back from the next level. The net result is that every node will have a level or rank, laid out in rough circles "annealed" around the root node. The next step or synchronization step will be done between neighboring nodes differing only of one unit in level. The next step is for the root node to initiate the

synchronization cascade, by sending a "time-sync" packet, that will be received by all level one nodes. Each level one node then sends a "synchronization_pulse" packet to the root node that will send back an "acknowledge" packet as described in more detail below; it then sets its own clock onto the root node clock. To avoid medium contention (i.e. packet collision at the transmission), each level one node transmits its request after a random delay time, and backs up randomly again if any packet collision has been detected. The processes is reiterated at level two nodes, as soon as they detect that the synchronization has taken place between level zero and one by monitoring the activity of their nearest level one. To complete the overall description before moving on to the basic synchronization mechanism, if a node joins the network after the sync sequence, it sends a "level_request" message, that will be answered by all nodes in the immediate vicinity, with their own level. The newly joined node assigns to itself the lowest level as its own.

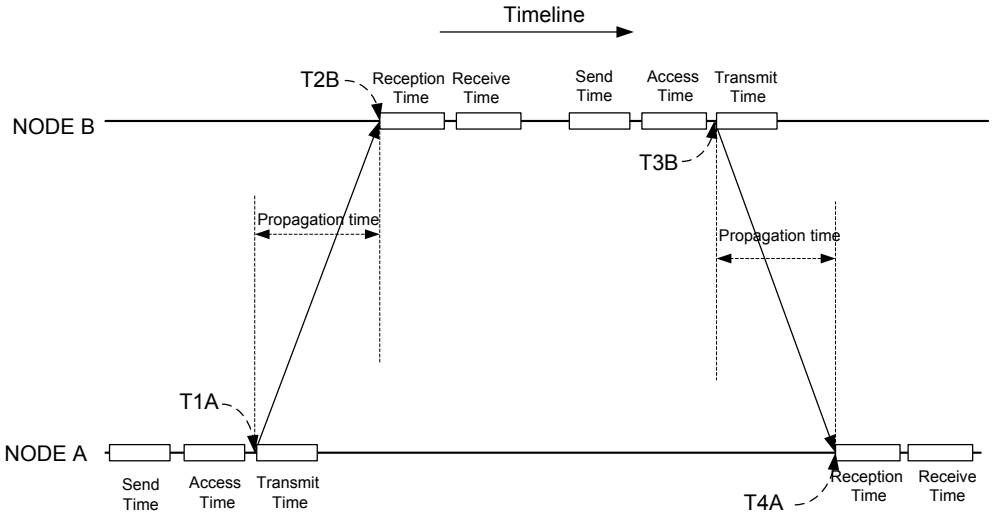


Figure 6 - TPSN Timing Sequence

As shown in Figure 6, the node A transmits a "synchronization_pulse" message at T1 (referred to node A local clock); node B receives the message at T2 (referred to node B local clock). After some processing time node B replies with an "Acknowledge" message at time T3 (referred to node B local clock). Finally, the node A receives the "Acknowledge" message at T4 (referred to its own local clock).

The values T2 and T3 (or their difference) are finally transmitted from node B to node A, either in the "Acknowledge" message, or a further message associated with this synchronization sequence.

Node A can derive the inter node clock offset Δ and the propagation delay d:

$$\Delta = \frac{(T_2 - T_1) - (T_4 - T_3)}{2} \tag{2.11}$$

$$d = \frac{(T_2 - T_1) + (T_4 - T_3)}{2} \tag{2.12}$$

It is worthy to note that the scheme is working, even if T1and T4 are measured against a time reference system (node A clock) and T2 and T3 are measured against another time reference system (node B clock).

Besides the "Propagation Time", which is the goal of the measurement and very short compared to all the other times, and the "Transmission and Reception Time" that can be predicted quite accurately, all the other times are quite unpredictable, and depend on details of implementation, and task priority. The TPSN either leaves outside the measurements, or measures out quite accurately all these unpredictable times, so they

have minimum impact on the final propagation time accuracy. It is worth to remark that TPSN assumes that incoming and returning propagation times are identical, that is sometimes a too simple assumption. Unless the packet creation and transmission are controlled in hardware, there is no guarantee that the packet will start to be transmitted at T3B; however, in one practical implementation, the value T3B is already on board of the returning packet, and needs to be predicted. A way around this difficulty is simply to send a second message from node B to node A, notifying node A that the return packet started to be transmitted at time T3B.

2.4.3 Flooding Time Synchronization Protocol (FTSP)

Table 1 gives some insight about the delays, their order of magnitude, and their unpredictability (Maroti 2004):

Table 1 - Flooding Time Synchronization Protocol - Processing Times

Time	Magnitude	Distribution
Send and Receive	0-100ms	Nondeterministic, depends on the processor load
Access	10-500ms	Nondeterministic, depends on the channel contention
Transmission, Reception	10-20ms	Deterministic, depends on message length
Propagation	<1µs for distances <300m	Deterministic, depends on the distance between sender and receiver
Interrupt Handling	<5µs in most cases, but can be as high as 30µs	Nondeterministic, depends on interrupts being disabled
Encoding plus Decoding	100-200µs <2µs variance	Deterministic, depends on radio chipset and settings
Byte Alignment	0-400µs	Deterministic, can be calculated

Note: Excerpts are from Marotti (2004)

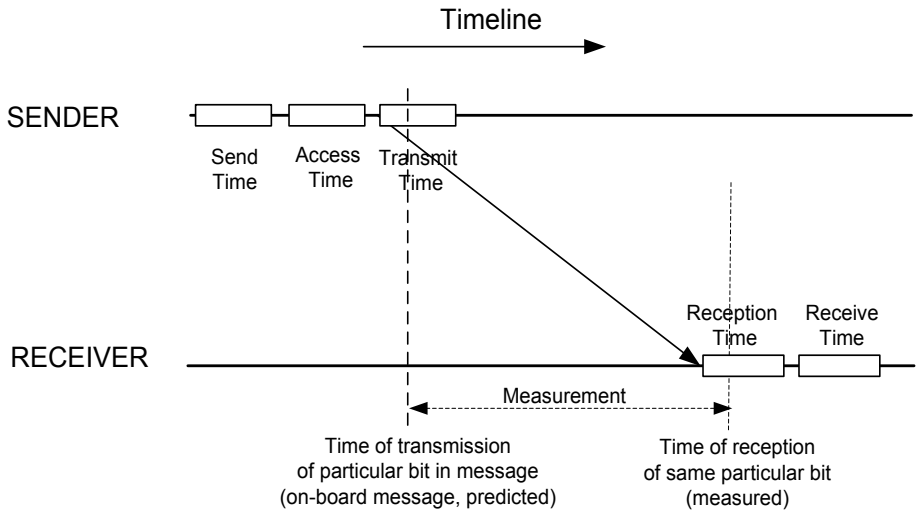


Figure 7 - FSTP Timing Sequence

The flooding technique is a simplified TPSN method in the sense that only one message is sent to the receiver, but the time indeterminations are kept to a minimum by hardware time tagging at transmission and at reception. The transmission time tag is part of the transmitted message. There is a weakness when the transmit time tag is part of the transmit message itself; this requires a relatively sophisticated prediction technique that could possibly be defeated in some atypical cases. The skew can be measured in a way similar to RBS, still necessitating multiple sync messages over a long period. It cannot measure the propagation time at all, therefore can exhibit up to 1µs of error that cannot be compensated. The real contribution of the authors is to have made the resolution to deeply redesign the radio MAC layers (sender and receiver), and to have proposed a sophisticated and robust recovery method, in case the root node would disappear or become disconnected from the rest of the network. It is believed than any of the other

described solutions would perform better than FTSP overall if they would use the same hardware time tagging technique.

2.4.4 Comparative Performance

All the methods described so far try to eliminate as many time indeterminations in the raw measurement, and keep only the propagation time. They just make different compromises and eliminate different parasitic times out of the measurements. To give an idea of a typical propagation time, the maximum distance between nodes in direct radio contact will be assumed at about 300 metres, as a consequence of the very low energy available at each node, and thus very low energy and power budgeted for wireless operation; this translates into a maximum propagation delay of about 1microsecond.

All these techniques deal with delays that are mainly created by the internal implementation of the nodes and that are further aggravated for RBS and TPSN by the decision to use only standard implementations of the radio and MAC layers for ease of implementation. The FTSP takes a radically different approach, postulating that all layers of the wireless protocol can be bypassed in simple WSN implementations, and then the time tagging of transmitted and received messages is done in hardware, whereby eliminating all software timing uncertainties.

For ease of node design, FTSP is the worst, because it requires a significant radio MAC hardware change, that may not be possible with some radio chips, or more complex radio protocols. For messaging protocol simplicity, FTSP is certainly the best, as it requires a single forward message (repeated quite often, however). TPSN needs two messages. RBS is the worst, as it requires the contribution of three nodes to synchronize only two, one of

them being outside of the synchronizer-synchronizee pair. The case where the sender is heard by both receivers, but the receivers are not at hearable distance of each other needs special handling, with receiver-to-receiver messages being forwarded by the sender. Regarding the power consumption, FTSP is certainly the worst, as it takes a while for the whole network to synchronize onto a single node (10 minutes for 60 nodes), this is the price paid for robustness. Also, for calibrating the clocks skew, it takes quite a few messages over a long period (10-15 minutes) to correctly observe the slow shift between clocks over time. Only TPSN is capable of measuring the propagation time, but with an uncertainty about one order of magnitude larger that completely obliterates the potential advantage. All the methods are local pair-wise techniques. Any attempt to synchronize multi hops from the same reference node will see the error accumulate at each hop, with no overall upper bound. The conversion from relative synchronization to absolute has to be done for any of these methods. Because the performance is so dependent on the implementation, no relevant performance uncertainty numbers can be simulated. Only numbers from Savaranos (2006b) and Maroti (2004), will be cited in Table 2 as a baseline.

Table 2 - Synchronization Techniques Relative Performance Comparison

	TPSN	RBS	FTSP
Average Error (μs)	16.9	29.1	1.48
Worst Case Error (μs)	44	93	10[(1)]
Best Case Error (μs)	0	0	0.5[(2)]
%Time Error < Average	64	53	Na

[(1):] this is the worst sync time for a stabilized network.

[(2):] multihop time uncertainty is taken here as the best case

This overview showed that none of these techniques can guarantee <5 μs in all cases, thus none is suitable for direct synchronization for carrier phase relative positioning. On the other side, all of them deliver ample margin to the worst case required uncertainty of 100ms if the WA-GPS is used for the fine synchronization. In practice any one would work fine as a bootstrap to the WA-GPS technique. The best would be the simplest with the lowest extra design complexity. A hybrid solution could be proposed that would use the single forward message of the FTSP, but with no changes in the hardware implementation, as implied by RBS or TPSN.

CHAPTER 3

SYNCHRONIZATION ANALYSIS

It is conventionally known that the satellite timing errors and receiver timing errors compensate in the Double Differencing observable. This is why this type of observable is so popular in the first place. A common rule of thumb is that, in order to become acceptable on their impact on integer ambiguity resolution, the timing errors between receivers must be in the order of few microseconds, and that the common timing error vs. GPS needs to be in the tens of milliseconds. One corollary is that, in conventional processing, a systematic drift or clock error in a receiver, which is known and can be accounted for during the computation of the geometric ranges between satellites, does not enter in this category. This type of problem is usually resolved by computing a pseudo-range based single position using measurements from the receiver in question and resolving for the receiver clock time error. The actual receiver clock is re-synchronized when its drift from GPS time exceeds a specific threshold, typically less than a few ms. This is important to ensure that carrier phase observations at the two receivers are made at approximately the same time or at known time difference, especially for kinematic applications.

This strategy is not applicable in a Wireless Sensor Network; the single position computation in each node is impractical. It taps into the limited power budget of each sensor node. It also imposes the implementation of a complete set of GPS functions in

each node. That runs against the intent to implement at each node only the GPS functions that need to be different, and to distribute and share across the network all the common functions necessary for a full GPS implementation. GPS measurement sampling time accuracy will depend exclusively on the inter node wireless time transfer accuracy. No further correction will be possible.

In this chapter, the impact of time synchronization in the relative positioning process will be investigated, in particular, the success rate, the time to fix ambiguities to their integer values and the processing load (number of ambiguities search before success) will be explored. To do so, the general positioning procedure that has been used for this synchronization analysis and that also forms the basis of the ARTI technique will be first introduced.

3.1 Sensitivity to the Timing Errors

Before delving into the details, an intuitive understanding of the effect of the timing errors on double difference carrier phase measurements is in order. For this demonstration, a very simple two-dimensional model will be used. The configuration is illustrated in Figure 8. The satellite orbits are perfectly circular, with a period of 12 hours, and an orbit radius of 25000 km. Earth is assumed spherical with a radius of 6400 km. Two satellites and two receivers only are considered. Both satellites are assumed to be on the same orbit, but in opposite directions of displacement. Earth is also supposed to be non-rotating in the inertial reference frame. Both satellites S1 and S2 start from the same position, at the intersection of the orbit and of the y axis, and move along on the orbit on

mirrored positions respectively to the y axis. The receivers R1 and R2 occupy positions symmetric to the y axis on the surface of the Earth.

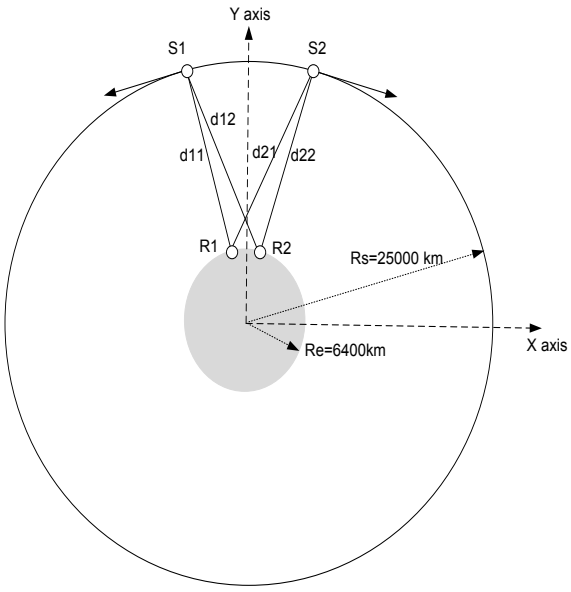


Figure 8 - Double Difference Sensitivity to Timing Errors - Simple Scenario

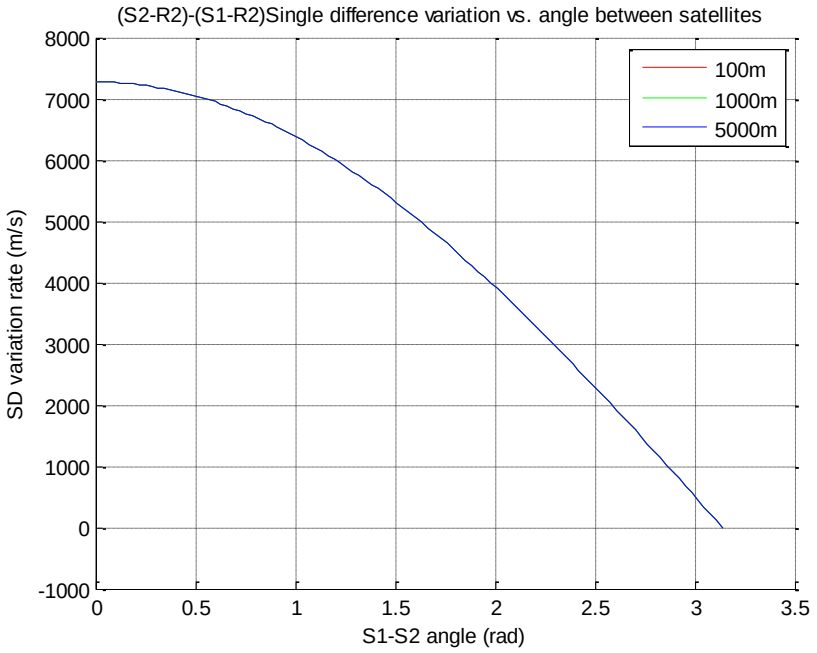


Figure 9 - Single Difference variation vs. angle between satellites

To maximize the error, it is necessary to select the satellite locations where the double difference changes per second are at their maximum possible. This point is when satellites start at the same location, at the intersection of the common orbit, and of the y axis. This claim will be demonstrated first.

The distance double difference can be expressed as:

$$DD_{12}^{12} = (D_1^1 - D_1^2) + (D_2^2 - D_2^1) \qquad (3.1)$$

where D_j^i represents the distance between satellite i and receiver j. Both terms represent a single difference. Both single differences are equal, because of the symmetry of the configuration, and add to each other. Only at a single difference variation over the angle

between satellites centered at the centre of the Earth will be considered.

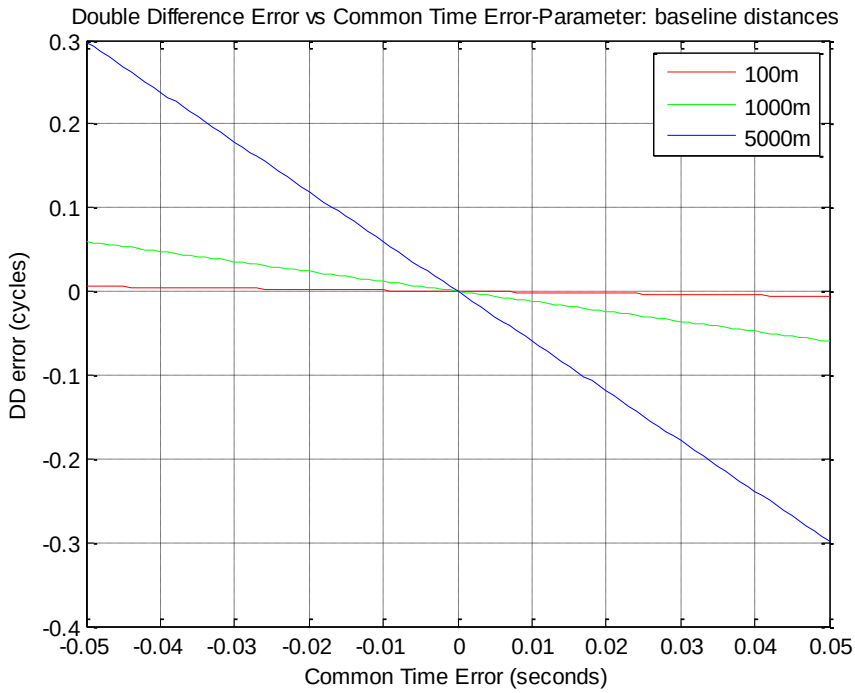


Figure 10 - Double difference in Cycles vs. Common Time Error for Multiple Baselines

Figure 9 shows the variation of the single difference in metres, vs. the S1-S2 angle. The maximum variation corresponds to about 7200 metres per second for both satellites being on the y axis. This single difference variation goes down to zero for both satellites in antipodal positions. Therefore, the common satellite position on the y axis will be chosen as the starting position for evaluation of the double difference sensitivity to the timing errors.

The double difference is relatively insensitive to the common sampling time errors at both receivers (Figure 10). It still largely depends on the inter receiver baseline distance. For a 1 km receiver baseline, the error is approximately linear with a slope of about 0.06

cycles for 50 ms of error, or about 1.2 cycles per second of common error. The sensitivity to an inter receiver time difference error is much more important: Figure 11 shows a double difference error of 0.0225 cycles for a difference time error of 5 microseconds, or a slope of about 4500 cycles per second of difference time error. This is not very sensitive to the receiver baseline. In order to completely characterize the acceptable timing errors, both common and differential, more information about the sensitivity of the ambiguity resolution to the double difference errors will be introduced later on in this chapter. As a last comment to the detail oriented reader, this demonstration could have been done in a more accurate fashion using the analytical formulas of satellite position and velocity shown in the Appendix 3, with the same conclusions, albeit, with less insight about the physics of the problems. As a conclusion to this section, in order to add an extra error to the double difference of no more than $1/10^{th}$ of the 3mm standard deviation thermal noise (or 0.015 cycles) on a single carrier, the common time error cannot be more than 0.015 /1.2=0.0125 s. With the same assumptions, the inter receiver timing error cannot be more than $0.015/4500=3.33\ 10^{-6}$ s. Please keep in mind this rule of thumb of 10ms common time error and 3.33 microseconds of differential time error.

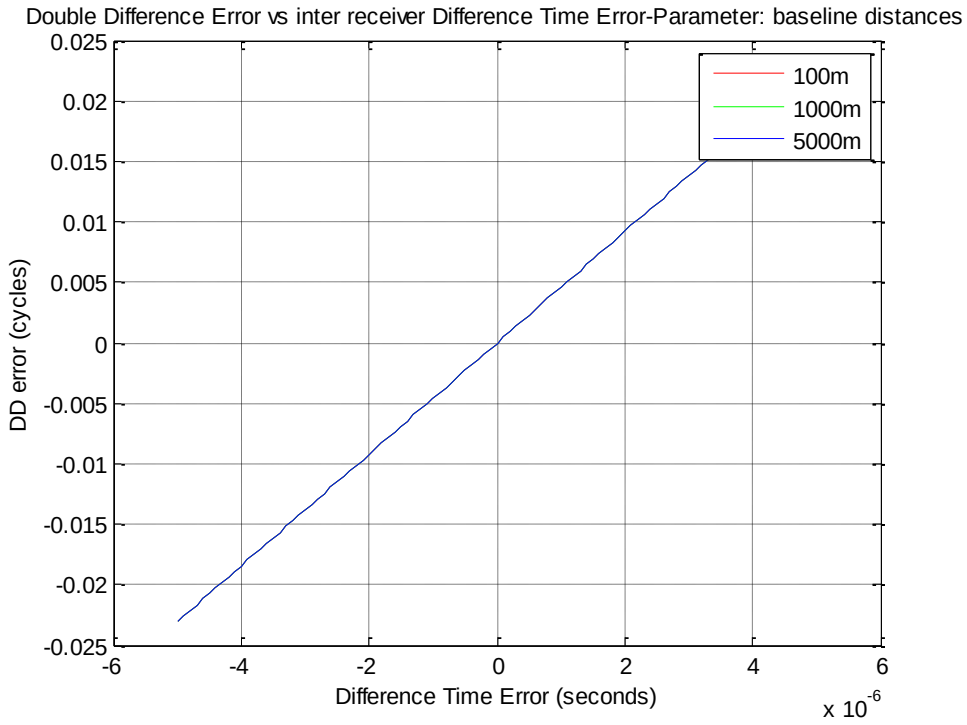


Figure 11 - Double Difference Error vs. inter receiver difference time error

3.2 Carrier Phase Double Difference Positioning

Before delving into the time synchronization problems, the foundations for these developments will be stated. The carrier phase based relative positioning technique retained for this application is rather classical, with the most limited processing power possible.

3.2.1 Carrier Phase Observables

Following the terminology of Leick (2004) page 173, the phase observable in number of cycles from a satellite p to a receiver k at L1 frequency can be expressed as:

$$\varphi_{k,1}^{p} = \frac{f_1}{c}\rho_k^p(\hat{t}^p) + N_k^p(1) - f_1 d\underline{t}_k + f_1 d\hat{t}^p + I_{k,1,\varphi}^p(t_k) + \frac{f_1}{c}T_k^p(t_k) + \delta_{k,1,\varphi}^p(t_k) + \varepsilon_{1,\varphi} \quad (3.2)$$

$$\delta_{k,1,\varphi}^p = d_{k,1,\varphi}(t_k) + d_{k,1,\varphi}^p(t_k) + d_{1,\varphi}^p(t_k) \quad (3.3)$$

with:

f_1	Nominal frequency L1(Hz)
c	Velocity of light(m/s)
$\rho_k^p(\hat{t}^p)$	Geometric distance between satellite p and receiver k at true time at epoch of transmission $\hat{t}^p$(m)
$\hat{t}^p$	True time at satellite p at epoch of transmission
$N_k^p(1)$	Integer ambiguity at first observation epoch (1). It remains constant until a cycle slip occurs
$d\underline{t}_k$	Nominal receiver clock error ($\hat{t}_k = t_k + d\underline{t}_k$)
$d\hat{t}^p$	p Satellite clock error at time of transmission ($\hat{t}^p = t^p + d\hat{t}^p$)
t^p	Clock time at satellite p at time of transmission
t_k	Time at receiver k at the epoch the code entered the antenna
$\hat{t}_k$	True time at receiver k at the epoch the code entered the antenna
$I_{k,1,\varphi}^p(t_k)$	Ionospheric C/A phase delay at L1 frequency between satellite p and receiver k (cycles)
$T_k^p(t_k)$	Tropospheric delay in distance (m)
$\delta_{k,1,\varphi}^p(t_k)$	Hardware delays and multipath effects on the L1 carrier phase (cycles)
$d_{k,1,\varphi}(t_k)$	Receiver k hardware delay at L1 (cycles)
$d_{k,1,\varphi}^p(t_k)$	Multipath delay between satellite p and receiver k at L1 (cycles)
$d_{1,\varphi}^p(t_k)$	Satellite p Hardware delay at L1 (cycles)

$\varepsilon_{1,\varphi}$ L1 phase measurement noise (cycles)

After double differencing, and eliminating residual ionospheric, tropospheric, satellite and receiver hardware delays, double difference terms as they are expected to cancel over short baselines and ignoring carrier phase multipath double difference and measurement noises treated as unknown error model and ignored, the simplified double difference formula is written as:

$$\varphi_{km}^{pq} = \frac{f}{c}\rho_{km}^{pq} + N_{km}^{pq} \tag{3.4}$$

$$= \frac{1}{\lambda}\{\|\overline{x}^p - \overline{x}_k\| - \|\overline{x}^p - \overline{x}_m\| - \|\overline{x}^q - \overline{x}_k\| + \|\overline{x}^q - \overline{x}_m\|\} + \{N_k^p - N_m^p - N_k^q + N_m^q\}$$

with

$$\|\overline{x}^p - \overline{x}_k\| = \sqrt{(x^p - x_k)^2 + (y^p - y_k)^2 + (z^p - z_k)^2} \tag{3.5}$$

Please note that the generic notation X_{km}^{pq} where X can be replaced by any observable or parameter, stands for:

$$X_{km}^{pq} = X_k^p - X_m^p - X_k^q + X_m^q \tag{3.6}$$

3.2.2 Equations Linearization

Under the assumption that the master receiver location $\overline{x}_k$ is known, the previous expression is linearized around an unknown secondary receiver location $\overline{x}_m$ as:

$$\varphi_{km}^{pq}\left(d\overline{x}_m, N_{km}^{pq}\right) = \frac{1}{\lambda}\left\{\rho_{km}^{pq} + \frac{\delta\varphi_{km}^{pq}}{\delta\overline{x}_m}\, d\overline{x}_m\right\} + N_{km}^{pq} \tag{3.7}$$

with:

$$\frac{\delta\varphi_{km}^{pq}}{\delta\overline{x}_m} = \frac{1}{\lambda}\left(\overline{e}_m^p - \overline{e}_m^q\right), \quad e_m^p = \frac{\overline{x}^p - \overline{x}_m}{\rho_m^p},\ e_m^q = \frac{\overline{x}^q - \overline{x}_m}{\rho_m^q}. \tag{3.8}$$

The retained form for each line of the linearized set of equations is:

$$\left\{\lambda\, \varphi_{km}^{pq}\left(d\overline{x}_m, N_{km}^{pq}\right) - \rho_{km}^{pq}\right\} = \frac{\delta\varphi_{km}^{pq}}{\delta\overline{x}_m}\, d\overline{x}_m + \lambda\, N_{km}^{pq} \tag{3.9}$$

The linearization point, in particular the initial position of the secondary receiver m is considered to be the same as the master receiver k. For m satellites, and 2 receivers (single baseline), the matrix equations can be written:

$$\overline{l} = A\overline{x} \tag{3.10}$$

with l^T, 1x(m-1) vector, A (m-1)x(m-1) matrix, and $\overline{x}^T$, the unknown 1x(3+(m-1)) vector:

$$l^T = \left[\lambda\varphi_{12}^{12} - \rho_{12}^{12} \quad \ldots \quad \lambda\varphi_{12}^{1m} - \rho_{12}^{1m}\right] \tag{3.11}$$

$$\mathrm{A} = \begin{bmatrix} \frac{\partial\varphi_{12}^{12}}{\partial x_m} & \frac{\partial\varphi_{12}^{12}}{\partial y_m} & \frac{\partial\varphi_{12}^{12}}{\partial z_m} & \lambda & \cdots & 0 \\ & \vdots & & & \ddots & \vdots \\ \frac{\partial\varphi_{12}^{1m}}{\partial x_m} & \frac{\partial\varphi_{12}^{1m}}{\partial y_m} & \frac{\partial\varphi_{12}^{1m}}{\partial z_m} & 0 & \cdots & \lambda \end{bmatrix} \tag{3.12}$$

$$\overline{x}^T = [dx_m \quad dy_m \quad dz_m \; N_{12}^{12} \quad \ldots \quad N_{12}^{1m}] \tag{3.13}$$

3.2.3 Formation of l, A, and x from raw measurements

Although the preceding sections dealt with single baselines or pairs of receivers at a single time epoch, the generalized operations to transform a vector of raw measurements into a set of linear equations will be shown. For R receivers, S satellites and T epochs, up to RST measurements can be extracted, if all same satellites are observed by all receivers. The process to go from RST individual measurements to (R-1)(S-1)T independent double differences is described. This development is inspired by Leick (2004) page 261.

The "per epoch" sequence of phase measurements is combined into a single vector of dimensions RS x 1, ordering them per epoch first, per receiver second, finally per satellite. In the following developments it is implied that the satellite index "1" and receiver index "1" are the satellite and receiver of reference.

For a single epoch i, the ordered vector is:

$$\psi_i = [\varphi_1^1(i) \quad \cdots \quad \varphi_1^S(i) \quad \cdots \quad \varphi_R^1(i) \quad \cdots \quad \varphi_R^1(i)]^T \tag{3.14}$$

And for T epochs:

$$\psi = \begin{bmatrix} \psi_1 \\ \vdots \\ \psi_T \end{bmatrix} \tag{3.15}$$

The associated undifferenced RST x RST covariance matrix is:

$$Q_\varphi = \sigma_\varphi^2 \overline{I} \tag{3.16}$$

where σ_φ is the standard deviation of the phase measurement in cycles, and $\bar{I}$ is the RST x RST unit matrix.

Δ_i, the double difference vector at single epoch i, can be expressed as:

$$\Delta_i = [\varphi_{12}^{12}(i) \quad \cdots \quad \varphi_{12}^{1S}(i) \quad \cdots \quad \varphi_{1R}^{12}(i) \quad \cdots \quad \varphi_{1R}^{1S}(i)]^T \tag{3.17}$$

The double difference vector at epoch i Δ_i can be directly obtained from ψ_i when applying:

$$\Delta_i = D_i \psi_i \tag{3.18}$$

where D_i of dimensions (S-1)(R-1) x SR, is defined according to the pattern of Figure 12.

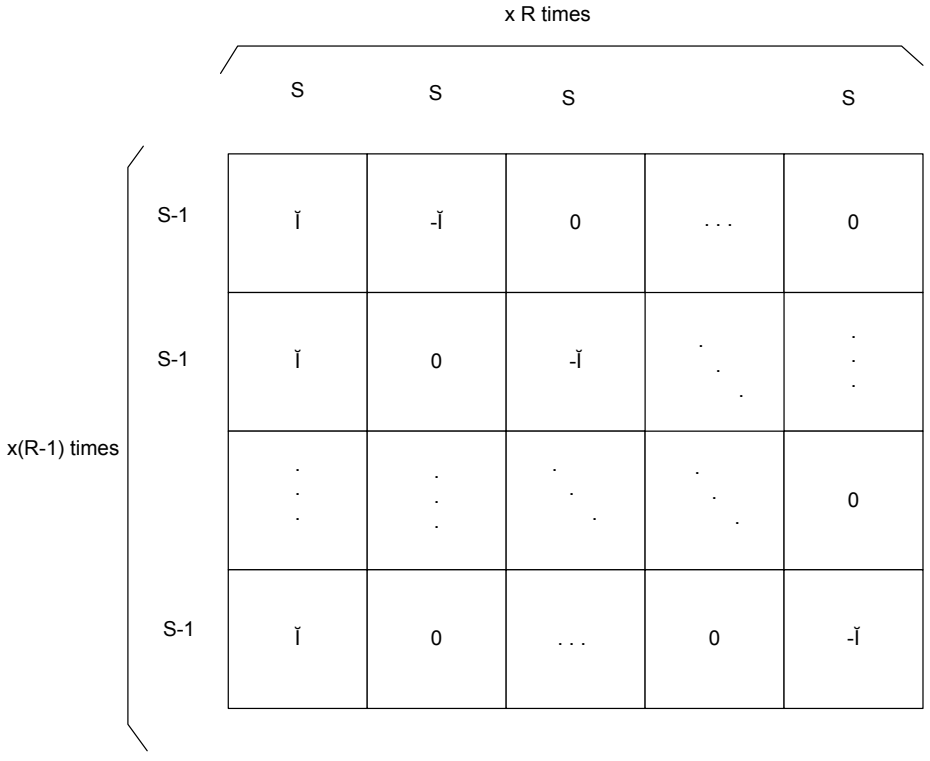


Figure 12 - Pattern of the Di matrix

The $\check{I}$ sub matrix of dimensions (S-1) x S appearing in D_i is defined by:

$$\check{I} = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ 1 & 0 & -1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 1 & 0 & \dots & 0 & -1 \end{bmatrix} \tag{3.19}$$

For example, if S=4:

$$\check{I} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix} \tag{3.20}$$

The double difference vector across satellites, receivers and epochs Δ, of dimensions (R-1)(S-1)T x 1,

$$\Delta = \begin{bmatrix} \Delta_1 \\ \vdots \\ \Delta_T \end{bmatrix} \tag{3.21}$$

can be directly combined as

$$\Delta = D\psi \tag{3.22}$$

where D pattern is defined in Figure 13.

Following the same logic, the double difference covariance matrix can be expressed as:

$$\Sigma_\Delta = \sigma_\varphi^2 DD^T \tag{3.23}$$

And the cofactor matrix is expressed by:

$$Q_\Delta = DD^T \tag{3.24}$$

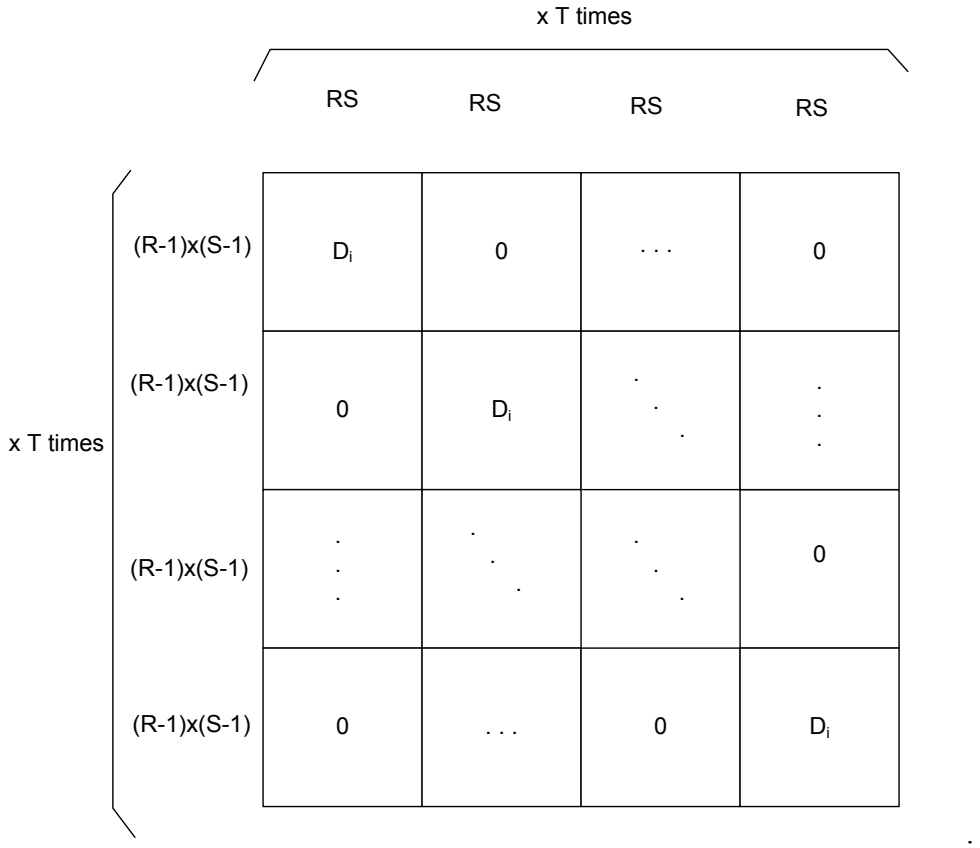


Figure 13 - Pattern of the D matrix

3.2.4 Rows Reduction using Normal Equations

In a general case, several epochs of measurements are required before the integer ambiguity resolution can be reliably attempted. According to the development of the previous section, the measurement vector $\bar{l}$ will have (R-1)(S-1)T rows, and the design matrix A will have dimensions [(R-1)(S-1)T] x [3+(S-1)]. The size of these equations can become significant, and the processing power to resolve them directly can be

overwhelming. For a simple case of R=2, S=8, and T=10, $\bar{l}$ will be 70 x 1 and A will be 70 x 10. Using a normal equation formulation, the size of the equations to be resolved can be kept under control, regardless of the number of epochs.

For a single epoch i, the standard form is as below:

$$\bar{l}_i = A_i \bar{x} \tag{3.25}$$

The measurement vector $\bar{l}_i$ and the design matrix A_i are dependent on the time epoch, whereas the vector of unknown parameters $\bar{x}$ is not (at least in a simple hypothesis of no cycle slips, and constant number of satellites with available measurements). The normal form equivalent is:

$$A_i^T \bar{l}_i = A_i^T A_i \bar{x} \tag{3.26}$$

That form can be justified as, at its minimum, the vector residual $\left(\bar{l}_i - A_i \bar{x}\right)$ is perpendicular to the subspace spanned by the columns of A. In other words,

$$A_i^T \left(\bar{l}_i - A_i \bar{x}\right) = 0 \tag{3.27}$$

Each of the T epochs will provide a similar set of equations:

$$\begin{cases} A_1^T \bar{l}_1 = A_1^T A_1 \bar{x} \\ \cdots \\ A_T^T \bar{l}_T = A_T^T A_T \bar{x} \end{cases} \tag{3.28}$$

or stacked together, left member being a [(R-1)(S-1)T] x 1 vector, and $A^T A$ being a (R-1)(S-1)T square matrix:

$$A^T \bar{l} = A^T A \bar{x} \tag{3.29}$$

By summing left equation members together and right equation members together, the compact form becomes:

$$\left(\sum_{i=1}^{T} A_i^T \bar{l}_i\right) = \left(\sum_{i=1}^{T} A_i^T A_i\right)\bar{x} \tag{3.30}$$

In its weighted linear square form, with weight matrix $\Sigma_l = \frac{1}{Q_l}$ the inverse of the cofactor matrix, it becomes:

$$\left(\sum_{i=1}^{T} A_i^T \Sigma_i^T \bar{l}_i\right) = \left(\sum_{i=1}^{T} A_i^T \Sigma_l A_i\right)\bar{x} \tag{3.31}$$

the left member is of (R-1) x (S-1) dimensions, and $(\sum_{i=1}^{T} A_i^T \Sigma_l A_i)$ being a square matrix of (R-1) x (S-1), regardless of the number of epochs. Exploiting the weight matrix symmetry property, the final form implemented in the simulations can be expressed as:

$$\bar{x} = \left(\sum_{i=1}^{T} A_i^T \Sigma_l A_i\right)^{-1} \left(\sum_{i=1}^{T} A_i^T \Sigma_i^T \bar{l}_i\right) \tag{3.32}$$

3.2.5 Integer Ambiguity Resolution using the Lambda Technique

The Lambda technique has been amply demonstrated and studied since Teunissen (1993) proposed it in the first place, and de Jonge and Tiberius described the method (deJonge & Tiberius 1996a) and proposed details of its implementation (deJonge et al 1996b). The development and the rationale will not be repeated here, as it is not the focus of this

dissertation. The original implementation proposed by TU Delft has been completed with the acceleration implementation as proposed by Chaitin-Chatelin et al (2000), that suited the purpose very well. The MILES technique proposes a similar approach (Chang & Zhou 2006).

3.3 Types of Synchronization Errors

3.3.1 Synchronization Error vs. GPS time

When the synchronization is between the gateway and a node in direct wireless connection with the gateway, the gateway can be considered as perfectly synchronized to the GPS system time. In practice this synchronization error is about 200 ns, but insignificant vs. the multi microseconds error that the first node will experience after a single synchronization hop with the gateway. From the node's point of view, the synchronization error to a clock source almost similar to GPS can be interpreted as absolute GPS time error. This is the first type of error that will impact the positioning algorithm.

3.3.2 Relative Synchronization Error between Nodes

Another case is when the synchronization with GPS time is very relaxed, but synchronization between neighboring nodes is still quite good. This may arise at a hierarchical level very far from the gateway. The absolute GPS time error is quite large (hundreds of microseconds), as it is the cumulated sum of all synchronization errors for

the long chain of connection hops up to the level of interest. Still, the relative synchronization error between nodes is relatively small (few tens of microseconds).

3.3.3 Simulator description

A simulator written in MATLAB was developed for generating all the results presented throughout this thesis. This is a two-part simulator that first generates a random placement for all nodes, according to simple high level rules. The second part deals with the GPS measurement simulations that need to be generated for the combination of each satellite and each node in the network at imposed intervals for the duration of the simulation run. These measurement simulations are also elaborated in two steps, first by computing the theoretical geometric distances once, and then by iteratively adding all the sources of error. This arrangement minimizes the important effort of computing the ranges from the Kepler parameters by doing it once, then adding all the sources of error by manipulating the theoretical results as many times as the number of ensemble runs with the same network configuration, same satellite configuration and same time of the day in the GPS week.

The sensor network is defined inside a cuboid volume defined as a geographical point in latitude, longitude and altitude, and symmetric ranges in metres in East-West, North-South and Up-Down directions around this center point. The "WSN simulator" function randomly positions all nodes in an uniform manner inside this cuboid, up to the number of requested nodes (Figure 14). The output is a sequence of X, Y, Z locations (point A in Figure 14), that are subsequently converted back to latitude, longitude and altitude in WGS 84. From these positions, a Receiver Independent Exchange Format (RINEX)

navigation file (Gurtner 2001), the simulation starting time, the duration of the simulation, and the time interval between measurements, the "GPS truth simulator" delivers a sequence of true geometric ranges at point B; it also delivers first and second partial derivatives vs. receive time that will be used in the iterative part of the simulation. This first part is run only once. The simulator is timed in nominal receive time, and backward computes the nominal transmit time using formulas in Appendix 2. The second part is run as many times as the number of runs with the same configuration. The first step is to compute the receiver clock error. The clock is characterized by the phase noise spectral parameters h_0,h_{-1} and h_{-2}. The Kasdin (1995) technique is applied to generate the time sequence of clock errors. The true receive time is then adjusted by adding this clock error to the nominal receive time. The inter node synchronization error is finally added modeled as a Gaussian random variable with its standard deviation being an adjustable simulation parameter. The well behaved Gaussian nature of the timing error has been demonstrated by Elson et al (2003) for the combination of all timing delays detailed in Section 2.4. In practice, this synchronization error is applied only to the secondary nodes. The first node, arbitrarily chosen to be the gateway, is assumed to have a much lower synchronization error, that is chosen as separate parameter. The true range, and true transmit time (only for reference) are then computed by correcting the nominal range using a Taylor series of the receive time error. This Taylor series uses the partial derivative of the range vs. receive time, that was computed along with the nominal range in the "GPS true Simulator" according to the formulas in Appendix 2 (Geometric Distance and its First Derivative vs. Receive Time). The true range is then converted into L1, L2 carrier phases, and L1, L2 code phases at the C point. The L2 measurements are

generated even if they are not used in this work. The next step is to add the thermal noises modeled as Gaussian noises, with parametrized standard deviations (point D). The last step is to keep track of the initial carrier phases, and to report only their variation over time (cumulated carrier phases) at point E. No multipath simulation has been added. If needed, the cycle slips are added later on at will on the cumulated carrier phases.

A Graphical User Interface (GUI) was also developed to streamline the simulation operations.

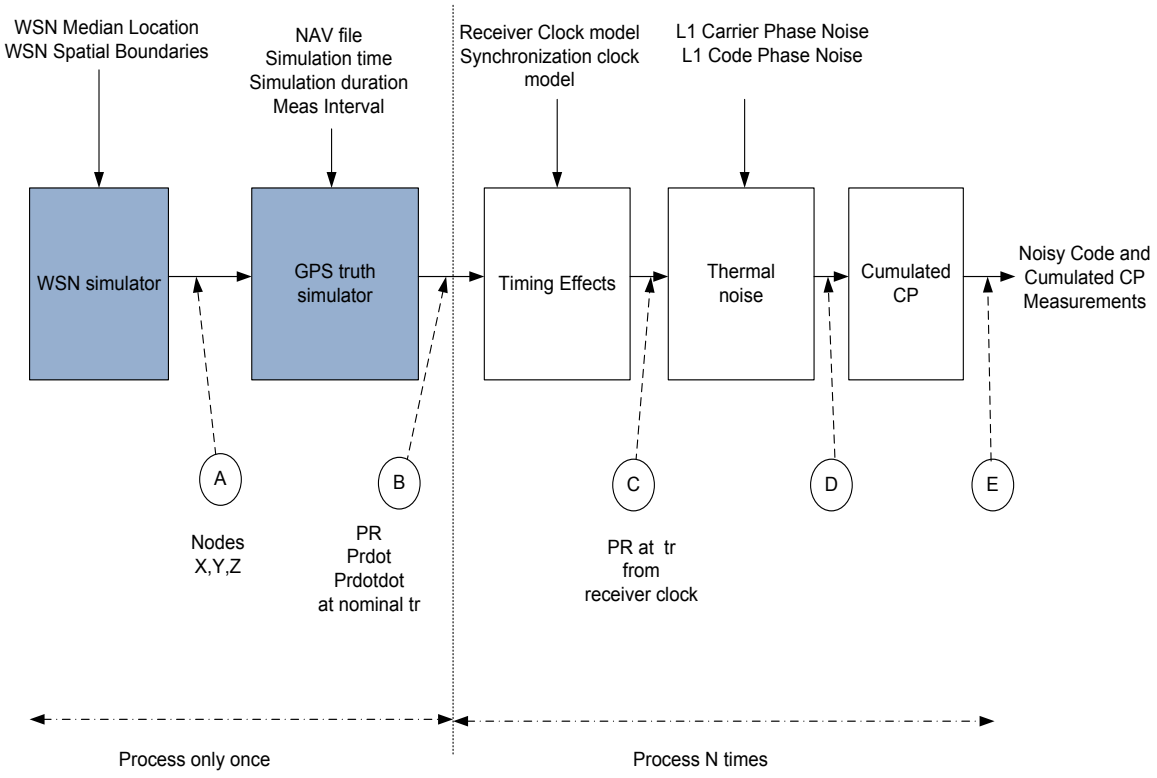


Figure 14 - Simulator Block Diagram

This interface with a typical example of configuration inputs is depicted in Figure 15. This user interface will be succinctly described as it conveniently illustrates the level of

control given over the simulations. The network configuration, i.e. the 3D random nodes location, is generated in STEP 1. The number of nodes to position is specified; its number can vary from 2 to a large number. By definition, the gateway node is always the one with the first index. A reference position that can be defined anywhere on the Earth as a latitude, longitude and altitude in WGS84 is used as the centre of a rectangular parallelepiped. The node positions are uniformly randomly chosen in a cuboid volume. The cuboid dimensions are expressed as symmetrical range variations around the reference position in local coordinates as altitude, North and East range around the reference position. STEP 2 deals with the generation of the TRUE satellite geometric range at every node, at each NOMINAL second (exact second intervals), including position, velocity and acceleration in ECEF coordinates.

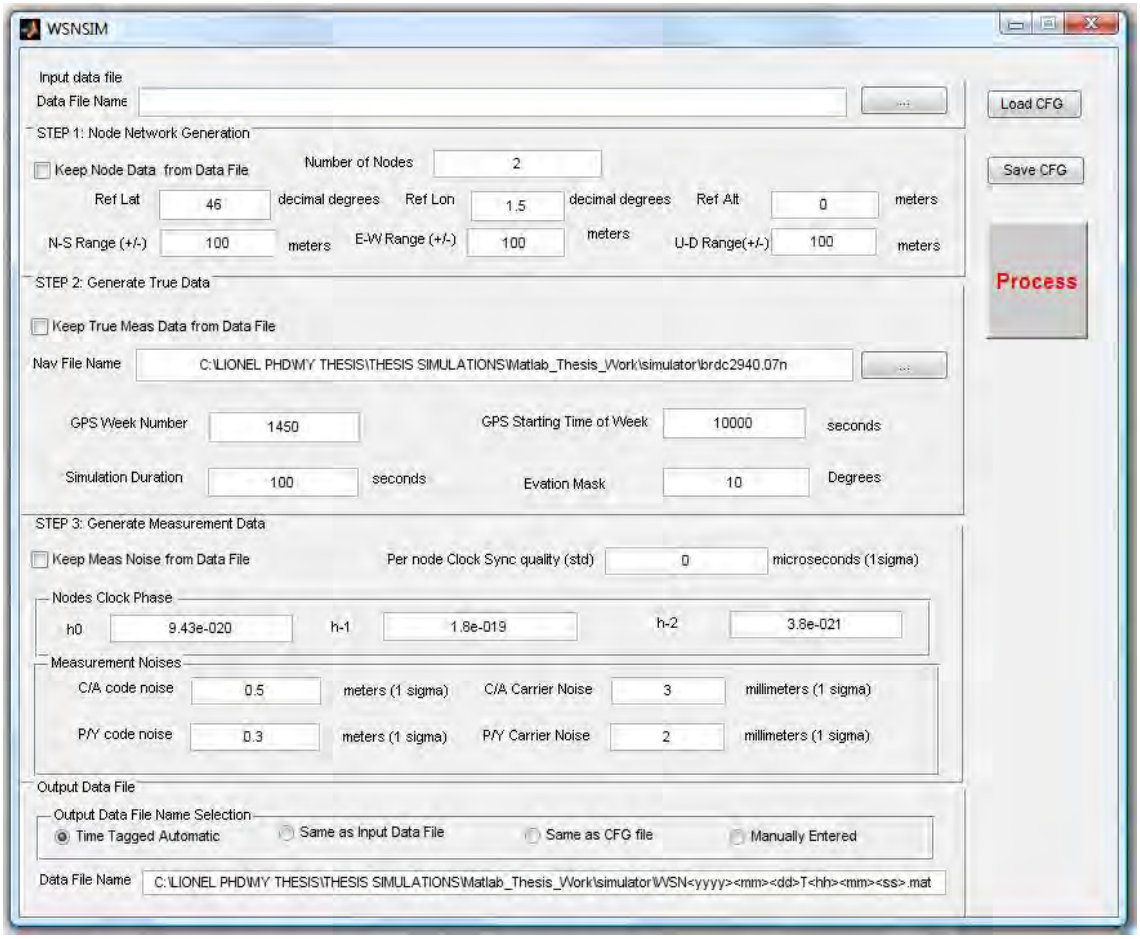


Figure 15 - Matlab Simulator Graphical User Interface

The completeness of the simulator allowed multiple experiments and comparisons between methods with the data during the study, among them, code phase single frequency relative positioning and dual frequency carrier phase relative positioning. A NAV RINEX file downloaded from the Internet is used for satellite location definition. Week number, starting time in the week, duration of the simulation and elevation angle are then specified. To be validated, week number and time of the week have to be in agreement with the data found in the RINEX file. STEP 3 first adjusts the actual time of

the measurements, and modifies the measurements accordingly. The time adjustment first includes the node synchronization error. This synchronization is considered Gaussian and randomly drawn per node. Its standard deviation is adjustable as a parameter. The first node, or the gateway has a separate time adjustment, representative of the residual time error after single positioning that cannot be modified through the configuration file; it was adjusted as 200 ns throughout all experiments, to represent a typical clock residual error in a Standard Positioning Service (SPS). On top of this, a simulation of the quality of the receiver clock was added per node. The receiver clock was specified through the coefficients $h_0 = 9.43\ 10^{-20}$, $h_{-1} = 1.8\ 10^{-19}$ and $h_{-2} = 3.8\ 10^{-21}$. These numbers represent the process noise elements of the linear portion of the clock model. These are the h_0, h_{-1}, and h_{-2} elements of the power law spectral density model used to describe the frequency noise characteristics of oscillators:

$$S_y(f) = \frac{h_{-2}}{f^2} + \frac{h_{-1}}{f} + h_0 \tag{3.33}$$

where

$S_y(f)$ is the clock power spectrum,

h_0, h_{-1} and h_{-2} are the elements of the power law spectral density model,

f is the sampling frequency.

These values correspond to a VCTCXO crystal oscillator (NOVATEL 2010, page 76, Table 23: Pre-Defined Values for Oscillators). When the true time of reception is adjusted at each node, the satellite receiver geometry is adjusted using a second order Taylor series around satellite position, velocity and acceleration derived following the

equations in Appendix 3. All measurements are then extracted from the geometric range, and consist of C/A-L1, P(Y)-L1 and P(Y)-L2 Pseudo Ranges, and carrier phases for C/A-L1, P(Y)-L1 and P(Y)-L2. For the rest of the simulations, only the C/A-L1 Pseudo Range and Carrier phases are considered to reflect the L1 only GPS node capabilities. The last step adds controlled noise around the TRUE measurements. It adds Gaussian noise to all measurements, 0.5 m 1 sigma for C/A code phase, 3 mm for C/A carrier phase and 2 mm for P(Y)-L1 and P(Y)-L2 carrier phase. Several refinements have been added, for example reusing the node locations and regenerating data at different times, or keeping the same node location, and same true data, but regenerating random times and measurement noises. All the capabilities help either in testing the robustness of the algorithm, or in accelerating the simulation time, by reiterating only the last randomization steps (time and measurement noise).

3.3.4 Simulation Conditions

All the plots and results presented in the rest of this chapter have been done in specific conditions described here. Only the parameters that have not been already defined in the previous section will be defined herein.

The number of trials was chosen 200 for each point in the plot for a good compromise between statistical significance and simulation time. Each trial was using the same position for reference receiver and secondary receivers. In order to focus on the time synchronization effects only, the receiver positions were kept the same for all runs across all synchronization error hypotheses and across both synchronization techniques. The potential influence of the node positions on the results was checked independently by

running multiple times the simulations with different node positions, with no noticeable difference. Ranges were +/-100 m around the reference position in all three local coordinates.

As a word of caution, it should be noted that simulated observations contain phase noise as the only error. Therefore, the reported accuracy of the positioning results is likely optimistic as no carrier phase multipath was simulated. However carrier phase multipath although cyclic, is typically below the 1 cm level and has a rms value of the order of 5 mm in most situations. Hence the results presented herein still provide a reasonable estimate of what is achievable.

3.4 Effects of Inter Node Synchronization Errors

The lack of synchronization between nodes is translated into effects on baseline accuracy convergence, correct integer ambiguity fix probability, incorrect ambiguity fix probability, time to fix ambiguity, number of operations to fix the ambiguities. All these effects will be analyzed in the next sections.

3.4.1 Baseline Length Convergence

Convergence before Ambiguity Fix

The quantity of interest is the baseline distance, computed with the float ambiguities, at the iteration before ambiguity fixing. The histogram is composed of 200 runs. For no synchronization error, in Figure 16, the maximum errors are about 12 metres, with a standard deviation in the order of 5 metres.

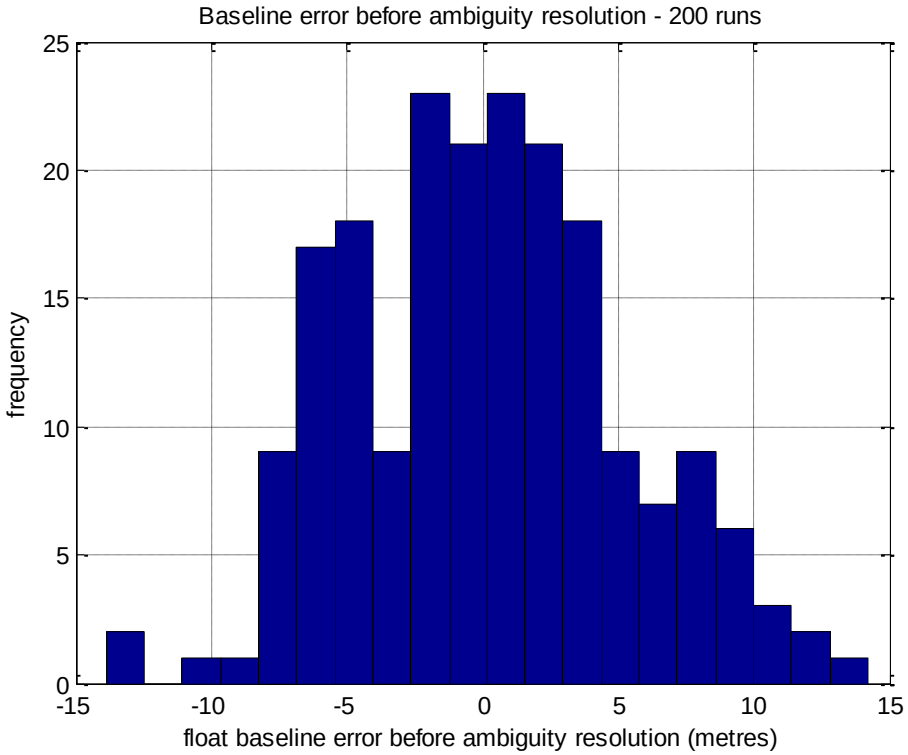


Figure 16 - Float baseline error before ambiguity resolution- no synchronization error

For the benign case of 2.5 microseconds of synchronization error in Figure 17, the situation is similar even slightly better with a standard deviation around 3 metres, and peak values at -6 metres and +8 metres. The clear conclusion is that the synchronization error does not significantly impact the baseline float errors.

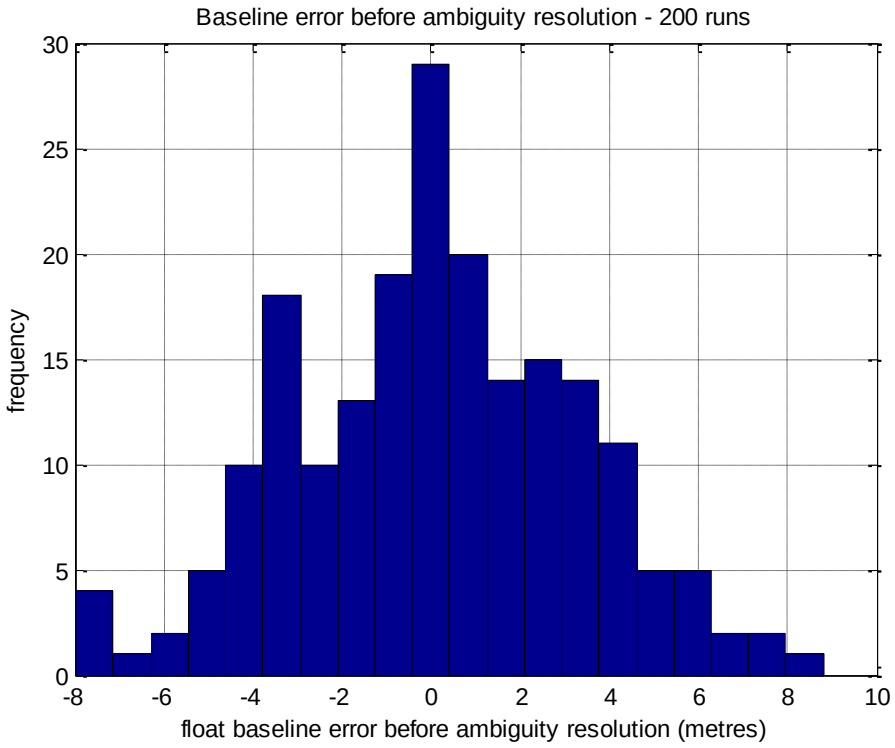


Figure 17 - Float Baseline Error before Ambiguity Resolution - Synchronization Error: 2.5µs

To answer the common argument that the position error using floating ambiguities should be in the sub-metre range, the observed standard deviation of few metres is due to the very short time to ambiguity fix. Further tests have shown that the floating ambiguity position error indeed falls below the metre level when 100 or more 1second epochs have been cumulated while forcing the solution to stay in floating mode. The same fact is also demonstrated by the evolution over time of the position solution covariance matrix position diagonal elements as shown in Figure 18 and Figure 19. In both figures, the relative position and Double Difference ambiguities RMS are shown.

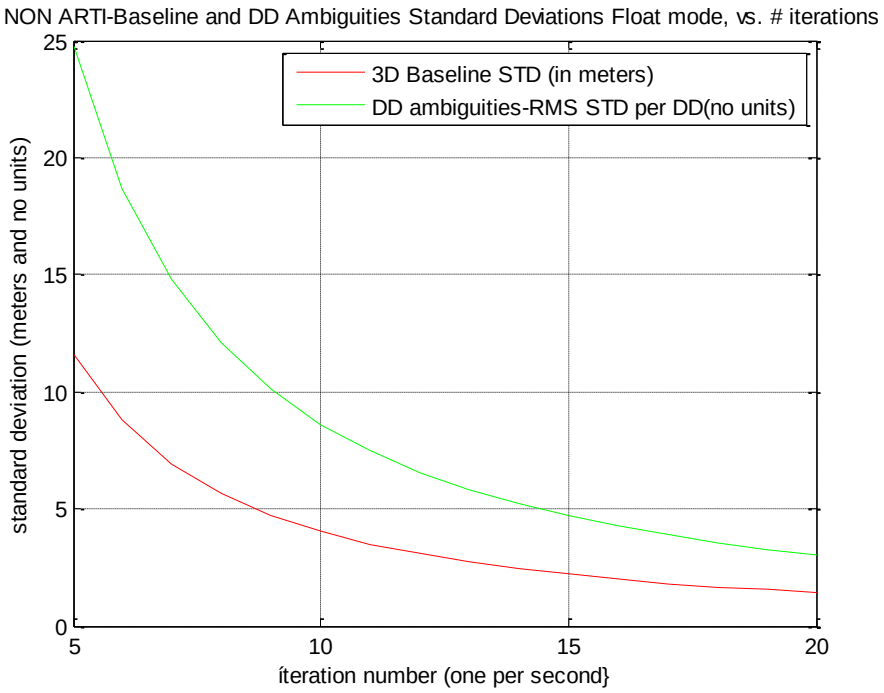


Figure 18 - Baseline and Ambiguities Standard Deviations in Float Mode vs. Number of seconds - 5 to 20 seconds

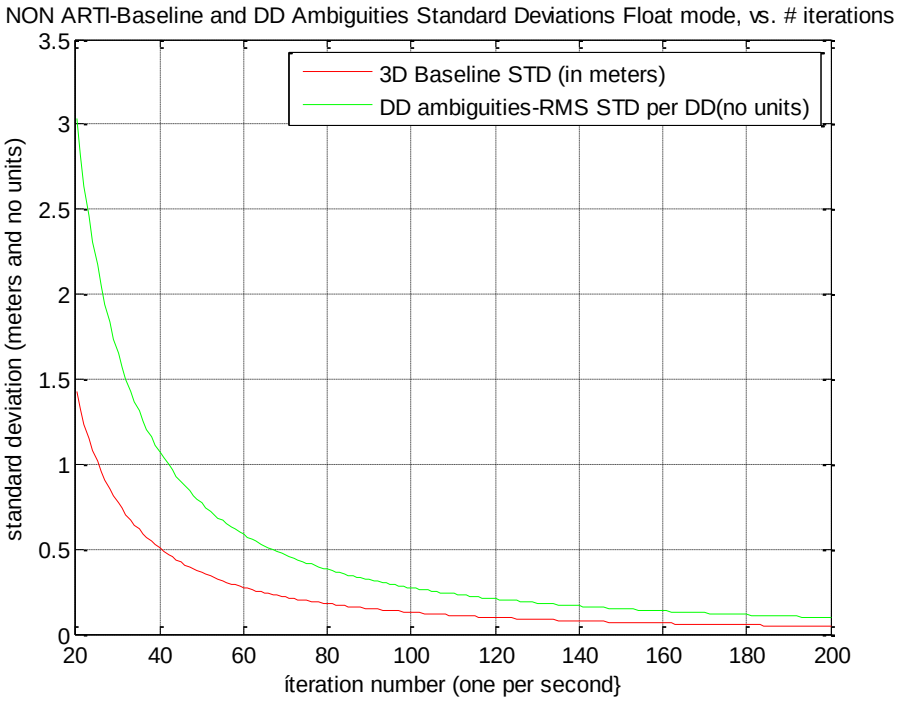


Figure 19 - Baseline and Ambiguities Standard Deviations in Float Mode vs. Number of seconds - 20 to 200 seconds range

Convergence after Ambiguity Fix

The next figures illustrate the baseline distance improvement rate with the number of cumulated measurements, after the Double Difference ambiguities have been resolved. With no synchronization error (Figure 20), the error of distance monotonically decreases to 0.2 mm 3σ, after 200 seconds of cumulated 1 second measurements after ambiguity fixing. If a very modest one sigma Gaussian synchronization error of 2.5 microseconds is assumed, the 95% distance error hovers at about 4 mm 3σ, with no tendency to reduce over the number of measurements (Figure 21). This is a qualitative indication of the extreme sensitivity of the position accuracy to the synchronization error.

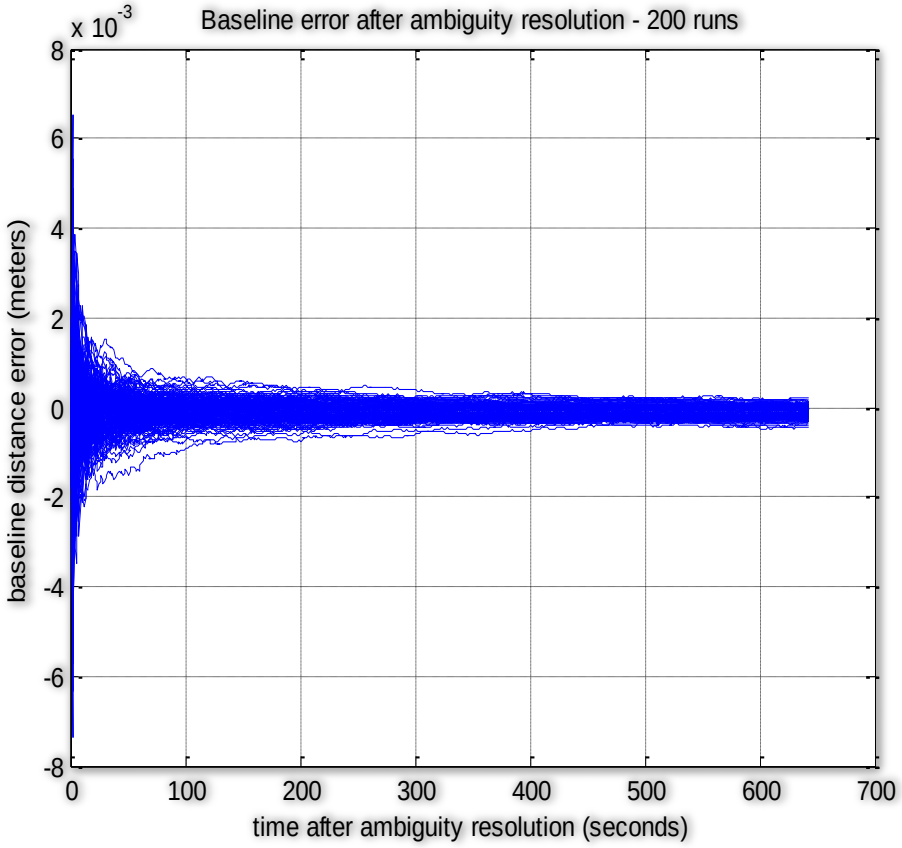


Figure 20 - Baseline Error after Ambiguity Fix - No Synchronization Error

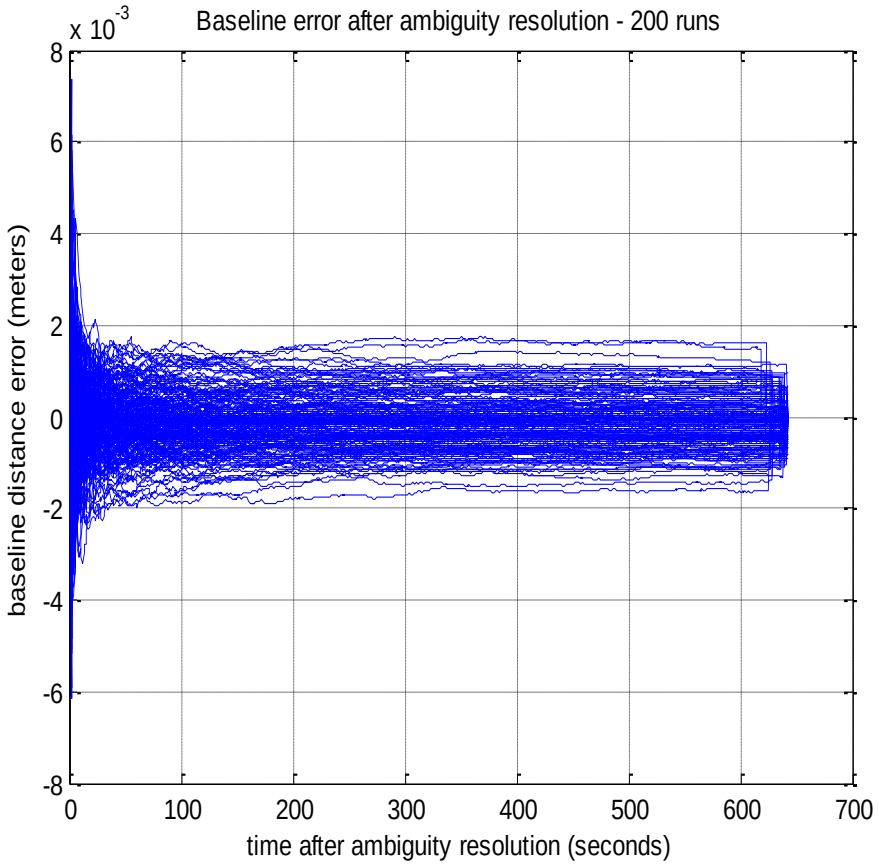


Figure 21 - Baseline Error after Ambiguity Fix - 2.5 us 1 sigma Synchronization Error

3.4.2 Correct and Incorrect Ambiguity fix probabilities

Additional degradation effects of the synchronization error on relative positioning will now be explored. The next question is how much synchronization error results in unacceptable degradation. The impact of synchronization on ambiguity resolution rate is shown in Figure 22. In the following figures, the "Success Rate" is defined as the ratio of cases where the DD ambiguities have been correctly resolved after the DD ambiguity resolution criterion has been satisfied vs. the total number of runs. The "Failure Rate" is

the ratio of cases where the DD ambiguity resolution criterion has never been satisfied over the total duration of the run. The "Type II Error Rate" or “False Negative Rate” is the ratio of cases where the DD ambiguity resolution criterion has been satisfied, but the DD ambiguity solution was incorrect. The ambiguity resolution success rate starts to noticeably degrade after 5 microseconds one sigma; simultaneously, the incorrect ambiguity resolution rate increases in the same proportions. At about 27 microseconds, there is an equal chance to correctly or incorrectly resolve the ambiguities. The results of this section only represent the effects of timing errors.

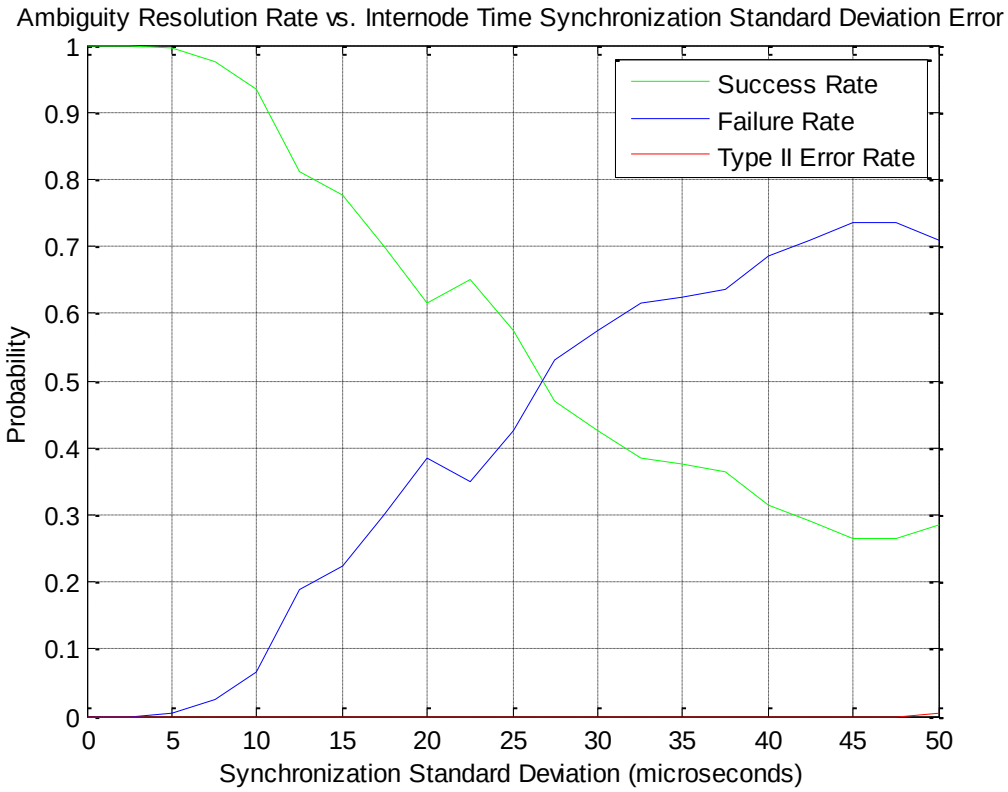


Figure 22 - Ambiguity Resolution Rate vs. Synchronization Error

3.4.3 Time to Fix Integer Ambiguities

In the order of importance for performance criteria, the Time to Fix Ambiguities (TTA) comes immediately after the resolution success rate. A benign case and an extreme case will be examined. Again these examples only explore the effects of a timing error.

Small synchronization error

For reference, Figure 23 shows the "no synchronization error case". The minimum is 6 seconds (imposed by the algorithm that does not try to resolve ambiguities until at least 6 sets of measurements are collected). 90 % of the cases, the resolution is done after these 6 seconds, with only 10 % up to 10 seconds maximum. The 2.5 microseconds case reduces the 6 second time at 70 % of the time the rest staggering up to 30 seconds (Figure 24).

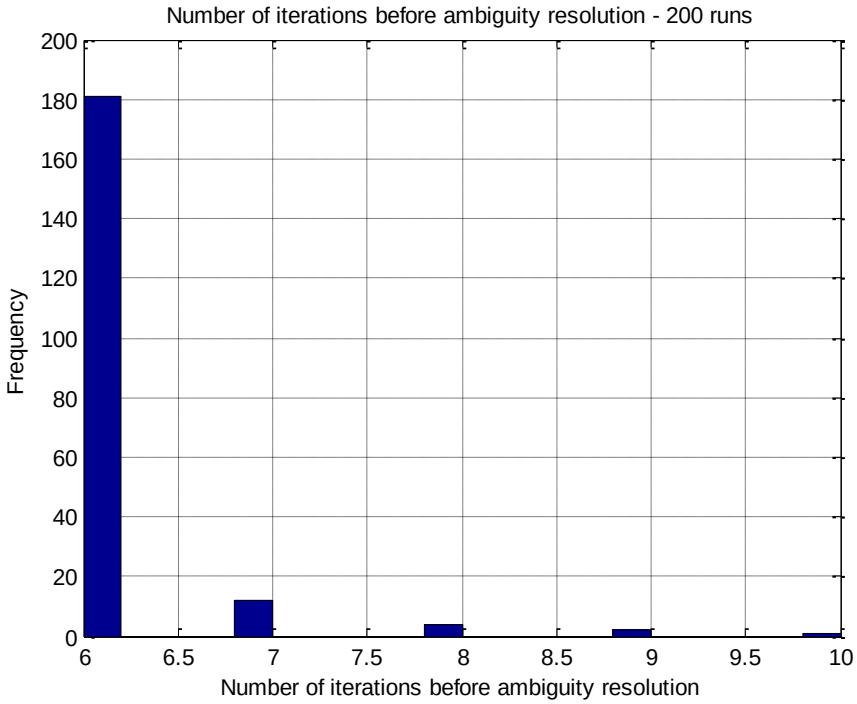


Figure 23 - Number of seconds before ambiguity resolution - no synchronization

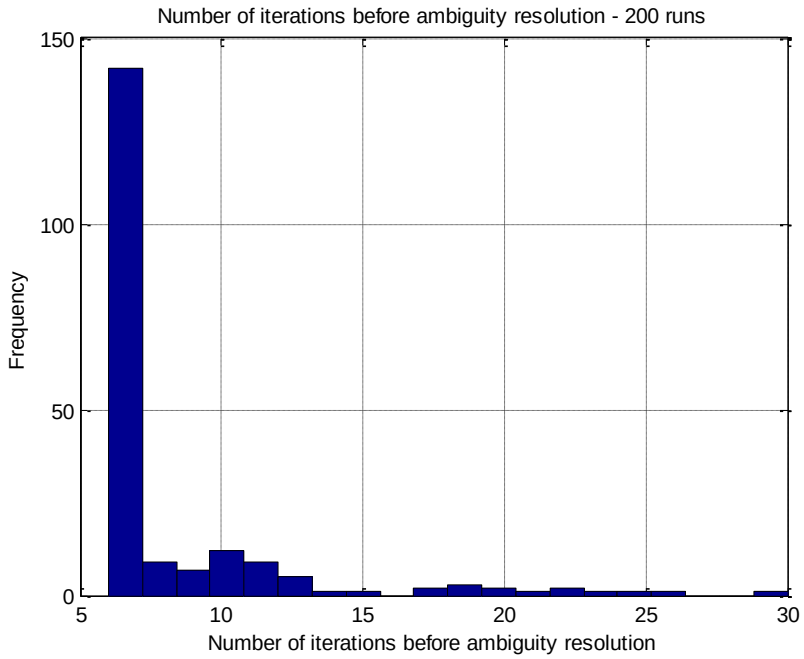


Figure 24 - Number of Seconds before Ambiguity resolution - 2.5 microseconds of synchronization error

Large synchronization error

For larger synchronization errors, the average time increases regularly up to 200 s for a missynchronization of 15 microseconds. In 95 % percentile or max values, the situation is much worse, with times increasing up to 800 to 1000 seconds. The 1000 second limit is artificial, and should be interpreted as no fix, as all runs were limited in time to the 1000 second limit.

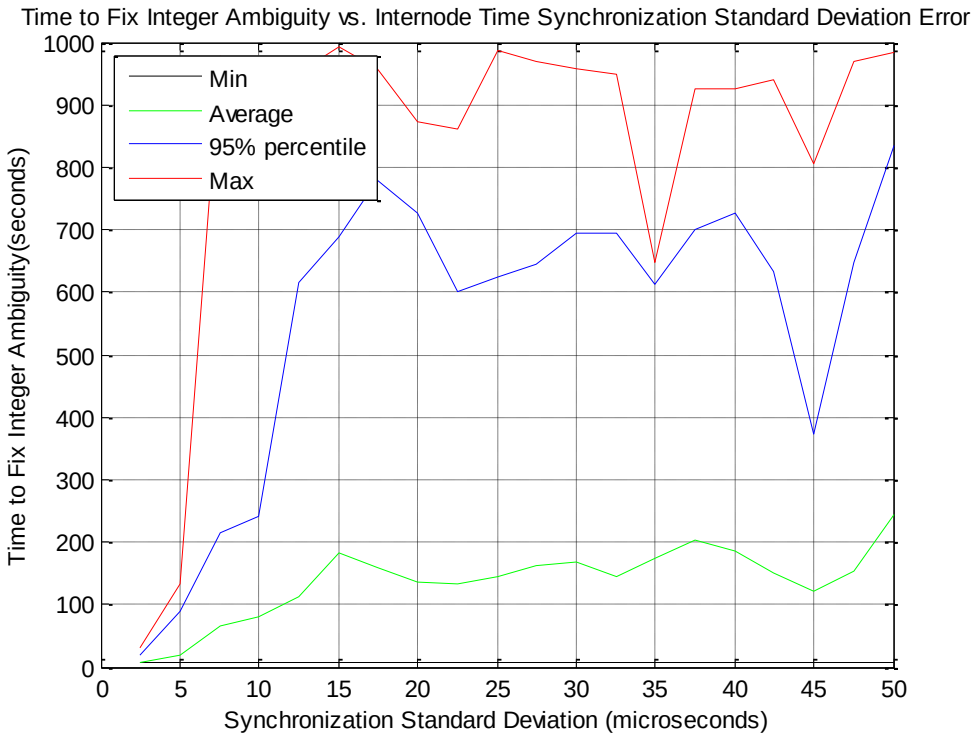


Figure 25 - Time to Fix Integer Ambiguity vs. Synchronization Error

Search Workload

To close this analysis a last figure of merit will be looked at, namely the total number of branches searched during the LAMBDA algorithm, before ambiguity fixing. It is quite cumbersome to keep track of all operations of integer transformations, and matrix inversions, so it was deemed more appropriate to keep track of all integer combinations after decorrelation operation that have been explored before, either resolution or exhaustion of the total number of seconds allowed per run. The limit of 7000 branches is an artificial number imposed by the run duration and the number of visible satellites, in practice meaning that no ambiguity was found.

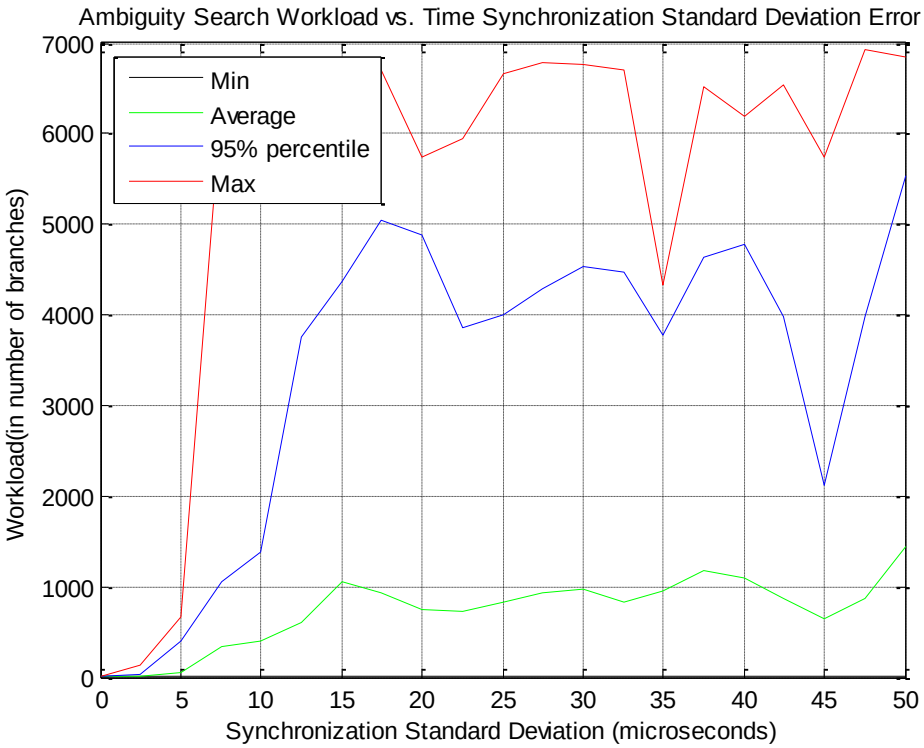
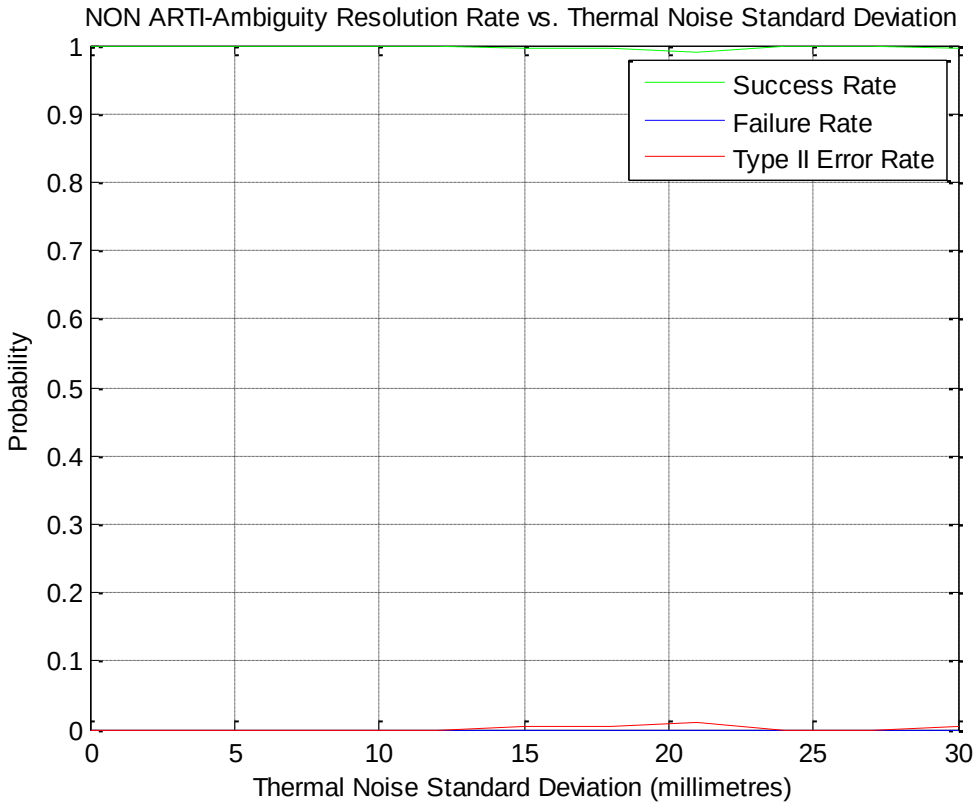


Figure 26 - Search workload vs. inter node time synchronization error

3.5 Influence of Thermal Noise

To demonstrate the influence of the thermal noise on the ambiguity fixing process, a systematic simulation was run with 0 microseconds of synchronization error and with thermal noise standard deviations varying between 0 and 30 millimetres in 3 millimetres steps. This range is believed to cover well the extent of the thermal noise standard deviations found in practice. The Time to-Fix-Ambiguities results reported in Figure 28 show a regularly growing mean TTA at about 10 seconds to around 5 millimetres, that

goes up to 40 seconds for 20 millimetres of thermal noise. The ambiguity resolution rate is reported in Figure 27 and the baseline position error in Figure 29. The position errors results are computed only for the cases with reported successful ambiguity fixing. The region with no noticeable effects on all three criteria mentioned earlier ranges from 0 to 5 mm.



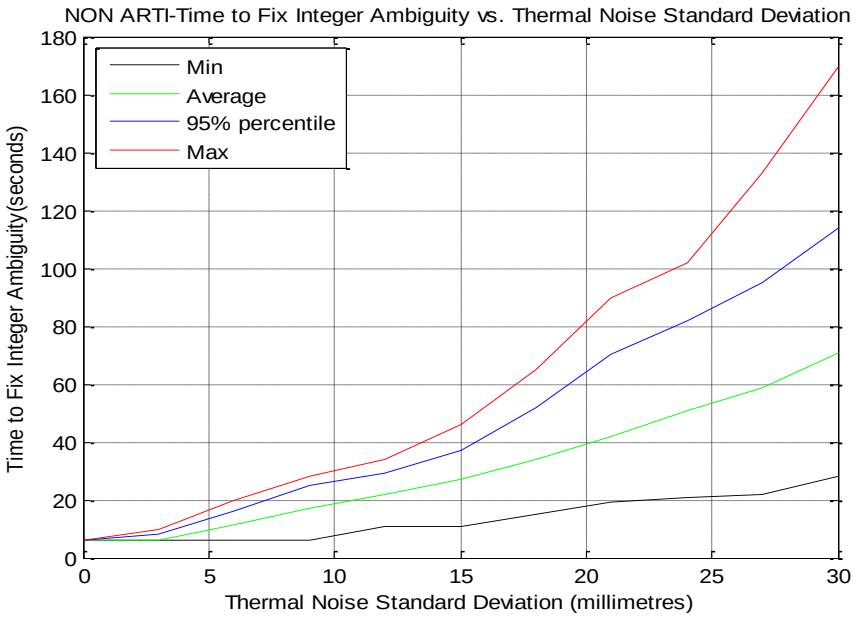


Figure 28 - NON ARTI - Time to Fix Ambiguities vs. Thermal Noise

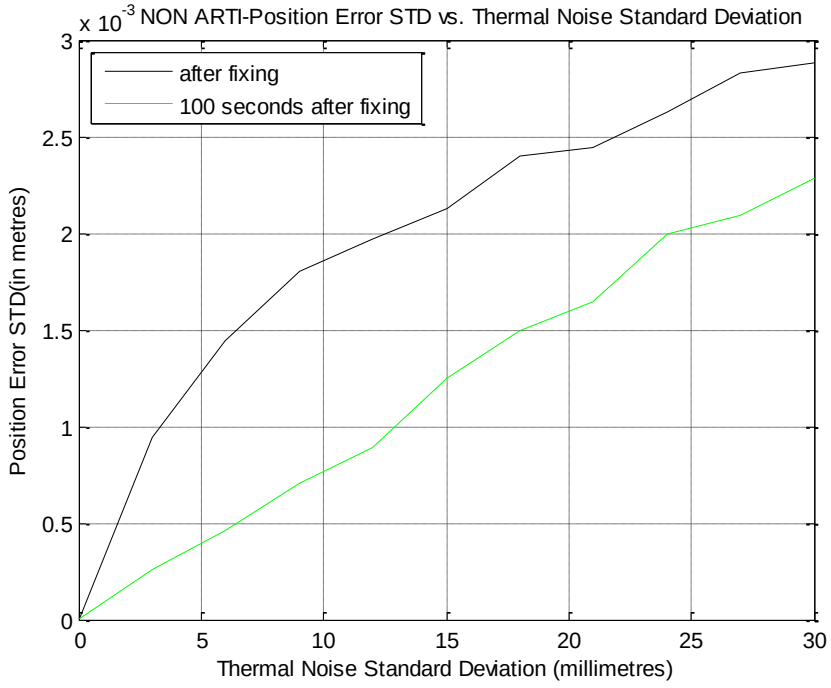


Figure 29 - NON ARTI - Position Error vs. Thermal Noise

3.6 Chapter Summary

After having demonstrated that Double Difference observables are sensitive to the differential time error between receivers at both ends of a baseline, this chapter described the mathematic foundations of the relative carrier phase positioning algorithms implemented in the simulations. The capabilities of the scenario simulator were described and the extreme sensitivity of the time error difference between receivers was confirmed by simulation to be noticeable in all performance parameters even for a difference a low as 2.5 microseconds.

CHAPTER 4

AMBIGUITY RESOLUTION of TIME INTEGER

After demonstrating in Chapter 3 that the synchronization accuracy delivered by conventional WSN time synchronization protocols is insufficient to support relative carrier phase positioning, a novel technique termed "Ambiguity Resolution of Time Integer" (ARTI) will be introduced. All mathematical implementation details of ARTI will be exposed in this chapter. The chapter will close by the presentation of simulation results of the ARTI technique, and will compare them against the limitations of the non-ARTI method results. It noted that there is a worldwide patent pending on the ARTI method.

4.1 Generalities

ARTI delivers 3D distance between 2 nodes with a sub-centimetre accuracy within 5-10 seconds after satellite acquisition, while breaking the usual timing requirements of 3 microseconds differential time error and 10 ms common time error as developed in Section 3.1. Both receivers need to be in the vicinity of each other (3-5 km). Both receiver nodes need to be wirelessly connected either to each other, directly or indirectly, or to a common position determination equipment. The present algorithm assumes a

reference receiver node that is accurately (sub microsecond) synchronized on GPS time and a secondary receiver that is loosely synchronized to GPS time by external means, i.e. wireless. So far, the study has demonstrated that ARTI can tolerate up to 100 ms of synchronization error as will be elaborated later in this chapter. Both receivers collect cumulated carrier phase information. The reference receiver collects them at a very accurate GPS time, and the rover collects them with a time tag uncertainty up to 100 ms, consistent with the synchronization error.

The secondary receiver is assumed to have limited processing capabilities, and can synchronously sample cumulated carrier phase on all visible satellites at an approximately known GPS time (within 100 ms). For accurate timing purposes that will become clear later, it needs to collect at the same time the sub millisecond code phase offset on the same satellites.

4.2 Formulas for transmit time at secondary receiver

A signal satellite transmit time approximation is equal to receive time ($t_{sync,second}$), minus average geometric distance converted into time by multiplication with velocity of light (t_{flight}). Thus, rounded to an integer millisecond, the approximate transmit time of a signal from a satellite received at the secondary receiver at $t_{sync,slave}$ is:

$$t_{s,second}^{approx} = round\left[\frac{\left(t_{sync,second} - t_{flight}\right)}{10^{-3}}\right].10^{-3} \quad (4.1)$$

where

$t_{s,second}^{approx}$ is approximate GPS transmit time for a signal received at secondary receiver rounded as an integer number of milliseconds (in seconds),

$t_{sync,second}$ is reception time at secondary receiver synchronized from master receiver (in seconds),

t_{flight} is satellite-receiver average time of flight ($t_{flight} \approx 75 \cdot 10^{-3} seconds$).

In the above formula, the flight time is approximated by an average value of 75 milliseconds to reduce the asymmetry and the range of values that transmit time correction can take, but it is not mandatory to compensate for it. It has to be noted that even if the approximate transmit time is rounded to an integer millisecond, the difference between this approximate time and the true transmit time can be larger than several milliseconds.

In the Carrier Phase Double Differencing technique, the satellites are partitioned between the reference satellite and other satellites. In the single differencing operation, every measurement of the integrated (i.e. cumulated) carrier phase for the non-reference satellite is subtracted from the reference carrier phase measurement. The following references to the “reference satellite” pertain to the same satellite in the double differencing operation. The reference satellite transmit time of a signal received at the secondary receiver can be precisely expressed in seconds from the beginning of the week. For convenience and without loss of generality, the reference satellite is assumed to have index "1" throughout the whole chapter:

$$t^1_{s,second} = t^{approx}_{s,second} + N^1_{s,second} \cdot 10^{-3} + C^1_{r,second} \cdot 10^{-3} - dt_1 \quad (4.2)$$

where

$t^1_{s,second}$ accurate Reference Satellite transmit time of the signal received at $t_{sync,second}$ at secondary receiver (in seconds),

$t^{approx}_{s,second}$ approximate transmit time for a signal received at secondary receiver rounded to an integer number of milliseconds (in seconds),

$N^1_{s,second}$ unknown integer number of milliseconds of correction to the transmit time of Reference satellite (no units),

$C^1_{r,second}$ sub millisecond PRN code offset of the Reference satellite at the secondary in fraction of one millisecond (no units),

dt_1 REF satellite clock error vs. GPS system time. It is found in the broadcast ephemeris (in seconds).

4.3 Inter satellite transmit time differences at Secondary receiver

Satellite transmit times of all remaining M-1 signals received at the secondary receiver can be expressed by similar formulas, but in order to reduce the number of unknowns to only one ($N^1_{s,slave}$) and to keep the problem solvable, transmit times for the remaining M-1 signals are expressed as corrections applied to the transmit time of the reference signal as follows:

$$\begin{cases} t^{2}_{s,second} = t^{1}_{s,second} - C^{1}_{r,second}.10^{-3} + dt_1 + N^{1,2}_{second}.10^{-3} + C^{2}_{r,second}.10^{-3} - dt_2 \\ \vdots \\ t^{(M-1)}_{s,second} = t^{1}_{s,second} - C^{1}_{r,second}.10^{-3} + dt_1 + N^{1,(M-1)}_{second}.10^{-3} + C^{(M-1)}_{r,second}.10^{-3} - dt_{(M-1)} \end{cases} \quad (4.3)$$

where

$t^{1}_{s,second}$ reference (i.e. first) satellite transmit time received at the secondary receiver (in seconds),

$C^{1}_{r,second}$ sub millisecond code offset of the first (reference) satellite at the secondary in fraction of one millisecond (no units),

$t^{second}_{s,(M-1)}$ (M-1) th satellite transmit time received at the secondary receiver (in seconds),

$N^{1,2}_{second}$ integer number of milliseconds of correction of second satellite transmit time at secondary minus reference satellite transmit time (no units),

$C^{2}_{r,second}$ sub millisecond code offset of the first non reference satellite at the secondary in fraction of one millisecond (no units),

$C^{(M-1)}_{r,second}$ is the sub millisecond code offset of the (M-1) th non reference satellite at the secondary in fraction of one millisecond (no units),

$dt_2, \dots, dt_{M-1}$ second to M-1 satellite clock error vs. GPS system time. It is found in the broadcast ephemeris (in seconds).

The integer millisecond transmit time differences between satellites are unambiguously computed at the Master Receiver from the integer transmit time differences as follows:

$$N_{Master}^{1,n} = round\left[\left(t_{s,Master}^{n}.\,10^{3} - C_{r,Master}^{n}\right) - \left(t_{s,Master}^{1}.\,10^{3} - C_{r,Master}^{1}\right)\right] \tag{4.4}$$

where

$N_{s,Master}^{1,n}$ integer number of milliseconds between nth satellite code offset reference point minus first (i.e. reference) satellite code offset point (no units),

$t_{s,Master}^{n}$ transmit time of the nth satellite received at the Master (in seconds),

$C_{r,Master}^{n}$ sub millisecond code offset for the nth satellite received at the master in fraction of millisecond (no units),

$t_{s,Master}^{1}$ transmit time of the first satellite received at the Master (in seconds),

$C_{r,Master}^{1}$ sub millisecond code offset for the nth satellite received at the master in fraction of one millisecond (no units).

The assumption that:

$$N_{second}^{1,n} = N_{Master}^{1,n} \tag{4.5}$$

is valid if the distance between the master receiver and the secondary receiver is much less than 300 km (or 1 millisecond in time equivalence), and the GPS time difference

between when the estimation has been done at the master receiver and used at the secondary receiver is less than 300km/4/3.5km/s ≈ 20 seconds. In other words, in order for the 1ms transmit ambiguity at the secondary receiver $N_{second}^{1,n}$ to be the same as $N_{Master}^{1,n}$ estimated at the master, either the master is no more than 300 km away from the secondary, and the estimation is made virtually at the same time as it is used at the secondary, or the master and secondary are virtually collocated and the time difference is no more than 20 seconds. All these assumptions are valid in the context of the WSN that has been introduced earlier.

Eventually, all transmit times can be expressed in terms of the single unknown variable, $N_{s,second}^{1}$:

$$\begin{cases} t_{s,second}^{1} = t_{s,second}^{approx} + N_{s,second}^{1} \cdot 10^{-3} + C_{r,second}^{1} \cdot 10^{-3} - dt_1 \\ t_{s,second}^{2} = t_{s,second}^{approx} + N_{s,second}^{1} \cdot 10^{-3} + N_{second}^{1,2} . 10^{-3} + C_{r,second}^{2} . 10^{-3} - dt_2 \\ \vdots \\ t_{s,second}^{(M-1)} = t_{s,second}^{approx} + N_{s,second}^{1} \cdot 10^{-3} + N_{second}^{1,(M-1)} . 10^{-3} + C_{r,second}^{(M-1)} . 10^{-3} - dt_{(M-} \end{cases} \quad (4.6)$$

4.4 Carrier Phase Observation Formula for the Master Receiver

The master receiver is referenced to its own receive time. The classical cumulated carrier phase equation in units of carrier phase cycles at L1 is given by (modified from Leick (2004), page 173, Equation 5.10):

$$\varphi_{master}^{p}(t_{r,master}) = \frac{f_1}{c} . \rho_{master}^{p}(t^p) + N_{master}^{p} - \quad (4.7)$$

$$f_1 . dt_{master} + f_1 dt^p + I_{master,\varphi}^{p}(t_k) + \frac{f_1}{c} . T_{master}^{p} +$$

$$\delta^{p}_{k,\varphi}(t_k) + \varepsilon_{\varphi}$$

where

$\varphi^{p}_{master}(t_{r,master})$	Carrier phase on L1 of pth satellite received at master receiver (in L1 cycles),
$t_{r,master}$	True Receive time at master at measure of φ^{p}_{master} (in seconds),
f_1	Nominal L1 frequency (in Hz),
c	Velocity of light (m/s),
$\rho^{p}_{master}(t^p)$	Geometric distance traveled by the code from transmission of satellite p to reception at master receiver antenna (in metres),
N^{p}_{master}	Integer Ambiguity of pth satellite at master. It remains constant during the period of observation (no units),
dt_{master}	master receiver clock time error (in seconds),
dt^p	pth satellite clock time error (in seconds),
$I^{p}_{master,\varphi}(t_k)$	pth satellite Ionospheric L1 carrier phase advance at master (negative value) (in L1 cycles),
T^{p}_{master}	pth satellite Tropospheric delay (always positive)(in metres),

$\delta^{p}_{master,\varphi}(t_k)$ Hardware delays and multipath delays on the L1 carrier phase (in L1 cycles),

ε_{φ} L1 phase measurement noise (in L1 cycles),

It is converted into its length equivalent formula:

$$\begin{aligned}\Phi^{p}_{master}(t_{r,master}) &= \rho^{p}_{master}(t_{r,master}) + \lambda_1 . N^{p}_{master} \\ &- c.dt_{master} + c.dt^{p} + I^{p}_{master,\Phi}(t_{r,master}) \\ &+ T^{p}_{master}(t_{r,master}) + \delta^{p}_{master,\Phi}(t_{r,master}) \\ &+ \varepsilon_{\Phi}\end{aligned} \tag{4.8}$$

where

$\Phi^{p}_{master.l}(t_{r,master})$ cumulated carrier phase measurement at $t_{r,master}$ at the master receiver at frequency L1 from the satellite p (in metres),

$t_{r,master}$ Receive time at the master receiver from p[th] satellite,

$\rho^{p}_{master}(t_{r,master})$ Geometric distance traveled by the signal from transmission at satellite p and received at master receiver antenna at $t_{r,master}$ (in metres),

λ_1 L1 Frequency Wavelength (in metres),

N^p_{master}	Number of cycle integer ambiguities between p satellite and master receiver at L1 frequency. Fixed number, unless cycle slips (no units),
c	Velocity of light (in m/s),
dt_{master}	Master receiver clock error vs. GPS system time at receive time (in seconds),
dt^p	p[th] satellite clock error vs. GPS system time. It is found in the broadcast ephemeris (in seconds),
$I^p_{master,\Phi}(t_{r,master})$	Ionospheric delay at receive time between satellite p and master receiver (in metres),
$T^p_{master}(t_{r,master})$	Tropospheric delay at receive time $t_{r,master}$ between satellite p and master receiver (in metres),
$\delta^p_{master,\Phi}(t_{r,master})$	Hardware delay and multipath effects on the L1 carrier phase in master receiver (in metres),
ε_Φ	L1 carrier phase measurement noise in master receiver (in metres).

4.5 Carrier Phase Observation Formula for the Secondary Receiver

Expressing cumulated carrier phase in metres, and referring the measurements to the transmit time (different for each satellite p, even when received at the same time at the secondary), the cumulated carrier phase is converted into:

$$\Phi^{p}_{second}\left(t^{p}_{s,second}\right) = \rho^{p}_{second}\left(t^{p}_{s,second}\right) + \lambda_{1}.N^{p}_{second} - c.dt_{second} + c.dt^{p} + I^{p}_{second\Phi}\left(t^{p}_{s,second}\right) + T^{p}_{second}\left(t^{p}_{s,second}\right) + \delta^{p}_{second,\Phi}\left(t^{p}_{s,second}\right) + \varepsilon_{\Phi} \quad (4.9)$$

where

$t^{p}_{s,second}$ Transmit time at the satellite p of the signal received at the secondary receiver,

$\Phi^{p}_{second}\left(t^{p}_{s,second}\right)$ Cumulated carrier phase measurement at the secondary receiver at frequency L1 of a signal that was transmitted at $t^{p}_{s,second}$ from the satellite p (in metres),

$\rho^{p}_{second}\left(t^{p}_{s,second}\right)$ Geometric distance traveled by the signal from transmission at satellite p at time $t^{p}_{s,second}$ to reception at secondary receiver antenna (in metres),

λ_{1} L1 Frequency Wavelength (in metres),

$N^p_{second}(l)$	Number of cycle integer ambiguities between p satellite and secondary receiver at l=L1 frequency. Fixed number, unless cycle slips (no units),
c	Velocity of light (in m/s),
dt_{second}	Receiver clock error vs. GPS system time at receive time (in seconds),
dt^p	p[th] satellite clock error vs. GPS system time. It is found in the broadcast ephemeris (in seconds),
$I^p_{second,l,\Phi}(t^p_{s,second})$	Ionospheric delay at transmit time t^p of the distance between satellite p and secondary receiver (in metres),
$T^p_{second}(t^p_{s,second})$	Tropospheric delay at transmit time t^p of the distance between satellite p and secondary receiver (in metres),
$\delta^p_{second,l,\Phi}(t^p_{s,second})$	Hardware delay and multipath effects on the L1 carrier phase (in metres),
$\varepsilon_{l,\Phi}$	L1 carrier phase measurement noise (in metres).

An important point to note that in this formula the geometric distance ρ^p_{second} is parametrized by $t^p_{s,second}$ instead of $t_{r,second}$ as usual.

As the true receive time at the secondary is not known, the Ionospheric, Tropospheric and satellite time corrections are computed at the approximate transmit time. This

simplification is justified by the slow variation of I, T and δ vs. time, and the short difference between t^p and t_{second}:

$$\begin{aligned} I^p_{second,\Phi}(t_{r,second}) &\cong I^p_{second,\Phi}(t^p_{s,second}) \\ T^p_{second}(t_{r,second}) &\cong T^p_{second}(t^p_{s,second}) \\ \delta^p_{second,\Phi}(t_{r,second}) &\cong \delta^p_{second,\Phi}(t^p_{s,second}) \end{aligned} \tag{4.10}$$

4.6 Double Difference Observables

The double difference observables, expressed for direct cumulated carrier phases observables are:

$$\begin{aligned} \Phi^{pq}_{ms}(t^p_{s,second}, t^q_{s,second}, t_{r,master}) & \\ = \Phi^p_{master}(t_{r,master}) - \Phi^p_{second}(t^p_{s,second}) & \\ - \Phi^q_{master}(t_{r,master}) + \Phi^q_{second}(t^q_{s,second}) & \end{aligned} \tag{4.11}$$

The major point is that all measurements made at the master receiver are time-tagged with the reception time $t_{r,master}$, whereas the measurements at the secondary are time-tagged at their own transmit time. As each satellite lies at a different distance from the secondary receiver, each measurement has a different time tag, i.e. $t^p_{s,second}$ for the pth satellite, and $t^q_{s,second}$ for the qth satellite. It should be clear that all measurements at the secondary receiver are made simultaneously, but the transmit times are spread over an interval.

$t_{r,master}$ is know very accurately, because the master receiver is a full receiver implementation, and estimates its own clock error as a by-product of a single positioning algorithm.

$t_{r,second}$ is very imprecise, being obtained only by inter-node synchronization. Moreover, $t^{p}_{s,second}$, transmit time at the pth satellite for the secondary, is composed of a sub millisecond contribution that is known very accurately, and an unknown number of integer milliseconds. The sub millisecond contribution is simply the phase of the PRN code used for despreading at the secondary, that can be directly read from the state of the local PRN code generator.

The whole idea behind ARTI is to take advantage of the accurately known (within the millisecond) but ambiguous (in multiple milliseconds) transmit times to compute the accurate transmit times without degradation of the relative position accuracy. The extra unknown becomes another integer ambiguity (of time) that will be resolved along with the other ambiguities (of carrier phase). All solutions that have been devised for carrier phase ambiguity resolution can be applied to this mixed ambiguity vector without any further mathematical complexity.

4.7 Linearized Equations

The following sections just amplify and detail what has been stated in the earlier section. The times around which the double difference formulas are linearized are multiple, the

receive time for the master receiver, and one approximate transmit time per satellite for the secondary receiver:

$$t^{p}_{s,second,0} = t^{approx}_{s,second} + N^{1,p}_{second}.10^{-3} + C^{p}_{r,second}.10^{-3} - dt_p \tag{4.12}$$

$$t^{q}_{s,second,0} = t^{approx}_{s,second} + N^{1,q}_{second}.10^{-3} + C^{q}_{r,second}.10^{-3} - dt_q \tag{4.13}$$

The linearization point of the double difference carrier phase is

$$\Phi^{pq}_{ms}\left(t_0^{p,second}, t_0^{q,second}, t_{r,master,0}\right) \tag{4.14}$$

where

$t_{r,master,0}$ known sampling time at the master receiver

The measured double difference carrier phase $\Phi^{pq}_{ms}(x, y, z, N_{second})$ can be expressed as the usual Taylor series. The unknowns to be resolved are the 3D position errors (dx, dy, dz), the double difference carrier phase ambiguities N^{pq}_{ms} and the single time ambiguity $\mathrm{N_{second}}$:

$$\begin{aligned}&\Phi^{pq}_{ms}(x, y, z, N_{second}) = \\ &\Phi^{pq}_{ms}\left(t_0^{p,second}, t_0^{q,second}, t_{r,master,0}\right) + \frac{\partial \Phi^{pq}_{ms}}{\partial x}.dx + \\ &\frac{\partial \Phi^{pq}_{ms}}{\partial y}.dy + \frac{\partial \Phi^{pq}_{ms}}{\partial z}.dz + \frac{\partial \Phi^{pq}_{ms}}{\partial \mathrm{N}}.d\mathrm{N_{second}} + \lambda_1 .N^{pq}_{ms}\end{aligned} \tag{4.15}$$

Each line of the linearized matrix can be expressed as:

$$\Phi_{ms}^{pq}(x,y,z,N_{second})-\Phi_{ms}^{pq}\left(t_{s,second,0}^{p},t_{s,second,0}^{q},t_{master,0}\right) \tag{4.16}$$
$$=\frac{\partial\Phi_{ms}^{pq}}{\partial x}.dx+\frac{\partial\Phi_{ms}^{pq}}{\partial y}.dy+\frac{\partial\Phi_{ms}^{pq}}{\partial z}.dz$$
$$+\frac{\partial\Phi_{ms}^{pq}}{\partial \mathrm{N}}.d\mathrm{N}_{\mathrm{second}}+\lambda_1.N_{ms}^{pq}$$

With M satellites, in equivalent matrix notation (the subscripts are the matrix and vector dimensions), one can write:

$$B_{(M-1),1}=A_{(M-1),(3+1+M-1)}.X_{(3+1+M-1),1} \tag{4.17}$$

where

$$B=\begin{bmatrix}\Phi_{ms}^{12}(x,y,z,N_{second})-\Phi_{ms}^{12}\left(t_{s,second,0}^{1},t_{s,second,0}^{2},t_{master,0}\right)\\ \vdots\\ \Phi_{ms}^{1M}(x,y,z,N_{second})-\Phi_{ms}^{1M}\left(t_{s,second,0}^{1},t_{s,second,0}^{M},t_{master,0}\right)\end{bmatrix} \tag{4.18}$$

$$A=\begin{bmatrix}\frac{\partial\Phi_{ms}^{12}}{\partial x} & \frac{\partial\Phi_{ms}^{12}}{\partial y} & \frac{\partial\Phi_{ms}^{12}}{\partial z} & \frac{\partial\Phi_{ms}^{12}}{\partial N_{second}} & \lambda_1 & 0 & \dots & \dots & 0\\ \vdots & \vdots & \vdots & \vdots & 0 & \lambda_1 & \ddots & & \vdots\\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \lambda_1 & \ddots & \vdots\\ \vdots & \vdots & \vdots & \vdots & \vdots & & \ddots & \lambda_1 & 0\\ \frac{\partial\Phi_{ms}^{1M}}{\partial x} & \frac{\partial\Phi_{ms}^{1M}}{\partial y} & \frac{\partial\Phi_{ms}^{1M}}{\partial y} & \frac{\partial\Phi_{ms}^{1M}}{\partial N_{second}} & 0 & \dots & \dots & 0 & \lambda_1\end{bmatrix} \tag{4.19}$$

$$X^t=\begin{bmatrix}x_{second} & y_{second} & z_{second} & N_{second} & N_{ms}^{12} & \dots & N_{ms}^{1(M-1)}\end{bmatrix} \tag{4.20}$$

where

$x_{second}, y_{second}, z_{second}$	are the relative positions of secondary receiver vs. master receiver in ECEF cartesian coordinates system (in metres),
N_{second}	is the transmit time integer millisecond ambiguity (no units),
N_{ms}^{12} ,..., $N_{ms}^{1(M-1)}$	are the double difference integer ambiguities (no units).

The detailed formulas for the double difference partial derivatives introduced in the matrix A are found in Appendices 1 and 2.

4.8 Resolution of unknown parameters

From this point on, the technique is the same as the carrier phase relative positioning technique:

1)-Get the float solution by applying the usual least squares solution on the matrix equation:

$$X = (A^T.C.A)^{-1}.A^T.C.B \tag{4.21}$$

With C being the weight measurement computed as the inverse of the measurement variance / covariance matrix,

2)-Apply the lambda technique on the float ambiguities and the ambiguities variance/covariance matrix. The DD ambiguities and the transmit time integer ambiguities will be resolved simultaneously.

3)-Backpropagate the ambiguities corrections into the float ambiguities and correct the relative positions and transmit time.

4)-If necessary, compute the receive time (i.e. sample time) at the receiver by adding geometric distance, ionospheric, tropospheric and satellite time corrections.

4.9 Results

The simulation results will be now discussed. The same simulations as presented in Chapter 3 have been carried out in the same conditions, using ARTI. The exact locations of the nodes are different as they are randomly chosen in a volume of imposed dimensions, but the maximum allowed dimensions were the same, namely a cube of 100 metres in each dimension. The satellite configuration and the base location were the same.

All the following plots exemplify the behavior of the same quality parameters as in Chapter 3, as a function of secondary receiver synchronization standard deviation error in milliseconds on the x axis.

Figure 15 represents the ambiguity resolution success rate in green, and failure rate in blue. The success rate stays at nearly 100 % up to 100 ms or synchronization error, then shows some minimum failure rate of up to 2 % from 100 ms to 200 ms. This result is to

be compared to Figure 22, where the success rate is 100 % up to 5 microseconds, and falls down to 50 % at 27 microseconds. It is important to note that the of the x axis scale is **3 orders of magnitude** larger in Figure 30.

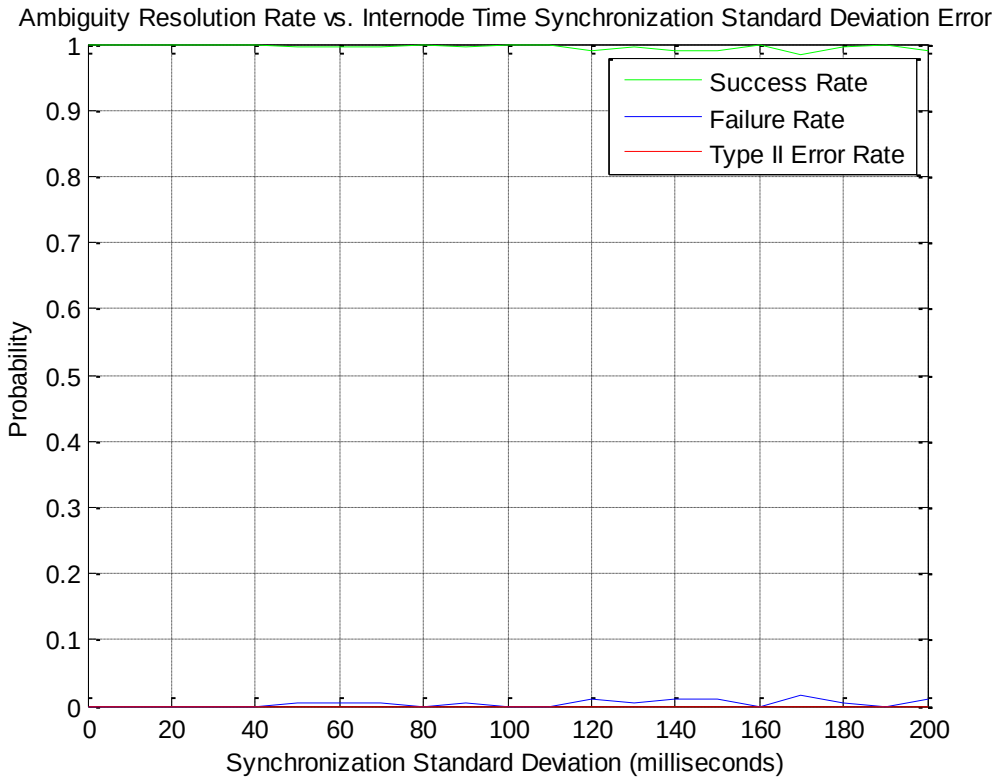


Figure 30 - ARTI Method - Success Rate

Figure 31 illustrates the minimum, average and maximum time to fix ambiguities. The average is less than 10 seconds up to 100 ms, then slowly increases to 20 seconds for 200 ms of synchronization error. The similar plot of Figure 25 exhibits a time to fix of about 10 s up to 2.5 microseconds, then dramatically increases around 150-200 seconds beyond

10 microseconds. The 2.5 microseconds situation is tolerable, but the best performance is for zero synchronization error.

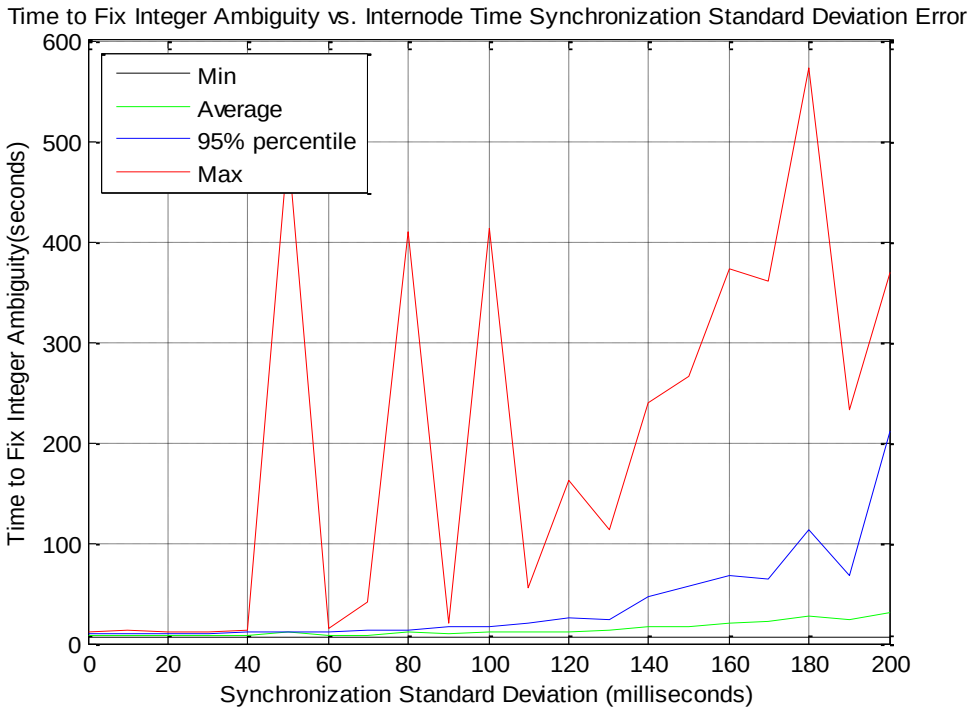


Figure 31 - ARTI Method - Time To Fix Ambiguities

The last plot in Figure 32 shows the number of potential ambiguities solutions to search for in the lambda solution before finding the correct one. This can be considered a figure of merit for the amount of computing resources that need to be dedicated to the search.

The ARTI solution has an average search of about 200 branches. The non ARTI case is about 1000 at 20 microseconds.

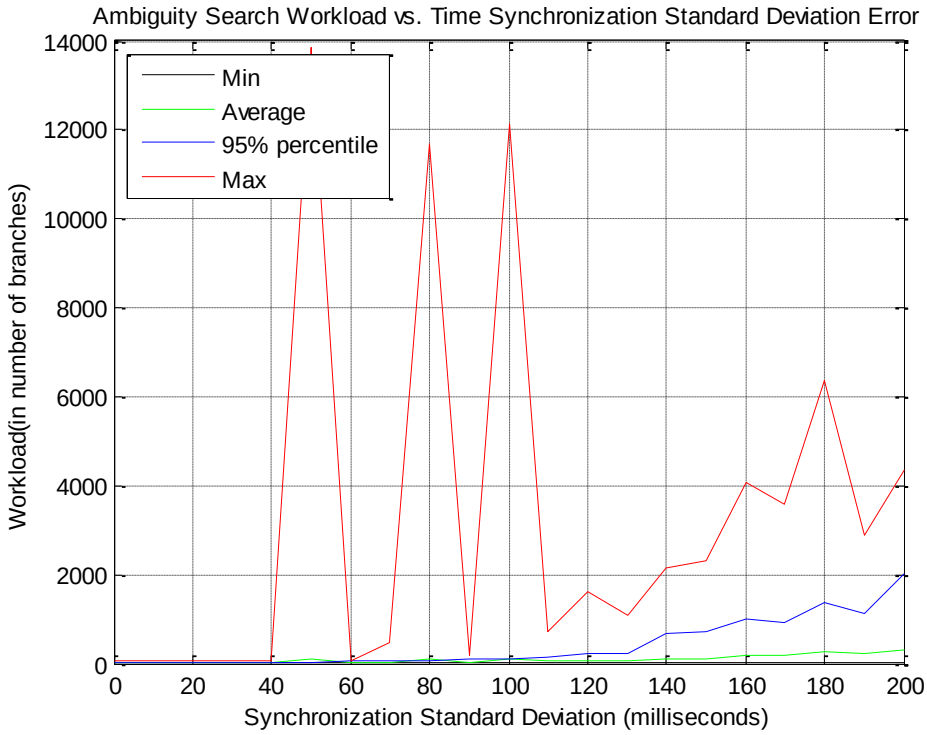


Figure 32 - ARTI Method - Number of Searches

4.10 Influence of Thermal Noise

Figure 33, Figure 34 and Figure 35 show the Ambiguity Resolution Rate, the Time-To-Fix the Integer Ambiguities and the Baseline Position Errors in the same simulation

conditions as in Section 3.5.

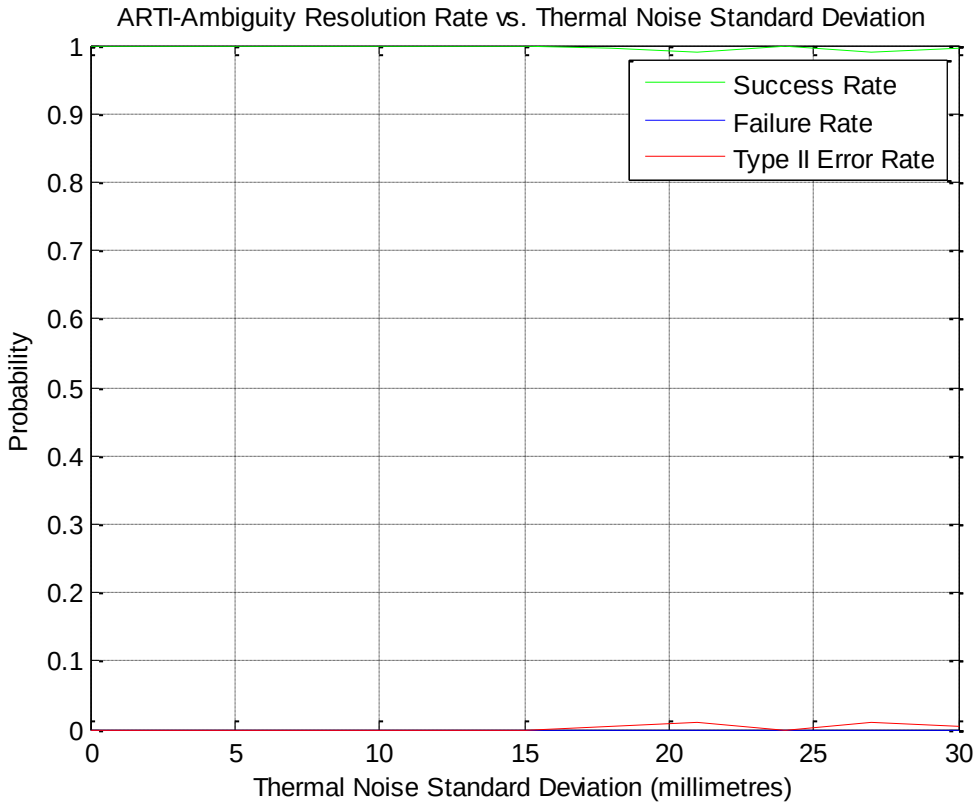


Figure 33 - ARTI - Ambiguity Resolution Rate vs. Thermal Noise

The emphasis will be put on the variations with the NON ARTI results. The ambiguity resolution rate is about the same, with Type II error rates starting to appear at about 20 mm. The Time-to-Fix Ambiguities mean value increases linearly at about 10-12 seconds for 5 millimetres and 60 seconds for 20 millimetres. There is a degradation compared to the NON ARTI case of about 10-20% at low noise levels, and up to 50% for noise levels at 20 mm. This can be straightforwardly explained by the extra time parameter that has to be estimated with the same number of measurements. The final position error at 2.5 mm seems not to be much worse in ARTI case. The position error improvement due to time

averaging after ambigutiy fixing is the most significant at 10 mm of noise standard deviation where it improves 1mm after 100 seconds. This time averaging effect is negligible at 30 mm of noise standard deviation. In summary, ARTI degrades gracefully over the thermal noise range.

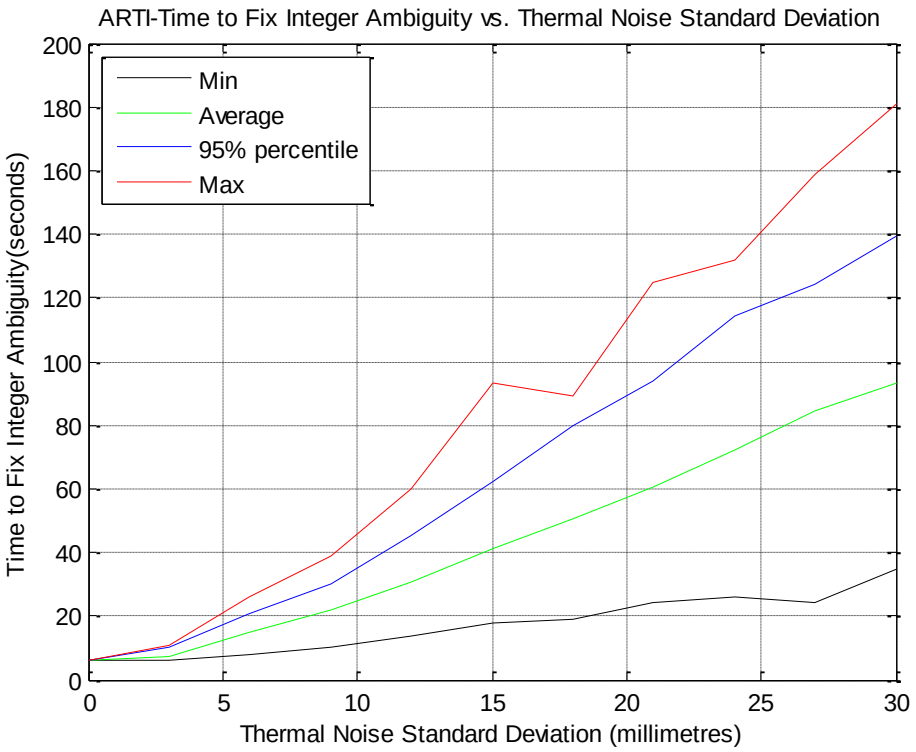


Figure 34 - ARTI Time-To-Fix Integer Ambiguities

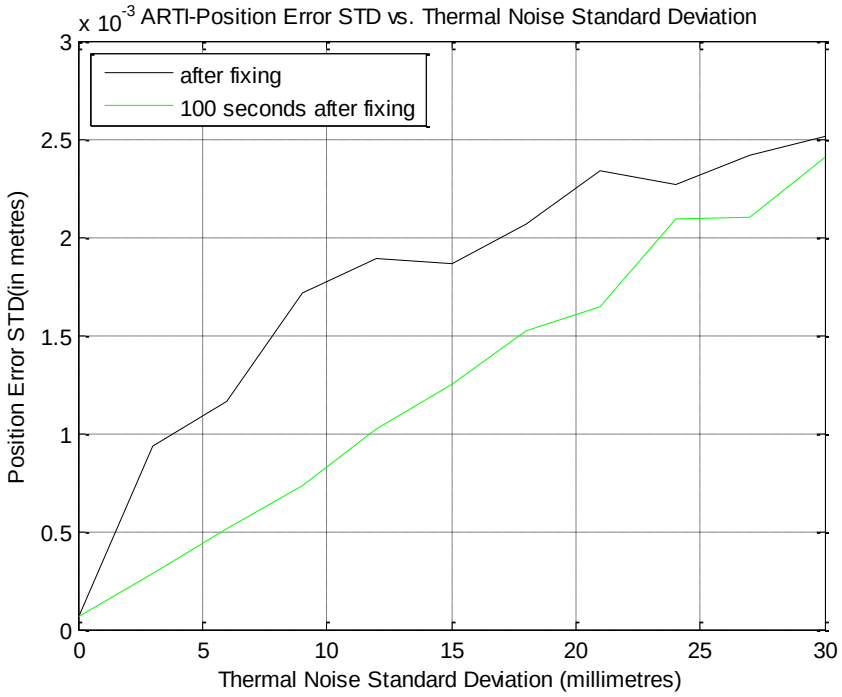


Figure 35 - ARTI - Position Error vs. Thermal Noise

4.11 Chapter Summary

The very accurate relative positioning process requires the use of carrier phase measurements. The carrier phase measurements are very precise with a thermal noise in the vicinity of 2-3 millimetres, but suffer from an unknown ambiguity of an integer number of times the wavelength of the carrier frequency (about 19 centimetres for GPS L1).

The very accurate 3D relative positioning of GPS or GNSS receivers is usually accomplished between full-fledged “surveyor grade” receivers with full processing capability.

The successful and correct determination of the DD integer ambiguities in the non ARTI case necessitates the **quasi simultaneous measurement of carrier phases at both ends of the baseline or the knowledge of the sampling time difference error within 5 microseconds**, otherwise the ambiguity resolution success rate drastically decreases.

ARTI technique presented in this chapter still performs with no notable degradation of performance with a straightforward inter node synchronization technique with errors of up to 100 ms of synchronization standard deviation. This accuracy is fairly easy to reach with about any time synchronization protocol that has been designed for Wireless Sensor Networks.

CHAPTER 5

WA-GPS WSN ARCHITECTURE

As shown in the previous chapter, multiple widely different WSN architectures have been proposed. Simplifying assumptions were made in order to keep the problem tractable and practical. In particular, all nodes are identical in hardware and software, and get the same initial energy budget. However, they can play different roles over time.

A special node, the gateway, or root node, has the special role, to interconnect the network to the rest of the world by internet or cellular connection. It benefits from unlimited power, as it would be usually connected to the power grid. It has the same standard GPS sensor function as all the other nodes. It has also the same wireless connectivity as all the nodes.

A given node is in wireless communication with only a subset of the neighbors. The average number of neighbors a given node directly connects to is called the connectivity index, and plays an important role in the robustness and protocol redundancy. A typical value for the connectivity index is between two and five.

The Network Architectures studied in this chapter will try to fulfill the seemingly contradictory constraints that are listed right after this introduction. The rest of this chapter will introduce the proposed hardware node, chosen for its relatively high performance, and characteristics of the current generation of nodes; some of the specifications will be used as parameters in the simulation of these solutions in the next chapters. Three WSN architectures will then be suggested. All are capable of delivering

the relative vector positions between gateway position and each node, the "Centralized", the "Decentralized", and the "Decentralized Clustered" Architectures. It will be soon obvious that each architecture will impose drastically different computational and wireless transmission constraints. To better cover and characterize these constraints, the few principal data flow categories within which all of the needed data flows will fall into will be described. The next step is an exhaustive introduction of all data flows present in all three architectures, and their intended roles. The implementation mode of each data flow for each architecture, their type and their relative differences will be explained. The chapter will conclude with a high-level description of the wireless protocols that will implement these data flows.

5.1 Constraints

Beyond data collection and delivery, the principal goal of a WSN architecture is to keep the network operational as long as possible. The first hurdle is the autonomous requirement wherein each node has a battery as single source of energy, that can be renewed only by energy scavenging, meaning it is directly collected in the physical environment of the node. The best example is solar power, but temperature differences, wind energy or mechanical vibration are also potential candidates.

Even power draw per node

A first critical requirement is to draw the same power for each node over a period of time. The premature energy exhaustion of some nodes would immediately translate into a loss

of nodes and a sparser and sparser network, with initially data collection loss at the lost nodes, and finally loss of connectivity with the remotest nodes.

This immediately translates in a simple assumption that no node can permanently assume a special function on top of its local GPS sensing function, such as rerouting measurement data to a central position computing facility. The closer the node is from the central facility, the more traffic the node will experience. This assumption can be relaxed by recognizing that an architecture where some nodes have a higher traffic processing load can still be accommodated, provided they swap this role with another node to achieve an even power drain over a longer period, if not instantaneous. This is the basis of the clustered architecture introduced further in this chapter. The same remark applies to local processing. No node can for example permanently process all measurement data of its neighbors for position extraction, on top of its own data collection and processing. The periodic swap of this function among neighbors is still possible. By straightforward arguments, it was demonstrated that only distributed data collection and processing is acceptable.

The reporting of the monitoring information is a special case, where an alarm can be generated from any node, and will have to be forwarded in a multi-hop fashion up to the gateway. This is inherent to this architecture, where the ultimate goal is to report events to the outside world. These events are few and hopefully with a large time interval in between, so that the stress on the network is not unbearable.

Per Node Conservative Energy Budget

A second requirement is to conservatively budget the energy consumption per node to maximize the lifetime. This translates into a wake up/sleep sequence of modes. The data collection, processing and transmission/reception activity are reduced to a very small percentage of the time, when all sections of the node are powered on. Most of the time, the node will sleep, with all functions powered down, except for a very low power clock in charge of waking up the node at predetermined intervals. This clock is critical; it is mandatory that each pair of nodes in direct RF visibility share common wake up periods, so that they can reestablish the wireless connection. This requires a correct adjustment between low power clock accuracies and wake up durations.

Several metrics to report the lifetime of a WSN have been proposed (Handy 2002). FND, or the time elapsed until First Node Dies, i.e. exactly one node has depleted its energy, is the most meaningful. The other metrics of HNA (Half Nodes Alive) and LND (Last Node Dies) occur when the operation of the WSN is seriously compromised and do not present the same interest. FND will be used throughout all the simulations.

Robustness

Even if in theory all dimensioning is made such as no node would prematurely die, because no maintenance or replacement is allowed in the field, the wireless message routing protocols need to be tolerant to the loss of some nodes, first by having enough wireless range to reach more than one single node, second having a capability to reroute the traffic around the dead nodes, de facto implementing an ad-hoc routing technique.

5.2 Hardware Architecture

5.2.1 Node Architecture

gives a sample of the current technologies available for the nodes today, and they range from the very advanced and sophisticated iMote, equipped with an ARMv5 with a maximum clock of 416 MHz and 16-32 bit data and instruction bus, with a consumption of 53 mA at 104 MHz, to the minimal TELOS-B powered by a TI MSP430 at a maximum of 12 MIPS and 1.8 mA at 8 MIPS. The intermediate category is well represented by the MICA II, powered by an ATMEL ATmega128L-16MHz, and a consumption of 8 mA at 16 MHz.

The MICA II architecture will be assumed in all simulations, but the study is parametric in nature, the simulation can be applied to any other node. In addition to the MICA II overall functionalities, the GPS receiver section consumption will be taken at 8 mA @ 3 V or 25 mW. Without justification, outside the scope of this dissertation, this is the state-of-the-art in embedded receiver power consumption. The GPS section runs only during the GPS data collection period along with the mote processor, but the position computation will be entirely supported by the mote processor at 8 mA @ 3 V.

Table 3 - Mote Specifications

	iMote	MICA II	TELOS-B
CPU			
Processor	Intel XSCALE PXA271-13-416 MHz	Atmel ATmega128L-16MHz	TI MSP430
Data Path	16-32bit-ARMv5TE	8bit-RISC	16bit-RISC
FPU(Floating Point Unit)	None	None	None
Throughput		16 MIPS	~12 MIPS
SRAM Memory	256 kB	4 kB	10 kB
SDRAM Memory	32 MB	NA	NA
FLASH Memory	32 MB	128kB(program)+512kB (logging)	48 kB(program) +1024 kB(logging)
Current Consumption Active Mode Full Power	53 mA @ 104 MHz	8 mA @ 16 MHz	1.8 mA @ 8 MHz
Current Consumption Active Mode Low Power	31 mA @ 13 Mhz	NA	NA
Current Consumption Deep Sleep Mode	390 µA	<15 µA	5.1 µA
OPERATING SYSTEM			
		TinyOS 1.1.11	TinyOS 1.1.11
RADIO			
Transceiver	TICC2420-IEEE802.15.4		TICC2420-IEEE802.15.4
Frequency Band	ISM 2.4-2.4835 GHz	868/916 MHz	ISM 2.4-2.4835 GHz
Data rate	250 kbps	38.4 kbps	250 kbps
Range (Line of Sight)	30 m (built-in antenna)	170 m (external antenna)	75-100 m (built-in antenna)

Table 3 - Mote Specifications (Continued)

	iMote	MICA II	TELOS-B
Current Consumption Tx/Rx	Tx+Rx: 13 mA	Tx: 27 mA/Rx: 10 mA	Tx+Rx: 23 mA
POWER			
Internal Batteries	3xAAA (1250 mA.H)	2XAA (2890 mA.h)	2xAA(2890 mA.h)
Battery Voltage	3x1.5 V	2x1.5 V	2x.5 V
LIFETIME			
Lifetime at 100% Duty Cycle-no Tx/Rx-full power	~1 day	~15 days	~67 days
Lifetime at 1% Duty Cycle-no Tx/Rx-full power	~98 days	~4 years	~18 years

5.3 Position Computation

The first question is to determine what kind of positioning is intended, and for what goal. A first intended usage of the network is either to report an event, or a physical measurement change detected at a particular node, along with the location of the node, either in absolute coordinates, or in coordinates relative to the reference gateway. Another end usage is to detect the change in a least one relative distance between nodes, and to report the location of the node (s) that observed this relative change.

5.3.1 General Considerations

Root node to any other node

The distance from a root node to other nodes is computed. In order to do so, measurements taken at the root node and measurements taken at the secondary node need to be combined in a single location, where the relative positioning will be made. This can be accomplished in two ways. The root node measurements can be "flooded" across the whole network, and each node does its own relative computation with the root node, using the broadcast measurements, combined with their own local measurements; the positioning processing load is very well distributed (one per node), and a well distributed wireless traffic (each node does one reception and one transmission per cycle). An alternative is that each node sends its own measurements to neighbors. The neighbors with lower hierarchic level aggregate received measurements with their own data before transmitting at their turn. There will be local redundancy (and local inefficiency) as there is no coordination between peer nodes to avoid the duplication of measurements in nodes at the same hierarchical level, but the redundancy will eventually eliminate itself. All computations will be made at the root node. This is the centralized architecture to be avoided.

Local node to local node

The distance from node to neighbouring node is computed locally. The minimum necessary traffic is reduced to the transmission of measurements from the node on one side of the baseline to the other side. The position computation is made at the head node, or main cluster. Depending on their energy level, the role between cluster head and cluster leaf (i.e. the computing/measuring node vs. measuring only node) can be reversed.

This technique looks more promising in terms of energy consumption, where all measurements are kept local, and the load is very well balanced between nodes. It looks less promising when one realizes that distance between any other nodes will be computed as a sum of baseline vectors, and that the distance error over the long baseline will be the sum of all individual errors along the path. On the other side, a lot of applications emphasize the relative accurate distance between local nodes (example beam forming), that will fare well with a local measurement technique. If the cumulated error becomes too high over the whole network, a network adjustment might be possible at the root node, using the individual measurements, to somewhat alleviate this problem.

How to avoid redundancy between nodes of the network

Before a first estimation of the distances between nodes is made, i.e. that the algorithm went through a full cycle first, there is no notion of Euclidian distance between nodes. The only information known a-priori is the minimum number of wireless hops from a node of interest to another node. To keep generality across all iterations, the neighbour nodes will have to be defined not using their Euclidian distance, but their hop distance. In this context, a node is a neighbour of another one if their shortest hop distance is 1. This definition has limitations, as the node distance can vary over time thanks to the propagation vagaries (one lost connection increases the distance at least 1 unit), even if the nodes are perfectly static. The Voronoi diagrams and Delaunay triangles become trivial in this limited distance norm. They cannot be used for reduction of the redundancy of the network.

A more suitable technique in the hop distance domain is the “spanning tree” where all nodes are connected together in a single domain, but where all cycles are eliminated. It is

worth to note that this works only because the measured baselines are in fact 3D oriented vectors in an ECEF reference system. If only the Euclidian distance was measured between nodes, much more redundancy would be needed (like the 3D Delaunay triangle technique) to resolve the 3D positions of each node. The minimum weight spanning tree will be constructed, with the minimum number of edges connected to any given node. The spanning tree could be built completely regardless of the network connectivity to keep the number of edges connected to a node to a minimum (i.e. 2 nodes separated by a distance of more than one hop could be connected by an edge), but for simplicity, and to reduce the number of messages, it is better to start from the connectivity diagram (showing only the connectivity edges), and eliminating the edges until there is a spanning tree left, with all edges matching existing radio connections, and connectivity number to a minimum per node. It will be shown how to build this tree, using only local information, and distributing the algorithm across the network.

5.3.2 Baseline Choice considerations

A secondary algorithm to clarify is how many baselines a given node must compute, given its place in the spanning tree and its energy depletion status. There are two competing requirements: a node being at the intersection of multiple edges will have to compute proportionally more baselines but this will compromise the ability to keep the network alive (failure will increase the risk to split the network into two independent non connected sub-networks). A good solution is a specialization of the nodes: the nodes on secondary branches will specialize in position computation, while those on the main chain will specialize in data transmission.

Because the network under consideration has a large set of ambiguity redundancies, and that rapid ambiguity fixing is of prime importance, the solution proposed by Luo & Lachapelle (2003) with the MULTIKIN technology has been adapted.

In a quick summary, MULTIKIN first attempts classical ambiguity resolution per baseline, where the notion of network and redundancy is absent. The outcome is a set of double difference ambiguities and the associated ratio test criterion per baseline (Leick 2004, p 278).

$$F = \frac{\Omega(\tilde{\mathrm{N}}^2)}{\Omega(\tilde{\mathrm{N}}^1)} \tag{5.1}$$

where

$\tilde{\mathrm{N}}^1$ is the best integer candidate,

$\tilde{\mathrm{N}}^2$ is the second best integer candidate

$\Omega(\tilde{N}) = (\tilde{N} - N_{float})^T C_{float|\tilde{N}}^{-1} (\tilde{N} - N_{float})$

is the weighted sum squared of ambiguity residuals

$C_{float|\tilde{N}}$ is the conditional covariance matrix for the float ambiguities

It then sequentially applies two tests for each group of ambiguities and their ratio test.

The first test is a closure test, named "necessity test" in the dissertation, where traversing all sides of a closed triangle, the sum of all ambiguities per double difference must be equal to zero. The second test is a sufficiency test, where each ratio test for each side of the triangle must individually be larger than a threshold F_{T1}. The second part is that the

sum of all ratio tests of the triangle must be larger than another threshold F_{T2}. In the traditional single baseline algorithm (Leick 2004), only the F_{T1} threshold is applied, with an empirical value of four. According to Luo & Lachapelle (2003), the F_{T1} threshold is relaxed down to a value of two in MULTIKIN. This relaxation is allowed by the network redundancy, which compensates for this weaker criterion. To conclude this brief explanation, if the two tests do not pass, MULTIKIN then takes advantage of the fact that every baseline is shared by two Delaunay triangles at most. A Delaunay triangle partition of a set of points is defined as a set of triangles such that no other point falls into their circumcircles. The minimum spanning tree, or shortest line connecting all points of a two-dimensional graph without closed loops can be simply constructed from a Delaunay triangle partition, which explains its importance. For each baseline, and each double difference ambiguity, the results coming from closures with both adjacent triangles are compared. If they are the same, the decision (positive or negative) is reinforced, otherwise the result is "undecided" and needs to collect more measurements. If there are at least three double differences across all satellites in agreement for this baseline, these best ambiguity candidates are selected as the best fixed solution. If at least three double differences disagree, the best integer ambiguity candidate is rejected and float ambiguities are kept for this baseline.

There are multiple reasons that prevent the application of the MULTIKIN technique without modification. MULTIKIN assumes mobile platforms whereas this is a pseudo static configuration. This observation alone would a-priori promise an easier implementation. Unfortunately, the other differences are not so favorable. MULTIKIN assumes a large redundancy (about 10 nodes), and a very large number of triangles on

each of the closure constraint is checked. In the WSN case, even if the number of nodes is significantly larger, by virtue of the distributed processing principle, no node will have access to the totality of the measurements. The closure constraint will have to be applied only to the nodes in direct radio visibility of each other. MULTIKIN also proposes to use a Delaunay partitioning of the network, in order to eliminate the multiple triangles having the same baseline. In a network partitioned with Delaunay triangles, each baseline will be shared with at most two triangles. The WSN is not conducive to a Delaunay partition. Only the relative distance to the gateway node or to the immediate neighbours will be known at each node, which does forbid systematic construction of a Delaunay network from local topology information. The construction of the triangles will be random in practice, and will be mapped on the radio connectivity; each two-way radio link will be a baseline candidate. Another consideration is the requirement of uniform processing power consumption. Even if a node has a high connectivity with its neighbours, the size of the triangle network cannot be arbitrarily increased. A better practice is to keep the number of processed triangles to a minimum, regardless of the connectivity.

Number of Triangles and Baselines

In view of the results published in Luo (2001), a decision on the size of the constraining network can be made. Test 1 on the impact of the number of platforms is the most relevant. In a scenario with a variable number of platforms, with one sigma carrier phase noise level at 1.9 mm, multipath error at 3.8 mm mild ionospheric and tropospheric effects, and with baseline lengths around 1.5 km, the results are shown in Table 4.

Table 4 - Time to Ambiguity Fixed improvements vs. Number of Platforms in MULTIKIN

Number of platforms	Number of Baselines (Number of triangles)	TTAF(s) Time To Ambiguity Fixed	Percentage Improvement vs. unconstrained 3 platforms
3 platforms unconstrained	3 (1)	41.7	0%
3 platforms constrained	3 (1)	28.2	32%
4 platforms constrained	5 (2)	25.3	40%
10 platforms constrained	18 (9)	24.1	42%

It is apparent that the improvement is significant at the first step, when taking advantage of the closure properties of the simplest case, namely that of a single triangle and three baselines. The incremental improvement thereafter is not so large, and the balance between complexity and performance is in favor of the single triangle. All the simulations will be made on the assumption of baseline computation by groups of three.

5.3.3 Proposed architectures

All the possible choices of the previous section can be summarized in three solutions, that will be obvious when reading the following descriptions.

A first simple solution (Centralized Architecture) would be to directly measure the oriented distance between each node in a Delaunay triangle configuration. A typical example of computed baselines and location of the baselines computation is depicted in

Figure 36. This would lead to an unacceptable volume of messages, and a guarantee to unequally exhaust the network finite energy, if the goal is to convey all measurements back to the gateway (message aggregation mode), and to a central computation architecture where the whole network can be resolved, and the outliers detection mechanism is the most efficient. The redundancy at each node can be exploited in a generalized adjustment procedure. The gateway would not need to send back any position to any node, as it is the end user of this information, and only the ID number of the triggered node would be sufficient to send back. In this configuration, a node cannot detect a relative variation of distance with another node, as neither previous nor current relative distances are known locally. The only solution is to periodically trigger a complete sequence and trigger very large energy consumption. This mode of Centralized Architecture will be analyzed along with the other recommended architectures as a performance baseline in order to compare all architectures.

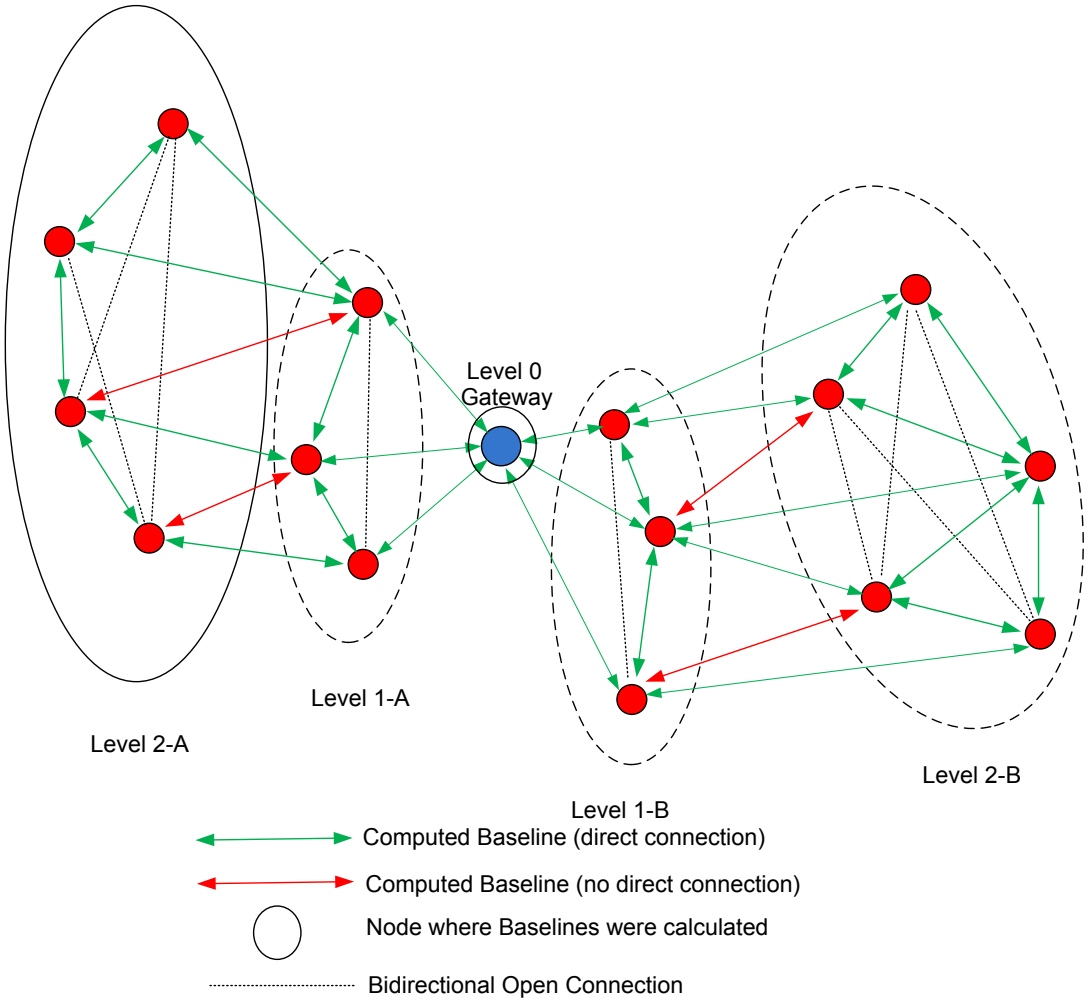


Figure 36 - WSN Centralized Architecture - Baselines Network

A second solution, illustrated in Figure 37 and referred to as "Decentralized Architecture" from now on, is to compute relative positions of each node vs. the location of one of the parents, and to create a particular type of spanning tree where all non cyclic edge sequences emanate from the gateway, and follows the general directions of connectivity between parents and children. This could be implemented without violating the rules stated earlier, in particular the non-use of aggregation messages. Only assistance from the gateway would be flooded to all nodes along the connectivity edges, and the same

information would be relayed from node to node. Each node would combine its own measurements with one parent's measurements, plus another node to provide a star like configuration set of baselines, where all resolved distances are indirectly from the gateway, and do not have knowledge about their closest neighbors. This implies that each local node computes its own relative position each time, regardless of its remaining energy status; the concept of cluster and rotating cluster head would be quite difficult to implement in this eventuality. This would also waive the capability to use other local measurements to create graph cycles to improve and accelerate the ambiguity resolution process.

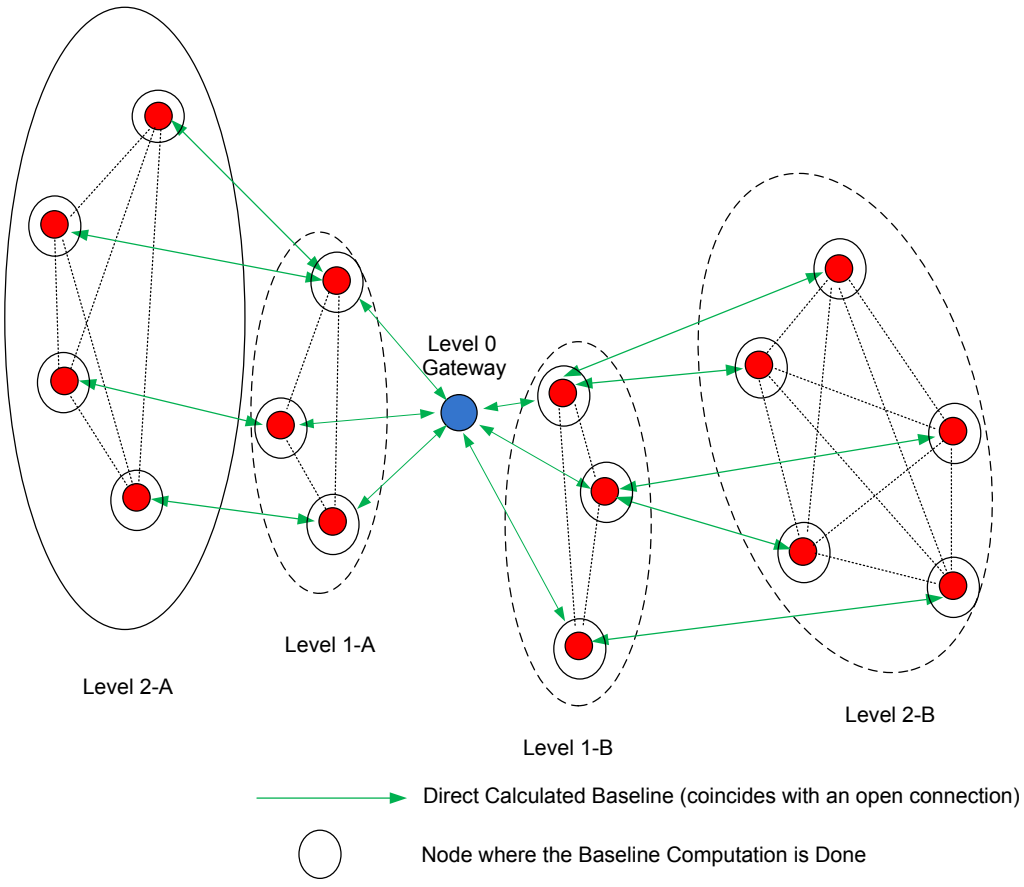


Figure 37 - WSN Decentralized Architecture - Baselines Network

A third solution, or "Decentralized Clustered Architecture" illustrated in Figure 38, is to compute relative distances only between directly connected nodes. This would imply the creation of local minimum span trees, implying only the nodes at a same level, plus a single edge with one of the parents. The cluster head selection can be simply done, relying only on direct connections, taking into account the relative remaining energy level in each node for determining the one that will perform the computations. The cluster head would compute its relative distance to all nodes of the same level, then forward the relative distances to each local node for local storage. When an event needs to be reported, the detecting node A will send its own relative position to another node B in the same chain, This next node will send the event to the next node C in the span tree and the combination of relative distance A-B (from message) and B-C (locally stored). The gateway will finally receive the notification of the event and the relative oriented distance from the triggering node to itself.

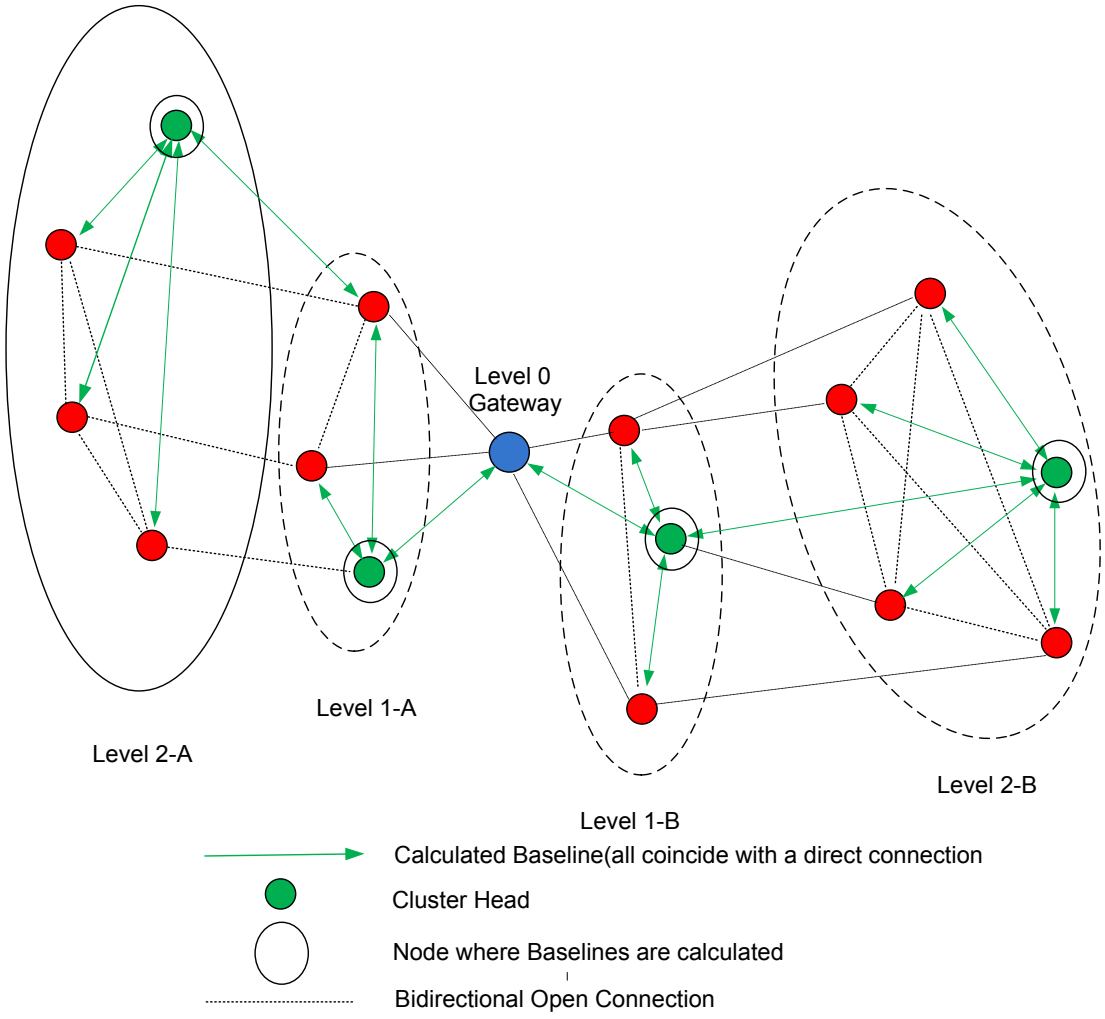


Figure 38 - WSN Network-Decentralized Clustered Solution - Baselines Network

5.4 Information Flow Categories

All types of information flow used herein to support this application fall into one of the following categories. They have widely different energy requirements. As the transmission channel is built on a common medium and a single frequency, packet collision is always a risk, and it implies a mechanism for collision detection, random time

backtrack and retransmission. All terminologies introduced below are not a research domain standard but are only for clarification.

Flooding sends the same information from a root node to all other nodes in the network. The typical use will be the dissemination of ephemeris or derived information and GPS assistance data to each node. The same information is relevant to all nodes. The transport effort per node is a single reception and retransmission. The message size is constant throughout the network, regardless of the hop distance to the root.

Aggregation concentrates individual information from every node into larger and larger messages until it reaches the root node. This would be the only solution for a centralized position computation at the root node. All node measurement messages are about the same size, and each node acting as a relay in the multiple hop transmission chain must aggregate several measurements received from other nodes and its own measurements before retransmission. The load is very unbalanced. The closer a node is to the root node, the higher is the transmission burden and power consumption. The larger the number of hops, the heavier is the transmission burden to the last nodes. In a hypothesis of uniform identical node architecture, this is a mode to avoid at any cost, as the WSN will cease to operate as soon as the nodes closest to the root are energy-depleted, and the connectivity with the farthest nodes is lost.

Disaggregation disseminates individual information to each node, from a single compound message created by the root node. This mode is the reverse case of aggregation and should be avoided for the same reasons. This is what would occur in case of tailored assistance information for each node. Besides the almost impossible root task

of keeping track of every node and of every node location in a centralized database to do this computation, it is very inefficient for each node. Either each node retransmits the same aggregated message, which is a very large energy waste at the end of the hop chain, or it has to know which nodes are downstream in the routing graph, and extracts only information relevant to these nodes before retransmitting. The aggravated requirement is that each node needs to locally keep track of the routing table for all downstream nodes. These multiple copies of partial routing tables at each node are very difficult to maintain up-to-date, especially if some nodes are lost. There is a significant possibility that some nodes will never receive any assistance information, and will be lost.

Short Broadcast sends information from a node to all neighbours with a maximum hop distance. All receiving neighbours resend the same information, but reduce the maximum hop distance by one unit, and do not retransmit if the hop distance is zero. The net effect is for all neighbours up to a maximum hop distance to receive this information. This is the preferred mode to exchange measurement data between neighbors for relative positioning. It must be coupled with a priority mechanism; only one node at each end of a baseline should do the computation, but information from a further distance allows the acceleration of the ambiguity resolution of the baseline, adding several closed loops and the associated constraints on the ambiguities including the baseline of interest. If the authorized number of hops is infinite, Flood from root node or Short Broadcast are equivalent.

5.4.1 Data Flow Enumeration

To support the three positioning modes previously introduced, all the relevant data flows will be defined. They are only related to the WSN activity, namely reception, transmission and computation tasks that take place in one of the connected nodes, including the gateway, which is assumed to benefit from unlimited computing power and energy resources. All data flows are summarized in Table 5.

Configuration Data Flow

Before the WSN settles in its steady state mode, the gateway needs to configure the nodes. The principal information is the periodicity of the position computation in seconds. It has to be accompanied by a synchronization phase where all local clocks of all nodes are synchronized to the gateway, before going to sleep. This is of the flooding type, as the same information is sent to each node.

Discovery Data Flow

In order to create clusters, and to guarantee the proper flow of data, each node needs to know its parents (nodes that relay to it gateway information), its peers (nodes with the same hierarchical level) and its children (node towards it will forward gateway information). This phase is meant to discover which local node is in direct wireless connection with others. This routing table is distributed throughout the whole network, and is not available at the gateway. It should be clear that this connectivity table is dynamic in nature. Changes can occur when two nodes are at the range limit, and lose connection due to fading, or when one node completely ceases to transmit altogether.

Time Synchronization Data Flow

In order to compute relative baselines, the nodes need to be synchronized to GPS time. The ARTI method significantly relaxes the synchronization demands, but a synchronization needs to take place with the gateway (the only one with true GPS time). This data flow assures synchronization in the first place. Synchronization is then maintained every time a position is computed.

Assistance Data Flow

The assistance data consists of visible satellites, and coefficients of the design matrix. The design matrix is computed at the gateway, with an assumption that the linearization point for each node position is identical to the gateway point. The same design matrix is flooded to every node that computes at least a baseline. The design matrix is not part of the assistance data flow in the Centralized Architecture. The number of satellites, and the satellite visibility list is implicit in the design matrix data for the other solutions.

Computing Entity Assignment Data Flow

This determines which node will compute the baselines at the current level. This phase exists only for the Decentralized Clustered Architecture.

Measurement Data Flow

Each time a measurement is done, the set of measurements at a node needs to be transmitted to the computing node. This can be a short broadcast or aggregation messages depending on the computing mode.

Position Reporting Data Flow

Sends back the relative positions to each node. Each node needs to know its own relative location to geolocate a local event, with further minimal interaction with the other nodes. This phase is not necessary for the Centralized Architecture.

Event Reporting Data Flow

This is a very infrequent event, when a physical parameter or the relative distance between nodes evolves, and needs to be reported to the gateway. It is not necessary for the centralized approach, if the node provides an ID number in the event, and the gateway does the association with the location upon reception of the event message.

Table 5 - Data Flows

Data Flow	Type	Periodicity	Conveyed Information
Configuration	Flooding	Once at start-up or at reconfiguration	Periodicity, and time of the next position computation
Discovery	Short Broadcast	At start-up, then combined with computing entity at each iteration	Number of hops from gateway
Time Synchronization	Short broadcast	At start-up only	Gateway GPS time
Assistance	Flooding	At each position computation	Design Matrix elements
Computing Entity Assignment	Short broadcast	At each position computation	
Measurements	Short broadcast or flooding	At each position computation	L1 Carrier Phase measurements for each visible satellite
Position Reporting	Short Broadcast or disaggregation	At each position computation	
Event Reporting	flood	Very infrequent	Position, and nature of the event, or ID and nature of the event.

5.5 Wireless protocols

5.5.1 MAC Layer

To keep this research at the highest applicability level possible, only some of the specifications necessary to carry on the study will be defined here. All nodes transmit at the same frequency, at a rate of 38.8 kbps in CSMA/CA mode. This means that all nodes share the same medium and same frequency. Before transmitting a packet, the node senses the medium and backs-off for a random amount of time if the channel is busy. The sender senses the channel again and starts to transmit if the channel is free. Because of the short but non zero time between sensing and transmitting, two nodes can overlap message transmission. The collision detection is done at the receiving node, and might trigger another request to send a Request To Send (RTS) packet. This approach is quite common in WSN and is perfectly aligned with the BERKELEY MICA implementation specifications. The CSMA/CD mode is not applicable in WSN for the same reason as that for all IEEE 802 wireless protocols, namely a node transceiver cannot transmit and sense the transmission medium activity at the same time.

Hidden Node Problem

There is an important problem with the CSMA/CA mode, where a "hidden" node can prevent a pair of nodes to communicate properly (Ilyas & Mahgoub 2005, "Section 18.4.2.2: Distributed MAC Protocols"). Two nodes A and B, not in direct radio visibility of each other, want to send a message to a third node C that is in direct visibility of both A and B (see possible configuration in figure). Let us suppose that A is already

transmitting a packet to C, and B tries to send a packet to C as well. B will first sense the medium to detect if a transmission is already ongoing. B cannot sense the signal from A, since they are not in direct visibility of each other. B will wrongly assume that the medium is idle (A is "hidden" from B) and will start the transmission of its packet; the B packet will be received on top of the A packet at node C, and both packets will be lost to C.

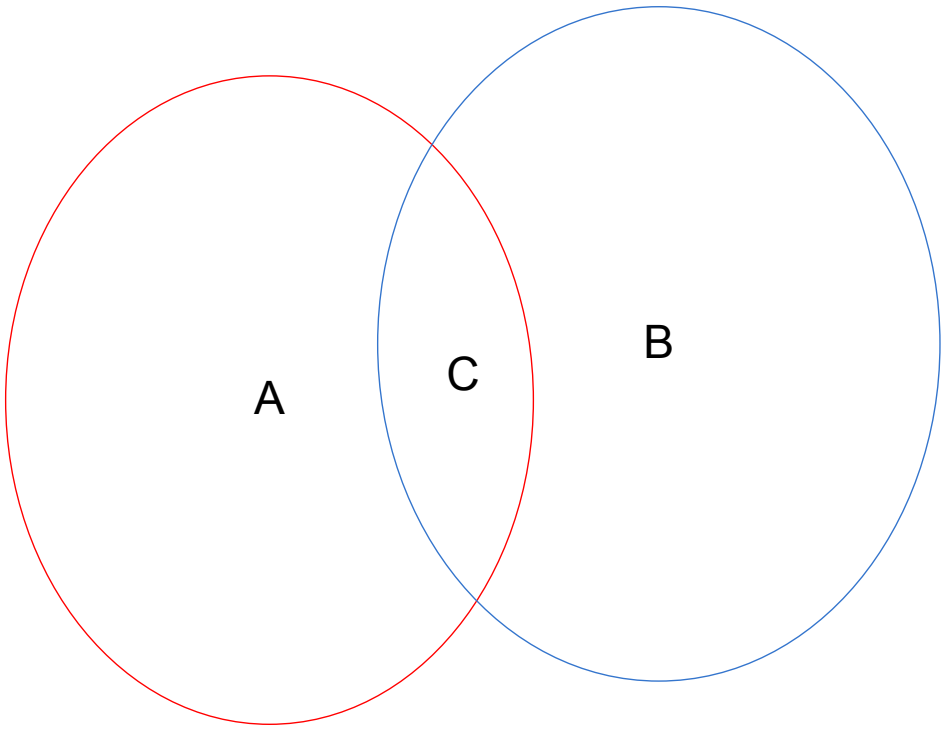


Figure 39 - Hidden Node Problem Configuration

This mechanism is the principal reason of packet collisions and ultimate loss. The random waiting and back-off techniques will work only if both nodes competing for access to the medium are in direct visibility of each other. This situation occurs multiple times in a dense WSN. A possible mitigation is the repetition of the same packet several

times with random waiting time. To be effective, the random waiting time range has to be larger than the duration of the packet. Another protection is an acknowledgment mechanism where B repeats its packet until it receives a reception acknowledgment from C.

5.5.2 Configuration protocol

Very few details will be given on this protocol. It will be used very rarely, essentially at startup when the network needs to be configured, or very rarely when the periodicity of the position computation needs to be changed. This type of message has not real impact on the power consumption, and does not need to be simulated. It is a flooding type. When a node has received this message and the synchronization message, it can start the wakeup/sleep sequence to minimize the power consumption. Before these two messages, at startup all nodes need to stay powered on continuously until they receive these first messages, and can synchronize onto it. The challenges posed by new nodes that try to insert themselves in an already synchronized network will not be examined.

5.5.3 Level and Connectivity Discovery Protocol

At startup

Each node needs to know how many hops from the root node a packet has been transmitted across; it also needs to know how many nodes with lower index and how many nodes with higher index it is directly connected to. This allows for a fine tuning of the energy conservation strategy over the lifetime of the WSN.

The selected algorithm is heavily inspired from the TPSN discovery phase and can be easily implemented as a "piggyback" of the FTSP protocol. The advantage is that the WSN can be "rediscovered" at every synchronization time. This is a definite advantage, as the connectivity varies over time, some nodes can disappear, and some connections can be intermittent.

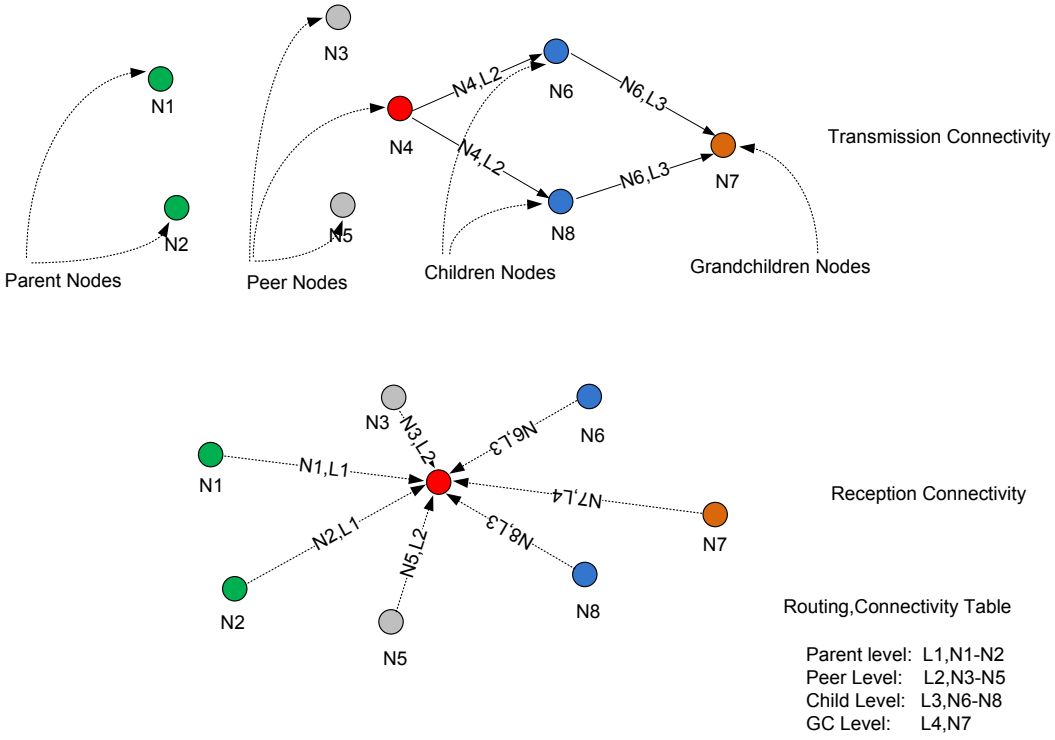


Figure 40 - Level and Connectivity Discovery Protocol

As shown in Figure 40 the first FTSP synchronization packet sent by the root node or node zero carries nodeID and nodeLEVEL(=0), along with the transmission local time tag. Each node receiving this packet will assign itself a nodeLEVEL of one unit higher than the nodeLEVEL received in the first packet. It also keeps track of how many packets

it received with nodeLEVEL less than its own nodeLEVEL (its Parents). After a random back-off time to avoid collision at the retransmission, each receiving node transmits a new FTSP packet carrying the nodeID of its Parent list, its own nodeID, its own nodeLEVEL, timetagged with their own clock. It continues to listen to other FTSP messages, and keeps track of the number of packets with nodeLEVEL lower, same, and larger than their own. This flooding occurs until all the reachable nodes have been hit, and the flow of FTSP messages dies down. Each node will have an internal local routing table similar to that shown in Table 6.

Table 6 - Local Routing Table Example

Parent nodeLEVEL	Parent ID List
Own nodeLEVEL	Peer ID List
Child nodeLEVEL	Children ID List
Grandchild nodeLEVEL	Grandchildren ID List
.....	....

There is an apparent oddity regarding the existence of Grandchildren. The current level node receives Grandchildren packets, but no Grandchild can receive any packet from it; if it were not true, the Grandchild would have intercepted the FTSP message from the current level, and would be in the Children category. The connectivity of a given node is the sum of the number of nodeIDs at all levels. It should be noted that some links can be missed in the accounting, as the packet sent by a node can collide with another packet, and stay undetected. It should be also noted that the subset of nodes constituting a level is not necessarily contiguous in general. This means that there is not always a path linking any node of a given level to any other node of the same level, stepping only through

nodes of the same level. This is not an impediment; both subsets will live independently, unaware of the existence of the other subsets of same level. This situation will be encountered mainly at the periphery of the network, at the levels the most distant from the gateway.

Periodic Update

The same mechanism is piggybacked onto the assistance and measurement protocols to keep track of the disappearing nodes, and occasionally of the appearing ones. Because the flood protocols have the ID of the currently transmitting node, and its level, listening to the "assistance messages" allows to keep track of the parents, peers and children still alive. In the same way, the listening of the "measurement messages" can help to update the routing table. The elimination of non transmitting nodes of the routing table should occur only after a number of missed transmissions over several position cycles. A single node can be missed from time to time, either through fading effects or message contention, and the corresponding node should be eliminated after reasonable confirmation.

5.5.4 Time Synchronization Protocol

The FTSP protocol already described in the previous chapter will be combined with the connectivity discovery protocol as already described in Section 5.5.3. It will be used periodically only for the centralized protocol, and only at the initial time for all decentralized implementations. The resynchronization is a byproduct of the position computation for the decentralized versions, and does not need to be explicitly triggered.

5.5.5 Assistance Protocol

The assistance protocol conveys various types of information whether it is a centralized or decentralized protocol. This is always a flooding protocol.

5.5.6 Clustering Protocol

At the end of the data collection, each node sends a "cluster head challenge message", composed of the node ID, its level, and a consumed energy level. The node that transmitted the lowest consumed energy level will be the cluster head for this sequence. In order to avoid the subtle effects of asymmetry in the connectivity, the winning node transmits a "cluster head declaration message" announcing it will be the cluster head, with the list of the nodes in the cluster it has won over. If this message is not received by one of the nodes that are to be part of the cluster, this node will itself declare the cluster head and will send its own "cluster head declaration message". If the first cluster head receives this message, it will reduce its list of nodes in the cluster to the nodes that are not in the second "cluster head declaration message"; this case is exemplified in Table 7, where two clusters are finally defined. A last comment is in order in this section: the cluster head is responsible to compute the relative positions of all nodes in the cluster, but no information was provided on how the cluster head partitions this list in closed triangles for relative positioning ambiguity.

Table 7 - Cluster Head Election Case Example

Node ID	Cluster head Challenge Message with Energy Level	Challenge Messages Received from	Cluster definition at this node	Cluster head declaration message (first is cluster head)	Corrected Cluster Definitions (first is cluster head)
1	10	2,3,4	1-2,3,4	1-2,3,4	1-
2	55	1,3,4	1-2,3,4	None	4-2,3
3	67	1,2,4	1-2,3,4	None	4-2,3
4	22	2,3	4-2,3	4-2,3	4-2,3

The decision to use the already consumed energy level as the cluster head decision criterion over a random assignment or other proposed criterion improves the lifetime of the network. The WSN will achieve the best energy consumption equalization within a cluster over multiple positioning cycles. This equalization will not occur across clusters, though.

5.5.7 Measurement Protocol

Centralized architecture

This is an aggregated message. Each node randomly chooses a parent in its parent list, transmits the measurement message, with its own ID, the target parent ID and its own carrier phase measurements. If no "acknowledgment message" is received from the target parent after a timeout period, it repeats the message with another target parent ID.

When a node receives a "measurement message" from a child, it sends an "acknowledgment message" with its own ID, and the list of children it acknowledges, and

takes the responsibility to route the measurements for them. It then aggregates all measurements it has received, including its own measurements, and keeping the sequence of parent IDs the measurements have followed from the originator, chooses a random target parent in its own parent list, and sends an aggregated "measurement message" similar to the elementary "measurement message" described earlier. The aggregated measurement messages sent by a given node can be numerous, depending on the order these measurements were received.

Decentralized Non Clustered Architecture

In the Decentralized Non Clustered Architecture, each node sent its own measurements in a short broadcast one hop message. The objective is for each node to get at least measurements from two other nodes besides their own measurements to allow a triangular closure at each node.

Decentralized Clustered Architecture

In the Decentralized Clustered Architecture, the node sends its measurements to the known cluster head, and waits for an "acknowledgment message". If no "acknowledgment message" is received after a timeout , the message is repeated until acknowledgement or full timeout.

5.5.8 Position Reporting Protocol

Centralized Architecture

The list of positions relative to the gateway for all nodes is sent in several aggregated position reporting messages. The message disaggregation structures are copied from the aggregated measurement messages, and sent individually to level one nodes. At each

level one node, the messages are disaggregated according to the level two nodes, and sent individually to each level two. This cycle continues to the terminal leaves of the network.

Decentralized non Clustered Architecture

No "position reporting message" is sent, as each node has its own position computing responsibility.

Decentralized Clustered Architecture

The cluster head sends back to each node of the cluster its relative position, and wait for an "acknowledgment message". The message is repeated if the "acknowledgment message" is not received within the timeout period.

5.5.9 Event reporting protocol

This is a very rare event that will have a very low impact on the operational lifetime of the WSN. No structure will be defined for it. Suffice to say that it is a directed message than needs to be conveyed to the gateway, along the routing tables (child-parent at each node), with acknowledgment, as the loss of any of these messages is not acceptable.

5.6 Chapter summary

To clarify the decisions made for each architecture, the details of the data flows are summarized in Table 8, Table 9 and Table 10.

Table 8 - Centralized Architecture - Data Flows I

Data Flow	Type	Periodicity	Conveyed Information
Configuration	Flooding	Once at start-up or at reconfiguration	Periodicity, and time of the next position computation
Discovery	Short Broadcast	At start-up only then piggybacking Assistance flow and Measurements at each position computation	Number of hops from gateway
Time Synchronization	Short broadcast	At each position computation	Gateway GPS time
Assistance	Flooding	At each position computation	Visible satellites, code offset, and Doppler
Computing Entity Assignment	N/A	N/A	N/A
Measurements	Aggregation along child parent routing	At each position computation	L1 Carrier Phase measurements for each visible satellite
Position Reporting	Disaggregation along measurement routes	At each position computation	Relative Position to gateway
Event Reporting	Flood	Very infrequent	Position, and nature of the event, or ID and nature of the event.

Table 9 - Decentralized Architecture - Data Flows II

Data Flow	Type	Periodicity	Conveyed Information
Configuration	Flooding	Once at start-up or at reconfiguration	Periodicity, and time of the next position computation
Discovery	Short Broadcast	At start-up only then piggybacking Assistance flow and Measurements at each position computation	Number of hops from gateway
Time Synchronization	Short broadcast	Only at start-up	Gateway GPS time
Assistance	Flooding	At each position computation	Visible satellites, code offset, and Doppler and elements of design matrix for local position computation
Computing Entity Assignment	N/A	N/A	N/A
Measurements	Short broadcast only one hop	At each position computation	L1 Carrier Phase measurements for each visible satellite
Position Reporting	Short Broadcast only one hop	At each position computation	Relative Position to gateway
Event Reporting	Flood	Very infrequent	Position, and nature of the event, or ID and nature of the event.

Table 10 - Decentralized Clustered Architecture - Data Flows III

Data Flow	Type	Periodicity	Conveyed Information
Configuration	Flooding	Once at start-up or at reconfiguration	Periodicity, and time of the next position computation
Discovery	Short Broadcast	At start-up only then piggybacking Assistance flow and Measurements at each position computation	Number of hops from gateway
Time Synchronization	Short broadcast	Only at start-up	Gateway GPS time
Assistance	Flooding	At each position computation	Visible satellites, code offset, and Doppler and elements of design matrix for local position computation
Computing Entity Assignment	Short Broadcast one hop	At each position computation	N/A
Measurements	Short broadcast only one hop	At each position computation	L1 Carrier Phase measurements for each visible satellite
Position Reporting	Short Broadcast only one hop	At each position computation	Relative Position to gateway
Event Reporting	Flood	Very infrequent	Position, and nature of the event, or ID and nature of the event.

CHAPTER 6

DATA SIMULATION

6.1 Generalities

After introducing the ARTI technology, the cycle ambiguity constraints, and the general architecture, this chapter attempts to put all pieces together and demonstrate the system performance. Short of building a real system, a simulation approach has been taken. This in particular will enable the quick comparison of the several proposed options, compared to the centralized solution.

As the feasibility and accuracy of ARTI have been already demonstrated in preceding chapters, only the power consumption and lifetime aspects, which are considered of prime importance in the overall design will be emphasized.

Each node will be simulated as an entity endowed with a limited amount of energy at setup, and the energy consumption will be monitored for each atomic operation, as a combination of operation time and power consumed during this operation. The power consuming events are the transmission of a message, the reception of a message, the collection of measurements, and the computation of the position solution. The dominant energy consumption lies in the transmitted power during transmission, as a combination of processor and receiver consumption during reception, and the number of cycles necessary to crunch the numbers during solution computations. The simulation

parameters will be extracted from the MICA II specifications. Since this is still a basic model, the ratio of energy between transmission and reception and the power ratio between transmission and solution computation will be introduced as a refinement.

6.2 PROWLER simulator introduction

There are many versions of WSN that have been written to cover a very large spectrum of needs (Sridharan et al 2004, Curren 2005). Some of them are focused on the correctness of the wireless protocols, and put a special emphasis on propagation, the probability of missing a message, and on "locking" the protocol. Others emphasize the power consumption, where any power activity is carefully monitored and logged. Some also are an "instruction exact" simulation, where the application running on the node is written in the target language and the simulator emulates the processor instruction set. This last category must be written in an efficient compiled language, as it simulates long periods of activity, and simulates the activity of all the processors in the network, which can be in the thousands.

Although assumptions about the hardware architecture were made (MICA II), an operating system (TinyOS) and a language (nesC), in which the operating system and the application are written, the "instruction exact" simulation would have necessitated a complete physical and link layer definition, with complete definition of each message, in addition to their error correcting codes; this would have been way beyond the scope of

this dissertation. It was decided to use the Probabilistic Wireless Network Simulator "PROWLER", introduced by Simon et al (2003), that is a middle-of-the-road simulator, written in MATLAB, with a fairly complete set of simulation capabilities for the transport, link and MAC (Media Access Control). Power consumption monitoring capabilities were added as a figure of merit of the relative quality of the options.

6.2.1 PROWLER High level description

For details, refer to Simon et al (2003). However, some important details that are not mentioned in the paper will be underlined here.

Prowler is a discrete event driven simulator, with an update clock granularity of 25 μs, which corresponds to the 40 kbps of the radio bit rate.

The simulator is composed of a finite state machine engine, MAC layer, radio propagation model, and application layer. Prowler can accommodate any sophisticated radio propagation model, but the following default model described in Simon et al (2003) was chosen:

$$P_{rec}(i,j) = P_{tr}\frac{1}{1+d_{i,j}^{\gamma}}\left[1+\alpha\left(d_{i,j}\right)\right][1+\beta(t)] \qquad (6.4)$$

The random variables $\alpha\left(d_{i,j}\right)$ representing geometry related fading effects and $\beta(t)$ representing time related fast fading effects are Gaussian random variables with the following characteristics:

$$\alpha(d_{i,j}) = N(0,0.45),$$
$$\beta(t) = N(0,0.02). \quad (6.5)$$

The decay parameter $\gamma = 2$, corresponds to a free space hypothesis.

The reception threshold is taken at 0.1. The transmit power at 0 m is chosen at "2890" arbitrary units. The RF model is a radio at 916.5 Mhz, with a transmitted power of +5 mW, with isotropically transmitting and receiving antennas. The advertised range is 170 m in open space, which corresponds to a minimum reception level of about -55dBm. The arbitrary transmitted power number is computed by back calculating the power necessary at 0 m to get an arbitrary reception level of "0.1" at 170 m.

The MAC layer is a CSMA/CA scheme. The default parameters that have been used throughout the simulation are listed in Table 11:

Table 11 - Prowler: Default Medium Access Control Default Parameters

Simulation Clock period (SCP)	25μs
Minimum Waiting time	200 SCP=5ms
Uniform random waiting time standard deviation	128 SCP=3.2ms
Minimum Backoff Time	100 SCP=2.5ms
Uniform random Backoff Time Standard Deviation	30 SCP=0.75ms
Packet Duration	960 bits=960 SCP=24ms

For reference, the MAC model borrowed from the paper is repeated in Figure 41.

When the application layer send a packet ("Send_Packet" event), the command needs to wait for the sum of minimum waiting time plus the random waiting time to simulate the time elapsed before the command reaches the lower Hardware layers of the RF chip. The chip then checks the RF channel availability ("Channel_Idle_Check"), and starts immediately the transmission if no other node is already transmitting. If there is already an on-going transmission, the RF chip backoffs for "minimum backoff time", plus the "random backoff time", before rechecking the medium for Idle mode. The transmission time is the duration of the packet, and the RF driver layer sends back an "Packet_Sent" event to the application layer.

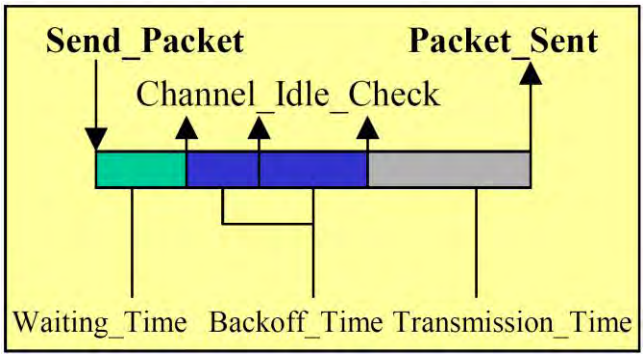


Figure 41 - Prowler MAC layer communication scheme

To conclude, Prowler accepts only two-dimensional node locations and simulates only messages of the same size, but not of the same content. These two limitations were not considered serious enough to justify the rewriting of the simulator engine.

6.2.2 Prowler Simulation Environment

This section describes the simulation environment available to the user to represent an application. There are three important MATLAB functions that must be provided by the

user, the topology of the network, the radio model and the application code. The topology consists of the list of the nodes Identifiers (ID), and their associated 2D coordinates in a cartesian coordinate system. The radio model determines the signal strength received at a node given the transmitted power, the inter node distance extracted from the topology, the propagation decay parameter, and the sensitivity threshold of the receiver. Any packet above the threshold is declared received and delivered to the receiving node. The application function is by far the most complex part. There is a limited list of time-tagged events that is generated by the Prowler engine, either internally, as "Init_Application" or "Application_Finished" delivered when the prescribed duration of the simulation is elapsed, or by reaction of Prowler to prior nodes actions, such as "Clock_Tick", "Packet_received", "Packet_Sent", "Collided_Packet_Received". There are very limited nodes actions allowed by Prowler, essentially "Send_Packet" for message transmission and "Set_Clock" for triggering a "Clock_Tick" event in the future . All nodes run exactly the same code over and over; the only personalization is by the global variable "ID", that let the node (in fact the application code running on behalf of the node) know what is its current identity at run time, and a global structure "memory" where the application code keeps all node context information from call to call. The application code is essentially a large "switch" structure with one "case" instruction block by event, that triggers new actions in reaction to the events received from the Prowler engine. All the future events are kept in a master queue "event_Q" maintained by the Prowler engine. An event stored in the master queue is essentially a time tag declaring when the event should be served, a target ID identifying which node should be notified and a data cell structure that will be delivered to the node for action. The data cell structure has different interpretations

depending on the type of event and is at the discretion of the user. The messages passed from transmitting node to Prowler and back to receiving node are a user configurable data structure. The message is actually defined as a data structure with one member per message information element. Some information elements are always present such as transmitter ID and message type, that are mandatory for the correct interpretation of the rest of the message. All the message specific information elements are extra structure members to be defined and interpreted according to the message type. This architecture is quite convenient and suitable for Monte-Carlo simulations, with complex interactions between messages. Conceptually, it should have been possible to call the GPS data simulator and the positioning algorithms from the Prowler application level, but it is not a feasible idea. Prowler simulates a large number of nodes with their own processing capability on a single processor limited by its own hardware. The processing power of all nodes to simulate is far surpassing the single processor processing power. A direct simulation approach would be hundred times slower than the real time. The sensitive approach used here is to extract a behavioral model from the positioning simulations (such as minimum data collecting time for 95 % of successful ambiguity fixing), and to implement it in Prowler. Only the message interactions are simulated, but with a level of realism good enough for the lifetime prediction. The last piece that is not built in Prowler is the energy monitoring. Every time a task is accomplished, such as running the processor for a given phase duration, or transmitting a message, a careful accounting of the energy consumption takes place in parallel, along with the number of bits transmitted, the number of messages sent and received, and so on. The practical experience demonstrates the usefulness of the approach, but with the well known problems of

parallel processing debugging, with no guaranteed sequence in message delivery. A large amount of time was dedicated to properly debug the proposed protocols. A seemingly simple protocol design was always showing serious flaws that could not be foreseen during the design phase. Invariably, limitations or deadlocks or starving conditions that were off the expected behavior were experienced. To the knowledge of the author, there is no formal protocol verification tool available for WSN, similar to the PETRI nets used years ago for formal verification of lossy server-client radio protocols. Even with a simulation limited to the message exchanges, with 100 simulated nodes, and a duty cycle of about 7 %, the simulation-to-true time ratio was about 1:1. That means that for simulating 10 minutes or a single positioning cycle of simulation time, about 10 minutes of run time are required.

6.2.3 Connectivity Simulation

To demonstrate the power of Prowler, the discovery algorithm as described in Section 5.5.3 will be simulated first. The network is build with 100 nodes uniformly spread on a square of 200 m by 200 m. The transmitted power is adjusted at "200" for a nominal range of 45 m, or about one fourth of the side length of the simulated network. The gateway position has been forced in the middle of Figure 42. The reason is to reduce the impact of the marginal effects due to finite network dimensions at least for the first routing levels. The last levels will be at the periphery of the network, and their connectivity index will suffer from a lack of immediate neighbours to connect to. Their number will also decrease. The figure graphically shows the levels by using colours. Starting from the gateway or node 1, that is assigned the level 0, all nodes of the first

level are connected by a network of green lines, representing the direct radio connections found between all members of the level 1. The second level color is blue, and is roughly located at the circumference of the level 1, with a larger number of nodes. The yellow level is 3. The level 3 is not very uniform but is very dense on the bottom left and rather sparse on the top left. The next level 4 is in cyan color. It is important to note that this next level is split in three disconnected subsets that are not aware of each other. This is one of the effects of the limited dimensions of the network, where the last level can be composed of only the remaining nodes that are at the periphery. The number of nodes is too small to guarantee an inter node average distance equal to the transmission range, and to stretch to the whole outer perimeter of the network.

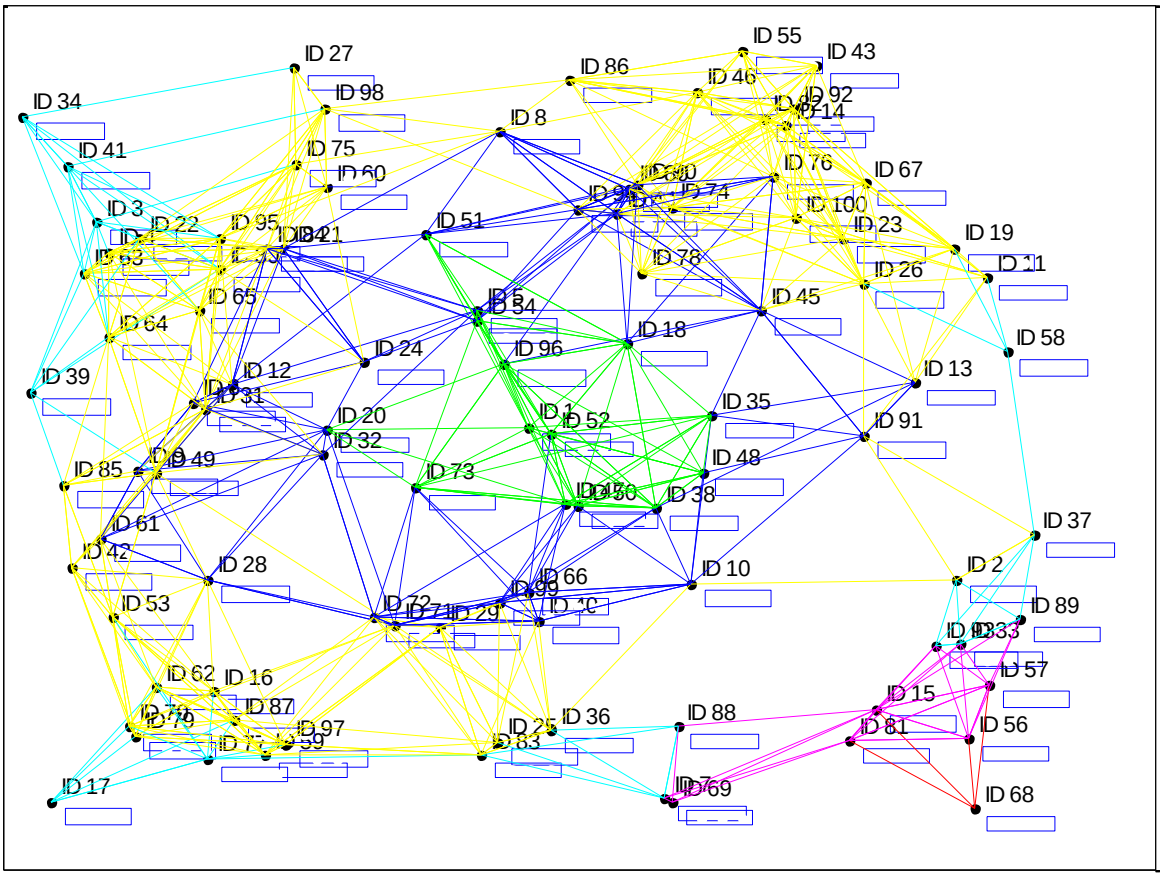


Figure 42 - Prowler - Discovery Protocol Simulation - WSN Network layout

Statistics on the number of nodes at each level have been summarized in Table 12. All nodes are visible from at least one node, as there is no node without an assigned level. The number of levels increases steadily up to level 3, where the limited total number of nodes forces the last level to be reduced in size.

Table 12 - Number of Nodes per Level

Level Number	No level assigned	0	1	2	3	4	5	6	All levels Combined
Number of Nodes	0	1	13	24	44	12	5	1	100

The simulation was updating the connectivity tables per node as described in Section 5.5.3. The statistics of these tables for all levels combined are summarized in Table 13. As expected, the maximum connectivity is at the peer level. Parent and child connectivities are similar as both are driven by the ratio of node density vs. node range. The grandchild connectivity, that is a measurement of the connectivity asymmetry where a node hears its grandchildren but the inverse is not true, is only imposed by the random fading effects and the loss of message probability that is artificially introduced by Prowler, is very small.

Table 13 - Inter and Intra Level Connectivity Statistics

	Parent	**Peer**	**Child**	**Grandchild**
Average connectivity	2.8	4.84	2.96	0.42
Standard Deviation	1.65	2.74	3.05	0.91

This simulation will be concluded with a histogram of peer connectivity cumulating all levels shown in Figure 43. The connectivity varies widely, namely between 0 and 11. Aside from the gateway that has zero peer connectivity by design, about 7 other nodes are

in the same situation. They are still connected to more than one parent node , but they are a single leaf at the end of the tree. This is another effect of the limited size of the network.

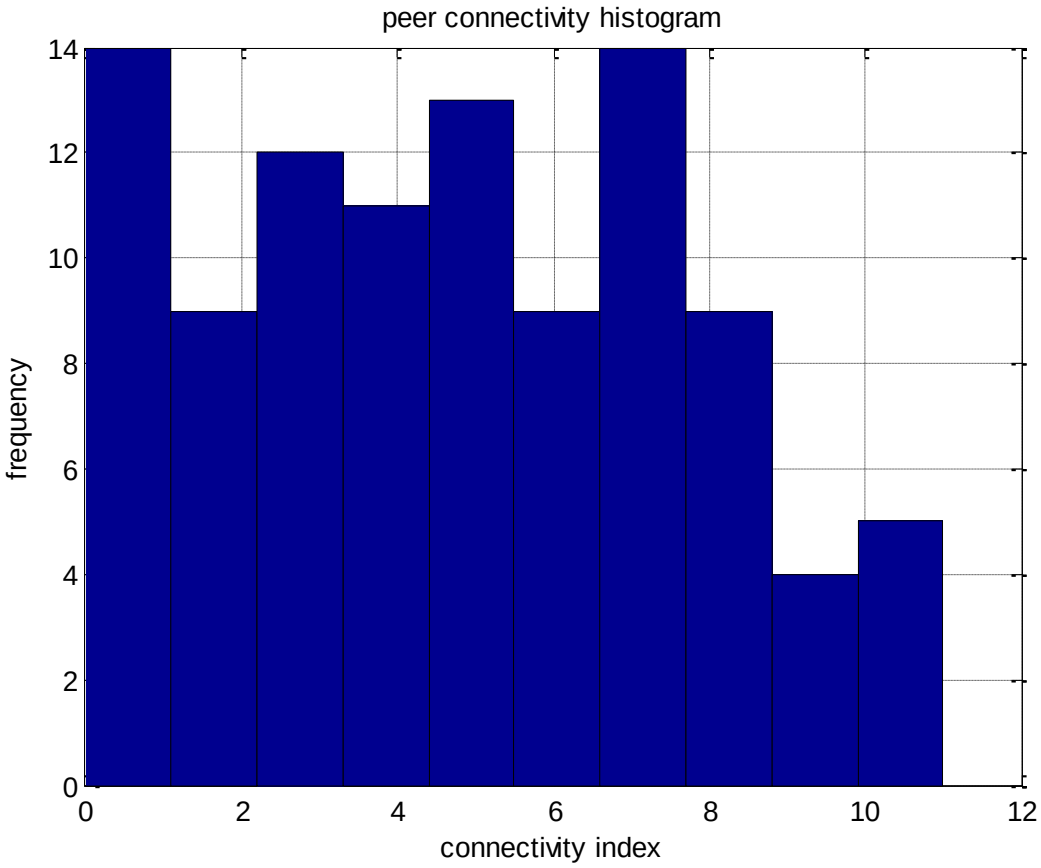


Figure 43 - Prowler - Peer Connectivity Histogram

Power models

In order to compare several options of message and processing architectures, a practical model for power consumption has to be defined. In order to keep this simulation as

realistic as possible, a specific type of node architecture was chosen, the MICA II; its power specifications were used as simulation input parameters. This section details all the assumptions made in the course of the power simulations.

RF power models

The specifications do not clearly distinguish between RF chip and the rest of the node consumption. This is not a limitation, as the power modes can be programmed as four combined states. The power consumption levels are given in Table 14 and Table 15.

Table 14 - MICA II Hardware States and Consumption Levels

RF state	Processor State	Description	Current consumption
Transmit High Power +5dBm @ 916.5 Mhz at antenna (See Table 15 for other power levels)	Active	Active transmit high power	27 mA @ 3 V
Receiving	Active	Active reception	10 mA @ 3 V
Turned OFF	Active computation	Position computation mode	8 mA @ 3 V
Turned OFF	Sleep mode		15 µA @ 3 V

Table 15 - MICA RF Transceiver chip consumption at various transmitted power levels

Output RF power @915MHz	CC1100 CHIPCON current consumption @3V	Total MICA II power consumption @3V	Energy per transmitted bit at 40kbps
+5dBm	19.9 mA	27.9 mA	2.1 µJ/bit
0dBm	16.7 mA	24.7 mA	1.85 µJ/bit
-5dBm	13.9 mA	21.9 mA	1.64 µJ/bit
-10dBm	14.3 mA	22.3 mA(1)	1.67 µJ/bit
-20dBm	12.3 mA	20.3 mA	1.522 µJ/bit
-30dBm	11.8 mA	19.8 mA	1.485 µJ/bit

(1): anomaly in the data sheet of the CC1100-Chipcon

In a similar manner, the energy per bit consumed by the MICA II in reception mode is estimated at 0.7µJ/bit.

Processor power models

The MICA II has a system clock of 16 MHz, and processes a single instruction per clock cycle. The power consumption is 8mA at full processing.

Position location Power Models

In order to estimate the total energy consumption to compute a single baseline relative position at a given node, and using three sets of measurement data, a systematic computation of the necessary number of float operations is needed. The size of the vectors and matrices to process and invert will be estimated first, and then will be converted into the number of individual additions, multiplications and divisions. Each of these individual operations will be then converted into an estimate of the total number of

cycles, first by estimating how many single instructions a single float operation takes, and then estimating the overhead. The approximations of the model are multiple, but it is not an impediment to the comparison between options, provided the assumptions are kept consistent throughout the simulations. The other solution that would have been more precise would have been to write the code on the node platform and to run this code, including the operating system in an instruction and cycle accurate simulator, specific to MICA II. This was clearly outside the scope of this dissertation.

The translation of single precision real numbers standard operations into the number of clock cycles and energy per operation are found in Table 16. All numbers have been extracted from Elmenreich et al (2007), which provides the number of clock cycles used for GCC implementation of double library. The double are 32 bits in the AVR GCC library and are therefore single precision.

Table 16 - AVR float operations conversion to number of clock cycles

Double Operation	Number of system clock periods @ 16Mhz	Duration in µs	Energy per operation
Addition/Subtraction	80	5 µs	~120 nJ
Multiplication	2851	178.19 µs	~4.28 µJ
Division	1385	86.56 µs	~2.1 µJ

Matrix operation count

Matrix inversion

The resolution of the system of linear equations represents a very large part of the operation count in the relative position computation process. The classical manual resolution technique is Cramer's rule. It is not efficient for computer implementation where one of the best practical algorithms is Gaussian Elimination. If a explicit matrix inversion is expected, rather than the solution of equations, the best practical algorithm is the Gauss-Jordan Elimination. Table 17 precisely quantifies the processing cost in term of the number of elementary operations for several well used resolution techniques. The rows three and four refer to an hybrid method, where the matrix inverse is first computed using either Gaussian Elimination or Gauss-Jordan Elimination, then the set of input parameters is multiplied by the inverse matrix. It can be seen that these hybrid methods are less effective than the direct resolution if an explicit form of the inverse matrix is not needed. Table 17 is extracted from Tapia et al (2001).

Table 17 - System of Linear Equations Resolution - Operation Count

Method	Multiplications	Additions
Gaussian Elimination	$\frac{1}{3}n^3 + n^2 - \frac{1}{3}n$	$\frac{1}{3}n^3 + \frac{1}{2}n^2 - \frac{5}{6}n$
Gauss-Jordan Elimination	$\frac{1}{2}n^3 + n^2 - \frac{5}{2}n + 2$	$\frac{1}{2}n^3 - \frac{3}{2}n + 1$
Solving using inverse obtained by Gaussian Elimination	$\frac{5}{6}n^3 + 2n^2 - \frac{5}{6}n$	$\frac{4}{3}n^3 - \frac{1}{2}n^2 - \frac{5}{6}n$
Solving using inverse obtained by Gauss-Jordan Elimination	$\frac{3}{2}n^3 + \frac{1}{2}n - 1$	$\frac{3}{2}n^3 - 2n^2 - \frac{1}{2}n$
Cramer's Rule	$(n+1)!$	$(n+1)!$

For reference, the formulas in the table have been plotted in Figure 44, Figure 45 and Figure 38. Cramer's rule does not appear on these plots, due to the orders of magnitude larger.

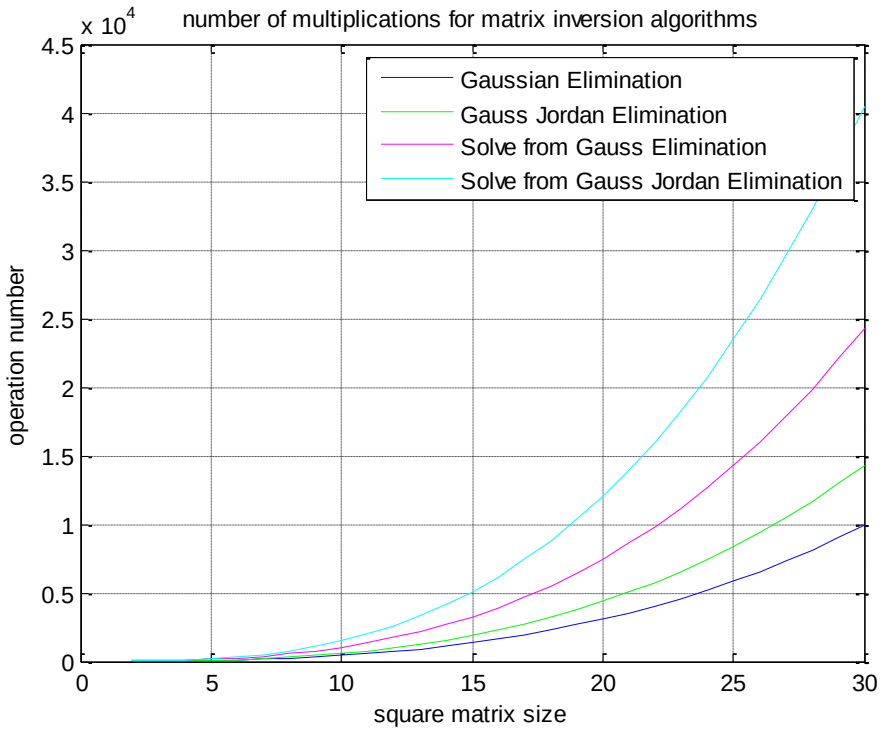


Figure 44 - Matrix Inversion Algorithms - Multiplication Count

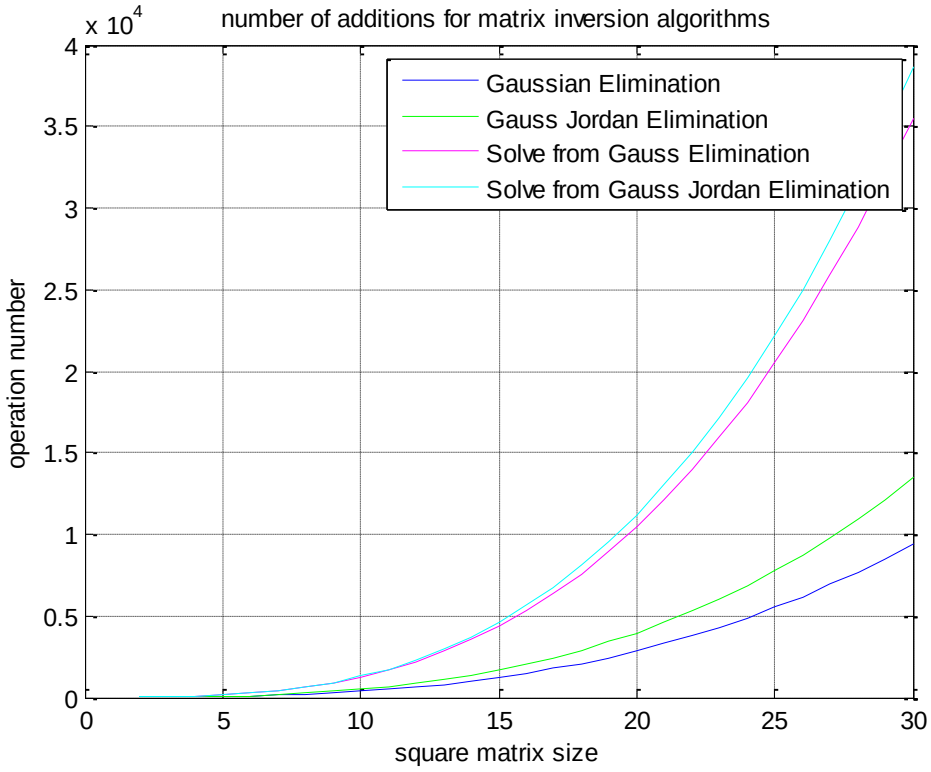


Figure 45 - Matrix Inversion Algorithms - Addition Count

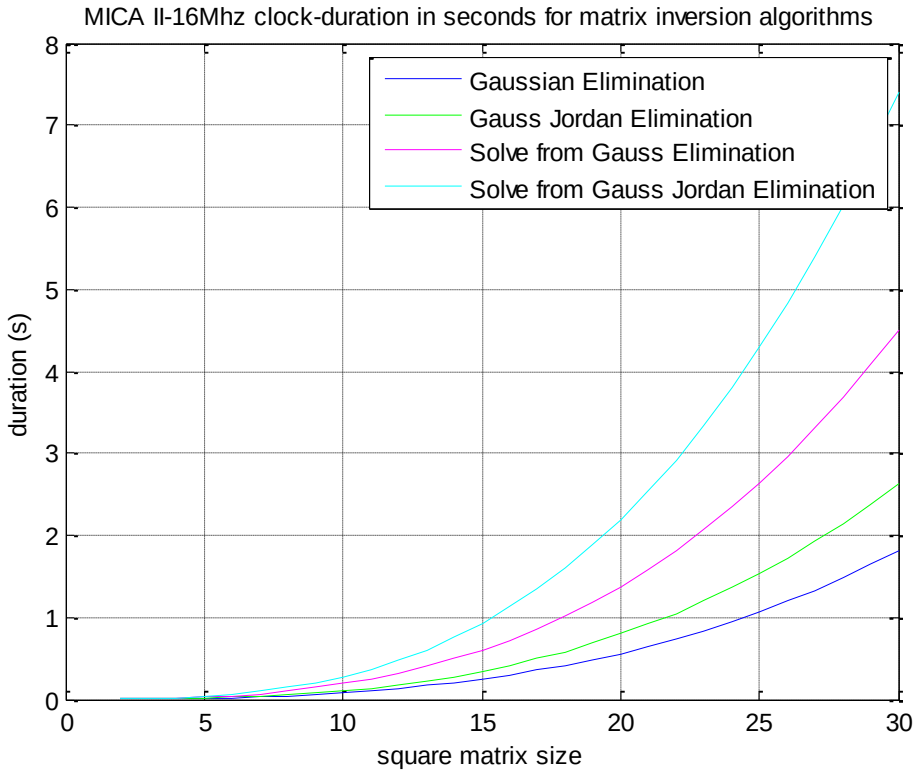


Figure 46 - Matrix Inversion algorithms - Execution Time on MICA II

Matrix multiplication

Aside from the traditional inner-product and faster algorithms with asymptotically fast complexity such as Strassen-Winograd, the most practical and efficient, have been proposed.

A (lxm) by (mxn) matrix inner product operation count, or more appropriately its time cost, can be expressed as a function of complexity $\boldsymbol{\mathcal{O}}(n^3)$ as (Hyodo et al 2005):

$$T_{inn}(l,m,n) = (lmn)t_* + \ln(m-1)t_+ \qquad (6.6)$$

where

$T_{inn}(l,m,n)$ total inner product execution time for a (lxm) by (mxn) matrix product,

l first matrix row count,

m First matrix column count and second matrix row count,

n Second matrix row count,

t_* Elementary Multiplication execution time,

t_+ Elementary Addition execution time.

In the same conditions, the Strassen-Winograd Algorithm, for a multiplication of square matrices $n = m2^k$, has a time cost of (Hyodo et al 2005):

$$T_{SW}(m,k) = 7^k(m^3 t_* + m^2(m-1)t_+) + 5m^2(7^k - 4^k)t_+ \qquad (6.7)$$

where the order is $\mathcal{O}(n^{2.708})$.

The main weakness of the Strassen-Winograd approach is its recursivity, which translates in multiple stack operations, and intermediate memory size that more than offset the operation count benefits (Kakaradov 2004). The crossover point in matrix size where the SW algorithm supersedes the inner-product in operation count is quite high and would not be reached in normal conditions. Kakaradov (2004) estimates this crossover point at n=60,750 in practical situations. In view of all these limitations, the traditional inner-product algorithm will be assumed.

Matrix addition

The addition of two matrices $A(mxn), and\ B(mxn)$ requires mxn additions.

Sequence of Operations

Combining all the elements together, the cost of the computation of the relative position is estimated, first in number of elementary float operations (addition/ subtraction/ multiplication/ division), then in number of clock cycles for the considered processor, and finally in duration.

It is assumed that m satellites are measured on both sides of the single baseline and two receivers are resolved together, and that one receiver needs to have the integer time ambiguity resolved. At each measurement epoch, 2 m carrier phase measurements will be ready for processing. After double differencing, the size of the measurement vector will be reduced to (m-1)

Table 18 - Position Computation Processing Steps and Number of Operations

Processing Step	Description	Number of Multiplications	Number of Additions
Measurements and design matrix data gathering	Carrier phase measurements vector $\psi_{(2m,1)}$ Design matrix $A_{(m-1,m+3)}$		
Double differences Computation	Double difference carrier phase vector $\bar{l}_{(m-1,1)} = D_{(m-1,2m)}\psi_{(2m,1)}$	$2m(m-1)$	$(m-1)(2m-1)$
Cofactor matrix computation	$Q_{\Delta(m-1,m-1)} = D_{(m-1,2m)}D^{T}_{(2m,m-1)}$	$2m(m-1)^2$	$(2m-1)(m-1)^2$
weight matrix computation by Gauss Jordan Elimination	$\Sigma_{\Delta(m-1,m-1)} = Q_{\Delta(m-1,m-1)}{}^{-1}$	$\frac{1}{2}(m-1)^3 + (m-1)^2 - \frac{5}{2}(m-1) + 2$	$\frac{1}{2}(m-1)^3 - \frac{3}{2}(m-1) + 1$

Table 18 - Position Computation Processing Steps and Number of Operations (Continued)

Processing Step	Description	Number of Multiplications	Number of Additions
First Partial results Computation	$\left(A^T \Sigma^T \bar{l} \right)_{(m+3,1)} = A^T{}_{(m+3,m-1)} \Sigma^T{}_{(m-1,m-1)} \bar{l}_{(m-1,1)}$	$(m+4)(m-1)^2$	$(m+4)(m-1)(m-2)$
First Partial results Accumulation	$\left(\sum_{i=1}^{T} A_i^T \Sigma_i^T \bar{l}_i \right)_{(m+3,1)} = \sum_{i=1}^{T} \left(A_i^T \Sigma_i^T \bar{l}_i \right)_{(m+3,1)}$		$(m+3)$
Second Partial Results Computation	$(A^T \Sigma A)_{(m+3,m+3)} = A^T{}_{(m+3,m-1)} \Sigma_{(m-1,m-1)} A_{(m-1,m+3)}$	$2(m+3)(m-1)^2$	$2(m+3)(m-1)(m-2)$
Second Partial results Accumulation	$\left(\sum_{i=1}^{T} A_i^T \Sigma_i A_i \right)_{(m+3,m+3)} = \sum_{i=1}^{T} (A_i^T \Sigma_i A_i)_{(m+3,m+3)}$		$(m+3)^2$
Linear System Resolution by Gaussian Elimination	$\left(\sum_{i=1}^{T} A_i^T \Sigma_i A_i \right)_{(m+3,m+3)} \overline{x}_{(m+3,1)} = \left(\sum_{i=1}^{T} A_i^T \Sigma_i^T \bar{l}_i \right)_{(m+3,m+3)}$	$\frac{1}{3}(m+3)^3 + (m+3)^2 - \frac{1}{3}(m+3)$	$\frac{1}{3}(m+3)^3 + \frac{1}{2}(m+3)^2 - \frac{5}{6}(m+3)$

6.3 Simulation 1: Centralized architecture

This first simulation is a reference case where the only role of the nodes is to collect measurement data and to forward them to the gateway for computation; the positions are

centrally computed in the gateway and kept there for comparison with the future positions computed at the same location.

As the acquisition and data measurement effort is the same for all architectures, only the non common functions will be simulated for relative comparison. For this centralized case, they encompass the assistance information forward transmission and the measurement backward transmission. The assistance data flow that follows the direct connection lines is depicted in Figure 47. The measurement data flow follows the inverse routing shown in Figure 48. In this case, no energy is consumed in the network for the position computation (the gateway benefits of an "always on" power connection).

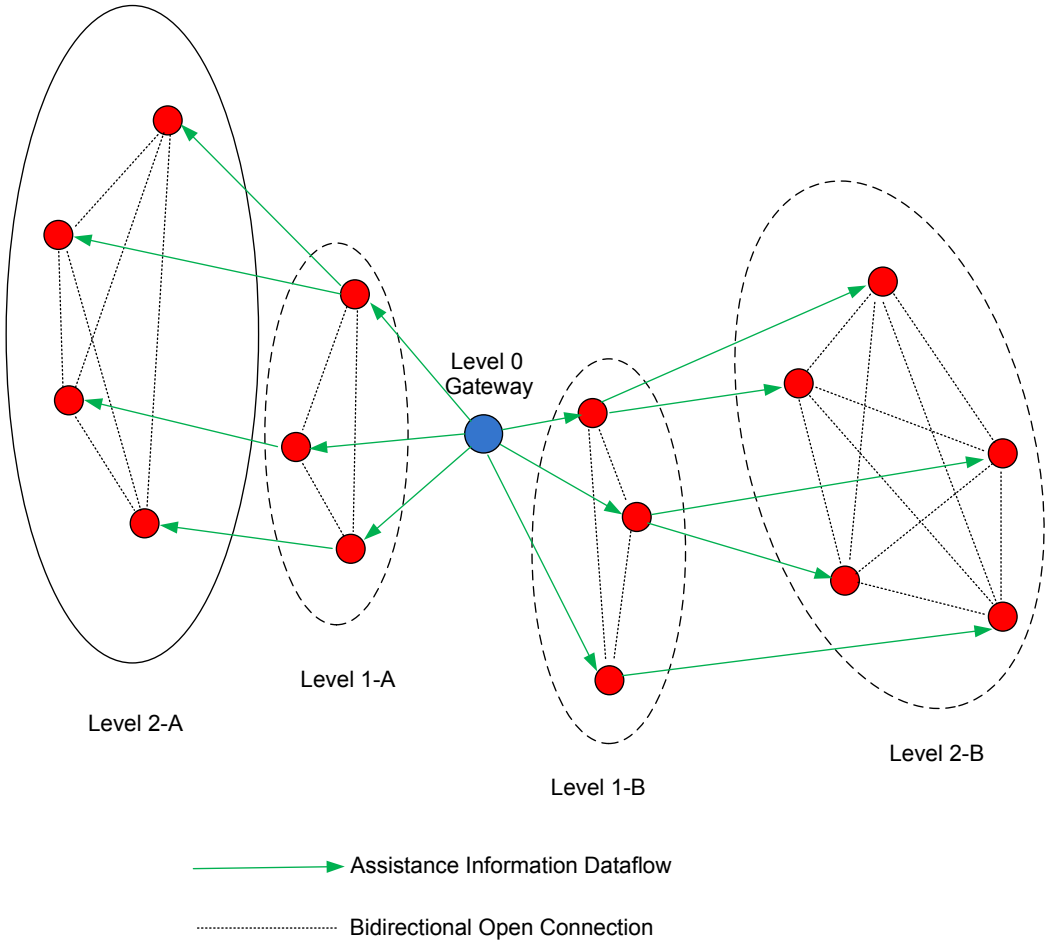


Figure 47 - WSN Centralized Architecture - Assistance Data Flow

The positions are computed at the gateway; the actually computed baselines have been already introduced in Figure 36. Is it worthwhile to note that assistance data flow is a "broadcast" data flow, where each message carries the same information, and each message transmission and reception cost is the same regardless of the location in the network. In contrast, the measurement data flow is of an "aggregation type" where, starting from the leaves of the connectivity diagram, each intermediate node aggregates its own measurements plus the measurements of all nodes it received from the other higher level nodes before forwarding the message to the next lower level.

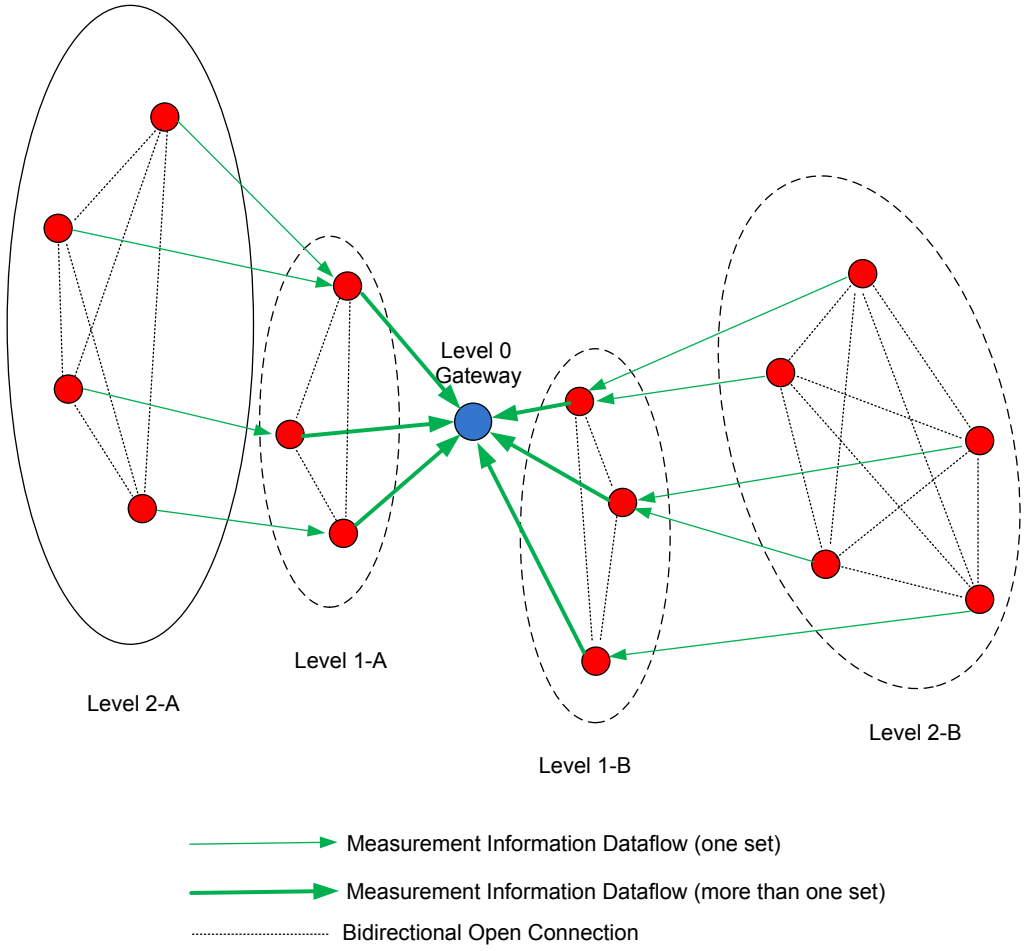


Figure 48 - WSN Centralized Architecture - Measurement Reporting Data Flow

Each level traversal will increase at least by one unit the size of the message, thus increasing the retransmission energy cost when closer to the gateway of level 0. The position redistribution is of disaggregation type.

6.3.1 Message Information Content and Size

There is still one piece of information that needs to be defined before the simulation can be attempted, namely the content and size of the messages. Assuming that m is the

number of satellites received at the gateway and that all parameters and measurements are expressed in float representation (32 bits or 4 bytes), one obtains the results shown in Table 19.

Table 19 - Centralized Architecture - Message Contents and Sizes

Message	Content	Size(in bytes)
Assistance	Acquisition Assistance: PRN number, code offset, Doppler offset (3parameters per satellite)	m x 3x 4 [(1)(2)(3)]
Measurements (one set)	Carrier Phase measurements	m x 4 [(1)(2)(3)]

(1) m, number of satellites is taken at 8 in the whole simulation
(2) all values (assistance data and measurements) are assumed to be in 4 bytes IEEE single float representation
(3) a byte is represented by 8 bits (no parity bit, or any redundancy)

All data are understood for a single node, or a single piece of information. The aggregation and disaggregation effects are not taken into account in Table 19.

As no positioning takes place locally at each node, except the gateway that benefits unlimited power, the position processing cost in terms of energy consumption is not considered in this first simulation.

None of the message sizes take into account any overhead information such as the message sender ID, message destination ID, and message type. The ID information is negligible compared to the size of the payload data. Error detection and correction overhead (such as parity bits, or CRC) are not included either as there are multiple ways to handle this problem with widely varying overhead percentages over the message

payload. As the same rule was applied for all simulations and the main interest is in the relative performance, this omission will have a negligible impact on the outcome.

6.3.2 Power Assumptions

For the simulations, the per-phase power consumption numbers of Section 6.2.3 are summarized in Table 20. The transmission activity is modeled as an extra consumption of 17 mA at 3 V over the reception consumption on top of the background consumption, or 1.275 micro Joules per transmitted bit, only during the transmission periods.

Table 20 - MICA II Simulated Power Consumptions per Phase

Phase	Processor	Wireless RF	GPS section	Current at 3V	Power Consumption
Assistance	8mA	2mA	off	10mA	30mW
GPS data collection	8mA	off	8mA	16mA	48mW
Report Measurement	8mA	2mA	off	10mA	30mW
Position Computation	8mA	Off	Off	8mA	24mW

6.3.3 Comments and results

All results reported below assume a total number of 100 nodes, uniformly spread on a square of 200 by 200 m, and a maximum radio range of 45 m, in line with the assumptions in Section 6.2.3. Five hierarchical levels are consistently obtained in all these simulations with a similar connectivity index.

The tuning of the timing parameters was attempted first to get consumption and performance results adjusted to allow a meaningful comparison with the other architectures. This operation has been found difficult to execute, first because of the large number of parameters to adjust, then because of their mutual interaction and finally because of the statistical nature of the results that need to be averaged over several runs to be significant. Eventually a sequential procedure has been sufficiently accurate to deliver a parameter tuned set believed to be within 10 % of the true optimum. The exhaustive list of timing parameters to adjust is as follows, in roughly lower to higher hierarchy:

The “random waiting time” and the “random back-off time” that are related to the sharing of transmission medium come first (for definition, see Figure 41 - Prowler MAC layer communication scheme). Then comes the timing of each phase within a complete positioning cycle. The cycle has three phases, first *assistance information* broadcast by flooding during the “assistance duration”, followed by carrier phase *data collection* during the “data collection duration”, and finally the *measurement reporting* back to the gateway during the “report measurement duration”. Within the *assistance phase*, the “assistance timeout duration” at the end of which a new “assistance request message” is sent if no “assistance message” has been received, and within the *report measurements* phase , the “report timeout duration” when the report messages are resent if no report acknowledgement message has been returned, have to be adjusted as well. The “positioning cycle periodicity” is at the higher level.

The adjustment procedure is as follows. First, the “positioning cycle periodicity” is imposed by the application constraints, arbitrarily imposed here at 10 minutes. The only constraint is a minimum period long enough to cover the durations of the three active

phases up to their full completion, plus a sleep period that can be reduced to zero if necessary. An overlap with the next positioning cycle period would be catastrophic, as the messages still undelivered in the network would compete with the new ones for delivery; besides not computing positions for all nodes in the network within one positioning cycle, an ever increasing number of messages would be temporarily stored across the network, until the total message memory capacity is reached at one node, when starting to irretrievably loose messages. The 10-minute position cycle will have to be validated “a posteriori” against this constraint. Then the “data collection duration” was adjusted at 10 seconds (1 measurement per second). This is imposed by the ambiguity fix rate analyzed in Chapter 4. It is worthwhile mentioning that all nodes need to collect data for a duration long enough to guarantee about 99 % (or whatever success rate is considered acceptable for the application) of integer ambiguity resolution for all baselines. As the positioning processing is deferred to the gateway, no early termination decision can be done locally at the node level. The next parameters to be adjusted are the “random waiting time” and the “random back-off time”. The “random waiting time” to “random back-off time” ratio has been adjusted to 20:1 from earlier tests with a similar connectivity coefficient and message volume. The fact that the optimal values for the assistance phase are different from the optimum values for the reporting phase will be justified later. For the assistance phase, the adjusted values are two milliseconds for the “random waiting time” and 100 microseconds for the “random back-off time”. The assistance phase “random waiting time” was adjusted for a shortest activity time after the first flood message (1.2 s) and the minimum number of repetitions cycles (1) in the *assistance phase*. During this adjustment, the “assistance timeout duration” and “report

timeout duration" are temporarily increased significantly to make sure that all message activity completely dies down before the next timeout and the next retries. For similar reasons, "assistance duration" and "report measurement duration" are temporarily largely increased to guarantee that all repeat cycles will have died down before the end of the phase. The next parameters to adjust will be the "assistance timeout duration" and "report timeout duration", that will be tuned at a duration about 10 % larger than the time it takes for the message activity to die down between repetitions. It is worthwhile to note that the "assistance timeout duration" and the "report timeout duration" will be different. The first one deals with a single assistance message flood while the second deals with multiple report message floods (one per node); it is expected that the second one will be larger than the first one. The adjusted value were 2 s for "assistance timeout duration" and 3 s for "report timeout duration". The last parameters to adjust will be the "assistance duration" and "report measurement duration" to minimize the dead time after which all the repetition cycles have been completed (more precisely after the last messages of the last repetition cycle have died down). The adjustment ended up with 2 s for "assistance duration" and 27 s for "report measurement duration". To come back to the random waiting and back-off times, the adjustment during the *report measurements phase* ended up with "random waiting time" at 0.75 s and "random back-off time" at 37.5 ms for an activity time of about 3 s and the number of repetition cycles at 10. The fundamental reason is that assistance is a single broadcast, where the reporting message is a simultaneous multiple broadcast. The number of cycles needed to report all measurements is proportional to the number of layers. If all messages are correctly transmitted, five cycles minimum are needed, as the device behaves like a Bucket

Brigade Device (BBD), which draws its name from the analogy with the line of people passing buckets of water along a line. With the collision rate, the efficiency is about 50 %.

Finally, some graphs of energy consumption per node were extracted from a full simulation. Figure 49 shows the energy consumption per positioning cycle, parametrized by the distance to the gateway in metres. This plot only shows energy consumption for message transmission and reception; no allowance has been made for position computation, as this would be done at the gateway with unlimited energy resources. The average energy consumption is about 1.355 Joules per cycle per node, with no significant increase closer to the gateway, as expected. Instead, an increase in the fluctuations around the medium value is observed when moving farther from the gateway, but with no significant changes in the median value. With no messages transmitted or received, the minimum floor power consumption is 1.35 Joules. A quick glance at Figure 49 reveals that the proportion of energy spent on message transmission is quite small, from minimum zero to a maximum of 0.04 J with a median at 0.005 J, or about 3 % of the total budget. The real cost of the Centralized Architecture is the increased report measurement time when the receivers need to stay powered on to route the report measurement messages, with multiple collisions. The results from cycle to cycle are not fundamentally different. If the same distribution of energy is assumed from cycle to cycle as in Figure 49, the energy differential between median and worst case node is 2.5% . The risk of premature loss of nodes compared to the average is quite minimal.

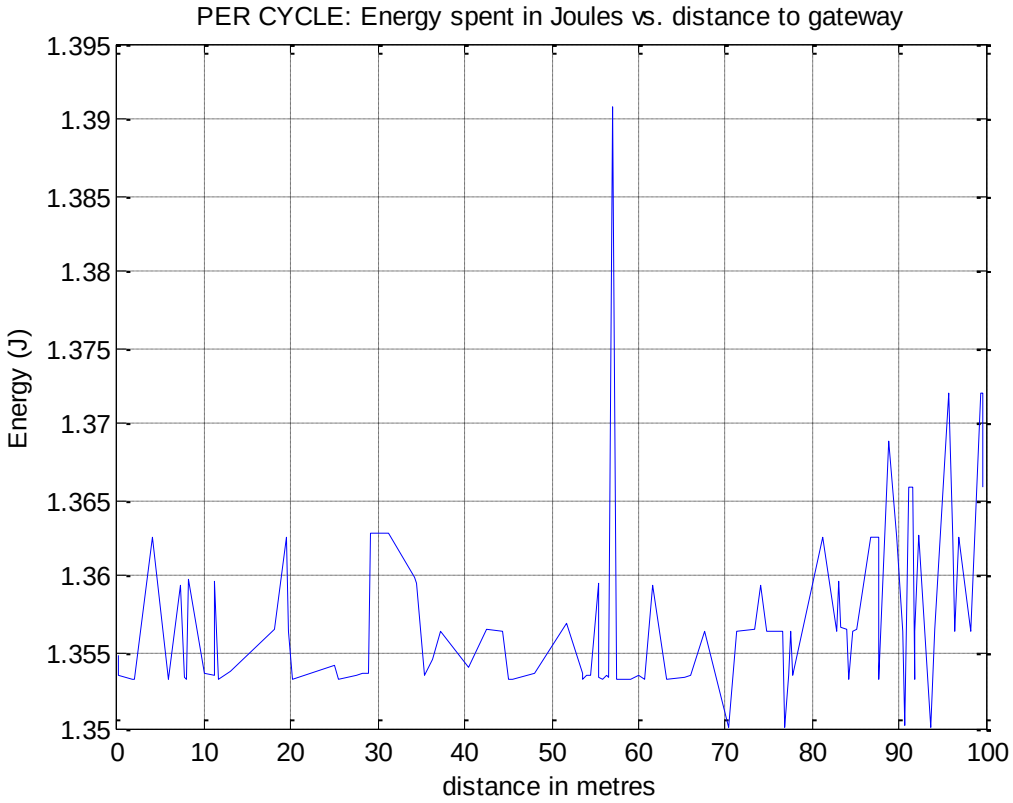


Figure 49 - Per node and per positioning cycle energy consumption vs. distance to gateway

On Figure 50, the number of reported measurement messages is roughly constant regardless of the node distance to the gateway. There are many other interesting observations that can be made in this case, but it will not be elaborated further as the main goal is not the analysis of the behavior of the WSN. As a temporary conclusion, this architecture imposes a ON/SLEEP duty cycle of 7% (42 s / 600 s) for a 10 minute position report rate, where the largest part of the time is spent in message exchanges rather than in data collection. This ratio is directly imposed by the tuning process, which would end up with quite different tuning points for different WSN layouts. This architecture is not very robust as it needs to be tuned individually. It also imposes a large

latency of about 30 seconds between data collection and position availability that is not acceptable for some applications. With an initial energy capital of 31212 Joules (MICA II) with a worst case consumption of 1.39 Joules per positioning cycle per node, the operational lifetime of the WSN without degradation (i.e. not losing any nodes) is about 156 days.

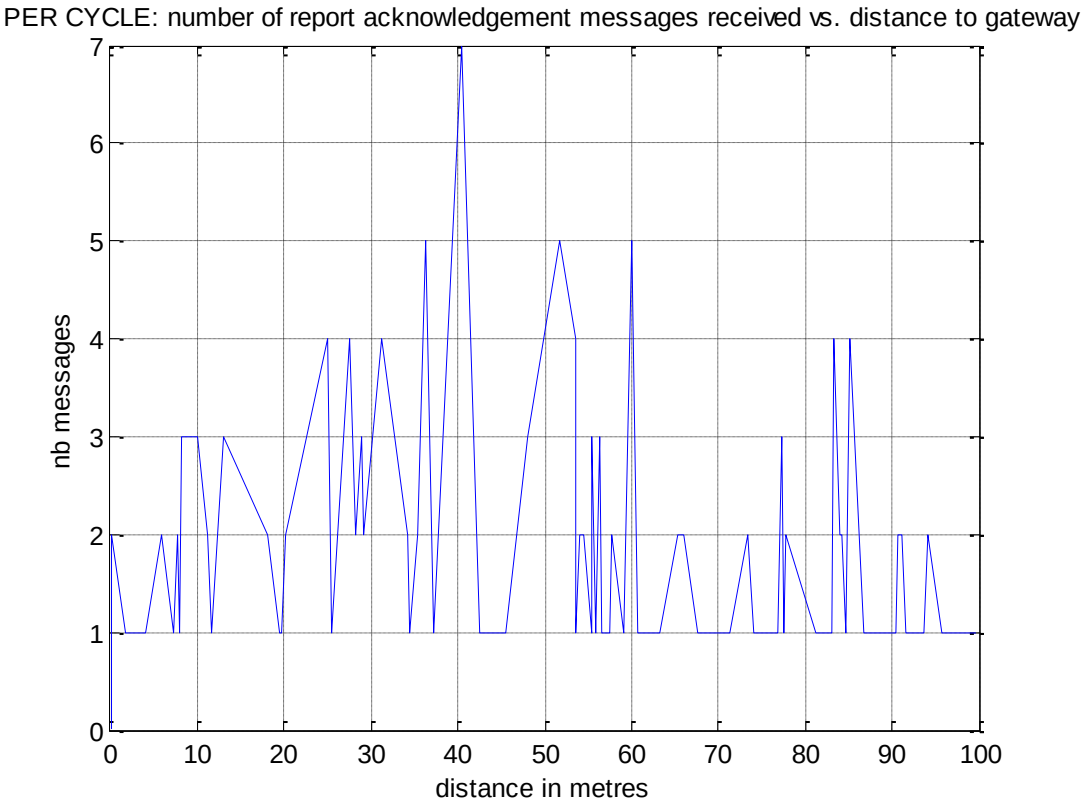


Figure 50 - Number of report message sent vs. distance to gateway

6.4 Simulation 2: Decentralized Architecture without Clusters

This second architecture is one of the candidates for the optimum implementation. This is so since this architecture minimizes the number of report measurement messages traffic to the local neighbourhood. There are two variants that are defined by which nodes are used in the triangle closure. The “local variant” uses one set of measurements from anyone of its parents, its own measurement set, and one extra measurement set received from any of the immediate neighbours. All the measurements are received in short broadcast mode limited to a single hop.

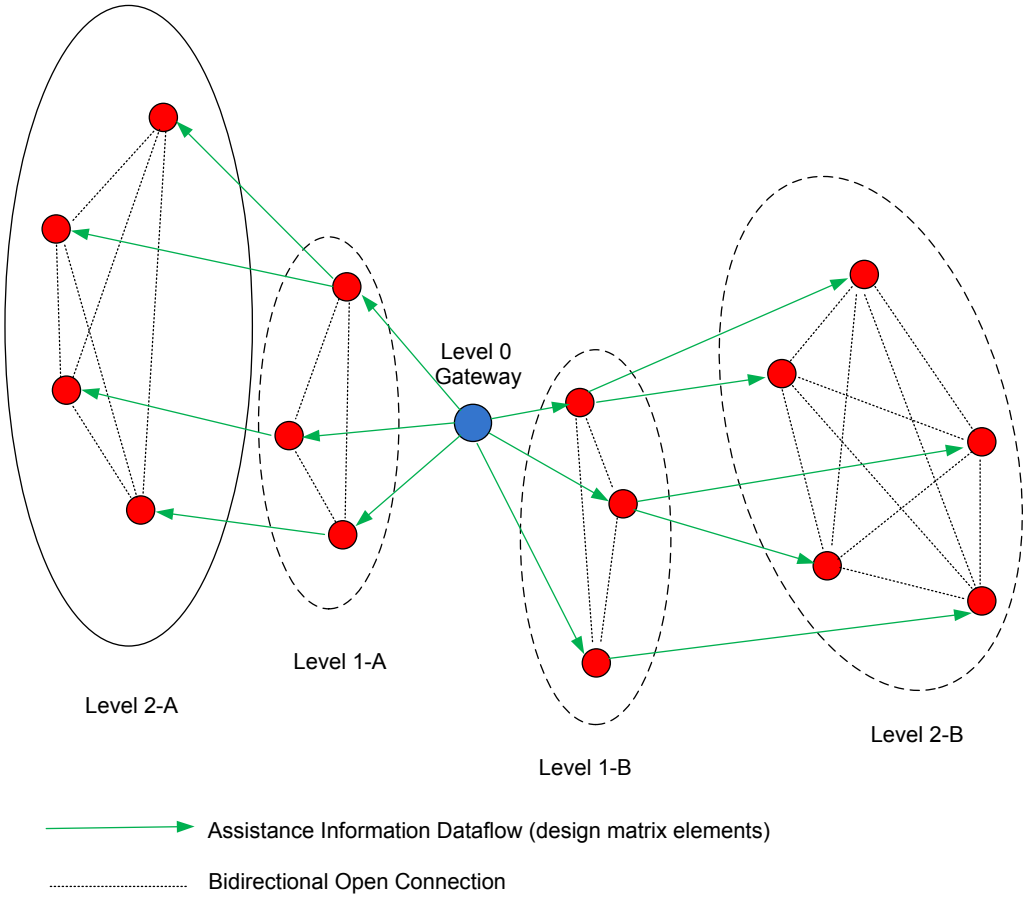


Figure 51 - WSN Decentralized non Clustered Architecture Assistance Flow

The only relative baseline of interest that will be stored by the node for future comparisons is between its parent and itself. The computation of a relative position with the gateway depends on the availability of an uninterrupted path of baselines starting at the gateway and going through several parent nodes (only one at each level), until its reaches the node of interest. Of course, the total position error from the gateway will cumulate all the errors at each baseline. It is still the best solution for relative local displacement detection between neighbouring nodes. There are also some other issues to consider such as the parent node included in the triangle that might change from cycle to cycle, and render the baseline comparison between cycles more difficult, but these

secondary issues will not be explored. Figure 51 and Figure 52 illustrate the assistance and measurement data flows. The related baseline spanning tree is found in Figure 37. As a last comment, the superposition of the closure triangles is not a set of Delaunay triangles. The closure triangles can overlap each other, as the choice of the third node is locally done quite arbitrarily at each node and is totally dependent on the propagation.

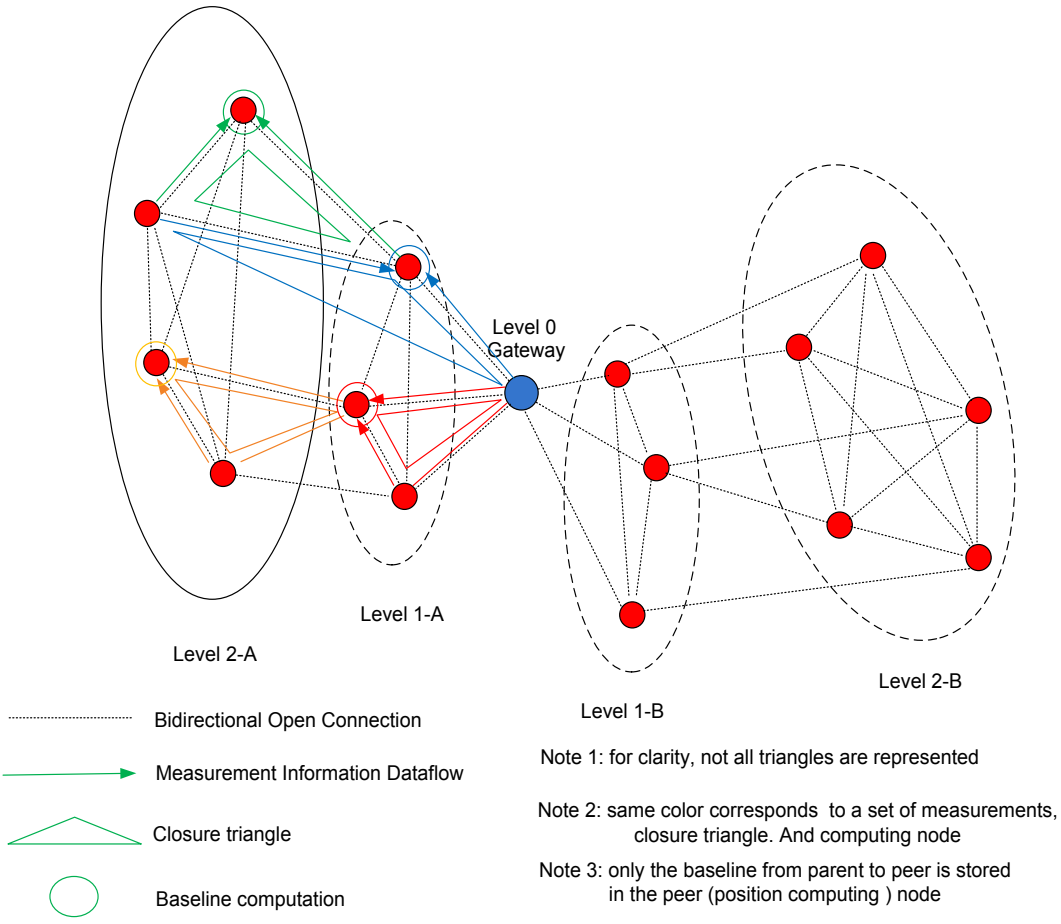


Figure 52 - WSN Decentralised Non Clustered Architecture - Measurement Reporting Data Flow

The second variant, the “global variant”, is defining the ambiguity resolution triangle by the gateway node, the node of interest, plus a third arbitrary node. The node of interest and the arbitrary node are only one hop from each other. Obviously, the global variant does not suffer from the accumulation of the errors from gateway to the node of consideration. It still needs another general flood message during the report measurement phase that sends the gateway measurements throughout the whole network; by analogy with the simulation 1 assistance, it can take about four seconds of extra time right at the report measurement phase before any position computation can be started. It is also expected that ambiguity resolution may be more difficult in the “global variant”, where one node (the gateway) can be quite far from the two other nodes, and therefore not in the same local environment; it can deliver another subset of satellite measurements, not common with the two others, and with different cycle slips qualities. The “local variant” will be the focus in this simulation.

6.4.1 Message Information Content and Size

The assistance phase requires the sending of the list of visible satellites, their relative code offset, and Doppler information. The values of partial derivatives in matrix A are also needed, valid at each measurement cycle of one second. The assumption is that the assistance message provides acquisition assistance time tagged for the beginning of the data collection period, and position computation assistance in the shape of the matrix A coefficients time tagged to be valid at the beginning of the collection period, plus a polynomial formula for extrapolating the coefficients at each second of data measurements.

Table 21 - Decentralized Non Clustered Architecture - Message Contents and Sizes

Message	Content	Size(in bytes)
Assistance Message	Acquisition assistance (3 floats 32-bit per satellite): a-PRN numbers, b-Code offsets, c-Doppler offsets	m x 3 x4 (1)(2)(3)
	Positioning assistance(2 floats per coefficient): a-design matrix A coefficients b-single coefficient polynomial approximation	(m-1) x (3+m) x 2 x 4 (1)(2)(3)
Measurements Message (one set)	Carrier Phase measurements	m x 4 (1)(2)(3)

(1) m, number of satellites is taken at 8 in the whole simulation
(2) all values (assistance data and measurements) are assumed to be in 4 bytes IEEE single float representation
(3) a byte is represented by 8 bits (no parity bit, or any redundancy)

Using the results from Table 18, and making the assumption that five seconds of data are accumulated before resolution is attempted, and that five attempts are made to resolve the linear system, all the individual operations necessary can be summed up. As there is no distinction between additions/ subtractions and multiplications/ divisions in Table 18, the combined sum of the additions/ subtractions will be tallied under the generic name of "additions" and the sum of the multiplications/ divisions categories under the generic name of "multiplications".

Table 22 - Total Additions and Multiplications for a Single Baseline Processing

Operation	Multiplications Number	Additions Number
Double difference computation	112	105
Cofactor Matrix Computation	784	735
Weight Matrix Computation by Gauss Jordan Elimination	205	162
First Partial Results Computation	588	504
First Partial Results Accumulation	0	11
Second to tenth partial results computation (1)	1078 x 9	924x9
Second to tenth Partial Results Accumulation (1)	0 x 9	121 x 9
Linear System Resolution by Gaussian Elimination (2)	561x 5	121 x 5
TOTAL Operations	**14106**	**13397**

(1) It is assumed that up to 10 seconds of data are accumulated by steps of 1 second. The first accumulation is a special case, and has other computation constraints.

(2) It is assumed that the first attempt to resolve the ambiguities will occur only when at least 5 seconds worth of data have been accumulated. Only 5 Linear System Resolutions by Gaussian Elimination will be attempted.

Table 23 - Total Processing Time for a Single Baseline

	Number	Number of cycles per operation	Time @16Mhz
Multiplications	14106	2851 (1)	2.5135 seconds
Additions	13397	80 (2)	0.0670 seconds
Total time for a single baseline			**2.5805 seconds**

(1): the worst case of Table 16 between multiplications and divisions was assumed.

(2): the worst case of Table 16 between additions and subtractions was assumed.

As a general conclusion, on a MICA II, for three baseline resolutions, the processing time is counted as three times the single baseline processing time, or about 7.75 seconds that consume 0.186 J per cycle.

6.4.2 Comments and results

The active phases are assistance, data collection, data reporting and finally position computation.

The tuning process converged to the values listed in Table 24.

Table 24 - Decentralized Non Clustered Architecture-Parameter Tuning

Parameter	Duration or number of iterations
Random waiting time during assistance phase:	2 ms
Random back-off time during assistance phase:	100 μs
Random waiting time during report phase:	1.6 s
Random back-off time during report phase:	80 ms
Assistance phase duration:	2 s
Collection phase duration:	10 s
Report measurement phase duration:	4 s
Position computation phase duration:	8 s
Assistance Request timeout	2 s
Report measurement repeat interval	2s
Report measurement maximum repetition number	2

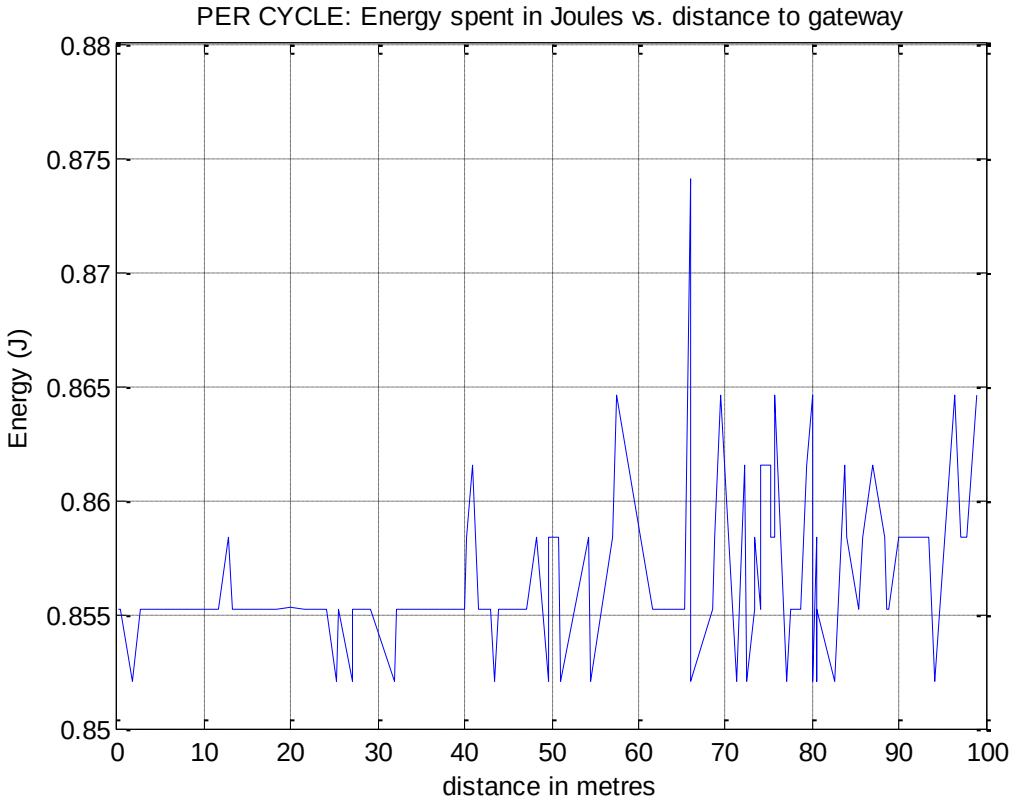


Figure 53 - Decentralized Architecture - Energy vs. Distance to the Gateway

The positioning success rate is not 100 % but between 95 to 100 %, only because of the reporting message collisions. The reporting messages are repeated several times ("report measurement maximum repetition number") at constant time intervals ("report measurement repeat interval") to increase the probability to successful delivery, in spite of the hidden node problem. Figure 53 shows the energy consumption per cycle at about 0.852 Joules of floor consumption. The median value is 0.855 J, and the peak to median ratio is 1.7 %. The interim conclusions are that the Decentralized Architecture has an advantage of 38% over the centralized one in terms of energy and lifetime or 253 days of full autonomy. The distribution of the energy consumption across nodes is quite comparable , at about 1.8 %, or a time difference of about 4 days between the first node

loss and the loss of the majority of the nodes. The decentralized method has a latency of 8 s to compare to the 30 s for the centralized one. It would be relatively easy to further increase the performance gap if the position computations were implemented in fixed arithmetic or in hardware floating point Arithmetic and Logic Unit (ALU). The position computation phase could be easily reduced to 1 second, with a floor consumption of 0.684 J and estimated median value of 0.686 J and a lifetime of 316 days. The fixed arithmetic single positioning implementation has been already done in the past, with much larger dynamic ranges for the pseudorange representation. The implementation of the carrier phase relative positioning in fixed arithmetic is not expected to be challenging either. The procedure, albeit reducing the number of inter node interactions, still calls for the computation of three baselines for only one that is actually needed, to benefit from the ambiguity closure properties. These extra baselines and the energy spent to compute them are wasted, as the choice of the third node is uncontrolled, and will vary from position cycle to position cycle, therefore cannot be used for cycle to cycle position monitoring. The Decentralized Architecture with Clusters has been designed to overcome this hurdle.

6.5 Simulation 3: Decentralized Clustered Architecture

The main goal of this architecture is not to better equalize the energy consumption between nodes, but rather to reduce the baseline redundancy, and thus the power consumption, by centralizing the baselines decisions in a single cluster head of a local

zone.

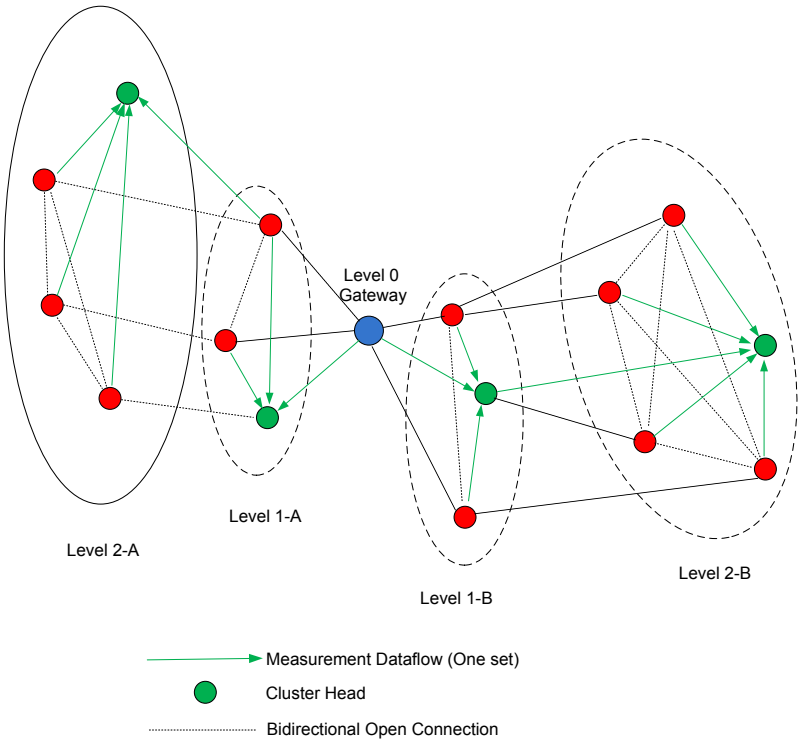


Figure 54 - WSN Decentralized Clustered Architecture - Measurement Reporting Data Flow

The clusterization goal in this case is quite different from the traditional one such as the LEACH protocol (Handy 2002). LEACH reduces the overall consumption by substituting a message transmission over a single large hop by a transmission over several smaller ones. It has been demonstrated that in a propagation model non linear over distance the overall energy budget is smaller for multiple smaller hops. Rather than transmitting to a distant node, all non cluster head nodes transmit at short distance to the cluster head. The cluster head has to spend more energy to forward all messages to the next cluster head, but the overall net energy consumption is still in favor of the clustered architecture. The operation results in a net gain in the aggregated energy consumption of the whole

network. This consumption saving is definitely not for the cluster head, which needs to be periodically relieved of its role and replaced by another node of the cluster to equalize the average power consumption within the cluster. The aim in this case is a reduction of the number of baselines to compute within the cluster by comparison to the Decentralized Non Clustered Architecture.

The layout of the computed baselines is shown in Figure 38. The assistance data flow is identical to the other solutions. Only the measurement data flow differs, as illustrated in Figure 54.

To implement this architecture, more phases are required, Assistance, Data Collection, Cluster Challenge, Cluster Decision, Position Computation and finally Position Reporting phases.

The departure from the Decentralized Architecture starts at the cluster challenge phase, where each node broadcasts its measurements and its own level of remaining energy. Each node can compare its remaining energy with the remaining energies in the other nodes of the cluster found in the challenge messages. The cluster head decision is done in parallel at each node, on the basis of the same data available to everyone. The newly elected cluster head then broadcasts back its cluster head status and the list of node IDs it considers part of the cluster, and takes responsibility to compute their positions; this occurs in the Cluster Decision phase. This extra phase is implemented to manage the ambiguous situations such as described in Section 5.5.6. Please also note the piggybacking of the measurement reporting over the Cluster Challenge messages; this allows for less message collisions and minimizes the need for repetitions. The position

computations are done in the cluster head only. It will definitely take more than the 8 seconds for a single triangle resolution. It is imperative that all the non cluster head nodes go back to the sleep mode right at the end of the Cluster Decision Phase. The last difficulty is about the position reporting back to the non-cluster head nodes. If the monitoring of the variation of the relative positions is made at each non-cluster head, they could wake up at the end of the position computation phase for receiving their current position. A second option is to wait until the beginning of the next cycle for the cluster head to retransmit the positions of the last cycle; in this case the monitoring latency is almost as large as the positioning cycle period. A last most preferred solution is for each node to broadcast its last relative position in the same message used for the Cluster Challenge. The cluster head will do the comparison with the last positions, and will trigger itself the alarm message if necessary before going to sleep. The position of the last cycle is returned to each node at the beginning of the next cycle as in option two.

A last piece of information is required before going to simulations, namely how to choose the baselines in a cluster. The improvement over the Decentralized Architecture in the number of effective baselines to compute, and the potential consumption improvement will be also examined.

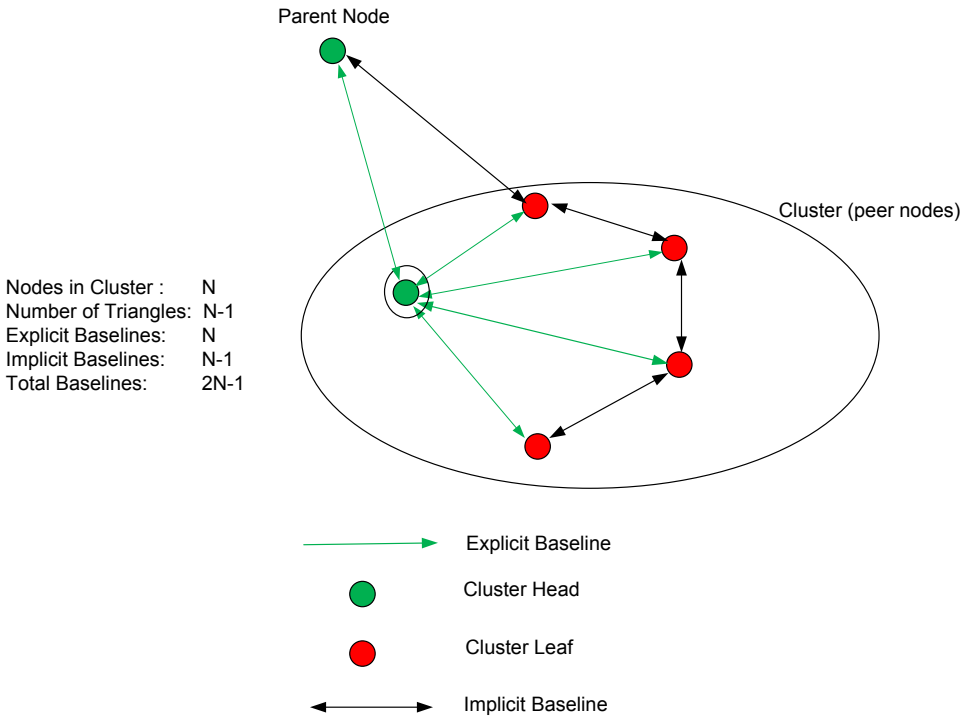


Figure 55 - Decentralized Clustered Architecture - Span Tree and Number of Baselines

In a network of N nodes, the total number of explicit edges in any span tree is (N-1). The decentralized solution computes three baselines for each explicit edge of the span tree, i.e. 3 (N-1) baselines. In a cluster with n nodes including the cluster head, each cluster head has to compute n+(n-1) baselines (see Figure 55). This counts all explicit baselines linking the cluster head to each other node in the cluster, plus one link with one parent to connect the new sub span tree to the main span tree, plus the implicit (n-1) extra edges to form n triangles from the sub span tree. If C clusters are present in an N node network, the number of nodes per cluster is n=N/C and the total number of baselines is 2N-C. For a network of N=100 nodes, and C=5 clusters, the reduced number of baselines to compute is 195, as compared to 297 baselines in the simple decentralized case which

translates in a 34% improvement. One still needs to demonstrate that this potential gain is not compromised by the overhead imposed by the cluster management algorithm.

The edge-to-node ratio is $2 - \frac{1}{n}$ for the Decentralized Clustered Architecture and 3 for the Decentralized Non Clustered one. This ratio can be interpreted as the average number of baselines each node has to compute at each positioning cycle. At the asymptotic limit of n going to infinity, the ratio is 2/3 at the clustered architecture advantage.

6.5.1 Message Information Content and Size

As for the other simulations, a list of the messages, their size, and their content is shown in Table 25.

Table 25 - Decentralized Clustered Architecture - Messages

Message	Content	Size (in bytes)
Assistance Message	Acquisition assistance (3 32 bit floats per satellite): a-PRN numbers, b-Code offsets, c-Doppler offsets	m x 3 x 4 (1)(2)(3)
	Positioning assistance: design matrix A coefficients and single coefficient polynomial approximation (2 floats per coefficient)	(m-1) x (3+m) x 2 x 4 (1)(2)(3)
Challenge Message	Node Energy Level: 1 float	4
	Carrier Phase measurements m 32-bit floats	m x 4 (1)(2)(3)
	Past positions for all ID in cluster: 3 32-bit floats per ID	c x 3 x 4 (2)(3)(4)
Cluster Decision Message	Decision flag	1
	List of node IDs in the cluster (one 16-bit integer per node)	c x 2

(1) m, number of satellites is taken at 8 in the whole simulation.
(2) all values (assistance data and measurements) are assumed to be in 4 bytes IEEE single float representation.
(3) a byte is represented by 8 bits (no parity bit, or any redundancy).
(4) c, number of nodes in cluster (including cluster head).

6.5.2 Comments and results

The active phases are Assistance, Data Collection, Challenge and Report, Cluster Decision and finally Position Computation (for a limited number of cluster head nodes).

The tuning process converged to the values shown in Table 26.

Table 26 - Decentralized Clustered architecture - Parameter Tuning

Parameter	Duration
Random waiting time during assistance phase:	2 ms
Random back-off time during assistance phase:	100 µs
Random waiting time during non assistance phases:	1.6 s
Random back-off time during non assistance phases:	80 ms
Assistance phase duration:	2 s
Collection phase duration:	10 s
Challenge report phase duration:	4 s
Cluster decision phase duration	4s
Position computation phase duration:	2.6 s (per node/per baseline)
Assistance request timeout	2 s
Challenge report repeat interval	2 s
Cluster decision repeat interval	1s

A first issue was found about the cluster head assignment algorithm. In order to come up with the same decision at each node, it is imperative that all use the same data during the decision process. The minimum consumed energy among peer nodes has been elected as the cluster head choice criterion, assuming that the initial energy budget is the same for all nodes. If the set of nodes has exactly the same level of consumed energy at challenge message time, all will report the same energy level. But at cluster head decision time, they will compare their current consumed energy level to the energy levels found in the received challenge messages. Unfortunately the local current consumed energy will be always higher than energy levels in received challenged messages. There will always be at least an extra challenge message transmission, plus at least one extra challenge

message reception energy cost, both having occurred since the challenge message transmission. No node will declare itself cluster head and this will break the protocol. One solution is to sample all consumed energy levels at the beginning of the challenge phase, and to report only this energy level in any challenge message, regardless of the number of repetitions or new requests. Figure 56 illustrates the energy consumption per node, after the first position cycle, ordered according to the distance to the gateway. The average consumption is 1.2077 J across all nodes. The maximum energy difference between the best and worst case is about 1.3 J, as Figure 57 shows, or 100 % of relative difference. The peaks of energy consumption are larger when closer to the gateway, which is ascribed to the higher collision rate, due to the higher node concentration.

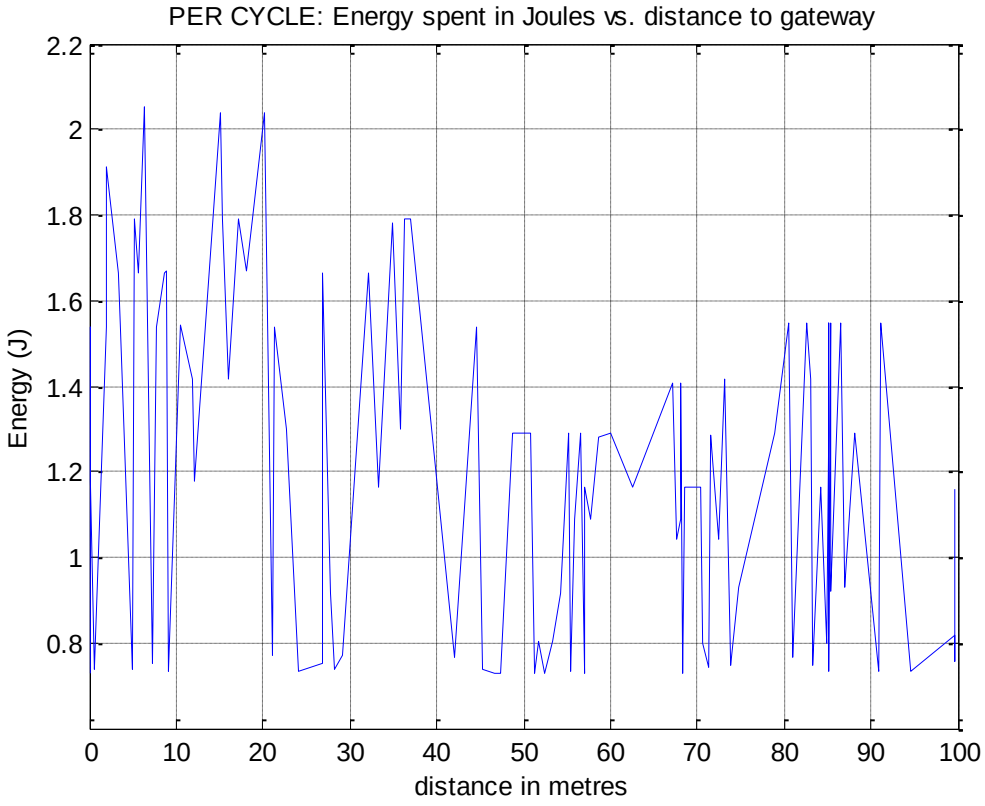


Figure 56 - Decentralized Clustered Architecture - Energy Consumption per Node at First Cycle vs. Distance from Gateway

Another issue arises from the fact that no message has a guaranteed delivery. A node might miss a cluster decision message informing they are part of a cluster and that the cluster head will compute a baseline from itself to this node. In this case, the node decides to compute a single baseline from one of its parents to itself, in a single triangle configuration. To do so, measurements from at least one parent node and any other node are mandatory to run the computation or this node is excluded from the computation cycle. Figure 58 shows an example of per node cumulated energy consumption over a period of about half a day (72 cycles).

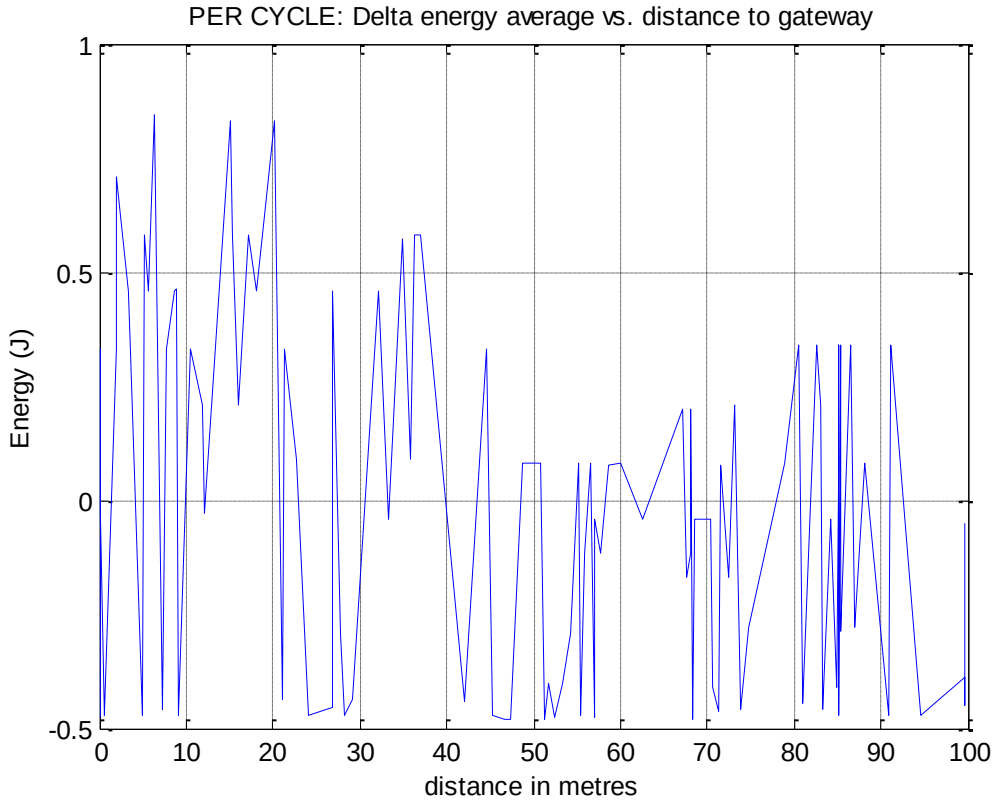


Figure 57 - Decentralized Clustered Architecture - Delta Energy Average vs. Distance to Gateway

The nodes are partitioned into very distinctive groups, according to their consumption levels. On the low side with a consumption of 52 Joules at the 72th cycle, a single node is not associated with any cluster, and does its one baseline computation at each cycle; because of a large distance to the gateway, there is a limited number of repetition messages, and thus the energy consumption is close to the minimum. On the other extreme, two nodes can be found at 90 Joules at cycle 72. They are not associated to any cluster or to each other compute their own baseline, but have more repetition messages,

which explains their high consumption. In between, the second cluster from the top is a group of three nodes that are managing their energy as a group, keeping the average consumption even among the cluster.

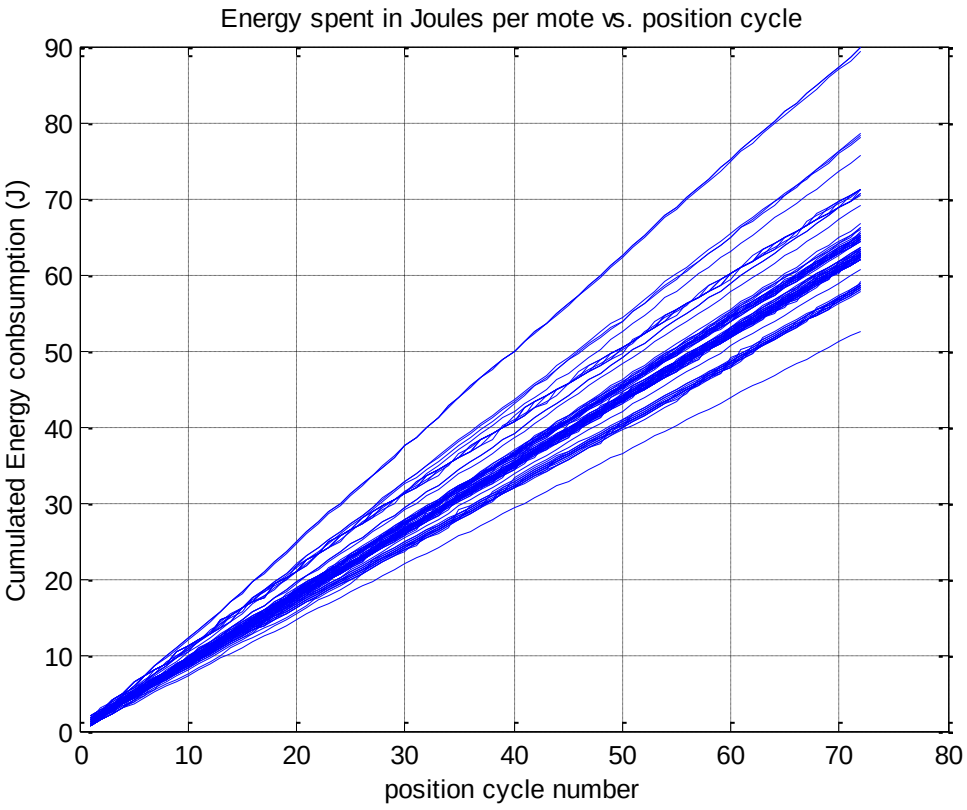


Figure 58 - Decentralized Clustered Architecture - Cumulated Energy Consumption per Node

Energy consumption at all nodes in a same cluster is always kept within 1 J between nodes. This can be observed in Figure 59, where the general upward tendency has been eliminated to emphasize the differences. The node with the lowest energy at the

beginning of the cycle will be solicited the most (i.e. elected cluster head) for the current cycle. This is the result of energy regulation within clusters.

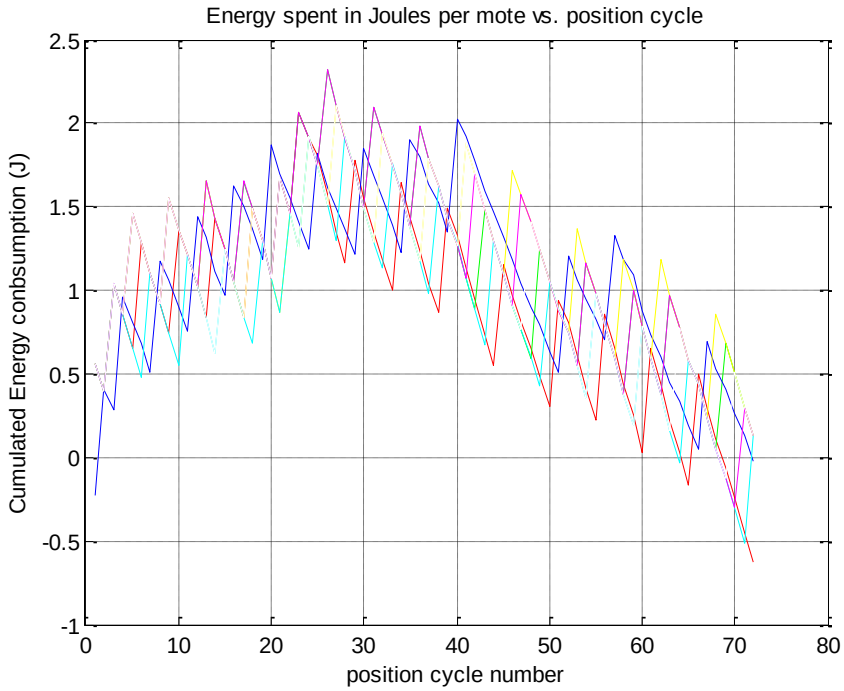


Figure 59 - Single Cluster Compared Cumulated Consumptions - General Tendency eliminated

A general conclusion is that this power management works quite well inside clusters, but this fails to equalize the energy between clusters. The consumption difference reaches about 70 % between the lowest and the highest case at cycle 72.

6.6 Chapter Conclusion

The main discovery emerging from these simulations is that the energy consumption is mainly due to the time the nodes are kept powered on and in a lesser measure, to the transmit time, and the number of exchanged bits. It was also practically shown how much a broadcast followed by specific requests to compensate for the messages lost to collision is much more efficient that a multiple repetition of the same broadcast message in terms of duration for a given success rate.

The Decentralized Clustered Architecture that has been applied successfully to the problem of the energy equalization for message transmission does not translate too well to position computation. The theoretical reduction of about one third of the number of baselines to compute is offset by the energy cost of the double short broadcast messages, and the limited size of practical self-forming clusters. The best solution so far is the Decentralized Non Clustered Architecture with a very constant 0.855 J consumption per cycle, which does not vary much with the location in the network. All nodes repeat their messages the same number of times and keep the message length identical by design. The uncertainty about which messages are received is compensated by the opportunistic approach provided that the position computation is possible for the node with measurements for at least one parent plus any other node. The two extra nodes also can change from cycle to cycle with no impact. This will complicate the monitoring of the relative positions, but it is not the main emphasis of this work.

The second best solution is the Decentralized Clustered architecture, with an average energy consumption of 1.2 J. In this case, all nodes that are loosely connected to the rest of the network will fail early (30 % earlier than the average), but the network will continue to perform without the missing nodes.

The worst solution is the Centralized Architecture, with 1.445 J per node per cycle. The inter nodes spread is quite small. It was not possible to extend the simulation until the loss of the first node (referred in the literature as FND or First Node Dies, see Handy 2002). Keeping in mind that under the Prowler simulation, a single processor simulates the activity of hundred of them, a real-time-to-simulation-time ratio of 1:1 was achieved. To reach the FND condition would mean a continuous simulation for 148 days, which is obviously not practical.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

The first contribution of this research is an accurate time resolution and synchronization technique (ARTI) primarily suitable for low cost simple architecture Wireless Sensor Nodes in the relative carrier phase double difference mode. This technique does not require one to run any code based single positioning solution as a preliminary step. That the relative positioning accuracy does not degrade when applying this technique has been demonstrated. It was also shown that virtually any inaccurate Wireless Sensor Network inter-node synchronization technique available is sufficient to bootstrap ARTI. This technique is an evolution in the carrier phase relative positioning domain of a code based technique (Sirola 2001) that resolves the transmit time when the timing information cannot be decoded from the GPS navigation message, by measuring only the sub millisecond code phases, and extending it to full pseudoranges in full millisecond equivalents of 300 km with systematic exploration of multiple hypotheses of location on the earth, searching for the one with lowest residuals. Syrjärinne (2000) proposed another technique that integrates the system time as a fifth unknown in the position solution, but still requires an approximate time known within a few milliseconds, obtained by time

synchronization in a cellular network in an A-GPS type of implementation. These early methods are widely used in consumer high sensitivity receivers which receive signals so low that the data decoding is unfeasible, or because the time to decode the data is considered prohibitive in view of the desired very fast time to first fix. In contrast, ARTI is not meant for high sensitivity or harsh environment deployment. It tries to minimize the number of constraints and the complexity at each node to maximize their chances of implementation on a very small hardware platform. ARTI does not require any satellite position determination at each node, any navigation message reception and decoding, and any single positioning algorithm. Only cumulated carrier phase and sub millisecond code offset measurement are needed. Its implementation blends well with the mainstream ambiguity resolution techniques. The extra time ambiguity is not distinguishable from any other double difference ambiguity, and is simply added to the number of ambiguities to resolve.

The integer ambiguity closure was implemented according to the method introduced for faster resolution by Luo & Lachapelle (2003). A set of three baselines arranged in a closed triangle and using the property of ambiguities closure over a closed path of baselines is faster to resolve and with better reliability. Therefore, all the proposed architectures have sought to compute baselines as groups of minimum three baselines. The baselines grouping does not follow a Delaunay partitioning in general; there is no knowledge of distances at least at the local node level, and the grouping is somewhat random depending on the conditions of reception. Based on ARTI and the triangle closure requirements, a WSN data flow architecture that implements these two results in a natural way was proposed. It is natural insofar as it blends and piggybacks with the

fundamental WSN protocols such as discovery, by message flooding with minimal additions. This architecture is robust as it imposes the minimum of constraints on the message exchanges, in particular the persistence of bidirectional links between neighbouring nodes. Practical ways to mitigate the large latency between measurements and position delivery were also shown.

ARTI is well suited for WSN implementation with CSMA/CA protocols. Its data flow requirements are a natural complement to the protocols designed for WSN for the last ten years.

ARTI can recover at least 100 milliseconds of initial synchronization error. This is largely sufficient to be provided by any WSN synchronization protocol available today, including the TPSN protocol.

The second contribution is a demonstration that the Decentralized Non Clustered Architecture is a recommended implementation method in a WSN that blends position computation with the WSN limitations in an unified protocol. Among the three proposed solutions, it is the best in terms of relative power consumption, performance robustness, and implementation simplicity. Putting aside the message transmission energy improvements that are only 20 % of the total energy budget, the clustered architecture does not seem to live up to its expectations of maximizing the operational lifetime of the network. This apparent contradiction becomes clearer when one realizes that the best positioning monitoring strategy is to collect, process and store all information locally. The clustered architecture principal goal is to move all decentralized information toward

the gateway in the most energy efficient way. The positioning monitoring application is quite satisfied with very infrequent anomaly reports sent to the gateway; the data shuffling protocol performance becomes irrelevant due to its very infrequent invocation.

7.2 Recommendations

The limitations of ARTI were not explored to the breaking point, as it was sufficient for the intended application. It would be wise to push to the limits, in particular the influence of the synchronization errors, and their impact on the ambiguity resolution.

Only the positioning techniques were explored in detail in this work. Other functions need a very detailed analysis, namely signal acquisition and measurement extraction techniques. The frequency drift of each node clock is a concern and should be addressed, as it cannot be measured well enough across multiple hops. The extended Doppler search domain around the assistance information delivered by the gateway needs to be examined very carefully. Radically novel techniques need to be designed in the framework of a very energy and computational power limited platform. A cooperative search algorithm where each node in a neighbourhood thoroughly searches only for a subset of satellites, then passes on the information to its immediate neighbours to speed up acquisition of these same satellites is worth consideration.

The impact of multipath errors on the convergence rate should also be analyzed, although no major departure from the performance of the well-established solution is anticipated.

The protocol analysis and tuning was done in a limited scope and would benefit by being revisited, in particular the random time delays and the random back-off delay, as these are critical for maximizing the throughput on the shared medium, yet they do not have known close form solutions, at least in a plane-limited configuration.

A fine-grain validation on a cycle accurate simulation of a MOTE II platform, TOSSIM (Levis et al. 2003) or TOSSF (TinyOS Scalable Simulation Framework, Perrone 2002), implemented in the native TinyOS language and running on the same operating system, would be the last step before actual deployment.

Another important point is the quality of the antennas. It does not make sense to use survey grade antennas with very well controlled centre of phases with very low cost nodes. There is a need to find a way to use compact small dimension antennas with acceptable performance in carrier phase differential techniques.

Additional improvements could be explored. In particular, the main issue with the CSMA/CA protocols is the collision problem, which surely is detected and circumvented by medium activity sensing before transmission and random backoff, but still considerably impacts the energy budget due to the massive amount of message redundancy. The very high quality common synchronization accuracy across all nodes achieved through ARTI could be put to better use by designing a synchronized TDMA MAC protocol method around it. It could be also exploited in a variant of the location aware routing protocols already proposed by targeting only the nodes that are in the general direction of the final receiver of a message.

This work could be also extended to applications with lesser positioning accuracy requirements, typically at the few metres level, using code phase only and limiting the integer search to transmit time millisecond ambiguities. This would reduce the implementation effort at each node to the sub-millisecond code phase measurements. The positioning algorithms could be formulated in relative code phase and the data flow kept very similar.

A project that came to the attention of the author after this work was started, namely "GGPhi", proves the practicality and the need for such a solution in the market. "GGPhi" is a Low-Cost, Low-Power Galileo/GPS Carrier Phase receiver that was initiated under the European GNSS Supervisory Authority, and led by the University of Leeds and the University of Nottingham (Aguado 2007). The applications are similar, namely land-slip monitoring in developing countries; the solutions are comparable using a minimal codeless snap shot carrier phase only receiver, and a wireless network time transfer. The proposed solution is similar, with a WSN architecture, a hybrid CSMA/CA and TDMA MAC protocol, a clustered architecture around a modified LEACH protocol, with centralized processing at the gateway. The TPSN synchronization protocol has been retained for the inter node synchronization, for reasons similar to the rationale of this study.

APPENDIX 1:

Cumulated Double Difference Carrier Phase partial Derivative Formulas

The linearization of the formulas introduced in Chapter 4 necessitates the computation of several partial derivatives of double difference carrier phases computed at reference times. The formulas below are unconventional in the sense that the reference time for measurements at the master is the nominal receive time ($t_{master,0}$), and reference times for measurements at secondary receiver are the transmit times per satellite ($t_0^{p,second}, t_0^{q,second}$).

These partial derivatives appear in the matrix A.

(A1.1)

$$\frac{\partial \Phi_{ms}^{pq}}{\partial x}\left(t_0^{p,second}, t_0^{q,second}, t_{master,0}\right) = \frac{x^p\left(t_0^{p,second}\right) - x_{second}}{\rho_{second}^p\left(t_0^{p,second}\right)} - \frac{x^q\left(t_0^{q,second}\right) - x_{second}}{\rho_{second}^q\left(t_0^{q,second}\right)}$$

$$\frac{\partial \Phi_{ms}^{pq}}{\partial y}\left(t_0^{p,second}, t_0^{q,second}, t_{master,0}\right) = \frac{y^p\left(t_0^{p,second}\right) - y_{second}}{\rho_{second}^p\left(t_0^{p,second}\right)} - \frac{y^q\left(t_0^{q,second}\right) - y_{second}}{\rho_{second}^q\left(t_0^{q,second}\right)}$$

$$\frac{\partial \Phi_{ms}^{pq}}{\partial z}\left(t_0^{p,second}, t_0^{q,second}, t_{master,0}\right) = \frac{z^p\left(t_0^{p,second}\right) - z_{second}}{\rho_{second}^p\left(t_0^{p,second}\right)} - \frac{z^q\left(t_0^{q,second}\right) - z_{second}}{\rho_{second}^q\left(t_0^{q,second}\right)}$$

$$\frac{\partial \Phi_{ms}^{pq}}{\partial N_{second}}\left(t_0^{p,second}, t_0^{q,second}, t_{master,0}\right) = \left(\frac{\partial \rho_{second}^p}{\partial N_{second}}\left(t_0^{p,second}\right) - \frac{\partial \rho_{second}^q}{\partial N_{second}}\left(t_0^{q,second}\right)\right)$$

The last equation, expressed in metres/unit, can also be rewritten:

$$\frac{\partial \Phi_{ms}^{pq}}{\partial N_{second}}\left(t_0^{p,second}, t_0^{q,second}, t_{master,0}\right) = \left(\frac{\partial \rho_{second}^{p}}{\partial t_0^{p,second}}\left(t_0^{p,second}\right) - \frac{\partial \rho_{second}^{q}}{\partial t_0^{q,second}}\left(t_0^{q,second}\right)\right).10^{-3} \quad \text{(A1.2)}$$

In order to compute these partial derivatives, a value of the geometric distances $\rho_{second}^{p}\left(t_0^{p,second}\right)$ (resp. $\rho_{second}^{q}\left(t_0^{q,second}\right)$) is calculated. It represent the geometric distance from satellite p (resp. q) to the secondary receiver at nominal $t_0^{p,second}$(resp. $t_0^{q,second}$).

The partial derivatives $\frac{\partial \rho_{second}^{p}}{\partial t_0^{p,second}}\left(t_0^{p,second}\right)$(resp. $\frac{\partial \rho_{second}^{q}}{\partial t_0^{q,second}}\left(t_0^{q,second}\right)$) also need to be computed. The details of these computations can be found in Appendix 2.

APPENDIX 2:

Geometric Distance between Satellite and Secondary Receiver and its First Derivative vs. Transmit Time

This section describes the procedure employed to derive the nominal geometric distances from satellite to receiver and their derivatives vs. time. The usual formulas that are parametrized with the receive time at the receiver will be derived first, then the not so usual formulas that are expressed as a function and a derivative of the transmit time at the satellite will be derived. The general formulas for converting position, velocity and acceleration from a rotating reference system to an inertial reference system will be introduced first. The next step will be a detailed description of the different inertial and non-inertial coordinate reference of interest for these developments. The formulas for the receive time at receiver then for the transmit time at satellite will finally be given.

General conversion for ECEF to inertial coordinates

First, let's introduce the general conversion formulas for Position, Velocity and Acceleration from Rotating Reference Frame into Inertial Reference Frame valid for any mobile, regardless of the used inertial reference system:

$$X_{INERT}(t) = R_{INERT,ECEF}(t).X_{ECEF}(t) \qquad \text{(A2.1)}$$

$$\dot{X}_{INERT}(t) = R_{INERT,ECEF}(t).\dot{X}_{ECEF}(t) + \dot{R}_{INERT,ECEF}(t).X_{ECEF}(t) \quad \text{(A2.2)}$$

$$\ddot{X}_{INERT}(t) = R_{INERT,ECEF}(t).\ \ddot{X}_{ECEF}(t) + \dot{R}_{INERT,ECEF}(t).\dot{X}_{ECEF}(t) + \ddot{R}_{INERT,ECEF}(t).\ \ddot{X}_{ECEF}(t) \quad \text{(A2.3)}$$

where

$X_{ECEF}(t), \dot{X}_{ECEF}(t), \ddot{X}_{ECEF}(t)$

Vectors Position, Velocity and Acceleration of an object in the ECEF reference system at time t

$X_{INERT}(t), \dot{X}_{INERT}(t), \ddot{X}_{INERT}(t)$

Vectors Position, Velocity and Acceleration of an object in the Inertial reference system at time t

$R_{INERT,ECEF}(t)$, $\dot{R}_{INERT,ECEF}(t), \ddot{R}_{INERT,ECEF}(t)$

Rotation matrix from ECEF to Inertial Reference Systems and their derivatives at time t

In the only case of interest here, where the mobile is the secondary receiver (with coordinates in reference system ECEF, rotating with the earth), inertial and ECEF rotating frames share the same z axis (this is the rotation axis). If $\alpha(t)$ is the variable rotation angle between ECEF x axis and INERT x axis, the rotation matrix is expressed by:

$$R_{INERT,ECEF}(t) = R_{INERT,ECEF}\big(\alpha(t)\big) = \begin{bmatrix} \cos\alpha(t) & \sin\alpha(t) & 0 \\ -\sin\alpha(t) & \cos\alpha(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A2.4}$$

and its derivatives by:

$$\dot{R}_{INERT,ECEF}(t) = \dot{R}_{INERT,ECEF}\big(\alpha(t)\big) = \begin{bmatrix} -\sin\alpha(t).\dot{\alpha}(t) & \cos\alpha(t).\dot{\alpha}(t) & 0 \\ -\cos\alpha(t).\dot{\alpha}(t) & \sin\alpha(t).\dot{\alpha}(t) & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{A2.5}$$

$$\ddot{R}_{INERT,ECEF}(t) = \ddot{R}_{INERT,ECEF}\big(\alpha(t)\big) = \begin{bmatrix} -\cos\alpha(t).\dot{\alpha}(t)^2 - \sin\alpha(t).\ddot{\alpha}(t) & -\sin\alpha(t).\dot{\alpha}(t)^2 + \cos\alpha(t).\ddot{\alpha}(t) & 0 \\ \sin\alpha(t).\dot{\alpha}(t)^2 - \cos\alpha(t).\ddot{\alpha}(t) & -\cos\alpha(t).\dot{\alpha}(t)^2 - \sin\alpha(t).\ddot{\alpha}(t) & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{A2.6}$$

For example, if the inertial reference frame is the one defined in the ICD 200 specifications [ICD-GPS-200D], where the x axis points towards the vernal point (or ascending node between Earth orbit plane and Celestial Equator), the $\alpha(t)$ angle is:

$$\alpha(t) = \Omega_0 + \left(\dot{\Omega} - \dot{\Omega}_e\right).t - \dot{\Omega}_e.t_{oe}, \tag{A2.7}$$

$$\dot{\alpha}(t) = \left(\dot{\Omega} - \dot{\Omega}_e\right), \tag{A2.8}$$

$$\ddot{\alpha}(t) = 0 \tag{A2.9}$$

and

$$\dot{\Omega} = \dot{\Omega}_{ref} + \Delta\dot{\Omega} \tag{A2.10}$$

where

$\alpha(t)$ angle between x axis in ECEF and x-axis in inertial reference system (radians),

Ω_0 longitude of ascending node of satellite orbit plane at t_{oe} (radians),

$\dot{\Omega}$ rate of right ascension angle (radians/s),

$\dot{\Omega}_e$ WGS value of earth rotation rate (radians/s),

t_{oe} Time of ephemeris (s),

$\dot{\Omega}_{ref}$ reference rate of right ascension angle (radians/s),

$\Delta\dot{\Omega}$ reference rate of right ascension difference (radians/s/s).

Description of coordinate reference systems used throughout this appendix

There are five reference frames of interest that will be described in sequence:

1)-"INERT" inertial coordinate system, aligned with vernal point.

This is the usual celestial reference frame, with x aligned with vernal point on ecliptic, z aligned with earth polar axis and y chosen to form a "direct" Cartesian reference system between x, y, z.

All Newton's laws apply in this reference frame. Satellite orbit, depending on gravitational forces, is natively computed in this reference frame. This is the "absolute" reference frame, against which all others are defined.

3)-Rotating Reference "ECEF".

This is the usual Earth-Centered-Earth-Fixed reference system "attached" to the Earth and that rotates with it. ECEF is continuously rotating vs. an inertial reference frame. The position of a static receiver does not change in this reference system.

4)-Inertial Reference "ECEF(t_s)".

It is defined as an Inertial Reference frame that coincides with ECEF at time t_s, or transmit time at satellite. It is obviously fixed vs. the INERT frame, with an arbitrary rotation angle, that depends on t_s.

The conversion formula between ECEF, the rotating frame, and ECEF(t_s), the fixed frame at time t, close to t_s, is:

$$X_{ECEF(t_s)}(t) = R_{ECEF(t_s),ECEF}\big(\alpha(t)\big).X_{ECEF}(t) \tag{A2.11}$$

$$\dot{X}_{ECEF(t_s)}(t) = R_{ECEF(t_s),ECEF}\big(\alpha(t)\big).\dot{X}_{ECEF}(t) + \dot{R}_{ECEF(t_s),ECEF}\big(\alpha(t)\big).X_{ECEF}(t) \tag{A2.12}$$

$$\begin{aligned}\ddot{X}_{ECEF(t_s)}(t) = {} & \ddot{R}_{ECEF(t_s),ECEF}\big(\alpha(t)\big).X_{ECEF}(t) \\ & + \dot{R}_{ECEF(t_s),ECEF}\big(\alpha(t)\big).\dot{X}_{ECEF}(t) \\ & + R_{ECEF(t_s),ECEF}\big(\alpha(t)\big).\ddot{X}_{ECEF}(t)\end{aligned} \tag{A2.13}$$

with:

$$\alpha(t) = \big(\dot{\Omega} - \dot{\Omega}_e\big).(t - t_s), \dot{\alpha}(t) = \big(\dot{\Omega} - \dot{\Omega}_e\big), \ddot{\alpha}(t) = 0. \tag{A2.14}$$

in particular, when t=t_s:

$$\alpha(t_s) = 0, \dot{\alpha}(t_s) = \big(\dot{\Omega} - \dot{\Omega}_e\big), \ddot{\alpha}(t_s) = 0. \tag{A2.15}$$

$$R_{ECEF(t_s),ECEF}\big(\alpha(t_s)\big) = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}, \tag{A2.16}$$

$$\dot{R}_{ECEF(t_s),ECEF}\big(\alpha(t_s)\big) = \begin{bmatrix}0 & \big(\dot{\Omega} - \dot{\Omega}_e\big) & 0\\ -\big(\dot{\Omega} - \dot{\Omega}_e\big) & 0 & 0\\ 0 & 0 & 0\end{bmatrix}, \tag{A2.17}$$

$$\ddot{R}_{ECEF(t_s),ECEF}\left(\alpha(t_s)\right) = \begin{bmatrix} -\left(\dot{\Omega}-\dot{\Omega}_e\right)^2 & 0 & 0 \\ 0 & -\left(\dot{\Omega}-\dot{\Omega}_e\right)^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \quad \text{(A2.18)}$$

5)-Inertial Reference ECEF(t_r).

It is defined as an Inertial Reference frame that coincides with ECEF at time t_r, or receive time at receiver. It is obviously fixed vs. the INERT frame, with an arbitrary rotation angle, that depends on t_r. Conversion formulas similar to the case "ECEF to ECEF(ts)" can be derived for this case "ECEF to ECEF(tr)".

Geometric Distance and its First Derivative vs. Receive Time

As a starting point to introduce the second set of formulas, this classical form first will be derived.

Geometric Distance vs. Receive Time

The most general algorithm to compute the nominal transmit time t_s geometric range and geometric range rate vs. t_r when only the receive time t_r is known is as follows. It takes into account receiver and satellite velocity:

1)-Converts known receiver position and velocity from ECEF to ECEF(t_r).

2)-Initialize first iteration of unknown transmit time $t_{s,0}$ to t_r. This temporarily assumes that the propagation time from satellite to receiver is zero.

3)-Compute satellite position and velocity at $t_{s,k}$, using the ICD GPS 200 formulas and derived. There are natively expressed in rotating ECEF coordinate system at time $t_{s,k}$.

4)- Converts satellite Position and Velocity at $t_{s,k}$ from ECEF to ECEF($t_{s,k}$). The satellite state is now in one inertial reference system.

5)-Converts satellite position and velocity at $t_{s,k}$ from ECEF($t_{s,k}$) to ECEF(t_r). Satellite state at transmit time and receiver state at receive time in the same inertial ECEF(t_r) coordinate system are therefore known. In this common and inertial coordinate system, the velocity of light propagation is c.

6)-A new iteration of the propagation time (t_r-t_s) is computed along with Geometric range $\rho_{Receiver}^{Satellite}$ as:

$$\rho_R^S = \sqrt{\left(X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_{s,k})\right)^2} \qquad \text{(A2.19)}$$

The new iteration of the transmit time $t_{s,k+1}$ can be computed by:

$$t_{s,k+1} = t_r - \frac{\rho_R^S}{c} \qquad \text{(A2.20)}$$

7)-Iterate from step 3 until the flight time or the range change goes under a threshold (range difference from iteration to iteration less than 10^{-4}m in this case) i.e. $|t_{s,k+1} - t_{s,k}| \leq 10^{-4}\ m$.

In practice, only a simplified version of it is implemented. Steps 1 and 4 are trivial if only position and geometric range are of interest.

Geometric Distance Partial Derivative vs. Receive Time

The previous section provided a solution to compute a numeric value of the geometric range rate $\frac{\boldsymbol{\delta\rho(t_r)}}{\boldsymbol{\delta t_r}}$ at t_r. An analytic formula of the same quantity is also derived in this section, assuming known only t_r and ρ_R^S.

In this algorithm t_r is the independent variable and t_s can be expressed in terms of t_r as:

$$t_s = t_r - \frac{\rho_R^S}{c} \tag{A2.21}$$

where

t_s is the send time from Satellite,

t_r is the receive time at Receiver,

c is velocity of light,

ρ_R^S is the geometric distance between Satellite and Receiver.

In the fixed reference frame ECEF(t_r) that coincides with ECEF at time t_r, the geometric range is:

$$[\rho_R^S(t_r)]^2 = \left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right]^2 \tag{A2.22}$$

After total differentiation ("dot" symbol represent differentiation vs. t_r in this section):

$$2.\rho_R^S(t_r).d\rho = \tag{A2.23}$$
$$2.\left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right].\left[\dot{X}_{R,ECEF(t_r)}(t_r) - \dot{X}_{S,ECEF(t_r)}(t_s)\right].dt_r$$

$$2.\rho_R^S(t_r).d\rho = 2.\left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right].\left[\dot{X}_{R,ECEF(t_r)}(t_r) - \frac{\partial X_{S,ECEF(t_r)}(t_s)}{\partial t_s}.\left(1 - \frac{\dot{\rho}_R^S(t_r)}{c}\right)\right].dt_r \tag{A2.24}$$

After reduction:

$$\dot{\rho}_R^S(t_r) = \frac{\partial \rho_R^S(t_r)}{\partial t_r} = \tag{A2.25}$$
$$\frac{\left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right].\left[\dot{X}_{R,ECEF(t_r)}(t_r) - \frac{\partial X_{S,ECEF(t_r)}(t_s)}{\partial t_s}\right]}{\rho_R^S(t_r) - \left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right].\frac{\partial X_{S,ECEF(t_r)}(t_s)}{\partial t_s}.\frac{1}{c}}$$

Geometric Distance and its First Derivative vs. Transmit Time

Geometric Distance vs. Transmit Time

Now introduced is the procedure to compute the nominal receive time t_r, geometric range and geometric range rate vs. t_s when only the transmit time t_s is known. It is essentially a similar iterative computation, but progressively adjusting the receive time until the variation in geometric distance from previous iteration falls below a threshold .

1)- Compute satellite position and velocity at t_s, using the ICD 200 formulas and derived. There are natively expressed in rotating ECEF coordinate system at time t_s.

2)- Converts satellite position and velocity at t_s from ECEF to ECEF(t_s).

3)-Initialize first iteration of unknown receive time $t_{r,0}$ to t_s. This assumes that the propagation time from satellite to receiver is zero.

4)-Convert receiver position and velocity from ECEF to ECEF(t_s). Satellite state at transmit time and receiver state at receive time are now in the same inertial ECEF(t_s) coordinate system. In this common and inertial coordinate system, the velocity of light propagation is c.

5)-A new iteration of the propagation time (t_r-t_s) is computed along with Geometric range $\rho_{Receiver}^{Satellite}$ as:

$$\rho_R^S = \sqrt{\left(X_{R,ECEF(t_s)}(t_{r,k}) - X_{S,ECEF(s)}(t_s)\right)^2} \tag{A2.26}$$

The new approximation $t_{r,k+1}$ of the receive time at this iteration is given by:

$$t_{r,k+1} = \frac{\rho_R^S}{c} + t_s \tag{A2.27}$$

6)-Iterate from Step 4 until the flight time or the range change goes under a threshold (range difference from iteration to iteration less than 10^{-4} m in this case). i.e. $\left|t_{r,k+1} - t_{r,k}\right| \leq 10^{-4}\ m$.

In practice, only a simplified version of it is implemented. Steps 2 and 5 are trivial and omitted if only position and geometric range are of interest.

Geometric Distance Partial Derivative vs. Transmit Time

The previous section provided a solution to compute a numeric value of the geometric range rate $\frac{\boldsymbol{\delta\rho(t_s)}}{\boldsymbol{\delta t_s}}$ at t_s. An analytic formula of the same quantity, assuming known only t_s and ρ_R^S, is also derived in this section.

In this algorithm, t_s is the independent variable, and t_r can be expressed in terms of t_s as:

$$t_r = t_s + \frac{\rho_R^S}{c} \tag{A2.28}$$

where

t_s is the send time from Satellite,

t_r is the receive time at Receiver,

c is velocity of light,

ρ_R^S is the geometric distance between Satellite and Receiver.

In the fixed reference frame ECEF(t_r) that coincides with ECEF at time t_r, the geometric range is:

$$[\rho_R^S(t_r)]^2 = \left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right]^2 \quad \text{(A2.29)}$$

After total differentiation ("dot" symbol represent differentiation vs. t_s in this section):

$$2.\rho_R^S(t_s).d\rho = 2.\left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right].\left[\frac{\partial X_{R,ECEF(t_r)}(t_r)}{\partial t_r}.\left(1 + \frac{\dot{\rho}_R^S(t_s)}{c}\right) - \dot{X}_{S,ECEF(t_r)}(t_s)\right].dt_s \quad \text{(A2.30)}$$

After reduction:

$$\dot{\rho}_R^S(t_s) = \frac{\partial\, \rho_R^S(t_s)}{\partial\, t_s} \quad \text{(A2.31)}$$

$$= \frac{\left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right].\left[\frac{\partial X_{R,ECEF(t_r)}(t_r)}{\partial t_r} - \dot{X}_{S,ECEF(t_r)}(t_s)\right]}{\rho_R^S(t_s) - \left[X_{R,ECEF(t_r)}(t_r) - X_{S,ECEF(t_r)}(t_s)\right].\frac{\partial X_{R,ECEF(t_r)}(t_r)}{\partial t_r}.\frac{1}{c}}$$

It is to note that this formula requires an expression of the velocity of the satellite in inertial ECEF(t_r) coordinate system. The conversion formulas between ECEF and ECEF(t_r) have been already introduced earlier in this appendix; the satellite velocity in ECEF at time t_s is still required. The ICD GPS 200 provides analytic formulas only for the position of the satellite in ECEF coordinate system. An extension of these formula for the satellite velocity in the same ECEF reference system is necessary. These extensions are introduced in the Appendix 3.

APPENDIX 3:

Closed Form Formulas for Satellite Velocity and Acceleration in ECEF Coordinates

This appendix extends the satellite position formulas in ECEF coordinate system found in [ICD-GPS-200D], Table 30 I and Table 30-II, to the satellite velocities in the same ECEF coordinate system. Only the navigation broadcast message parameters directly used in the velocity formulas definitions will be recalled below in Table 27.

Table 27 - Orbit Input Parameters

Input Parameters	
Symbol	Description
t_{oe}	Ephemeris data reference time of week
M_0	Mean Anomaly at reference time
n	Corrected mean motion (note 1)
e	Eccentricity
ω	Argument of perigee

Table 27 - Orbit Input Parameters (Continued)

Symbol	Description
C_{us}	Amplitude of the sine harmonic correction term to the argument of latitude
C_{uc}	Amplitude of the cosine harmonic correction term to the argument of latitude
C_{rs}	Amplitude of the sine harmonic correction term to the orbit radius
C_{rc}	Amplitude of the cosine harmonic correction term to the orbit radius
C_{is}	Amplitude of the sine harmonic correction term to the angle of inclination
C_{ic}	Amplitude of the cosine harmonic correction term to the angle of inclination
A_k	Semi major axis (note 1)
i_0	Inclination angle at reference time
$\dot{\imath_0}$	Rate of inclination angle
$\dot{\Omega}_e$	WGS 84 value of the earth's rotation rate (note 2)

Note 1:this is not a parameter from navigation broadcast message see [ICD-GPS-200D] Table 30 for the computation from navigation broadcast parameters

Note 2: this is a physical constant

In Table 28, the original formula will be recalled in the value column, and the corresponding first time derivative will be expressed in the "first derivative" column

Table 28 - Satellite Position Velocity and Acceleration Formula

Parameter	Value	First derivative
Time from ephemeris reference time	$t_k = t - t_{oe}$	$\dot{t}_k = 1$
Mean Anomaly	$M_k = M_0 + nt_k$	$\dot{M}_k = n$
Eccentric Anomaly	$M_k = E_k - e.\sin E_k$	$\dot{E}_k = \frac{\dot{M}_k}{(1 - e\cos E_k)}$
True Anomaly	$\nu_k = \tan^{-1}\left(\frac{\sqrt{1-e^2}\sin E_k}{\cos E_k - e}\right)$	$\dot{\nu}_k = \frac{\sin E_k \;\; \dot{E}_k(1 + e\cos \nu_k)}{\sin \nu_k \,(1 - e\cos E_k)}$
Argument of latitude	$\phi_k = \nu_k + \omega$	$\dot{\phi}_k = \dot{\nu}_k$
Argument of latitude correction	$\delta u_k = C_{us}\sin(2\phi_k) + C_{uc}\cos(2\phi_k)$	$\delta \dot{u}_k = 2C_{us}\cos(2\phi_k)\;\dot{\phi}_k - 2C_{uc}\sin(2\phi_k)\,\dot{\phi}_k$
Radial Correction	$\delta r_k = C_{rs}\sin(2\phi_k) + C_{rc}\cos(2\phi_k)$	$\delta \dot{r}_k = 2[C_{rs}\cos(2\phi_k) - C_{rc}\sin(2\phi_k)]\dot{\nu}_k$
Inclination correction	$\delta i_k = C_{is}\sin(2\phi_k) + C_{ic}\cos(2\phi_k)$	$\delta \dot{\imath}_k = 2[C_{is}\cos(2\phi_k) - C_{ic}\sin(2\phi_k)]\dot{\nu}_k$
Corrected Argument of Latitude	$u_k = \phi_k + \delta u_k$	$\dot{u}_k = \dot{\phi}_k + \delta \dot{u}_k$
Corrected Radius	$r_k = A_k(1 - e\cos E_k) + \delta r_k$	$\dot{r}_k = \frac{Ae\sin E_k\, n}{(1 - e\cos E_k)} + \delta \dot{r}_k$
Corrected Inclination	$i_k = i_0 + \dot{\imath}_0 t_k + \delta i_k$ (see Note 1)	$\dot{\imath}_k = \dot{\imath}_0 + \delta \dot{\imath}_k$ (see Notes 1 and 2)
Rate of right ascension	$\dot{\Omega} = \dot{\Omega}_{REF} + \Delta\dot{\Omega}$	

Table 28 - Satellite Position Velocity and Acceleration Formula (Continued)

Parameter	Value	First derivative
Corrected Longitude of Ascending Node	$\Omega_k = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e)t_k - \dot{\Omega}_e t_{oe}$	$\dot{\Omega}_k = \dot{\Omega} - \dot{\Omega}_e$
Satellite x-position in orbit plane	$x_k{}' = r_k \cos u_k$	$\dot{x}_k{}' = \dot{r}_k \cos u_k - y_k{}' \dot{u}_k$
Satellite y-position in orbit plane	$y_k{}' = r_k \sin u_k$	$\dot{y}_k{}' = \dot{r}_k \sin u_k + x_k{}' \dot{u}_k$
ECEF x-coordinate	$x_k = x_k{}' \cos \Omega_k - y_k{}' \cos i_k \sin \Omega_k$	$\dot{x}_k = (\dot{x}_k{}' - y_k{}' \cos i_k \dot{\Omega}_k) \cos \Omega_k + (y_k{}' \dot{\imath}_k \sin i_k - x_k{}' \dot{\Omega}_k - \dot{y}_k{}' \cos i_k) \sin \Omega_k$ (Note 2)
ECEF y-coordinate	$y_k = x_k{}' \sin \Omega_k + y_k{}' \cos i_k \cos \Omega_k$	$\dot{y}_k = (\dot{x}_k{}' - y_k{}' \cos i_k \dot{\Omega}_k) \sin \Omega_k + (x_k{}' \dot{\Omega}_k + \dot{y}_k{}' \cos i_k - y_k{}' \dot{\imath}_k \sin i_k) \cos \Omega_k$ (Note 2)
ECEF z-coordinate	$z_k = y_k{}' \sin i_k$	$\dot{z}_k = \dot{y}_k{}' \sin i_k + y_k{}' \dot{\imath}_k \cos i_k$ (Note 2)

Note 1: $\dot{\imath}_0$ is the first derivative of i_0, Note 2: $\dot{\imath}_k$ is the first derivative of i_k

Mathematically equivalent formulas can be found in the paper [Korvenoja2000]. It is important to note that the symbols and the formulas found in the paper have a different form than in this appendix, so that the formulas cannot be compared one to one. The appendix refers to Table 30-I and Table 30-II of the [ICD=GPS-200D] dated December 7th 2004. The paper publication predates this ICD publication date. All paper notations refer to an earlier version of the same ICD-GPS-200, probably the version C.

REFERENCES

Aguado, L.E. ,C. O'Driscoll, P. Xia, K. Nurutdinov, C. Hill, P. O'Beirne (2007) "A Low-Cost, Low Power Galileo/GPS Positioning System for Monitoring Landslides", http://www.ggphi.eu/monitoring_landslides.pdf last accessed December 2009.

Ashkenazi, V. , A.H. Dodson, T. Moore and G. W. Roberts (1997) "Monitoring the movements of bridges by GPS," ION GPS 1997.

Biswas, P., and Y. Ye (2004) " Semidefinite Programming for Ad Hoc Wireless Sensor Network Localization" in IPSN'04, April 26–27, 2004.

Boser, B. (2004) "Autonomous Sensors", http://www.eecs.berkeley.edu/~boser/publications last accessed December 20th 2008.

Boukerche, A., H.A.B.F. Oliveira, E.F. Nakamura, and A.A.F. Loureiro (2007) "Localization Systems for wireless sensor networks", IEEE Wireless Communications Magazine, 45 (9) (2007) 82-90.

Chaitin-Chatelin, F., S. Dallakyan, and V. Frayssé (2000) "GPS Carrier Phase Ambiguity Resolution with the LAMBDA method,1. A stability Analysis, 2. An exponential speed-up", CERFACS Final report August 15, 2000.http://www.cerfacs.fr/algor/reports/2000/CR_PA_00_52.ps.gz last accessed October 2009.

Chang, X-W, and T. Zhou (2006) "MILES: MATLAB package for solving Mixed Integer LEast Square problems-Theory and Algorithms", Scientific Computing Laboratory, School of Computer Science McGill University, October 2006, http://www.cs.mcgill.ca/~chang/pub/ChaZ07.pdf last accessed October 2009.

Cosser, E. , G. Roberts, X. Meng and A. Dodson (2003) "the comparison of single frequency and dual frequency GPS for bridge deflection and vibration monitoring, " Proceedings, 11th FIG Symposium on Deformation Measurements, Santorini, Greece, 2003.

Cosser, E. (2004) "Bridge deflection monitoring and frequency identification with single frequency GPS receivers," ION GNSS 17th International Technical Meeting of the Satellite Division, 21-24 Sept 2004 Long Beach, CA.

CrossBow Wireless Technology Overview (2008) http://www.xbow.com/Technology/Overview.aspx last accessed December 20th 2008.

Curren, D. (2005) "A Survey of Simulation in Sensor Networks", project report (CS580) University of Binghamton, 2005.

de Jonge, P., and C. Tiberius (1995) "Integer ambiguity estimation with the lambda method," in Proceedings IAG Symposium No. 115, GPS Trends in Precise Terrestrial, Airborne and Spaceborne Applications, XXI General Assembly of IUGG, July 2-14, Boulder, CO, July 2-14, 1995, G. Beutler et.al., (Eds.), Springer Verlag, pp. 280-284.

de Jonge, P., and C. Tiberius (1996a) "The LAMBDA method for integer ambiguity estimation: implementation aspects", LGR-series, TU delft, August 1996, http://enterprise.lr.tudelft.nl/publications/files/lgr12.pdf last accessed October 2009.

de Jonge, P. , C. Tiberius, and P. Teunissen (1996b) "Computational Aspects of the LAMBDA Method for GPS Ambiguity Resolution", TU Delft 1996, http://enterprise.lr.tudelft.nl/publications/files/ioncp.pdf last accessed October 2009.

Doherty, L., K. Pister, and L. El Ghaoui (2001) " Convex Position Estimation in Wireless Sensor Networks", IEEE INFOCOM 2001.

Elmenreich, W., M. Rosenblatt, and A. Wolf (2007) "Fixed Point Library Based on ISO/IEC Standard DTR 18037 for Atmel AVR Microcontrollers", Institut für Technishe Informatik, Vienna University of Technology, Vienna, Austria, 2007.

Elson J., L. Girod, and D. Estrin (2003) "Fine-grained network time synchronization using reference broadcasts," USENIX Association, 5th Symposium on Operating Systems Design and Implementation, 2003.

Fu, Y., H. Liu, and T. Xing (2006) "The Localization of Wireless Sensor Network Nodes Based on DSSS" Electro/ information Technology, 2006 IEEE International Conference , 7-10 May 2006, pp 465-469.

Ganeriwal, S., R. Kumar, and M.B. Srivastava (2003) "Timing-sync protocol for sensor networks," Sensys'03, November 5-7, 2003, Los Angeles, California, USA.

Gurtner,W. (2001) "RINEX: The Receiver Independent Exchange Format Version 2.10"Astronomical Institute, University of Berne. www.ngs.noaa.gov/CORS/Rinex2.html last accessed January 2010.

Handy, M.J., M. Haase, and D. Timmermann (2002) "Low Energy Adaptive Clustering Hierarchy with Deterministic Cluster-Head Selection", Institute of Applied Microelectronics and Computer Science, University of Rostock, Germany, Fourth IEEE Conference on Mobile and Wireless Communications Networks, Stockholm, September 2002.

Hyodo, N., H. Murao, and T. Saito (2005) "Matrix Multiplication Made Fast-Practical View of Fast Matrix Operation for Computer Algebra System", AlphaOmega Inc.,

The Univ. of Electro-Communications, Japan Society for Symbolic and Algebraic Computation, 2005.

NAVSTAR (2004) Global Positioning System-Interface Specification-IS-GPS-200-Revision D-Dec 07, 2004. http://www.navcen.uscg.gov/gps/geninfo/IS-GPS-200D.pdf last accessed September 2009.

NOVATEL (2010) "OEM 4 Family-User Manual Volume 2-Command and Log Reference" http://www.novatel.com/Documents/Manuals/om-20000047_arc.pdf last accessed January 2010.

Ilyas, M., and I. Mahgoub (2005) "Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems" 2005, CRC Press.

Islam, A., J.M.P. Langlois, and A. Noureldin (2009)"A design methodology for the implementation of embedded vehicle navigation systems", IEEE International Conference on Electro/Information Technology, 7-9 June 2009.

Kakaradov, B. (2004) "Ultra-Fast Matrix Multiplication: An Empirical Analysis of Highly Optimized Vector Algorithms", Computer Science Magazine Spring 2004.

Kasdin, N.J. (1995) "Discrete simulation of colored noise and stochastic processes and $1/f^{\alpha}$ power law noise generation", W.W. Hansen Exp. Phys. Lab., Stanford Univ., CA, Proceedings of the IEEE Publication Date: May 1995 Volume: 83, Issue: 5.

Korvenoja, P., and R. Piché (2000) "Efficient Satellite Orbit Approximation", ION GPS 2000, September 19-22, 2000 Salt Lake City, UT.

Krishnamachari, B. (2005) "Networking Wireless Sensors", Cambridge University Press, 2005.

Leick, A. (2004) "GPS Satellite Surveying-Third Edition", John Wiley & Sons, Inc., 2004, ISBN 0-471-05930-7.

Levis, P. , N. Lee, M. Welsch, and D. Culler (2003) “TOSSIM: Accurate and Scalable Simulation of Entire TiniOS Applications”, in Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys), 2003.

Liu, J. (2003) "Implementation and Analysis of GPS Ambiguity Resolution Strategies in Single and Multiple Reference Station Scenarios". MSc Thesis 2003, University of Calgary, Department of Geomatics Engineering.

Luo, N. (2000) “Centimetre-level relative positioning of multiple moving platforms using ambiguity constraints,” ION GPS 2000 Conference, September 19-22, 2000 Salt Lake City, UT.

Luo, N. (2001) "Precise Relative Positioning of Multiple Moving Platforms Using GPS Carrier Phase Observables", University of Calgary, Department of Geomatics Engineering, UGCE Reports, Number 20147.

Luo, N., and G. Lachapelle (2003) “Relative positioning of multiple moving platforms using GPS,” in IEEE transactions on Aerospace and Electronic Systems, Vol. 39, N° 3 July 2003.

LaMarca,A., J. Hightower, I. Smith, and S. Consolvo (2005) "Self-Mapping in 802.11 Location Systems" , Intel Research Seattle, LECTURE NOTES IN COMPUTER SCIENCE, 2005 - Springer, ISSN 0302-9743.

Maróti, M., B. Kusý, and G.Balogh (2005) “Radio Interferometric GeoLocation” , SenSys’ 05, November 02-04, 2005, San Diego, California, USA.

Maróti, M., B. Kusy, G. Simon, and A. Ledeczi (2004) "the Flooding time synchronization protocol," SenSys'04, November 3-5, 2004, Baltimore, Maryland, USA.

Moeglein, M., N. Krasner, (1998) "An Introduction to SnapTrack Server-Aided GPS Technology" Proceedings of the 11th International Technical Meeting of the Satellite Division of the Institute of Navigation ION GPS 1998, September 15 - 18, 1998 Nashville, TN.

Niculescu, D., and B. Nath (2003) "Ad Hoc Positioning Sysem (APS) Using AoA" Proc INFOCOM 2003, San Francisco, CA, 2003

Patwari, N., A.O. Hero, M. Perkins, N.S. Correal, and R.J. Odea (2003) "Relative location estimation in wireless sensor networks", IEEE Transactions on Signal Processing, 51 (8) (2003) 2137-2148.

Perrone, L.F., and D. Nicol (2002) “ A Scalable Simulator for TinyOS Applications” Proceedings of the 2002 Winter Simulation Conference.

Priyantha,N.B., A. Chakraborty, and H. Balakrishnan (2000) " The cricket location-support system" in Proc. of the Sixth Annual ACM International Conference on Mobile Computing and Networking (MOBICOM), August 2000.

Ramadurai, V. , and M. L. Sichitiu (2003) "Localization in WIrelss Sensor Networks: A Probabilistic Approach" *Proc.ICWN 2003*, Las Vegas, NV, June 2003, pp. 275–81.

Savaranos, Y (2006a) "Energy-Aware Synchronization in Wireless Sensor Networks", MS Thesis, 2006, University of North Texas, http://www.cse.unt.edu/~rakl/Yanos06.pdf last accessed December 2008.

Savaranos, Y (2006b) "Energy-Aware Synchronization in Wireless Sensor Networks", MS Thesis presentation , 2006, University of North Texas, http://www.cse.unt.edu/~rakl/Yanos06.pdf last accessed December 2008.

Savvides, A. , C-C. Han, and M.B. Strivastava (2001) "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors", ACM SIGMOBILE Rome, Italy July 2001.

Shang, Y.,W. Ruml, Y. Zhang, and M. Fromherz (2003) " Localization from Mere Connectivity", MobiHoc'03, June 1–3, 2003, Annapolis, Maryland, USA.

Shang, Y., W. Ruml, and M.P.J. Fromherz (2004) "Positioning Using Local Maps", Ad-Hoc Networks, 4 (2004), 240-253.

Simic, S. N., and S. Sastry (2001) "Distributed Localization in Wireless Ad Hoc Networks" UC Berkeley, Tech. rep. UCB/ERL M02/26, 2002.

Simon, G., P. Vólgyesi, M. Maróti, and Á. Lédeczi (2002) "Simulation-based optimization of communication protocols for large-scale wireless sensor networks", Vanderbilt University, IEEEAC, October 10th, 2002.

Sirola, N. (2001) "A Method for GPS Positioning without Current Navigation Data" Master of Science Thesis, Tampere University of Technology, Departement of Electrical Engineering, June 6th ,2001.

Sridharan, A., M. Zuniga, and B. Krishnamachari (2004) "Integrating Environment Simulators with Network Simulators", University of Southern California, Los Angeles, 2004.

Sun, H., , M.E. Cannon, and T. Melgard (1999) " Real-time GPS reference Network carrier phase ambiguity resolution," in Proceedings of the ION 1999 National Technical Meeting, San Diego, January 25-27, pp193-199.

SUNSPOT (2008) http://www.sunspotworld.com/ last accessed December 20th 2008.

Syrjärinne,J. (2000) "Time recovery through fusion of inaccurate network timing assistance with GPS measurements" Proceedings of the 3rd International Conference on Information Fusion, Vol II, pp WeD5-3-WeD5-10.

Tapia, R.,C. Lanius, C. McZeal, and T. Parks (2001) "Computational Science: Tools for a Changing World-A High School Curriculum", Rice University, 2001, http://ceee.rice.edu/Books/CS last accessed October 2009.

Teunissen, P. J. G. (1993) "Least-squares estimation of the integer GPS ambiguities", Invited Lecture, Section IV Theory and Methodology, IAG General Meeting, Beijing, China, August 1993.

Teunissen, P. (2000) "Probabilistic aspects of GNSS integer ambiguity estimation," *Earth Planet Space*, 52, 801-805, 2000.

TinyOS (2008) Operating System for motes and WSN, http://www.tinyos.net last accessed December 2008.

Whitehouse,K. , and D. Culler (2002) "Calibration as a Parameter Estimation in Sensor Networks" WSNA '02: Proc 1st ACM Int'l Wksp Wireless Networks and Apps., ACM Press, 2002, pp59-67.

Wikipedia (2010) "The Speed of Sound", http://en.wikipedia.org/wiki/Speed_of_sound last accessed January 2010.

ZigBee (2008) "ZigBee Alliance" Official Website, http://www.zigbee.org/en/spec_download/zigbee_downloads.asp last accessed December 20th 2008.