**ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ**

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ**
**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

**ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ**
**"ΕΠΕΞΕΡΓΑΣΙΑ ΣΗΜΑΤΟΣ ΓΙΑ ΕΠΙΚΟΙΝΩΝΙΕΣ ΚΑΙ ΠΟΛΥΜΕΣΑ"**

**ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ**

# Ανάλυση Κυρίων Συνιστωσών για την Αποτελεσματική Μετάδοση Πληροφορίας σε Ασύρματα Δίκτυα Αισθητήρων

**Αγγελική Π. Σουλέ**

**Επιβλέποντες:** **Ευστάθιος Χαντζηευθυμιάδης,** Επίκουρος Καθηγητής
**Αναγνωστόπουλος Χρήστος**, PhD

**ΑΘΗΝΑ**

**ΑΥΓΟΥΣΤΟΣ 2011**

# ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάλυση Κυρίων Συνιστωσών για την Αποτελεσματική Μετάδοση Πληροφορίας σε Ασύρματα Δίκτυα Αισθητήρων

**Αγγελική Π. Σουλέ**

**Α.Μ.:** Μ1032

**ΕΠΙΒΛΕΠΟΝΤΕΣ:** **Ευστάθιος Χαντζηευθυμιάδης,** Επίκουρος Καθηγητής
**Αναγνωστόπουλος Χρήστος,** PhD

Αύγουστος 2011

# ΠΕΡΙΛΗΨΗ

Οι εφαρμογές που βασίζονται σε Ασύρματα Δίκτυα Αισθητήρων (Wireless Sensor Networks, WSN) επηρεάζονται από πολλούς παράγοντες, όπως σφάλματα μετάδοσης, τοπολογία του δικτύου και την κατανάλωση ενέργειας. Κατά συνέπεια, η ανάπτυξη τέτοιων εφαρμογών εισάγει διάφορες ερευνητικές προκλήσεις. Στη διπλωματική εργασία προτείνεται ένα νέο σχήμα συμπίεσης πληροφορίας πλαισίου με τη βοήθεια των μαθηματικών τεχνικών της Ανάλυσης Κύριων Συνιστωσών (Principal Component Analysis). Το σχήμα αυτό επιτυγχάνει υψηλή συμπίεση σε συσχετισμένα δεδομένα (μετρήσεις θερμοκρασίας και υγρασίας που έχουν ληφθεί σε ανοικτούς χώρους), χωρίς παράλληλα να παρατηρείται σημαντική αύξηση στο σφάλμα.



**ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ**: Ασύρματο Δίκτυο Αισθητήρων, Συμπίεση Πλαισίου, Ενεργειακό Κέρδος σε Ασύρματες επικοινωνίες.

**ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ**: Συμπίεση, Ανάλυση Κύριων Συνιστωσών, Ενεργειακό Κέρδος, Ασύρματο Δίκτυο Αισθητήρων.

# ABSTRACT

Applications based on Wireless Sensor Networks (WSN) are influenced by many factors such as transmission errors, network topology and power consumption. Consequently, developing such applications introduces several research challenges. The thesis proposes a new compression format information framework with using of mathematical techniques of Principal Component Analysis (Principal Component Analysis). The scheme achieves high compression associated data (temperature and humidity measurements taken outdoors), while no significant increase in error.

**SUBJECT AREA**: Wireless Sensor Network, Context Compression, Energy Efficient Wireless Communications.

**KEYWORDS**: Compression, Principal Components Analysis, Wireless Sensor Networks, Energy Efficiency,

## AKNOWLEDGEMENTS

I would like to thank Mr. Efstathios Hadjiefthymiades for giving me the chance to work on this subject, as well as Christos Anagnostopoulos for all the time he devoted, his useful suggestions and instructions, which were necessary for the completion of this paper.

I would also like to thank with all my heart my parents Panagiotis and Rita, for their support and for giving me strength during my education studies and my sister Irene for her understanding on the difficult moments.

Finally, I would like to thank my friends who listened patiently all this time and put their own bit to finish this thesis, Marianna and Lefteris.

# CONTENTS

## LIST OF FIGURES

# LIST OF TABLES

# PREFACE

This thesis was carried out under the Post-Graduate Program of Studies of the direction of Signal Processing for Communications and Multimedia, department of Informatics and Telecommunications, Faculty of Sciences of the University of Athens, during the Academic year 2010-2011. The first chapter describes some basic characteristics of the Wireless Sensor Network and then analyses the Principal Component Analysis' stages. Subsequently, the issue of energy consumption of the sensors is being analysed. In the fourth chapter we present the information compression system we propose. Apart from the necessary theoretical background, in order to successfully complete the paper it was required to execute code as a simulation on the computer. The results of the simulations and the interesting conclusions constitute the last three chapters of the thesis.

Athens 2011,
Angeliki Soule.

# CHAPTER 1

# 1. Introduction to Wireless Sensor Networks

## 1.1 Introduction

Recent advances in micro-electro-mechanical systems (MEMS) technology, wireless communications, and digital electronics have enabled the development of low-cost, low-power, multifunctional sensor nodes that are small in size and communicate untethered in short distances. These tiny sensor nodes, which consist of sensing, data processing, and communicating components, leverage the idea of sensor networks based on collaborative effort of a large number of nodes.

Sensor networks represent a significant improvement over traditional sensors, which are deployed in the following two ways [13]:

- Sensors can be positioned far from the actual phenomenon, i.e., something known by sense perception. In this approach, large sensors that use some complex techniques to distinguish the targets from environmental noise, are required.
- Several sensors that perform only sensing can be deployed. The positions of the sensors and communications topology are carefully engineered. They transmit time series of the sensed phenomenon to the central nodes where computations are performed and data are fused. A sensor network is composed of a large number of sensor nodes, which are densely deployed either inside the phenomenon or very close to it.

A sensor network is composed of a large number of sensor nodes, which are densely deployed either inside the phenomenon or very close to it. The position of sensor nodes need not be engineered or pre-determined. This allows random deployment in inaccessible terrains or disaster relief operations. On the other hand, this also means that sensor network protocols and algorithms must possess self-organizing capabilities. Another unique feature of sensor networks is the cooperative effort of sensor nodes. Sensor nodes are fitted with an on-board processor. Instead of sending the raw data to the nodes responsible for the fusion, sensor nodes use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data.

The above described features ensure a wide range of applications for sensor networks. Some of the application areas are health, military, and security. For example, the physiological data about a patient can be monitored remotely by a doctor. While this is more convenient for the patient, it also allows the doctor to better understand the patient's current condition. Sensor networks can also be used to detect foreign chemical agents in the air and the water. They can help to identify the type, concentration, and location of pollutants. In essence, sensor networks will provide the end user with intelligence and a better understanding of the environment. We envision that, in future, wireless sensor networks will be an integral part of our lives, more so than the present-day personal computers.

Realization of these and other sensor network applications require wireless ad hoc networking techniques. Although many protocols and algorithms have been proposed for traditional wireless ad hoc networks, they are not well suited for the unique features and application requirements of sensor networks. To illustrate this point, the differences between sensor networks and ad hoc networks [31] are outlined below:

- The number of sensor nodes in a sensor network can be several orders of magnitude higher than the nodes in an ad hoc network.
- Sensor nodes are densely deployed.
- Sensor nodes are prone to failures.
- The topology of a sensor network changes very frequently.
- Sensor nodes mainly use broadcast communication paradigm whereas most ad hoc networks are based on point-to-point communications.
- Sensor nodes are limited in power, computational capacities, and memory.
- Sensor nodes may not have global identification (ID) because of the large amount of overhead and large number of sensors.

Since large number of sensor nodes are densely deployed, neighbor nodes may be very close to each other. Hence, multihop communication in sensor networks is expected to consume less power than the traditional single hop communication. Furthermore, the transmission power levels can be kept low, which is highly desired in covert operations. Multihop communication can also effectively overcome some of the signal propagation effects experienced in long-distance wireless communication.

One of the most important constraints on sensor nodes is the low power consumption requirement. Sensor nodes carry limited, generally irreplaceable, power sources. Therefore, while traditional networks aim to achieve high quality of service (QoS) provisions, sensor network protocols must focus primarily on power conservation. They must have inbuilt trade-off mechanisms that give the end user the option of prolonging network lifetime at the cost of lower throughput or higher transmission delay.

## 1.2 Sensor networks applications

Sensor networks may consist of many different types of sensors such as seismic, low sampling rate magnetic, thermal, visual, infrared, acoustic and radar, which are able to monitor a wide variety of ambient conditions that include the following [10]:

- temperature,
- humidity,
- wind speed,
- vehicular movement,
- lightning condition,
- pressure,
- soil makeup,
- noise levels,
- the presence or absence of certain kinds of objects,
- mechanical stress levels on attached objects, and

- the current characteristics such as speed, direction, and size of an object.

Sensor nodes can be used for continuous sensing, event detection, event ID, location sensing, and local control of actuators. The concept of microsensing and wireless connection of these nodes promise many new application areas. We categorize the applications into military, environment, health, home and other commercial areas. It is possible to expand this classification with more categories such as space exploration, chemical processing and disaster relief.

### 1.2.1 Military applications

Wireless sensor networks can be an integral part of military command, control, communications, computing, intelligence, surveillance, reconnaissance and targeting (C4ISRT) systems. The rapid deployment, self-organization and fault tolerance characteristics of sensor networks make them a very promising sensing technique for military C4ISRT. Since sensor networks are based on the dense deployment of disposable and low-cost sensor nodes, destruction of some nodes by hostile actions does not affect a military operation as much as the destruction of a traditional sensor, which makes sensor networks concept a better approach for battlefields. Some of the military applications of sensor networks are monitoring friendly forces, equipment and ammunition; battlefield surveillance; reconnaissance of opposing forces and terrain; targeting; battle damage assessment; and nuclear, biological and chemical (NBC) attack detection and reconnaissance.

Monitoring friendly forces, equipment and ammunition: Leaders and commanders can constantly monitor the status of friendly troops, the condition and the availability of the equipment and the ammunition in a battlefield by the use of sensor networks. Every troop, vehicle, equipment and critical ammunition can be attached with small sensors that report the status. These reports are gathered in sink nodes and sent to the troop leaders. The data can also be forwarded to the upper levels of the command hierarchy while being aggregated with the data from other units at each level.

Battlefield surveillance: Critical terrains, approach routes, paths and straits can be rapidly covered with sensor networks and closely watched for the activities of the opposing forces. As the operations evolve and new operational plans are prepared, new sensor networks can be deployed anytime for battlefield surveillance.

Reconnaissance of opposing forces and terrain: Sensor networks can be deployed in critical terrains, and some valuable, detailed, and timely intelligence about the opposing forces and terrain can be gathered within minutes before the opposing forces can intercept them.

Targeting: Sensor networks can be incorporated into guidance systems of the intelligent ammunition.

Battle damage assessment: Just before or after attacks, sensor networks can be deployed in the target area to gather the battle damage assessment data.

Nuclear, biological and chemical attack detection and reconnaissance: In chemical and biological warfare, being close to ground zero is important for timely and accurate detection of the agents. Sensor networks deployed in the friendly region and used as a chemical or biological warning system can provide the friendly forces with critical reaction time, which drops casualties drastically. We can also use sensor networks for detailed reconnaissance after an NBC attack is detected. For instance, we can make a nuclear reconnaissance without exposing a recce team to nuclear radiation.

### 1.2.2 Environmental applications

Some environmental applications of sensor networks include tracking the movements of birds, small animals, and insects; monitoring environmental conditions that affect crops and livestock; irrigation; macroinstruments for large-scale Earth monitoring and planetary exploration; chemical/ biological detection; precision agriculture; biological, Earth, and environmental monitoring in marine, soil, and atmospheric contexts; forest fire detection; meteorological or geophysical research; flood detection; bio-complexity mapping of the environment; and pollution.

Forest fire detection: Since sensor nodes may be strategically, randomly, and densely deployed in a forest, sensor nodes can relay the exact origin of the fire to the end users before the fire is spread uncontrollable. Millions of sensor nodes can be deployed and integrated using radio frequencies/ optical systems. Also, they may be equipped with effective power scavenging methods [6], such as solar cells, because the sensors may be left unattended for months and even years. The sensor nodes will collaborate with each other to perform distributed sensing and overcome obstacles, such as trees and rocks, that block wired sensors' line of sight.

Biocomplexity mapping of the environment [5] : A biocomplexity mapping of the environment requires sophisticated approaches to integrate information across temporal and spatial scales [24]. The advances of technology in the remote sensing and automated data collection have enabled higher spatial, spectral, and temporal resolution at a geometrically declining cost per unit area [8]. Along with these advances, the sensor nodes also have the ability to connect with the Internet, which allows remote users to control, monitor and observe the biocomplexity of the environment. Although satellite and airborne sensors are useful in observing large biodiversity, e.g., spatial complexity of dominant plant species, they are not fine grain enough to observe small size biodiversity, which makes up most of the biodiversity in an ecosystem [14]. As a result, there is a need for ground level deployment of wireless sensor nodes to observe the biocomplexity. One example of biocomplexity mapping of the environment is done at the James Reserve in Southern California. Three monitoring grids with each having 25 – 100 sensor nodes will be implemented for fixed view multimedia and environmental sensor data loggers.

Flood detection [7]: An example of a flood detection is the ALERT system deployed in the US [25]. Several types of sensors deployed in the ALERT system are rainfall, water level and weather sensors. These sensors supply information to the centralized database system in a pre-defined way. Research projects, such as the COUGAR Device Database Project at Cornell University [3] and the DataSpace project at Rutgers [12], are investigating distributed approaches in interacting with sensor nodes in the sensor field to provide snapshot and long-running queries.

Precision Agriculture: Some of the benefits is the ability to monitor the pesticides level in the drinking water, the level of soil erosion, and the level of air pollution in realtime.

### 1.2.3 Health applications

Some of the health applications for sensor networks are providing interfaces for the disabled, integrated patient monitoring, diagnostics, drug administration in hospitals, monitoring the movements and internal processes of insects or other small animals; telemonitoring of human physiological data, and tracking and monitoring doctors and patients inside a hospital [16].

Telemonitoring of human physiological data: The physiological data collected by the sensor networks can be stored for a long period of time, and can be used for medical exploration. The installed sensor networks can also monitor and detect elderly people's behavior, e.g., a fall. These small sensor nodes allow the subject greater freedom of movement and allow doctors to identify pre-defined symptoms earlier. Also, they facilitate a higher quality of life for the subjects compared to the treatment centers [2]. A ''Health Smart Home'' is designed in the Faculty of Medicine in Grenoble—France to validate the feasibility of such system [16].

Tracking and monitoring doctors and patients inside a hospital: Each patient has small and light weight sensor nodes attached to them. Each sensor node has its specific task. For example, one sensor node may be detecting the heart rate while another is detecting the blood pressure. Doctors may also carry a sensor node, which allows other doctors to locate them within the hospital.

Drug administration in hospitals: If sensor nodes can be attached to medications, the chance of getting and prescribing the wrong medication to patients can be minimized. Because, patients will have sensor nodes that identify their allergies and required medications. Computerized systems as described in [23] have shown that they can help minimize adverse drug events.

### 1.2.4 Home applications

Home automation: As technology advances, smart sensor nodes and actuators can be buried in appliances, such as vacuum cleaners, micro-wave ovens, refrigerators, and VCRs [18]. These sensor nodes inside the domestic devices can interact with each other and with the external network via the Internet or Satellite. They allow end users to manage home devices locally and remotely more easily.

Smart environment: The design of smart environment can have two different perspectives, i.e., human-centered and technology-centered [1]. For human-centered, a smart environment has to adapt to the needs of the end users in terms of input/ output capabilities. For technology-centered, new hardware technologies, networking solutions, and middleware services have to be developed. A scenario of how sensor nodes can be used to create a smart environment is described in [11]. The sensor nodes can be embedded into furniture and appliances, and they can communicate with each other and the room server. The room server can also communicate with other room servers to learn about the services they offered, e.g., printing, scanning, and faxing. These room servers and sensor nodes can be integrated with existing embedded devices to become self-organizing, selfregulated, and adaptive systems based on control theory models as described in [11]. The computing and sensing in this environment has to be reliable, persistent, and transparent.

### 1.2.5 Other commercial applications

Some of the commercial applications are monitoring material fatigue, building virtual keyboards, managing inventory, monitoring product quality, constructing smart office spaces, environmental control in office buildings, robot control and guidance in automatic manufacturing environments, interactive toys, interactive museums, factory process control and automation, monitoring disaster area, smart structures with sensor nodes embedded inside; machine diagnosis, transportation, factory instrumentation, local control of actuators, detecting and monitoring car thefts, vehicle tracking and detection; and instrumentation of semiconductor processing chambers, rotating machinery, wind tunnels, and anechoic chambers.

Environmental control in office buildings: The air conditioning and heat of most buildings are centrally controlled. Therefore, the temperature inside a room can vary by few degrees; one side might be warmer than the other because there is only one control in the room and the air flow from the central system is not evenly distributed. A distributed wireless sensor network system can be installed to control the air flow and temperature in different parts of the room.

Vehicle tracking and detection: There are two approaches as described in [22] to track and detect the vehicle: first, the line of bearing of the vehicle is determined locally within the clusters and then it is forwarded to the base station, and second, the raw data collected by the sensor nodes are forwarded to the base station to determine the location of the vehicle.

Detecting and monitoring car thefts: Sensor nodes are being deployed to detect and identify threats within a geographic region and report these threats to remote end users by the Internet for analysis.

## 1.3 Factors influencing sensor network design

A sensor network design is influenced by many factors, which include fault tolerance; scalability; production costs; operating environment; sensor network topology; hardware constraints; transmission media; and power consumption. However, none of these studies has a full integrated view of all factors that are driving the design of sensor networks and sensor nodes. These factors are important because they serve as a guideline to design a protocol or an algorithm for sensor networks. In addition, these influencing factors can be used to compare different schemes.

### 1.3.1 Fault tolerance

Some sensor nodes may fail or be blocked due to lack of power, have physical damage or environmental interference. The failure of sensor nodes should not affect the overall task of the sensor network. This is the reliability or fault tolerance issue. Fault tolerance is the ability to sustain sensor network functionalities without any interruption due to sensor node failures. The reliability R(t) or fault tolerance of a sensor node is modeled in using the Poisson distribution to capture the probability of not having a failure within the time interval (0, t):

$$R(t) = \exp(-\lambda_K t)$$

where $\lambda_K$ and t are the failure rate of sensor node k and the time period, respectively.

Note that protocols and algorithms may be designed to address the level of fault tolerance required by the sensor networks. If the environment where the sensor nodes are deployed has little interference, then the protocols can be more relaxed. The fault tolerance level depends on the application of the sensor networks, and the schemes must be developed with this in mind.

### 1.3.2 Scalability

The number of sensor nodes deployed in studying a phenomenon may be in the order of hundreds or thousands. Depending on the application, the number may reach an extreme value of millions. The new schemes must be able to work with this number of nodes. They must also utilize the high density nature of the sensor networks. The density can range from few sensor nodes to few hundred sensor nodes in a region, which can be less than 10 m in diameter [7]. The density can be calculated according to [4] as:

$$\mu(R) = (N\pi R^2) / A$$

where N is the number of scattered sensor nodes in region A; and R, the radio transmission range.

Basically, μ(R) gives the number of nodes within the transmission radius of each node in region A. In addition, the number of nodes in a region can be used to indicate the node density. The node density depends on the application in which the sensor nodes are deployed. For machine diagnosis application, the node density is around 300 sensor nodes in a 5 x 5 $m^2$ region, and the density for the vehicle tracking application is around 10 sensor nodes per region. In general, the density can be as high as 20 sensor nodes/$m^3$. A home may contain around two dozens of home appliances containing sensor nodes, but this number will grow if sensor nodes are embedded into furniture and other miscellaneous items. For habitat monitoring application, the number of sensor nodes ranges from 25 to 100 per region [5, 22].

### 1.3.3 Production costs

Since the sensor networks consist of a large number of sensor nodes, the cost of a single node is very important to justify the overall cost of the networks. If the cost of the network is more expensive than deploying traditional sensors, then the sensor network is not cost-justified. As a result, the cost of each sensor node has to be kept low. The state of the art technology allows a Bluetooth radio system to be less than 10$ [21]. Also, the price of a PicoNode is targeted to be less than 1$ [20]. The cost of a sensor node should be much less than 1$ in order for the sensor network to be feasible [20]. The cost of a Bluetooth radio, which is known to be a low-cost device, is even 10 times more expensive than the targeted price for a sensor node. In addition, it may be equipped with a location finding system, mobilizer, or power generator depending on the applications of the sensor networks. As a result, the cost of a sensor node is a very challenging issue given the amount of functionalities with a price of much less than a dollar.

### 1.3.4 Hardware constraints

A sensor node is made up of four basic components as shown in Figure 1: a sensing unit, a processing unit, a transceiver unit and a power unit. They may also have application dependent additional components such as a location finding system, a power generator and a mobilizer. Sensing units are usually composed of two subunits: sensors and analog to digital converters (ADCs). The analog signals produced by the sensors based on the observed phenomenon are converted to digital signals by the ADC, and then fed into the processing unit. The processing unit, which is generally associated with a small storage unit, manages the procedures that make the sensor node collaborate with the other nodes to carry out the assigned sensing tasks. A transceiver unit connects the node to the network. One of the most important components of a sensor node is the power unit. Power units may be supported by a power scavenging unit such as solar cells. There are also other subunits, which are application dependent.

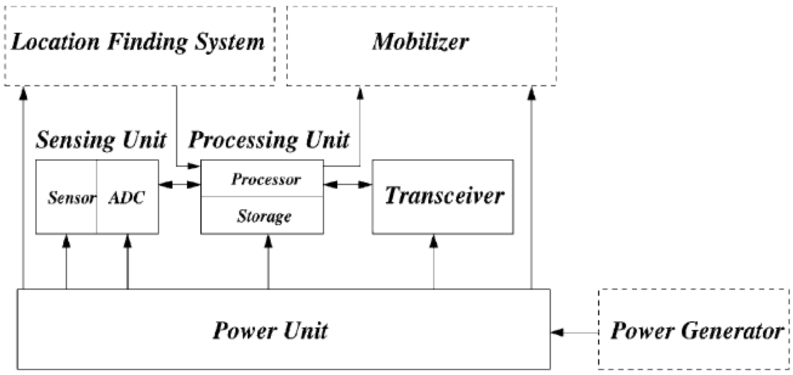


Figure 1: The components of a sensor node.

Most of the sensor network routing techniques and sensing tasks require the knowledge of location with high accuracy. Thus, it is common that a sensor node has a location finding system. A mobilizer may sometimes be needed to move sensor nodes when it is required to carry out the assigned tasks. All of these subunits may need to fit into a matchbox-sized module. The required size may be smaller than even a cubic centimeter which is light enough to remain suspended in the air. Apart from the size, there are also some other stringent constraints for sensor nodes. These nodes must:

- consume extremely low power,
- operate in high volumetric densities,
- have low production cost and be dispensable,
- be autonomous and operate unattended,
- be adaptive to the environment.

Since the sensor nodes are often inaccessible, the lifetime of a sensor network depends on the lifetime of the power resources of the nodes. Power is also a scarce resource due to the size limitations. For instance, the total stored energy in a smart dust mote is on the order of 1 J [19]. For wireless integrated network sensors (WINS) [32], the total average system supply currents must be less than 30 lA to provide long operating life. WINS nodes are powered from typical lithium (Li) coin cells (2.5 cm in diameter and 1 cm in thickness) [32].

The transceiver unit of sensor nodes may be a passive or active optical device as in smart dust motes [19] or a radio frequency (RF) device. RF communications require modulation, band pass, filtering, demodulation and multiplexing circuitry, which make them more complex and expensive. Also, the path loss of the transmitted signal between two sensor nodes may be as high as the fourth order exponent of the distance between them, because the antennas of the sensor nodes are close to the ground [19]. Nevertheless, RF communication is preferred in most of the ongoing sensor network research projects, because the packets conveyed in sensor networks are small, data rates are low (i.e., generally less than 1 Hz) [21], and the frequency re-use is high due to short communication distances. These characteristics also make it possible to use low duty cycle radio electronics for sensor networks. However, designing energy efficient and low duty cycle radio circuits is still technically challenging, and current commercial radio technologies such as those used in Bluetooth is not efficient enough for sensor networks because turning them on and off consumes much energy [22].

### 1.3.5 Environment

Sensor nodes are densely deployed either very close or directly inside the phenomenon to be observed. Therefore, they usually work unattended in remote geographic areas. They may be working

- in busy intersections,
- in the interior of a large machinery,
- at the bottom of an ocean,
- inside a twister,
- on the surface of an ocean during a tornado,
- in a biologically or chemically contaminated field,
- in a battlefield beyond the enemy lines,

• in a home or a large building,
• in a large warehouse,
• attached to animals,
• attached to fast moving vehicles, and
• in a drain or river moving with current.

This list gives an idea about under which conditions sensor nodes are expected to work. They work under high pressure in the bottom of an ocean, in harsh environments such as a debris or a battlefield, under extreme heat and cold such as in the nozzle of an aircraft engine or in arctic regions, and in an extremely noisy environment such as under intentional jamming.

**1.3.6 Transmission media**

In a multihop sensor network, communicating nodes are linked by a wireless medium. These links can be formed by radio, infrared or optical media. To enable global operation of these networks, the chosen transmission medium must be available worldwide.

One option for radio links is the use of industrial, scientific and medical (ISM) bands, which offer license-free communication in most countries. The International Table of Frequency Allocations, contained in Article S5 of the Radio Regulations (Volume 1), species some frequency bands that may be made available for ISM applications. They are listed in Table 1.

Much of the current hardware for sensor nodes is based upon RF circuit design. The IAMPS wireless sensor node, described in [22], uses a Bluetooth- compatible 2.4 GHz transceiver with an integrated frequency synthesizer. The low-power sensor device described in [33], uses a single channel RF transceiver operating at 916 MHz. The WINS architecture [19] also uses radio links for communication.

Another possible mode of internode communication in sensor networks is by infrared. Infrared communication is license-free and robust to interference from electrical devices. Infrared based transceivers are cheaper and easier to build. Many of today's laptops, PDAs and mobile phones offer

| Frequency band | Center frequency |
|---|---|
| 6765–6795 kHz | 6780 kHz |
| 13,553–13,567 kHz | 13,560 kHz |
| 26,957–27,283 kHz | 27,120 kHz |
| 40.66–40.70 MHz | 40.68 MHz |
| 433.05–434.79 MHz | 433.92 MHz |
| 902–928 MHz | 915 MHz |
| 2400–2500 MHz | 2450 MHz |
| 5725–5875 MHz | 5800 MHz |
| 24–24.25 GHz | 24.125 GHz |
| 61–61.5 GHz | 61.25 GHz |
| 122–123 GHz | 122.5 GHz |
| 244–246 GHz | 245 GHz |

Table 1: Frequency bands available for ISM applications

an infrared data association interface. The main drawback though, is the requirement of a line of sight between sender and receiver. This makes infrared a reluctant choice for transmission medium in the sensor network scenario.

The unusual application requirements of sensor networks make the choice of transmission media more challenging. For instance, marine applications may require the use of the aqueous transmission medium. Here, one would like to use long-wavelength radiation that can penetrate the water surface.

### 1.3.7 Power consumption

The sensor node consumes power for sensing, communicating and data processing. More energy is required for data communication than any other process. The energy cost of transmitting 1 Kb a distance of 100 metres (330 ft) is approximately the same as that used for the execution of 3 million instructions by a 100 million instructions per second/W processor. Power is stored either in batteries or capacitors. Batteries, both rechargeable and non-rechargeable, are the main source of power supply for sensor nodes. They are also classified according to electrochemical material used for the electrodes such as NiCd (nickel-cadmium), NiZn (nickel-zinc), Nimh (nickel-metal hydride), and lithium-ion. Current sensors are able to renew their energy from solar sources, temperature differences, or vibration. Two power saving policies used are Dynamic Power Management (DPM) and Dynamic Voltage Scaling (DVS).[6] DPM conserves power by shutting down parts of the sensor node which are not currently used or active. A DVS scheme varies the power levels within the sensor node depending on the non-deterministic workload. By varying the voltage along with the frequency, it is possible to obtain quadratic reduction in power consumption. More on the energy of the WNS will be seen in the following chapter.

### 1.4 Sensor networks communication architecture

The sensor nodes are usually scattered in a sensor field as shown in Fig. 2. Each of these scattered sensor nodes has the capabilities to collect data and route data back to the sink and the end users. Data are routed back to the end user by a multihop infrastructureless architecture through the sink as shown in Figure 2. The sink may communicate with the task manager node via Internet or Satellite.

The protocol stack used by the sink and all sensor nodes is given in Fig. 3. This protocol stack combines power and routing awareness, integrates data with networking protocols, communicates power efficiently through the wireless medium, and promotes cooperative efforts of sensor nodes. The protocol stack consists of the application layer, transport layer, network layer, data link layer, physical layer, power management plane, mobility management plane, and task management plane. Depending on the sensing tasks, different types of application software can be built and used on the application layer. The transport layer helps to maintain the flow of data if the sensor networks application requires it. The network layer takes care of routing the data supplied by the transport layer. Since the environment is noisy and sensor nodes can be mobile, the MAC protocol must be power aware and able to minimize collision with neighbors' broadcast. The physical layer addresses the needs of a simple but robust modulation, transmission and receiving techniques. In addition, the power, mobility, and task management planes monitor the power, movement, and task distribution

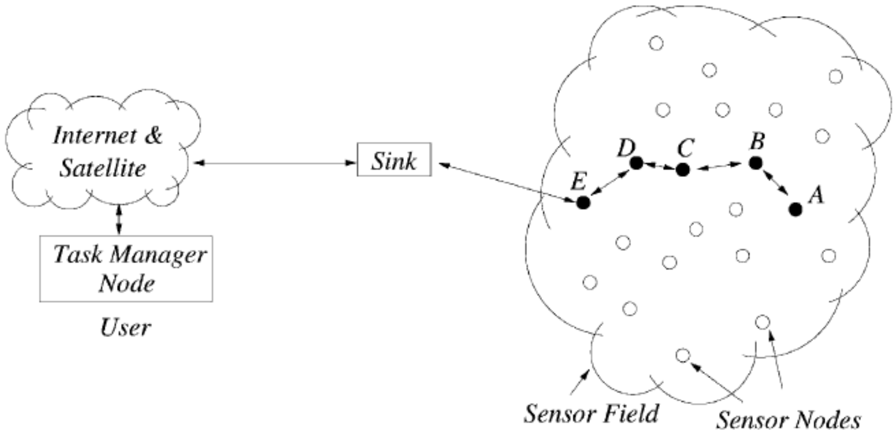


Figure 2: Sensor nodes scattered in a sensor field.

among the sensor nodes. These planes help the sensor nodes coordinate the sensing task and lower the overall power consumption.

The power management plane manages how a sensor node uses its power. For example, the sensor node may turn off its receiver after receiving a message from one of its neighbors. This is to avoid getting duplicated messages. Also, when the power level of the sensor node is low, the sensor node broadcasts to its neighbors that it is low in power and cannot participate in routing messages. The remaining power is reserved for sensing. The mobility management plane detects and registers the movement of sensor nodes, so a route back to the user is always maintained, and the sensor nodes can keep track of who are their neighbor sensor nodes. By knowing who are the neighbor sensor nodes, the sensor nodes can balance their power and task usage. The task management plane balances and schedules the sensing tasks given to a specific region. Not all sensor nodes in that region are required to perform the sensing task at the same time. As a result, some sensor nodes perform the task more than the others depending on their power level. These management planes are needed, so that sensor nodes can work together in a power efficient way, route data in a mobile sensor network, and share resources between sensor nodes. Without them, each sensor node will just work individually. From the whole sensor network standpoint, it is more efficient if sensor nodes can collaborate with each other, so the lifetime of the sensor networks can be prolonged. Before we discuss the need for the protocol layers and management planes in sensor networks, we map three existing work [19,22] to the protocol stack as shown in figure 3.

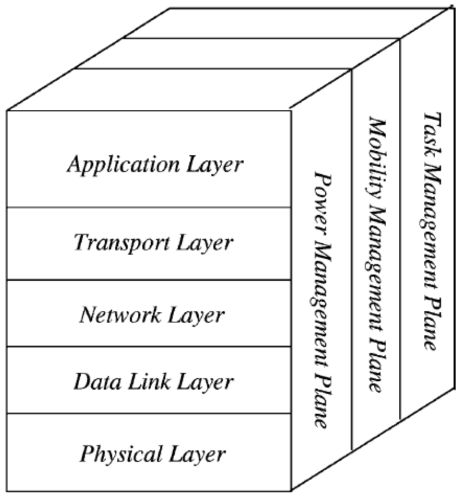


Figure 3: The sensor networks protocol stack.

The so-called WINS is developed in [19], where a distributed network and Internet access is provided to the sensor nodes, controls, and processors. Since the sensor nodes are in large number, the WINS networks take advantage of this short distance between sensor nodes to provide multi hop communication and minimize power consumption. The way in which data is routed back to the user in the WINS networks follows the architecture specified in Fig. 2. The sensor node, i.e., a WINS node, detects the environmental data, and the data is routed hop by hop through the WINS nodes until it reaches the sink, i.e., a WINS gateway. So the WINS nodes are sensor nodes A, B, C, D, and E according to the architecture in Fig. 2. The WINS gateway communicates with the user through conventional network services, such as the Internet. The protocol stack of a WINS network consists of the application layer, network layer, MAC layer, and physical layer. Also, it is explicitly pointed out in [19] that a low-power protocol suite that addresses the constraints of the sensor networks should be developed.

The smart dust motes, i.e., sensor nodes, may be attached to objects or even float in the air because of their small size and light weight. They use

MEMS technology for optical communication and sensing. These motes may contain solar cells to collect energy during the day, and they require a line of sight to communicate optically with the base station transceiver or other motes. Comparing the smart dust communication architecture with the one in Figure 2, the smart dust mote, i.e., the sensor node, typically communicates directly with the base station transceiver, i.e., sink. A peer- to- peer communication is also possible, but there are possible collision problems in medium access due to ‘‘hidden nodes’’. The protocol layers in which the smart dust motes incorporate are application layer, MAC layer, and the physical layer.

Another approach to design protocols and algorithms for sensor networks is driven by the requirements of the physical layer [22]. The protocols and algorithms should be developed according to the choice of physical layer components, such as the type of micro-processors, and the type of receivers. This bottom–up approach of the IAMPS wireless sensor node also addresses the importance of the application layer, network layer, MAC layer, and physical layer as illustrated in Fig. 3 to be tightly integrated with the sensor node’s hardware. The IAMPS wireless sensor node also communicates with the user according to the architecture specified in Fig. 2. Different schemes, such as time division multiple access (TDMA) versus frequency division multiple access (FDMA) and binary modulation versus M-ary modulation are compared in [22]. This bottom–up approach points out that sensor network algorithms have to be aware of the hardware and able to use special features of the micro-processors and transceivers to minimize the sensor node’s power consumption. This may push toward a custom solution for different types of sensor node design. Different types of sensor nodes deployed also lead to different types of sensor networks. This may also lead to different types of collaborative algorithms.

### 1.4.1 Application layer

To the best of our knowledge, although many application areas for sensor networks are defined and proposed, potential application layer protocols for sensor networks remains a largely unexplored region. In this survey, we examine three possible application layer protocols, i.e., sensor management protocol (SMP), task assignment and data advertisement protocol (TADAP), and sensor query and data dissemination protocol (SQDDP), needed for sensor networks based on the proposed schemes related to the other layers and sensor network application areas. All of these application layer protocols are open research issues.

### 1.4.2 Transport layer

The need for transport layer is pointed out in the literature. This layer is especially needed when the system is planned to be accessed through Internet or other external networks. However, to the best of our knowledge there has not been any attempt thus far to propose a scheme or to discuss the issues related to the transport layer of a sensor network in literature. TCP with its current transmission window mechanisms does match to the extreme characteristics of the sensor network environment. An approach such as TCP splitting [4] may be needed to make sensor networks interact with other networks such as Internet. In this approach, TCP connections are ended at sink nodes, and a special transport layer protocol can handle the communications between the sink node and sensor nodes. As a result, the communication between the user and the sink node is by UDP or TCP via the Internet or Satellite; on the other hand, the communication between the sink and sensor nodes may be purely by UDP type protocols, because each sensor node has limited memory.

Unlike protocols such as TCP, the end-to-end communication schemes in sensor networks are not based on global addressing. These schemes must consider that attribute-based naming is used to indicate the destinations of the data packets. The factors such as power consumption and scalability, and the characteristics like data centric routing makes sensor networks need different handling in transport layer. Thus, these requirements stress the need for new types of transport layer protocols.

### 1.4.3 Network layer

Sensor nodes are scattered densely in a field either close to or inside the phenomenon as shown in Fig. 2. Special multihop wireless routing protocols between the sensor nodes and the sink node are needed. The ad hoc routing techniques already proposed in the literature do not usually fit the requirements of the sensor networks. The networking layer of sensor networks is usually designed according to the following principles:

- Power efficiency is always an important consideration.
- Sensor networks are mostly data centric.
- Data aggregation is useful only when it does not hinder the collaborative effort of the sensor nodes.
- An ideal sensor network has attribute-based addressing and location awareness.

### 1.4.4 Data link layer

The data link layer is responsible for the multiplexing of data streams, data frame detection, medium access and error control. It ensures reliable point-to-point and point-to-multipoint connections in a communication network. In the following two subsections, we discuss some of the medium access and error control strategies for sensor networks.

### 1.4.5 Physical layer

The physical layer is responsible for frequency selection, carrier frequency generation, signal detection, modulation and data encryption. Frequency generation and signal detection have more to do with the underlying hardware and transceiver design and hence are beyond the scope of our paper. In the following, we focus on signal propagation effects, power efficiency and modulation schemes for sensor networks.

It is well known that long-distance wireless communication can be expensive, both in terms of energy and implementation complexity. While designing the physical layer for sensor networks, energy minimization assumes significant importance, over and above the decay, scattering, shadowing, reflection, diffraction, multipath and fading effects. In general, the minimum output power required to transmit a signal over a distance d is proportional to dn, where 26n < 4. The exponent n is closer to four for low-lying antennae and near ground channels, as is typical in sensor network communication. This can be attributed to the partial signal cancellation by a ground-reflected ray. While trying to resolve these problems, it is important that the designer is aware of inbuilt diversities and exploits this to the fullest. For instance, multihop communication in a sensor network can effectively overcome shadowing and path-loss effects, if the node density is high enough. Similarly, while propagation losses and channel capacity limit data reliability, this very fact can be used for spatial frequency re-use. Energy efficient physical layer solutions are currently being pursued by researchers. Although some of these topics have been addressed in literature, it still remains a vastly unexplored domain of the wireless sensor networks. A discussion of some existing ideas follows.

# CHAPTER 2

# 2. Principal Components Analysis

## 2.1 Introduction

Principal component analysis (PCA) involves a mathematical procedure that transforms a number of possibly correlated variables into a number of uncorrelated variables called principal components, related to the original variables by an orthogonal transformation. This transformation is defined in such a way that the first principal component has as high a variance as possible (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it be orthogonal to the preceding components. PCA is sensitive to the relative scaling of the original variables. Depending on the field of application, it is also named the discrete Karhunen–Loève transform (KLT), the Hotelling transform or proper orthogonal decomposition (POD).

PCA was invented in 1901 by Karl Pearson. [1] Now it is mostly used as a tool in exploratory data analysis and for making predictive models. PCA can be done by eigenvalue decomposition of a data covariance matrix or singular value decomposition of a data matrix, usually after mean centering the data for each attribute. The results of a PCA are usually discussed in terms of component scores (the transformed variable values corresponding to a particular case in the data) and loadings (the variance each original variable would have if the data were projected onto a given PCA axis).

PCA is the simplest of the true eigenvector-based multivariate analyses. Often, its operation can be thought of as revealing the internal structure of the data in a way which best explains the variance in the data. If a multivariate dataset is visualised as a set of coordinates in a high-dimensional data space (1 axis per variable), PCA can supply the user with a lower-dimensional picture, a "shadow" of this object when viewed from its (in some sense) most informative viewpoint. This is done by using only the first few principal components so that the dimensionality of the transformed data is reduced.

Principal Components Analysis (PCA) is a way of identifying patterns in data, and expressing the data in such a way as to highlight their similarities and differences. Since patterns in data can be hard to find in data of high dimension, where the luxury of graphical representation is not available, PCA is a powerful tool for analysing data. The other main advantage of PCA is that once you have found these patterns in the data, and you compress the data, ie. by reducing the number of dimensions, without much loss of information. This technique used in image compression. This chapter will take you through the steps you needed to perform a Principal Components Analysis on a set of data. I am not going to describe exactly why the technique works, but I will try to provide an explanation of what is happening at each point so that you can make informed decisions when you try to use this technique yourself.

## 2.2 Goals of PCA

The goals of pca are to (a) extract the most important information from the data table, (b) compress the size of the data set by keeping only this important information, (c) simplify the description of the data set, and (d) analyze the structure of the observations and the variables.

In order to achieve these goals, PCA computes new variables called principal components which are obtained as linear combinations of the original variables. The first principal component is required to have the largest possible variance (i.e., inertia and therefore this component will “explain" or “extract" the largest part of the inertia of the data table). The second component is computed under the constraint of being orthogonal to the firrst component and to have the largest possible inertia. The other components are computed likewise [26]. The values of these new variables for the observations are called factor scores, these factors scores can be interpreted geometrically as the projections of the observations onto the principal components.

## 2.3 Method

**Step 1: Organize the data set**

Suppose you have data comprising a set of observations of M variables, and you want to reduce the data so that each observation can be described with only L variables, L < M. Suppose further, that the data are arranged as a set of N data vectors $x_1$ . . . $x_N$ with each $x_n$ representing a single grouped observation of the M variables.

- Write $x_1$ . . . $x_N$ as column vectors, each of which has M rows.
- Place the column vectors into a single matrix X of dimensions M × N.

**Step 2: Subtract the mean**

For PCA to work properly, you have to subtract the mean from each of the data dimensions. The mean subtracted is the average across each dimension.

- Find the empirical mean along each dimension m = 1, ..., M.
- Place the calculated mean values into an empirical mean vector u of dimensions M × 1.

$$u[m] = \frac{1}{N}\sum_{n=1}^{N} X[m,n]$$

Mean subtraction is an integral part of the solution towards finding a principal component basis that minimizes the mean square error of approximating the data. Hence we proceed by centering the data as follows:

- Subtract the empirical mean vector u from each column of the data matrix X.
- Store mean-subtracted data in the M × N matrix B.

$$B = X - uh$$

where h is a 1 × N row vector of all 1s: h[n] = 1 for n=1,…,N

**Step 3: Calculate the covariance matrix**

- Find the M × M empirical covariance matrix C from the outer product of matrix B with itself:

$$C = E[B \cdot B^*] = \frac{1}{N} \sum B \cdot B^*$$

where E is the expected value operator, and * is the conjugate transpose operator. Note that if B consists entirely of real numbers, which is the case in many applications, the "conjugate transpose" is the same as the regular transpose.

**Step 4: Calculate the eigenvectors and eigenvalues of the covariance matrix**

- Compute the matrix V of eigenvectors which diagonalizes the covariance matrix C:

$$V^{-1}CV = D$$

where D is the diagonal matrix of eigenvalues of C. This step will typically involve the use of a computer-based algorithm for computing eigenvectors and eigenvalues.

- Matrix D will take the form of an M × M diagonal matrix, where

$$D[p,q] = \lambda_m \quad \text{for} \quad p = q = m$$

is the mth eigenvalue of the covariance matrix C, and

$$D[p,q] = 0 \quad \text{for} \quad p \neq q$$

- Matrix V, also of dimension M × M, contains M column vectors, each of length M, which represent the M eigenvectors of the covariance matrix C.
- The eigenvalues and eigenvectors are ordered and paired. The mth eigenvalue corresponds to the mth eigenvector.
- Sort the columns of the eigenvector matrix V and eigenvalue matrix D in order of decreasing eigenvalue.

**Step 5: Choosing components and forming a feature vector**

- Save the first $L$ columns of V as the $M \times L$ matrix W:

$$W[p,q] = V[p,q] \quad \text{for} \quad p = 1,\ldots,M \quad q = 1,\ldots L$$

$$\text{where } \operatorname{cov}(Y)P^{-1} = P^{T}$$

Here is where the notion of data compression and reduced dimensionality comes into it. The eigenvectors and eigenvalues are quite different values. In fact, it turns out that the eigenvector with the highest eigenvalue is the principle component of the data set. In general, once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. This gives the components in order of significance. The next step is to decide to ignore the components of lesser significance. Data lose some information, but if the eigenvalues are small, there isn't much loss. If the decision is to leave out some components, the final data set will have less dimensions than the original. To be precise, if originally are -n- dimensions in the first data, and find -n- eigenvectors and eigenvalues, and then the choose is to keep only the first p-eigenvectors, then the final data set has only p- dimensions. What needs to be done now is you need to form a feature vector, which is just a fancy name for a matrix of vectors.

**Step 6: Deriving the new data set**

Once we have chosen the components (eigenvectors) that we wish to keep in our data and formed a feature vector, we simply take the transpose of the vector and multiply it on the left of the original data set, transposed.

$$F = W^{T}B$$

Where F is the final data, and W and B have been reported in the previous steps. F is L x M matrix, because W is M x L matrix and B is M x N matrix.

Basically we have transformed our data so that is expressed in terms of the patterns between them, where the patterns are the lines that most closely describe the relationships between the data. This is helpful because we have now classified our data point as a combination of the contributions from each of those lines.

### 2.4 Getting the old data back

Wanting to get the original data back is obviously of great concern if you are using the PCA transform for data compression. Before we do that, remember that only if we took all the eigenvectors in our transformation will we get exactly the original data back. If we have reduced the number of eigenvectors in the final transformation, then the retrieved data has lost some information. Recall that the final transform is this:

$$F = W^T B$$

which can be turned around so that, to get the original data back,

$$B = (W^T)^{-1} F$$

where $(W^T)^{-1}$ is the inverse of $W^T$. However, when we take all the eigenvectors in our feature vector, it turns out that the inverse of our feature vector is actually equal to the transpose of our feature vector. This is only true because the elements of the matrix are all the unit eigenvectors of our data set. This makes the return trip to our data easier, because the equation becomes

$$B = WF$$

But, to get the actual original data back, we need to add on the mean of that original data (remember we subtracted it right at the start). So, for completeness,

$$B = WF + uh$$

where h is a 1 × N row vector of all 1s: h[n] = 1 for n=1,…,N , and u[m] is the mean values, so the uh is the original mean.

This formula also applies to when you do not have all the eigenvectors in the feature vector. So even when you leave out some eigenvectors, the above equation still makes the correct transform.

# CHAPTER 3

# 3. Energy Efficiency

## 3.1 Introduction

A wireless sensor network consists of a large number of nodes each of which integrates one or more sensors, a processing subsystem and a short range transceiver. The nodes are capable of organizing themselves to establish and maintain a network and carry out reliable sensing. However, when considered individually, each node is a simple device; the components that make up its subsystems are cheap (and often commonplace) off-the-shelf components. Ideally, the network should have a long life and operate unattended. But several factors put a limit to the energy reservoir:

1) Considering the complexity of the task for which they are deployed – namely, sensing, processing, and communication –, the nodes are very small in size to accommodate high capacity batteries.

2) Typically, a wireless sensor network consists of a large number of nodes and the network is wished to operate unattended. This makes manually replacing or recharging batteries a formidable challenge.

3) Whereas research is being conducted to employ renewable energy and self-recharging mechanisms, still the size of presently available nodes makes the task difficult.

4) The failure of a few numbers of nodes may fragment the entire network prematurely.

The problem of power consumption has been addressed in two different ways in the literature. In the first, a large number of energy-efficient communication protocols – most significantly, MAC, routing and self-organization protocols – that take the peculiarities of wireless sensor networks into account are proposed. In the second, dynamic power management

(DPM) strategies are developed to recognize and minimize the impact of wasteful and inefficient activities in the network.

The problem of power consumption has been addressed in two different ways in the literature. In the first, a large number of energy-efficient communication protocols – most significantly, MAC, routing and self-organization protocols – that take the peculiarities of wireless sensor networks into account are proposed. In the second, dynamic power management (DPM) strategies are developed to recognize and minimize the impact of wasteful and inefficient activities in the network.

Wasteful and inefficient activities can be accidental side effects or results of non-optimal software and hardware configurations. For example field observations revealed that some nodes exhausted their batteries prematurely because of unexpected overhearing of traffic that caused the communication subsystem to become active for a time longer than originally anticipated [27]. Similarly, some nodes exhausted their batteries prematurely because they aimlessly attempted to establish links with a network that has become inaccessible to them.

Most inefficient activities are, however, results of nonoptimal configurations in hardware and software components. For example, a considerable amount of power can be dissipated in an idle processing or communication subsystem. Similarly, a receiver that aimlessly receives packets that are not destined to it; or overhears while neighboring nodes communicate with each other consumes a significant amount of power.

A local DPM strategy ensures that such wasteful activities are avoided and power is consumed frugally. Ideally, it provides each subsystem of a node with the amount of power that is sufficient enough to carry out a task at hand. When there is no task to be processed or executed, it forces the subsystem to operate at the most economical power mode or turns it off altogether.

There has been a considerable interest in the past, and as a consequence, a significant body of work, in dynamic power management, particularly, in the context of embedded systems. But wireless sensor networks bring their own challenges and peculiarities into the research field. To begin with, unlike embedded systems, which function, by and large, stand-alone, no individual node is of interest in and of itself. Secondly, a local decision made by a node can have a global impact. This paper attempts to provide a comprehensive insight into aspects of DPM in wireless sensor networks. It presents the challenges, the results that are achieved so far, and some outstanding research issues in need of attention.

### 3.2 Power consumption

The wireless sensor node, being a micro-electronic device, can only be equipped with a limited power source (<0.5 Ah, 1.2 V). In some application scenarios, replenishment of power resources might be impossible. Sensor node lifetime, therefore, shows a strong dependence on battery lifetime. In a wireless sensor network, in which the communication is achieved through multi-hop information dissemination, each node plays the dual role of data originator and data router. The dis-functioning of few nodes can cause significant topological changes and might require re-routing of packets and re-organization of the network. Hence, power conservation and power management take on additional importance. It is for these reasons that researchers are currently focusing on the design of power-aware protocols and algorithms for sensor networks.

In other mobile and ad hoc networks (MONET), power consumption has been an important design factor, but not the primary consideration, simply because power resources can be replaced by the user. The emphasis is more on QoS provisioning than the power efficiency. In sensor networks though, power efficiency is an important performance metric, directly influencing the network lifetime. Application specific protocols can be designed by appropriately trading off other performance metrics such as delay and throughput with power efficiency.

The main task of a sensor node in a sensor field is to detect events, perform quick local data processing, and then transmit the data. Power consumption can hence be divided into three domains: sensing, communication, and data processing.

The sensing unit and its components were introduced in Section 1.3.4 Sensing power varies with the nature of applications. Sporadic sensing might consume lesser power than constant event monitoring. The complexity of event detection also plays a crucial role in determining energy expenditure. Higher ambient noise levels might cause significant corruption and increase detection complexity. Power consumption in data communication and processing are discussed in detail in the following subsections.

### 3.2.1 Communication

Of the three domains, a sensor node expends maximum energy in data communication. This involves both data transmission and reception. It can be shown that for short-range communication with low radiation power (~0 dbm), transmission and reception energy costs are nearly the same. Mixers, frequency synthesizers, voltage control oscillators, phase locked loops (PLL) and power amplifiers, all consume valuable power in the transceiver circuitry. It is important that in this computation we not only consider the active power but also the start-up power consumption in the transceiver circuitry. The start-up time, being of the order of hundreds of micro-seconds, makes the start-up power non-negligible. This high value for the start-up time can be attributed to the lock time of the PLL. As the transmission packet size is reduced, the start-up power consumption starts to dominate the active power consumption. As a result, it is inefficient in turning the transceiver ON and OFF, because a large amount of power is spent in turning the transceiver back ON each time.

In [22], the authors present a formulation for the radio power consumption (Pc) as:

$$Pc = N_T [P_T (T_{on} + T_{st}) + P_{out} (T_{on})] + N_R [P_R (R_{on} + R_{st})]$$

where $P_T$=R is the power consumed by the transmitter/ receiver; $P_{out}$, the output power of the transmitter; T = $R_{on}$, the transmitter/receiver on time; T = $R_{st}$, the transmitter/receiver start-up time and $N_T$=R, the number of times transmitter/receiver is switched on per unit time, which depends on the task and medium access control (MAC) scheme used. Ton can further be rewritten as L = R, where L is the packet size and R, the data rate. Today's state-of-the-art low power radio transceiver has typical PT and PR values around 20 dbm and Pout close to 0 dbm [15]. Note that PicoRadio aims at a Pc value of -20 dbm.

The design of a small-sized, low-cost, ultralow power transceiver is discussed in [28]. A direct conversion architecture is proposed for the transceiver circuitry. Based on their results, the authors present a power budget and estimate the power consumption to be at least an order of magnitude less than the values given above for $P_T$ and $P_R$ values.

### 3.2.2 Data processing

Energy expenditure in data processing is much less compared to data communication. The example described in [19], effectively illustrates this disparity. Assuming Rayleigh fading and fourth power distance loss, the energy cost of transmitting 1 KB a distance of 100 m is approximately the same as that for executing 3 million instructions by a 100 million instructions per second (MIPS)/W processor. Hence, local data processing is crucial in minimizing power consumption in a multihop sensor network.

A sensor node must therefore have built-in computational abilities and be capable of interacting with its surroundings. Further limitations of cost and size lead us to the choice of complementary metal oxide semiconductor (CMOS) technology for the micro-processor. Unfortunately, this has inbuilt limitations on energy efficiency. A CMOS transistor pair draws power everytime it is switched. This switching power is proportional to the switching frequency, device capacitance (which further depends on the area) and square of the voltage swing. Reducing the supply voltage is hence an effective means of lowering power consumption in the active state. Dynamic voltage scaling, explored in [29,30], aims to adapt processor power supply and operating frequency to match workloads. When a micro-processor handles time-varying computational load, simply reducing the operating frequency during periods of reduced activity results in a linear decrease in power consumption, but reducing the operating voltage gives us quadratic gains. On the other hand, this compromises on peak performance of the processor. Significant energy gains can be obtained by recognizing that peak performance is not always desired and therefore, the processor's operating voltage and frequency can be dynamically adapted to instantaneous processing requirements.

The power consumption in data processing ($P_p$) can be formulated as follows:

$$P_p = CV_{dd}^2 f + V_{dd} I_0 e^{V_{dd}} / n' V_T$$

where C is the total switching capacitance; Vdd, the voltage swing and f, the switching frequency. The second term indicates the power loss due to leakage currents. The lowering of threshold voltage to satisfy performance requirements results in high subthreshold leakage currents. Coupled with the low duty cycle operation of the micro-processor in a sensor node, the associated power loss becomes significant [22].

It is to be noted that there may be some additional circuitry for data encoding and decoding. Application specific integrated circuits may also be used in some cases. In all these scenarios, the design of sensor network algorithms and protocols are influenced by the corresponding power expenditures, in addition to those that have been discussed.

# CHAPTER 4

# 4. Principal Component-based Data and Context Forwarding Model

## 4.1 Introduction

In a Wireless Sensor Network (WSN) sensor nodes measure the temporal-spatial field of a wide variety of contextual (environmental) scalar parameters, e.g., temperature, wind speed, humidity, and return their measurements to a sink (or data fusion center). This gives the maximum amount of information on the data field but at the same time requires the maximum amount of energy to collect information. Instead, it may be possible to use a reduced amount of energy to obtain sufficiently accurate approximation of the data field.

Contextual data collected by a WSN is a challenging problem. Due to very limited resources, e.g., energy, computational power, data storage and bandwidth, it is not a sound technical decision to apply strategies like forwarding any sensor data directly to a sink that does the corresponding processing. Although this problem has received considerable attention in the last years both from industry and research, it is of high importance to take into consideration the nature of the sensed data in order to avoid significant energy consumption and improve bandwidth utilization.

Given the resource constraints in the underlying network infrastructure (e.g., a wireless sensor network), we try to exploit the vector nature of the exchanged contextual information and the possible statistical dependencies between the different contextual (information) components. Our compression scheme capitalizes on the above assumptions which are frequently encountered in real life applications and integrated systems. The proposed context compression is accomplished by mapping–projecting (prior to transmission) the generated context vectors to a reduced dimensions space defined on the principal components (PC) basis. In such basis, contextual data are uncorrelated. The basis is obtained by adopting the Principal Component Analysis (PCA), a well known technique in multivariate data analysis. Reducing the dimensions of contextual vectors is catalytic to the compression task. Practically many of the contextual components are described through the PCA–identified principal components and eliminated from the context reporting messages. We propose, implement and evaluate a Principal Components-based Context Forwarding model that improves the forwarding efficiency of context in a wireless sensor network (WSN) based on the current principal components. The model presents two important benefits: the PCA provides varying levels of context compression accuracies, ranging from constant approximations to the full recovery of the original context; principal components contain sufficient information for re–producing the original data for further processing/ interpretation or forwarding. The model adopts a periodic calculation of the principal components. The exact PCA scheduling is based on a short history of context vectors, the induced re–production error and error change. This periodic calculation affects the context transmission process and the overall context accuracy.

### 4.2 Model Assumptions and Description

The term diffusible calculation (pervasive computing) is used to describe the trend in the use of numerous devices and accessible account, which often are mobile (phones, pads) and escape the attention of people as embedded in the environment, connected directly to a network structure which is becoming more ubiquitous.

Diffusible calculation in interactive environments is based mainly on sensory data originated from the real environment. We collect these data from real environment with a number of sensors which are spread in it. The sensors are being organized so they can route their measurements towards a base station or a sink. The base station is able to communicate through internet with the application's management application. This organization is a

WSN. Sensors are either inside the subtended area (e.g., flow meter in river) or outside of it (e.g., camera opposite woodland). They can be in a standard position (e.g., smoke detector in a building) or movable (e.g., sensor in wild animal).

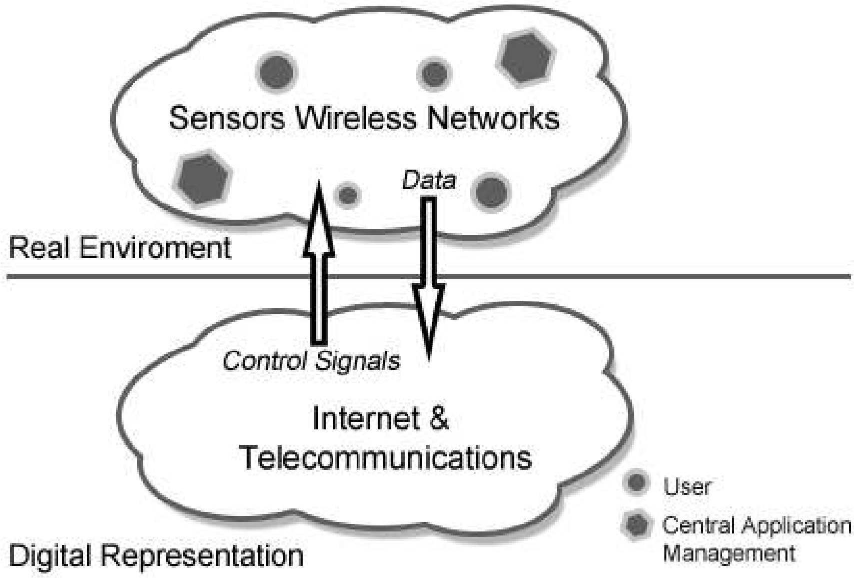


Figure 4: Real environment – digital representation.

Necessary requirement for PCA can be applied is the exported data format by the sensors of the WSN. These networks produce vectorial contextual information and data are represented as vector of dimension N > 0. Examples of such data may be derived from WSN located in environment and refer to temperature, humidity, wind speed, wind direction and others where each vector consists of as many components as there are elements that counting sensor one time. From one such process is expected to occur a number of vectors which should be promoted on the basis of WSN the sink. In the next table (table 2) are shown some of the sensors measurements of a WSN with 7-dimensional vectorial pieces of context . So the data are vectors of the nine dimensions, where have been measured at four different times.

| temp52 | hum52 | temp62 | hum62 | temp71 | hum71 | wind-speed2 |
|---|---|---|---|---|---|---|
| 14.4000 | 64.2000 | 17.4000 | 55.5000 | 15.0000 | 65.8000 | 0.3300 |
| 15.2000 | 59.8000 | 20.7000 | 44.9000 | 15.2000 | 61.7000 | 0.4200 |
| 15.7000 | 56.2000 | 17.9000 | 50.7000 | 15.3000 | 63.2000 | 0.6000 |
| 15.8000 | 61.0000 | 16.3000 | 53.5000 | 15.5000 | 59.2000 | 1.0400 |

Table 2: Examples of Vectors from measurement nodes. The data are temperature and humidity measured by three nodes and wind speed from a different node.

Then, the structure of the WSN must be described according to the nodes so the mathematical processes that nodes implementing can be analysed. More specifically, the simple model of WSN which will be examined is a network consisted of two nodes, X and Y, where node X sends data to the peer node Y. This simple model that we are considering is depicted in Figure 5. The X node produces the data – vectors from measurements for a specific time period. Then node X executes PCA analysis in order to calculate and extract the principal components of the m vectors from total data set and export the correlation matrix of an M vector with a periodicity (i.e. the correlation matrix is generally sent fewer times than PCA is applied). After that node X sends to node Y the data which they are compressed by PCA, and sends the correlation matrix. Then node Y re-products the compressed data that have been received, by applying the reverse process.

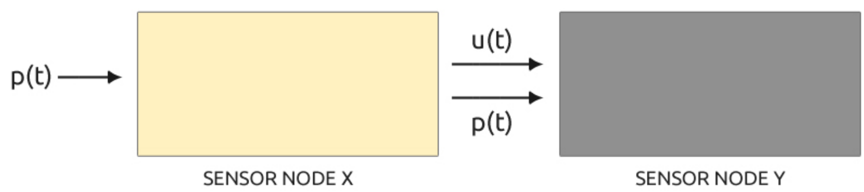


Figure 5: Node X transmits either data p or a data reproduction signal u to the upstream node Y.

With this process we will prove that an important reduction actually exists in the energy consumption in the node which sends the data, as all vectors that measure the node are not constantly dispatched. It should be emphasized however, that in order for this PCA technique has positive results, the elements that calculate the nodes of WSN should be connected to each other. Otherwise this process will bring no results.

### 4.3 Model Evaluation and Performance

Afterwards we will analyze the process more, in order to focus in the subject of energy consumption. In the general case, we have m lines from data with dimension n. As it has been said in order to execute PCA analysis in the m x n data of - table and we have two phases:

In the first phase the node - sender X sends the m lines from n dimensions in the node - recipient [Y]. Thus there is a transportation cost in m x n data, from where we can calculate for each data at a time its cost of transportation per Joule (bit/Joule).

In the second phase PCA is executed, and after the most important (principal) component (or the most important components) is calculated from precedents m x n data, we send for k times data k x q, where q is the least of n, therefore it is reported in those dimensions which are the most basic components.

Hence we have:

[F1] m times we send n prices, we have the cost of mission for every price.

[F2] k times we send q prices, we have the cost of mission for every price.

If the cost for a price is J in byte per Joule then:

CostF1F2 = cost-phase-1 + cost-phase-2 = (m x n x J) + (k x q x J).

Finally, it should be marked that in the beginning of the second phase, phase F2, we send the (covariance matrix) table of dependences -eigenvectors- of dimension n x q from the sender in the recipient (node H in node Y). Hence the transportation cost of the dependences table is (n x q x J).

Also, in order to calculate the n x q table we should consume energy per Joule (instructions per Joule) (in this spot we are talking about energy consumption for the calculation of mathematic action). Here, we keep in mind the complexity of PCA is a function of m, n. Thus we also have the calculation cost of PCA that is K (PCA). This cost is calculated, therefore:

Cost = CostF1 F2 + (n x q x J) + K (PCA)

Do note that this is the cost for a "season" that is to say a period that is constituted by the phases from F1 and F2.
If we have e periods and calculate the additive cost repetitive we have:

Cost = 0
For period = 1: e
Cost = Cost + Cost F1F2 + (n x q x J) + K (PCA)
End_For

In case we do not have this policy, we do not have periods, and in general node X sends all given n lines for all time periods, that is, simply from the start until the measurements finish. Hence the cost in the simple case will be:

Cost Y = 0
For each n prices that come each time periods until the end
Cost Y = Cost Y + (n x J)
End_For

With this analysis of two cases we can pronounce in energy gain that is Cost/Cost Y, that is the ratio which indicates the energy gain. The smaller the ration than the unity is the more effective the technique in the WSN particular, which studies some certain concrete characteristics.

After we have analyzed the General case of our model, then we will see some technical applications of PCA's technique to a WSN, in order to analyze the basic parameters of the algorithm as well as exporting them. The two fundamental parameters of PCA's algorithm is the memorystep (m) where is the number of vectors from data that counts a node on the network, where the amount of data becomes the PCA analysis as described in paragraph 2.3. Also worth mentioning that this amount of data are integral transferred, meaning that all the vector components are transferred as measured.

I.e. from node X are transferred to Y m times, from n elements (where n are the components of vectors). Furthermore, as it is understandable, the memorystep takes place during the first phase. The second important parameter is the bellepoque (k) where is the number of vectors by sending compressed data, based on PCA analysis that has been done. In other words, during the second phase of this procedure, k vectors are being sent of q components, where q < n.

In this point we should mention that from the moment we seek greater energy savings, it is important that the data are compressed. Therefore, in this case we choose bellepoque parameter be greater than the memorystep parameter. In general, bellepoque can be smaller than memorystep, but in this case it actually “provides“ fewer vectors relative to the number of vectors from which comes to the conclusions for the data correlation that the network counts.

But to greater efficiency must memorystep <= bellepoque be valid, and generally as bigger the bellepoque is, as wider the data compression is achieved. But it must be said, that the larger the bellepoque is compared to memorystep, then the bigger the mistake is towards to the PCA' s compressed data. Also, it should be mentioned that the bigger the memorystep parameter is, the better the PCA's results from the extracted analysis are therefore we can have a smaller amount of mistakes in compressed data. Therefore, since this relation between these two parameters exists, the results should be examined by different sets of values for the memorystep and bellepoque parameters.

In the Matlab code where this procedure has been applied, the parameters memorystep and bellepoque change in order to be compared with the exported results so the most suitable values to be found. More specifically, the code of the memorystep starts with a value of 20, that is that a

PCA corresponds for 20 vectors and it comes as a result a value of associations between elements, and ascends gradually with step 10 and reaches a maximum value up to 50. As for the bellepoque parameter begins with 20 as the smallest value and increases in step 10 to 100. And with these combinations of the two parameter's values, executions are being made and the appropriate results are being exported.

We adopted the Mica2 energy consumption model [9]. Mica2 operates with a pair of AA batteries that approximately supply 2200 mAh with effective average voltage 3V. It consumes 20mA if running a sensing application continuously which leads to a lifetime of 100 hours. The energy costs for single CPU instructions (energy per second) and transmitting/receiving either data p(t) or signal u(t) (energy per bit) are summarized in Table 3.

| Node Operation | Mode Energy Cost |
|---|---|
| Instruction Execution | 4 nJ/instruction |
| Idle – Stand by | 9.6 mJ/s - 0.33 mJ/s |
| Transmitting - Receiving | 720 nJ/bit - 110 nJ/bit |

Table 3: Energy Costs

According to Table 3 the costs of models have been calculated, i.e. the energy consumed by the simple transmission (Simplest Data Forwarding (SDF) model) and the model that applies the PCA method. More specifically, in Matlab code the costPCA () function calculates the values classic_schema_Cost and advanced_schema_Cost which are respectively the costs of the SDF model and PCA model for a repetition of the PCA.

Thus, the total costs for all vectors to be sent are respectively classic_schema_Cost * and advanced_schema_Cost * maxloop maxloop where maxloop is the number of repetitions performed by the PCA.

Then, the value of energy_gain is energy_gain=cost_model/cost_classic where cost_model, cost_classic are the corresponding values for the proposed PCA model and the SDF model. Since energy_gain is a fraction, the desired value is as close to zero as possible. Apart from the energy_gain value, the Reproduction Error must be calculated as well, i.e. whether data in node Y are identical to the data sent from the node X. This value is the error code.

This value is the error_model of the PCADataTransmissionModel () function of the code and is calculated as follows

$$\text{error\_model} = \text{mean}\left(\left|\frac{X - rX}{X}\right|\right)$$

where X is the initial data which are vectors, rX are the remanufactured data, similarly vectors and the word “mean” indicates that it takes the average value. The relevant line of the code is
error_model = mean (mean (abs (((Xp-rXp)/Xp.)))).

From the error_model expression, we can see that the desired value is equally close to zero. PCADataTransmissionModel () function performs the PCA method and calculates both costs and Reproduction Error.

Considering that energy gain and Reproduction Error are calculated , i.e. the amounts energy_gain and error_model, we define an holistic metric H, a metric that takes into account the two amounts. The expression of this metric is

$$H = \sqrt{\text{energy\_gain}^2 + \text{error\_model}^2}$$

As soon as the desired values of these two characteristics are as close to zero as possible, so the desired value of the metric H is near zero as well.

The code tests were made for data with 9 contextual parameters and 521 measurements. Part of the data has been shown in table 2 and all the data appear in the Annex at the end.

Afterwards the results of PCA are shown for the data process described above, for various values of the parameters memory_step and bellepoque, in order to find the best results, i.e. the greatest gain of energy (the largest energy_gain) and simultaneously the smallest Reproduction Error (less error_model), commonly referred to as the best value of the metric H (the smallest H).

## 4.4 Performance Assessment

### 4.4.1 Simulation Setup

The runs of the PCA-based model refer to different values of memory_step and bellepoque. More precisely, the variable memory_step has a starting value of 20, and by using steps of 10, it reaches the value 40. The variable bellepoque has a starting value of 20, and by using steps of 10, it reaches the value 100.

After further analysis of the application in MATLAB, function maim.m reads the WSN data from a text file and then defines the maximum and minimum parameter values as well as the number of principal components the PCA will have to keep.

Afterwards, there are two loops with which the code tests the results of PCA for various values of the previously mentioned memory_step (m) and bellepoque_step (k) parameters. Then, it produces a plot with the results of energy_gain and error for a constant memory_step but different bellepoque_step values.

In these loops, the called function is [error_model cost_model cost_classic] = PCADataTransmissionModel(m, k, n, q, A), which produces

- the error_model which is the Reproduction Error
- the cost_model which is the energy cost of PCA model and
- the cost_classic which is the energy cost of SDF model.

The inputs of this function are:

- m, which is memory_step
- k, which is bellepoque_step
- n, which is the vector-data population inside the text file
- q, which represents the number of principal components that will be kept from the vectors and
- A, which represents all the WSN data from the text file

The function PCADataTransmissionModel () is a loop that executes the PCA procedure for every number of data m defined by the memory_step and which will be promoted as they are. It also provides the following k vectors defined by bellepoque_step. Therefore, the number of iterations which PCA will be executed, is approximately n/(m + k). In these instances function [P Y] = reduceparameter (X) is required, which is basically the princomp() Matlab function, and in which x is the data number of m, P is the eigenvector matrix sorted by relevance from the highest to lowest and Y is the transformed data (pure of mean) on the eigenvector axes. In fact, inside the large loop code initially gets the m data first, performs PCA and extracts the eigenvector matrix P and transformed data Y. Subsequently, during the high recurrence, an individual repetition is being executed which will provide the next k data, Xp. The individual repetition is made with step two, that is commonly in order for the system predicts the data it takes into account the initial data as pairs.

In order for this process to be complete, we must first calculate the average prices, meanXp, in k vectors. Then, the transformed data, Yp should be calculated, according to the eigenvector matrix P which has been extracted from previous vectors m.

$$Yp = meanXp \cdot P$$

And finally, to calculate the tale data, rXp, the reverse process of PCA is being done .

$$rXp = Yp \cdot P^{T} + meanXp$$

When k predicted data are calculated, function calculates the differences which have been identified by the normally corresponding data in table Diff, so the error can be calculated. To calculate the energy costs the function [classic_schema_Cost advanced_schema_Cost] = costPCA (m, k, n, q) is required, where classic_schema_Cost and advanced_schema_Cost are respectively the SDF and PCA costs and are calculated accordance with the values in table 3.

### 4.4.2 Performance Assessment with One Principal Component

The following table (table 4) summarizes all the values of the results according to the parameter's values Considering that is being kept one principal component. The largest energy gain (low value of the energy_gain) is 0.2274 and is displayed for values m = 20 and k = 100, i.e. ratio (k/m) = 5. A smaller Reproduction error is 0.003155 and is displayed for values m = 40 and k = 100, i.e. ratio (k/m) = 2.5. However, the catalytic result is the value of H metric, where the smallest value of H is 0.228, i.e. 22.8% and and is exported from the values m = 20 and k = 100, i.e. ratio (k/m) = 5.

Analyzing a little further the values of the results, it can be distinguished that prices of Reproduction error is very small for all the tests performed. The maximum value of the error is 0.02451 and minimum as stated is 0.003155, i.e. the rank of error is 0.3%-2.5%. The of Energy gain can be distinguished that it takes greater range of values. That is, the maximum value of the energy gain (where we have little energy saving) is 0.7368 and the minimum is 0.2274, which means that in percentage is approximately 74%-23%. Finally, the metric H is a function of error and energy gain, and it takes a wide range of values. The largest H value (which indicates that there are no good results) is 0.737 and the minimum is 0.228 which indicates that in these parameter values we have the best results. The range of the values of H is approximately 74%-23%. Since the small Reproduction error is being held in a low level and does not vary in all tests of m and k in contrast to the energy gain, there is a wider range of values, we can say that the energy gain contributes more in value of H thereby the decision that will take,for the best values of the parameters memory_step (m) and bellepoque_step (k). Therefore, the ideal values of these parameters are m = 20 and k = 100, ie ratio = 5 (k / m).

| Memory step - m | Bellepoque step - k | Ratio - k/m | Energy gain | Error | H |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| 20 | 20 | 1 | 0,5393 | 0,0125 | 0,5395 |
| 20 | 30 | 1,5 | 0,4457 | 0,01864 | 0,4461 |
| 20 | 40 | 2 | 0,3833 | 0,00708 | 0,3834 |
| 20 | 50 | 2,5 | 0,3388 | 0,01871 | 0,3393 |
| 20 | 60 | 3 | 0,3054 | 0,02451 | 0,3064 |
| 20 | 70 | 3,5 | 0,2794 | 0,004867 | 0,2794 |
| 20 | 80 | 4 | 0,2586 | 0,01745 | 0,2592 |
| 20 | 90 | 4,5 | 0,2416 | 0,01476 | 0,242 |
| 20 | 100 | 5 | 0,2274 | 0,0168 | 0,228 |
| 30 | 20 | 0,66667 | 0,6315 | 0,01794 | 0,6317 |
| 30 | 30 | 1 | 0,5381 | 0,005689 | 0,5382 |
| 30 | 40 | 1,33333 | 0,4715 | 0,01861 | 0,4718 |
| 30 | 50 | 1,66667 | 0,4214 | 0,02427 | 0,4221 |
| 30 | 60 | 2 | 0,3826 | 0,008532 | 0,3827 |
| 30 | 70 | 2,33333 | 0,3514 | 0,01872 | 0,3519 |
| 30 | 80 | 2,66667 | 0,326 | 0,01671 | 0,3264 |
| 30 | 90 | 3 | 0,3048 | 0,01456 | 0,3051 |
| 30 | 100 | 3,33333 | 0,2868 | 0,01143 | 0,2871 |
| 40 | 20 | 0,5 | 0,6929 | 0,006688 | 0,6929 |
| 40 | 30 | 0,75 | 0,6041 | 0,01789 | 0,6044 |
| 40 | 40 | 1 | 0,5375 | 0,02423 | 0,5381 |
| 40 | 50 | 1,25 | 0,4857 | 0,008894 | 0,4858 |
| 40 | 60 | 1,5 | 0,4443 | 0,01844 | 0,4447 |
| 40 | 70 | 1,75 | 0,4104 | 0,01685 | 0,4108 |
| 40 | 80 | 2 | 0,3822 | 0,01245 | 0,3824 |
| 40 | 90 | 2,25 | 0,3583 | 0,01137 | 0,3584 |
| 40 | 100 | 2,5 | 0,3378 | 0,003155 | 0,3378 |
| 50 | 20 | 0,4 | 0,7368 | 0,01786 | 0,737 |
| 50 | 30 | 0,6 | 0,6536 | 0,02425 | 0,6541 |
| 50 | 40 | 0,8 | 0,5889 | 0,008516 | 0,589 |
| 50 | 50 | 1 | 0,5372 | 0,0183 | 0,5375 |
| 50 | 60 | 1,2 | 0,4948 | 0,01679 | 0,4951 |
| 50 | 70 | 1,4 | 0,4596 | 0,01147 | 0,4597 |
| 50 | 80 | 1,6 | 0,4297 | 0,01139 | 0,4298 |
| 50 | 90 | 1,8 | 0,4041 | 0,003234 | 0,4041 |
| 50 | 100 | 2 | 0,3819 | 0,01835 | 0,3824 |

Table 4: Summary table for one principal component with results and parameter values.

In the following figures, the bellepoque step/memory step ratio is represented on the axis x. The following figures (6-9) graphically represent energy gain and reproduction error in relation to the ratio of the two abovementioned parameters, with the condition that the system keeps one principal component.

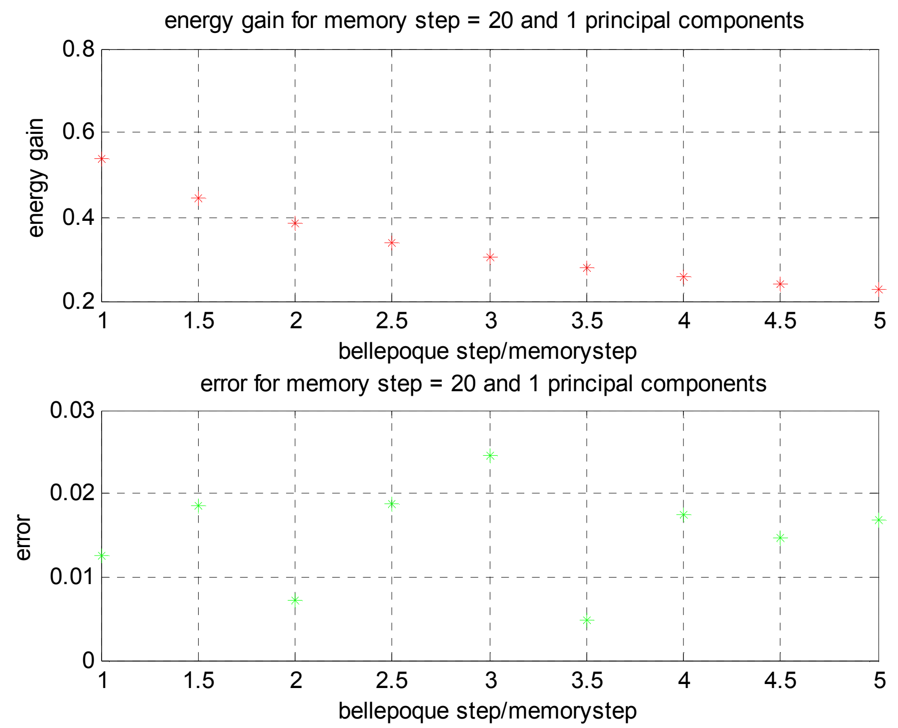


Figure 6: Energy gain and reproduction error for one principal component and memory step=20.

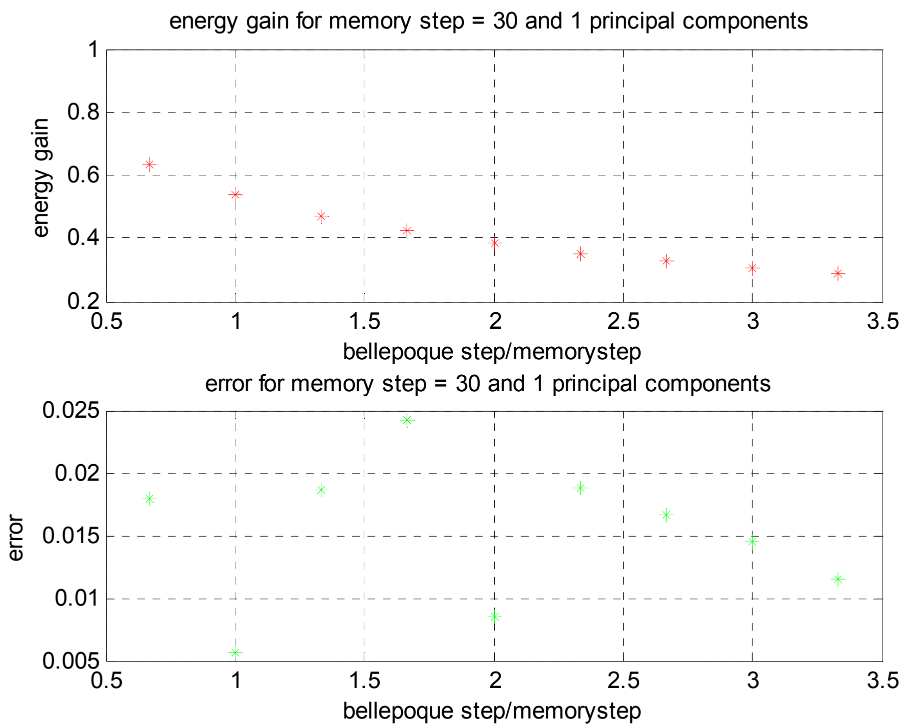


Figure 7: Energy gain and reproduction error for one principal component and memory step=30.

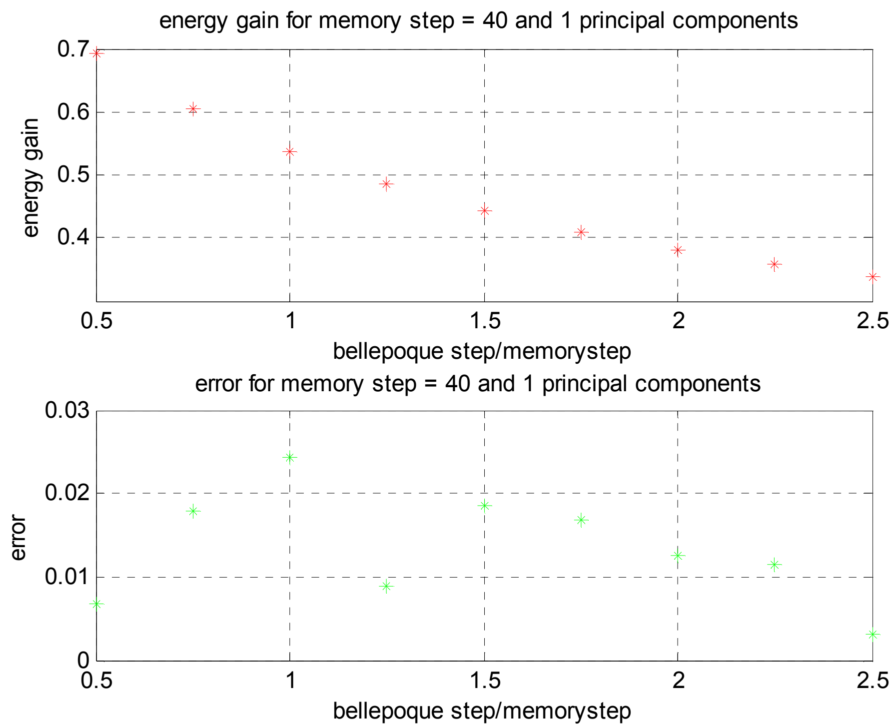


Figure 8: Energy gain and reproduction error for one principal component and memory step=40.

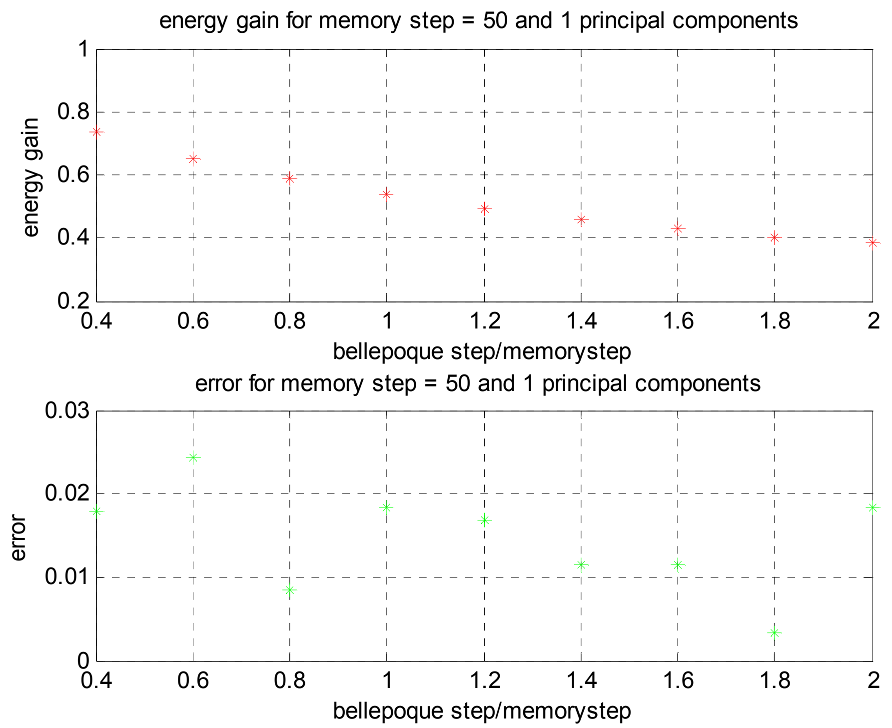


Figure 9: Energy gain and reproduction error for one principal component and memory step=50.

Figure 10 shows the values of metric H for the above values of the variables. More precisely, in x-axis we have the ratio bellepoque_step / memory_step (k / m) and in the y-axis we have the values of the metric H that correspond to the respective ratio (remember that they take values from zero to unit).

More specifically, in the graphic representation are displayed four graphs of the metric H, one for each parameter memory_step (m), that is for each run of PCA procedure for a specific value of memory_step (m). The graphical representation of the metric H is shown in relation to the ratio bellepoque_step / memory_step (k / m), as mentioned above. As shown in Table 4, each run of PCA for a specific memory_step (m) and for a different number of values of bellepoque_step (k) some values ratio are displayed, which they appear in different pairs and memory_step bellepoque_step. In brief, seeing the third column of Table 4 is understood that the same ratio appears more than once during the executions. As a result, the four graphs of H metric to overlap. For this reason, there are some minor differences in some parts of the scenic H.

The conclusions that can be exported from the graphic of the H metric are that the value of H minimizes continuously as the ratio bellepoque_step / memory_step (k / m) increases. The lowest value of H is obtained for the ratio 5, in which we have the best results in energy savings. In other words, this shows that having five times bellepoque_step (k) compared to memory_step (m) maximum power is gained, that is for a specific number of data the system can predict five times the amount of data. The largest value of H is for ratio less than 0.5, where once again saving of energy is gained, but this is not the maximum that can be achieved from the system.

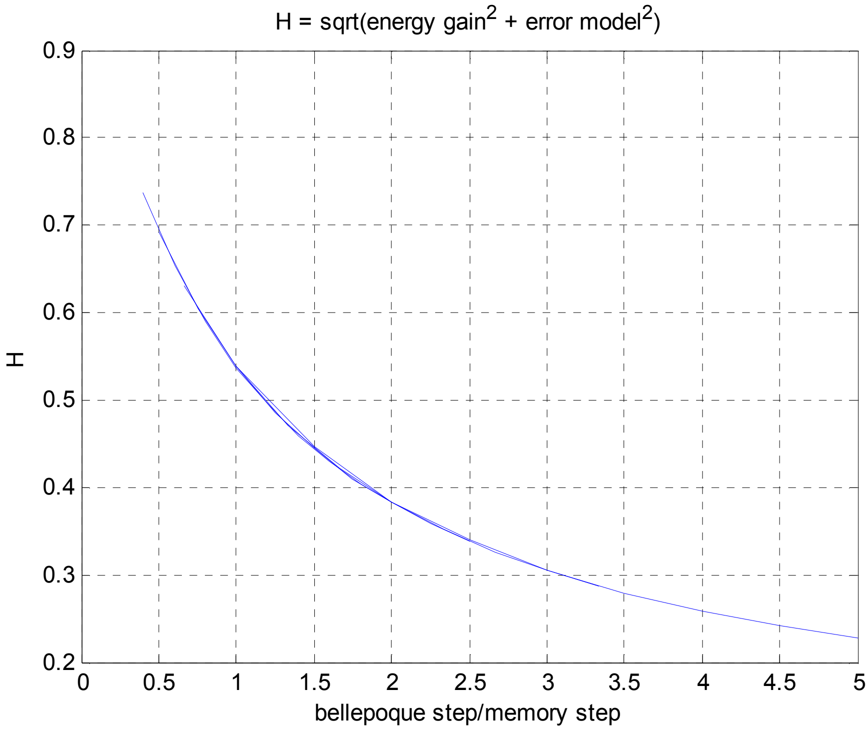


Figure 10: Metric H for one principal component.

### 4.4.3 Performance Assessment with Two Principal Components

Table 5 summarizes all the values of the results according to the parameter's values Considering that is being kept two principal components. The largest energy gain (low value of the energy_gain) is 0.2905 and is displayed for values m = 20 and k = 100, i.e. ratio (k/m) = 5. A smaller Reproduction error is 0.002967 and is displayed for values m = 50 and k = 90, i.e. ratio (k/m) = 1.8. However, the catalytic result is the value of H metric, where the smallest value of H is 0.2908, i.e.  29.08% and is exported from the values m = 20 and k = 100, i.e. ratio (k/m) = 5.

Making a similar analysis of the values of the results for two principal components, we can say that the ranges of results averaged approximately the same levels as for one principal component. In particular, the values of Reproduction error is enough small for all tests taken. The biggest error value is 0.01988 and minimum, as it was mentioned before, is 0.002967, ie generally, the rank of error is 0.29% -1.99%. It is visible, that the Energy gain receives a greater range of values.

| Memory step - m | Bellepoque step - k | Ratio - k/m | Energy gain | Error | H |
|---|---|---|---|---|---|
| 20 | 20 | 1 | 0,5857 | 0,01038 | 0,5858 |
| 20 | 30 | 1,5 | 0,4972 | 0,01821 | 0,4975 |
| 20 | 40 | 2 | 0,4381 | 0,006239 | 0,4382 |
| 20 | 50 | 2,5 | 0,3959 | 0,01938 | 0,3964 |
| 20 | 60 | 3 | 0,3642 | 0,0159 | 0,3646 |
| 20 | 70 | 3,5 | 0,3397 | 0,004919 | 0,3397 |
| 20 | 80 | 4 | 0,32 | 0,01988 | 0,3206 |
| 20 | 90 | 4,5 | 0,3039 | 0,01365 | 0,3042 |
| 20 | 100 | 5 | 0,2905 | 0,01358 | 0,2908 |
| 30 | 20 | 0,66667 | 0,6686 | 0,01808 | 0,6688 |
| 30 | 30 | 1 | 0,581 | 0,006124 | 0,581 |
| 30 | 40 | 1,33333 | 0,5184 | 0,01904 | 0,5187 |
| 30 | 50 | 1,66667 | 0,4714 | 0,01576 | 0,4717 |
| 30 | 60 | 2 | 0,4349 | 0,007069 | 0,435 |
| 30 | 70 | 2,33333 | 0,4057 | 0,0191 | 0,4062 |
| 30 | 80 | 2,66667 | 0,3818 | 0,01457 | 0,3821 |
| 30 | 90 | 3 | 0,3619 | 0,01206 | 0,3621 |
| 30 | 100 | 3,33333 | 0,3451 | 0,006424 | 0,3451 |
| 40 | 20 | 0,5 | 0,7238 | 0,005772 | 0,7239 |
| 40 | 30 | 0,75 | 0,6408 | 0,01832 | 0,6411 |
| 40 | 40 | 1 | 0,5786 | 0,01711 | 0,5789 |
| 40 | 50 | 1,25 | 0,5302 | 0,008641 | 0,5303 |
| 40 | 60 | 1,5 | 0,4915 | 0,01815 | 0,4918 |
| 40 | 70 | 1,75 | 0,4598 | 0,0151 | 0,46 |
| 40 | 80 | 2 | 0,4334 | 0,01002 | 0,4335 |
| 40 | 90 | 2,25 | 0,411 | 0,006963 | 0,4111 |
| 40 | 100 | 2,5 | 0,3919 | 0,002974 | 0,3919 |
| 50 | 20 | 0,4 | 0,7633 | 0,01742 | 0,7635 |
| 50 | 30 | 0,6 | 0,6858 | 0,0177 | 0,686 |
| 50 | 40 | 0,8 | 0,6254 | 0,008243 | 0,6255 |
| 50 | 50 | 1 | 0,5772 | 0,01832 | 0,5775 |
| 50 | 60 | 1,2 | 0,5377 | 0,015 | 0,5379 |
| 50 | 70 | 1,4 | 0,5048 | 0,008524 | 0,5049 |
| 50 | 80 | 1,6 | 0,4769 | 0,007142 | 0,477 |
| 50 | 90 | 1,8 | 0,4531 | 0,002967 | 0,4531 |
| 50 | 100 | 2 | 0,4324 | 0,01164 | 0,4326 |

Table 5: Summary table for two principal component, with results and the parameter values.

That is, the maximum value of the energy gain (where we have little energy saving) is 0.7633 and the minimum is 0.2274, which means that in percentage is approximately 76%-29%. Finally, the metric H is a function of error and energy gain, and it takes a wide range of values too. The largest H value (which indicates that there are no good results) is 0.7635 and the minimum is 0.2908 which indicates that in these parameter values we have the best results. The range of the values of H is approximately 76%-29%. Therefore, the ideal values of these parameters are m = 20 and k = 100, ie ratio = 5 (k / m).

In figures 11 to 14, the energy gain and reproduction error are presented in relation with the bellepoque and memory step ratio, this time by keeping two principal components. Additionally, Figure 15 presents metric H for the same amount of principal components. And in this case, we have exactly the same results compared to the graphical representation of H for the one principal component. That is, maximum and minimum values of H are achieved on the same ratios k/m, but the only difference is that the values of H are a bit higher.

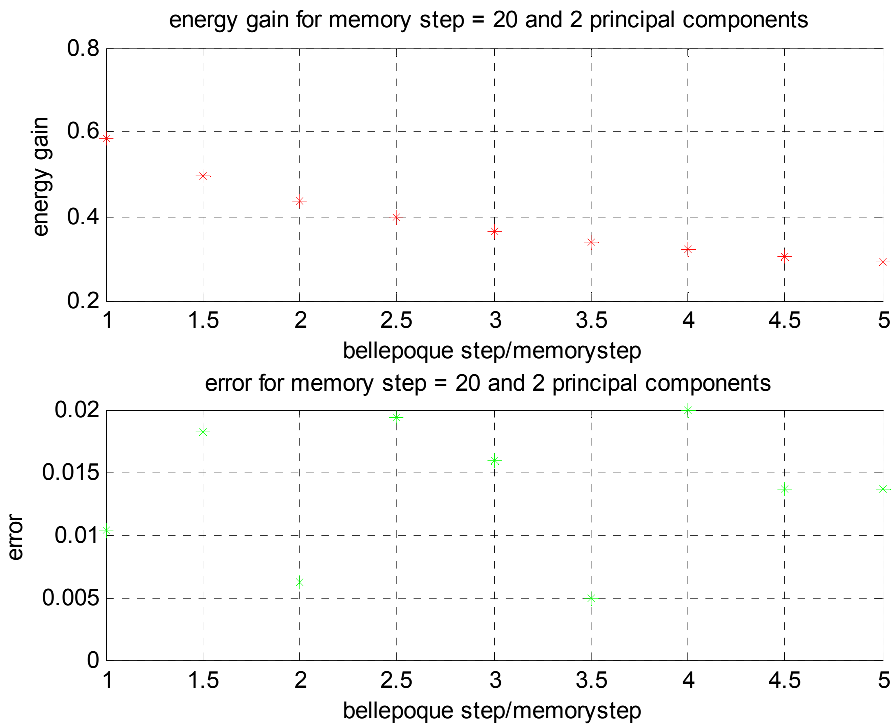


Figure 11: Energy gain and reproduction error for two principal component and memory step=20.

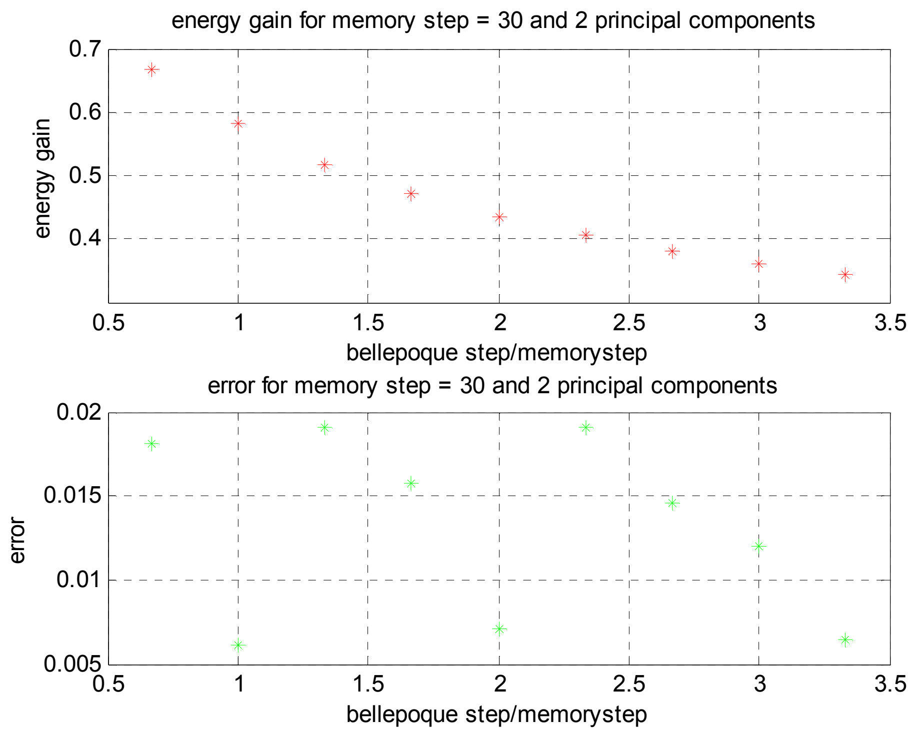


Figure 12: Energy gain and reproduction error for two principal component and memory step=30.

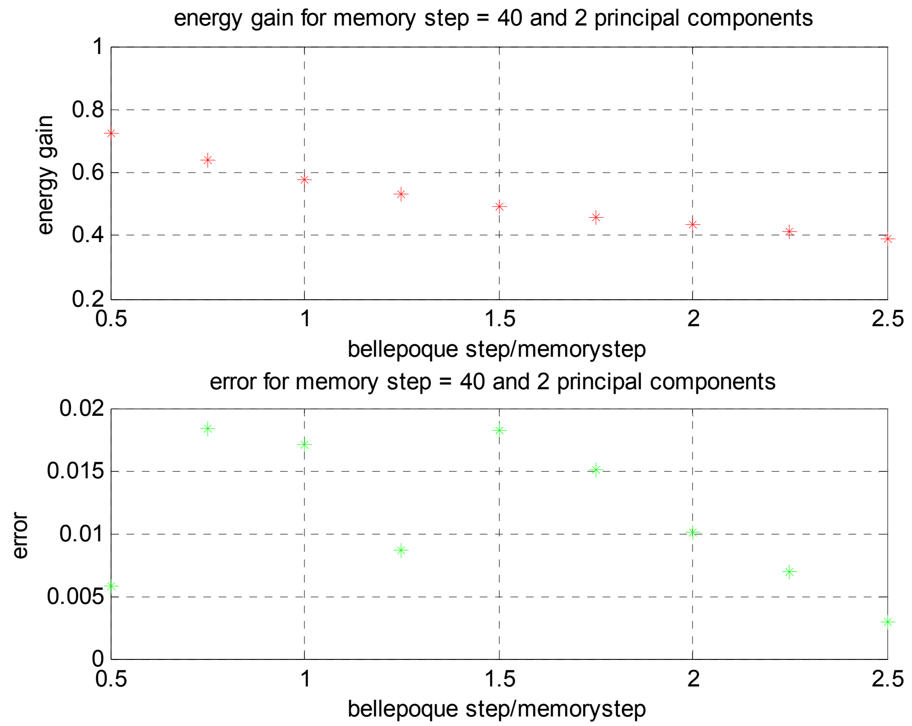


Figure 13: Energy gain and reproduction error for two principal component and memory step=40.

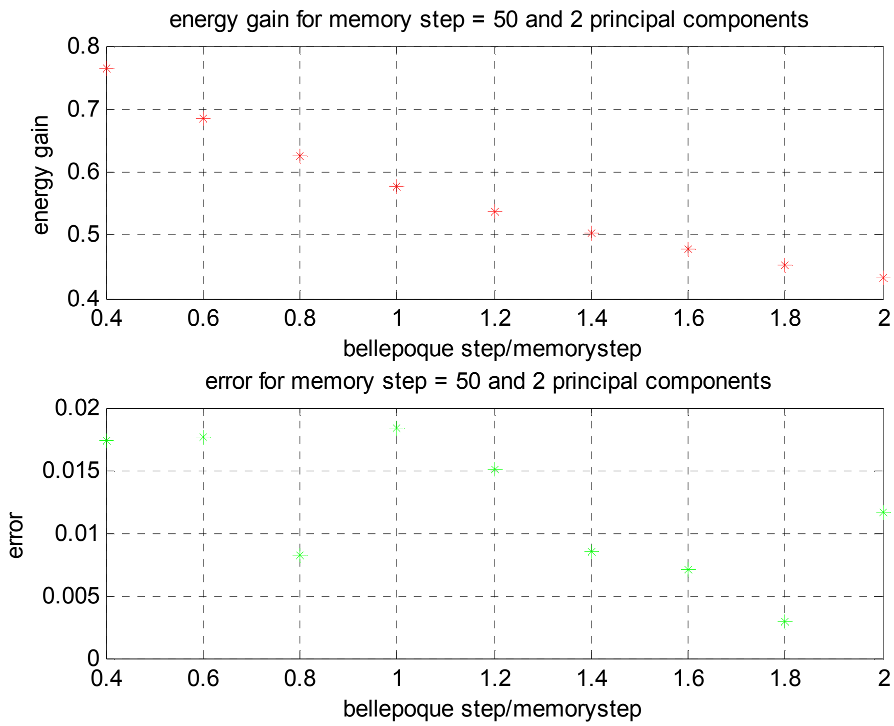


Figure 14: Energy gain and reproduction error for two principal component and memory step=50.

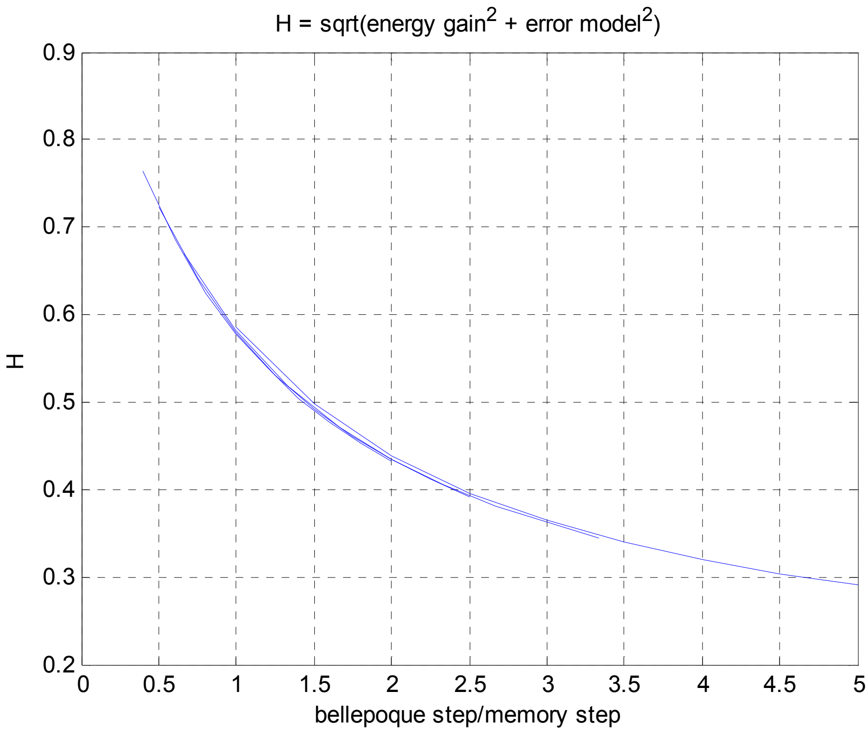


Figure 15: Metric H for two principal components.

In order to understand further the results we can compare them to one principal component and two principal components. Initially it seems that for one principal component we have smaller values of the metrics H in relation to these two principal components. Do recall that the smaller the value of the metrics H the greater the data compression is, which means that we gain more energy. The reproduction of one principal component error is greater than the reproduction error of two principal components. This makes sense, since keeping one principal component there is higher compression and more losses, compared to the two principal components. As for the energy gain, there is greater energy conservation when PCA is being performed for one principal component (smaller values of energy_gain) in relation to energy saving for two components.

This is reasonable too, since the higher compression the one principal component achieves implies to the fact that less number of bit are sent, so there is less energy consumption. Subsequently to mention that the price ranges of energy_gain, error and H vary to the same level in both cases. And finally, comparing the best values of the results, going from one principal component in two principal components is:

for metric H, increasing 6.28%

for reproduction error, reducing 0.0188%

and for the energy_gain, increasing 6.31%.

# CHAPTER 5

# 5. Conclusions

Contextual data collected by a WSN is a challenging problem. Due to very limited resources, e.g., energy, computational power, data storage and bandwidth, it is not a sound technical decision to apply strategies like forwarding any sensor data directly to a sink that does the corresponding processing. Given the resource constraints in the underlying network infrastructure (e.g., a wireless sensor network), we try to exploit the vector nature of the exchanged contextual information and the possible statistical dependencies between the different contextual (information) components. Our compression scheme capitalizes on the above assumptions which are frequently encountered in real life applications and integrated systems. The proposed context compression is accomplished by mapping–projecting (prior to transmission) the generated context vectors to a reduced dimensions space defined on the principal components (PC) basis. In such basis, contextual data are uncorrelated. The basis is obtained by adopting the Principal Component Analysis (PCA), a well known technique in multivariate data analysis. Reducing the dimensions of contextual vectors is catalytic to the compression task. Practically many of the contextual components are described through the PCA–identified principal components and eliminated from the context reporting messages. We propose, implement and evaluate a Principal Components-based Context Forwarding model that improves the forwarding efficiency of context in a wireless sensor network (WSN) based on the current principal components. The model presents two important benefits: the PCA provides varying levels of context compression accuracies, ranging from constant approximations to the full recovery of the original context; principal components contain sufficient information for re–producing the original data for further processing/ interpretation or forwarding. The model adopts a periodic calculation of the principal components. The exact PCA scheduling is based on a short history of context vectors, the induced re–production error and error change. This periodic calculation affects the context transmission process and the overall context accuracy.

The basic idea of the model is that for any m number of data PCA is being performed on initial node, the m data are sent to the final node as they are, but also the eigenvector matrix P is sent which resulting from the PCA. For the next k number of data, PCA is performed in reverse order, on the final node, given the eigenvector matrix P of previous m data and the matrix of average values of k data, which is the only item that has been sent from the initial node to the final. So in this way can provide k data from the m data which have been sent, having just sent the average values of the components of k data.

In our model some configuration should be made in order to extract the best results. These parameters are: the q number of principal components that the model will keep from the PCA process, the m number of the data from which the eigenvector matrix P will be exported and the number k of provided data.

In order to understand the results of the model, we created a mathematical metric H which is exported from the energy gain and the reproduction error, which are calculated in comparison with those of the SDF model. This metric takes values from zero to one, and the metric is close to zero in the ideal results.

According to the tests performed, there were better results when the number k was five times the number m and the number of principal components were one. When tests were performed for two principal components the metric was increased approximately 6%.

# ANNEX

## Code in Matlab

main.m

```
close all;
clear all;
clc;
load clear_sensor_values.txt;
A = clear_sensor_values + 1;
% load nineparams.txt;
% A = nineparams +1;

%bellepoque = 60    %k
%memorystep = 20     %m
MINm = 20;
MAXm = 50;
MINk = MINm;
MAXk=100;
n = length(A(1,:));
q = 2;              %q
H = [];
for  memorystep=MINm:10:MAXm
   figure

   for bellepoque = MINk:10:MAXk
      if (floor(length(A(:,1))/(memorystep+bellepoque)) == 0)
         break;
      end
      [error_model cost_model cost_classic] = PCADataTransmissionModel(memorystep,
bellepoque, n, q, A);
      error_model;
      cost_model;
      cost_classic;
      energy_gain=cost_model/cost_classic;
      H = [H sqrt(energy_gain^2 + error_model^2)];

      subplot(2,1,1); plot(bellepoque/memorystep,energy_gain,'r*')
      title(['energy gain for memory step = ',num2str(memorystep), ' and ',num2str(q), '
principal components ']);
      ylabel('energy gain');
      xlabel('bellepoque step/memorystep');
      hold on;
      grid on;

      subplot(2,1,2); plot(bellepoque/memorystep,error_model,'g*')
      title(['error for memory step = ',num2str(memorystep), ' and ',num2str(q), ' principal
components ']);
      ylabel('error');
      xlabel('bellepoque step/memorystep');

      hold on;
      grid on;
   end
  % subplot(3,1,3); plot(H)
```

```
    %grid on;
    pause();
    hold off;
end
i = (MAXm - MINm)/10 + 1
j = (MAXk - MINk)/10 + 1
x=1;
y= j;
figure
for o=1:i
    plot(H(x:y))
    title(['H = sqrt(energy gain^2 + error model^2)']);
    ylabel('H');
    xlabel('');
    %xlabel('bellepoque step');

    grid on;
    hold on;
    y = y + j;
    x = x + j;
    pause();
end
```

## PCADataTransmissionModel.m

```
function [error_model cost_model cost_classic] = PCADataTransmissionModel(m, k, n, q, A)
starter = 1;
memorer = m;
Diff = [];
%hold;
%scheme is: memory for learning and bellepoque+1 for prediction
maxloop = floor(length(A(:,1))/(m+k));%8;
nofprincomp = q; %maximum number of principal components
for loop = 1:maxloop % for (9,10 => 52), for (39,40 => 12), for (50,40 => 11), for (50,20 =>
14),
    memorer;
    starter;
    X =A(starter:memorer,:); %[X1(starter:memorer) X2(starter:memorer) X3(starter:memorer)
X4(starter:memorer) X5(starter:memorer) X6(starter:memorer) X7(starter:memorer)
X8(starter:memorer) X9(starter:memorer)];

    [reducedX reducedP reducedY meanX P Y]= reduceparameter(X,nofprincomp);
    %stem3(X(:,1),X(:,2),X(:,3),'o');
    %stem3(reducedX(:,1),reducedX(:,2),reducedX(:,3),'r*');
    %the step is two measurements
    step = 2;
    meamemoryvalue = repmat(mean(X),step,1);
    for i=1:step:k
        %the step is two measurements
        index = memorer + i;
        if (index+1>length(A(:,1)))
            break;
        end
        Xp = A(index:index+1,:); % [X1(index:index+1) X2(index:index+1) X3(index:index+1)
X4(index:index+1) X5(index:index+1) X6(index:index+1) X7(index:index+1)
X8(index:index+1) X9(index:index+1)];

        meanXp = Xp - repmat(mean(Xp),size(Xp,1),1);
        Yp = meanXp*P;
        rXp = Yp(:,1:nofprincomp)*P(:,1:nofprincomp)' + repmat(mean(Xp),size(Xp,1),1);
        Diff = [Diff; abs(((Xp-rXp)./(Xp)))];
```

```
    end
    starter = starter + m + k;
    memorer = m + starter;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%cost
[cost_classic cost_model ]=costPCA(m,k,n,q);
cost_model = cost_model * maxloop;
cost_classic = cost_classic * maxloop;
error_model=mean(mean(abs(((Xp-rXp)./Xp))));


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

### reduceparameter.m

```
function [reducedX reducedP reducedY meanX P Y]= reduceparameter(X,q)
%P: eigenvector sorted by relevance from the highest to lowest
%Y: the transformed data (pure of mean) on the eigenvector axes
[P, Y, eigValues] = princomp(X);
%meanX*P == Y
%reduce Y w.r.t. the highest 1st component
reducedY = Y(:,1:q);
%get the 1st component
reducedP = P(:,1:q);
%reconstruct of the original data w.r.t. transformation
reducedX = reducedY*reducedP' + repmat(mean(X),size(X,1),1);
meanX = repmat(mean(X),size(X,1),1);
```

### costPCA.m

```
function [ classic_schema_Cost advanced_schema_Cost ] = costPCA (m,k,n,q)
%PCA complexity
PCAComplexity = (m^(1.5)) + m;
%cost per PCA instruction
PCAcost = PCAComplexity * 4 * (10^-9);
%cost per transmission
costPerTransmition = 830 * (10^-9);
%data load for a value
dataLoad = 11*8;

%accumulative statistics for cost energy on the specific Bellepoque
classic_schema = n*(k+m);    %/*gia n diastaseis*/
advanced_schema = n*m + q*k*0.5 + q*q;  %/*gia q epilegmenes principal components*/

classic_schema_Cost = classic_schema * (costPerTransmition*dataLoad);
advanced_schema_Cost = advanced_schema * (costPerTransmition*dataLoad) + PCAcost;
```

## Data

```
14.4000   64.2000   17.4000   55.5000   15.0000   65.8000    0.3300
15.2000   59.8000   20.7000   44.9000   15.2000   61.7000    0.4200
15.7000   56.2000   17.9000   50.7000   15.3000   63.2000    0.6000
15.8000   61.0000   16.3000   53.5000   15.5000   59.2000    1.0400
15.5000   56.1000   16.3000   53.5000   16.3000   55.0000    0.4500
15.5000   55.2000   18.9000   46.1000   15.8000   50.9000    0.7700
15.6000   56.8000   16.6000   50.3000   15.5000   52.7000    0.4300
….
```

# REFERENCES

[1] G.D. Abowd, J.P.G. Sterbenz, Final report on the interagency workshop on research issues for smart environments, IEEE Personal Communications (October 2000) 36–40.
[2] P. Bauer, M. Sichitiu, R. Istepanian, K. Premaratne, The mobile patient: wireless distributed sensor networks for patient monitoring and care, Proceedings 2000 IEEE EMBS International Conference on Information Technology Applications in Biomedicine, 2000, pp. 17–21.
[3] P. Bonnet, J. Gehrke, P. Seshadri, Querying the physical world, IEEE Personal Communications (October 2000) 10–15.
[4] N. Bulusu, D. Estrin, L. Girod, J. Heidemann, Scalable coordination for wireless sensor networks: self-configuring localization systems, International Symposium on Communication Theory and Applications (ISCTA 2001), Ambleside, UK, July 2001.
[5] A. Cerpa, J. Elson, M. Hamilton, J. Zhao, Habitat monitoring: application driver for wireless communications technology, ACM SIGCOMM'2000, Costa Rica, April 2001.
[6] A. Chandrakasan, R. Amirtharajah, S. Cho, J. Goodman, G. Konduri, J. Kulik, W. Rabiner, A. Wang, Design considerations for distributed micro-sensor systems, Proceedings of the IEEE 1999 Custom Integrated Circuits
Conference, San Diego, CA, May 1999, pp. 279–286.
[7] S. Cho, A. Chandrakasan, Energy-efficient protocols for low duty cycle wireless microsensor, Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, Maui, HI Vol. 2 (2000), p. 10.
[8] R. Colwell, Testimony of Dr. Rita Colwell, Director, National Science Foundation, Before the Basic Research Subcommitte, House Science Committe, Hearing on Remote Sensing as a Research and Management Tool, September 1998.
[9] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, B. Krogh, "Energy-efficient surveillance system using wireless sensor networks", in Proc. of 2nd Intl. Conf. on Mobile Systems, Applications, and Services, (ACM MobiSys '04), pp.270- 283, June 2004.
[10] D. Estrin, R. Govindan, J. Heidemann, S. Kumar, Next century challenges: scalable coordination in sensor networks, ACM MobiCom'99, Washingtion, USA, 1999, pp. 263–270.
[11] C. Herring, S. Kaplan, Component-based software systems for smart environments, IEEE Personal Communications, October 2000, pp. 60–61.
[12] T. Imielinski, S. Goel, DataSpace: querying and monitoring deeply networked collections in physical space, ACM International Workshop on Data Engineering for Wireless and Mobile Access MobiDE 1999, Seattle, Washington, 1999, pp. 44–51.
[13] C. Intanagonwiwat, R. Govindan, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, Proceedings of the ACM Mobi- Com'00, Boston, MA, 2000, pp. 56–67.
[14] T.H. Keitt, D.L. Urban, B.T. Milne, Detecting critical scales in fragmented landscapes, Conservation Ecology 1 (1) (1997) 4. Available from <http://www.consecolo.org/vol1/iss1/art4>.
[15] National Semiconductor Corporation, LMX3162 Single Chip Radio Transceiver, Evaluation Notes and Datasheet, March 2000.
[16] N. Noury, T. Herve, V. Rialle, G. Virone, E. Mercier, G. Morey, A. Moro, T. Porcheron, Monitoring behavior in home using a smart fall sensor, IEEE-EMBS Special Topic Conference on Microtechnologies in Medicine and Biology, October 2000, pp. 607–610.
[17] N. Priyantha, A. Chakraborty, H. Balakrishnan, The cricket location-support system, Proceedings of ACM MobiCom'00, August 2000, pp. 32–43.
[18] E.M. Petriu, N.D. Georganas, D.C. Petriu, D. Makrakis, V.Z. Groza, Sensor-based information appliances, IEEE Instrumentation and Measurement Magazine (December 2000) 31–35.
[19] G.J. Pottie, W.J. Kaiser, Wireless integrated network sensors, Communications of the ACM 43 (5) (2000) 551– 558.
[20] J. Rabaey, J. Ammer, J.L. da Silva Jr., D. Patel, Pico- Radio: ad-hoc wireless networking of ubiquitous lowenergy sensor/monitor nodes, Proceedings of the IEEE Computer Society Annual Workshop on VLSI (WVLSI'00), Orlanda, Florida, April 2000, pp. 9–12.
[21] J.M. Rabaey, M.J. Ammer, J.L. da Silva Jr., D. Patel, S. Roundy, PicoRadio supports ad hoc ultra-low power wireless networking, IEEE Computer Magazine (2000) 42–48.

[22] E. Shih, S. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, A. Chandrakasan, Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks, Proceedings of ACM MobiCom'01, Rome, Italy, July 2001, pp. 272–286.

[23] B. Sibbald, Use computerized systems to cut adverse drug events: report, CMAJ: Canadian Medical Association Journal 164 (13) (2001) 1878, 1/2p, 1c.
[24] B. Walker, W. Steffen, An overview of the implications of global change of natural and managed terrestrial ecosystems, Conservation Ecology 1 (2) (1997). Available from <http://www.consecol.org/vol1/iss2/art2>.
[25] http://www.alertsystems.org
[26] http://www.utdallas.edu/~herve/abdi-wireCS-PCA2010-inpress.pdf
[27] X. Jiang, J. Taneja, J. Ortiz, A. Tavakoli, P. Dutta, J. Jeong, D. Culler, P. Levis, and S. Shenker. An architecture for energy management in wireless sensor networks. SIGBED Rev., 4(3):31–36, 2007.
[28] A. Porret, T. Melly, C.C. Enz, E.A. Vittoz, A low-power low-voltage transceiver architecture suitable for wireless distributed sensors network, IEEE International Symposium on Circuits and Systems'00, Geneva, Vol. 1, 2000, pp. 56–59.
[29] R. Min, T. Furrer, A. Chandrakasan, Dynamic voltage scaling techniques for distributed microsensor networks, Proceedings of ACM MobiCom'95, August 1995.
[30] T. Pering, T. Burd, R. Brodersen, The simulation and evaluation of dynamic voltage scaling algorithms, Proceedings of International Symposium on Low Power Electronics and Design ISLPED'98, August 1998, pp. 76–81.
[31] C. Perkins, Ad Hoc Networks, Addison-Wesley, Reading, MA, 2000.
[32] S. Vardhan, M. Wilczynski, G. Pottie, W.J. Kaiser, Wireless integrated network sensors (WINS): distributed in situ sensing for mission and flight systems, IEEE Aerospace Conference, Vol. 7, 2000, pp. 459–463.
[33] A. Woo, D. Culler, A transmission control scheme for media access in sensor networks, Proceedings of ACM MobiCom'01, Rome, Italy, July 2001, pp. 221–235