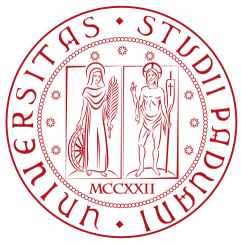


UNIVERSITY OF PADOVA

Department of Information Engineering

---

Ph.D. School in Information Engineering

Information Science and Technology

XXV Class

# Design and Evaluation of Compression, Classification and Localization Schemes for Various IoT Applications

*Ph.D. candidate:*

Matteo DANIELETTO

*Supervisor:*

Prof. Michele ZORZI

*Course coordinator:*

Prof. Carlo Ferrari

*Ph.D. School director:*

Prof. Matteo BERTOCCO

---

Academic Year 2013-2014

# Abstract

Nowadays we are surrounded by a huge number of objects able to communicate, read information such as temperature, light or humidity, and infer new information through exchanging data. These kinds of objects are not limited to high-tech devices, such as desktop PC, laptop, new generation mobile phone, i.e. smart phone, and others with high capabilities, but also include commonly used object, such as ID cards, driver license, clocks, etc. that can made smart by allowing them to communicate.

Thus, the analog world of just a few years ago is becoming the a digital world of the Internet of Things (IoT), where the information from a single object can be retrieved from the Internet. The IoT paradigm opens several architectural challenges, including self-organization, self-managing, self-deployment of the smart objects, as well as the problem of how to minimize the usage of the limited resources of each device. The concept of IoT covers a lot of communication paradigms such as WiFi, Radio Frequency Identification (RFID), and Wireless Sensor Network (WSN). Each paradigm can be thought of as an IoT island where each device can communicate directly with other devices.

The thesis is divided in sections in order to cover each problem mentioned above. The first step is to understand the possibility to infer new knowledge from the deployed device in a scenario. For this reason, the research is focused on the web semantic, web 3.0, to assign a semantic meaning to each thing inside the architecture. The sole semantic concept is unusable to infer new information from the data gathered; in fact, it is necessary to organize the data through a hierarchical form defined by an Ontology. Through the exploitation of the Ontology, it is possible to apply semantic engine reasoners to infer new knowledge about the network.

The second step of the dissertation deals with the minimization of the usage of every node in a WSN. The main purpose of each node is to collect environmental data and to ex-

change them with other nodes. To minimize battery consumption, it is necessary to limit the radio usage. Therefore, we implemented Razor, a new lightweight algorithm which is expected to improve data compression and classification by leveraging on the advantages offered by data mining methods for optimizing communications and by enhancing information transmission to simplify data classification. Data compression is performed studying the well-know Vector Quantization (VQ) theory in order to create the codebooks necessary for signal compression. At the same time, it is requested to give a semantic meaning to unknown signals. In this way, the codebook feature is able not only to compress the signals, but also to classify unknown signals. Razor is compared with both state-of-the-art compression and signal classification techniques for WSN .

The third part of the thesis covers the concept of smart object applied to Robotic research. A critical issue is how a robot can localize and retrieve smart objects in a real scenario without any prior knowledge. In order to achieve the objectives, it is possible to exploit the smart object concept and localize them through RSSI measurements. After the localization phase, the robot can exploit its own camera to retrieve the objects. Several filtering algorithms are developed in order to mitigate the multi–path issue due to the wireless communication channel and to achieve a better distance estimation through the RSSI measurement.

The last part of the dissertation deals with the design and the development of a Cognitive Network (CN) testbed using off the shelf devices. The device type is chosen considering the cost, usability, configurability, mobility and possibility to modify the Operating System (OS) source code. Thus, the best choice is to select some devices based on Linux kernel as Android OS. The feature to modify the Operating System is required to extract the TCP/IP protocol stack parameters for the CN paradigm. It is necessary to monitor the network status in real-time and to modify the critical parameters in order to improve some performance, such as bandwidth consumption, number of hops to exchange the data, and throughput.

# Sommario

In questi ultimi anni siamo circondati da una grande quantità di oggetti che sono capaci di comunicare tra di loro e leggere in tempo reale grandezze fisiche come temperatura, luce e umidità. Un passo successivo consisterà nel combinare il contenuto informativo di queste grandezze fisiche per estrarre ulteriore informazione non osservabile analizzando il singolo dato.

Questi oggetti non sono solamente hi-tech, come PC da tavolo, PC portatili, tablet, telefoni mobili di ultima generazione, i.e., smartphone, e altri con elevate capacità, ma soprattutto quegli oggetti di uso comune, come carte d'identità, patenti di guida, orologi, che possono essere connessi ad Internet semplicemente applicando micro dispositivi di comunicazione. Così, del mondo analogico di qualche anno fa, ci si sta sempre più spostando verso il mondo digitale dell'Internet of Things (IoT), dove le informazioni di ogni singolo oggetto possono essere cercate in Internet. Il paradigma IoT pone a diverse sfide architetturali, come auto organizzazione, auto controllo e auto disposizione degli smart object. Inoltre esiste il problema di come minimizzare l'utilizzo delle risorse limitate di ogni dispositivo. Il concetto di IoT ricopre molti paradigmi di comunicazione, che possono essere elencati come reti WiFi, Radio Frequency Identificator (RFID), e reti di sensori senza fili (WSN). Ogni paradigma sopra elencato può essere pensato come un'isola dell'architettura IoT dove ogni dispositivo all'interno di essa è capace di comunicare con gli altri dispositivi.

La tesi è divisa in diverse sezioni per ricoprire alcune problematiche menzionate poc'anzi. Il primo argomento trattato dalla tesi riguarda la possibilità di inferire nuova conoscenza dai nodi disposti all'interno di uno scenario. Per questa ragione, la ricerca si è focalizzata sul web semantico, web 3.0, dove si assegna un significato semantico ad ogni "thing" che si trova all'interno dell'architettura. Il solo concetto di semanticità è inutilizzabile ai fini di estrarre nuove informazioni, ma deve essere organizzato secondo una struttura chiamata

Ontologia. Con l'ausilio del concetto di ontologia, è possibile applicare ragionatori ontologici per l'inferenza di nuova informazione da quella già esistente.

Il secondo passo della tesi è stato minimizzare l'utilizzo complessivo di ogni nodo presente all'interno di una WSN. Partendo dall'idea che ogni nodo è capace di collezionare dati ambientali e scambiare questo tipo d'informazione con altri, questo comporta un consumo energetico non "ammissibile" per dispositivi che devono avere un tempo di vita quasi "illimitato". Un punto chiave per aumentare il tempo di vita è minimizzare il consumo energetico di ogni dispositivo. Per questo motivo è stato concepito RAZOR, un algoritmo leggero capace di migliorare la compressione e la classificazione dei dati basandosi sui vantaggi offerti dai metodi di data mining per comunicazioni, ottimizzando e potenziando la trasmissione dell'informazione per la classificazione dei dati. La parte di teoria riguardante la compressione dei dati è basata sulla quantizzazione vettoriale (VQ) per la creazione di dizionari di codifica necessari a RAZOR nella fase di compressione dei segnali. Allo stesso tempo è richiesto di dare un significato semantico ai segnali per poi poter classificare segnali sconosciuti. RAZOR è stato comparato con lo stato dell'arte della compressione e classificazione di segnali.

Un terzo passo della tesi riguarda il concetto di smart object applicato al campo della robotica. Un punto cruciale sta nella possibilità di utilizzare gli smart object per creare un'architettura hardware/software dove non è necessaria la presenza di un operatore umano per il suo funzionamento. Un esempio può essere il riconoscimento e recupero di smart object tramite l'utilizzo di un robot completamente autonomo. Durante l'inizializzazione del sistema, robot e smart objects non avranno nessuna conoscenza dello scenario che li circonda. Un'idea consiste nell'utilizzare dispositivi poco costosi, come i nodi sensore, da applicare ad ogni oggetto e al robot, per poi strutturare la loro capacità di comunicazione wireless per essere localizzati dal robot. Dopo che gli oggetti sono stati localizzati, il robot utilizza la propria videocamera per riconoscere e recuperare l'oggetto. Per ottenere una localizzazione attendibile tramite misure di RSSI sono stati realizzati e comparati diversi algoritmi di stima e filtraggio per ovviare a problemi di multi–path che si verificano durante la comunicazioni wireless.

L'ultima parte della tesi riguarda la progettazione e lo sviluppo di un Cognite Network (CN) testbed realizzato con dispositivi commerciali. La scelta dei dispositivi è stata fatta

considerando determinati punti chiave come: costo, usabilità, configurabilità, mobilità e possibilità di modificare il Sistema Operativo (OS). Perció la migliore scelta è stata quella di utilizzare dispositivi basati su kernel Linux come Android OS. Un sistema operativo modificabile è la chiave per realizzare il CN testbed perché dà la possibilità di estrarre i parametri dello stack protocollare TCP/IP per migliorarne le prestazioni come consumo della banda, numero di passaggi che un segmento deve fare per arrivare al nodo destinazione e il throughtput di una connessione punto a punto.

# List of Acronyms

| | |
|---|---|
| **6LoWPAN** | IPv6 over Lowpower Wireless Personal Area Network |
| **AdaBoost** | Adaptive Boosting |
| **ADC** | Analog to Digital Converter |
| **ACK** | Acknowledgment |
| **AIFS** | Arbitration Inter-Frame Spacing |
| **AIMD** | Additive Increase Multiplicative Decrease |
| **AODV** | Ad hoc On Demand Vector routing |
| **API** | Application Programmable Interface |
| **AV** | Actual Value |
| **AWMN** | Android Wireless Mesh Network |
| **BATMAN** | Better Approach for Mobile Ad hoc Network |
| **BCI** | Brain Computer Interface |
| **BER** | Bit Error Rate |
| **BF** | Brute Force |
| **BSC** | Binary Stream cipher Carrier |
| **BSS** | Basic Service Set |
| **BoW** | Bag of word |

| | |
|---|---|
| **CASM** | Coherent Adaptive Subcarrier Modulation |
| **CDM** | Compression-based Dissimilarity Measure |
| **CPU** | Central Process Unit |
| **CSIRO** | Commonwealth Scientific and Industrial Research Organisation |
| **CSMA** | Carrier Sense Multiple Access |
| **CN** | Cognitive Network |
| **CR** | Cognitive Radio |
| **CWmax** | Contention Window max |
| **CWmin** | Contention Window min |
| **CWND** | Congestion Window |
| **DCT** | Discrete Cosine Transform |
| **DL** | Description Language |
| **DLL** | DataLink Layer |
| **DM** | Distance Matrix |
| **DTW** | Dynamic Time Warping |
| **ECGF** | Encrypted Code Generation Facility |
| **EKF** | Extended Kalman Filter |
| **ETX** | Expected Transmission |
| **FPGA** | Field Programmable Gate Array |
| **GNSS** | Global Navigation Satellite System |
| **GNU** | GNU Not Unix |
| **GPS** | Global Positioning System |

| | |
|---|---|
| **IBSS** | Independent Basic Service Set |
| **ID** | Identifier |
| **IEEE** | Institute of Electrical and Electronics Engineers |
| **iid** | Independent and Identically Distributed |
| **IP** | Internet Protocol |
| **IPv6** | Internet Protocol version 6 |
| **IoT** | Internet of Things |
| **IoT-A** | Internet of Thing Architecture |
| **KMF** | Key Management Facility |
| **LOS** | Line Of Sight |
| **LQH** | Link Quality Hello |
| **LTC** | Lightweight Temporal Compression |
| **LTE** | Long Term Evolution |
| **M2M** | Machine to Machine |
| **MAC** | Medium Access Control |
| **MAI** | Multiple Access Interface |
| **MB** | Matching Based |
| **MDS** | Multi Dimensional Scaling |
| **MGF** | Message Generation Facility |
| **MM** | Matching Matrix |
| **MoBIF** | Motion Bluer Invariant Feature transform |
| **NS** | Normal Shape |

| | |
|---|---|
| **NDK** | Native Development Kit |
| **NL** | Network Link |
| **OLSR** | Optimized Link State Routing |
| **OLSRD** | Optimized Link State Routing Daemon |
| **ORBIT** | Open-Access Research Testbed for Next-Generation Wireless Networks |
| **OS** | Operating System |
| **OWL** | Ontology Web Language |
| **OWL2** | Ontology Web Language version 2 |
| **OWLAPI** | Ontology Web Language API |
| **PEIS** | Physically Embedded Intelligent System |
| **PF** | Particle Filter |
| **PHY** | Physical layer |
| **PN** | PseudoNoise |
| **RAM** | Random Access Memory |
| **RANSAC** | Random Sample Consensus |
| **RDF** | Resource Description Framework Schema |
| **RFID** | Radio Frequency Identifier |
| **ROC** | Receiver Operation Characreristic |
| **ROM** | Read Only Memory |
| **RSS** | Received Signal Strength |
| **RSSI** | Received Signal Strength Indicator |
| **RTO** | Retransmission TimeOut |

| | |
|---|---|
| **RTT** | Round Trip Time |
| **SA** | Selective Availability |
| **SAS** | Signal Authentication Sequence |
| **SAX** | Symbolic Aggregate approXimation |
| **SCR** | Security Code Replica |
| **SDR** | Software Define Radio |
| **SDK** | Software Development Kit |
| **SLAM** | Simultaneous Localization And Mapping |
| **SIFT** | Scale Invariant Feature Transform |
| **SIR** | Sampling Importance Resampling |
| **SNR** | Signal to Noise Ratio |
| **SSSC** | Spread Spectrum Security Codes |
| **SSTHR** | Slow Start Threshold |
| **SUMO** | Suggested Upper Merged Ontology |
| **SRTT** | Smoothed Round Trip Time |
| **SWRL** | Semantic Web Rule Language |
| **TCP** | Transmission Control Protocol |
| **TDMA** | Time Division Multiple Access |
| **TOW** | Time Of Week |
| **TTA** | Time To Alert |
| **URI** | Unique Resource Identifier |
| **UMTS** | Universal Mobile Telecommunication System |

| | |
|---|---|
| **USRP** | Universal Software Radio Peripheral |
| **VT-CoRNET** | Virginia Technology Cognitive Network |
| **WSN** | Wireless Sensor Network |
| **WS&AN** | Wireless Sensors and Actuators Network |
| **WS&RN** | Wireless Sensors and Robotic Network |
| **XML** | eXtensible Markup Language |

# Contents

Chapter 1

# Introduction

Every day, hundreds of millions of people connect heterogeneous devices such as Personal Computer (PC), Smartphone, tablet, SmartTV for a pletora of purposes, e.g., browsing the Internet, sharing multimedia data, video streaming, chatting with other people and so on. Most of the times those devices can exchange data through the IEEE 802.11 standard while being under WiFi coverage. Otherwise, devices with a radio mobile transceiver can communicate using either Universal Mobile Telecommunication System (UMTS) or Long Term Evolution (LTE), two standards of high-speed data wireless communication for mobile phones and data terminals. In addition, these devices run an Operating System (OS) with the features similar to a standard OS running on common PCs.

Although they are qualified as *smart*, actually these devices are not entirely *smart* because a device's owner cannot exploit all potential information that can be extracted from the ambient surrounding him in order to improve his life (augmented reality).

Most of the times, in the literature or in commercial advertisements the word *smart* is used whenever an electronic device has an OS and can access the Internet, e.g., smartphone or smartTV. However, at the end of the twentieth century there were already mobile phones available with the functionalities of sending email and surfing the Internet as well, but they were not called smartphones.

Therefore, a first concern regards the philosophical inquiry about the real meaning of the word *smart* in our context. First of all, the word *smart* is a paradigm: it covers more than just one concept, i.e., a list of notions. This paradigm includes engineering the environment with sensor devices (sense), designing and developing the algorithms to collect and

extract new information (plan) and finally using this new knowledge to achieve (act) the desired objectives. The *smart* concept can also be applied to a location, as such a city or a house, where thousands of sensors are deployed to monitor every physical quantity. The data gathered could be employed to prevent dangerous risks to both infrastructures and people. Some physical quantities could measure the air pollution in a well-defined city area or to check the status of the street lights. It is clear that this paradigm has countless applications, e.g., track animals in danger of extinction through every kind of sensors in a wide area. The *smart* paradigm is similar to the sense-plan-act paradigm applied in the Artificial Intelligence and robotic fields; in fact, the already designed algorithms can be employed directly or readapted to be exploited in *smart* environments. At the end, the algorithms have as input either a single or multiple combined physical quantities to extract new information from, and the outcome can be visualized through a smartphone. As an example, the user's opportunity to find the best bicycle route from his current location to a desired destination with the lowest air pollution and the safest streets. The sensors can be deployed as wired or wireless sensor (WS) nodes. At the beginning, the WS nodes were used only to monitor some scenario, but more recently, especially in the last few years, some interesting applications can be designed attaching a WS node to every object. In this manner, a dummy object as a box can become a *smart*–box [1,2] through adding a WS node in order to get some information from both the users and the environment.

At this point, a new era is emerging based on the Internet of Things (IoT) paradigm [3]. The idea behind the IoT paradigm is that every object in the world can be connected to the Internet through some wired or wireless technology such as IEEE 802.15.4, Radio Frequency Identification (RFID), WiFi, or any new wireless communication. Therefore, the number of connected objects in the future will increase to many billions, connected together and generating a massive amount of data traffic as a consequence. The huge amount of wireless data traffic can bring new interesting challenges, mainly for two reasons. The former is that the bandwidth shared with other nodes could lead to a high rate of re–transmitted packets due to collision problems; the latter is to limit the energy consumption used to transmit the data, due to the fact that WS nodes are battery powered. Besides, it is fundamental to extend the sensors' life trying to limit any unnecessary use of energy to diminish the number of battery replacement operations.

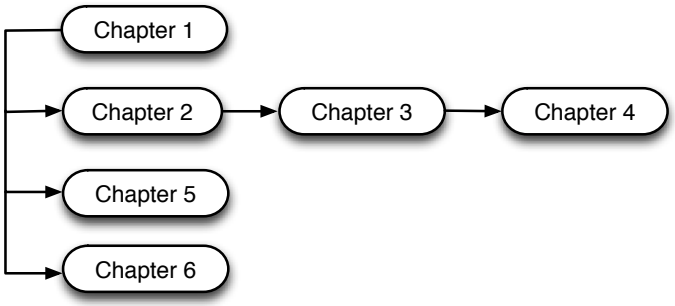



**Figure 1.1.** *Thesis' structure*

A possible idea is to decrease the number of packets over the air through compressing the raw data. In this case, it is essential to design a compression algorithm that is a trade–off between the number of operations requested in order to achieve a prefixed compression rate and to limit the use of the radio transceiver, i.e., the component with the highest energy consumption inside the node. The compression algorithm can be lossless or lossy, but it has to be considered that even lossy compression must not remove any important information the raw data is carrying.

The Internet of Things is a very complex and challenging paradigm. We give here a brief summary of the problems dealt with in this thesis. Self–management and self–organization of a Wireless Sensor Network through semantic web; signal compression and classification to decrease the problem of massive data exchange in IoT; a real scenario of IoT exploiting the *smart* object concept through an autonomous robot; a Cognitive Network (CN) [4] testbed based on commercial cheap devices with Android Operating System. The objective is to design and develop a CN testbed cheaper than the current CN testbeds built around the world, and easily extensible to test cognitive network algorithms , and assess their performance in real scenarios.

The thesis is organized as in Figure 1.1:

- Chapter 2 will focus on the description and the realization of an ontology system for Wireless Sensors and Actuators Networks dealing with heterogeneous device integration and composite event detection. The main features of the proposed system will be the complete interoperability thanks to the support of advanced web languages on constrained devices, the capability of classifying any node of the network according to

its sensors and its geographic position, and a general method for detecting events and anomalies among the data collected by the network;

- Chapter 3 will present the Motifs concept, to be applied in the signal compression area. The Motif concept is already used in data mining literature for signal classification. The idea will be to leverage the Motifs concept both to compress the signal and at the same time to classify unknown signals. Mathematical definitions, the main idea behind the Motif concept, and some preliminary results will be explained;

- Chapter 4 presents RAZOR, that leverages the concept of Motifs as recurrent features used for signal categorization, in order to compress data streams: in such a way, it will be possible to achieve good compression levels, while maintaining the signal distortion within acceptable bounds and allowing for simple lightweight distributed classification. In addition, RAZOR is designed to keep the computational complexity low, in order to allow its implementation in the most constrained devices;

- Chapter 5 will present a complete solution for the integration of robots and wireless sensor networks in an ambient intelligence scenario. The basic idea consists in shifting from the paradigm of a very skilled robot interacting with standard objects, to a simpler robot able to communicate with *smart* objects;

- Chapter 6 will describe a novel IEEE 802.11 mesh network testbed that integrates Android based devices. The aim is to build a flexible testbed to observe in-stack and out-stack parameters of interest, that can be used to test many networking techniques. It will provide all the implementation details to create an ad hoc network among these inexpensive commercial devices, and it will specify how to observe and modify the networking parameters at different layers of the protocol stack.

Finally, there will be a conclusions section and a small appendix which reports some additional work on security system for Global Navigation Satellite System (GNSS), realized through a GPS encoder developed as part of a PhD course project.

Chapter 2

# A first step towards an IoT system based on ontology

## 2.1 Introduction

In this chapter we will present a communication system based on the concepts of ontology and semantic web [5] to move a first step towards a self–configurable and self–manageable system with the purpose of alleviating the problem of the huge amount of data exchange in a Wireless Sensor Network (WSN)

Every day, thousands of terabytes of data are exchanged through the Internet. This huge amount of information would be useless without applications that link the single bits to their context and use them for the purpose they have been produced. Also, these applications can only work on data they already know how to process.

A common standard for data interpretation and enhancement is a mandatory requirement for information exchange. To this end, the concept of semantic web language has been introduced in the Web, that consists of leveraging on metadata to characterize data [5]. The eXtensible Markup Language (XML) is the de–facto standard for semantic web language.

Also modern communication systems aim at enabling machine to machine (M2M) interoperability. To this aim, data need to be presented in the most efficient format for their use by machines. Applying this concept to the Internet scenario led to the creation of autonomic agents for optimizing search within web pages [6] and the production of several sets of metadata, such as the Dublin Core [7].

The philosophical study of the nature of being, existence or reality is referred to as ontology. In information and communication science, an ontology is a formal representation of knowledge as a set of concepts within a domain, and also includes the relationships among those concepts [8]. Ontologies are frameworks used to organize data under any possible categorization system; they can be specialized for a particular application domain or merged into more general structures [9]. Tools capable of providing enhanced information by inspecting the logical relationships among different ontology classes are called *reasoners*.

This chapter aims at providing an ontology-based framework for the Internet of Things (IoT) [3] and at validating it by implementation and testing of two fundamental services for the interoperability of heterogeneous systems, namely, *i)* an autonomic classification method for Wireless Sensors and Actuators Network (WS&AN) devices and *ii)* an event detection solution capable of managing any type of sensed information regardless of the particular source.

Among many solutions present in the literature for writing, editing and visualizing ontologies, it is considered for this framework: Protégé [10], Ontolingua [11] and Ontostudio [12]. Protégé, which is the most widely used, offers the additional support to OWL2, whereas Ontostudio is only available as a trial version.

As for reasoners, it is evaluated a few examples available in the literature such as Pellet [13], Fact++ [14], and RacerPro [15]. Among these reasoners the most interesting is Pellet, which is OpenSource and accepts new axioms from the Semantic Web Rule Language (SWRL [16]).

The framework differs from others, such as CSIRO [17] or OntoSensor [18] as it combines ontology systems to event detection solutions for WS&ANs to adaptively treat generic events generated by heterogeneous devices, instead of focusing on single specific applications. In addition, this chapter describes an actual implementation of the framework and its validation on an actual building size testbed [19].

## 2.2 Software components

The actual implementation and the effective use of the ontology concepts into the service layer of information and communication systems makes it possible to extend the capabilities of ordinary databases in terms of flexibility, scalability, adaptability and self–management.

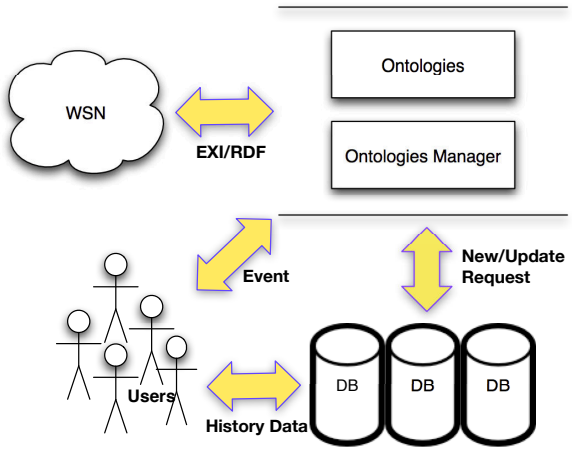



**Figure 2.1.** *Information flows among data generated from a WS&AN, the ontologies, the database and the users of a typical system*

Among the advantages provided are the capability of dealing with incomplete and semi-structured information and of integrating heterogeneous sources. For instance the WS&AN in Figure 2.1 can consist of several types of sensor nodes varying in terms of hardware and offered services.

In order for a monitoring system to manage huge quantities of information belonging to several different categories an ontology framework is needed. Each type of data can be associated to a set of semantic features of a vocabulary and a Reasoner/Classifier application will perform this data–features association. In the following is provided details on the main tools to work with ontologies: semantic languages, ontology editors and reasoners.

### 2.2.1 Languages

Ontologies are usually represented as stacks of knowledge abstraction layers. Starting from the bottom, there are the plain data layer, a machine-readable metadata layer, which uses the Resource Description Framework (RDF) [20], the Resource Description Framework Schemas (RDFSs) [21] layer and, at the top level, an expressive ontology layer using the Ontology Web Language (OWL) [22].

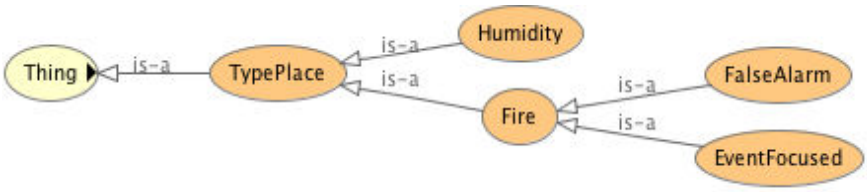



**Figure 2.2.** *An example of syntactical taxonomy associating event types to places, illustrated with Protégé.*

Raw data are generally gathered in XML documents, as in:

```
<?xml version='1.0' encoding='UTF-8'?>
    <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
                                                    syntax-ns#"xmlns:res="urn:sensei:rai">
    <res:Temperature rdf:about="2" res:hasDecimalValue
                                                                    ="27.5"/>
</rdf:RDF>
```

RDF represents data through directed graphs, where nodes are either subjects, $s$, or objects, $o$, and arcs are predicates, $p$, connecting the two. Each declaration in the graph comes as a triple $(s, o, p)$, where $s$, $o$ and $p$ are Unique Resource Identifiers (URIs) and objects can become subjects of other triples. RDFS is an extensible knowledge representational language defining a vocabulary for RDF and providing a syntactical organization [21].

OWL adds semantics to RDFS to better characterize properties and classes. In this work we used an OWL subclass defined by Description Logics (DL) to model concepts, roles and individuals, and their relationships. More in detail, DL is a subset of first order logic with the characteristic to be decidable (essential) and simple (low complexity description), which allows to efficiently infer further relationships on the available set.

Logical relations and rules connecting OWL concepts are called axioms and can be specified using the Semantic Web Rule Language [16], which is based on a combination of sublanguages of the OWL with one of the Rule Markup Languages.

### Editor: Protégé

The ontology framework is designed using Protégé [10], which allowed us to concentrate on the definition of an application domain and the related concepts without having to define all the layers of the stack (RDF/RDFS/OWL). Protégé provides developers with efficient

build/update methods and, thanks to the OWLViz plugin, a graphic interface showing the hierarchical structure of the whole ontology (Figure 2.2 shows a simple ontology illustrated with Protégé). Protégé is compatible with the main reasoners available in the literature and is the most widely accepted tool.

### 2.2.2 Reasoner: Pellet

Ontologies are just a representation of the application domain; in order for the system to be able to enhance its level of information a reasoner is needed. A reasoner is an application capable of inferring logical deductions from original axioms and accepting new axioms provided by users of the system. Among the available reasoners, we opted for Pellet, since it is open source, is compatible with open OWL tools, such as OWLAPI [23], supports OWL2 [24–26] and SWRL languages, and provides a consistency check function.

#### Library: OWLAPI

As shown in Figure 2.1 ontologies are usually supported by a management function block, which is in charge of the update procedure. Also, this logical block provides an interface between ontologies and the reasoner. This interface needs to be tightly coupled with the reasoner and to support the languages used in the system. The ontologies we are working with need to support OWL2, thus it is opted for for OWLAPI [23] to bridge Pellet with the ontology system.

## 2.3 An event detecting WS&ANs ontology

In this section we provided details about our WS&AN ontology design and deployment as well as the reasoner axioms needed to identify and categorize events in the system. The objectives of our design of this system are twofold: on one hand the system should be able to classify inputs produced by generic devices and, on the other hand, it must automatically detect anomalies among the data gathered in the network.

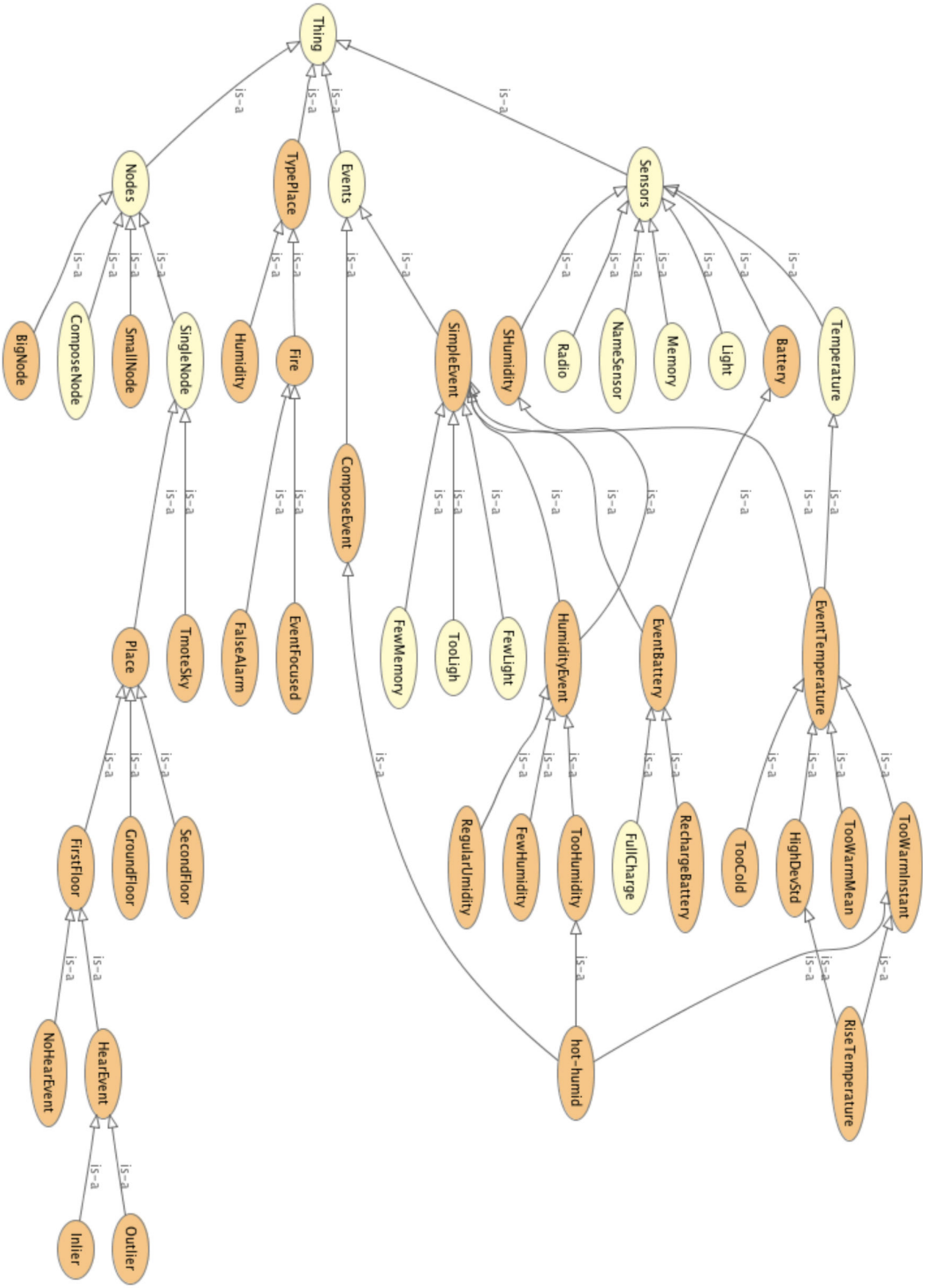



**Figure 2.3.** *The designed Ontology*

Figure 2.3 shows the ontology taxonomy[1] for our system. The first tier includes four classes: Places, Nodes, Sensors and Events. The Places class contains geographic information used to relate devices deployed in the same room or close to each other. The Nodes class contains an element per device in the network. The Sensors class contains an element for each accessible parameter of every device in the network as well as two additional elements for the hardware provider and the version. Finally, the Events class includes all the inputs generated by the event detection system. Arrows show which classes are included (IS-A) into other upper–tier classes; for instance, Temperature IS-A Sensor IS-A Thing.

In order to let the system automatically adapt to the insertion of a new device, the ontology will act as follows. When a new sensor is added to the network the system tries to classify it according to the collected data: upon detecting data coming from an identified source, the system can cross check with the available sensor types and, in case of a complete match, the new device version and provider are fetched from the local database; in case of partial match, the device type can be gathered from an online database of the same provider; when there is a complete mismatch, different vendor databases can be asked for getting the device information.

Such a structure allowed the system to classify any sensing device based on its capabilities and any given event based on its source and the place where it started. Of particular interest are those events reported when a monitored parameter reading exceeds a given threshold. The reasoner is able to distinguish among *general events* (most sensors agree on a critical condition), *focused events* (a single sensor detects a critical condition and the surrounding nodes report variations in the same direction but with smaller amplitude), and *outlier events* (a critical condition is reported by a single node but all other sensor readings are normal).

The skeleton of the ontology is not stationary, but it adapts according to the data gathered by the sensor nodes deployed in any given area of interest. For instance classes act as peer elements of the ontology such that, when nodes providing a certain feature are added to an area where this feature was not available, the new features are linked to the appropriate class. In turn, this can be used to enhance the characterization of a given class and to update the system according to the modification of the environment.

[1] http://www.dei.unipd.it/~danielet/ontology_work/taxonomy.jpg

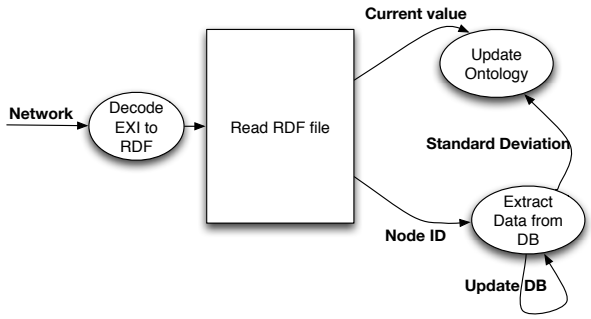



**Figure 2.4.** *Ontology Manager processing diagram*

The adaptation procedure works like in SUMO [9]: each node brings a piece of the information (subclass of the peer) forming the ontology, and the Ontology Manager (see Figure 2.4) can update the state of the current ontology. In addition, this design allows to combine different sensors to build new types of (logical) nodes, which are the combination of one or more sensors present in different nodes.

The Ontology Manager is in charge of updating the ontology with the data stored in a local database and those gathered from the WS&AN, and of exploiting the reasoner to translate the new knowledge to events triggered in the environment. Whenever new information is available, the Ontology Manager extracts the values and the node ID from the RDF message.

In order to explain how this system detects and categorizes events, we introduce the following quantities: $s_{ij}(t), i \in N$, is the reading of the sensor of type $j$ provided by node $i$ at time $t$, $\mu_{ij}(t)$ and $\sigma_{ij}(t)$ are the moving window time average and standard deviation of $s_{ij}(t)$, $\mu_j^L(t)$ and $\sigma_j^L(t)$ are the local average and standard deviation of $s_{ij}$ at time $t$, computed on those nodes belonging to the same Places class. Moreover, we defined two additional parameters $\mu_j^S$ and $\sigma_j^S$ which are the seasonal average and standard deviation of $s_{ij}$. All these quantities, except the seasonal ones, are updated whenever new data are available to the system.

The system monitors the evolution of the readings of each sensor in the system and generates $e_{ij}$, an event associated to sensor $j$ of node $i$, if

$$|s_{ij}(t) - \mu_{ij}(t)| > k_1\sigma_{ij}(t) \vee |s_{ij}(t) - \mu_j^S| > k_2\sigma_j^S,$$

where $k_1$ and $k_2$ are constants used to fine tune the system. By performing these two com-

parisons, the system can detect both when a reading varies too fast and when it exceeds the seasonal normal conditions.

However, a single sensor reading a critical condition is often not enough to rise a general alarm. Our system aims at associating events to their source positions, thus creating localized events, $e_j^L$, and classifying them as *general*, *focused*, or *outlier* events by exploiting the spatial correlation of sensor readings.

A general event is generated when $1/L \sum_{i \in place} I(e_{ij}) > 0.5$, where $L$ is the number of nodes belonging to the same Places class and $I(\cdot)$ is the indicator function; this acts like a consensus operation: when most of the nodes in a given place agree on signaling the event, it can be considered general for that place. When the inequality does not hold, focused events must be distinguished from outliers. Assuming that monitored quantities are varying slowly with respect to the node positions, nodes close to an event should detect some variations on their reading if not a proper event. Hence, if

$$\Big(1/L \sum_{i \in place} I(\sigma_{ij}(t)/\mu_{ij}(t) < k_3)\Big) > 0.5,$$

where $k_3$ is a tunable parameters, the event is considered an outlier, because most of the nodes show a low standard deviation, thus their readings are stable. Otherwise, the event is classified as focused, since nodes nearby detect the influence of the event.

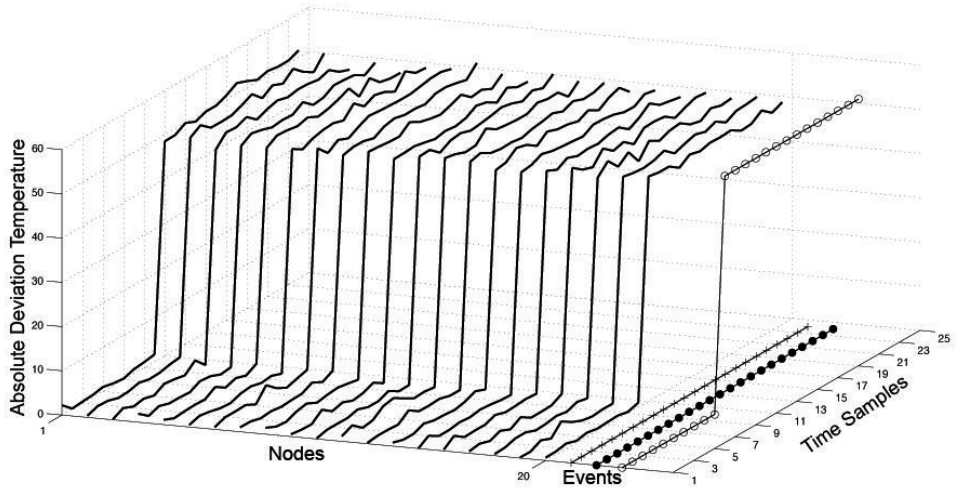
Finally, even though localized events are only relevant for a single parameter, alarms can be characterized by a set of critical conditions: for instance, a fire alarm will be the result of three abnormal conditions: temperature is too high, and a variation of the light and humidity levels is observed. Also, the tunable constants make it possible to adjust the system to the different physical quantities being monitored. In the following section this system is specialized to the case of temperature readings in the testbed deployed.

## 2.4 Results

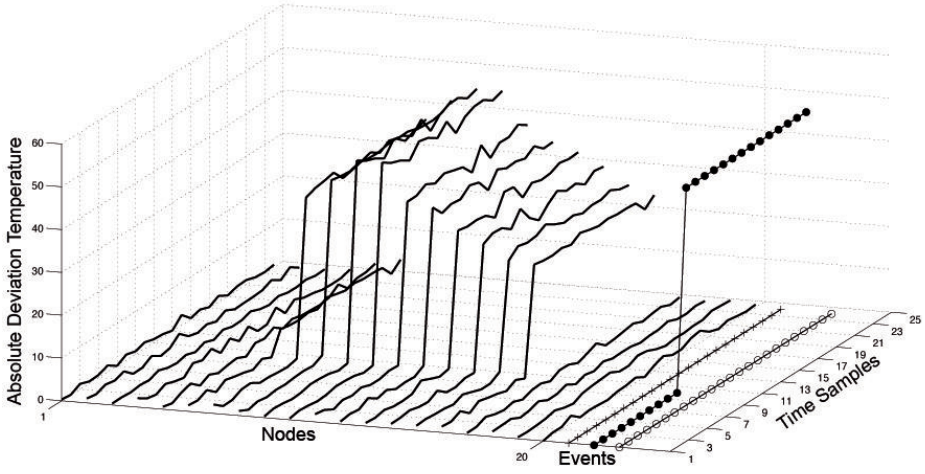
The validation of the framework has been performed by executing the ontology framework on the main WS&AN server of a 350–nodes testbed [19] deployed on a multi–hop, multi–room topology in the Department of Information Engineering at the University of Padova. The testbed devices are telosb wireless sensors [27] equipped with three transducers for temperature, humidity and light.

Since the devices in the testbed belong to the same sensor type, it has not been possible to validate experimentally the capability of the framework to manage heterogeneous hardware. However, we simulated such a situation by feeding the system with RDF documents generated by different technologies and the system has been able to distinguish the different hardware and to combine the output of similar readings. Further research will be devoted to validating the framework in an actual multi–technology testbed.

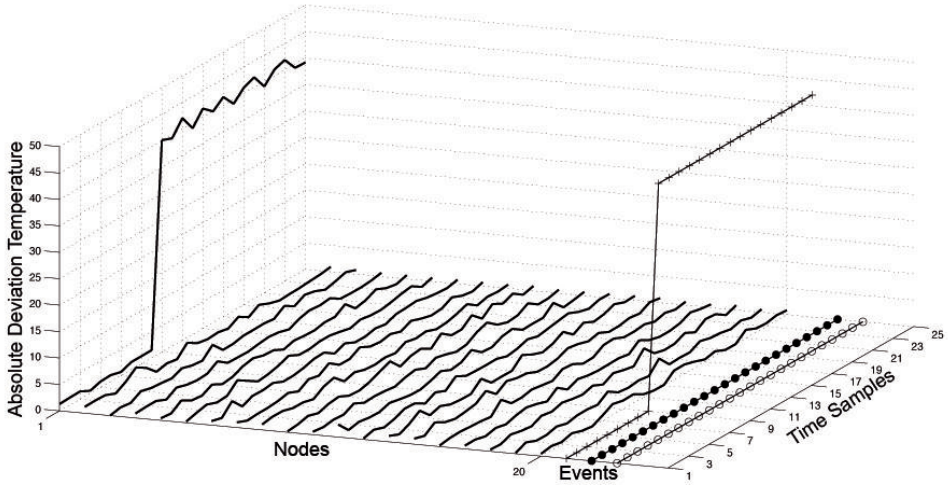
In order to limit the complexity of the campaign, we focused on the temperature readings of $20$ devices deployed in a single room and collecting samples every $5$ minutes over a $2$ hour period. This allowed us to deal with a slowly varying physical quantity, highly correlated in both time and space, hence we could easily simulate the three different classes of events by programming sensors to add a predefined bias to their sensor readings at any given time. In the series of plots of Figure 2.5 the lines without marker represent the trend of the sensor readings of each node in the room, whereas lines with $+$, black circle and white circle markers represent the detection of general, focused and outlier events, respectively. The first objective of the experiment was to test the system against an event detected by all the sensor in the room, see Figure 2.5-(a). The graph shows how the additional bias is added to the noisy sensor readings. The system reports that all nodes measured a fast increase of the temperature in the room, as a consequence the inequality $|s_{ij}(t) - \mu_{ij}(t)| > k_1\sigma_{ij}(t)$ holds for all nodes and $1/L \sum_{i \in place} I(e_{ij}) = 1$, hence the system outputs a general event of having reached a critical temperature. The second experiment has been set up to trigger a focused event. To do so, the additional bias has been designed to be very high on a single node while its intensity on other nodes is decreased proportionally to the distance. In this case $|s_{ij}(t) - \mu_{ij}(t)| > k_1\sigma_{ij}(t)$ holds only for a single node, but $(1/L \sum_{i \in place} I(\sigma_{ij}(t)/\mu_{ij}(t) < k_3)) < 0.5$, because only few other nodes are detecting unbiased readings. The reasoner can infer that only a single node reached a critical condition, but many others experience the side effects of the event; it can deduce that something is happening in the room, and it has a distinct source position, hence it classifies it as a focused event. Figure 2.5-(b) shows the output of the system, where the yellow line represents a focused event detection.




(a)




(b)




(c)

**Figure 2.5.** *Event detection: solid lines represent sensor readings on a* 2 *hour period, sampled every* 5 *minutes, while lines with markers show the detection of different events; general (+), focused (black circle), outlier (white circle).*

The last experiment is similar to the second, but the bias is added to only a single node: hence $|s_{ij}(t) - \mu_{ij}(t)| > k_1\sigma_{ij}(t)$ holds again for a single node, but $(1/L\sum_{i\in place} I(\sigma_{ij}(t)/\mu_{ij}(t) < k_3)) > 0.5$. The reasoner can then infer that the readings of the node rising the event are isolated and marks it as an outlier event. Figure 2.5-(c) shows what is happening in the system and the blue line reports the outlier.

Although the results are encouraging, there exist many *gray zones* in the detection system, such as when the network is too sparse or has too few nodes to consider the spatial correlation assumption as valid.

## 2.5 Conclusions

In this part of thesis was proposed an ontology system to deal with two of the most characteristic issues in WS&ANs: dealing with heterogeneous devices and monitoring events. The designed ontology framework proved to be highly flexible and allowed us to treat any device inserted in the network by simply classifying it according to the offered sensing capabilities. In addition, the system is able to classify nodes based on their geographical positions and to exploiting this information for event detection.

It was developed an event detection system leveraging local, time and seasonal information in order to distinguish among general, focused and outlier events. Also, the system can combine multiple detected events to rise an alarm in response to critical conditions. Finally, we validated our system on a building size testbed, showing that the framework is in fact able to distinguish among the three types of events. This system leaves open other two interesting challenges: the former regards the amount of data exchanged among the sensor nodes and the latter regards the classification of unknown data. Both challenges fit well within the IoT paradigm because the idea to connect billions of objects raises the problem of limiting the use of some consumable resources for each WS node like bandwidth, battery and memory. Also, within the IoT paradigm it will occur frequently that a node joins or leaves the network, therefore when a new unknown node will join the system, it is important to classify which are the physical quantities that the node is able to gather. In the following Chapters 3,4 we will improve the system by introducing a novel technique to compress and classify unknown signals gathered from the sensor nodes present into a *smart* environment.

Chapter 3

# Motif concept from data mining to the Internet of Things

## 3.1 Introduction

In the previous chapter, we presented a high level concept for IoT to self–manage heterogenous devices deployed in a scenario to detect sudden events. In this chapter we will present in detail our strategy to reduce the data exchanged and to classify unknown data.

*Do not say a little in many words, but a great deal in a few* [28]. Even though Pythagoras' words of wisdom focused on dialectic, they are still relevant more than two thousand years later if applied to communications. In fact, with the advent of the Internet of Things , the growth of information sources in number and density is increasingly becoming a bottleneck, especially for constrained networks, where bandwidth and computational capabilities are limited. In this chapter, we treasure the Greek mathematician's words and will focus our research on improving constrained device communications by leveraging on compression and classification.

This chapter starts from the following considerations on the IoT: (1) data is measured by a huge number of (mostly) constrained devices; (2) a variety of physical phenomena, often incommensurable with one another, are monitored (typical information exchanged in the IoT includes simple readings (e.g., temperature, light and humidity), and simple operational states (e.g., energy consumption, production chain triggers and presence detection)); (3) recurrent data patterns can be found in the temporal and spatial evolution of the data (as

an example, similar outputs are likely to be gathered by closely deployed devices and to be repeated over the same hours of different days). As a concrete application example, Wireless Sensor and Actuator Networks (WS&AN) capabilities will be the reference for the rest this chapter, and in particular, the telosb architecture [27] will be taken as representative for the whole category, *i.e.*, 1 MHz CPU and a 250 kbps nominal data rate.

An important question arises: Is it possible to exploit pattern similarities in time series for realizing a general information processor for data source classification and communication enhancement? In our opinion it is and, to demonstrate it, a lightweight communication solution are being proposed to leverage on recurrent patterns, called Motifs, for both data compression and classification. With this solution it will be possible to realize a universal data processor capable of inferring the category the data belongs to from the particular patterns identified and, when a data source is associated to a given category, it will be able to use the most frequent patterns to code its information: instead of sending the actual sensed samples, a single pattern identifier can be used to transmit the whole sequence.

The rest of the chapter is structured as follows: Section 3.2 is reported the related works exploited to achieve the prefixed objectives; Section 3.3 describes the system model and introduces some terminology; Section 3.4 elaborates on the pattern representation and identification procedures, presents two lightweight solutions and evaluates them against their system parameters; Section 3.5 discusses about the quality achievable using the Motif-based communication and provides an evaluation in terms of distortion, compression and classification performance. Finally, Section 3.6 concludes the chapter and gives an outline of the ongoing activities.

## 3.2 Related work

First of all, this work would not have been possible without the fundamental contributions by Gersho and Gray on vector quantization [29], Keogh and his group on data mining [30] and Bishop on pattern recognition [31].

Vector quantization techniques are the most similar to the algorithm developed in this work, since they are based on the creation of a Codebook from a training dataset and its subsequent usage for replacing new segments of the time series with one element from the Codebook. Furthermore, Murakami in [32] developed a technique that stores normalized

signals in the Codebook, a concept similar to the solution proposed in this chapter (see Section 3.4).

Bishop in [31] provides a thorough survey on pattern recognition techniques using both parametric and non parametric models. However it was the work of Keogh [33–36] that gave the inspiration for the studies. In particular, reference [36] studies a parameter free data mining technique using a Compression-based Dissimilarity Measure (CDM) which proves to be very efficient for classification, while [34, 35] use a Symbolic Aggregate approXimation (SAX) technique for identifying patterns in a given time series. However, all these techniques are either too computationally complex (the former) or too lossy (the latter) to achieve the objectives prefixed. Another important milestone is [37] which proposes a clustering technique based on compression and provides a rigorous analytical framework based on the notion of Kolmogorov complexity.

In the literature, several other attempts to combine compression and communication are found: for a detailed survey on in-network aggregation techniques the reader is referred to [38], while recent studies on Compressive Sensing can be found in [39, 40]. Reference [41] studies time series compression and prediction quality in WS&ANs. However, all these techniques, while being very efficient in compressing information how it is reported in Chapter 4, but they do not provide any means to recognize and classify data sources.

## 3.3 System overview

The networking community uses the term IoT with several different meanings. However, all the different opinions agree on the need for *smart* objects to be connected through the Internet. An object becomes *smart* when a common object is attached a *smart* those devices that, though hardware constrained, are still capable of performing their task by exploiting their limited capabilities in clever ways. This objective can be achieved in several ways and with different technologies, such as Radio Frequency Identifier (RFID), Wireless Sensor Networks (WS&AN), *etc*. An example of *smart* objects is presented in Chapter 5. Each solution can adopt a different communication paradigm, such as ZigBee or IPv6 over Low power Wireless Personal Area Networks (6LoWPAN), and it is referred to each subnetwork adopting a homogeneous communication solution as an IoT island. The IoT has several goals, one is presented during the Chapter 2 and another one is the optimization of

data exchange through strategic resource preservation, *i.e.*, saving battery charge and reducing communication overhead. As a concrete example of a typical IoT island, we focus on a single WS&AN composed of many constrained devices, referred to as *nodes*, characterized by sensing and/or actuation capabilities, limited resources ($\sim 10$ kB RAM , $\sim 50$ kB ROM, battery powered) and computational capabilities (1 MHz CPU) and low data rate ($\leq 250$ kb/s). Usually, these nodes need to use more powerful devices, referred to as *gateways*, to be connected to the Internet and to be able to be interacted with. The reader is referred to [42] for details about constrained node communication and Internetworking.

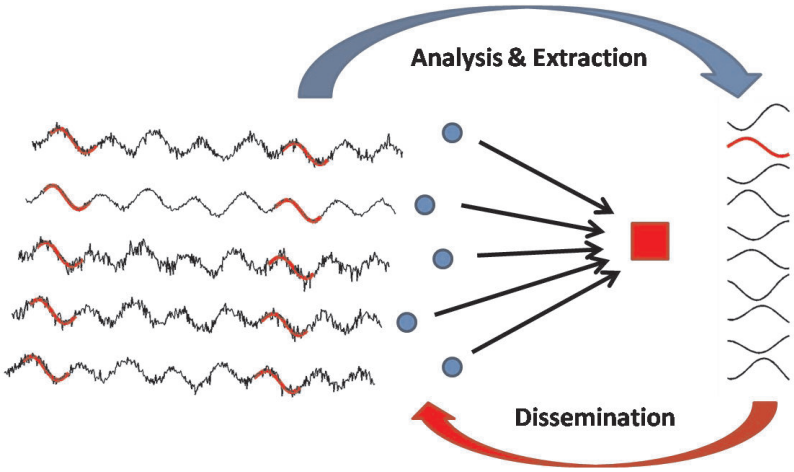



**Figure 3.1.** *Schematic diagram of a simple Internet of Things (IoT) network: blue circles and the red square represent sensor nodes and a gateway node, respectively.*

Figure 3.1 illustrates how the system works: sensor nodes (blue circles) produce information flows related to the same physical phenomenon and transmit the raw information towards a gateway. For simplicity, we assume that a single phenomenon is monitored and that all nodes are of the same type, and leave more complex scenarios for future work. The gateway, which is unaware of the particular type of information carried by the data flows, analyzes the raw data in order to identify the most frequent trends. After an initial training period, the gateway is able to compute a set of patterns and to disseminate them back to the nodes (details are in Section 3.4.3). Then, the nodes compare the sensed information with the received pattern set, and instead of sending the whole data sequence, they just send the pattern identifier (details are in Section 3.4).

This chapter is concentrated on signal compression only; however, it is also possible to

exploit the same patterns to detect outliers and anomalies as in [43,44] or for data prediction as in [45].

## 3.4 Motif extraction

In this section we will focus on segments of $N$ samples of a given training set of the time series $S(t)$ and on how to identify those representing the most frequent shapes of the signal. We will refer to these segments as Motifs in the rest of the thesis. The idea behind looking for frequent configurations of the signal is that of using them for building a sort of dictionary that can be used both for compressing the signal and as a descriptor for the signal category.

A first question on Motif extraction is the following: which features make a segment representative of a given category of shapes and, thus, a Motif? Reference [46] provides the most complete answer, however, given the computational complexity of the proposed metrics, we cannot adopt the same solution proposed therein, but we will propose two alternative simpler segment definitions and two lightweight extraction procedures that are conceptually similar to the technique described in [34]. However, while the objective in [34] is signal classification, here we are more interested in signal transmission. For this reason we need to use a larger set of Motifs.

### 3.4.1 Mathematical definitions

The Motif concept is inspired by two different applications of signal processing for time series: on the one hand, our solution can be categorized as part of the vector quantization techniques [29], but on the other hand, we apply solutions derived from data mining and pattern recognition [31, 34] during the learning phase. In order to understand the impact of the two inspiration sources, we will provide a mathematical definition of the problem as follows.

The original signal that we want to analyze is:

$$S(t) \in \mathbb{R}, t \in [0, +\infty) \tag{3.1}$$

but in order to maintain the signal as processable in a finite time, we will focus on a subset of $S(t)$, called the *training set*, of length $M_S$ and defined as:

$$T(t) = S(t), t \in [0, M_S] \tag{3.2}$$

We assume that the signal and the training set are sampled with period $T$, respecting the Nyquist–Shannon sampling theorem, and quantized on an alphabet, $\mathcal{A} \subset \mathbb{R}$, of $L$ symbols:

$$\begin{aligned} S_i &= Q(S(iT)) \in \mathcal{A}, \\ T_i &= S_i, i = 0, 1, \ldots, M-1 \end{aligned} \tag{3.3}$$

where $S_i$ and $T_i$ are samples of the original signal and of the training set, respectively, $M = M_S/T$ is the length of the sampled signal and $Q(\cdot)$ is an $L$-level quantization function. Notice that, while the sampling period, $T$, and the number of levels, $L$, are usually given by the application, it is also possible to adapt these two parameters dynamically; however, this opportunity is not covered in the current work.

As in standard Vector Quantization, our objective in this section is to extract a Motif Codebook from the training set. The parameters governing this Codebook are three:

1. $K$, the number of motifs in the Codebook,
2. $N$, the length of the motifs in samples,
3. $b$, the number of bits used to encode each sample of the motifs in the Codebook.

In Section 3.5, we will analyze the impact of each of these parameters on the performance metrics. The first step of the extraction algorithm is to identify in the training set a number of segments of $N$ samples each. Each segment is defined as:

$$T^{(i)} = [T_i, T_{i+1}, \ldots, T_{i+N-1}],\ i = 0, 1, \ldots, M-N+1 \tag{3.4}$$

representing a set of $N$ consecutive samples of the training set starting from sample $T_i$.

### 3.4.2 Segment representation

A segment $S_i$ can be used with the own Actual Value (AV) or the segment can be modified through a segment normalization to achieve better results. Thus, the idea of using normalized segments, instead of their absolute values, can also be found in standard Vector Quantization [32]; in such a way, the Codebook contains a set of templates that can be used as building blocks for the original signal after appropriate de-normalization. Normal shapes are obtained from the original segments by subtracting the offset and dividing by an amplitude gain. Firstly, let $\mu_{S^{(i)}}$ and $\sigma_{S^{(i)}}$ be the average and standard deviation of the

segment $S^{(i)}$. In order to analyze the signal we can either use the *actual values* (AV) of the segments $S^{(i)}$ or their *normal shapes* (NS), $\overline{S}^{(i)}$: by normal shape, we mean a segment which has the same shape as the original segment, but has zero mean and unit standard deviation so that it is invariant with respect to amplitude and offset [46]: $\overline{S}^{(i)} = (S^{(i)} - \mu_{S^{(i)}})/\sigma_{S^{(i)}}$. If normal shapes are used to transmit the signal, they need to be sent along with their $\mu_{S^{(i)}}$ and $\sigma_{S^{(i)}}$. Also, the considered training set, having $M$ samples, contains at most $M - N + 1$ different shapes.

In the following are defined the de-normalized version of a given segment or motif with the subscript, $R$. For instance, the de-normalized version of $\overline{S}^{(i)}$, will be:

$$S_R^{(i)} = \overline{S}^{(i)}\sigma_{S^{(i)}} + \mu_{S(i)} = S^{(i)} \tag{3.5}$$

which is the same as the original version if no additional quantization is used.

In this section the amplitude function is segment's standard deviation, instead in Section 4.3.2 is reported more comparison with several amplitude gain functions.

In the following section will be presented two lightweight algorithms to extract the segments that are most representative under certain constraints. The algorithms will be ran with both the segment representation.

### 3.4.3 Algorithms

In the following, we will detail our methods for selecting $K$ representative Motifs out of the possible $M - N + 1$ segments, for both the AV and NS representations. We will also take into account the impact of the system parameters: $b$, the number of bits per sample of the stored Motifs; $N$, the Motif length[1]; and $K$, the number of Motifs.

In order to compare the different segments we choose to adopt the Euclidean distance $d(S^{(i)}, S^{(j)}) = \|S^{(i)} - S^{(j)}\|_2 = \sqrt{\sum_{k=0}^{N-1} |S_{i+k} - S_{j+k}|^2}$ to build the matrix $\mathbf{DM}$, whose element $DM_{ij}$ contains $d(S_R^{(i)}, S_R^{(j)})$ or $d(\overline{S}_R^{(i)}, \overline{S}_R^{(j)})$ in case we are considering normal shapes. Without any loss of generality, we will only write the case for AV segments.

We propose two lightweight algorithms for extracting $K$ Motifs using only the information contained in $\mathbf{DM}$. These algorithms work independently of the particular distance used, hence it is straightforward to extend this work using other distances like reported in

[1]Here, we considered segments of the same length $N$ only, because of ease of storage and computation.

Section 4.3.3. Motifs will be referred to as $X(i), i \in \{1, \ldots, K\}$. The two algorithms are based on the following criteria, respectively: *i*) prefer those segments that show a lower average distance from all the other segments; *ii*) prefer those segments that are similar to the largest number of other segments. We will refer to the former as *brute force* (BF) and to the latter as *matching based* (MB).

The BF approach selects the $K$ segments having the smallest average distance computed on all the other segments. The actual procedure is illustrated in Algorithm 1. Instead, the

**Algorithm 1** Brute Force Motif selection algorithm.

**Require:** Precomputed distance matrix **DM** with $d(\cdot)$ a dissimilarity metric

$j \leftarrow 1$

$\forall m, \mu_m \leftarrow 1/(M-N+1) \sum_n DM_{mn}$

**while** $j \leq K$ **do**

$i \leftarrow \arg\min_m(\mu_m)$

$X(j) \leftarrow S_Q^{(i)}$

delete the $i$-th row and column in **DM** and $\mu_i$

$j \leftarrow j + 1$

**end while**

MB algorithm leverages on the concept of matching: we say we have a matching when $DM_{ij} < d_{th}$, where $d_{th}$ is a given threshold. At every iteration the number of matchings is computed for every segment against every other segment in the training set. The segment with the highest matching count is inserted in the Motif set. (Ties are broken using the average distance as in Algorithm 1.) Then rows and columns of the selected Motif and those of all its matching segments are removed from **DM**. This means that a given Motif will be used to represent any other segment which is less than $d_{th}$ distant from it. Since it is difficult (and sometime impossible) to define a priori a $d_{th}$ that leads to selecting exactly $K$ Motifs, we proceed using a bisection iterative method. The whole process is illustrated in Algorithm 2 (where $I(\cdot)$ is the indicator function). Note that Algorithm 2 might end with $j \neq K$; in such a case we will either reduce the number of Motifs selected by the algorithm or we will integrate them by selecting the remaining $K - j$ with the BF approach.

Finally, we accounted for a further quantization step using $2^b$ levels of quantization ($b$ being the number of bits per sample), which reduces the size of the Motif set, decreases the

**Algorithm 2** Matching Based Motif selection algorithm.

**Require:** Precomputed distance matrix **DM** with $d(\cdot)$ a dissimilarity metric

$d_{max} \leftarrow \max(\mathbf{DM})$

$d_{min} \leftarrow \min(\mathbf{DM})$

**while** $d_{max} - d_{min} > \epsilon$ **do**

$j \leftarrow 1$

$d_{th} \leftarrow (d_{max} + d_{min})/2$

$\forall m, n, MM_{mn} \leftarrow I(DM_{mn} < d_{th})$

**while DM** $\neq \emptyset$ **do**

$\forall m, MC_m \leftarrow \sum_n MM_{mn}$

$i \leftarrow \arg\max_m(MC_m)$

$X(j) \leftarrow S_Q^{(i)}$

delete the $i$-th row and column in **DM** and **MM**

$j \leftarrow j + 1$

**end while**

**if** $j = K$ **then**

exit

**else if** $j < K$ **then**

$d_{max} \leftarrow d_{th}$

**else**

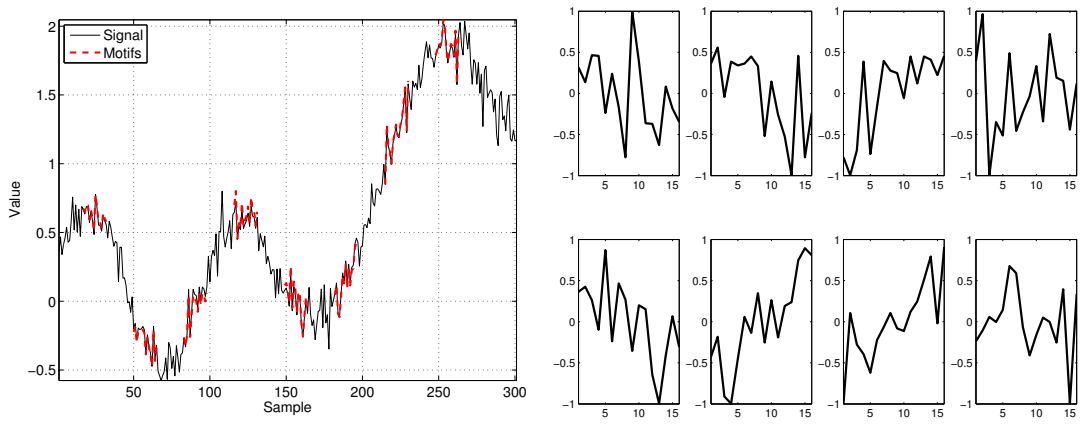
$d_{min} \leftarrow d_{th}$

**end if**

**end while**

matching complexity and improves the Motif storage. We will use the following notation: $S_Q^{(i)} = Q(S^{(i)})$, and $\overline{S}_Q^{(i)} = Q(\overline{S}^{(i)})$ represent quantized versions of the segments for AV and NS, while $S_R^{(i)}$ and $\overline{S}_R^{(i)}$ are the real valued representations of $S_Q^{(i)}$ and $\overline{S}_Q^{(i)}$, respectively.

The selected Motifs will be referred to as $X(k), k \in \{1, \ldots, K\}$. Figure 3.2 shows a training set example (on the left), highlighting, in bold red, those segments selected by the extraction algorithm to form the Codebook and the generated Codebook (on the right), for $K = 8$, $N = 16$ and $b = 8$.




(a) Training Set (b) Codebook

**Figure 3.2.** *An example of the extraction procedure for* $N = 16$, $K = 8$ *and* $b = 8$*: (**a**) shows a given training set, highlighting, in bold red, the segments selected to form the Codebook; (**b**) represents the Codebook. Note that in the Codebook, the motif values range is* $[-1, 1]$.

### 3.4.4 Motif selection.

Given a Codebook with K Motifs, it is necessary to establish a procedure to find the best Motif $X(k)$ that can replace the segment $S^{(i)}$. Therefore, for a given segment, $S^{(i)}$, we compute its distance to each of the motifs in the Codebook and pick the motif $\widehat{X}^{(i)}$ that minimizes that distance (note that we need to use the same dissimilarity metric used in the extraction phase in order to preserve the physical meaning of the distance). In symbols, this is:

$$\widehat{X}^{(i)} = X_R(l),\ l = \arg\min_k d(S^{(i)} - X_R(k)) \tag{3.6}$$

where $k, l \in \{1, \ldots, K\}$, $i \in 0, 1, \ldots, M-N+1$ and $X_R(l) \in \mathbb{R}$ is the de-normalized version of $X(l)$, the $l$-th Motif in the Codebook. Algorithm 3 provides the listing of the Motif selection process.

**Algorithm 3** : Motif selection algorithm.

**Require:** $B$, the Codebook of the motifs; $S^{(i)}$, the segment to be compressed; $d(\cdot)$, a dissimilarity metric; $d_{th}$, the compression distortion threshold.

$d_{min} \leftarrow \infty$

$\overline{\mu} \leftarrow Q(\mu_{S^{(i)}})$

$\overline{\sigma} \leftarrow Q(\sigma_{S^{(i)}})$

**for all** $X(k) \in B$ **do**

$X_R(k) \leftarrow X(k)\overline{G} + \overline{O}$

**if** $d(S^{(i)}, X_R(k)) < d_{min}$ **then**

$\widehat{X}^{(i)} = X_R(k)$

$d_{min} = d(S^{(i)}, X_R(k))$

**end if**

**end for**

Return $k$

## 3.5 Evaluations

In this section is reported the evaluation metrics and comparison among the all combination between two Motif extractor algorithms and two different segment representation. The evaluation will cover the three main objectives exploiting the Motif concept: the reconstruction error, the data compression and the classification.

### 3.5.1 Brute Force against Matching Based

The first step is to evaluate the goodness of the two segment representations, AV and NS, and the two Motif extraction algorithms, BF and MB. We decided to use, as the evaluation criterion, the average error introduced when a Motif is used to represent a segment instead of the segment itself. Note that the nature of the error is twofold: on the one hand there is the error introduced by replacing the actual samples with those of the Motif and, on the other hand, there is the error introduced by the quantization.

In this phase we are not aiming at using the selected Motif for representing the signal as a whole, but we are focusing on single segment representation only: in order to calculate

the error, $E^{(i)}$, introduced by using Motif samples instead of those of the actual signal, for any given segment $S^{(i)}$, we find the Motif that best represents it, $\widehat{X}$, where $\widehat{X} = X(i), i = \arg\min_j d(S^{(i)} - X_R(j))$, with $j \in \{1, \ldots, K\}$, $X_R(i)$ being the real valued version of $X(i)$; hence, for the AV representation, the error is $E^{(i)} = \|S^{(i)} - \widehat{X}_R\|_2/(NA)$, where $A$ is the maximum amplitude of the original signal; Instead, for NS representation it is necessary to rescale the Motifs: $E^{(i)} = \|S^{(i)} - \sigma_{S^{(i)}}\widehat{X}_R + \mu_{S^{(i)}}\|_2/(NA)$. Finally, we compute the distortion rate as the average error among all the possible segments in the training set:

$$E = \frac{1}{M - N + 1} \sum_{j=1}^{M-N+1} E^{(j)}, \tag{3.7}$$

$E \in [0, 1]$ quantifies the average sample difference between the raw signal and that represented with the best Motif normalized to the maximum amplitude $A$.

The objective of this evaluation campaign is to select among the segment representation methods and the Motif extraction methods those providing the best results and, possibly, to obtain some insight on the impact of the parameters. For convenience, we performed our evaluation campaign on the datasets available at the UCR Time Series Classification/Clustering Homepage [47]: in particular we tested our algorithms on the Synthetic Control dataset only, because we needed some experimental evidence about the feasibility of our methods before applying them to wireless sensor communications.

We proceeded performing the Motif extraction with the four combinations of the two segment representations, AV and NS, and the two extraction methods, BF and MB, letting the system parameters vary as follows: the stored bits per sample $b \in \{3, \ldots, 12\}$, the Motif length $N \in \{3, \ldots, 20\}$ and the number of Motifs $K \in \{3, \ldots, 20\}$.

Figure 3.3 shows our results: in particular the curves with circle and square markers refer to the AV and NS segment representation methods, while solid and dashed curves refer to BF and MB algorithms, respectively. Curves are plotted against one parameter at a time and averaged over the remaining two, thus Figure 3.3-(a) analyzes the average behaviors of the four solutions as a function of $b$, Figure 3.3-(b) as a function of $K$ and Figure 3.3-(c) as a function of $N$.

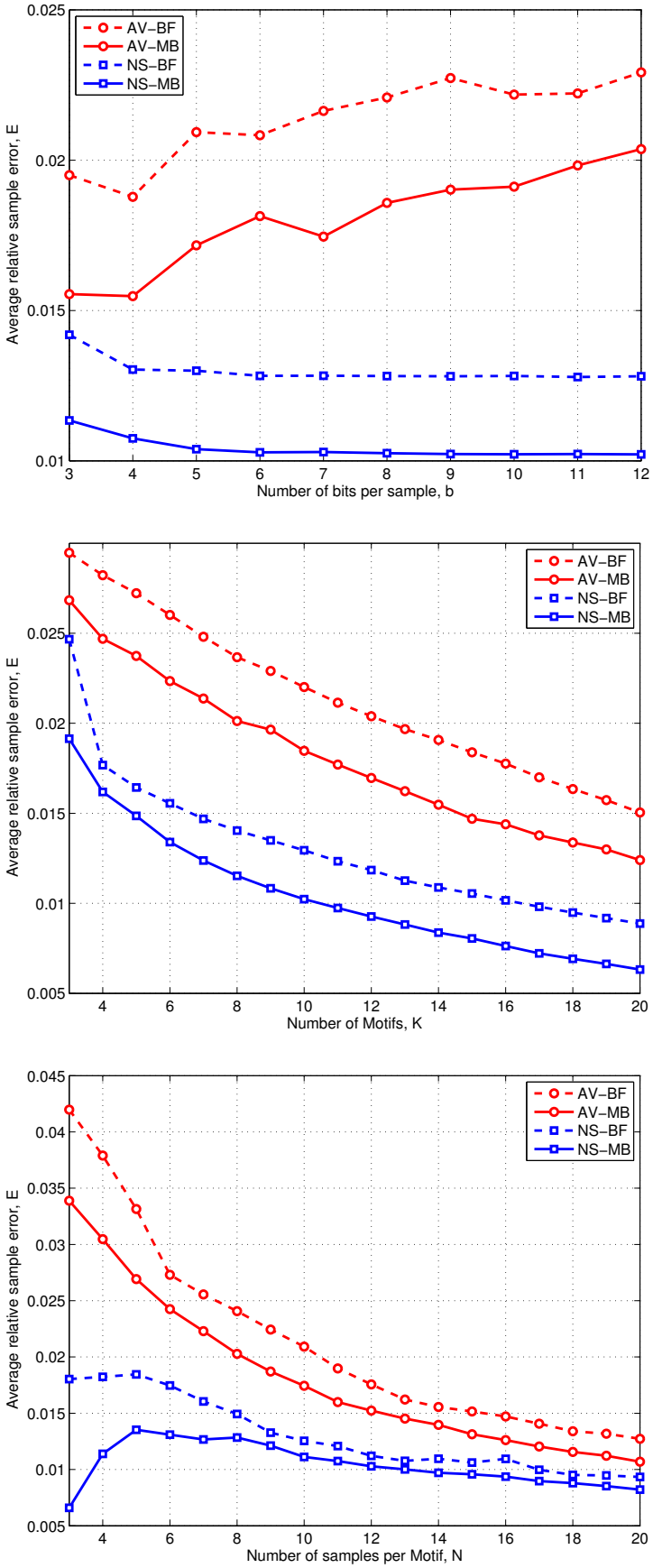



**Figure 3.3.** *Distortion rate, $E$, as a function of $b$ (a), $K$ (b) and $N$ (c), averaged on the other two parameters.*

While increasing the number of bits beyond $6$ seems not to provide any perceivable benefits, increasing either $K$ or $N$ substantially reduces the distortion rate, $E$. However, the best performance in terms of $E$ is obtained for very low $N$ and NS representation: with such configuration Motifs are just reproducing very simple signal shapes, such as increasing or decreasing slopes or positive or negative peaks, hence it is very likely that the signal can be closely represented, however, such short Motifs are not representative of any signal category at all and do not provide any compression.

Finally, the most important considerations are: *i*) NS representation almost halves $E$ with respect to AV as it is effective for pattern recognition and *ii*) MB outperforms BF because it favors the most frequently used Motifs.

### 3.5.2 Application of Motifs to compress and classify signals

This section elaborates on how to leverage on pre-defined Motifs to represent a time series before transmitting it. In particular our objectives are: *i*) to quantify the distortion introduced on the sensed signal, *ii*) to quantify the compression achieved, *iii*) to verify that the selected Motifs are also usable for classification and *iv*) to quantify the memory needed on the constrained nodes.

We assume here that a suitable learning phase has been performed to collect the $K$ best Motifs for a given parameter set $\{b, K, N\}$, and that any given node can use them to code its information. We choose the simplest communication solution in order to keep the problem complexity low and the error sources limited: basically, any given node waits until $N$ samples are available to be transmitted, then it compares the buffered samples with the stored Motifs and, instead of sending the actual samples, it just sends the identifier of the best Motif, $\widehat{X}$.

To this extent, we first define a few metrics needed to quantify the results obtained by the communication protocol and to optimize it: we will express the reconstruction error $E_R$ on a given test dataset $S(t)$ as follows:

$$E_R = 1/(AN\lfloor M/N \rfloor) \sum_{j=0}^{\lfloor M/N \rfloor - 1} \|S^{(jN+1)} - \widehat{X}_R)\|_2 \tag{3.8}$$

where $A$ is the maximum amplitude of the test set, $M$ is its number of samples and $\widehat{X}$ is the best Motif for representing a particular segment of the test set.

Secondly, we define the compression factor $C_F$ as the ratio between the number of bits needed to represent a segment using Motifs and its original size. Depending on the actual segment representation, $C_F$ can have different expressions, but its simplest expression, obtained using AV, is given by: $C_F = \lceil \log_2(K) \rceil / N b_r$, where $b_r$ is the number of bits per sample of the raw signal and $\lceil \log_2(K) \rceil$ is the number of bits needed to select one Motif out of $K$. Hence, the maximum compression achievable is $C_{Fmax} = \lceil \log_2(K) \rceil / N b_r$. When using NS segment representation, it is necessary to transmit the actual mean and standard deviation of the segment to be reconstructed along with the Motif index, hence

$$C_F = (b_{\mu_{S^{(i)}}} + b_{\sigma_{S^{(i)}}} + \lceil \log_2(K) \rceil) / N b_r, \tag{3.9}$$

where $b_{\mu_{S^{(i)}}}$ and $b_{\sigma_{S^{(i)}}}$ are the numbers of bits needed for coding the mean and the standard deviation, respectively. For the sake of simplicity, we will consider $b_{\mu_{S^{(i)}}} = 16$ and $b_{\sigma_{S^{(i)}}} = 16$ in the rest of the thesis, because $16$ bits is a plausible resolution for accurate numbers in constrained devices.

To compute the reconstruction error $E_R$, we built a Motif set for any category of the training data set [47] and for any parameter configuration, $\{b, K, N\}, b \in \{3, \ldots, 10\}, K \in \{3, \ldots, 20\}, N \in \{3, \ldots, 20\}$ according to the segment representation NS and the extraction method MB of Section 3.4. Subsequently, we used those Motif sets to reconstruct the test sets of all categories. In such a way, we were able to quantify the performance of our technique with signals belonging to all categories.

We considered four of the categories available in the dataset, namely *Periodic*, *Noise*, *Ramp* and *Decreasing Ramp*. Each category contains $50$ training signals and $50$ test signals, of which we used the first of the training set for computing the Motif set and the remaining $99$ for evaluating the performance.

Finally, we used a simple classification method based on the reconstruction error: any given signal is reconstructed using the Motifs computed for the four categories and is classified in the category scoring the smallest $E_R$. In particular, we counted a correct classification in a given category whenever the $E_R$ scored with the Motif generated by the same category is the lowest among the considered categories and we counted a false positive when the lowest $E_R$ is scored with the Motif generated from a signal category different from that of the tested signal.

Figure 3.4 gives a partial overview of the results obtained on communication aspects.

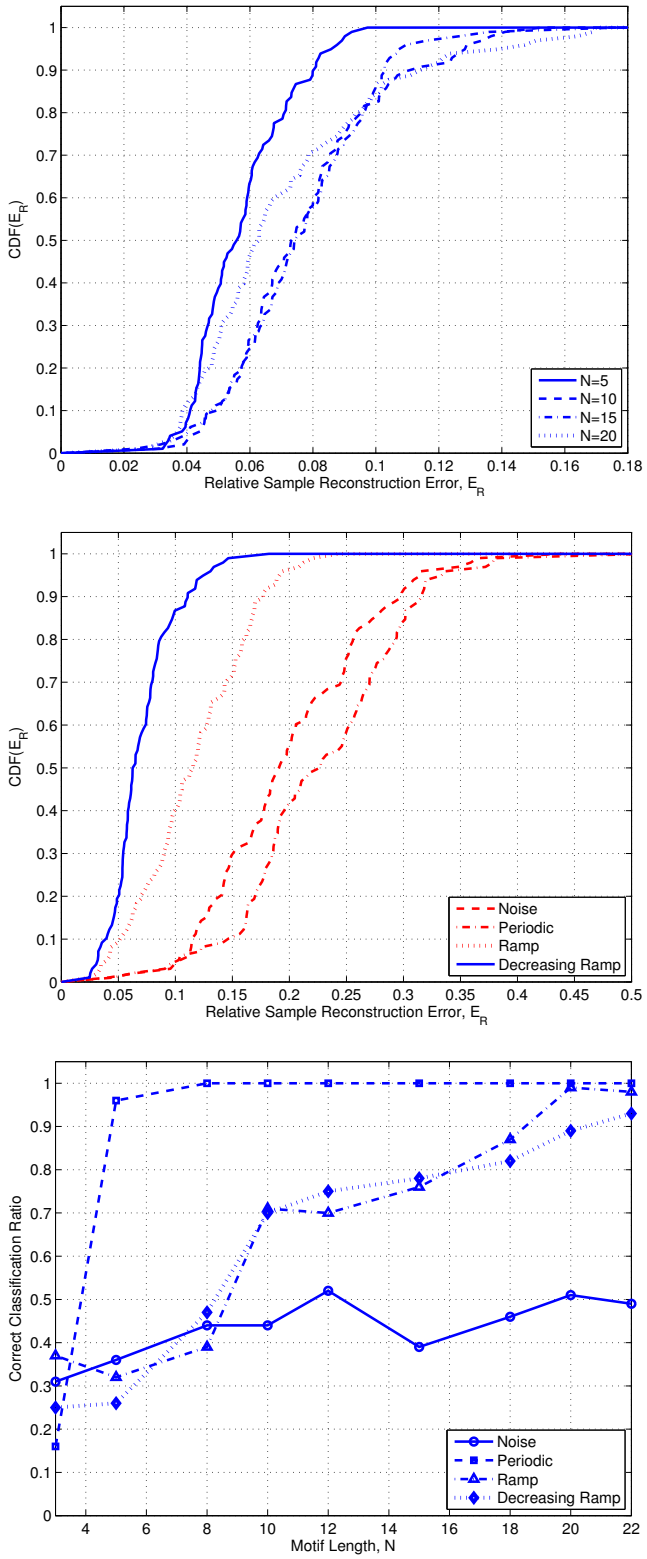



**Figure 3.4.** *Performance evaluation in terms of signal distortion and classification accuracy. 3.4(a) shows $E_R$ CDF of signals of the Decreasing Ramp category using Motifs generated from the same category varying $N$ with $b = 8$ and $K = 12$. 3.4(b) shows $E_R$ CDF of signals of all four categories using Motifs generated from the Decreasing Ramp category with $b = 8$, $N = 18$ and $K = 12$. 3.4(c) shows the correct classification ratio in the four categories varying $N$ with $b = 8$ and $K = 18$.*

In particular, Figure 3.4-(a) shows the cumulative distribution function of the relative representation error in the Decreasing Ramp category using Motifs obtained with the same category and varying $N \in \{5, 10, 15, 20\}$ for fixed $b = 8$ and $K = 12$. Figure 3.4-(b) shows the cumulative distribution function of the relative representation error in the four categories using Motifs obtained on a Decreasing Ramp signal (solid line for same category, other line types for other categories) for fixed $b = 8$, $N = 18$ and $K = 12$. Figure 3.4-(c) shows the correct classification ratio of the four categories varying $N$ with fixed $b = 8$ and $K = 12$.

Although, again, low $N$ results in the best $E_R$ distribution (Figure 3.4-(a)), it is for highest $N$ that the correct classification ratio approaches $1$ (Figure 3.4-(c)). This is further confirmed by Figure 3.4-(b), where the red dashed and/or dotted curves, obtained using the *wrong* Motif set, achieve the worst $E_R$ distributions. Moreover, the $E_R$ distribution for high $N$ is still quite close to the best achievable. This suggests that longer Motifs are better for combining classification and communication. In contrast with the good results obtained on Periodic, Ramp and Decreasing Ramp categories, our methods could never achieve correct classification ratios higher than $0.5$ for the Noise category: our conjecture is that our method fails in identifying recurrent patterns in noisy signals. Although this was somewhat foreseeable, we need to identify a priori which signal category we are able to handle with our solution, hence we will further investigate this phenomenon in future works.

A final confirmation of this is provided by Table 3.1, which reports a wide selection of results: the first three columns give the system parameters, $b$, $K$ and $N$; columns four to six report quality parameters, i.e., the memory space needed for Motif storage expressed in bits, Mem $= bNK$, $C_F$ and $E_R$; the last eight columns contain the correct classification and the false positive ratios. Rows are grouped by constant memory occupation. What is evident from the result comparison is that given a fixed amount of memory it is better to invest it on $N$ and $K$ than on $b$. Also, between $N$ and $K$, $N$ is to be preferred since, for comparable distortion rate, it provides better classification and compression.

## 3.6 Conclusions

In this chapter is proposed a novel lightweight approach capable of improving WS&AN data classification and communication by leveraging on Motifs (recurrent data features used

**Table 3.1.** *Comparative results varying the system parameters.*

| Parameters | | | Quality | | | Classification | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $b$ | $N$ | $K$ | Mem | $C_F$ | $E_R$ | Correct Classification Ratios | | | | False Positive Ratios | | | |
| 4 | 4 | 8 | 128 | 0.55 | 0.0360 | 0.36 | 0.60 | 0.27 | 0.30 | 0.26 | 0.13 | 0.13 | 0.10 |
| 4 | 8 | 4 | 128 | 0.27 | 0.0603 | 0.23 | 1.00 | 0.25 | 0.49 | 0.15 | 0.21 | 0.10 | 0.06 |
| 4 | 8 | 16 | 512 | 0.28 | 0.0454 | 0.43 | 1.00 | 0.51 | 0.53 | 0.16 | 0.10 | 0.09 | 0.04 |
| 4 | 16 | 8 | 512 | 0.14 | 0.0453 | 0.32 | 1.00 | 0.83 | 0.76 | 0.07 | 0.07 | 0.08 | 0.06 |
| 8 | 4 | 16 | 512 | 0.56 | 0.0261 | 0.35 | 0.85 | 0.12 | 0.54 | 0.15 | 0.10 | 0.08 | 0.20 |
| 8 | 16 | 4 | 512 | 0.13 | 0.0497 | 0.24 | 1.00 | 0.65 | 0.85 | 0.06 | 0.16 | 0.07 | 0.03 |
| 8 | 8 | 8 | 512 | 0.27 | 0.0507 | 0.44 | 1.00 | 0.35 | 0.41 | 0.17 | 0.15 | 0.08 | 0.04 |
| 8 | 16 | 8 | 1024 | 0.14 | 0.0454 | 0.39 | 1.00 | 0.62 | 0.75 | 0.09 | 0.10 | 0.08 | 0.04 |
| 8 | 8 | 16 | 1024 | 0.28 | 0.0448 | 0.40 | 1.00 | 0.43 | 0.42 | 0.18 | 0.09 | 0.12 | 0.05 |
| 4 | 16 | 16 | 1024 | 0.14 | 0.0424 | 0.36 | 1.00 | 0.78 | 0.92 | 0.06 | 0.04 | 0.08 | 0.06 |
| 8 | 32 | 8 | 2048 | 0.07 | 0.0507 | 0.44 | 1.00 | 0.35 | 0.41 | 0.18 | 0.15 | 0.08 | 0.04 |
| 8 | 8 | 32 | 2048 | 0.29 | 0.0401 | 0.43 | 1.00 | 0.47 | 0.55 | 0.17 | 0.06 | 0.12 | 0.05 |
| 8 | 16 | 16 | 2048 | 0.14 | 0.0422 | 0.42 | 1.00 | 0.80 | 0.85 | 0.08 | 0.03 | 0.08 | 0.05 |
| **4** | **32** | **32** | **2048** | **0.07** | **0.0340** | **0.44** | **1.00** | **0.97** | **0.99** | **0.01** | **0.04** | **0.07** | **0.04** |

to describe categories) to optimize communications, and by enhancing transmitted information to simplify data classification.

Our solution adopts Motifs to compress data streams: in such a way it is possible to achieve compression levels of up to an order of magnitude, while maintaining the signal distortion rate within acceptable bounds and allowing for simple lightweight distributed classification and anomaly detection.

We compared two different segment representation techniques, AV and NS, and two Motif extraction algorithms, BF and MB, proving the superiority of the combination of NS with MB by means of a thorough simulation campaign. Moreover, we are already studying how to implement an online Motif discovery and maintenance procedure based on [48].

Subsequently, we proposed a simple compression and communication solution leveraging on the extracted Motifs, which is able to achieve compression factors smaller than $1/10$,

with a distortion rate not higher than $5\%$ using as few as $256$ bytes of storage memory (the best result we obtained is highlighted in the table). These are just preliminary results, but they can easily lead to the implementation of online anomaly detection [43, 44] and prediction [45] solutions and to an improvement of the bandwidth usage efficiency.

In the following Chapter 4, the Matching Based algorithm will be improved and tested with differente distance metrics and amplitude gain functions. The improved new version will be compared against state-of-the-art compression algorithms suitable for WS&AN and a data mining solution based on the Motif concept to classify long time series.

Chapter 4

# Razor: an algorithm based on Motifs to compress and classify time series

## 4.1 Introduction

In the previous Chapter 3, it was demonstrated that *motifs* [33], i.e., recurrent patterns of a given time series, can be used to improve communications and to classify data.

We report another sentence that inspired us during the design of our lightweight algorithm: *Whenever possible, substitute constructions out of known entities for inferences to unknown entities* [49]. This quote from "The Logical Atomism" by Bertrand Russell is a modified version of the widely known Occam's Razor and is a good summary of the present work, in which, we propose a lightweight algorithm, that we named RAZOR, for data compression and classification for the Internet of Things (IoT) that consists of the integration of *smart* connected objects into the standard Internet.

RAZOR is developed to exploit historical data information to:

1. Compress IoT data using recurrent patterns of a specific physical phenomenon.
2. Classify IoT data according to the occurrence of the same recurrent patterns.

In this chapter, we further optimize our technique and provide a thorough performance evaluation, including a comparison with state-of-the-art compression and classification solutions. We also discuss the computational complexity of the RAZOR procedures and its impact on the energy consumption of constrained devices. Finally, in addition to a thorough

analysis on synthetic datasets, we test our solution against real data collected in an actual WS&AN deployment measuring temperature, humidity and light in a public building.

The main contributions of the chapter are the following:

1. Improved the lightweight data compressor and classificator presented in Chapter 3.1, called RAZOR;

2. We show that RAZOR is able to achieve satisfactory results in terms of both compression and classification when tested on both synthetic and real data;

3. We show that RAZOR can be successfully implemented on constrained devices (e.g., wireless sensor nodes), as it does not require a large amount of memory, nor a powerful processor;

4. We study RAZOR's applicability to multi-hop and lossy environments.

The rest of the chapter is organized as follows: Section 4.2 illustrates RAZOR's inspiration sources and the state-of-the-art algorithms we took as benchmarks; Section 4.3 describes RAZOR and its evolution from the previous Algorithm 2 presented in Chapter 3, highlighting the roles of the many parameters and variants; Section 4.4 reports the results obtained in our evaluation campaign providing insight on the best parameter selection, the performance comparison and the results on the real scenario; finally, Section 4.5 concludes the chapter.

## 4.2 Related work

In this chapter, we will compare the RAZOR's performance against the state-of-the-art compression for WSN and classification solutions. This section will list a subset of the most important algorithms. Several other algorithms to combine compression and communication can be found in the literature: the reader is referred to [38] for a detailed survey on in-network aggregation techniques and to [50] for studies on Compressive Sensing. Reference [41] studies time series compression and prediction quality in WS&ANs. However, all these techniques, while being very efficient at compressing information, do not provide any means to recognize and classify data sources. In addition, [51] provides one of the latest compendia on compression algorithms for constrained devices. Finally, the techniques we chose as a benchmark for RAZOR are: (1) lossy compression through the Discrete Cosine

Transform (DCT) [52]; (2) Lightweight Temporal Compression (LTC) [53]; and (3) Enumeration of Motifs through Matrix Approximation (EMMA) [33]. The motivations for such a selection are the popularity of DCT for lossy compression in audio and video application, the effectiveness, while maintaining a low computational complexity, of LTC and the efficiency of EMMA in classifying many different signal types.

Furthermore, exploiting the parallelism between bi-dimensional data maps and images, Wang [54] proposed the application of image compression techniques to WS&ANs. Starting from this idea and using Low Energy Adaptive Clustering Hierarchy (LEACH) [55] to partition the network into clusters, the cluster-based and robust information-driven architecture (RIDA [56]) is proposed to apply compression algorithms, such as DCT or Wagner-Discrete Wavelet Transform (Wagner-DWT), to logical maps of data collected within clusters in order to improve compression performance, communication efficiency and robustness.

## 4.3 The RAZOR algorithm

In this section, we will describe the algorithms composing RAZOR: we will start from the extraction of the Motif Codebook based on the Matching based presented in Chapter 3, and then we will discuss the usage of this Codebook(s) for compressing and identifying the original signal in Section 4.3.7.

### 4.3.1 Mathematical definitions

In this part is summarized some useful definitions already reported in Section 3.4 The original signal that we want to analyze is:

$$S(t) \in \mathbb{R}, t \in [0, +\infty) \tag{4.1}$$

The *training set*, of length $M_S$ is defined as:

$$T(t) = S(t), t \in [0, M_S] \tag{4.2}$$

Each segment is defined as:

$$T^{(i)} = [T_i, T_{i+1}, \ldots, T_{i+N-1}],\ i = 0, 1, \ldots, M - N + 1 \tag{4.3}$$

representing a set of $N$ consecutive samples of the training set starting from sample $T_i$.

In the following we will discuss the case when the training set can be composed of the inputs from different sources.

### 4.3.2 Segment normalization.

In previous Chapter 3 was analyzed two representation techniques for storing the Motifs of the Codebook. In particular, we found that our *Normal Shapes* representation technique, aimed at obtaining Motifs with zero mean and unit variance, were appropriate for our objectives of compression and classification for constrained devices.

Although in Section 3.4 we only considered the standard deviation (STD) (as in [32]) as the amplitude gain, here, we will extend our investigation to two more quantities:

1. the square root of the segment energy (GAIN), as defined in [29]; and
2. the maximum deviation (MAXABS).

In the following, we will indicate the Normal Shape of a given segment, $S^{(i)}$, as $\overline{S}^{(i)}$, which is obtained as follows:

$$\overline{S}^{(i)} = (S^{(i)} - O^{(i)})/G^{(i)} \tag{4.4}$$

where $O^{(i)}$ is the offset of segment $S^{(i)}$ and is computed as:

$$O^{(i)} = (1/N) \sum_{k=0}^{N-1} S_{i+k} \tag{4.5}$$

and $G^{(i)}$ is the amplitude gain of $S^{(i)}$ and can assume one of the three following expressions:

$$G^{(i)} = \begin{cases} \sigma_{S^{(i)}} & (STD) \\ \sqrt{(\sum_{k=0}^{N-1} S_{i+k}^2)/N} & (GAIN) \\ \max\limits_{k \in [0,N-1]} (|S_{i+k} - O^{(i)}|) & (MAXABS) \end{cases} \tag{4.6}$$

In the following, we will define the de-normalized version of a given segment or Motif with the subscript, $R$. For instance, the de-normalized version of $\overline{S}^{(i)}$, will be:

$$S_R^{(i)} = \overline{S}^{(i)} G^{(i)} + O^{(i)} = S^{(i)} \tag{4.7}$$

which is the same as the original version if no additional quantization is used.

We will provide a numerical comparison among the three techniques in Section 4.4.

As for standard Vector Quantization techniques, we now need to compare all the normalized segment couples, $\overline{T}^{(i)}$ and $\overline{T}^{(j)}$, in terms of their relative distance:

$$d(\overline{T}^{(i)}, \overline{T}^{(j)}) \in \mathbb{R} \tag{4.8}$$

where $d(\cdot)$ is the adopted dissimilarity function.

### 4.3.3 Dissimilarity functions

In this chapter, to compute the distance between two segments of a signal, we take into consideration the following dissimilarity functions: Euclidean distance, infinity norm, minimum distance (MINDIST) [34], Compression-based Dissimilarity Measure (CDM) [36] and DTW [46]. With respect to previous Chapter 3, we added the infinity norm, which has two interesting properties for RAZOR:

1. it provides an upper bound on the distance between two samples;
2. it is computationally simpler than the Euclidean distance.

We already dismissed MINDIST, since it is specific for piecewise aggregate approximation (PAA), CDM, since it is only efficient for large files or for a large number of samples, and DTW, since it is too computationally demanding for our reference hardware. Hence, in this work, our dissimilarity functions will be $L2$ and the infinity norm, and we will compare RAZOR's performance adopting these two distances in Section 4.4. Thus:

$$d(\overline{T}^{(i)}, \overline{T}^{(j)}) = \begin{cases} \|\overline{T}^{(i)} - \overline{T}^{(j)}\|_2 \\ \|\overline{T}^{(i)} - \overline{T}^{(j)}\|_\infty \end{cases} \tag{4.9}$$

where the two norms are applied as follows:

$$\begin{aligned} \|\overline{T}^{(i)} - \overline{T}^{(j)}\|_2 &= \sqrt{\textstyle\sum_{k=0}^{N-1} |\overline{T}_{i+k} - \overline{T}_{j+k}|^2} \\ \|\overline{T}^{(i)} - \overline{T}^{(j)}\|_\infty &= \max_{k\in[0,N-1]}(|\overline{T}_{i+k} - \overline{T}_{j+k}|) \end{aligned} \tag{4.10}$$

### 4.3.4 Codebook selection

In the following, we will summarize our Codebook selection algorithm highlighting the differences from the previous version. In Section 3, we analyzed two different algorithms for the selection of the Motifs to create the Codebook. In particular, we found that our

*Matching-Based* approach, a lightweight Motif selection algorithm inspired by data mining applications [34], is appropriate for our objectives of compression and classification for constrained devices.

The basic idea for creating the Codebook is that of selecting those Motifs that have the most frequent shape among the normalized segments in the training set. In order to run our algorithm, which is listed in Algorithm 4, we first need to compute the matrix of the distances among each normalized segment couple, $\mathbf{DM}$, whose elements are $DM_{ij} = d(\overline{T}^{(i)}, \overline{T}^{(j)})$. At the same time, we compare the computed distance with a threshold, $d_{th}$, representing the maximum acceptable distortion, due to compression (depending on the adopted dissimilarity metric, this threshold may take a different meaning; for instance, in the case of the infinity norm, it is the maximum relative deviation of the decompressed sample compared to the original one). When the distance between two segments $i$ and $j$ is lower than the threshold, we say that the two segments *match*. Hence, we can write the matching matrix $\mathbf{MM}$, whose elements are $MM_{ij} = I(DM_{ij} < d_{th})$, where $I(\cdot)$ is the indicator function. For the objective, the best segment to be included in the Codebook is the one obtaining the highest number of matchings and that is representative of all the segments it matches with, since their distance is lower than the threshold. Hence, our algorithm iterates, until either the training set is empty or the desired Codebook size, $K_{target}$, is reached by selecting at each step the best segment and deleting it and all the segments that match with it from $\mathbf{DM}$ and $\mathbf{MM}$. In case of ties, when two or more segments have the same number of matches, our algorithm picks that with the smallest average distance from the segments for which it is representative. The selected Motifs will be referred to as $X(k), k \in \{1, \ldots, K\}$ and in the Codebook, the segments are stored in their normalized version, as they take values in $[-1, 1]$.

Note that in the previous version of the Algorithm 2 we chose not to fix $d_{th}$, but to let it vary as a function of the desired Codebook size. In this chapter, instead, we preferred to have $d_{th}$ fixed to ease the comparison between RAZOR and other compression algorithms from the literature. However, this choice implies that, depending on the original signal variability, the desired Codebook size may be either higher or lower than the actual number of Motifs needed for a complete representation of the training set. In turn, this translates in either a distortion higher than $d_{th}$ or a suboptimal usage of the Codebook size. In fact, since

**Algorithm 4** : Matching-Based Motif extraction algorithm.

**Require:** **DM**, the precomputed distance matrix with $d(\cdot)$, a dissimilarity metric; $d_{th}$, the threshold for the matching definition; $K_{target}$, the maximum size for the Codebook.

**for all** $m, n$ **do**

$MM_{mn} \leftarrow I(DM_{mn} < d_{th})$

**end for**

$k \leftarrow 1$

**while** **DM** $\neq \emptyset$ and $k \leq K_{target}$ **do**

**for all** $m$ **do**

$MC_m \leftarrow \sum_n MM_{mn}$

**end for**

$i \leftarrow \arg\max_m(MC_m)$

$X(k) \leftarrow T^{(i)}$

delete the $l$-th rows and columns in **DM** and **MM** for which $MM_{il} = 1$

$k \leftarrow k + 1$

**end while**

$K \leftarrow k$

the compressed signal is sent by using Motif indices, a Motif belonging to a 16-element Codebook needs four bits to be sent; however, if the Codebook has only 9–15 Motifs, its indices still need four bits, which, however, are used less efficiently.

Finally, we fixed the threshold $d_{th} = 16\%$, because, based on our results in Section 3.5, we found that too small a threshold would lead to a larger number of segments having the same number of matchings, thus forcing the algorithm to pick a large number of Motifs at the beginning of the training set without selecting segments over the whole time series, while too large a threshold would lead to an increased number of matchings for each segment, thus forcing the algorithm to select a limited number of Motifs; $d_{th} = 16\%$ was found to be a good trade off value capable of building codebooks with $K \leq 64$ (small enough) and a good distribution of the selected segments within the whole signal (large enough).

### 4.3.5 Motif representation

Finally, in order for the Codebook to maintain a limited size, we quantize the Motifs $X(k)_Q = Q(X(k))$ using an alphabet of $2^b$ symbols before saving them in the Codebook. Given our previous decompression procedure, we found that increasing $b$ above eight provides minimal benefits, while using a $b$ as low as four will further reduce the Codebook size, while, at the same time, maintaining good performance.

### 4.3.6 Multiple signal sources

Usually, in the IoT, multiple devices are in charge of delivering readings of the same phenomenon to a shared gateway. In this case, the description above holds with the following variation:

$$\begin{aligned} &S_n(t) \in \mathbb{R}, t \in [0, +\infty), n \in \mathcal{N} \\ &S_{i,n} = Q(S_n(iT)) \in \mathcal{A}, i = 1, \ldots, M, n \in \mathcal{N} \end{aligned} \tag{4.11}$$

which are the signal and the related quantized sample read by device $n$ of the set $\mathcal{N}$ of the nodes of the network. Thus, the training set, $T$, becomes:

$$T = \bigcup_n S_{i,n}, i = 1, \ldots, M, n \in \mathcal{N} \tag{4.12}$$

where the $\bigcup(\cdot)$ operator stands for the concatenation of the different inputs. Moreover, the gateway must avoid creating a segment from samples belonging to different devices, because this would lead to physically infeasible signal shapes.

### 4.3.7 Motif communication

As described in Section 3.3, IoT typical environments consist of many constrained devices and a few more powerful devices in charge of gathering data from the constrained devices and sending them towards the final destination. Due to the different computational capabilities, we designed RAZOR to perform the processing intensive operations (e.g., the Motif extraction algorithm) at the gateway, while simple devices only need to use the Codebook computed by the gateway to compress their data.

The gateway, after receiving the needed amount of uncompressed data, runs the Motif extraction algorithm, which eventually generates a Codebook that is representative for all the devices that contributed to the training set. For the signal we used to test RAZOR, we

found that a few hundred samples (e.g., $300$) were sufficient to discover the most significant patterns. However, this depends on the actual signal under analysis and must be dealt with carefully when applying the compression algorithm to unknown information sources.

A node will continue to send uncompressed data until the gateway distributes the Codebook to the network, but after its reception, each node is able to exploit the received information to compress $N$ consecutive samples, sending the index of the best Motif representing them. Note that these indices need $\log_2(K)$ bits to be sent, but, since the Codebook contains normalized Motifs, nodes have to also send offset and gain values along with the Motif index to allow the receiver to decompress the signal properly.

Depending on the application accuracy requirement, it is possible to further quantize the offset and gain values with $b_O$ and $b_G$ bits, respectively. The quantized version of offset and gain are defined as $\overline{O} \in [\min(S), \max(S)]$ and $\overline{G} \in [\min(G), \max(G)]$, respectively ($\min(S), \max(S), \min(G)$ and $\max(G)$ are computed on the training set by the gateway and sent along with the Codebook). As a result, the number of bits needed to send $N$ consecutive samples is the sum of $\log_2(K)$, $b_O = 8$ and $b_G = 8$ instead of the actual values of the samples that, for typical applications, may need up to $16$ bits each, for a total of $16N$ bits.

A final consideration about Motif communications is related to packet losses. As in all compression techniques, each single packet carries a lot more information than uncompressed data. Although we are aware that packet losses can heavily impact RAZOR's performance, here we aimed at comparing RAZOR's effectiveness at the application layer with the state-of-the-art compressors and classificators for WSAN. Although a thorough study of RAZOR in multi-hop lossy networks and of how to improve its robustness is left as an interesting subject of further investigation, in this chapter we provide some evaluations, where the effects of packet losses and multi-hop communications are assessed.

**Motif selection.** Here the Algorithm 5 is a generic version of Algorithm 3 to select the best Motif that substitute a segment $S^{(i)}$:

$$\widehat{X}^{(i)} = X_R(l),\ l = \arg\min_k d(S^{(i)} - X_R(k)) \tag{4.13}$$

where $k, l \in \{1, \ldots, K\}$, $i \in 0, 1, \ldots, M - N + 1$ and $X_R(l) \in \mathbb{R}$ is the de-normalized version of $X(l)$, the $l$-th Motif in the Codebook and $G$ is the amplitude gain function selected for RAZOR.

**Algorithm 5** : Motif selection algorithm.

**Require:** $B$, the Codebook of the Motifs; $S^{(i)}$, the segment to be compressed; $d(\cdot)$, a dissimilarity metric; $d_{th}$, the compression distortion threshold.

$d_{min} \leftarrow \infty$

$\overline{O} \leftarrow Q((1/N)\sum_{k=1}^{N} S_{i+k-1})$

$\overline{G} \leftarrow Q(GAIN(S^{(i)}))$

**for all** $X(k) \in B$ **do**

$X_R(k) \leftarrow X(k)\overline{G} + \overline{O}$

**if** $d(S^{(i)}, X_R(k)) < d_{min}$ **then**

$\widehat{X}^{(i)} = X_R(k)$

$d_{min} = d(S^{(i)}, X_R(k))$

**end if**

**end for**

Return $k$

Since we are dealing with constrained devices, it is important to limit the computational complexity of the procedure applied to select which Motif $X(k)$ to use to compress the segment $S^i$. Later in this Section, it will be presented an its improvement to reduce the computational complexity to limit the energy consumption to find a Motif that match with $S^{(i)}$.

Although a more thorough analysis on the computational cost of this operation will be provided in Section 4.4, we will give here some insight on the processing requirements. In order to find the best Motif, the distance metric has to be computed $K$ times. Hence, the more complex the evaluation of the distance and of the gain, the more computational demanding the process. However, since our Codebook is ordered according to the number of matchings obtained by each Motif in the training set, it is more likely that the best Motif is one of the first in the Codebook. Hence, it is possible to stop the search as soon as the computed distance is lower than the given compression distortion threshold, $d_{th}$. Note that stopping the algorithm before the search is finished decreases the computational complexity, but might not find the best Motif in the Codebook, as just the first Motif satisfying the requirement is picked.

Moreover, in order to provide a hardware-independent computational complexity evaluation, we note that our algorithm depends on two parameters: $N$ and $K$. In particular,

RAZOR's Motif selection, in the worst case, needs to perform $N$ distance computations between each of the $K$ Motifs in the dictionary and the segment under analysis. Considering a single segment, the complexity is equal to $\mathrm{O}(KN)$, and in order to compare RAZOR with other compress techniques, we need to consider the entire signals of length $M$. Thus, a signal of $M$ samples is divided in segments of $N$ samples; then, considering the complexity for a single segment, the total computational complexity is equal to $\mathrm{O}(KM)$ and does not depend on any network parameter. Table 4.1 reports the complexity for each of the techniques considered.

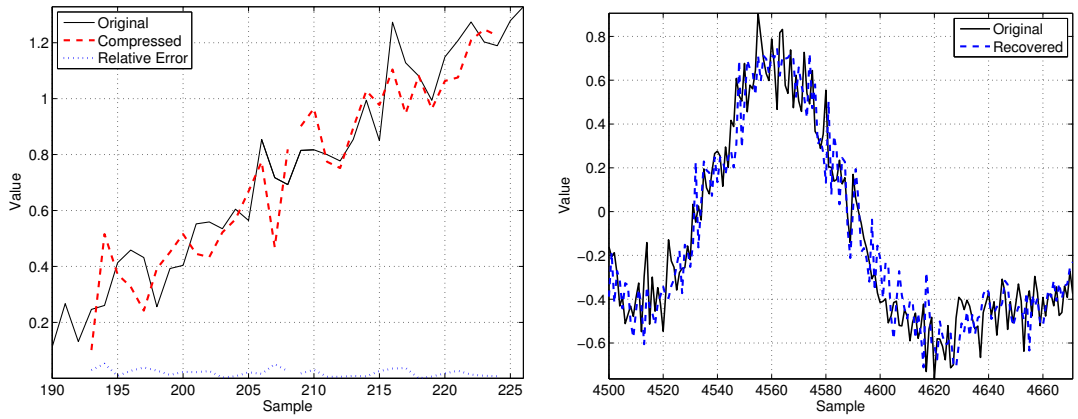
**Table 4.1.** *Computational complexity. DCT, Discrete Cosine Transform; LTC, Lightweight Temporal Compression (LTC); Robust Information-Driven Data Compression Architecture (RIDA)*

| **RAZOR** | **DCT** | **LTC** | **RIDA** |
|---|---|---|---|
| $\mathrm{O}(KM)$ | $\mathrm{O}(MlogM)$ | $\mathrm{O}(M)$ | $\mathrm{O}(M)$ |

After the Motif selection phase, the node is able to send $k$, the index of $\widehat{X}^{(i)}$, using $\log_2(K)$ bits, $\overline{O}$, the quantized value of the offset using $b_O$ bits, and $\overline{G}$, the quantized value of the gain using $b_G$ bits. At the receiver side, the decompressed segment, $S_D$, is computed as:

$$S_D = X_R(k) = X(k)\overline{G} + \overline{O} \tag{4.14}$$

Figure 4.1 shows a qualitative example of the decompression procedure for $N = 16$, $K = 8$ and $b = 8$: on the left side, Figure 4.1-(a) illustrates two consecutive segments (thin black solid line) of $N$ samples, as well as the related compressed values (bold red dashed lines) and the relative compression error (thin blue dotted lines in the bottom part of the chart), while on the right side, Figure 4.1-(b) shows the result of the decompression operation comparing a longer segment of the original signal (thin black solid line) with the received data after the decompression (bold blue dashed line). These figures have been chosen to provide graphical examples of what a Motif looks like and how the re-constructed signal compares to the original one.




(a) Compression (b) Reconstruction

**Figure 4.1.** *A qualitative example of the transmission procedure for $N = 16$, $K = 8$ and $b = 8$: (**a**) shows two consecutive segments (thin black solid line) of $N$ samples, as well as the related compressed values (bold red dashed lines) and the relative compression error (thin blue dotted lines in the bottom part of the chart); (**b**) shows the result of the decompression operation comparing a longer segment of the original signal (thin black solid line) with the received data after the decompression (bold blue dashed line).*

**Motif prediction.** After the Codebook has been computed, it is possible to perform an additional analysis, which will enhance RAZOR's performance. Since the training set is a realization of the physical phenomenon under study, it is possible to compress it with the Codebook and study the sequence of the best Motif used for each segment.

In particular, we can count how many times a given Motif, $X(l)$, is used given that another Motif, $X(k)$, has been used for the previous segment. In such a way, we build the prediction matrix, **PM**, so that the element $PM_{kl}$ is the probability that $X(l)$ is the best Motif after $X(k)$:

$$PM_{kl} = Prob\{\widehat{X}^{(i)} = X(l) | \widehat{X}^{(i-N)} = X(k)\} \tag{4.15}$$

where $\widehat{X}^{(i-N)}$ is the best Motif at the previous, $i - N$-th step of the algorithm. This information can be exploited to order the Motifs in the selection algorithm, so that they are tested in order of decreasing probability of having $X(l)$ as the best Motif given that $X(k)$ was selected for the previous segment. Algorithm 6 provides the listing of the Motif selection process with prediction.

**Algorithm 6** : Motif selection algorithm with prediction.

**Require:** $B$, the Codebook of the Motifs; $S^{(i)}$, the segment to be compressed; $d(\cdot)$, a dissimilarity metric; $d_{th}$, the compression distortion threshold; $\mathbf{PM}$, the prediction matrix; $\widehat{X}^{(i-N)}$, the previous best Motif.

$d_{min} \leftarrow \infty$

$\overline{O} \leftarrow Q((1/N)\sum_{k=1}^{N} S_{i+k-1})$

$\overline{G} \leftarrow Q(GAIN(S^{(i)}))$

sort $B$ according to the row of $\mathbf{PM}$ related to $\widehat{X}^{(i-N)}$

**for all** $X(k) \in B$ **do**

  $X_R(k) \leftarrow X(k)\overline{G} + \overline{O}$

  **if** $d(S^{(i)}, X_R(k)) < d_{min}$ **then**

    $\widehat{X}^{(i)} = X_R(k)$

    $d_{min} = d(S^{(i)}, X_R(k))$

  **end if**

  **if** $d_{min} \leq d_{th}$ **then**

    Break

  **end if**

**end for**

Return $k$

Note that the benefits provided by this improvement are two-fold: on the one hand, it is possible to further reduce the number of bits for the representation of the compressed segment, and on the other hand, it is more likely that the selection algorithm stops before the end of the search operation.

The former advantage is justified by adding another bit to those sent by the compressing node. This bit is a flag, $F$, of size $b_F = 1$ denoting whether or not the sent Motif is the first, $F = 1$, according to the order suggested by $\mathbf{PM}$, *i.e.*, in other words, the *predicted* Motif. If it is, it is possible not to send the index of the Motif itself, because the receiver can infer it from $\mathbf{PM}$; conversely, if the prediction is wrong, the Motif index is sent, and the flag is set

to zero, $F = 0$. Hence, the number of bits sent, $b_S$, becomes:

$$b_S = \begin{cases} b_F + \log_2(K) + b_O + b_G & \text{if } F = 0 \\ b_F + b_O + b_G & \text{if } F = 1 \end{cases} \tag{4.16}$$

Although it is not assured that the predicted Motif is the best, it is sufficient for this to happen with a frequency, $f$, higher than $1/\log_2(K)$ (e.g., for $K = 8$, the prediction needs to succeed once every three segments).

The latter benefit, instead, is motivated by observing that **PM** provides the marginal distribution of the best Motif given the Motif selected at the previous step. Hence, the selection algorithm is more likely to end sooner, because the most probable Motifs are considered first. Note that this method is equivalent to encoding the Codebook using a Huffman code, where only the most probable Motif is known and all the others are assumed equiprobable; we choose not to use a complete Huffman code for the Codebook, because that would have required a much longer training set than what we used.

**Final considerations.** Two final aspects are to be described in this section, before starting the evaluation of the solution: the numerical precision used within the online algorithm running on the nodes and the usage of different Codebooks to identify which physical phenomenon a given sensor is measuring.

The first topic has an impact on both the decompression distortion and the computational cost: in fact, the use of floating point numbers will dramatically increase the accuracy in the analysis phase, but has almost no impact on the decompression operation, since this process is dominated by the quantization of the offset and the gain. At the same time, imposing a distortion threshold, $d_{th}$, makes it useless to obtain a representation of the decompressed Motif more accurate than the threshold used during the extraction phase. Moreover, since floating point operations cost one or two orders of magnitude more than fixed point operations, the latter are to be preferred for constrained device implementation.

Finally, RAZOR, as introduced before, can be used to tell which physical phenomenon a given sensor is measuring. To do so, the simplest procedure is to obtain the Codebooks related to the variety of phenomena of interest for the considered application and, then, use each of the Codebooks to compress a given training set of the data produced by the sensor

under analysis. Data is more likely to belong to the phenomenon whose Codebook obtains the smallest distortion.

Even when the signal classification should fail, our technique lets the system find the best Codebook from the known ones to compress and transmit the output of an unknown device, which is still a desirable feature.

## 4.4 Results

This section focuses on the evaluation of the proposed solution and its comparison with state-of-the-art techniques for both compression and classification. The section is split into four parts: Section 4.4.1 provides the definition of the evaluation metrics, as well as a few additional concepts used in the rest of the section for the evaluation; Section 4.4.2 evaluates our solution alone, analyzing the impact of the various setup parameters; Section 4.4.3 extends the previous evaluation comparing our solution against other signal processing techniques; an analysis of the performance in a real deployment scenario is provided in Section 4.4.4.

As to the datasets used in our evaluation, we considered three different databases:

1. The first dataset is obtained with a synthetic signal generator [57], configurable in terms of noise and correlation length. We let the correlation length vary between $20$ and $100$ samples, and we will refer to this set as the *SYN* dataset. This dataset has been used to test the compression capabilities of the three algorithms with different noise levels, *i.e.*, changing the correlation length of the signal. EMMA has not been tested on this signal, because all the generated synthetic signals were likely to be classified in the same category, as they are intended to reproduce the same phenomenon.

2. The second dataset is also synthetic and is taken from the University of California Riverside (UCR) Time Series Classification/Clustering Homepage [47]: in particular, we tested our algorithms on the Synthetic Control dataset only, because it is the closest to IoT requirements. This dataset, referred to as *UCR*, has been used to compare RAZOR's and EMMA's classification performance. Instead, we could not use this set to test the compression performance, because it consists of fairly short sequences.

3. The last dataset that we used for testing is obtained from a real deployment, where we

measured usual IoT readings, such as temperature, humidity and light. We refer to this dataset as *REAL*. We used this dataset to test RAZOR's compression performance on actual measurements, but we omit showing the comparison with the other algorithms for this dataset, because the obtained results were almost exactly the same as those obtained for the SYN dataset.

The algorithms have been studied using Matlab 2012. Each simulation has been performed varying each tunable parameter on a Sun Grid Engine (SGE) managed cluster with 100 CPUs on 16 racks with 16 GB RAM each. SGE has the purpose of scheduling the submitted simulations into different CPUs and to report the output of each simulation.

### 4.4.1 Evaluation metrics

As in previous Chapter 3, the metrics considered to evaluate our technique are the compression rate, the signal distortion introduced as a consequence of the lossy compression and the classification accuracy. In addition, we will evaluate the processing cost and the energy expenditure on reference constrained hardware.

**Compression.** Although many definitions can be applied to quantify the compression effectiveness of an algorithm, we define $C$ as the ratio of the compressed size to the original data size, which gives a simple formulation for deriving the number of bits to be sent. In its most basic formulation (see Section 4.3) and assuming that the uncompressed data are represented using $16$ bits per sample, the compression rate is given by the following deterministic formula:

$$C = (\log_2(K) + b_O + b_G)/(16N) \tag{4.17}$$

where $K$ is the number of motifs in the Codebook, $b_O$ and $b_G$ are the numbers of bits needed to represent the offset and the gain of the compressed segment, respectively, and $N$ is the length of the segment in samples. The evaluation of the compression achieved using the prediction enhancement also depends on the success rate, $f$, of the motif prediction: in fact, in the case of a successful prediction, only one bit more than those for the gain and offset is needed, while in the negative case, all the information is to be sent, plus the additional bit needed for the prediction outcome. Hence, the compression rate becomes:

$$C = \big[f(1 + b_O + b_G) + (1 - f)(1 + \log_2(K) + b_O + b_G)\big]/(16N) \tag{4.18}$$

**Distortion.** In order to avoid confusion between the symbols used for errors and energy later on, we prefer to indicate the distance between the decompressed and the original signal samples as distortion. Furthermore, we will provide three different measures for the distortion: the maximum distortion, $D_M$, the average distortion, $D_A$, and the root mean square distortion, $D_R$. The three quantities are defined as follows:

$$\begin{aligned} D_M &= \max_i |S_{D,i} - S_i|/A, \\ D_A &= (1/M) \sum_{i=1}^{M} |S_{D,i} - S_i|/A, \\ D_R &= (1/A)\sqrt{(\sum_{i=1}^{M}(S_{D,i} - S_i)^2)/M}, \end{aligned} \tag{4.19}$$

where $S_{D,i}$ and $S_i$ are the $i$-th samples of the decompressed and the original signals, respectively, $A$ is the amplitude of the original signal and $M$ is the length of the signal.

**Complexity and energy consumption.** When dealing with constrained devices, it is of paramount importance to maintain the computational complexity limited and, in particular, to ensure that the improvement provided by the technique does not require too much energy to be achieved.

For this reason, in Section 4.3.7, we have considered a hardware-independent metric (the complexity of the algorithm in terms of number of operations), which is useful for comparing different solutions, regardless of the specific device used. Here, in order to provide a more detailed assessment of the complexity and energy cost of our technique, we will consider two additional metrics that depend on the specific hardware used, namely, the number of CPU cycles needed to compress a sample, $P$, and the amount of energy spent to compress and transmit a sample, $E$. Finally, we will use as a quality indicator for a given technique the ratio, $R$, of the difference between the energy spent by the analyzed technique and the baseline energy expenditure (*i.e.*, not compressing the signal) to the baseline expenditure itself. $R > 0$ translates into a proportional increase of the energy needed using the procedure, while, *vice versa*, $R < 0$ stands for a proportional energy saving.

The definitions of the aforementioned quantities are as follows: $P$ is obtained as the ratio of the total number of CPU cycles needed for a given technique to process a signal, $C_{CPU}$, to the number of samples of the signal, $M$; while $E$ is calculated as the ratio of the sum of the number of bits sent times $E_{RF}$ (the energy needed to send a bit) and $C_{CPU}$ times the energy needed for a single CPU cycle, $E_{CPU}$, divided by $M$. Finally, the baseline energy

expenditure, $E_B$, used to compute $R$ is that of the telosb [27] wireless sensor architecture, because it is one of the most popular IoT architectures currently used by researchers. The three quantities can be summarized as follows:

$$\begin{aligned} P &= C_{CPU}/M \\ E &= (16ME_{RF} + C_{CPU}E_{CPU})/M \\ R &= (E - E_B)/E_B \end{aligned} \tag{4.20}$$

Telosb devices are equipped with a Texas Instrument (TI) CC2420 [58] radio transceiver, which spends $E_{RF} = 0.23\mu$J per transmitted bit and with a TI MSP430 [59] that has an average consumption of $E_{CPU} = 0.726$ nJ per cycle. For every compression method, we have recorded the number of operations to process the original time series, accounting for the number of additions, subtractions, multiplications, divisions, powers, square roots, assignments, comparisons and absolute value operations. Thus, we have mapped these figures into the corresponding number of clock cycles, and we have subsequently mapped the latter into the corresponding energy expenditure: as a reference, we listed the number of processor cycles needed for each operation in Table 4.2 (the operator $\gtrless$ stands for comparisons between the two operands), where we gave both fixed- and floating-point costs. Note that, in our experiments, we accounted for fixed-point operations for all the algorithms.

**Table 4.2.** *Telosb operation costs.*

| | $a+b$ | $a-b$ | $ab$ | $a/b$ | $a^b$ | $\sqrt{a}$ | $a \leftarrow b$ | $a \gtrless b$ | $\lvert a \rvert$ |
|---|---|---|---|---|---|---|---|---|---|
| Fixed point | 4 | 4 | 15 | 23 | 15 | 250 | 1 | 4 | 4 |
| Floating point | 184 | 187 | 395 | 405 | 395 | 720 | 10 | 37 | 184 |

**Classification accuracy.** Our last evaluation metric is related to the classification performance. In order to compute it, we first have to define how the classification procedure is performed by our solution.

Our assumptions are that this procedure is performed by the gateway and that some different Codebooks related to a variety of physical phenomena against which to test a given signal are available.

In particular, the classification is performed in a similar fashion as the construction of the prediction matrix: in fact, after the reception of the needed number of uncompressed sam-

ples, the gateway tries to compress them according to all the Codebooks available and analyzes the resulting distortion. If the distortion due to compression exceeds a given threshold (note that this threshold is higher than that used in the motif extraction phase, since the system must tolerate some variability), the signal is assumed not to belong to the reference category. In case multiple Codebook distortions do not exceed the threshold, the signal is assumed to belong to that category whose Codebook obtained the smallest distortion.

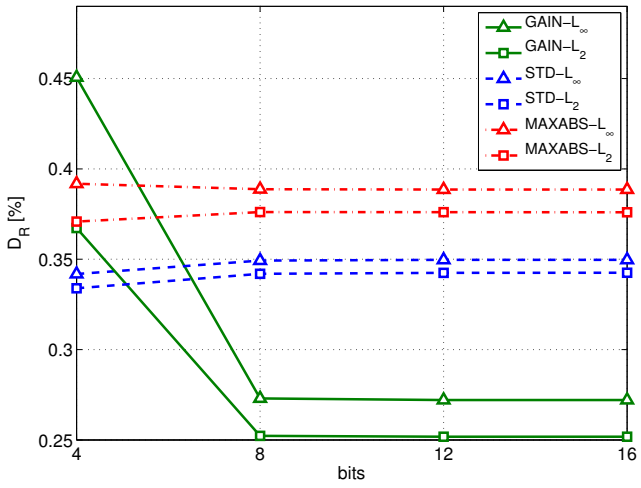
To evaluate the accuracy of such classification, we take into account three quantities: $CC$, the correct classification frequency; $EC$, the erroneous classification frequency (a signal belonging to the category under examination is classified, instead, in another category); and $FP$, the false positive frequency (a signal classified in the category under examination belongs to another category).

### 4.4.2 Parameter analysis

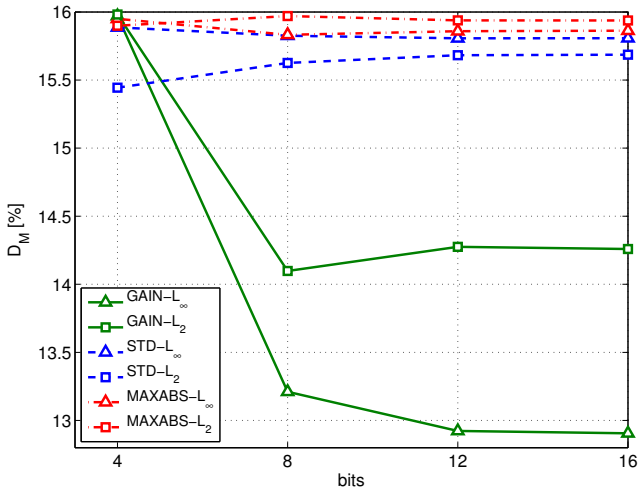
The first set of results is related to the selection of RAZOR's parameters. This selection has been made using the SYN dataset and spanning the whole parameter space. However, in what follows, we will study the impact of one parameter at a time in order to simplify the discussion.

We start our analysis by discussing the effectiveness of the prediction phase of the RAZOR algorithm: we evaluated RAZOR with prediction with respect to the standard version in terms of the energy saving percentage and the distortion loss for $N = 16, \ldots, 32$ and $K = 8, \ldots, 32$. The obtained results (not reported here) showed that guessing the most probable motif given the motif chosen at the previous step can save about $10\%$ of the energy expenditure, sacrificing less than $1.5\%$ of the distortion.

Figure 4.2 shows the effect of the number of bits used on the compression distortion. In particular, Figure 4.2-(a) provides the Root Mean Square Error (RMSE) for different dissimilarity metric and gain combinations, while Figure 4.2-(b) provides the maximum error. The Codebook has been computed using $K = 16$ motifs of $N = 20$ samples and $d_{th} = 16\%$ of the original segment amplitude. In the figures, the GAIN, STD and MAXABS techniques are plotted with green solid lines, blue dashed lines and red dash-dotted lines, respectively. Instead, triangular and square markers represent the infinity and $L2$ norm, respectively.




(a) Root mean square error, $D_R$

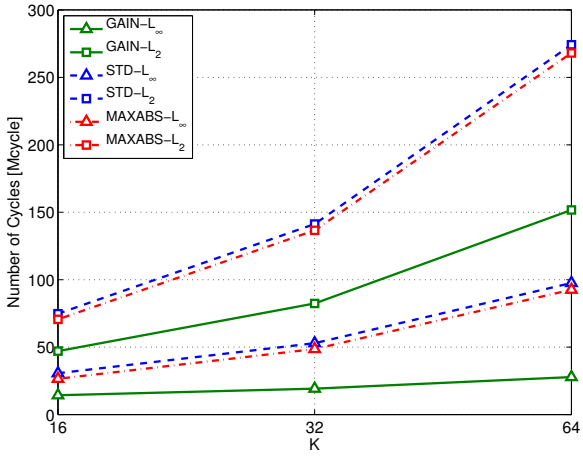



(b) Maximum error, $D_M$

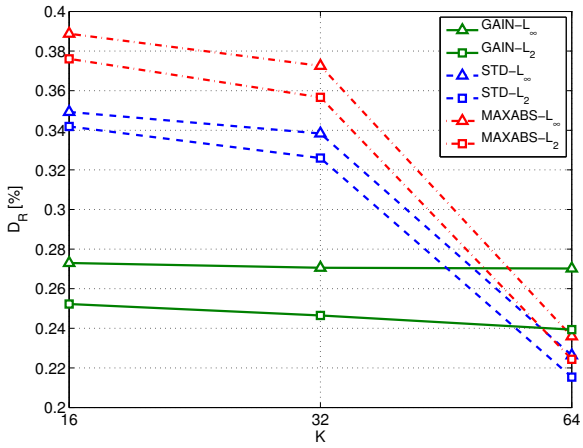
**Figure 4.2.** *Analysis of the impact of the number of bits used on the compression distortion: (**a**) shows the RMSE of the different dissimilarity metric and gain combinations; while (**b**) provides the maximum error on the same combinations. The Codebook has been computed using $K = 16$ motifs of $N = 20$ samples.*

The results show that it is not useful to increase the number of bits, $b$, beyond eight, because both distortion figures do not show proportional improvements, while the memory occupation of the Codebook increases linearly with the number of bits.

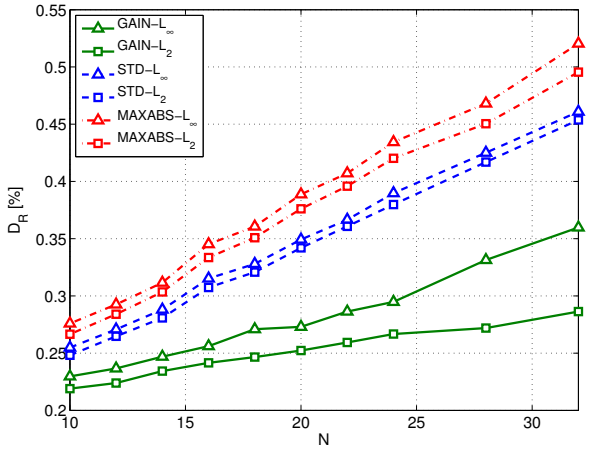
To complete the analysis on the gain and the dissimilarity metrics, RAZOR needs to be tested varying $N$ and $K$, while keeping fixed $b = 8$. Figure 4.3 provides the results we obtained to conclude the selection of the gain and the distance to be used.




(a) Cycles per sample




(b) Root mean square error varying K




(c) Root mean square error varying N

**Figure 4.3.** *Analysis of the impact of the gain formula and of the norm used on the computational cost ((**a**), with $N = 20$ and varying $K$) and on the RMSE varying the number of motifs, $K$ with $N = 20$ (**b**) or the motif length, $N$, with $K = 16$ (**c**). All graphs are plotted for $b = 8$.*

In particular, Figure 4.3-(a) shows the cycles per sample performed by the device CPU to compress the signal; this figure has been obtained for $K = 16$, 32 and 64 and $N = 20$. Note that the values picked for $K$ are those that correspond to an integer value for $\log_2(K)$. Similarly, Figure 4.3-(b) provides the root mean square error using the same parameters.

Conversely, Figure 4.3-(c) lets the motif length, $N$, vary while keeping fixed $K = 16$. The line styles and markers follow the same rules as above. (The number of cycles per sample is not included in the figures, because we found that it is not affected by $N$.)

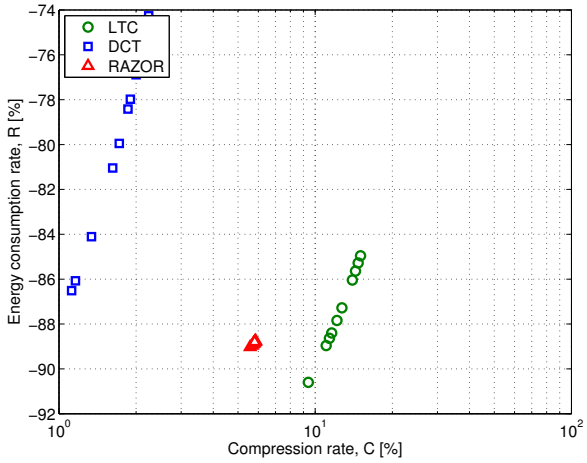
From all these graphs, we can infer that, although it does not provide the best distortion, the combination of the GAIN formulation and the infinity norm obtains the best tradeoff between computational complexity and distortion. In what follows, the RAZOR algorithm will, therefore, always use the GAIN formulation, the infinity norm and eight bits for storing motifs in the Codebook.

As a final outcome of this parameter impact evaluation, we decided to use RAZOR with the prediction enhancement activated and fixing $N = 20$ and $K = 16$, because they grant good performance, while maintaining the size of the Codebook limited and below 500 bytes.

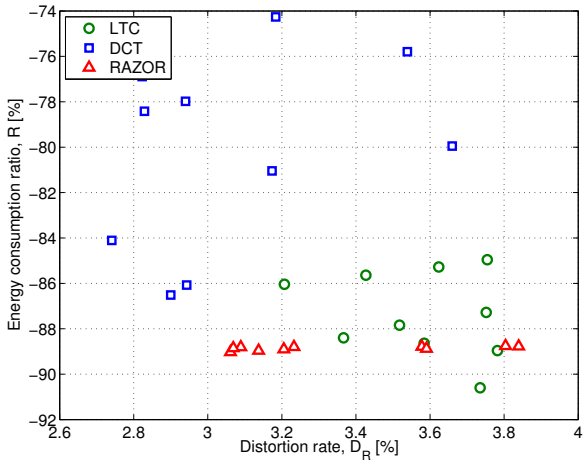
### 4.4.3 Comparison of the techniques

In order to compare RAZOR against the other compression techniques, DCT and LTC, we tested the compression rate and the energy consumption ratio for the same distortion achieved by the different techniques. The signals we used for this test are those from the SYN dataset varying the correlation length from 20 to 200 samples.

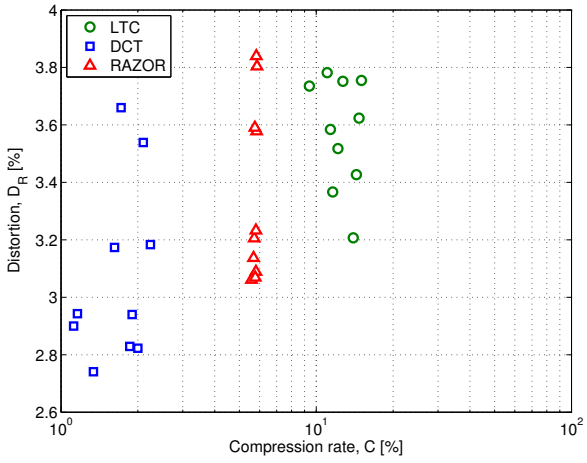
Figure 4.4 plots the outcome of this evaluation and shows RAZOR, in red with triangular markers, LTC, in green with round markers, and DCT, in blue with square markers. The three figures explore three different trade-offs: Figure 4.4-(a) plots the compression rate percentage, $C$, on the x-axis and the energy consumption percentage with respect to the baseline consumption (no compression), $R$, on the y-axis; Figure 4.4-(b) plots the distortion, $D_R$, on the x-axis and $R$ on the y-axis; while Figure 4.4-(c) plots $C$ against $D_R$.




(a) $R$ vs. $C$




(b) $R$ vs. $D_R$




(c) $D_R$ vs. $C$

**Figure 4.4.** *Performance comparison of three data compression solutions: RAZOR, in red with triangular markers, LTC, in green with round markers, and DCT, in blue with square markers. The three algorithms have been compared using three metrics: the compression rate, $C$, the energy consumption ratio, $R$, and the distortion, $D_R$. The signals used in the graph are those of the SYN dataset, varying the correlation length from 20 to 200 samples.*

The results show that RAZOR is able to save about $89\%$ of the energy needed for transmitting uncompressed data, which is better than $86\%$–$74\%$ of DCT and on par with the $85$–$91\%$ of LTC. In terms of compression rates, RAZOR shows intermediate performance ($6\%$) between DCT (the best, with $3\%$) and LTC (the worst, with more than $8\%$). The sensitivity of RAZOR's compression and energy consumption performance to the correlation length is very weak (all points are very close to each other), whereas in the other two schemes a greater variability can be observed. The distortion performance is more dispersed for all three schemes, and while $R$ and $C$ present a strong degree of correlation with each other (see Figure 4.4-(a)), $D_R$ is essentially uncorrelated with both $R$ and $C$ in all cases. For the sake of fairness, the computational cost of the three algorithms was computed assuming fixed-point operations for all of them.

Finally, we analyzed RAZOR's classification accuracy against EMMA [33]; to this end, we used the UCR dataset, averaging the results over different categories. Furthermore, we tried to set EMMA's parameters to offer a fair comparison with our solution. However, it must be said that EMMA has been designed to classify much longer time series than those we selected from the dataset and, therefore, cannot be expected to work very well in our case; this selection has been made to respect the computational constraints of our target devices.

In Figure 4.5, we plotted the correct classification, false positive and erroneous classification ratios for the two algorithms, where blue bars on the left refer to RAZOR, while red bars on the right provide EMMA's results. These results show that RAZOR is able to obtain a very good correct classification ratio, whereas one of the state-of-the-art solutions for data mining fails when used with a very constrained configuration. Based on the results in Figures 4.4 and 4.5, we can conclude that, even though RAZOR is neither the best compressor nor the best classifier, it is able to achieve good performance in both functionalities, while maintaining a low computational complexity, and can, therefore, be considered as a good lightweight tool for signal analysis and transmission in constrained environments. Its very low computational complexity reduces the energy consumption of constrained devices by about $90\%$ with respect to the energy needed to send the measured data uncompressed. To strengthen this conclusion, in the final subsection, we will provide a short overview of RAZOR's results obtained on real data.

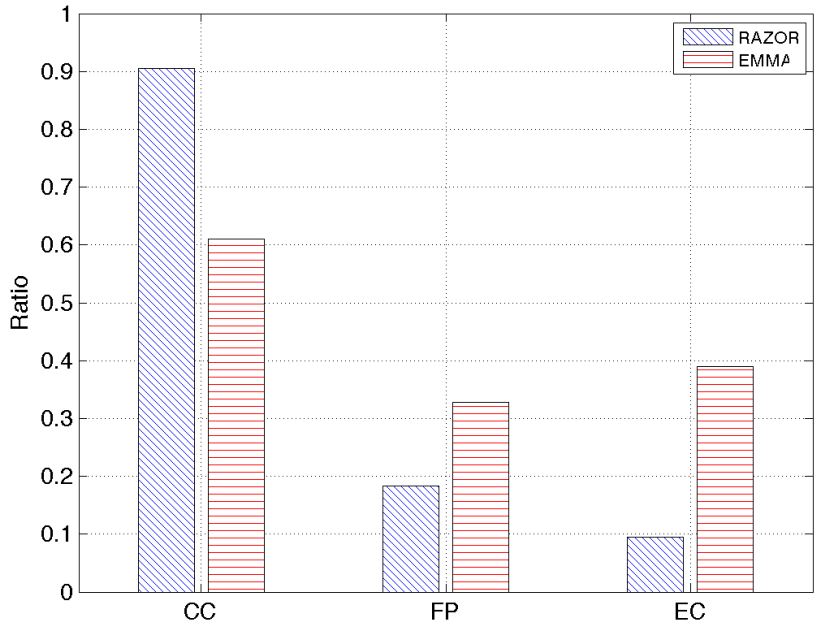



**Figure 4.5.** *Classification accuracy comparison between RAZOR and Motifs through Matrix Approximation (EMMA): the CC, FP and EC bars report the correct classification, the false positive and erroneous classification ratios.*

### 4.4.4 Real scenario

The last set of results is obtained by applying RAZOR to a real dataset containing sensed information for temperature, humidity and light from $25$ telosb nodes deployed in a building during two months. The training set has been computed over the data obtained during a single day, and all RAZOR parameters are set as described in the previous section.

Figure 4.6 provides an overview of the obtained results: the compression rate percentage, $C$, the root mean square error, $D_R$, and the percentage of energy used with respect to the energy needed to send uncompressed data, $E/E_B$. Since all the percentages obtained here are very low, RAZOR's good performance is confirmed for real scenarios as well. In particular, our algorithm is capable of transmitting data over the network, consuming less than $10\%$ of the energy needed to transmit the same data uncompressed, maintaining the root mean square error as low as a few percent.

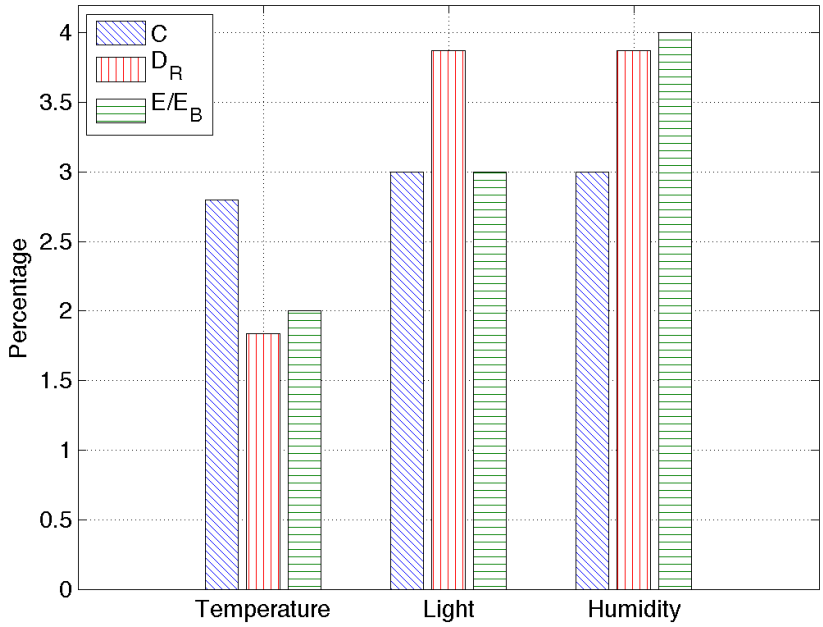



**Figure 4.6.** *RAZOR's performance overview in a real scenario for different sensors: temperature, humidity and light. The performance figures are the compression rate percentage, $C$, the relative root mean square error, $D_R$, and the percentage of energy used with respect to the energy needed to send uncompressed data, $E/E_B$.*

### 4.4.5 Lossy and Multi-Hop communications

So far, we only considered single hop communication and perfect channels (*i.e.*, no packet loss). In this section, we will address both lossy channel and multi-hop communications in order to evaluate RAZOR's performance under more realistic environments.

First of all, we consider the impact of channel errors on the information loss, $I_L$, computed as the ratio between the average number of bits lost over the total number of bits of the uncompressed signal, and on the communication efficiency, $\eta$, computed as the ratio between the average number of correct bits received over the total number of bits sent, under a non-perfect channel. In order to deal with different packet sizes, we computed our results starting from the bit error probability, $p_b$, which is the ratio of the average number of erroneous bits over the total number of bits sent. From this and assuming errors to be independent and identically distributed (i.i.d.), we can derive the packet error probability, $p_p = 1 - (1 - p_b)^{n_b}$, where $n_b$ is the number of bits in a packet. (We leave the analysis of bursty channels for future work.)

Under these assumptions and with no retransmissions, transmitting the raw signal over a lossy channel with $p_b$ results in the following:

$$I_L = \sum_{i=1}^{n_p} n_b p_p / (n_b n_p) = p_p \tag{4.21}$$

where $n_p$ is the number of packets in the uncompressed signal. Instead, when using a compressor, the number of bits lost when a packet is corrupted amounts to the number of bits coded in the lost packet. Thus, Equation (4.21) becomes:

$$I_L = \sum_{i=1}^{n_{p_c}} n_{b_c} p_p / (n_b n_p) = n_{p_c} n_{b_c} p_p / (n_b n_p) = \frac{n_b}{C} n_p \, C \, p_p / (n_b n_p) = p_p \tag{4.22}$$

where $n_{b_c} = n_b/C$ and $n_{p_c} = n_p C$ are the number of bits carried by a compressed packet and the number of compressed packets needed to send the whole uncompressed signal. Note that the information loss is exactly equal to the packet error rate for both the raw signal and the compressed signal, under our assumptions and if the packet size remains the same; however, the impact of information loss for compressors, such as DCT and LTC, depends on which packets are lost (in fact, in the case of DCT compression, the impact is much larger when the lost packet carries information about the low frequency coefficients of the transform, and in the LTC case, since a packet carries the parameters of the model for a large part of the signal, it is possible that a larger amount of bits of the original signal are lost as a consequence of losing a single packet.)

RAZOR's case is different, as a packet is smaller than a regular packet for WS&AN; thus, RAZOR's packet error rate becomes $p_{p_R} = 1 - (1 - p_b)^{n_{b_R}}$, where $n_{b_R} < n_b$ is the size in bits of a RAZOR's packet. Obtaining RAZOR's information loss rate is now straightforward:

$$I_L = \sum_{i=1}^{n_{p_c}} n_{b_c} p_{p_R} / (n_b n_p) = p_{p_R} < p_p \tag{4.23}$$

which amounts to RAZOR being more robust to channel errors, due to its smaller packet size.

Subsequently, if we account for a given number of retransmissions, $r$, the packet loss rate becomes $p_p(r) = p_p^{r+1}$, as a packet is only lost if all its transmissions are lost, for the raw and the general compressor case. For the RAZOR case, instead, given the smaller RAZOR's packet size, it is possible to piggyback retransmissions in subsequent communications by increasing the packet size: as an example, if the first packet is lost, the second packet can

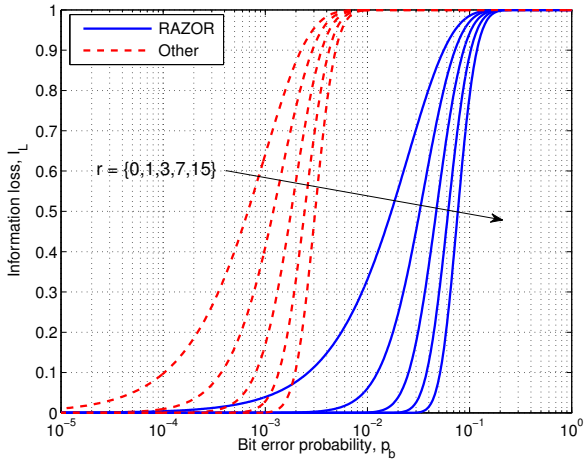
carry both the second and the retransmission of the first motif, thus saving bandwidth at the price of a slightly higher packet error rate and delay. The RAZOR packet error rate with $r$ retransmission attempts becomes $p_{p_R}(r) = \prod_{i=1}^{r+1}(1-(1-p_b)^{i\,n_{b_R}})$. Now, it is possible to compute the information loss given any number of retransmissions for both the raw and RAZOR's case, which is equal to $p_p(r)$ and to $p_{p_R}(r)$, respectively.

In order to evaluate $\eta$, we need to recompute the average number of bits sent, which, in the raw case, is $n_S(r) = n_b \sum_{i=0}^{r} p_p(i) = n_b \frac{1-p_p(i+1)}{1-p_p}$, while for RAZOR it is slightly more complex, due to the increasing size of packets, and becomes $n_{S_R}(r) = \sum_{i=0}^{r} n_{b_R}(i+1)p_{p_R}(i)$. Finally, $\eta$ is obtained as a function of the number of retransmissions as:

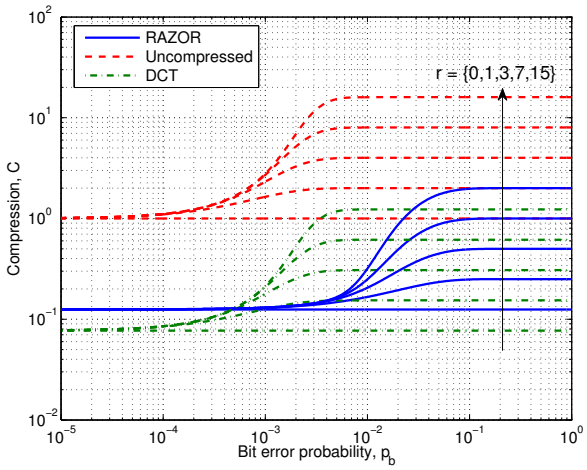
$$\eta(r) = \frac{1 - I_L(r)}{n_S(r)}. \tag{4.24}$$

Finally, note that the compression rate, $C$, changes when retransmissions are accounted for, and it becomes $C(r) = n_S(r)/(n_b n_p)$, which translates in the ratio between the average number of bits sent over the size of the uncompressed signal in bits.

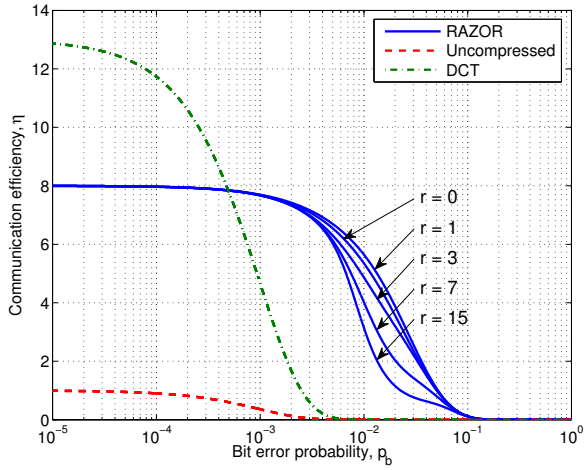
Figure 4.7 shows the results of the previous analysis drawing with blue solid lines, red dashed lines and green dash-dotted lines the curves related to RAZOR, the uncompressed signal and DCT compression, respectively. In addition, the effect of the increasing number of retransmissions is directly annotated in the figures. In particular, Figure 4.7-(a), Figure 4.7-(b) and Figure 4.7-(c) plot the information loss, the compression and the efficiency as a function of $p_b \in [10^{-5}, 1]$.




(a) Information loss




(b) Compression




(c) Communication Efficiency

**Figure 4.7.** *Performance evaluation in lossy channels varying the bit error probability of RAZOR (blue solid line), uncompressed data (red dashed line) and DCT (green dash-dotted line). RAZOR has been compared using three metrics: information loss, $I_L$, compression, $C$, and communication efficiency, $\eta$. For each chart, we considered a number of retransmissions $r \in \{0, 1, 3, 7, 15\}$.*

Figure 4.7-(a) does not show any line for the DCT, as this technique offers neither improvement nor degradation in terms of $I_L$ over the uncompressed technique, as is evident from Equation (4.22). Instead, due to its reduced packet size, RAZOR outperforms the other solutions by two orders of magnitude, as it is able to deliver almost all the information, even for $p_b = 10^{-2}$ when at least three retransmissions are allowed.

Figure 4.7-(b), which depicts the compression achievable if $r = \{0, 1, 3, 7, 15\}$ retransmissions are allowed, demonstrates that DCT outperforms RAZOR almost everywhere, except in the central region of $p_b$, thanks to its highest compression factor.

Finally, Figure 4.7-(c), which illustrates the efficiency of the three solutions, shows that DCT is the best compressor when the information loss is negligible, while RAZOR is clearly the preferable solution when $p_b > 10^{-3}$. Moreover, in terms of communication efficiency, $r$ has no effect for both DCT and uncompressed transmission, while we noted that, for RAZOR, only a single retransmission improves the communication efficiency; thus, choosing the correct value for $r$ is a trade off between robustness (high $r$) and energy consumption (low $r$).

For what concerns multi-hop communications, we analyzed the energy consumption figures for RAZOR, DCT compression and RIDA. For what concerns RAZOR and DCT, we computed the energy expenditure as the sum of the single hop consumption plus the energy needed to forward either the motifs (RAZOR) or the coefficients (DCT) from when they are computed in the network to the gateway. Instead RIDA, which is a hierarchical solution [56], needs first to compress the signal locally within each cluster and, subsequently, to send the computed coefficients to the gateway.

Figure 4.8 shows with blue solid, red dashed and green dash-dotted lines the results for RAZOR, RIDA and DCT, respectively. While RAZOR and DCT have almost the same behavior with RAZOR outperforming DCT, thanks to its lower computational complexity, RIDA has a very high energy consumption (as high as what is needed to send the uncompressed signal) in small networks, but shows a rapid improvement as the network size increases, outperforming both DCT and RAZOR when the number of hops is larger than $12$.

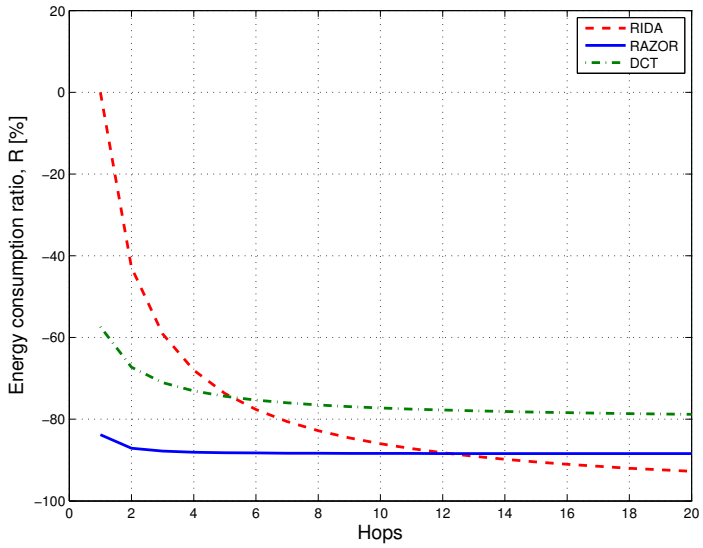



**Figure 4.8.** *Performance comparison of three data compression solutions: RAZOR (blue solid line), DCT (green dash-dotted line) and RIDA (red dashed line). The three algorithms are compared in terms of the energy consumption ratio, R, varying the number of hops of the network.*

## 4.5 Conclusions

In this chapter we presented RAZOR, a novel lightweight algorithm for data compression and classification targeting constrained devices, such as those that will be encountered in the Internet of Things.

The main concept driving the design of our solution is derived from vector quantization and pattern recognition techniques: in fact, RAZOR works by first creating a Codebook at the gateway from a given training set of uncompressed data and, subsequently, distributing this Codebook to constrained devices to compress the following readings.

The Codebook is then used again by the gateway to interpret the compressed data received from the nodes. In addition, the gateway, or some other computationally powerful devices, can use different Codebooks to classify data produced by an unknown source.

In order to obtain the final definition of the RAZOR algorithm, we started from the analysis of our previous work presented in Chapter 3, and we complemented it by considering different normalization formulas for data segments, different dissimilarity functions for the motif extraction algorithm and a novel technique to predict the motif to be transmitted, which is able to provide further improvements in terms of the compression rate.

We ran a thorough evaluation campaign to select RAZOR's best configuration and to compare it against state-of-the-art signal processing solutions, showing that our algorithm is able to obtain performance similar to that of the most relevant competitors in terms of both compression and classification, when used on constrained devices. In addition, we studied the impact of multi-hop and lossy communication on RAZOR performance, showing that it outperforms standard compressors when the error impact is not negligible, and can be preferable to a more complex hierarchical solution, when the network size is smaller than 12 hops.

Thus, RAZOR is a very good candidate for developing a versatile signal processing tool to be used in the IoT both to reduce the communication overhead when transmitting known signals and to classify unknown information sources.

Chapter 5

# An application of IoT to robotics

## 5.1 Introduction

Service robots are looking for their killer applications to leave research laboratories and enter in our daily lives, being progressively deployed in houses, on roads and in public spaces. The same trend is experienced by WSN technologies, which are breaking through the academic boundaries to spread over the market.

The complementarities of the WSN and robot technologies result in the synthesis of a novel network paradigm, generally called Wireless Sensor and Robot Networks (WS&RN), where the two technologies are intimately interconnected in order to enable a large set of advanced and innovative services [60–63]. A similar trend is followed by the Networked Robotics community that is investigating the idea of exploiting the communication between the robot and sensors distributed in the environment to lower the computational burden and the intelligence requested to the robot to effectively operate in complex environments.

Until now, the main problem covered by this thesis was to reduce the amount of data exchanged in a Wireless Sensor Network (WSN) with the goal of preserving the information carried in each node and limiting the energy consumption to compress and send data. The Motif concept was leveraged to achieve that goal to compress the data sent from each sensor node and also to classify unknown signals.

However, in the literature it is possible to find several feature descriptors that were developed with other purposes, but whose characteristic can fit very well within the IoT con-

cept. This is the case of the scale invariant feature transform (SIFT) [64], designed with the goal of retrieving an object inside a picture.

In this chapter the Motif descriptors are substituted with the SIFT descriptors and the main objective is that a robot, in an automated fashion, can recognize and catch any object in a unknown scenario. Although a robot can have a big database with all SIFT already computed for each object, the time to compare each single object is not negligible. Another interesting way is to store only the SIFT descriptors of each single object inside a flash memory of an device. Thus, it is the same idea behind *smart* object and it is possible to design and create a system fully independent of the objects that are present in a scenario. A node has only the features to communicate through wireless technology and a small flash memory where store the descriptors. The reader can understand that we are applying the same concept used with the Motif; the signals are replaced with image and there is a dictionary for each differente *smart* object. However more details are given in the following sections.

Inspired by the PEIS Ecology developed by Saffiotti et al. [1], where intelligent and complex behaviors are achieved through the cooperation of many simple robots and sensors, we here propose a system that combines in a synergetic way the complementarities of the robot and WSN technologies to enable the autonomous exploration of unknown intelligent environments by the robots. The key ingredient of our system is the presence in the environment of objects tagged with wireless sensor nodes, or *motes* for shortness, which provide communication, processing, and data storage capabilities, thus turning a dummy object into a *smart* object. The same mote is applied to the robot, in order to enable radio communication with the *smart* objects. In our solution, the robot does not have any prior knowledge about the shapes of the objects, their positions and their number: all these informations are obtained by interacting with the objects. Therefore, conversely to most of the solutions proposed in the literature, we can deal with any object from the simplest to the most complex (like those in Figure 5.1), without necessity of any *a priori* knowledge about the objects. We foresee the application of this system in many different scenarios, including industrial, domestic, and assisted living. As an example, teams of robots may be deployed in automated warehouses to catalog, localize and retrieve objects upon request, without knowing in advance the type nor the location of the objects to be collected. Similarly, robots may be used in libraries to put back books on the shelves after closure. In a home scenario, the same system can be

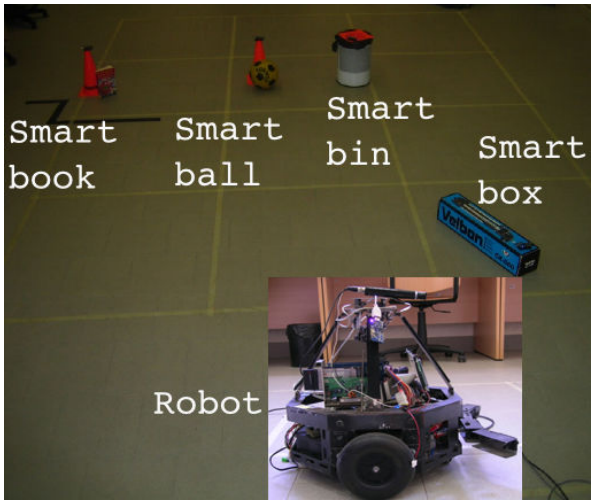



**Figure 5.1.** *The robot and some of the* smart *objects used in the presented experiments. In order to demonstrate the flexibility of the system we chose objects that usually people move around in a room.*

used to tidy-up the play-room of kids. At the end of the day, a robot can locate all toys in the room and store them back into the proper containers, provided both toys and containers are *smart* objects. In the assisted living domain, we may envision a system where an intelligent portable device may drive a visual-impaired person through a partially unknown environment by vocally describing the *smart* objects recognized along the path. The audio description of the *smart* objects may be embedded in the motes' memory by the objects manufacturer and wireless transmitted to the portable *smart speakerphone* upon request. Yet, a personal robot may display to a motion–impaired user the list of *smart* objects (remote controls, mobile phones, books, and so on) that it discovers in the environment thanks to wireless and, possibly, multi–hop communication. Then, the robot may go and grab an object for the user.

The realization of this vision requires the robot to be able to identify, locate and, finally, recognize the different objects in the environment. Whereas the radio communication may be successfully exploited to discover which *smart* objects are located in the environment, the standard localization schemes based on the radio signal strength (RSS) do not provide a precise geographical location of the nodes [65]. This is particularly true in indoor environment, because of the self interference effects due to multiple reflections of the radio signals. Currently, the most popular RSS-based localization algorithm can locate the motes with respect to the robot with a precision of a meter or so, that is not enough for a reliable localization

of the object based only on its ID. We hence propose to complement the radio-based localization with a visual detection of the object by the robot. To this end, we store locally, in the memory of the *smart* object, the appearance of the object itself. This information is then transmitted to the robot upon request, so that the robot can visually recognize that object in the image taken by the on-board camera.

Object recognition from visual features is a well known problem in computer vision for which many different solutions have been proposed, based both on the analysis of the global appearance of the object and on the extraction of a relevant set of features. Concerning the first approach one of the most popular approaches was proposed by Viola and Jones [66], where an Adaptive Boosting (AdaBoost) learning algorithm is applied to a set of Haar-like features efficiently extracted from images. Lowe [64] proposed an object recognition system based on local image features (SIFT) invariant to image scaling, translation, and rotation. SIFT features have proved to be very robust and effective in many practical object recognition applications. In the last years, bag-of-words (BoW) methods have received a great attention for object recognition and categorization tasks (e.g., [67]): these methods aim to represents images with a set of clusterized visual descriptors (e.g. SIFT features) taken from a visual vocabulary generated offline. Recently, BoW approaches have been improved by indexing descriptors (visual words) in a vocabulary tree with a hierarchical quantization of the features descriptors [68] and using quantization methods based on randomized trees [69]. Despite their efficiency and effectiveness, BoW methods require an accurate off-line learning step. Moreover, all the object signatures (i.e., collections of visual words that describes the objects) must share the same dictionary: this dependency limits somehow the portability of these methods. For these reasons, we choose to employ an object recognition system similar to the one proposed in [64], successfully exploited in other recent robotic recognition frameworks (e.g. [70]).

We observe that some authors proposed the use of RFID technology to ease the recognition of objects by the robot. Although the use of RFID technology may boost the cooperation and interaction between object and robot, its scope is limited by the intrinsic constraints of the RFID technology, such as the rather short operational range (especially for the passive RFIDs) and the extremely limited computational capabilities of the RF tags. Moreover, the use of active motes may also provide measurements involving either the object itself, like

object temperature, level of filling, inclination, weight, deterioration, or the surrounding environment, as temperature, pollution, humidity, light, and so on. The *smart* objects may form a self-established multi-hop wireless network that can be used to enlarge the sensorial and communication range of the robot, augmenting its environmental awareness. The synergetic combination of the two systems is finally realized by means of a suitable suite of communication protocols and algorithms, whose aim is to enable the interaction of the robot with the *smart* objects located in the area.

In summary in our approach, the robot can create a map with a coarse localization of the *smart* objects of interest obtained via radio and move toward them. When the robot is in the room where the object of interest is located, it can seek for its appearance in the images it has acquired. In this way, the robot can locate much more precisely the object and navigate toward it in order to perform the desired interaction. This process develops along three main phases:

- *Object Discovery*: the *smart* objects, which are usually dozing to save energy, are waken, identified and time synchronized by the robot by using the radio interface.

- *Object Mapping*: the robot starts navigating the environment and uses the information provided by the onboard odometers and the radio signals received from the *smart* objects to map the objects in space.

- *Object Recognition*: once the *smart* objects' mapping is sufficiently accurate, the robot can move toward the estimated location of a target object. When in proximity, the robot asks the *smart* object to send its visual appearance descriptors, which are continuously matched against those extracted from the stereo camera images. When the matching exceeds a given threshold, the object is recognized in the images taken by the onboard cameras of the robot that can then exactly localize it in space, approach it and start fine-grain interaction.

In the following of this chapter we describe in greater detail the algorithms that we used to realize the Object Discovery, Object Mapping and Object Recognition services. We observe that these services can actually be realized by using state of the art solutions taken from the WSN, autonomous robotic or WS&RN areas. However, the approaches proposed in the literature generally fail to catch the potential synergies among the three different phases, in

particular the object mapping and recognition, whereas in this manuscript we offer a more complete and organic vision of the system as a whole. Furthermore, we present a large selection of experimental results obtained by using a proof-of-concept testbed that we developed to prove the feasibility of the idea and to identify pitfalls and technical challenges.

Summing up, the strength of the proposed approach with respect to the state of the art consists in the following main points. i) We take the concept of *smart* object in the WS&RN picture, which is a common object tagged with a mote that stores object-related information, such as weight, size, appearance, status and so on. This information can be wirelessly transmitted to other nodes, thus enabling the seamless inclusion of new objects into the system, without the need for database updating or similar operations that are instead required in classical object-recognition systems. ii) The use of motes to realize the *smart* objects rather than passive RFIDs makes it possible to establish a multi-hop wireless network that, besides providing the usual WSN services, may relay messages from remote nodes to the robot in case direct communication is not available. This feature improves the flexibility and robustness of the system with respect to passive RFID solutions. We observe that, concerning the communication and computational capabilities, WSNs and active RFID are rather similar. Nonetheless, the WSN technology natively supports environmental monitoring functionalities that can integrated into the system in different ways. For instance, the data collected by light sensors can be used to suitably tune the sensitivity of the camera to the environmental brightness or to select the best sets of image features to be transmitted to the robot. iii) Complementing the localization information obtained by radio-based localization schemes with information extracted from the visual appearance of the *smart* objects improves the accuracy of the discrimination capability of the robot in presence of similar objects, the accuracy in the localization of the objects with respect to the robot, and finally enhances its capability of interaction with the objects.

Part of the results presented in this manuscript appeared in the proceedings of some international conferences, namely [2,71,72]. In this chapter, however, we offer a more complete and organic vision of the system as a whole, which was missing in each of the previous publications. Furthermore, we detail the communication protocol between *smart* objects and robot and propose an innovative multichannel strategy for radio-based localization. Moreover, we assess the improvements with respect to classical single-channel methods by pro-

viding extensive experimental results and, as a side result, we compare the localization techniques considered in our previous publications [2,72] with another algorithm, based on the multi-dimensional scaling technique. For this last technique, we also investigate the accuracy gain obtained by considering inter-object measurements into the localization algorithm, which was not presented in our previous publications. Finally, we extend the analysis of the *smart* object visual recognition by means of MoBIFs, as presented in [71], by introducing a stereoscopic visual recognition setup that makes it possible to estimate the object distances by applying triangulation on the correspondences found in MoBIF descriptor clouds.

The rest of the chapter is organized as follows. In Section 5.2 we describe the object discovery process. In Section 5.3 we discuss the object mapping problem and we present the multichannel ranging technique and the communication protocol that we designed to coordinate the nodes during this phase. Furthermore, we describe the multi dimensional scaling method for objects mapping. In Section 5.4, we describe the object appearance descriptor method we used to realize the visual object recognition module. In Section 5.5, we present experiments in which the robot performs all the steps to discover and approach the *smart* objects. Finally, in Section 5.6 we draw conclusions and discuss future extensions of the work. WS&RN

## 5.2 Object discovery

As we said the number of smart objects and their position in the environment are initially unknown to the robot. The robot thus n*smart*eeds first to discover and, then, localize the objects in the environment.

The problem in objects discovery is that motes are not always awake and ready to reply to inquiry messages. In fact, the radio transceiver is the most energy-hungry component of a sensor node, so that motes generally operate according to a periodic ON-OFF pattern: in ON periods, all the sensing and communication capabilities are active, whereas in OFF periods the radio transceiver and, possibly, other hardware modules are powered off to save energy. Hence, a node is active for only the fraction of time $d$, typically referred to as *duty cycle*. The ON-OFF patterns followed by the different nodes are, in general, asynchronous, since achieving time synchronization in multi-hop WSN requires rather sophisticated algorithms (see, e.g., [73]). Clearly, the duty cycle $d$ and the ON-OFF cycle period have a

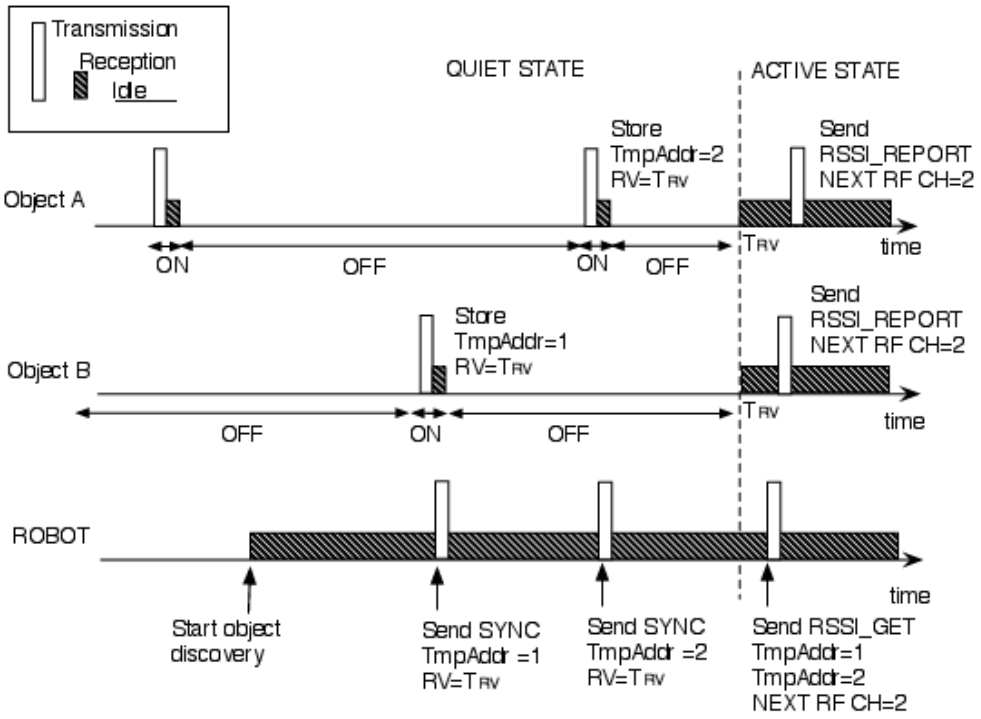



**Figure 5.2.** *Example of Object Discovery procedure, followed by part of RSSI harvesting procedure.*

direct impact on the node's lifetime, i.e., the time before a sensor node exhausts its battery charge, and on the reactiveness of the node to the occurrence of events or solicitations by other nodes. These two aspects are obviously in contrast and striking the most convenient balance between them is a subtle design problem that has stimulated different solutions (see, for instance, [74] and references therein). In our scenario, however, the node discovery problem is greatly simplified by the presence of the robot that can keep always active its wireless interface, thus intercepting any transmission by other nodes in its coverage range. This feature makes it possible to adopt simple *rendezvous* strategies that, on the one hand, allow motes to implement ON-OFF patterns with minimal duty cycles, thus saving energy and, on the other hand, provide quick reaction time, compatibly with the length of the cycle time $T_c$.

More specifically, in this work we propose the following Object Discovery strategy, which is graphically exemplified in Figure 5.2. The motes attached to the *smart* objects can operate in two states, namely *Active* and *Quiet*. In Active state, the *smart* objects keep their transceiver switched on and ready to receive command messages, whereas in Quiet state each mote follows a periodic ON-OFF pattern. In the ON phase, the mote switches on its

radio interface and broadcasts a HELLO message containing basic information about the attached object. Channel access is managed according to the classic CSMA algorithm, so that the message transmission is deferred to a later time if the channel is occupied by other transmissions. The ON phase ends when the channel remains idle for a certain time interval $T_{listen}$. During the ON period the mote will process all the received packets and act accordingly.

When a robot wishes to discover the *smart* objects in its proximity, it switches on its radio interface and continuously listens for HELLO messages sent by nearby objects. The robot retrieves and stores the MAC address, the object profile and the other info contained in the received HELLO messages. Furthermore, the robot replies to each message by sending a SYNC message within the time interval $T_{listen}$. The SYNC message contains two main fields, the *Temporary Address*, and the *Rendezvous Time*. The Temporary Address is a short identifier that the robot assigns to the addressed *smart* object and that will be used in subsequent communications to refer to that object. The Rendezvous Time, instead, denotes the time interval after which the robot expects the *smart* object to be in Active state. The *smart* object that receives the SYNC message, then, stores its Temporary Address and schedules the transition from Quiet to Active mode after the Rendezvous Time interval. The Rendezvous Time value is updated at each SYNC message in order to synchronize the activation of the *smart* objects around the same time instant.

The communication protocol is unreliable, since it does not entail any explicit acknowledgment mechanism: if the SYNC message is not correctly decoded by the addressed *smart* object, the object will stay in Quiet state and keep performing the ON-OFF cycle. In turn, the information collected by the robot are stored in a *soft* form and will be deleted after a certain time period if not refreshed by the reception of new messages from the corresponding *smart* object. However, the robot can reply to HELLO messages at any time, even when performing other tasks than Object Discovery, in order to identify objects that were initially out of the robot's range.

## 5.3 Objects mapping

The Object Discovery procedure provides the robot with information concerning the objects in its coverage range. The following step is to map the objects in the area, i.e., deter-

mine their geographical coordinates, in order to enable fine-grain interaction. The problem of nodes localization in WSN has been since long recognized as an important and challenging issue and lot of research has been carried out in this context. Many solutions assume the presence of a limited number of nodes, called *beacons* or *anchors*, that know their own position and are used by other nodes to locate themselves through triangulation techniques. Many of these schemes make use of the Received Signal Strength Indicator (RSSI) to determine a rough estimate of the distance between transmitter and receiver, an operation referred to as *ranging*. This approach offers the advantage of being readily employable in any radio device, since the RSSI is supported by basically all radio transceivers. Another advantage of RSSI-based ranging is that it does not require the node to be in line of sight with the robot, since the radio signal passes through obstacles, as people, furniture or even walls. Unfortunately, the range estimate based on RSSI measurements is unreliable and subject to random fluctuations due to a number of environmental factors. Therefore, the accuracy that can be obtained with RSSI-based localization techniques in indoor environments is rather poor, with errors of the order of $1$ to $6$ meters [65], depending on the number of beacons.

The presence of the robot, however, can drastically enhance the performance of the localization techniques. For instance, in [75] we showed that the robot, which is fairly well localized by virtue of the on-board odometers and navigation system, can act as a sort of mobile beacon drastically augmenting the number of reference signals to be used in classical localization algorithms. However, the presence of robots opens the way to much more advanced and sophisticated localization methods. A very flexible and powerful technique is the Simultaneous Localization And Mapping (SLAM) algorithm realized by means of an Extended Kalman Filter (EKF) approach that merges the information provided by the robot's odometers with the RSSI samples provided by the surrounding objects to simultaneously track the motion of the robot in the environment and refine the mapping of the objects in the area. We experimented this approach in [72] and observed that the accuracy of the mapping provided by EKF-SLAM is strongly affected by the initial guess of the mote position, which is required at the beginning of the SLAM procedure to initialize the system state $\Theta$, which is a vector containing the current estimate of robot and objects locations. Therefore, in [2] we proposed to couple the EKF-SLAM algorithm with a mote position initialization based on particle filters. According to our experimental results, this approach reduces both mean

and variance of the final location estimate error with respect to the simple EKF approach.

In this chapter we further advance the investigation of the object mapping problem along four directions: first we summarize the EKF-SLAM definitions and the delay initialization approach; second, we propose a method to increase the accuracy of the RSSI-based ranging by exploiting the capability of the sensor nodes to operate on different RF channels; third, we include in our experiments another localization method, namely the Weighted Multi-Dimensional Scaling (MDS), that is computationally lighter than EKF and is based on the same engine used for the object recognition functionalities that will be described in the next section; fourth, we evaluate the extent to which inter-object RSSI measurements may ameliorate the objects mapping. As a side result, in Section 5.5 we provide an experimental performance comparison among the various mapping techniques considered in our work, namely EKF with delayed initialization, Particle Filter only (PF), MDS, and MDS with inter-object measurements.

In the following of this section we explain the principle of multichannel ranging and describe the communication protocol we designed to collect RSSI samples over different RF channels. Successively, for reader convenience, we overview the basics of the MDS approach that was proposed by Costa et al. in [76], whereas for the details of the other aforementioned localization techniques we refer the reader to our previous publications [2,72,77].

### 5.3.1 Distance estimation based on RSSI

The most widely used RSSI-based ranging model is based on the well-known path loss plus shadowing signal propagation model, according to which the received power at distance $d$ from the transmitter can be expressed (in dBm) as

$$P_{rx} = P_{tx} + K - 10\eta \log_{10}\left(\frac{d}{d_0}\right) + \Psi\,, \tag{5.1}$$

where $P_{tx}$ is the transmitted power in dBm, $K$ is a unit less constant that depends on the environment, $d_0$ is the reference distance for the far field model to be valid, $\eta$ is the path loss coefficient and $\Psi$ is a random variable that takes fading effects into account [78]. In general, the $\Psi$ term is assumed to be a zero-mean Gaussian random variable, though this model is not always the most appropriate, especially in presence of line of sight (LOS) between transmitter and receiver [79]. In this case, in fact, the variability of the received power is mainly due to the random phase shifts between the direct path and the strongest reflections, typically

on floor, ceiling and close-by objects. At the frequency of 2.4 GHz, which is typically used by sensor nodes, moving transmitter or receiver of few centimeters can result in a totally different combination of the signal reflections at the receiver, with a significant variation of the received signal power. We observe that the same effect can be obtained by changing the carrier frequency without moving the nodes. As an example, by increasing the carrier frequency of the radio signal from $2.4$ GHz to $2.45$ GHz, the phase shift between the direct signal and a copy that follows a path $3$ meters longer (e.g., ceiling reflection) will be $\approx \pi$. This suggest that it is possible to reduce the impact of $\Psi$ in 5.1 by collecting RSSI samples on different RF channels and, then, using their mean value $\bar{P}_{rx}$ in the ranging equation:

$$\hat{d} = d_0 10^{\frac{P_{tx}+K-\bar{P}_{rx}}{10\eta}} . \tag{5.2}$$

### 5.3.2 EKF SLAM

In this paragraph we wish to give a brief introduction to EKF SLAM a well-known technique that recursively solves the online SLAM problem where the map is feature-based. It estimates at the same time the position of the robot and the position of the features: in our case the features are the motes. Thus, the state vector of the system that should be estimated is:

$$q_k = \left[ \; x_k, y_k, \theta_k, x_{m_1}, y_{m_1}, \ldots, x_{m_n}, y_{m_n} \; \right]^\top \tag{5.3}$$

where $[x_k, y_k, \theta_k]^\top$ is the robot position and heading at time $k$ and $[x_{m_i}, y_{m_i}]^\top$ is the location of the $i-th$ mote.

Let $u_k = [\Delta D_k, \Delta\theta_k]^\top$ denote the input vector at time $k$, where $\Delta D_k$ is the distance traveled by the robot since the previous state update and $\Delta\theta_k$ is the heading change, both measured by the robot's odometers. During the EKF SLAM prediction phase, then, the mean $\mu_k$ and covariance $\Sigma_k$ of the state $q_k$ are updated as follows:

$$\mu_k^- = f\left(u_k, \mu_{k-1}\right) \tag{5.4}$$

$$\Sigma_k^- = A_k \Sigma_{k-1} A_k^\top + B_k Q_k B_k^\top \tag{5.5}$$

where $Q_k$ is the covariance matrix that models the uncertainty in the state transition, while $A_k$ and $B_k$ are the following Jacobians:

$$A_k = \left.\frac{\partial f}{\partial q_k}\right|_{q=\mu_k^-}, \; B_k = \left.\frac{\partial f}{\partial u_k}\right|_{q=\mu_k^-} . \tag{5.6}$$

When a new RSSI value $P_{k,i}$ for the $i$th mote is received by the mote, it is first converted into a distance measure $d_{k,i}$ by reversing the classical path-loss model

$$P_{k,i}\,[dBm] \;= P_{TX_i} + K - 10\eta \log_{10}\left[\frac{d_{k,i}}{d_0}\right] \;, \tag{5.7}$$

Note that this is an ideal model that does not include the noise in the RSSI measurements that is instead handled by the adopted Kalman filtering scheme. Then, the correction phase of the EKF can be performed. Let $h(q_k, i)$ be the function that returns the distance between the robot and the $i$-th mote according to the current state $q_k$, i.e.,

$$h(q_k, i) = \sqrt{(x_k - x_{m,i})^2 + (y_k - y_{m,i})^2}\,.$$

Hence, the mean and covariance of the state can be updated as

$$\mu_k = \mu_k^- + K_k\left(d_{k,i} - h\left(\mu_k^-, i\right)\right) \tag{5.8}$$

$$\Sigma_k = \left(I - K_k H_k\right)\Sigma_t^- \tag{5.9}$$

where $H_k$ is the Jacobian of the function $h$ and $K_k$ is the Kalman gain. For a more detailed description refer to [72].

### 5.3.3 Delayed initialization

The SLAM works pretty well, provided that the motes position used to initialize the Kalman filter at the beginning of the SLAM procedure is sufficiently accurate. However, most of the times the motes positions do not know, therefore we propose the use of a particle filter to implement a delayed initialization of the Kalman filter. In practice, when the robot discovers a new mote, it first computes an estimate $r$ of the distance to the mote by using 5.7 and, then, it instantiates a particle filter in which the samples, which are hypotheses of the mote position, are spread around the robot at distances drawn from a Gaussian distribution, with mean $r$ and standard deviation $\sigma$ that depends on the variance of the RSSI measures. Each particle is also weighted in accordance with the value of the normal distribution in that point. When the robot receives new messages from the mote, it calculates a new estimate of the mote distance that correspond to new annular probability distributions centered in the current robot position. We then apply a Sampling Importance Resampling (SIR) algorithm each time a new distance estimation is available, making the particle to condensate toward the real mote position, as exemplified in Figure 5.5 in Section 5.5. To solve

the outliers problem that likely occur when estimating the distance from RSSI measures, we also distribute a certain percentage of particles (namely, 5 %) uniformly in the environment. This enables the particle filter to recover from a totally wrong estimation, as in the case of the well-known "*kidnapped robot problem*" [80].

### 5.3.4 Multichannel RSSI-based ranging

Clearly, to gather RSSI measurements on different channels, nodes need to coordinate in order to concordantly change the carrier frequency. We designed the following protocol, which is initiated by the robot when the objects contacted during the Object Discovery phase enter Active mode at the rendezvous time[1].

The multichannel RSSI harvesting process occurs in successive rounds. Each round is initiated by the robot that broadcasts an `RSSI_GET` message. This message contains the list of *smart* objects that are required to collect RSSI samples, and the transmission order of the nodes. For compactness, nodes are identified by means of the Temporary Addresses assigned during the Object Discovery phase, rather than using the MAC addresses, which are typically longer. Channel access occurs according to a Time Division Multiple Access (TDMA) scheme: time is partitioned in transmission slots of constant duration (slightly longer than the transmission time of a full data packet), and each node is assigned to a single slot in an exclusive manner. Each node listed in the `RSSI_GET` message, then, waits for its assigned slot and, then, broadcasts an `RSSI_REPORT` message that contains the vector of RSSI values collected in the previous slots, included the robot's one. Furthermore, the `RSSI_GET` and `RSSI_REPORT` messages will also carry an indication of the RF channel that will be used in the following round. In this way, nodes that miss the robot's packet, but overhear a report message can synchronize again in the following round. We observe that the number of RSSI samples reported by the nodes is not homogeneous across the round, since the first nodes that transmit have not yet received messages from the others. To overcome this drawback, the robot may permute the transmission order of the nodes in each subsequent rounds. Furthermore, each round may be repeated multiple times, without changing channel.

[1]The reader can understand that RSSI measurements for single channel, to estimate the distance among motes and robot, it is a sub case of multichannel estimation. Therefore we give directly the protocol designed to multichannel.

Once again, the communication is unreliable and no ACK mechanism is considered. If a *smart* object fails replying to the robot's message for two consecutive rounds, its entry is deleted in the robot's memory and the Temporary Address is released. On the other hand, if a *smart* object does not receive *any* message (either from the robot or other objects) for a interval $T_{timeout}$, it switches back to the Quiet mode.

When the multichannel RSSI harvesting is complete, the robot can move into a new location and repeat the full process.

### 5.3.5 MDS

In the following we describe the MDS algorithm for object mapping that we used in our experiments. Let us enumerate from 1 to $n$ the *smart* objects included in the mapping process. Furthermore, let $n+1, n+2, \ldots, n+k$ denote the locations where the robot stopped to collect RSSI samples from the surrounding objects. In the following, we denote these positions as *virtual beacon* nodes. Let $\theta_i = (x_i, y_i)^T$ the vector of cartesian coordinates for node $i$. Our aim is to determine an estimate of $\theta_i$ for $i = 1, 2, \ldots, n$ knowing the exact position of the virtual beacons and the ranging values given by 5.2. The MDS approach consists in minimizing the following cost function

$$S(\Theta_k) = \sum_{i=1}^{n} \sum_{j=n+1}^{n+k} 2w_{i,j} \left( \hat{d}_{i,j} - d_{i,j}(\Theta_k) \right)^2 \tag{5.10}$$

where $\Theta_k = [\theta_1, \ldots, \theta_{n+k}]$ is the state vector, $\hat{d}_{i,j}$ is the estimated distance between *smart* object $i$ and virtual beacon $j$, whereas $d_{i,j}(\Theta_k)$ is the distance between the same nodes given the state vector $\Theta_k$. Finally, the scalar $w_{i,j} = e^{-\bar{P}^2_{rx_{i,j}}/P^2_{th}}$ accounts for the accuracy of $\hat{d}_{i,j}$ where $\bar{P}_{rx_{i,j}}$ and $P_{th}$ are respectively the power received by node $i$ from node $j$ , averaged over different channels, and the power threshold for ranging. The cost function $S(\Theta_k)$ can also be modified to include the measurements between *smart* objects in the following way:

$$\begin{aligned} S(\Theta_k) = \sum_{i=1}^{n} \Bigg( & \sum_{\substack{j=1 \\ j \neq i}}^{n} w_{i,j} (\hat{d}_{i,j} - d_{i,j}(\Theta_k))^2 + \\ & \sum_{j=n+1}^{n+k} 2w_{i,j} (\hat{d}_{i,j} - d_{i,j}(\Theta_k))^2 \Bigg). \end{aligned} \tag{5.11}$$

The minimization of $S(\Theta_k)$ cannot be performed in closed form, but the problem can be solved iteratively. Given the state vector at the iterative step $h$, $\Theta_k^{(h)} = \left[\theta_1^{(h)}, \ldots, \theta_{n+k}^{(h)}\right]$,

the next state can be computed by applying this simple updating function (see [76] for the details):

$$\theta_i^{(h+1)} = a_i \Theta_k^{(h)} \mathbf{b}_i^{(h)} \tag{5.12}$$

where

$$a_i = \left( \sum_{\substack{j=1 \\ j \neq i}}^{n} w_{i,j} + \sum_{j=n+1}^{n+k} 2w_{i,j} \right)^{-1} \tag{5.13}$$

and $\mathbf{b}_i^{(h)} = \left[ b_{i,1}^{(h)}, \ldots, b_{i,n+k}^{(h)} \right]^T$ is a vector whose entries are given by:

$$b_{i,j}^{(h)} = \begin{cases} \alpha w_{i,j} \left( 1 - \frac{\hat{d}_{i,j}}{d_{i,j}(\Theta_k^{(h)})} \right) & j \neq i \\ \sum_{\substack{\ell=1 \\ \ell \neq i}}^{n} w_{i,\ell} \frac{\hat{d}_{i,\ell}}{d_{i,\ell}(\Theta_k^{(h)})} + \sum_{\ell=n+1}^{n+k} 2w_{i,\ell} \frac{\hat{d}_{i,\ell}}{d_{i,\ell}(\Theta_k^{(h)})} & j = i \,, \end{cases} \tag{5.14}$$

with $\alpha = 1$ if $j \leq n$ and $\alpha = 2$ otherwise. The iterative procedure stops when $S(\Theta_k^{(h-1)}) - S(\Theta_k^{(h)}) < \varepsilon$ for a certain $\varepsilon$. We observe that, although the updating equations are simple to compute, the number of operations grows linearly with the number of virtual beacons, so that the execution of the MDS algorithm progressively slows down as the number of sampling positions increases. The same scalability problem, however, affects the other localization algorithms considered in our previous work. In particular the complexity of EKF is roughly order of $O(m\, n^3)$, with $m$ number of steps of the robot and $n$ number of objects in the area, while the complexity of the MDS algorithms is instead $O(n\, m\, L)$, where $L$ is the number of recursions performed by the algorithm to converge to the solution. The value of $L$ grows with the number of sampling positions $m$, though the dependence of $L$ on $m$ is not available in an explicit form. Nonetheless, we experimentally found that MDS is lighter than EKF for reasonable values of $n$ and $m$.

## 5.4 Visual object recognition and interaction

The mapping algorithm described in the previous section is generally capable of localizing the *smart* objects in the environment with a residual error of the order of magnitude of a meter. Although this precision may be sufficient to correctly steer the robot towards a target destination, it is not enough for enabling physical interaction between the robot and

the object. To this end, we need a much more precise localization of the object, which can be obtained by recognizing its appearance in the images provided by the robot's camera. Operatively, when the robot is in the surrounding of the object of interest, it starts sending `Descriptor Request` messages to that object. The object replies by sending packets containing descriptors of its appearance, which are passed to the object identification module in the robot. Note how each objects stores only its own descriptors, so the memory requirements for the motes are very limited and can fit inside the used motes' memory. The robot controller then executes the following steps:

- gets images from the on-board cameras;
- extract the descriptors of these images;
- compare these descriptors with those transmitted by the object's mote;
- if the object is recognized in the camera images, its position is computed from the descriptor's locations and is passed to the robot navigation module.

Clearly, the performance of the object recognition module depends on the method used to represent the object appearance. Ideally, the descriptors shall permit the robot to perform a fast and reliable recognition of the object in its visual perspective, irrespective of its distance, orientation and light exposition. The robot should also be able to recognize the object even if it is partly occluded by other elements of the scene. Many of these features are possessed by the *Scale-Invariant Feature Transform (SIFT)* descriptors [64]. Furthermore each SIFT feature descriptor occupies just 128 bytes of memory and, thus, the set of feature descriptors corresponding to the object can be stored in commercial motes. The extraction of SIFTs from the images grabbed by the camera and their comparison with other descriptors taken from a sample image is also fast enough to allow object recognition in about a second.[2] Moreover, extracted features are particularly robust to affine transformations and occlusions. Unfortunately, in indoor environments, most of the time the ambient is dim and the light is not enough for grabbing clear images. In this case, the images taken by the robot while moving will likely be affected by motion blur. To solve this problem, we propose the use of a new feature detector scheme called Motion Blur Invariant Feature detector (MoBIF)

[2]This value refers to the SIFT feature extraction and matching on an $1032 \times 778$ image using the hardware platform described in Subsection 5.5.1.

that was originally developed for humanoid robots [81]. Instead of trying to restore the original, unblurred images, the MoBIF approach uses an adapted scale-space representation of the image that tries to overcome the negative effect of the motion blur in the invariant features detection and description process. This approach can deal also with non-uniform, non-linear motion blur. Like SIFT, MoBIF descriptors are particularly robust to affine transformations and occlusions, thus allowing our approach to work correctly even in the case of partially occluded objects. Furthermore, MoBIF descriptors proved to perform similarly to SIFT on standard datasets and outperformed SIFT in images affected by motion blur or in dim images.

Unfortunately, MoBIF descriptor (like SIFT) are not robust to large perspective transformations nor to large rotations of the object along the vertical axis that occur when the robot observes the objects from very different points of view (as reported in [64] performances decrease when the rotation is larger than 30 degrees). Moreover, SIFT and MoBIF descriptors show a good invariance to the scale only until a certain limit (in the order of a couple of meters for the objects considered in this study). We addressed these two problems by taking many pictures of each object from different points of views, separated by $20 \div 30$ degrees (thus ensuring that there is always a view corresponding to a rotation for which SIFT descriptors work properly) and at several distances (i.e. $1$ m, $3$ m, and $5$ m). We then extracted the descriptors from each image and added each of them in a single *descriptor cloud*, in an incremental way. If a descriptor is too similar to another descriptor already present in the cloud, it is rejected. For example in our experimental setting we took images of each object from 18 different viewing directions and at three different distances. For each image, furthermore, we selected approximately the 10 most informative MOBIFs, so that the total memory footprint of the object description is at most about $70$ kB, which fits in the flash memory of 1 MB of TmoteSky nodes. The set of descriptors resulting from this process completely describes the external surface of the object. Searching for objects in a frame is hence done by looking for correspondences on this set. We remark that merging the information coming from all the pictures and deleting redundant descriptors makes it possible to recognize the object from any point of view, while maintaining a reasonable computational complexity.

Note how visual recognition is also very useful in order to distinguish between different objects placed too close to be distinguished from the RSSI technique: MoBIF descriptors have a very good matching accuracy and are able to distinguish between two not too similar objects. The only critical case is when almost identical objects are placed very close and the object recognition algorithm may fail because the visual features of the objects are very similar.

In order to improve the precision of the robot localization we also tested an improved version of the visual recognition system where a second camera has been added to the proposed setup.Each robot is so provided with two cameras arranged in a binocular stereo setup. This not only allows to improve the recognition performance but also, as well known from computer vision theory, allows to compute the object distance by triangulation from the positions of its features in the images of the two cameras.

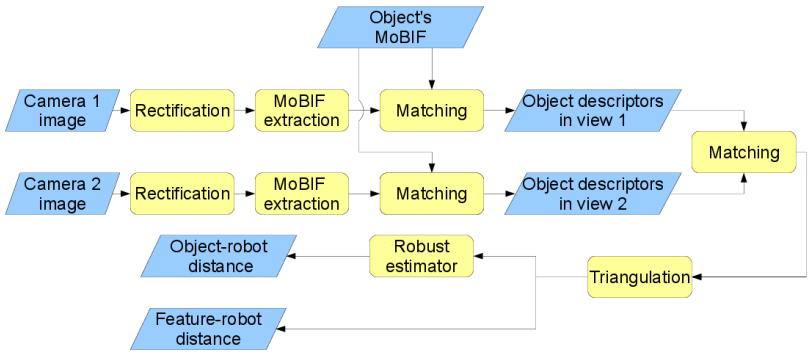



**Figure 5.3.** *Architecture of the stereoscopic feature extraction system*

We exploited the stereoscopic camera setup during the autonomous navigation using the approach depicted in Figure 5.3. First of all, before starting the autonomous navigation the two cameras are jointly calibrated, a rectification transform between the two images is computed [82] and the resulting rectification map is stored on the robot. This allows to rectify in real time the images acquired by the cameras during the navigation using the pre-computed map in order to make the descriptor matching easier. Then, as previously introduced, when the robot comes close to an object the MoBIF descriptors are extracted from the rectified images of both cameras and the descriptors extracted from each of the two cameras are matched with the object ones. Note that some of the object descriptors will be present in the images of both cameras while other ones could appear just in one of them because they could be out of the field of view, occluded by other objects or simply not

detected by the feature extraction algorithm. Following this observation a first advantage of using two cameras is that it is possible to get more feature points and obtain a slight improvement in the object detection performance. However the main improvement offered by the stereo setup is the additional information on the features locations. More precisely, as well known from computer vision, the projection of the same 3D point to two different views corresponding to a pair of cameras is shifted by an amount inversely proportional to the distance between the object and the cameras. In particular in the simple configuration of Figure 5.4 (referring to rectified images) it can be easily shown that the relation between the object distance $Z$ and the position shift of the feature location between the view of the first camera and of the second camera is given by [82]:

$$Z = \frac{(\mathbf{C_1} - \mathbf{C_2})f}{d_2 - d_1} = \frac{bf}{d_2 - d_1} \tag{5.15}$$

where $b = (\mathbf{C_1} - \mathbf{C_2})$ is the distance between the two cameras' optical centers $\mathbf{C_1}$ and $\mathbf{C_2}$ (stereo system *baseline*) and $f$ is the cameras' focal length (we used the same focal length for both cameras).

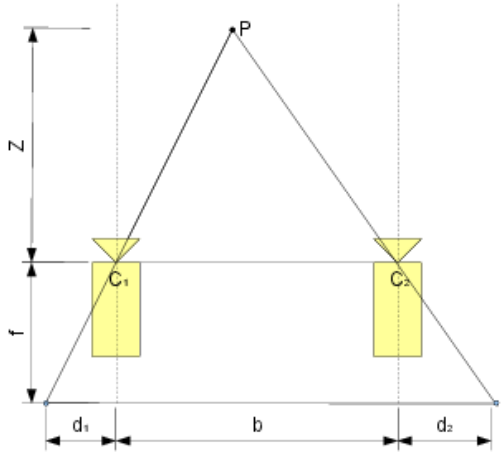



**Figure 5.4.** *Geometry of a binocular stereo setup*

After this computation for each couple of corresponding descriptors we have a distance measure between the feature point and the robot. The usefulness of this information is twofold: firstly all these distance values can be used independently when the robot is very

close to an object to help the robot in its interaction with the object. Furthermore when the robot is coming in proximity of one of the objects it is also possible to estimate the distance between the robot and the object by taking the average of the distance between the various features on that object and the robot. This corresponds to assume that all the features are located in the object centroid, thus introducing an error in the actual computed distance, however, at least for small objects, this approximation is reasonable. In computing the distance we also used the RANSAC [83] robust estimator in order to exclude from the computation outliers due to incorrect matches that can arise because of errors in the feature matching. These mismatches could appear, for example, if the object has simmetrical or repeating patterns and a point seen from one of the cameras is matched with another one in the other camera's image corresponding to another instance of the same pattern or if some features of the object get matched with others not belonging to it. The estimate of the distance between the robot and the centroid of the object obtained from the stereo setup can then be used as an initialization information for the WSN localization algorithm of Section 5.3 when the robot starts moving again towards a new object.

## 5.5 Experiments

In order to prove the feasibility of the proposed system and identify drawbacks and problems, we developed a proof-of-concept prototype that we used to run some experiments. Below we briefly describe the platform and present a selection of results.

### 5.5.1 The hardware platforms

Smart objects have been realized by glueing TmoteSky wireless sensor nodes to sample items. The TmoteSky radio transceiver is the Chipcon CC2420, whose PHY and MAC is compliant to the IEEE 802.15.4 standard, operating in ISM band at $2.4$ GHz and providing a bit rate of $250$ kbit/s. The module also provides an $8$-bit register named Received Signal Strength Indicator (RSSI), whose value is proportional to the power of the received radio signal. The core of the mote is the MSP430, a Texas Instrument low-power microcontroller, which is used to control the communication and sensing peripherals. The microcontroller is provided with $10$ kB of RAM and $48$ kB of integrated flash memory, used to host operating system and programs, whereas additional $1$ MB of flash memory is available for data storing.

Besides, the board is equipped with integrated light and humidity sensors. Motes have been programmed in NesC, using the TinyOS open-source operating system.

The robot, named *Bender*, was a custom-built wheeled differential drive platform based on the Pioneer 2 by MobileRobots Inc, depicted in Figure 5.1. The robot is equipped with a standard ATX motherboard with an 1.6 GHz Intel Pentium 4, a 256 MB RAM and a 160 GB hard disk, running Linux OS. The only on–board sensors are a stereoscopic camera and the odometers connected to the two driven wheels. Communication with the laboratory Intranet is provided by a PCMCIA wireless ethernet card, whereas the connection with the WSN is obtained by a Tmote Sky connected to one of the robot's USB ports.

### 5.5.2 Objects mapping experiments

In this section we give an example of SLAM with delayed initialization, thus the reader can familiarized with the objective of our work, latter we compare the performances of the three localization algorithms introduced in Section 5.3, namely EKF, PF, MDS, in terms of mean localization error. We also considered the MDS with inter-object RSSI measurements, denoted as MDS Internode.

Here we reported an experiment of delayed initialization strategy with the same dataset where our previous range-only SLAM approach, based on initialization with trilateration [72], failed due to a wrong initial guess about the mote positions. In such cases, the Extended Kalman Filter may quickly diverge, making impossible to correctly map the motes in the environment. With the proposed delayed strategy based on a particle filter, the initial mote position is estimated over several consecutive range measures, filtering out in a probabilistic way the outliers. As an example, we report in Figure 5.5 six consecutive snapshots of the SLAM convergence process with the proposed method. The stepwise solid line with right angles is the ground-truth path of the robot, whereas the less regular line shows the estimated path provided by the SLAM algorithm. Dots represent the motes' position hypotheses, corresponding to the particles of the particle filter. Comparing the successive frames, we observe that, before incorporating a new mote inside the map, the robot maintains and updates for some steps a set of particles for every new detected mote: the mote is incorporated once its position uncertainty decreases under a threshold. Despite the wide noise in the RSSI range measurements, our approach makes it possible to recover with a

good accuracy the robot trajectory and roughly estimate the motes positions (Figure 5.5 (f)). The accuracy of the proposed algorithm in localizing the different SOs is shown in Table 5.1 while Table 5.2 shows the accuracy of robot localization (the different *steps* correspond to various positions of the robot along its path).

**Table 5.1.** *SLAM Robot & Landmark: Landmark's mean error position*

| Mote ID | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| $e_x$ (cm) | 29.6 | -12.8 | -24.3 | 37 | 51.8 | -41.7 | -85.1 |
| $e_y$ (cm) | -46.1 | 3.4 | -7.3 | -17.3 | -0.2 | -29.7 | 2.6 |

**Table 5.2.** *SLAM Robot & Landmark: Robot's error position*

| *Step* | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|
| $e_x$ (cm) | -26.5 | 20.2 | 29.1 | -17.0 | 76.0 | 7.8 | 41.8 | -1.4 |
| $e_y$ (cm) | -35.8 | 33.2 | -4.5 | 91.9 | 24.3 | -35.4 | 11.3 | 44.1 |

Instead in Figure 5.6 and Figure 5.7 we report the results obtained with also PF and MDS in indoor (IN) and outdoor (OUT) scenarios, respectively. Figures (a) show the location of the nodes in the experiments and the trajectory followed by the mobile robot across the area. Each RSSI harvesting station along the path is marked by a cross. Figures (b) and (c) show the final localization error of each node for the different algorithms, when using single-channel (b) and multichannel ranging (c), respectively. Furthermore, in In Table 5.3 we collect the mean localization algorithm over all the nodes, in the different cases.

First of all, comparing the results achieved with the same setting in the two scenarios, we observe that all the algorithms provide better location estimate in outdoor, because of the less severe multipath fading. Second and more interesting, observing the results reported in Figures 5.6 5.7-(b) and 5.6 5.7-(c), we see that in almost all the cases the localization error of all the considered algorithms is reduced when using multichannel ranging rather than single-channel ranging. This experimental evidence confirms the intuition according to which averaging the RSSI samples over multiple channels reduces the uncertainty of the ranging estimate. The counterpart is that the collection of RSSI samples over multiple channels requires a more sophisticated communication algorithm and, in general, may take a longer time. However, we observe that with single-channel ranging, it is still necessary

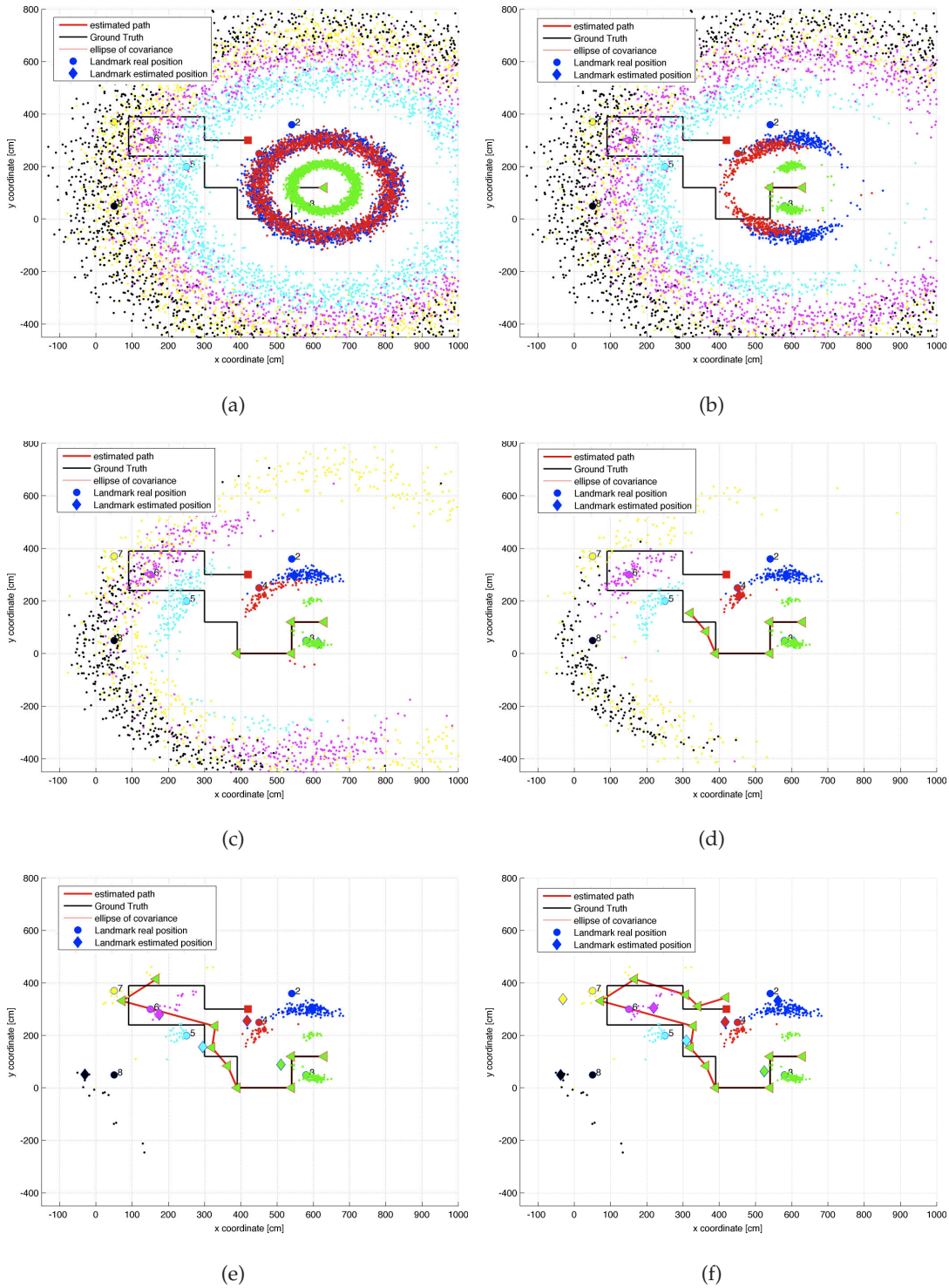
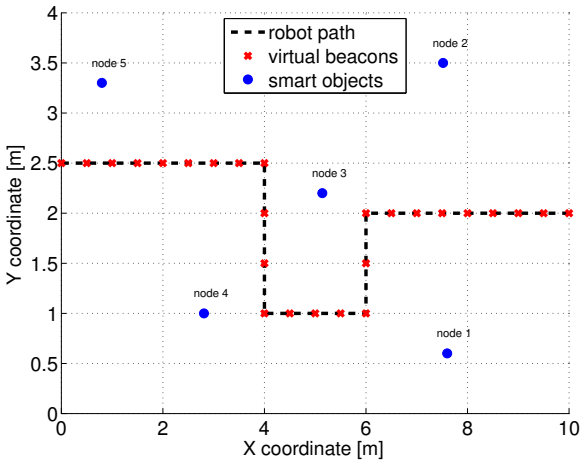


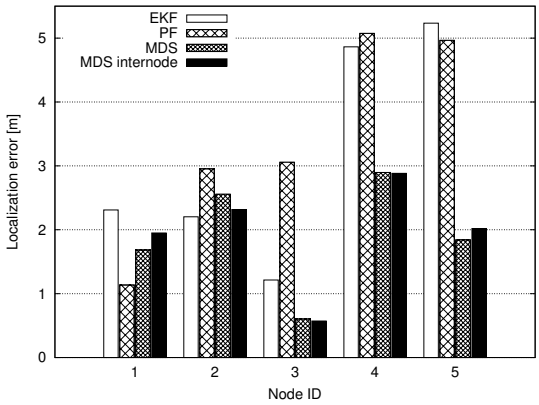

**Figure 5.5.** *The delayed initialization with the particle filter while the robot (the green triangle) is moving in the environment. (First row)(a) The robot has received the first RSSI measure and the annular rings are created around the robot, covering the motes; (b) The robot has moved; new measures are inserted in the particle filter and the samples cluster around the mote position; (c) the particle filter has converged and the node is initialized in the Extended Kalman Filter. (Second row) More complex situation along the robot path showing the convergence of multiple particle filters (d), (e), (f).*

to collect multiple RSSI samples for each pairs of nodes, in order to average the fast fading term. Conversely, with multichannel ranging we collect one or a few RSSI samples in each RF channel, but we repeat the operation in successive time instants in different channels, so that the fast fading is still averaged out when taking the mean RSSI value. Therefore, at the end the multichannel RSSI ranging takes approximately the same time as single-channel ranging. In particular note how the time taken to collect RSSI samples over $k$ different channels can be roughly estimated as $M = k\,(n\,T + S)$, where $n$ is the number of in-range nodes, $T$ is the slot duration and $S$ is the switching delay that accounts for the time taken by the nodes to switch to the next channel and receive the next `RSSI_GET` packet from the robot. With the TmoteSky sensor nodes we used, the slot time turns out to be approximately equal to $T \simeq 10$ [ms], while the switching time is $S \simeq 50$ [ms]. Hence, collecting RSSI samples over $k = 4$ maximally spaced-apart RF channels from $n = 10$ nodes takes approximately $M = 600$ [ms].

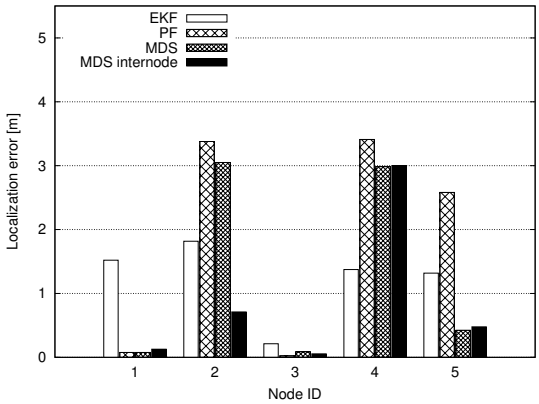
Finally, we note that the MDS Internode algorithm yields, in a few cases, slightly worse localization accuracy than the standard MDS. In the other cases, however, the MDS Internode scheme may provide significant improvements as, for instance, for nodes 2 in Figure 5.6. The reason is that rough internode ranging estimates may generally impact negatively on the localization accuracy provided by the MDS algorithm when the first guess of the nodes position is good. However, in case the nodes are severely misplaced at the beginning of the MDS algorithm, the availability of internode ranging information makes it possible to correct this deficiencies. This is the case of node 2 in Figure 5.6. In fact, observing the time evolution of the state vector $\Theta_k$ during the execution of the mapping algorithms (not reported here for space constraints) we could see that the initial guess for this node position, obtained by applying the Particle Filter initialization approach, was close to the position of node 1, which is actually symmetric with respect to the robot trajectory. With the path followed by the robot in this experiment, the EKF, PF and MDS algorithms were not able to recover node 2 from that erroneous initialization, so that the final localization error was large. Conversely, using the internode ranging information between nodes 1 and 2, the MDS Internode algorithm was able to correct the initial error and enhance the accuracy of the final position estimation of node 2.




(a)

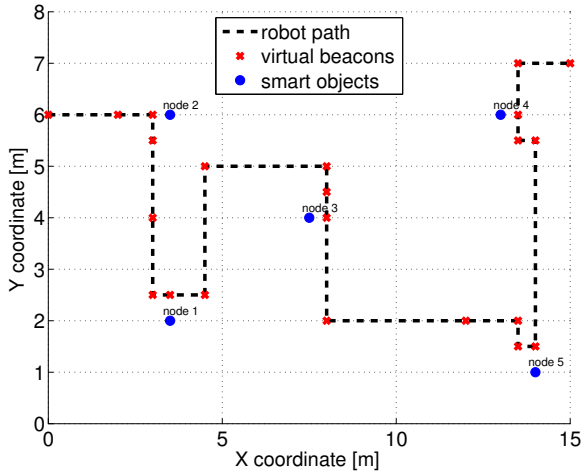



(b)

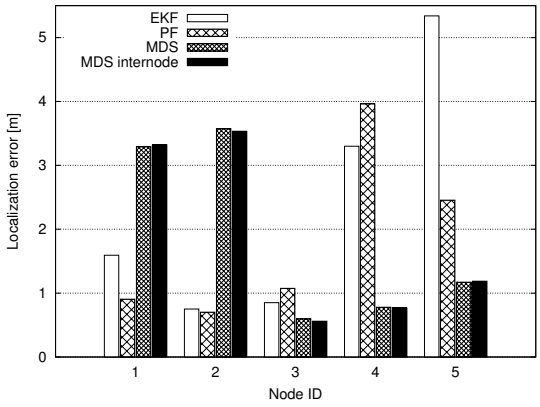



(c)

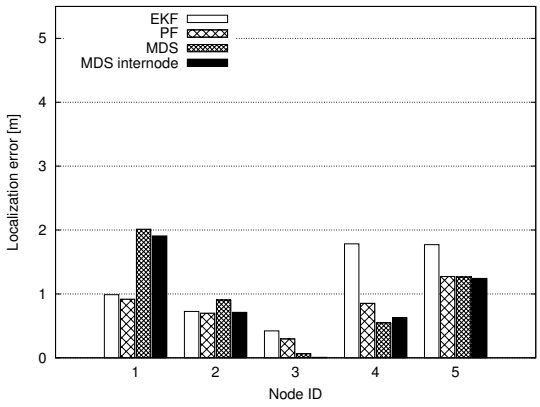
**Figure 5.6.** *Indoor scenario: (a) experimental setup, (b) mean estimate error using single-channel RSSI ranging, (c) mean estimate error using multichannel RSSI-ranging*




(a)




(b)




(c)

**Figure 5.7.** *Outdoor scenario: (a) experimental setup, (b) mean estimate error using single-channel RSSI ranging, (c) mean estimate error using multichannel RSSI-ranging*

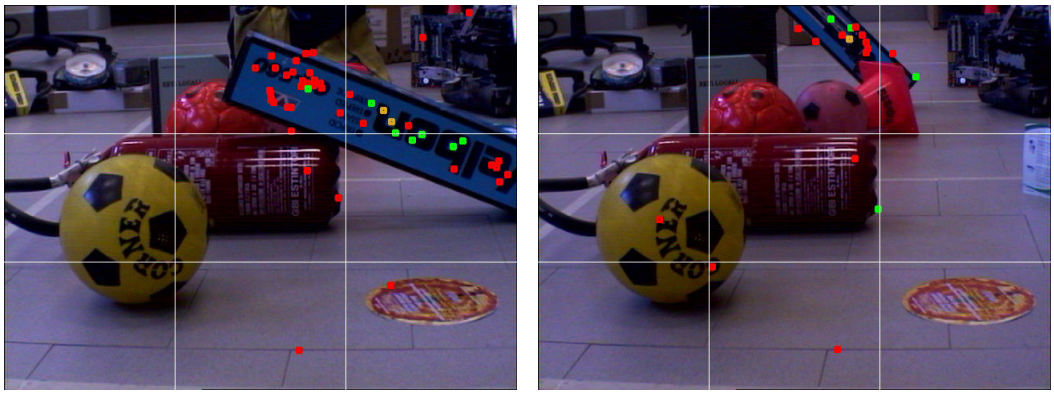
**Table 5.3.** *Mean localization errors for indoor (first and second rows) and outdoor (third and fourth rows) environments, using single-channel and multichannel ranging.*

| | EKF | PF | MDS | MDS Internode |
|---|---|---|---|---|
| IN singlech | 3.16 m | 3.44 m | 1.92 m | 1.95 m |
| IN multich | 1.25 m | 1.9 m | 1.32 m | 0.87 m |
| OUT singlech | 2.37 m | 1.82 m | 1.88 m | 1.87 m |
| OUT multich | 1.14 m | 0.8 m | 0.95 m | 0.9 m |

### 5.5.3 Visual recognition experiments

When the robot receives from the mote the MoBIF-based object descriptors, it starts looking for this object in the surrounding environment. In our experiments, we use several type of *smart* objects, as those depicted in Figure 5.1. In all the experiments we performed, the robot, in addition to correctly localize itself and build a map of the perceived motes, was able to correctly recognize the *smart* objects in its visual perspective. Figure 5.8 exemplifies the result of MoBIF descriptors matching (red dots) for a *smart* object, the blue box, in a cluttered environment. We notice that the matching is correct irrespective of the distance and orientation of the target object. In Figure 5.9 we show how by using the MoBIF descriptors we are able to obtain a correct recognition even in presence of motion blur.

As described in Section 5.4 we also introduced a second camera to improve the performance of the visual recognition module. Figure 5.10 shows the extracted features: it can be clearly seen that the matching of the extracted MoBIF features is very reliable. Table 5.4 shows the number of extracted features for the example of Figure 5.10-(a), there are features present in both cameras (shown in green) but some of the object features are present only in the left or in the right camera image (shown in blue and red respectively). This shows how the stereoscopic setup allows to extract and match more features than each of the two cameras alone, thus allowing a more reliable object matching. Furthermore Figure 5.10-(c) shows an example of occluded object detection. Even if a smaller number of features is available due to the occlusion, the robustness of the MoBIF descriptors to occlusions and the additional information provided by the second camera allows to correctly detect and



(a) (b)

**Figure 5.8.** *MoBIF descriptor match test in a complex environment and at different distances. The blue box is the object to recognize and in both images there are only a few incorrect descriptor matches. In the images the dots have different colors to identify at which cloud distance MoBIF is associated.*


(a) (b)

**Figure 5.9.** *Example of correct matches in presence of motion blur: above, a zoom of an image grabbed by the robot without motion blur, below the image grabbed while moving, thus with motion blur.*

**Table 5.4.** *Number of matched descriptors for the image of Figure 5.10-(a).*

| | Matched descriptors | | | |
|---|---|---|---|---|
| Camera | Obj. and both im. (Green) | Obj. and left im. (Blue) | Obj. and right im. (Red) | Not belonging to the obj. (Yellow) |
| Left | 26 | 105 | - | 982 |
| Right | 26 | - | 136 | 1117 |

**Table 5.5.** *Comparison between the distance estimates and the ground truth data.*

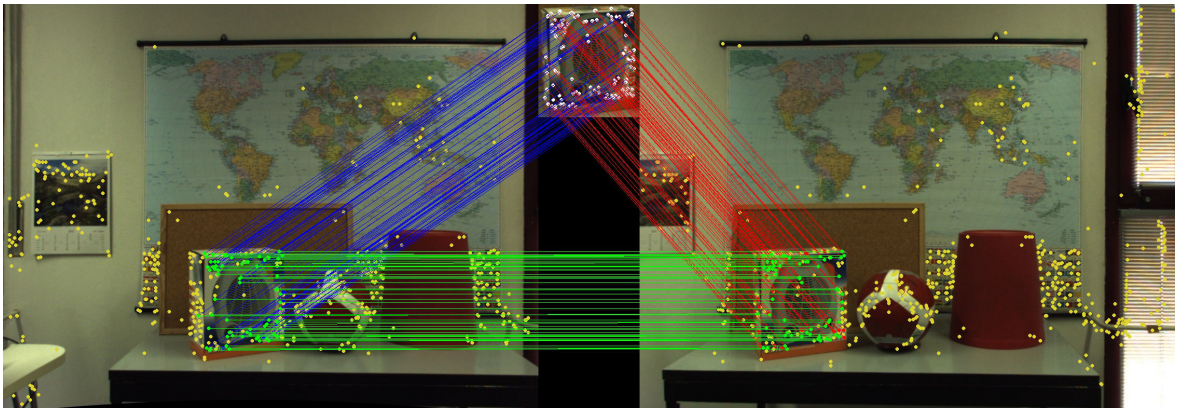
| Object | Estimate distance | Real distance | Error |
|---|---|---|---|
| Box | 133.7 cm | 140 cm | 6.3 cm |
| Ball | 136.1 cm | 140 cm | 3.9 cm |
| Plush | 130.3 cm | 133 cm | 3.3 cm |

localize also the partly occluded object.

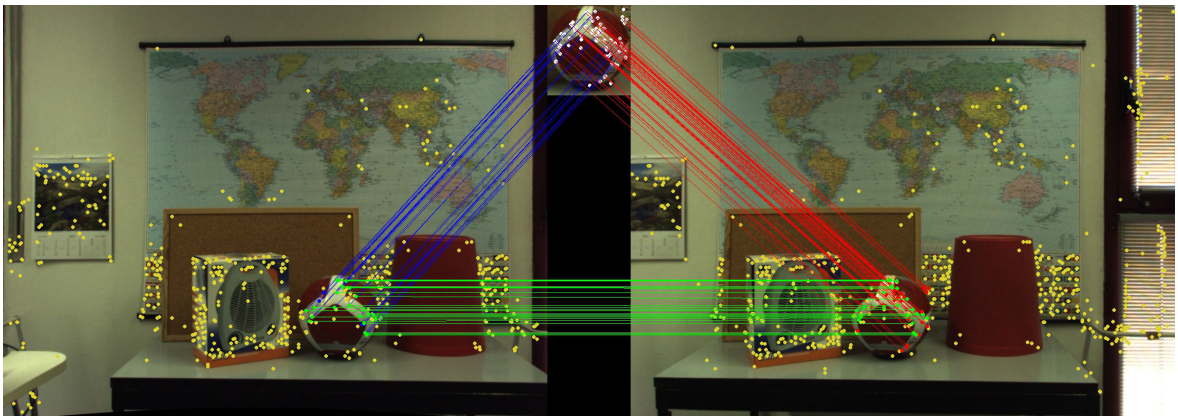
Following the approach introduced in Section 5.4 the features visible in both cameras can be matched together (as shown in green in Figure 5.10) and their respective position can be used to estimate the object position. The stability of the matching of the MoBIF descriptors and the RANSAC robust estimator provide a reliable distance estimate as can be seen from Table 5.5. The table shows the estimate of the distance between the object and the robot in the three example cases of Figure 5.10 and compares it with a ground truth measure acquired by a Time-Of-Flight sensor. The error in the estimates is just a few centimeters.

## 5.6 Conclusions and future work

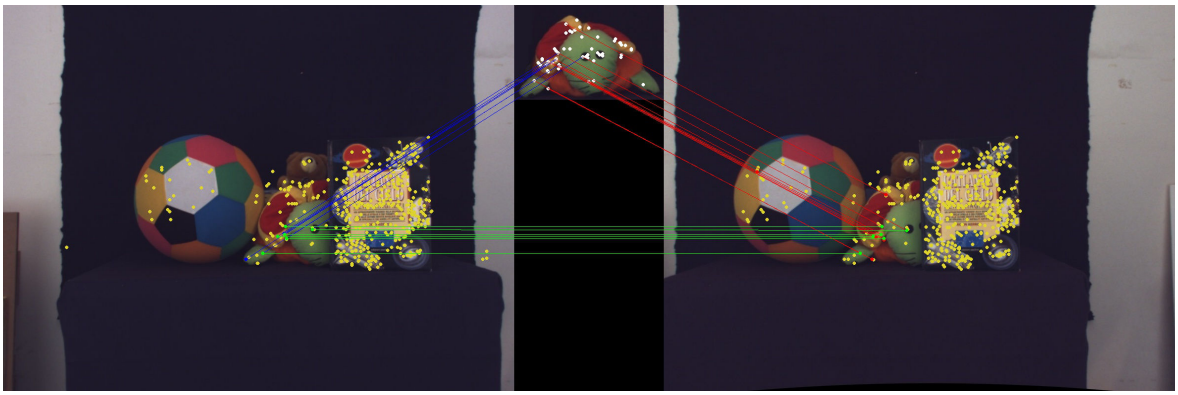
A system that enables the autonomous exploration of *smart* environments by a robot is presented in this chapter. The application stems from the capability of the robot to closely interact with some objects that are enabled to wireless communications and capable of simple computational task and limited data storage. The proposed approach allows the robot to progressively acquire environmental awareness by interacting with the *smart* objects located in the space. The feasibility of this vision has been proved by means of an experimental prototype of the system, in which a robot has proved to be able to discover the objects in radio



(a) Box experiment



(b) Ball experiment



(c) Occluded object (plush) experiment

**Figure 5.10.** *Feature matching with a stereoscopic setup: the green dots represent features present in the object and in both cameras while blue and red features are present in the object but only in the left or right camera respectively. Yellow features do not belong to the object.*

range by using RF communication, then to roughly map them into the area through an RSSI-based localization algorithm coupled with a proper initialization scheme based on particle filters, and finally to recognize the objects in its visual perspective by matching the information transmitted from the object with the appearance descriptors obtained from the onboard cameras.

Chapter 6

# Cognet: a cognitive network testbed with commercial devices

## 6.1 Introduction

The Internet of Things is a paradigm that covers many problems and many technologies. In this thesis we have often considered *smart* objects as IEEE 802.15.4 wireless communication nodes, but the same concepts can be applied to people as well. A person can use a smartphone and/or a tablet to communicate with other people or interact directly with other devices. In the case of WiFi technology, it is possible to create an independent basic service set (IBSS) network with tens of users that produce data traffic. As in WSN, the main objective is to improve the communication among other devices, and for realistic studies it is useful to create a testbed where it is possible to design, develop and finally test novel algorithms. The testbed must have the feature to implement cognitive network algorithms and even interact with the IEEE 802.15.4 testbed already implemented.

Cognitive networking (CN) [4, 84] is a network paradigm that deals with how wireless systems learn the relationships between the network behavior and its performance, and plan and make decisions to achieve local, end-to-end, and network-wide goals. CN, different from cognitive radio networking (CR) [85], tries to jointly optimize the different layers of the protocol stack. Novel MAC, routing, transport and application layer techniques for wireless mesh networks have been proposed and analyzed by the networking community, and can

be adopted in a cognitive network. The evaluation of their performance is not an easy task and is in fact an interesting research problem.

Different approaches have been used to compare the network performance in a realistic scenario. One of them is based on a discrete event network simulator (NS) [86], which is sufficiently accurate for most research works. NS is a powerful tool for a preliminary test, and the software developed to test a specific network scenario can be easily reused to test other techniques in slightly different scenarios. Unfortunately, it is not always clear how realistic such simulations are, as it is not possible to predict and simulate all the factors that play an important role in a real network scenario.

Another approach to test the network performance relies on ad hoc network deployments, appropriately designed to test a specific layer of the protocol stack. In [87] a limited number of nodes, deployed in fixed positions, are organized in a wireless network to compare the performance of two routing protocols, optimized link state routing (OLSR), and better approach for mobile ad hoc network (BATMAN).

In [88] another routing protocol, ad hoc on-demand distance vector routing (AODV) is compared with OLSR. Also in this case, the wireless nodes are not mobile. Another interesting network deployment is Roofnet [89], an experimental IEEE 802.11b/g mesh network composed of $50$ nodes, deployed on the MIT campus to test different routing protocols. However, such network includes only fixed nodes, and there is no mobility involved. The main disadvantage of these solutions is that the deployments are designed to only test some specific networking techniques, and they can not be easily reused to test different layers of the protocol stack.

There is another approach that includes most of the CR testbeds presented in the literature, which are based on specific and often expensive hardware. Among them, CalRadio [90] is an open and reprogrammable IEEE 802.11b-compatible development platform for experimental purposes at the data link layer (DLL), designed and developed at the California Institute for Telecommunication and Information Technology, UC San Diego. Instead, two examples of civilian testbeds based on expansive ad hoc devices can be found in [91] and [92]. The first one is the Virginia Tech Cognitive Radio Network (VT-CoRNET) and the second one is the Open-access Radio Testbed for next-generation wireless network (ORBIT), developed at WINLAB, Rutgers University. These testbeds can enable the CR paradigm, ac-

cording to which it is possible to learn in an automated fashion the behavior of the network by observing all networking parameters, and to exploit this knowledge to predict the future performance of the network and act accordingly.

Instead our way is to build a testbed that is flexible enough to be used in different tactical and civilian scenarios, easy to manage, based on relatively inexpensive hardware, and can be easily recreated by other researches around the world. In other words we are looking to balance the above approaches' drawbacks. We have chosen to use tablets and smartphones running the Android operating system (OS), since these devices are commercially available, relatively inexpensive, mobile, and highly customizable. With these devices, we have realized the Android Wireless Mesh Network (AWMN) testbed, based on the IEEE 802.11 standard.

AWMN can be used to test cognitive networking techniques, since each node can observe and modify its in-stack parameters (i.e., those parameters that can be directly observed at different layers of the protocol stack). Furthermore, the devices are equipped with other sensors to observe out-stack parameters (i.e., those parameters that are not directly related to the protocol stack, e.g., environmental or positioning data) that can be exploited to design and to test novel cognitive networking protocols.

In the rest of the chapter, the reader will have overview some network testbeds, which, similarly to AWMN, are flexible enough to test cognitive networking techniques at different layers of the protocol stack. Then we detail our approach, which is based on the following principles:

1. the nodes in our testbed are commercial devices (smartphones and tablets), which are relatively inexpensive if compared with the ad hoc hardware commonly used for building a wireless mesh network testbed;

2. the testbed allows the collection and the modification of parameters at different layers of the protocol stack, thus many cognitive networking techniques can be tested within such framework;

3. the testbed can be reproduced with a limited effort, as detailed in this chapter.

In a nutshell, the main contributions of this chapter are:

- a qualitative comparison among different networking testbeds and a description of the main advantages of a testbed based on Android;
- the realization of a wireless mesh network among Android devices with the description of the software modifications needed to enable such network using commercial devices;
- a study on how to observe and modify key TCP/IP in-stack parameters in an Android based system;
- some preliminary results to show the realistic time evolution of such parameters in our system.

In Section 6.2 is listed a subset of the most known CR/CN deployed in academic structure around the world. In Section 6.4 we describe the software changes needed to enable mesh networking in an Android device, while in Section 6.5 we detail how to modify the Android OS in order to measure some key networking parameters. Then, in Section 6.6 we describe the testbed deployment and provide some preliminary results on the time evolution of such parameters. Section 6.7 concludes the chapter and presents some future work.

## 6.2 Network testbeds: an overview

There exist a few testbeds which are flexible enough to test cognitive networking techniques, and are more realistic than NS, and more flexible than the networking deployments designed to test specific routing techniques. They are summarized in Table 6.1, where they are compared in terms of hardware and deployment characteristics.

These testbeds are based on software defined radio (SDR), a radio communication system where all the layers (including PHY) of the protocol stack are implemented in software on a device. The device is connected to an ad hoc hardware responsible for producing the modulated signal. The universal software radio peripheral (USRP) is a common choice to build the physical layer on a CN node; another option is to use the wireless open access research platform (WARP, http://warp.rice.edu/). The architecture of a CN node is detailed in Figure 6.1, where the protocol stack is implemented in software and the modulation is implemented in a USRP2 (an advanced version of USRP), which includes a field

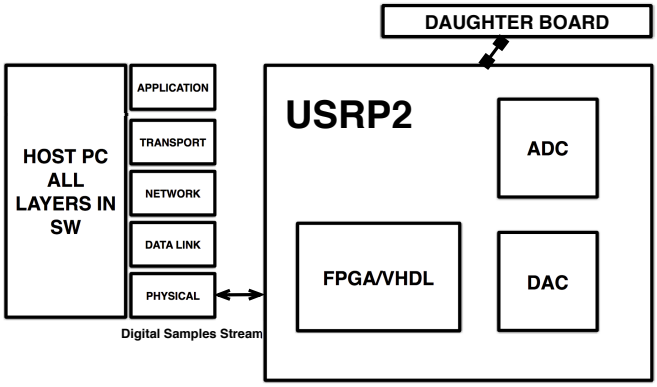



**Figure 6.1.** *Cognitive Networking node with USRP2.*

programmable gate array (FPGA), a digital to analog converter (DAC) and an analog to digital converter (ADC). The analog signal is received and transmitted through the daughter boards (RF transceiver), where each daughter board can operate in a specific frequency range. The USRP family device can hold a single transceiver daughterboard, and multiple devices can be connected together to form a multiple-input multiple-output (MIMO) system. Instead a single WARP board can attach up to four daughterboards to create a MIMO 4x4 system.

The main disadvantage of USRP is the latency introduced by the GNU Radio and OS kernel, by the wired communication between the host computer and the USRP, and by the USRP hardware itself [93]. With the WARP board, instead, both PHY and MAC layer can be implemented on the FPGA to reduce latency. However, the FPGA on board of WARP is not powerful enough to accommodate a full-duplex communications, which is often desirable in Cognitive Radio testbeds.

The Virginia Tech cognitive radio network (VT-CoRNET) [91] is a SDR testbed developed using 48 USRP2 nodes, which are a newer version of the USRP nodes. The nodes are spread over four floors of a building and equipped with a custom-made daughterboard spanning a frequency range from 100 MHz to 4 GHz. There exists a similar project also in Europe, at the Trinity College Dublin, Ireland, named implementing radio in software (IRIS) [94]. The main idea is the same as for VT-CoRNET: it is composed of a highly flexible architecture for real-time radio reconfiguration. Its nodes are fully reconfigurable SDR (USRP2).

Open-access radio testbed for next-generation wireless network (ORBIT) [92] is a hetero-

**Table 6.1.** *Selected Cognitive Networking testbeds.*

| Testbed name | Location and website | Hardware | Number of nodes | Deployment | Access |
|---|---|---|---|---|---|
| VT-CORNET | Virginia Tech<br>http://cornet.wireless.vt.edu/ | USRP2 | 48 | Spread over 4 floors | Open |
| IRIS | Trinity College Dublin<br>http://www.crew-project.eu/iris/ | USRP2 | Not specified | Not specified | Open |
| ORBIT | WinLab (Rutgers University)<br>http://www.orbit-lab.org/ | USRP1-2<br>PC Orbit | 28<br>400 | 20x20 grid with 1-m spacing | Open |
| EMULAB | University of UTAH<br>http://www.emulab.net/ | USRP1-2<br>PC Emulab grid<br>PC Emulab spread | 35<br>36<br>18 | Not specified<br>grid size is 300 cm x 224 cm with 6x6<br>nodes spread on multiple floors | Open |
| UCR | UCR<br>http://networks.cs.ucr.edu/testbed/ | WARP<br>USRP<br>PC UCR a/b/g<br>PC UCR n | 6<br>16<br>50<br>8 | Not specified<br>spread on single floor<br>spread on single floor<br>spread on single floor | Closed |
| CORES | UCLA<br>http://cores.ee.ucla.edu/ | BEE2 and USRP2 | Not specified | Not specified | Closed |

geneous testbed developed at WINLAB, Rutgers University. It is made of 23 SDR (USRP and USRP2), and 400 programmable radio nodes with standard WiFi cards, such as the Atheros chipset with driver MADWifi or the Intel chipset with driver IPW2200.

EMULAB [95] (University of Utah) is a testbed focused on higher-layer network protocol research, both wired and wireless. The nodes are Pentium III PCs, 18 are deployed on multiple floors of a building, and 36 are inside a laboratory. There are also 35 SDR nodes (USRP and USRP2) connected to the testbed.

Another heterogeneous testbed is deployed on a floor of a building at UC Riverside [96]. It is composed of 50 IEEE 802.11a/b/g nodes, 8 IEEE 802.11n nodes, 6 WARP boards and 15 USRPs.

A different testbed is CORES [97] (UCLA). The testbed is based on the Berkeley emulation engine (BEE2), a generic multipurpose emulation platform. It includes a multi-FPGA emulation engine, and it is possible to connect up to 18 front-end boards via multi-gigabit interfaces. These interfaces can be divided into primary and secondary users, in order to test cognitive networking protocols.

All these testbeds are composed of a very flexible architecture, and allow the implementation of most of the recently designed cognitive networking techniques to optimize different layers of the protocol stack. VT-CoRNET, ORBIT and EMULAB can also be controlled remotely, and are available to the research community for network experiments. There are

two main drawbacks which are common to all these approaches. (1) The network nodes are realized with specific and often expensive hardware, so that replicating such testbeds at different locations requires a considerable investment of time and resources. (2) In such testbeds, either there is no mobility, or the mobility is simulated via software (as in ORBIT), or the mobility is limited.

In our testbed we address these two drawbacks and design a flexible network testbed, which is composed of commercial devices easy to reconfigure, inexpensive and highly mobile.

## 6.3 Android ecosystem

Android is the most popular operating system for portable devices (smartphones and tablets), running on about 70% of the smartphones in the world. Android is very flexible thanks to its software development kit (SDK), i.e., the toolkit needed to develop an application (APP), which is released for all the three main commercial OSs for personal computers: Microsoft Windows, GNU/Linux, and Apple OSX. As a result of this open source philosophy, there exists a huge Android software community that shares key information to develop new APPs.

The core of the Android system is a Linux kernel, whose source code is publicly available. It can be modified and compiled through the native development kit (NDK), a free toolset released by the Android team to cross compile C/C++ code. Thus, developers with basic programming skills on Linux kernel module can design, write and enable new features in a relatively simple way. In particular, thanks to the NDK, it is possible to modify the Linux kernel source code and change the communication mode of the mobile devices, thus enabling the pure ad hoc communication mode. Furthermore, it is possible to observe and modify some network parameters at different layers of the protocol stack, thus allowing a real implementation of cross-layer techniques that follows a cognitive network paradigm.

Similar to the GNU community, the Android software community can be a valid help in this task because the developers are led to share their own information and ideas to improve the device functionalities.

The rest of the Android OS structure is divided into three macro layers. The top layer is the Application layer, where there reside APPs like contacts, browser and several manager

programs like location manager, network manager, etc. The application programming interfaces (APIs), provided by the SDK, are designed to exploit the OS architecture such that an APP can run in almost all devices. In this way, the hardware device is completely transparent to the the APP developer. The middle layer provides all libraries and the Dalvik Virtual Machine necessary to interface an APP with the lower layer. The lower layer is based on the Linux kernel. The Android team leaves the possibility, by rooting the device, to customize it by modifying directly the Kernel source code. Indeed, there are some limitations to this flexibility, e.g., when an APP needs to use some very specific hardware functionality that may not be present in all devices.

The flexibility of the Android ecosystem has been exploited by some software projects to develop an Android network testbed. In this section, we describe the two that are most closely related to our work. Commotion [98] aims to create a decentralized mesh network through an open-source communication tool that uses heterogeneous devices like mobile phones, computers, and other wireless devices. Serval [99] aims only to create a mesh network through mobile devices. Serval started like an independent project, and now is a branch of the Commotion project.

In Serval, an Android–based multi–hop ad hoc network is implemented among smartphones running the Android OS. The mobile phones can communicate even in the absence of a fixed infrastructure via the ad hoc network mode. Unfortunately, these projects do not allow the observation and modification of in-stack parameters, since they are designed for a more specific application to provide voice calls, text messaging and file sharing between mobile phones using WiFi, without the need for a SIM or any other infrastructure like mobile cell towers, WiFi hotspots or Internet access. These features are very useful when the mobile network is down during and/or after a natural disaster.

## 6.4 Android wireless mesh network

The implementation of the Android wireless mesh network (AWMN) can be divided into three phases. First, an ad hoc communication among the nodes should be enabled. Then, the nodes should be organized in a multi–hop network, and finally some networking parameters at different layers of the protocol stack should be read and modified, thus allowing the implementation of the cognitive networking paradigm.

### 6.4.1 Ad hoc communication mode

In a standard Android OS, the only node to node communication mode allowed by default is a high level paradigm based on a client/server architecture named WiFi Direct [100]. Although WiFi Direct allows the direct connection between two devices, the routing can be implemented only as an application above the transport layer, limiting the efficiency and the flexibility of a multi–hop network. Furthermore, only few MAC parameters are observable by the WiFi chipset on board of the device.

We adopt a different solution to enable the ad hoc mode, dealing directly with the IEEE 802.11 modes, called service sets. The standard one is the basic service set (BSS), in which a central controller (the access point) manages the connections among the devices, organized in a star topology. Another mode is named independent basic service set (IBSS), also known as ad hoc mode, in which each driver manages its own connections without a central controller. The IBSS is disabled in the most recent versions of Android, but it must be re-enabled to create a mesh network that allows multi–hop communications. Depending on the specific device, the IBSS mode is disabled either in the Application Framework or in the Linux Kernel. In the first case, in order to connect a device in IBSS mode to a network, we use the *iw* tool, i.e., a tool to configure the IEEE 802.11 driver. In other words, the *iw* can force the device to connect to an ad hoc network. This is the case of, e.g., a Samsung Galaxy Tab 7.0 device with *Atheros AR6003* chipset and *ath6kl* driver.

In the second case, when the IBSS mode is disabled in the Linux kernel, it is necessary to act directly on the Linux kernel and to modify the driver source code to activate the IBSS mode and create an ad hoc network. This is the case of, e.g., a Nexus 7 device with *BCM4330* chipset and *bcmdhd* Linux driver. In most devices, in order to perform these operations, it is necessary to log in as a root user and to flash the ROM memory. We used the following patch for Nexus 7:

```
+++ b/drivers/net/wireless/bcmdhd/wl_cfg80211.c
-  BIT(NL80211_IFTYPE_STATION)
+ BIT(NL80211_IFTYPE_STATION) |
    BIT(NL80211_IFTYPE_ADHOC)
```

Another solution to enable the IBSS mode is to add a second WiFi card to the Android device, as shown in Figure 6.2. We have chosen the Atheros *AR9280* chipset, running the

**Figure 6.2.** *Two Android devices with an external WiFi card.*

*ath9k_htc* driver. This driver is very similar to the standard *ath9k*, which is used by many laptops and which implements the whole datalink layer, with the difference that the rate control algorithm runs on firmware, so it is difficult to modify. In the case of the *AR9280* chipset, however, the firmware source code is under open source license, thus also the rate control algorithm can be modified if requested.

### 6.4.2 Multi–hop network

Once the ad hoc network mode is enabled, the multi–hop routing at the network layer should be set up. We have adopted the optimized link state routing protocol (OLSR) [101], which is a proactive routing protocol for MANETs working over the datalink layer. OLSR sends control packets to regularly exchange topology information with the other nodes of the network. The source code of OLSR, named OLSR daemon (OLSRD), is available online (http://www.olsr.org/), and can be compiled through the NDK and installed, once the root user is enabled. In particular, it is necessary to modify *ifnet.c* in order to run OLSRD on any Android device.

### 6.4.3 Cognitive networking platform

The cognitive networking platform should enable the implementation of cognitive networking techniques, i.e., it should allow the observation and modification of networking parameters at different layers of the protocol stack. We developed a Client/Server software architecture, where the nodes in the network are servers and run the *CognetNode* (the cognitive software core we developed), while there is a remote PC which acts as a client and runs the *CognetManager* (a software to collect the network parameters of all the nodes, to change the values of some of them, and to manage the network).

*CognetNode* can exploit the two WiFi interfaces. The internal WiFi, which has limited flexibility in terms of the parameters that can be observed and modified, is used to communicate with the remote *CognetManager*, which can start or stop an experiment. The external WiFi is instead used to transmit packets to the other nodes of the network. Furthermore, *CognetNode* can observe and modify the networking parameters, with the TCP observation/modification (O/M) thread and the MAC O/M thread.

The TCP O/M thread is in the user space and can communicate with the kernel space (in particular, with a kernel module named *CognetModule*) via an inter-process communication (IPC) channel (Netlink socket). The *CognetModule* can observe and modify the values of the TCP parameters by observing and modifying the *struct tcp_sock*, a C structure which implements TCP in a Linux machine. This data structure contains all the information about the TCP connections, and is refreshed every time a new TCP acknowledge packet is received by the node. The TCP O/M thread can modify in real time some TCP parameters by creating some *proc* files which are stored in the folder */proc/COGNET*.

The MAC O/M thread can observe/modify MAC parameters through the NL80211 socket, which communicates directly with the driver [1]. A CognetNode operates in ad hoc mode, can hear all the other CognetNodes around, and can communicate directly with them. Therefore, the MAC O/M thread saves the MAC parameters as a function of the MAC address of the CognetNodes that are under its WiFi coverage. The sampling rate at which the MAC parameters are observed can be set by the *CognetManager*.

[1] In order to fully exploit the *ath9k_htc* driver a modification of the firmware is also required.

## 6.5 Network parameters: observation and action

The TCP O/M thread is able to observe the following parameters:

- congestion window (CWND): the congestion window value at the sender;
- slow start threshold (SSTHR): the slow start threshold used by the congestion avoidance algorithm;
- TCP CLAMP: the maximum size reachable by CWND;
- packets outstanding: the number of packets transmitted but not yet acked;
- packets acked: the number of packets acked by the last ack packet received;
- throughput: the number of bytes acked per unit time;
- packets in flight: the number of packets outstanding plus the number of packets retransmitted. This number is a function of the size of the CWND (the larger the CWND the more packets in flight, and consequently the higher the probability to have a timeout or a packet loss);
- round trip time (RTT) [ms]: the time interval from when a packet is sent to when the sender receives the corresponding ack;
- smoothed round trip time (SRTT): a weighted average of the past RTT. $SRTT[i] = (1-\beta) \cdot (SRTT[i-1]) + \beta \cdot RTT[i]$, with $0 < \beta \leq 1$;
- RTO: the retransmission timeout which is equal to twice the SRTT and varies between $200$ ms and $2$ min. A wrong setting of the RTO can induce a high rate of packet dropping.

**Table 6.2.** *TCP/IP stack parameters observed and modified for different devices integrated into our network testbed.*

| Observable/Writable | Layer | Parameters | Laptop b43 (bdcm4331) | Alix 3d2 ath5kl (ar5413) | Tab-7 P6210 ath6kl (ar6003) | Nexus 7 bdcmhd (bdcm4330) | USB Adapter ath9k_htc (AR9271) |
|---|---|---|---|---|---|---|---|
| Observable Writable | TCP | CWND | r/w | r/w | r/w | r/w | r/w |
| | TCP | SSTHR | r/w | r/w | r/w | r/w | r/w |
| | TCP | TCP CLAMP | r/w | r/w | r/w | r/w | r/w |
| | TCP | RTO | r/w | r/w | r/w | r/w | r/w |
| | MAC | CWmin | r/w | r/w | NO | NO | r/w |
| | MAC | CWmax | r/w | r/w | NO | NO | r/w |
| | MAC | AIFS | r/w | r/w | NO | NO | r/w |
| | MAC | TX power | r/w | r/w | r | r/w | r/w |
| Observable | TCP | IP address | r | r | r | r | r |
| | TCP | Port | r | r | r | r | r |
| | TCP | # lost packets | r | r | r | r | r |
| | TCP | # Packets in flight | r | r | r | r | r |
| | TCP | # Packets outstanding | r | r | r | r | r |
| | TCP | # Packets acked | r | r | r | r | r |
| | TCP | # Packets lost | r | r | r | r | r |
| | TCP | Throughput | r | r | r | r | r |
| | TCP | RTTvar | r | r | r | r | r |
| | TCP | SRTT | r | r | r | r | r |
| | MAC | Transmission Channel | r | r | r | r | r |
| | MAC | RSSI | r | r | r | r | r |
| | MAC | Bytes RX | r | r | r | r | r |
| | MAC | # Frames RX | r | r | r | r | r |
| | MAC | # Frames RX duplicate | r | r | r | NO | r |
| | MAC | # Frames RX fragments | r | r | r | NO | r |
| | MAC | # Frames RX dropped | r | r | NO | NO | r |
| | MAC | Bytes TX | r | r | r | r | r |
| | MAC | # Frames TX | r | r | r | r | r |
| | MAC | # Fragments TX | r | r | r | r | r |
| | MAC | # Frames TX retry count | r | r | r | NO | r |
| | MAC | # Frames TX retry failed | r | r | NO | NO | r |
| | MAC | Inactive station | r | r | NO | NO | r |

r = observable, w = writable, NO = to be implemented, TX= transmitted, RX= received, # = number.

It is possible to modify also some of these parameters, as specified in Table 6.2.

The MAC O/M thread can observe the following parameters at the data link layer:

- the number of frames transmitted (TX) and received (RX), as well as the number of bytes TX and RX. These parameters are recorded in the file */proc/net/dev*;
- the transmission channel, the transmission power and the received signal strength indicator (RSSI) can be observed through the specific driver Input/Output control function, named *ioctl*;
- the number of fragmented packets transmitted and received (# fragments TX/RX), the number of dropped frames (# frames RX dropped), the number of retransmitted frames (# frames TX retry count) and the number of frames whose retransmission failed (# frames TX retry failed);
- the arbitration inter-frame spacing (AIFS) time and the contention window range, specified by the minimum value (CWmin) and by the maximum value (CWmax) allowed.

The possibility to read and to modify these parameters varies as a function of the chipset used, as specified in Table 6.2.

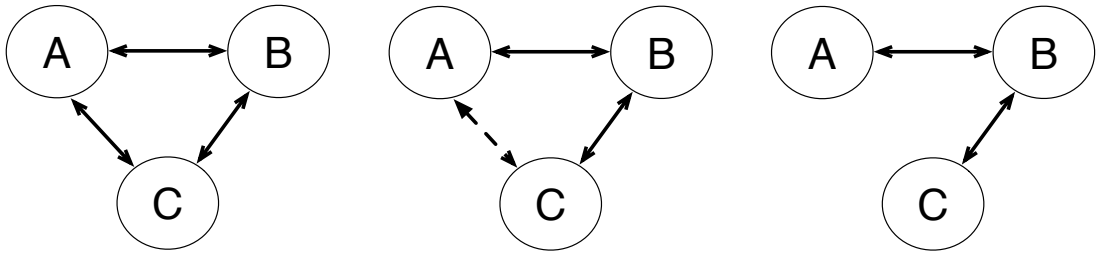
In a commercial tablet there are also some out-stack parameters available, which can be used by a cognitive networking technique, including:

- accelerometer: it can measure the value of the acceleration, in $m/s^2$, and its direction in the three axes;
- gyroscope: it is used to measure the orientation of the device in the three axes;
- global positioning system (GPS): if outdoor, it provides the geographical position of the device.

## 6.6 Results

The software solutions presented in the previous sections have been integrated in the deployment of the AWNM testbed, and have been tested through several experiments as reported in this section.

In the first experiment we tested the stability of the software that manages the ad hoc network. Others experiments tested the time evolution of some measured network param-




(a) All nodes are within coverage.

(b) *iptables* forces node C to drop all packets from A.

(c) OLSR updates the routing table.

**Figure 6.3.** *Changes in the network topology with* iptables.

eters as a function of different scenarios as single–hop, multi–hop static network and finally in a multi–hop mobile network to test the software stability.

We have deployed a wireless mesh network with several devices: a laptop with chipset *Broadcom bdcm4331* and driver *b43*; three Nexus 7 tablet connected to an external WiFi USB adapter with driver *ath9k_htc*; a Galaxy Tab 7.0 (P6210) tablet with chipset *Atheros AR6003* and driver *ath6kl*; an Alix board *3d2* with chipset *Atheros AR5413* and driver *ath5kl*. The Alix *3d2* board is an inexpensive board with low power consumption and limited capabilities, equipped with a 500 MHz (LX800) AMD Geode LX CPU, which can be exploited as a router or a firewall.

### 6.6.1 Experimental setup

We performed the networking experiments in our department (indoor scenario). Due to the limited size of the laboratory, all nodes were in line of sight of each others (the indoor range for such devices is approximately 40 meters), as shown in Figure 6.3-(a) for a simple case with 3 nodes. In the figure, node A is the source of traffic and node C is the intended receiver. In order to create a multi–hop network, we have used *iptables*, a tool available in every Linux distribution to set up different rules inside the firewall. In particular, we forced node C to drop all packets coming from node A, see Figure 6.3-(b). In this way, according to the OLSR routing protocol, since node A is not receiving any acknowledge message from node C, its routing table is updated and node A sends packets to node C via node B, as shown in Figure 6.3-(c), and the multi–hop network is set up.

The *iptables* commands to drop or to accept the incoming packets are:

```
iptables -A INPUT -m mac --mac-source MAC  -j DROP

iptables -A INPUT -m mac --mac-source MAC  -j ACCEPT

iptables flush
```

The data traffic is synthetically generated at node A using *iperf* [102], a common testing tool in the networking community.

Finally, in order to assure the repeatability of the experiment, at the beginning of the experiment we start a new TCP connection with the command:

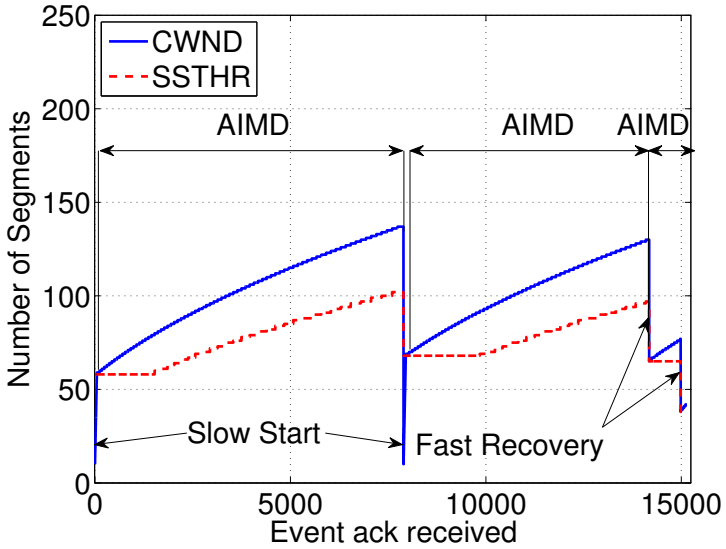
```
sysctl -w net.ipv4.tcp_no_metrics_save=1
```

### 6.6.2 Experiment: software stability

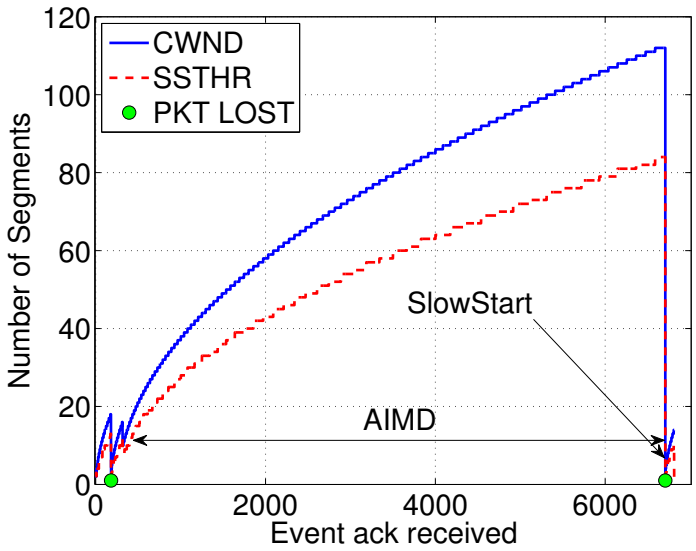
In the first experiment, there are four nodes that communicate through a multi–hop network, i.e., a Nexus 7, a Galaxy Tab 7.0, an Alix 3d2, and a laptop. The data is generated by *iperf*, and a stream of data is generated every $60$ seconds and lasts for $10$ seconds. The experiment is three days long, and during this period the topology of the network is changed dynamically every 6 minutes using the *iptables* tool. The purposes of this experiment are to test if the network testbed is stable and to figure out if there is any problem in updating the routing table with the OLSR protocol. We have verified that the routing protocol works correctly and that the network testbed can easily handle dynamic changes in the topology.

### 6.6.3 Single–hop experiment: TCP/MAC parameters evolution

In the second experiment, we have tested the software developed to observe the parameters, in order to observe the TCP and MAC parameters as a function of time. The list of all the parameters observed is reported in Table 6.2, where we detail for each device which parameters are observable, and which can also be modified. For this experiment, which lasted a total of $8$ hours, a single–hop topology has been created. In the figures we report a time interval of $30$ seconds only, since we want to focus on a single stream of data generated by *iperf*.




(a) Fast recovery mode with TCP Reno congestion avoidance algorithm.




(b) Slow start mode with TCP Reno congestion avoidance algorithm.

**Figure 6.4.** *Behavior of TCP RENO showing both fast recovery and slow start mode.*

In Figure 6.4 we can observe the two phases of the TCP Reno congestion control, i.e., the slow start mode, and the Additive Increase/ Multiplicative Decrease (AIMD) mode. In Figure 6.4-(a) we show the values of the CWND and SSTHR during the slow start mechanism that occurs when a new TCP connection is opened. In this case we set manually the CWND to 10 packets. The values of CWND and SSTHR are updated at every ack packet received,

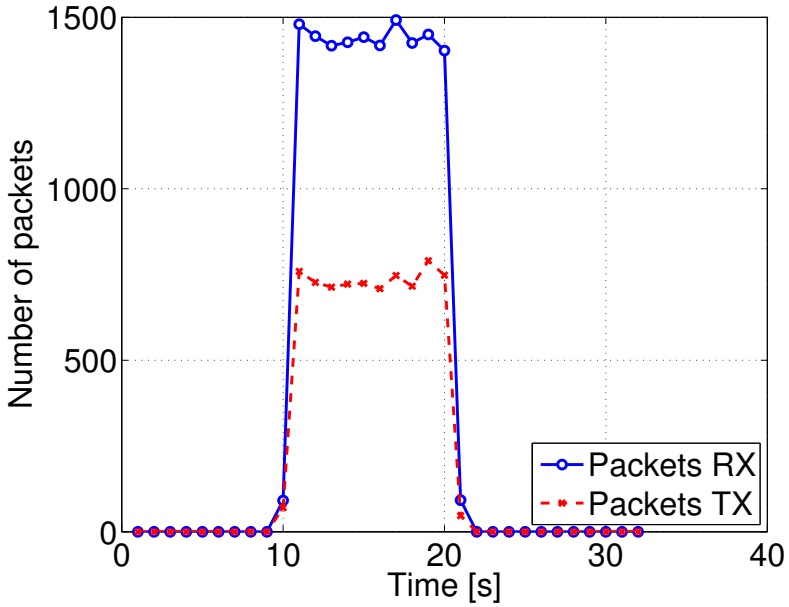



**Figure 6.5.** *Packets transmitted and received by the receiver node C.*

so we represent their value as a function of the number of acks received in a temporal window of 10 seconds[2] . Initially, the CWND exponentially increases until it reaches the current value of SSTHR, then the AIMD mode starts. We can also observe the fast recovery event that occurs after three duplicate acks are received. In this case CWND and SSTHR are both set to a value equal to half of the previous value of the CWND.

Instead, in Figure 6.4-(b) a packet is lost and the TCP Reno protocol goes into slow start mode with the CWND set equal to 1 packet. In this experiment we have also verified that the CWND value is equal to the sum of the packets in flight and packets acknowledged parameters when there are no retransmissions.

In Figure 6.5 we show the number of packets per second transmitted and received by the receiver node as a function of time. The total data received corresponds to a stream with bitrate of 17 Mbps generated at the source node by *iperf*, lasting for 10 seconds for a total of approximately 22 MB of data. We notice that the number of packets sent from the receiver are half of the number of packets received. This is due to the TCP delayed acknowledgment, a technique used to improve network performance in which the receiver combines together into a single response two acks, reducing the protocol overhead.

[2]Note that the fact that the value of the SSTHR observed is not constant between consecutive timeout events is due to an implementation detail of the corresponding TCP counter in the Linux kernel.

**Table 6.3.** *Parameters observed at the receiver node.*

| Average RSSI [dBm] | Average RTT [ms] | RX packets | RX fragments | RX duplicates | RX Dropped | TX retry count | Data RX [MBytes] |
|---|---|---|---|---|---|---|---|
| -62.1 | 65.7 | 15104 | 261 | 14 | 12 | 1035 | $\sim 22$ |

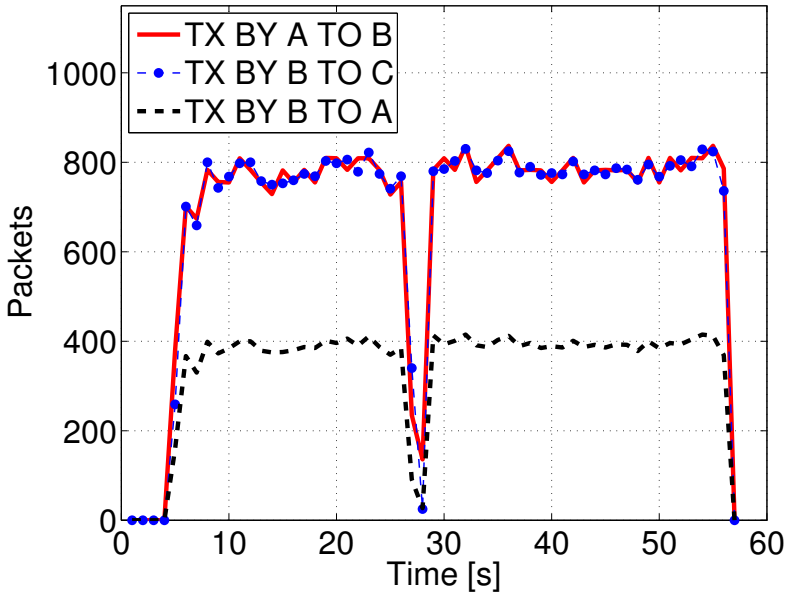



**Figure 6.6.** *Packets transmitted by the source node A and the relay node B.*

For this experiment in Table 6.3 we report a summary of the observed data during the 30 seconds of the experiment.

### 6.6.4 Multi–hop experiment: MAC behavior

Finally, in the last experiment we tested the software reliability and observed some TCP and MAC parameters as a function of time in a multi–hop scenario. The network topology is shown in Figure 6.3-(c), where node A is the source node, node C is the destination node and node B is the relay node. In Figure 6.6 we show the number of MAC packets sent by node A to node B, and the corresponding number of MAC packets sent by node B to node C. As expected, these two numbers are very close, since B is just a relay for the communication from A to C. In the same figure, we also show the number of MAC packets sent from node B to node A, which correspond to the TCP acknoledges sent by C to A and forwarded by B. We also notice that after 28 seconds node C becomes congested due to a timeout event, thus the number of packets transmitted and received rapidly goes to zero.

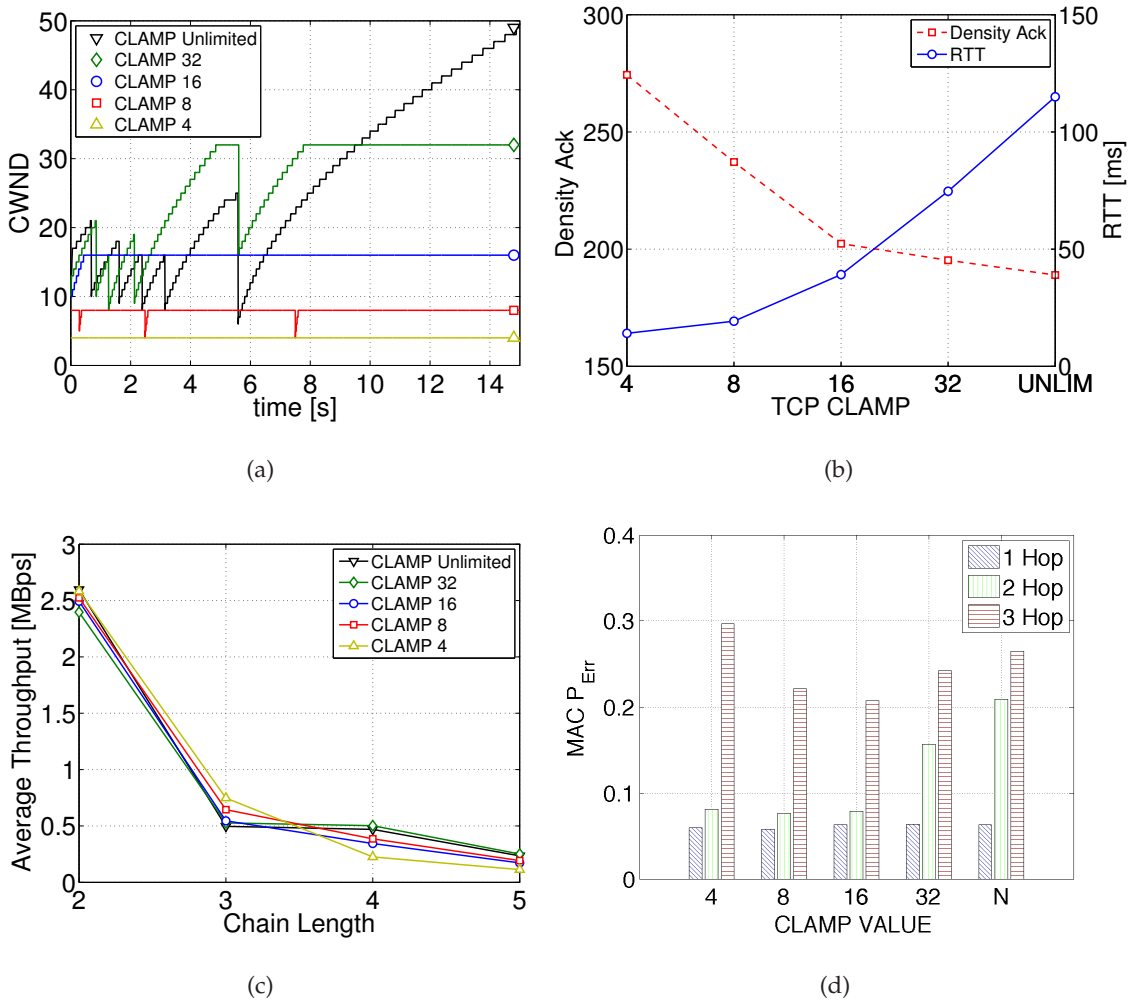



**Figure 6.7.** *Testbed measurement for different values of TCP_CLAMP: (a) CWND as a function of time, (b) average RTT and estimated ack density, (c) average throughput as a function of the chain length, and (d) estimated MAC probability of error.*

### 6.6.5 Multi–hop experiment: bounded CWND

In this experiment, we reproduce in our testbed an experiment described in [88] with OLSR, where a limit (TCP_CLAMP) is imposed to the CWND value. We performed twenty 60 s long experiments for each of the chosen values for TCP_CLAMP.

In Figure 6.7-(a) we show the results for a 2-hop topology with three nodes, where we can see that for small values of TCP_CLAMP the limit is rapidly reached and then the value of the CWND remains constant, while without a limit on the CWND value we have the standard TCP behavior, similar to what observed in [88] and [89]. Using the same topology, in

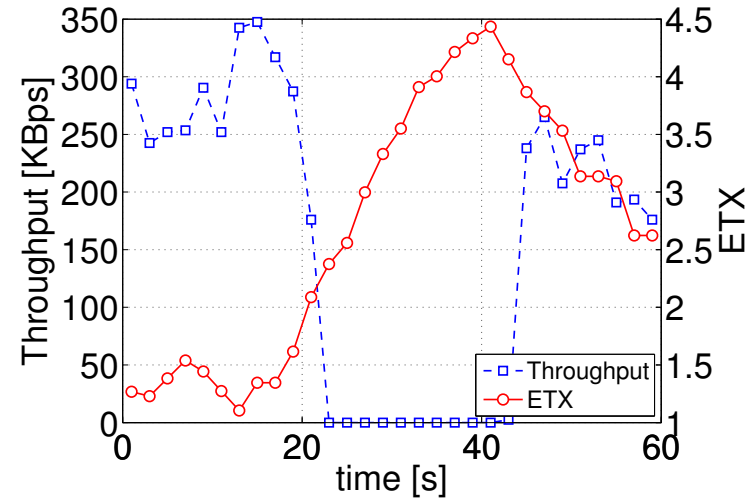
Figure 6.7-(b) we show instead the average values of the RTT and the number of ack packets per second, as a function of the limit in TCP_CLAMP. This shows that, for this particular network, in order to maximize the throughput we should use a value of TCP_CLAMP$= 4$. In Figure 6.7-(c) we observe the average throughput as a function of the number of nodes in the network, organized in a multi-hop chain topology. As expected, the throughput decreases as a function of the number of hops. We measured also the number of retransmissions at the MAC layer, and we estimated the retransmission probability as a function of the number of hops and of the value of TCP_CLAMP.

### 6.6.6 Multi–hop experiment: mobility

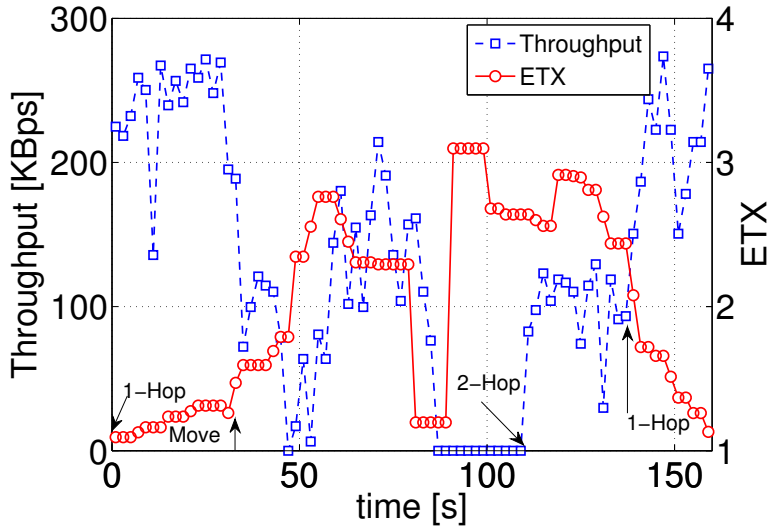
In this experiment, we observe the TCP throughput variation in the presence of mobility events for a network with three nodes, the source S, the destination D, and a relay R. We show two performance metrics: the TCP throughput and the *Expected Transmission Count* (ETX) metric for the path, i.e., the expected number of transmissions for a MAC packet (frame) in order to be received at the destination.

In Figure 6.8-(a), we have simulated mobility using *Iptables*, a tool available in every Linux distribution to set up different rules inside the firewall. In particular, we can force a node to drop or to accept the incoming packets from another node, thus forcing the network to be multi–hop even if all the nodes are within transmission range [103]. The experiment is organized as follows. At the beginning the three nodes are in range, thus node S sends packets directly to node D. After 20 seconds, with *Iptables* we block the direct communication between S and D, thus the topology changes and the transmissions are delivered through node R. We notice that it takes approximately $20$ seconds before the path from S to D through R is set up. This is due to the standard operation of the OLSR protocol, where a link quality hello (LQH) packet is sent every 2 seconds, and the quality estimation is done by averaging over the past 10 LQH packet received. The curve of ETX confirms this behavior.

Finally, in Figure 6.8-(b) we show the performance in the presence of real mobility. At the beginning the three nodes are in range. After 30 seconds node D starts to move in the opposite direction with respect to S, and after approximately 90 seconds the communication between S and D is interrupted. It takes 20 seconds for the OLSR protocol to establish another path through node R. At this point node D rapidly comes back to the initial position,




(a)




(b)

**Figure 6.8.** *Throughput and EXT in the presence of: (a) mobility simulated with* Iptables*, and (b) real mobility.*

and approximately 140 seconds after the beginning of the experiment the link between S and D is reestablished, and the final values of the TCP throughput and the ETX are similar to their initial values.

## 6.7 Conclusions

In this chapter was described AWMN a IEEE 802.11 mesh network testbed which is scalable and integrates relatively non-expensive Android based devices. The AWMN could be a valid solution for many networking research groups that are interested in creating their own cognitive networking experimental tool. It is composed of relatively inexpensive devices and is flexible enough to test different cognitive networking techniques at different layers of the protocol stack. We have provided the software details to create an ad hoc network among these inexpensive commercial devices specifying how to observe and modify the networking parameters at different layers of the protocol stack, and tested its functionality through some repeatable experiments.

The software developed to observe and manage the TCP and IP layers is fully re-usable on any GNU/Linux device. Instead, the implementation of the MAC layer depends on the specific chipset of the device, and also we gave the solution to avoid that problem.

The flexibility of this testbed will be exploited by our research group to test our newly developed cognitive networking techniques.

Chapter 7

# Conclusion and Future work

In this thesis we have discussed the design and development of different algorithms and their application to various IoT scenarios.

In detail, the task of the first three chapters was to find a way to design an IoT system able to handle the massive and heterogeneous amount of data collected in a typical IoT island, e.g., a Wireless Sensor Network. The main objective was to create a novel lightweight algorithm, called RAZOR, to compress signals (time series) of physical quantities while minimizing the amount of lost information to compress raw data given that RAZOR is a lossy compressor. At the same time, we wanted to minimize the amount of energy necessary to achieve a good compression. We designed RAZOR with the additional goal of classifying unknown signals. In order to achieve these goals, we exploited the Motif concept, a well-know technique used in data mining, with the purpose of classifying very long time series.

At first we tested the goodness of RAZOR with different design options and we selected the best designed solution evaluating compression and classification metrics. Afterwards, we evaluated RAZOR against the state-of-the-art signal solutions applied in WSN scenarios in terms of compression and energy consumption, and we reported the goodness of RAZOR in comparison with those compression algorithms applied in a WSN when the impact of errors was not negligible, and found that it is preferable compared to a more complex hierarchical solution. Regarding the energy consumption, it was measured by counting the number of CPU cycles for each operation necessary to compress the raw data for every algorithm tested. Moreover, to test RAZOR's ability to classify signals, we compared it to a solution based on the Motif concept, called EMMA , for the classification of very long time

series.

Our future research will focus on exploiting RAZOR to develop a data-based ontology capable of classifying signals depending on the actual data streams, and to use the classification information in order to optimize communication in the constrained part of the network. For this purpose, a first ontology system was already made to self–manage different kinds of sensors to handle the signals representing the different physical quantities gathered by each sensor. The ontology was designed to support heterogenous sensor nodes, in order to create a system able to self–manage different kinds on WS nodes. The idea was to move from the concept of physical node to the notion of logical sensor node, where the system could pair different devices to build in software a new node to extract new knowledge from the data gathered by each paired sensor.

Another line of research was based on WSN and other kinds of feature descriptors, SIFT/MOBIF, with the purpose to retrieve several physical objects in a typical home scenario through a videocamera installed on board a completely autonomous robot. In this scenario as well the feature descriptors were necessary to reduce the amount of data exchange. Contrary to the Motif used to classify time series, in this case the descriptor feature, SIFT, was used to accomplish the goal to retrieve a predefined object within a picture of the environment taken by the robot from a random position. Afterwards, a WS node was employed to transform a *dummy* object into a *smart* object to leverage the wireless communication for two reasons: the former was to exchange informations between the robot and the *smart* objects under the robot's radio coverage; the latter was to use every possible sensor on board the WS node to compute a rough localization of the *smart* objects. We compared different filtering algorithms and combined them to apply the SLAM paradigm for the localization and tried to mitigate the problem of the RSSI high variability affecting the IEEE 802.15.4 due to the multi–path problem. The obtained results suggested that the MDS algorithm is more accurate and lighter than classical EKF applied to the SLAM paradigm. In order to improve the RSSI value estimation, we designed and developed a communication protocol to change on the fly the transmission channel frequency, and we observed that the RSSI value is less sensitive to the multi–path problem. Hence, the estimated distance between robot and *smart* object becomes more accurate. Finally, we substituted the single videocamera on board the robot with a stereo videocamera to get a more accurate distance estimation between robot

and visible objects. We are also planning to extend the proposed approach in order to obtain a 3D localization of the robots inside a 3D scene representation, to allow a better interaction between the robots and complex environments. In this project, the final action will be to integrate this complete system with a robotics Brain Computer Interface (BCI). The BCI system will make it possible to select the required *smart* object by "*thinking it*" and the robot will retrieve it in an automated fashion. This will allow people with severe disabilities such as Amyotrophic Lateral Sclerosis (ALS) to interact with a domestic house or an intelligent ambient.

Finally, in this thesis we described a scalable IEEE 802.11 mesh network testbed, composed of relatively in-expensive Android based devices. We provided the software details to create an ad hoc network among these inexpensive commercial devices, and specified how to observe and modify the networking parameters at different layers of the protocol stack. The software developed to observe and manage the TCP and IP layers is fully re-usable on any GNU/Linux device. Instead, the implementation of the MAC layer depends on the specific chipset of the device. In order to by-pass this problem we added a second WiFi interface, independent of the device used. Moreover, this solution gives greater freedom during the testbed design because it is possible to exploit both WiFi cards either to perform experiments splitting the same ad hoc network in several islands with non overlapping different transmission channels or using the internal WiFi card to control remotely each single device. The results covered the validation of the most important network parameters in different scenarios such as single–hop, static multi–hop and dynamic multi–hop. The proposed testbed can be an important tool for future performance comparisons among cognitive networking techniques.

In future works, we are planning to extend the current capabilities of the proposed network testbed to the observation of other out-stack parameters, and to integrate such observations into a novel cognitive networking framework, exploiting both in-stack and out-stack parameters. Moreover, we are planning to improve the Android APP to display in real time the node networking status in terms of traffic data generated, congestion of the node and number of hops needed to reach the other nodes. Such APP can allow the user to select a cognitive networking technique among a set already implemented and to test in real time the performance of such techniques. In this way we will open up new opportunities to

switch in real time the cognitive network algorithm among an available set of possibilities, depending on the specific networking scenario and on the user needs. In the future, we will expect other groups to adopt this paradigm, which will build a network of developers able to enhance the platform functionalities. Therefore, our future plans also include a public release of this software and the creation of a user group.

Appendix A

# Anti-spoofing and open GNSS signal authentication

In this appendix we report a work finalized during the PhD program. It is not included in the main thesis' body because it is not directly related to the core issues of the PhD work. However it is possible to connect it with Chapter 6 because we implemented in Matlab software the entire encoder of a Global Navigation System (GPS) to simulate a GPS constellation. In this way exploiting a USRP it is possibile to recreate in the laboratory the entire GPS satellite constellation.

Global Navigation Satellite System (GNSS) signal authentication is a requirement for a number of applications. GNSS authentication has been proposed with aiding techniques that can be applied to the existing GPS and as a new security function for future GNSS. This work proposes a concept of a new authentication scheme based on signal authentication sequences that can be integrated in GNSS. The method works on systems that provide an open and encrypted service on the same frequency. The scheme would require minimum impact to the system. The architecture is explained in the different components of ground, space and user segment. A simulation of the architecture has been implemented in Matlab and performances and test results are shown.

The demand for techniques capable to authenticate the Global Navigation Satellite System (GNSS) signal and detect simulation attacks (spoofing) has increased exponentially in the last years, mainly targeted to financial and safety critical applications. The proposals and developments focused to two different directions:

- Aided GNSS authentication services: Leveraging of the existing services in order to detect signal spoofing and;
- Integrated GNSS authentication services: design of signal authentication schemes integrated in the system.

While the first approach can leverage existing and under development systems that do not provide a signal authentication services, such as the GPS C/A and Galileo E1-B signal, the latter requires a new system design and / or system architecture modification. This work focuses to a new design of an integrated authentication service, however both aided and integrated authentication proposals are reviewed and discussed.

**Aided GNSS authentication services.**

The concept of constructing aided GNSS authentication services was first explored in [104]. The idea was based on observing unpredictable information given by the Selective Availability (SA) characteristic (added noise to time and ephemeris) both on the receiver and a reference station in order to detect a spoofing attack. This concept lacked his security fundamentals when SA was switched off.

One characteristics of interest is the presence of unpredictable information in the signal and data that can be used as a mean of authentication: an attacker could not predict those information and therefore would not be able to simulate them. A similar approach has been proposed in [105]. In this method the GPS P(Y) signal on a particular time and location is sampled and transmitted to a remote party, which has sampled the same signal sample at the same time with a high gain antenna. A cross correlation of the two signals would results in a correlation peak if the received signal is transmitted by the satellites and therefore authentic.

**Integrated GNSS authentication services.**

The first attempt to integrate an authentication mechanism for open signals in GNSS was introduced by Logan Scott in [106]. The concept was based on secret spreading sequences, called spread spectrum security codes (SSSC), that were modulated in the signal for 10ms every 1 second of modulation with a known spreading sequence. SSSC are transmitted in the navigation messages and used for correlation with the received signal in order to verify

the authenticity. One limitation of such approach was the need to modify the existing modulation scheme, creating an important impact on the system infrastructure. Furthermore, introducing noise (the receiver cannot track the code for 10ms) could create implications in some DLL and PLL designs. A similar concept was explained in [107]. A proposed authentication scheme with less impact was later presented in [108] and further explained in [109] and [110]. In this concept only the navigation data is authenticated. One limitation of this scheme is that the navigation data can be acquired by a receiver and retransmitted with a different delay, exploiting a reply attack. It has been demonstrated as part of the trusted Innovative GNSS receiver (TIGER) project [111] how a trusted clock can limit this vulnerability.

## Signal authentication sequences architecture

Access control to satellite navigation signals can be implemented at the data level or at the signal modulation level. The data level access control foresees the encryption of the messages (in whole or partly). With this approach a receiver can perform the code search process, track the code delay and phase but cannot decode the encrypted message content. Therefore, if parameters such as transmitted time, ephemeris and clock errors are encrypted, a Position, Velocity and Time (PVT) solution cannot be computed.

Signal level access control requires the encryption of the ranging codes. A direct block cipher encryption of the code is typically a not preferred design option as a time limited code can be obtained with a high gain directional antenna (autocorrelation of a signal sample can be used to find the beginning of the code and time integration can be used to amplify the signal). A more robust approach is the use of a stream cipher, as the code never repeats. A stream cipher is a symmetric key cipher where plain bits are combined with a pseudorandom cipher bit stream (keystream), typically by an exclusive-or (xor) operation. In order to encrypt a Pseudo Noise (PN) sequence, the plain sequence is modulo 2 summed with the stream cipher, resulting in an encrypted PN sequence. For the purpose of the concept demonstration, it is assumed a signal with an open code modulated in-phase and an encrypted code modulated in quadrature. The transmitted signal (neglecting signal ampli-

tude) will be A.1 :

$$s(t) = \sum_{k=1}^{N} \left[ O_a^k(t) D^k(t) \cos(2\pi f_{L1} t) \right] + \sum_{k=1}^{N} \left[ O_b^k(t) SC^k D^k(t) \sin(2\pi f_{L1} t) \right] \tag{A.1}$$

Where *N* is the number of visible satellites, $O_a^k$ and $O_b^k$ are the publicly known spreading codes for every *K* satellite, $SC^k$ is the is the stream cipher and $D^k$ is the transmitted data. The same concept can be applied to a coherent adaptive subcarrier modulation (CASM) where multiple channels are multiplied together.

One design factor of our interest is the frequency of the stream cipher versus the chipping frequency of the PN sequence. We define (A.2) for our analysis this variable as Binary Stream cipher Carrier (BSC):

$$BSC(m,n) \text{ or } BSC(F_{sc}, F_c) \tag{A.2}$$

where $F_{sc}$ is the stream cipher $SC^k$ frequency, $F_c$ is the non encrypted code $O_b^k$ chipping frequency, $m = F_c/F_{sc}$ , $n = F_c/F_{ref}$ and $F_{ref}$ = 1,023 Mcps is the GPS C/A reference code. For example, a signal with a $O_b^k$ chipping rate of 10.23Mhz and a stream cipher $SC^k$ frequency of 1Mhz will be encrypted with a *BSC(10,10)*.

The objective of the system is to authenticate the open GNSS signal. The proposed security architecture can be integrated in GNSS that use direct-sequence spread spectrum (DSSS) as modulation technique. The system shall provide on the same frequency an open signal service, where the spreading code is publicly released, and an encrypted service where the spreading code is ciphered.

The security concept is based on the unpredictability of the encrypted PN sequence, which is assumed to be generated *a priori* by a secure function. The concept is the following: the stream cipher $SC^k$ is observed in a predetermined period. A portion of the binary sequence is extracted with its epoch time reference for a specific time frame, e.g. $SC^k[0 : 5000, n_0]$ if 5000 chips are observed at the discrete time $n_0$ . The sequence is processed, and transmitted in the open service navigation messages together with an authentication and/or encryption scheme. This message is defined as signal authentication sequence (SAS). SAS is defined asA.3:

$$SAS^k = SC^k[n_0 : n_0 + l] \tag{A.3}$$

Where $l$ is the length of the SAS code and $n_0$ is the first chip of the $SC^k$ observation time.

During the open service data decoding process, and after authentication verification or decryption, the receiver obtains the SAS, generates the PN sequence for that specific epoch and correlates it with samples of the encrypted code. The correlation result is fed to a security algorithm that determines the signal security state based on a estimated threshold.

### Architecture overview

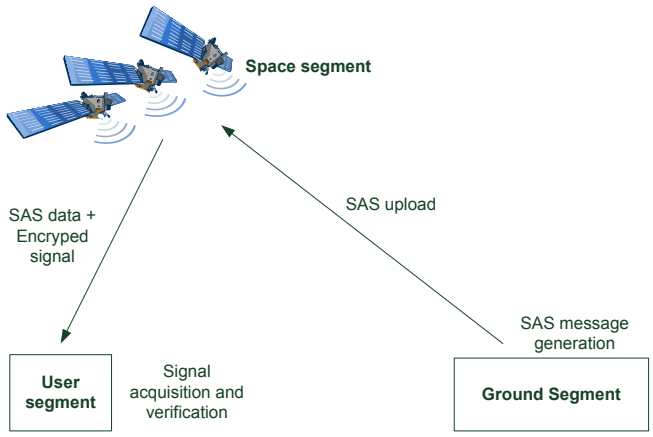



**Figure A.1.** *Architecture overview*

Figure A.1 describes an high level architecture of the authentication architecture. The SAS messages are generated at the ground segment and uploaded to the satellites together with the navigation messages. The SAS messages are transmitted in the open signal and received by the user receiver, which verifies the integrity and/or decrypt the data content. The user receiver also acquire the encrypted message at the predefined epoch and verifies the signal authenticity with an algorithm described later. The receiver determines the signal authenticity of the signal.

### Ground segment

The SAS messages are generated on the ground segment. The idea is that a proper dedicated service uses a key management facility (KMF) and an encrypted code generation facility (ECGF) to obtain the cipher stream in a particular epoch as defined in (A.3). The binary sequence is than formatted and packed into the messages by a message generation facility (MGF) (Figure A.2).

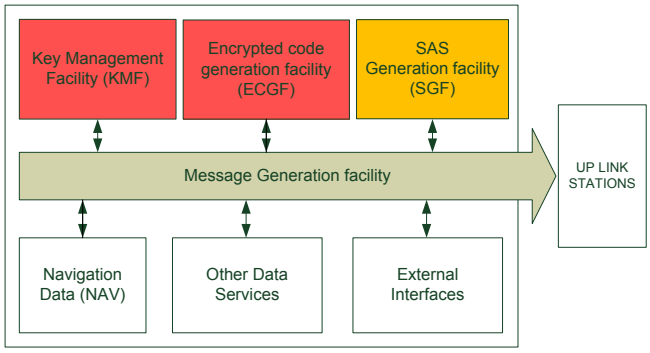



**Figure A.2.** *Ground segment SAS block diagram*

### Space Segment

The SAS are transmitted in the open signal data messages. Considering the evolution of GNSS and the number of different existing services, is not the objective of this work to define a protocol for the SAS. Instead, various considerations for design parameters and specifications are provided:

- Message overhead and SAS Data truncation. A number of navigation data must be received by the receiver with a certain priority, such as Time of Week (TOW), clock corrections and ephemeris data. Therefore SAS messages shall be transmitted in order to not interfere with such data. Depending on the channel bit rate and spare available data the designer of the system can decide to transmit the entire SAS sequence or to truncate it in sub-messages. The SAS size is determined by a number of factors, including encryption scheme and modulation type, signal power, expected receiver noise floor;

- SAS advance / delay approach. There are two approaches to SAS transmission: advance the SAS in time with respect to the encrypted signal or delay the transmission. Delaying the SAS would reduce the risk that an attacker regenerates the sequence. The Time to Alert (TTA) would be proportional to the frequency of SAS transmissions;

- The SAS message can be authenticated only or can be encrypted. This design option shall be considered taking into account two factors: a) privacy of the encrypted code. Releasing part of the code could be an opportunity for brute force attacks to the stream

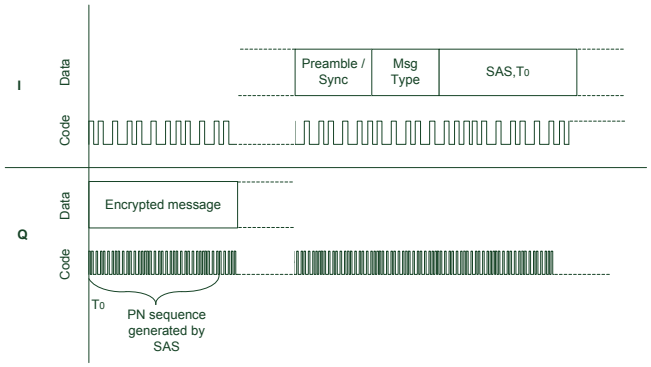



**Figure A.3.** *SAS transmission example*

cipher and crypto analysis; b) integrity of the data. An attacker can reproduce the entire SAS architecture if the data source is not authenticated;

- Timing. The SAS time reference could be set with two approaches. A predetermined recurring time slot (for example the first code phase of the first subframe) or randomized in order to increase the security (in this case the SAS message shall contain the precise epoch were to perform the correlation search). The timing should be projected in order to avoid a bit transition. Figure A.3 shows an example of SAS transmission (I arm) advanced with respect of the encrypted code (Q arm).

### User segment

The receiver shall be capable of receiving both the open GNSS signal and the encrypted signal, with the adequate bandwidth and sampling rate. After the AD conversion the signal will be (A.4):

$$\begin{aligned} s(n) = \sum_{k=1}^{N} [O_a^k(n) D^k(n) \cos(\omega_{IF} n)] + \\ \sum_{k=1}^{N} [O_b^k(n) SC^k(n) D^k(n) \sin(\omega_{IF} n)] \end{aligned} \tag{A.4}$$

Where $e(n)$ is the thermal noise introduced in the sampling process. It is assumed that Doppler frequency wipe off, code and phase locks are performed on the on the open code $O_a^k$ . The receiver will attempt to store the encrypted signal at the discrete time $[n_0 : n_0 + l]$ as defined by the protocol. After the carrier removal by multiplication with $\sin() \omega_{IF} n$ and after application of a low pass filter to cut the frequency, the remaining signal A.5 is:

$$\sum_{k=1}^{N} \left[ \tfrac{1}{2} O_b^k(n) SC^k(n) D^k(n) \right] + e(n) \tag{A.5}$$

The receiver can generate the spreading sequence by modulo 2 sum of the SAS code defined in (A.3) and the public spreading code, resulting in a short local security code replica (SCR) A.6.

$$SCR^j(n) = SAS^k(n)O_b^k(n) \tag{A.6}$$

Where *j* is the specific satellite code. A security processing function will evaluate the correlation value $C^j$ A.7 of the encrypted signal and the local replica based on a threshold for every *k* satellite.

$$C^j = \sum_{n=n_0}^{n_0+l} SCR^j(n) \left\{ \sum_{k=1}^{N} \left[ O_b^k(n)\, SC^k(n) D^k(n) \right] + e(n) \right\} \tag{A.7}$$

The security processing function will determine the signal authentication state based on a $C^j$ threshold that can be set as parameter in the receiver.

## Test results

The proposed method has been tested with a Matlab simulation in order to prove the feasibility of the authentication scheme. Furthermore the receiver operating characteristic (ROC) can be analyzed based on the probability of false positives (spoofing detected erroneously) and false negatives (spoofing not detected). The transmission block simulates a BSC(m,10) signals, where the spreading code has 10.23Mhz chipping rate and the stream cipher frequency m can be set in the software. A subset of the stream cipher is used to generate the SAS, that is stored simulating a transmission in the open signal. The spreading code and stream cipher are modulo 2 summed, obtaining the final modulation code. The software correlates the SAS with the encrypted code in the predetermined period and performs analysis on the correlation results.

The idea is that a correlation peak indicates a correspondence between the SAS and the unpredictable encrypted code, resulting in a high confidence that the signal is authentic (as mentioned before the security is based on the fact that an attacker could not generate the encrypted signal). A low correlation value means that the SAS is different from the encrypted code, indicating a possible spoofing attack. The following parameters have been considered for the simulation:

- **l**: SAS length. Different values of this parameters have been tested during the study;

- **th**: correlation threshold;
- **n**: visible satellites. The sum of different PN sequences results in multiple access interference (MAI), reducing the system performance. The value for the simulation has been set to 8;
- **m**: Binary stream cipher carrier frequency. It has been assumed a BSC(20,10) for the simulation;
- **k**: correlation windows length. This value has been set to 40 chips (about 0.004 ms);
- **SNR**: Signal to noise ratio. This value has been assumed -30dB.

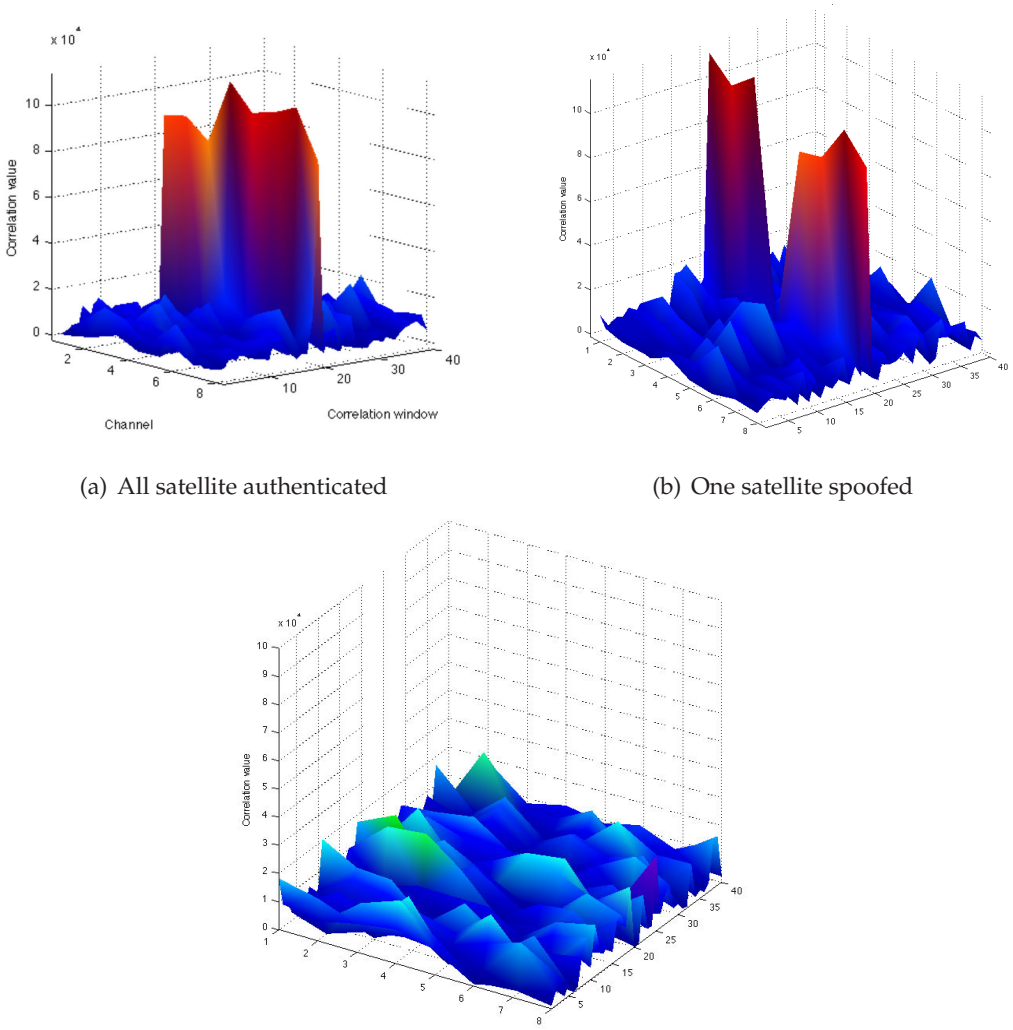
Figure A.4-(a) shows the results with a SAS length of 5000, that with a BSC(20,10) results in a local code SCR length in the order of $10^5$ . (For visualization purposes the codes has been shifted in order to align the SAS position of every SV). The picture shows a correlation peak in correspondence of the SAS code ($> 8 \times 10^4$ ), indicating a conditions where all the signals are authentic.

In Figure A.4-(b) we have spoofed SV 4. The software simulated a random PN code instead of the original encrypted sequence, simulating a single SV signal spoofing (for example performed with a receiver-spoofer) or by buffering and retransmission (with delay) of the original signal. It is noticeable that all the satellites are authenticated except SV 4 where the code replica $SCR^4$ had a low correlation with the spoofed sentence.

Figure A.4-(c) shows the case where all satellites are spoofed. The correlation at the SAS epoch is lower ($< 2 \times 10^4$ ).

Tests on a large simulation number have been performed in order to determine the false negative and false positive probabilities. The threshold is defined as the ratio between the highest correlation value and lowest one. Varying the SAS length and the threshold two plots are shown. The first one (FigureA.5-(a)) shows the false positive variation. It can be seen that varying the SAS length the false positive probability decreases. False negatives instead (Figure A.5-(b)) seems not affected by SAS length (within the plot range 500 to 5000 chips). The threshold affects differently the two plots: false positives increase when increasing the threshold, while false negatives decrease.

From the analysis it can be concluded that a good compromise for SAS length and threshold is *l=5000* and *th=2*.




(a) All satellite authenticated

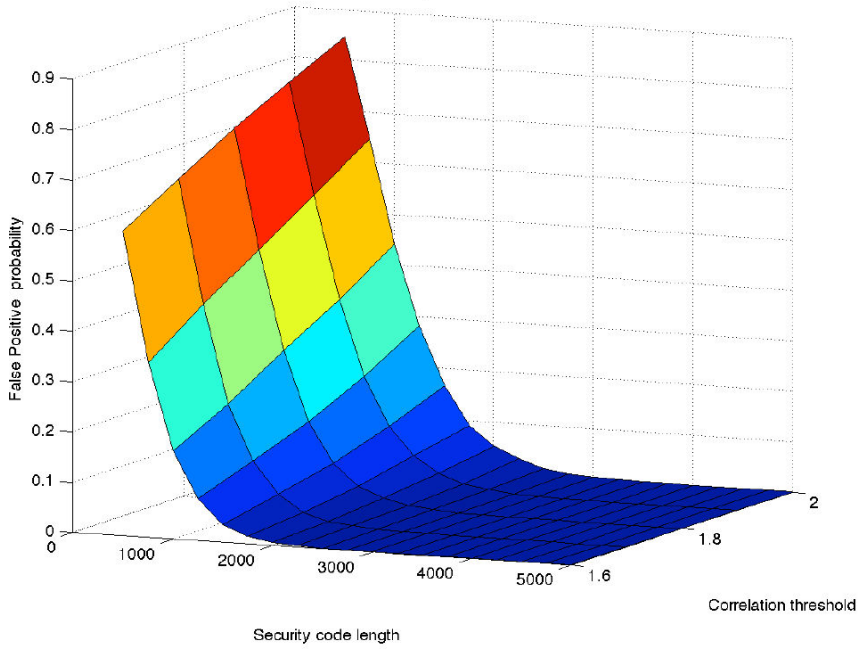
(b) One satellite spoofed

(c) All satellites spoofed

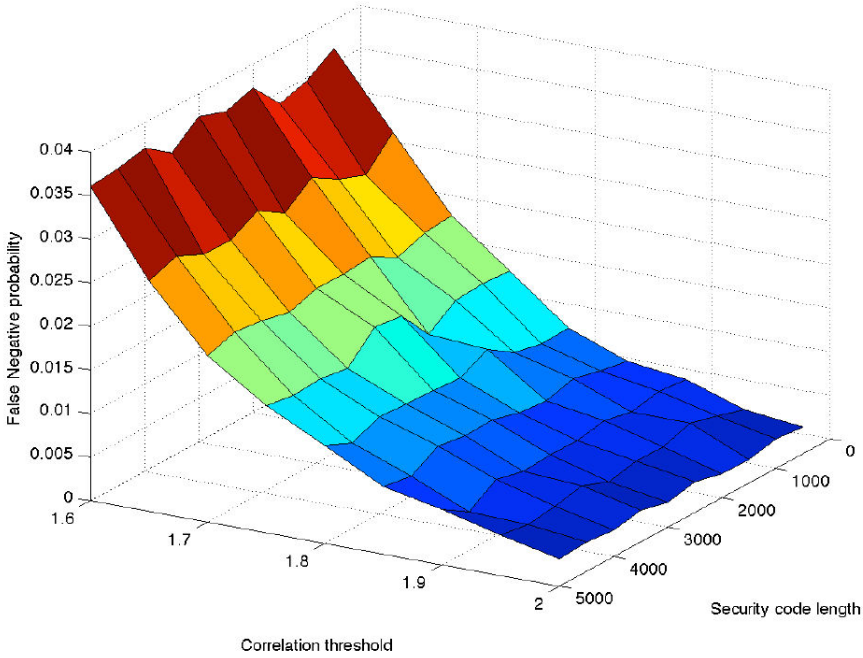
**Figure A.4.** *Satellite authentications*

With this values we obtain the following probabilities:

- False positive probability: $4 \times 10^{-5}$
- False negative probability: $3.31 \times 10^{-3}$




(a) False positive




(b) False Negative

**Figure A.5.** *False positive and negative probabilities authentication*

## Conclusions

The work presented a new concept for an authentication mechanism that could be integrated in GNSS. The proposed method requires a minimum impact in the system design, as only the data subsystem would be affected in an hypothetical update. The test results demonstrated that the signal authentication sequences concept can be used for authentication in systems that provide both open and encrypted signals, achieving a higher security level compared to navigation message authentication schemes. Further work is needed in order to define a transmission protocol and test the performance with different modulation schemes.

# List of Publications

The work presented in this thesis has appeared in the articles reported below.

### Journal papers

[J1] A. Bardella, **M. Danieletto** , E. Menegatti, A. Pretto, P. Zanuttigh, "Autonomous robot exploration in smart environments exploiting wireless sensors and visual features", *Springer-Verlag Annals of telecommunication* , vol. 67, No 7-8, pp. 297-311, August 2012

[J2] **M. Danieletto** , N. Bui, M. Zorzi, "RAZOR: A Compression and Classification Solution for the Internet of Things", *MDPI Sensor Network*, Sensors 2014, 14(1), 68-94; doi:10.3390/s140100068

[J3] **M. Danieletto**, G. Quer, R. Rao, M. Zorzi, "A Cognitive Networking Testbed on Android OS Devices", *IEEE Communications Magazine*, Submitted

### Conference papers

[C1] **M. Danieletto**, M. Mina, A. Zanella, P. Zanuttigh, E. Menegatti, "Recognition of Smart Objects by a Mobile Robot using SIFT-based Image Recognition and Wireless Communication.", in *Proc. of: European Conference on Mobile Robots (ECMR 09)*, pp. 73-79, June 2009.

[C2] E. Menegatti, **M. Danieletto**, M. Mina, A. Pretto, A. Bardella, A. Zanella, P. Zanuttigh, "Discovery, Localization and Recognition of Smart Objects by a Mobile Robot", in *Proc. of: Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR 2010)*, November 2010.

[C3] E. Menegatti, **M. Danieletto**, M. Mina, A. Pretto, S. Zanconato, A. Zanella, P. Zanuttigh, "Autonomous discovery, localization and recognition of smart objects through WSN and image features", in *Proc. of: IEEE International Workshop Towards Smart Communications and Network technologies applied on Autonomous Systems (SaCoNAS), IEEE GLOBECOM 2010*, December 2010 [Best paper award].

[C4] O. Pozzobon, L. Canzian, **M. Danieletto**, A.Dalla Chiara, "Anti-spoofing and open GNSS signal authentication with signal authentication sequences", in *Proc. of: IEEE Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing (NAVITEC), 2010 5th ESA* , December 2010.

[C5] **M. Danieletto**, N. Bui, M. Zorzi, "An Ontology-Based Framework for Autonomic Classification in the Internet of Things", in *Proc. of: IEEE Communications Workshops (ICC)*, June 2011.

[C6] **M. Danieletto**, N. Bui, M. Zorzi, "Improving Internet of Things communications through compression and classification", in *Proc. of: Tenth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2012* , March 2012.

[C7] **M. Danieletto**, G. Quer, M. Zorzi, "On the Exploitation of the Android OS for the Design of a Wireless Mesh Network Testbed", in *Proc. of: Military Communication 2013 (MILCOM 2013)*, November 2013.

# Bibliography

[1] A. Saffiotti, M. Broxvall, M. Gritti, K. LeBlanc, R. Lundh, J. Rashid, B. Seo, and Y. Cho, "The peis-ecology project: Vision and results," in *Proceedings on the IEEE/RSJ International Conference on Intelligente Robots and Systems*, Nice, France, 22–26 September 2008.

[2] E. Menegatti, M. Danieletto, M. Mina, A. Pretto, A. Bardella, A. Zanella, and P. Zanuttigh, "Discovery, localization and recognition of smart objects by a mobile robot," in *Proceedings of the 2nd International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, Darmstadt ,Germany, November 2010.

[3] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, "From today's INTRAnet of things to a future INTERnet of things: a wireless- and mobility-related view," *IEEE Wireless Communications Magazine*, vol. 17, no. 6, pp. 44–51, December 2010.

[4] B. Manoj, R. Rao, and M. Zorzi, "CogNet: a cognitive complete knowledge network system," *IEEE Wireless Communications*, vol. 15, no. 6, pp. 81–88, December 2008.

[5] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," vol. 284, no. 5, May 2001.

[6] M. Nagy, E. Motta, and M. Vargas-Vera, "Multi-Agent Ontology Mapping with Uncertainty on the Semantic Web," in *IEEE International Conference on Intelligent Computer Communication and Processing*, Cluj-Napoca, Romania, September 2007.

[7] "DublinCore." [Online]. Available: http://dublincore.org/

[8] B. Smith, "Basic concepts of formal ontologies," *in N. Guarino (Ed.) Formal Ontology in Information Systems*, vol. IOS Press, 1998.

[9] I. Niles and A. Pease, "Origins of the ieee standard upper ontology," in *In Working Notes of the IJCAI-2001 Workshop on the IEEE Standard Upper Ontology*, Seattle, WA, USA, August 2001.

[10] Stanford University, "Protege." [Online]. Available: http://protege.stanford.edu/

[11] ——, "Ontolingua." [Online]. Available: http://www.ksl.Stanford.EDU/software/ontolingua/

[12] Ontoprise, "Ontostudio." [Online]. Available: http://www.ontoprise.de/en/home/products/ontostudio/

[13] Clark and Parsia, "Pellet: OWL 2 Reasoner for Java." [Online]. Available: http://clarkparsia.com/

[14] University of Manchester, "FACT++." [Online]. Available: http://owl.man.ac.uk/factplusplus/factplusplus.html

[15] Racer Systems GmbH & Co. KG, "RacerPro." [Online]. Available: http://www.racer-systems.com/

[16] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," W3C Member Submission . [Online]. Available: http://www.w3.org/TR/Submission/SWRL/

[17] M. Compton, H. Neuhaus, K. Taylor, and K.-N. Tran, "Reasoning about sensors and compositions," in *Proceedings of the 2nd International Workshop on Semantic Sensor Networks (SSN)*, Washington, DC, USA, October 2009.

[18] M. Eid, R. Liscano, and A. E. Saddik, "A universal ontology for sensor networks data," in *Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CISMA)*, Ostuni, Italy, June 2007.

[19] N. Bressan, L. Bazzaco, N. Bui, P. Casari, L. Vangelista, and M. Zorzi, "The Deployment of a Smart Monitoring System Using Wireless Sensor and Actuator Networks,"

in *Proceedings of the IEEE International Conference on SmartGridComm*, Gaithersburg, MD, USA, October 2010.

[20] Brian McBride, "RDF Semantic," W3C Recommendation. [Online]. Available: http://www.w3.org/TR/rdf-mt/

[21] ——, "RDF Vocabulary Description Language 1.0: RDF Schema," W3C Recommendation. [Online]. Available: http://www.w3.org/TR/rdf-schema/

[22] M. K. Smith and C. Welty, "OWL Web Ontology Language," W3C Recommendation. [Online]. Available: http://www.w3.org/TR/owl-guide/

[23] University of Manchester, "The OWL API: A Java API for Working with OWL 2 Ontologies." [Online]. Available: http://owlapi.sourceforge.net/

[24] "OWL2 Web Ontology Language Document Overview," W3C Recommendation. [Online]. Available: http://www.w3.org/TR/owl2-overview/

[25] B. C. Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler, "OWL2: The next step for OWL," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 4, pp. 309–322, 2008.

[26] I. Horrocks, O. Kutz, and U. Sattler, "The Even More Irresistible $\mathcal{SROIQ}$," in *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, Manchester, UK, 2-5 June 2006.

[27] Crossbow. [Online]. Available: http://www.xbow.com

[28] Pythagoras, 570–495 BC, Greek mathematician, philosopher.

[29] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Publishers: Norwell, MA, USA, 1991.

[30] Eamonn Keogh, "Selected papers:," 2003–2012. [Online]. Available: http://www.cs.ucr.edu/~eamonn/selected_publications.htm

[31] C. Bishop, *Pattern Recognition and Machine Learning*. Springer: Berlin/Heildelberg, Germany, 2006.

[32] T. Murakami, K. Asai, and E. Yamazaki, "Vector quantiser of video signals," *Electronics Letters*, vol. 18, no. 23, pp. 1005 –1006, 1982.

[33] J. Lin, E. Keogh, S. Lonardi, and P. Patel, "Finding motifs in time series," in *Proceedings of the 2nd Workshop on Temporal Data Mining*, Edmonton, Canada, 23–26 July 2002.

[34] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms," in *Proceedings of the 9th ACM International conference on Knowledge discovery and Data Mining (SIGKDD)*, San Diego, CA, USA, 9–12 June 2003.

[35] J. Shieh and E. Keogh, "iSAX: indexing and Mining Terabyte Sized Time Series," in *Proceedings of the 14th ACM International conference on Knowledge discovery and Data Mining (SIGKDD)*, Las Vegas, NV, USA, 24–27 August 2008.

[36] E. Keogh, S. Lonardi, and C. Ratanamahatana, "Towards parameter-free data mining," in *Proceedings of the 10th ACM International conference on Knowledge Discovery and Data Mining (SIGKDD)*, Seattle, WA, USA, 22–25 August 2004.

[37] R. Cilibrasi and P. M. Vitányi, "Clustering by compression," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1523–1545, 2005.

[38] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-Network Aggregation Techniques for Wireless Sensor Networks: A Survey," *IEEE Wireless Communications Magazine*, vol. 14, pp. 70–87, April 2007.

[39] R. Masiero, G. Quer, D. Munaretto, M. Rossi, J. Widmer, and M. Zorzi, "Data Acquisition through joint Compressive Sensing and Principal Component Analysis," in *Proceeding of the IEEE International Conference on Global Telecommunications Conference (GLOBECOM)*, Honolulu, Hawaii, USA, 30–4 Nov.-Dec. 2009.

[40] R. Masiero, G. Quer, M. Rossi, and M. Zorzi, "A Bayesian Analysis of Compressive Sensing Data Recovery in Wireless Sensor Networks," in *Proceedings of the Workshop on Scalable Ad Hoc and Sensor Networks (SASN)*, Saint Petersburg, Russia, 12–14 October 2009.

[41] I. Lazaridis and S. Mehrotra, "Capturing sensor-generated Time Series with Quality Guarantees," in *Proceedings of the 19th International Conference on Data Engineering*, Long Beach, CA, USA, 5–8 March 2003.

[42] A. Castellani, M. Gheda, N. Bui, M. Rossi, and M. Zorzi, "Web Services for the Internet of Things through CoAP and EXI," in *Proceedings of the IEEE International Conference on Communications Workshops (ICC)*, Kyoto, Japan, 5–9 June 2011.

[43] V. Niennattrakul, C. Ratanamahatana, and E. Keogh, "Data editing techniques to allow the application of distance-based outlier detection to streams," in *Proceedings of the IEEE International conference on Data Mining (ICDM)*, Sydney, Australia, Dec. 2010.

[44] M. Danieletto, N. Bui, and M. Zorzi, "An ontology-based framework for autonomic classification in the internet of things," in *Proceedings of the IEEE International Conference on Communications (ICC) Workshops RWFI*, Kyoto, Japan, 5–9 June 2011.

[45] X. Ge and P. Smyth, "Deformable Markov model templates for timeseries pattern matching," in *Proceedings of the 4th ACM International conference on Knowledge Discovery and Data Mining (SIGKDD)*, Boston, MA, USA, August 2000.

[46] G. E. Batista, X. Wang, and E. J. Keogh, "A Complexity-Invariant Distance Measure for Time Series," in *Proceedings of the International conference on SIAM Data Mining (SDM)*, Phoenix, AZ, USA, 28–30 April 2011.

[47] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana, "The UCR Time Series Classification/Clustering," 2006. [Online]. Available: http://www.cs.ucr.edu/~eamonn/time_series_data

[48] A. Mueen and E. Keogh, "Online discovery and maintenance of Time Series Motifs," in *Proceedings of the 16th ACM international conference on Knowledge discovery and data mining (SIGKDD)*, Washington, DC, USA, 24–28 July 2010.

[49] B. Russell, "Logical Atomism," in *The Philosophy of Logical Atomism*, D. Pears, Ed. La Salle: Open Court, 1924, pp. 157–181.

[50] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, 2008.

[51] T. Srisooksai, K. Keamarungsi, P. Lamsrichan, and K. Araki, "Practical data compression in wireless sensor networks: A survey," *Journal of network and computer applications*, vol. 35, no. 1, pp. 37–59, 2012.

[52] J. Bello, "Measuring structural similarity in music," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2013–2025, September 2011.

[53] T. Schoellhammer, B. Greenstein, E. Osterweil, M. Wimbrow, and D. Estrin, "Lightweight temporal compression of microclimate datasets," in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, Tampa, FL, USA, 16–18 November 2004.

[54] Y.-C. Wang, *Data Compression Techniques in Wireless Sensor Networks*. Nova Science Publisher Inc.: Hauppauge, NY, USA, 2012.

[55] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Maui, HI, USA, 4–7 Januray 2000.

[56] T. Dang, N. Bulusu, and W.-C. Feng, "Robust Data Compression for Irregular Wireless Sensor Networks Using Logical Mapping," *ISRN Sensor Networks*, vol. 2013, 2013.

[57] D. Zordan, G. Quer, M. Zorzi, and M. Rossi, "Modeling and Generation of Space-Time Correlated Signals for Sensor Network Fields," in *Proceeding of the IEEE International Conference on Global Telecommunications Conference (GLOBECOM)*, Houston, TX, USA, 5–9 December 2011.

[58] Texas Instruments, "2.4 GHz IEEE 802.15.4 / ZigBee-Ready RF Transceiver (Rev. B)," 2012.

[59] ——, "Datasheet for MSP430 16-bit Ultra-Low Power MCUs," 2012, available online: http://www.ti.com/lsds/ti/microcontroller/16-bit_msp430/overview.page (accessed on 30 November 2013).

[60] J.-H. Lee and H. Hashimoto, "Intelligent spaceconcept and contents," *Advanced Robotics*, vol. 16, no. 3, pp. 265–280, 2002.

[61] J. Kim, Y. Kim, and K. Lee, "The third generation of robotics: Ubiquitous robot," in *Proceeding of the 2nd International Conference on Autonomous Robots and Agents*, Palmerston North, New Zealand, 13–15 December 2004.

[62] "Network robot forum." [Online]. Available: www.scat.or.jp/nrf/English/

[63] F. Dressler, "Self-organization in Autonomous Sensor and Actuator Networks," in *Proceedings of the 19th IEEE International Conference on Architecture of Computing Systems*, 2006.

[64] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[65] G. Zanca, F. Zorzi, A. Zanella, and M. Zorzi, "Experimental comparison of rssi-based localization algorithms for indoor wireless sensor networks," in *Proceedings of the ACM workshop on Real-world wireless sensor networks (REALWSN)*, Glasgow, Scotland, April 2008.

[66] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[67] J. Sivic and Z. A., "Video Google: A text retrieval approach to object matching in videos," in *Proceedings of the 9th International Conference on Computer Vision (ICCV)*, Nice, France, 13–16 October 2003.

[68] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 17–22 June 2006.

[69] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proceedings of the IEEE International conference on Computer Vision and Pattern Recognition*, Minneapolis, MI, USA, 18–23 June 2007.

[70] D. Meger, P.-E. Forssén, K. Lai, S. Helmer, S. McCann, T. Southey, M. Baumann, J. J. Little, and D. G. Lowe, "Curious george: An attentive semantic robot," *Robotics and Autonomous Systems*, vol. 56, no. 6, pp. 503–511, 2008.

[71] A. Pretto, E. Menegatti, and E. Pagello, "Reliable features matching for humanoid robots," in *Proceedings on the 7th IEEE-RAS International Conference on Humanoid Robots*, Pittsburg, PA, USA, 29–01 November–December 2007.

[72] E. Menegatti, A. Zanella, S. Zilli, F. Zorzi, and E. Pagello, "Range-only SLAM with a mobile robot and a Wireless Sensor Networks," in *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, 12–17 May 2009.

[73] L. Schenato and F. Fiorentin, "Average timesync: A consensus-based protocol for time synchronization in wireless sensor networks," in *Proceedings of the 1st IFAC Workshop on Estimation and Control of Networked Systems (NecSys09)*, Venice, Italy, 24–27 September 2009.

[74] P. Dutta and D. Culler, "Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications," in *Proceedings of the 6th ACM conference on Embedded network sensor systems (SenSys)*, Raleigh, NC, USA, 5–7 November 2008.

[75] A. Zanella, E. Menegatti, and L. Lazzaretto, "Self localization of wireless sensor nodes by means of autonomous mobile robots," in *Proceedings of the 19th Tyrrhenian International Workshop on Digital Communications*, Ischia, Italy, 9–12 September 2007.

[76] J. A. Costa, N. Patwari, and A. O. Hero, III, "Distributed weighted-multidimensional scaling for node localization in sensor networks," *ACM Transaction Sensor Network*, vol. 2, pp. 39–64, February 2006.

[77] E. Menegatti, M. Danieletto, M. Mina, A. Pretto, A. Bardella, S. Zanconato, P. Zanuttigh, and A. Zanella, "Autonomous discovery, localization and recognition of smart objects through wsn and image features," in *Proceedings of the IEEE International Workshop Towards SmArt COmmunications and Network technologies applied on Autonomous Systems (SaCoNAS)*, Miami, USA, December 2010.

[78] A. Goldsmith, *Wireless Communications*. Cambridge University Press: New York, NY, USA, 2005.

[79] A. Bardella, N. Bui, A. Zanella, and M. Zorzi, "An experimental study on IEEE 802.15.4 multichannel transmission to improve RSSI-based service performance," in *Proceed-*

*ings of the 4th Workshop on Real-World Wireless Sensor Networks (REALWSN )*, Colombo, Sri Lanka, 16–17 December 2010.

[80] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press: Cambridge, MA, USA, 2005.

[81] A. Pretto, E. Menegatti, M. Bennewitz, W. Burgard, and E. Pagello, "A Visual Odometry Framework Robust to Motion Blur," in *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, 12–17 May 2009, pp. 2250–2257.

[82] Hartley, R. I. and Zisserman, A., *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press: Cambridge, UK, 2004.

[83] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[84] R. W. Thomas, D. H. Friend, L. A. DaSilva, and A. B. MacKenzie, "Cognitive Networks: Adaptation and Learning to Achieve End-to End Performance Objectives," *IEEE Communication Magazine*, vol. 44, no. 12, December 2006.

[85] J. Mitola and G. Q. Maguire Jr., "Cognitive radio: making software radios more personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, 1999.

[86] "The NS–3 network simulator," Last time accessed: Sept. 2013. [Online]. Available: http://www.nsnam.org/

[87] L. Barolli, M. Ikeda, G. De Marco, A. Durresi, and F. Xhafa, "Performance analysis of OLSR and BATMAN protocols considering link quality parameter," in *Proceedings of the International Conference on Advanced Information Networking and Applications*, 2009.

[88] G. Anastasi, E. Ancillotti, M. Conti, and A. Passarella, *Experimental analysis of TCP performance in static multi-hop ad hoc networks*. Nova Science Publisher Inc.: Hauppauge, NY, USA, 2007, no. 97-114.

[89] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 121–132, August 2004.

[90] A. Jow, C. Schurgers, and D. Palmer, "CalRadio: a portable, flexible 802.11 wireless research platform," in *Proceedings of the 1st International Workshop on System Evaluation for Mobile Platforms*, San Juan, Puerto Rico, 2007.

[91] T. Newman, A. He, J. Gaeddert, B. Hilburn, T. Bose, and J. Reed, "Virginia Tech cognitive radio network testbed and open source cognitive radio framework," in *Proceedings of the 5th International Confeference Testbeds and Research Infrastructures for the Development of Networks and Communities and Workshops*, Washington, DC, USA, 6–8 April 2009.

[92] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh, "Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols," in *Proceedings of the IEEE Wireless Communications and Networking Conference*, New Orleans, LA, USA, 13–17 March 2005.

[93] N. B. Truong, Y.-J. Suh, and C. Yu, "Latency Analysis in GNU Radio/USRP-based Software Radio Platforms," in *Proceedings of the IEEE Military Communication (MILCOM)*, San Diego, CA, US, 18–20 November 2013.

[94] P. Sutton, J. Lotze, H. Lahlou, S. Fahmy, K. Nolan, B. Ozgul, T. Rondeau, J. Noguera, and L. Doyle, "IRIS: an architecture for cognitive radio networking testbeds," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 114–122, 2010.

[95] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," *ACM SIGOPS Operating Systems Review*, vol. 36, pp. 255–270, 2002.

[96] I. Broustis, J. Eriksson, S. Krishnamurthy, and M. Faloutsos, "A blueprint for a manageable and affordable wireless testbed: Design, pitfalls and lessons learned," in *Proceedings of the 3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities (TridentCom)*, Orlando, FL, USA, 21–23 May 2007.

[97] P. Urriza, E. Rebeiz, and D. Cabric, "Hardware implementation of Kuiper-based modulation level classification," in *Conference on Signals, Systems and Computers (ASILOMAR)*, 2011, pp. 919–923.

[98] "Commotion Project," Last time accessed: December 2013. [Online]. Available: https://code.commotionwireless.net/projects/commotion

[99] P. Gardner-Stephen and S. Palaniswamy, "Serval mesh software-wifi multi model management," in *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief*, Amritapuri, Kollam, Kerala, India, 18–21 December 2011.

[100] Wi-Fi Alliance, "Mobile Ad-Hoc Networking: Wi-Fi certified IBSS with Wi-Fi Protected Setup," December 2013. [Online]. Available: http://www.wi-fi.org/

[101] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized Link State Routing protocol for ad hoc networks," in *Proceedings of the IEEE International Multi Topic Conference (INMIC)*, 30 December 2001.

[102] "IPERF website," Last time accessed: December 2013. [Online]. Available: http://iperf.sourceforge.net/

[103] M. Danieletto, G. Quer, R. Rao, and M. Zorzi, "On the Exploitation of the Android OS for the Design of a Wireless Mesh Network Testbed," in *Proceedings of the IEEE Military Communication (MILCOM)*, San Diego, CA, US, 18–20 November 2013.

[104] S. Anderson, M. Coffey, D. Denning, K. Gold, P. MacDoran, M. Mathews, and F. Ziel, "Method and apparatus for authenticating the location of remote users of networked computing systems," May 26 1998, US Patent 5,757,916. [Online]. Available: http://www.google.com/patents/US5757916

[105] S. Lo, D. De Lorenzo, P. Enge, D. Akos, and P. Bradley, "Signal Authentication: A Secure Civil GNSS for Today," *2009*, pp. 30–39, September.

[106] L. Scott, "Anti-spoofing & authenticated signal architectures for civil navigation systems," in *Proceedings of the 16th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS/GNSS 2003)*, 2001, pp. 1543–1552.

[107] M. G. Kuhn, "An asymmetric security mechanism for navigation signals," in *Information Hiding*. Springer, 2005, pp. 239–252.

[108] C. Wullems, O. Pozzobon, and K. Kubik, "Signal authentication and integrity schemes for next generation global navigation satellite systems," in *Proceedings of the European Navigation Conference GNSS*, 2005.

[109] G. Hein, F. Kneissl, J.-A. Avila-Rodriguez, and S. Wallner, "Authentication GNSS: Proofs against spoofs, Part 1," in *Inside GNSS*, July/August 2007, pp. 58–63.

[110] ——, "Authentication GNSS: Proofs against spoofs, Part 1," in *Inside GNSS*, September/October 2007, pp. 71–78.

[111] "Trusted Innovative GNSS receiver (TIGER) project," Galileo Supervisory Authority grant agreement n. 228443. [Online]. Available: http://www.tiger-project.eu