Distributed Video Signal Processing for Wireless Multimedia Sensor Networks

Christian Nastasi

ReTiS Lab.

Scuola Superiore Sant'Anna

A thesis submitted for the degree of

Doctor of Philosophy

Tutor: Prof. Marco Di Natale
Supervisor: Dr. Paolo Pagano

October the 26^{th}, 2012

Contents

Introduction

FOR more than a decade, the research community has been attracted by the theoretical and practical challenges of Wireless Sensor Networks (WSNs). The motivation of the great interest towards such systems is probably to be attributed to the large number of application areas, spanning from civil to military domain. Technological advances have been the enabling factor for such research field, reducing the gap between theoretical systems and every-day life problems, posing new challenges given by the availability of "small" embedded devices deployable in large number.

Wireless Sensor Networks can be defined as distributed systems composed by *autonomous* sensors (nodes) interacting each other through wireless communication links and designed for specific *pervasive* monitoring applications. Two main characteristics are peculiar of WSNs: *autonomy* and *pervasiveness*. Each node of the network is capable of collecting information about the physical environment (sensing), performing local processing and communicating with other nodes. At the same time, WSNs are meant to be deployed in large number of nodes and with a high degree of density. Moreover, devices are usually required to be cost-effective and easily embeddable in the environment they operate on. As a consequence, a typical sensor node relies on limited resources, in terms of: energy supply and harvesting capabilities; computation power (small micro-controllers with few KB of memory are generally adopted); communication capability (short transmission range and/or low transmission bandwidth).

Recent advancements in technology have permitted to extend the original idea of WSNs, based on simple scalar sensors (e.g. temperature, light, ...), to what are nowadays called *Wireless Multimedia Sensor Networks* (WMSNs), where more complex vectorial sensors are used (e.g. audio and video). WSNs have been initially designed for simple monitoring applications where the objective is data collection, i.e. typically raw scalar values, and transmission towards a central collection point. However, algorithms and techniques developed for such networks have to be reconsidered in case of WMSNs. Vectorial raw data, such as images, cannot be exchanged in the network as they are. On the contrary, processing techniques are necessary to extract only that subset of information which is of interest for a given application context. However, computation resources are scarce, and executing complex algorithms on the sensor nodes might be unfeasible. A solution is offered by the distributed nature of the WMSNs, where collaborative approaches should exploit the (parallel) computation capabilities of the network nodes. In other words, each node must collaborate with the others, extracting and exchanging features, in order to accomplish an overall distributed task.

CHAPTER 1

Towards In-Network Processing

SINCE their initial proposal, sensor networks have been always associated to the idea of distributed systems, mainly characterized by a distributed sensing logic and a centralized processing and control logic. Only recently the word "distributed" has assumed a deeper and more appropriate meaning, referring more to collaborative nature of the overall application logic, implementing the distributed system in terms of a distributed application.

Depending on the place where most of the application processing is executed we can identify three types of systems: centralized, decentralized and distributed [**TC11b**].

(1) *Centralized* In case of centralized processing, data are acquired, possibly pre-processed locally (e.g. extraction of features) on the nodes and finally transferred to a central node that executes the actual processing. This solution is theoretically optimal, since all the information is available at a single point. The computation on each sensor node is independent on the other nodes, since no collaboration exists.

(2) *Decentralized* Another option is to use several central points of elaboration in the network as closer as possible to the events generating the data to be processed. In this way, a small subset of sensors create a cluster and the elaboration is performed in a centralized manner on single node, the cluster-head.

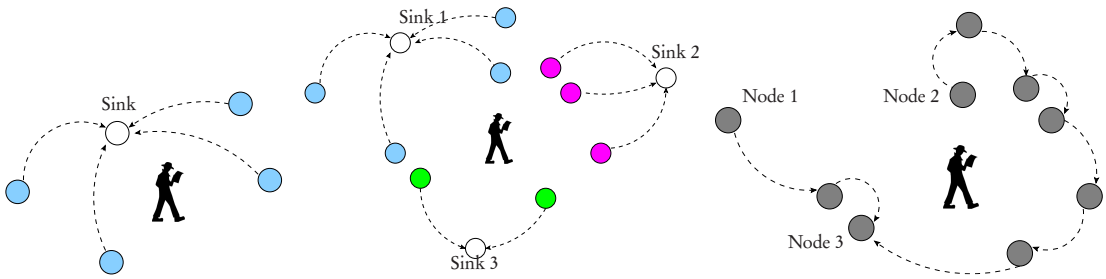
(3) *Distributed* Finally, it also possible to rethink the overall application to work in a completely distributed way, i.e. part of the processing is performed by the nodes, and information is exchanged to achieve a common goal. This case, represents the frontier of the research in WMSNs.

The expression that more precisely describes the set of methodologies and techniques necessary to truly achieve the idea of distributed application is known in literature as *in-network processing*. The methodologies presented in literature to enforce in-network processing can be classified in two main categories:

(1) *local processing*, where the single node performs some processing on local sensor data and/or incoming (received) data;

(2) *distributed processing*, where a set of nodes perform a collaborative algorithm with an overall processing logic which is explicitly shared.

Following the classification proposed by [**YMG08**], applications for WMSNs can be dived in two main categories: *monitoring* and *tracking*. The work presented in this thesis addresses the problem of in-network processing, focusing on both the local and distributed cases. Scenarios from the domains of monitoring and tracking are respectively considered for local and distributed processing.



(a) Centralized (b) Decentralized (c) Distributed

FIGURE 1. Sensor data processing approaches. *Centralized* (a): all sensor measurements (possibly pre-processed) are sent to a central processing point (Sink); *Decentralized* (b): network is partitioned in 3 clusters and for each cluster a centralized approach is used; *Distributed* (c): possible distributed approach based on aggregation chain, where Node 1 and Node 2 start the iteration and Node 3 produces final results.

1.1. Motivations and Challenges

The simplest and the most adopted approach for monitoring and tracking applications is based on a *centralized* system and a *fetch and forward* paradigm. Sensor nodes are connected to a single collection point, generally known as *sink*, as depicted in Figure1(a). Each node samples one or more environmental variables (e.g. temperature, light) and sends their content to the sink node. The actual monitoring logic resides on the sink node, which is responsible for data processing and control behaviour: alert notification, actuation, and so on. This approach merely requires nodes to be a wireless extension of sensors "wired" to a central processing unit.

Although fetch-and-forward is appreciable for its simplicity, at least from the sensor point of view, it results obsolete and unfeasible for new generation networks. With the advent of multimedia sensors and with the availability of much larger deployment units, more complexity has to be added to the sensor processing logic, moving the horizon from distributed sensing to actual distributed processing.

Communication cost is, by far, the first reason to discard the centralized fetch-and-forward approach. Given the significant amount of data produced by multimedia sensors, transmission of raw data is simply unaffordable if not feasible at all. Even considering scalar sensors with low sampling rates might yield unfeasible scenarios as the network scale increases. Techniques that use some local processing on the node side can be used to reduce the amount of sensor data.

Energy consumption is another key issue for WMSNs, and once more communication cost represents one of the most important contribution in such way. New generation of sensors, such as cameras, demands non-negligible amount of energy to operate, a problem that only recently has being investigated. See for instance the work by Casares and Velipasalar [**CV10**]. Local processing should be exploited to reduce transmissions has an impact on the power consumptions. However, code execution has its own contribution to node energy depletion and, given the complexity of multimedia raw data, such impact might not be negligible. Using technologically constrained architecture, such as low-power micro-controllers, yields good compromises in terms of power consumptions. However, feasibility of

complex signal processing algorithms might be compromised [**MRX08**]. Nonetheless, given the fast development of electronic devices is reasonable to think that the local processing and multimedia sensing contribution will be more and more marginal compared to the communication one.

Focusing all the research efforts in local (light-weight) processing is not the only way to be followed. Another point of weakness of the aforementioned centralized processing approach is given by the responsiveness to single node failures. In case the sink nodes is not reachable or not operating any more, the entire system shall fail. Recovery or replacement of the sink node might be possible. However, the network should be automatically (re-)configurable, and a non-negligible convergence time shall be required to have the system fully operating again. In case of manual configuration the setup time increases dramatically. If a distributed processing logic is adopted, the system might be designed to promptly respond to a single node failure. In principle, the application logic might be coded in a distributed way so that if one of the nodes in the network fails, the algorithm continues its normal operation without even requiring a reconfiguration phase.

1.1.1. Towards pervasive computer vision. Our interest in distributed (signal) processing for WMSNs has been drawn mostly by the sub-specific domain related to image and video signals, a domain known as *computer vision*. As a matter of fact, the domain of WMSNs has been mainly investigated by researcher from computer vision area and networked distributed systems. For this reason, several research groups have re-targeted their efforts trying to move from computer vision to networking and vice versa.

Computer Vision. Computer vision is the science and technology of "machines that see", in the sense that the machine (computer) is able to extract information from an image (vision) in order to accomplish several tasks. Operations performed by computer vision can be classified according to a taxonomy where techniques and algorithms can be organized according to the level of abstraction. This classification is known as *vision pipeline*, since each stage of processing follows a logical execution order [**GW08**].

- *Acquisition*: a digital image is created, either using classical camera sensors (visible-light sensitive cameras) or using other type of sensors (e.g. tomography devices, radar, ultrasonic, thermal, etc.).
- *Pre-processing*: a digital image is enhanced for further processing, using various techniques to emphasize some particular characteristic of the pixels (e.g. noise removal, contrast enhancement, blurring, etc.).
- *Feature Extraction*: elements of interest are identified in the image (e.g. foreground pixels, bounding boxes, histograms and any other possible descriptor).
- *Detection/Segmentation*: the elements of interest (features) become object of interests (e.g. blobs, region of interest, etc.).
- *Decision*: higher-level processing that reasons on the segmented object to undertake further decisions (e.g. classification, recognition, tracking).

For many years research has been carried out addressing all the problems of the vision pipeline. However, only with recent advancement in VLSI technology, researcher began to extend the horizon of

computer vision to embedded systems. *"Smart camera"* is the expression typically used in such community to address computer vision for embedded systems.

Evolution of Smart Cameras. With the advent of WSNs, also smart cameras have evolved in the same direction. Rinner and Wolf [**RW09**] have discussed the evolution of smart cameras towards WSNs, which ultimately is what other researchers [**AMC07**] have generalized as *Wireless Multimedia Sensor Networks*. According to Rinner and Wolf smart camera evolution can be represented in three steps.

- *Single smart cameras* (SSCs) represent the first generation of embedded computer vision. Such systems are nothing more than an embedded systems integrating one or more image sensors and a processing unit. An SSC is capable of performing all the steps of the vision pipeline, thus is meant to be as system producing high-level information (e.g. classifications, decision, actions). The challenge of SSCs are more or less the same challenges of local processing for WMSNs.
- *Distributed smart cameras* (DSCs) are the next evolutionary step, introducing the concept of distributed sensing and distributed processing to computer vision. DSCs are characterized by cooperative algorithms, were low- and/or high-level information is exchanged between the nodes to accomplish an overall strategy that might involve decision taking.
- *Pervasive smart cameras* (PSCs) are the natural extension of DSCs to the technological domain of Wireless Sensor Networks. PSCs are, by all means, another possible way to define WMSNs, representing the technological effort to move towards the scientific frontier of pervasive computer vision.

The reason to move towards PSCs is given by several advantages of pervasive vision systems.

(1) Large-scale deployments. The possibility to have deployments with a large number of nodes allows to extend the area coverage much more beyond the possibility offered by single node technology.
(2) Solve occlusion problem. A dramatic problem of line-of-sight sensors can be easily solved introducing a certain level of redundancy in the are covered by the nodes (using partially overlapping field of view).
(3) Fault tolerance. If a node of the network fails, a backup might be provided using sensors and/or processing of other nodes of the network.
(4) Sensor diversity. Limitations of a given technology (e.g. vision in darkness condition) might be overcome by employing heterogeneous sensors and complementing them (e.g. audio/video system).

1.1.2. Need for system-level QoS. Extending the research scope towards pervasive multimedia networks shall not be possible without taking into consideration the problem of communication. The intrinsic nature of multimedia applications requests the network to have a behaviour as predictable as possible, offering some guarantees when transmitting sensible information. In other words, some sort of "system-level" Quality of Service has to be enforced, considering both communication and application performances at the same time. In particular, the main issue we will focus our attention on is *timeliness*, the capability of the network to deliver information in time. This is a crucial property

when considering distributed collaborative scenarios, where a node processing depends on the arrival of information from other nodes within a certain amount of time (e.g. tracking application).

Since the wireless channel is shared, multiple transmission cannot take place in the same broadcast region, as to doing so might result in packet collision (ALHOA-like protocols) [**BDWL10**]. If a packet is corrupted due to collisions, it could be retransmitted, provided a mechanism to detect the packet was lost (acknowledgment mechanisms). Even so, the retransmission might keep going in an unpredicted way, thus resulting in an increasing of the end-to-end communication delay and bandwidth consumption. Mechanism to regulate the access to the channel are required at the Medium Access Control (MAC) layer of the network stack by implementing specific protocols. Generally, the purpose of a MAC protocol is that of maximizing the usage of the channel (typically defined as throughput). Event though, the time to wait before a packet is correctly delivered might be unbounded (contention-based access mechanism).

Due to the large scale of a WMSNs, it is extremely unlikely and unrealistic that all the nodes are in single-hop communication range (single broadcast region). Therefore, routing protocols are required to achieve multi-hop packet transmission. Although a lot of literature has been developed about routing protocols for WSNs (see [**AY05**] for a survey), the enforcement of real-time guarantees on end-to-end communication delay is still an active research field. The problem has been shown to be NP-Hard even with simplified rules for bandwidth reservation [**AGJM04**], most of the existing solutions make use of approximations based on heuristics.

Bit error rate, due to the noisy nature of the wireless channel, might result in packet corruption. There are two main techniques to solve this problem. The first is on error detection (checksums) and automatic retransmission request (ARQ). The recipient is able to detect an error in the packet and ask the source node to retransmit the packet. The second is based on error correction (redundant information), i.e. the recipient is able to correct the errors on its own; for this reason this technique is called forward error correction (FEC). Both strategies introduce overhead, increasing the unpredictability of the communication delay. We will not focus on these techniques, although studies have been carried out find optimal trade-offs between overheads and error recovering.

1.1.3. Need for tools and platforms. The understanding and the modeling of the complexity of distributed applications based on WMSNs require competences from several areas, ranging from networking to control theory, and from computer vision to data management. However, until now researchers have studied algorithms, applications and protocols for WMSNs without an holistic approach that addresses complementary and interconnected issues from all these disciplines.

An obstacle in such a way is represented by the lack of a common simulation framework where all the aspects related to such different disciplines can be modeled and analysed simultaneously. For this reason we decided to address the problem, studying the state of the art of network simulators, comprehending the approaches used in signal processing and computer vision, and finally delivering a possible solution for WMSN scenarios.

Another area which has been covered by our work is related to real world testbeds, that represent an important step to fill the gap between theory and practice. In such way, hardware and software platforms suitable for WMSNs have been defined and used for the implementation of a real monitoring

application in the context of intelligent transportation systems. The case study consists of a WMSN for parking lot and traffic flow monitoring deployed at the airport of Pisa, counting more than 15 camera-based sensors.

1.2. Local Processing

Since the introduction of WSNs, researchers started exploiting the computation capabilities of the sensor nodes to obtain "smart" sensing behaviours. While accepting the limitations introduced by a centralized architecture, the first research efforts have been focused on node local processing, involving competences from the networking and data processing research fields. With the advent of multimedia sensors, those efforts that have been renewed, attracting interest from other research fields such as multimedia signal processing and computer vision.

1.2.1. Data mining: compression, aggregation and fusion. In order to reduce the amount of data of a sensor's raw measurements, and consequently minimizing the communication cost, several techniques have been proposed. They can be classified in three main categories: *data compression*, *data aggregation* and *data fusion* [**MRX08**]. The basic idea behind them is to reduce or remove some redundant information to avoid communicating data which are not essential for the system. As a matter of fact, sensors networks are characterized by an high density of sensors, meaning that a certain environment is generally observed by more than a sensor at the same time. The result is that each network has an intrinsic level of spatial correlation between the sensors' measurements, which is exactly the inspiration and the origin of all data mining techniques [**MGTG11**].

Data compression. Data compression is a class of techniques aiming at reducing the sensor local measurement before its transmission [**YMG08**, **MRX08**, **AMC07**]. In other words, raw sensor measurements are processed on-site in order to produce a more compact representation thereof. Such processing might yield a loss of information with respect to the original data, in which case we refer to *lossy compression*. On the other hand, if it is possible to completely reconstruct the original data from the processed (and transmitted) one, we refer to *lossless compression* techniques. In both cases, compression techniques rely only on the correlation between data produced by the sensor itself, not exploiting any inter-sensor correlation whatsoever. In this sense we can classify these techniques as *pure* local processing approaches.

In the context of WMSNs, data compression is generally associated to the problem of multimedia streaming. In such way, Misra et.al. [**MRX08**] have presented an interesting classification of the application components as *coding paradigm / compression technique*, with a particular focus to the case of video/image streaming.

Single-layer compression is the most adopted in literature, where JPEG is the most representative technique. Several works have applied this technique for streaming of still images in WMSNs. Some representative examples are:

- Pekhteryev et al. [**PSOB05**], demonstrating the feasibility of JPEG and JPEG 2000 [**ISO99**] over an IEEE 802.15.4 network [**IEE06**];

- Chiasserini et al. [**CM02**], where a fixed point Discrete Cosine Transform (DCT) is adopted to reduce the computation complexity with respect to the floating point on, enabling implementation of low-powered architectures;
- Magli et al. [**MMM03**], where a change detection algorithm has been used for differential and selective coding to reduce the amount of data to be transmitted.

Multi-layer compression is generally based on the JPEG2000 [**ISO99**] technique. In particular, the wavelet decomposition that characterizes JPEG2000, is used to create multi bit-streams that can be transmitted using different paradigms. Some example of application are [**SPM96**, **YSV04**].

Data aggregation and fusion. Data aggregation exploits the redundancy of information to reduce the amount of transmitted bits. It refers to the capability of nodes to combine incoming data, possibly with local data, in order to create smaller data representations that approximate the whole data set. This is typically done using suppression functions such as min, max and average. Data fusion is used to identify data aggregation mechanisms that are based on signal processing techniques. The idea is use the combination of incoming and local data to create a more accurate information, possibly reducing the amount of bytes required for its representation.

Both techniques aggregation and fusion exploit inter-sensor correlation. Processing is still performed on a single node, however there is an implicit distributed (sensing) model. In this way, we might consider data aggregation/fusion an evolution of compression towards distributed processing. Some relevant example of data processing techniques for WSNs are:

- Yu et al. [**YGS+04**], where synopsis of sensor readings are distributed, fused and re-collected to implement a data aggregation/fusion framework applicable to any network topology; the algorithm is known to be robust with respect to single-node failure while addressing energy issues;
- Shrivastava et al. [**SBAS04**], where nodes with significant values are organized in a tree-like structure called q-digest (Quantile digest), i.e. the q-digest encodes information about the distribution of sensor values; the algorithm is known to have good trade-off between memory/power and error.

1.2.2. Signal processing: computer vision. Data mining techniques are generally meant to operate on simple scalar information that can be directly produced by sensors (i.e. raw data) or by some local processing algorithm (e.g. filtering, fusion). The latter case is more crucial when dealing with WMSNs, since images are complex raw data that potentially contain several types of information. It is therefore necessary to extract a relevant subset of those information with respect to the problem at hand (application-dependent). Such extraction process typically involves the first stages of the vision pipeline, from acquisition to feature extraction. Depending on the level of autonomy of the node, also detection/segmentation and decision-making steps might be performed locally. In the following, some of the most relevant efforts from literature are presented.

Foreground detection. Extracting foreground is fundamental to nearly any other computer vision technique [**GW08**] In case of video sequences, detection of moving foreground objects is based on a technique known as background subtraction. Basically, inter-frame correlation is exploited to distinguish pixels that are changing from one frame the other, i.e. foreground moving objects, from pixels

that do not change, i.e. background. The rational of this technique is to create a model of the back-

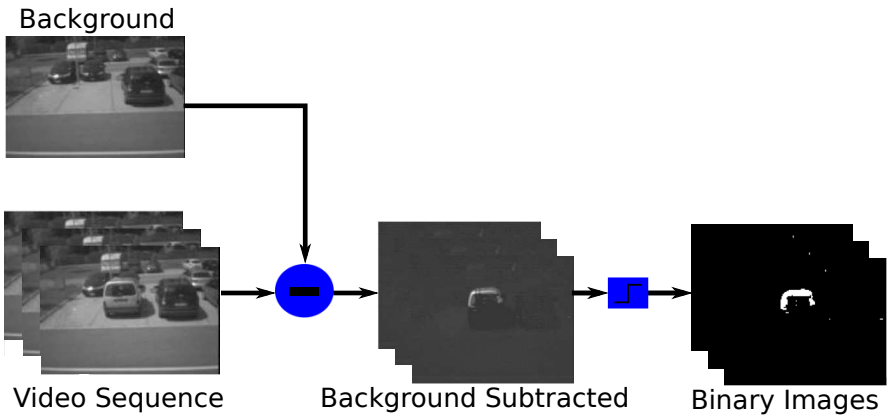


FIGURE 2. Example of background subtraction with final thresholding applied to generate a binary image.

ground image and to subtract the current frame in order to highlight pixels that have changed with respect to the background. However, in actual scenarios the background itself is subject to changes in time, although with different dynamics than the foreground objects. For this reason, actual implementations of the technique require background update mechanisms, ranging from static background to complex statistical modelling.

Due to the limitations imposed by low-computation architectures, new light-weight algorithms have been studied for efficient foreground detection. Two significant contribution in such a way are:

- Casares and Velipasalar [**CV09**], where a light-weight algorithm is to distinguish *salient* foreground motion from non- salient background motion based on the reliability of a pixel's location, and by incorporating information from neighboring pixel locations into decision making. The proposed background model reduces the amount of data stored per pixel when compared with other state-of-the-art approaches; background pixels are selectively updated with an adaptive rate, thus minimizing the number of operations.
- Tessens et al. [**TMP+09**], where an even lighter approach, based on horizontal and vertical *scan lines*, is adopted to efficiently approximate foreground pixels using background update.

In [**TMP+09**], the basic idea is to reduce the number of operations and the memory occupancy for foreground detection by limiting the processing to horizontal and vertical scan lines. The algorithm effectively allows to obtain from the scan lines the bounding box of the moving objects, a feature used in a wide range of applications. Figure 3 shows an example taken from the authors' work where scan lines are used for foreground approximation. There are, however, scenarios in which the algorithm does not operate correctly, unless a proper partitioning of the image is considered (e.g. Figure 4). The scan lines are obtained as Radon transform along the two directions. The background models the value assumed by a scan line pixel in the absence of movement. The authors show a decomposition, along the two scan lines, of a classic 2D background update technique based on combination of long- and short-term background models. Experiments show how the scan line method fairly approximate the behavior of

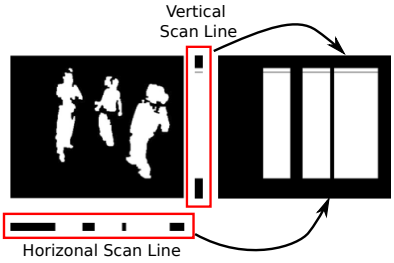


FIGURE 3. Example of the approach by Tessens et al. [**TMP+09**] based on *scan lines*; correct bounding-box foreground approximation.

FIGURE 4. Example of the approach by Tessens et al. [**TMP+09**] based on *scan lines*; incorrect bounding-box foreground approximation.

the full 2D algorithm, outperforming with respect to the static background case. The scan lines approach represents the best trade off between reduced computation power and efficient approximation. Foreground detection on full (2D) images is very expensive memory-wise. Performing the detection on a reduced version of the image alleviates the memory problem. Foreground detection on a sub-sampled image requires less arithmetic operations than the scan lines method, as soon as the sub-sampling factor exceeds two. However, to keep memory needs as low as for the scan line approximation method, the sub-sampling factor needs to be higher than 8 for images of size 230×306 and even higher for increasing image sizes.

Decision-making. One of the highest level of autonomy of sensor node is reached when local processing aims at "extracting" decisions out of the observed scene, rather than just features. In other words, almost the entire overall (distributed) logic is locally implemented on each node, leaving just general supervisory logic at the global scope. Two relevant examples of such systems are:

- Teixeira and Savvides [**TS08**], where an approach is proposed for people counting and localization in indoor scenario, based on size and movement information. The problem is simplified by two main assumptions on environment and deployment: a) people are assumed to be moving; b) cameras are placed on the ceiling (facing straight down) and fixing the ceiling height the size of the person lies in predefined ranges. A motion histogram is constructed from frame-differenced images and used to derive the discrete person locations (histogram peaks) which best fit the moving pixels. The authors proved the feasibility of their approach through experiments, showing a deployment of a simple network with non-overlapping FOV cameras where each node reports the person-count to an aggregation base station.

- Chitnis et al. [**CCM+12**], where processing on degenerated 1D images is exploited to perform an efficient counting algorithm in a distributed visual surveillance scenario.

The idea of the *line sensor* proposed in [**CCM+12**] is to detect moving object crossing a particular horizontal (or vertical) line in the image plane (e.g. see Figure 5). Image acquisition and processing along a

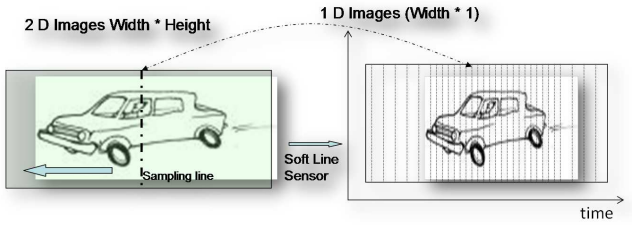


FIGURE 5. The idea of the *line sensor* by Chitnis et al. [**CCM+12**]; Vertical line sampling over time allows reconstruction of moving objects.

single line result much more efficient than the 2D case. With this approach, state-of-the-art algorithm can be applied almost directly to the 1D image, without requiring further approximation. For instance, the authors show how the line sensor can be combined with background subtraction techniques to extract the foreground objects. Figure 6 shows a sequence over time of vertical background lines. The

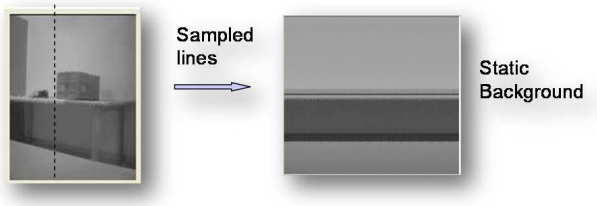


FIGURE 6. The idea of the *line sensor* by Chitnis et al. [**CCM+12**]; vertically sampled background lines.

authors show the effectiveness of their approach in a traffic monitoring scenario presenting two type of applications. In the first case they focus on traffic flow measurement, where basically a single line sensor node is used to count the number of vehicles passing in a given road section. In this case the application executes locally on the node. In the second case a distributed level of processing is used to make a speed measurement of the vehicles passing by. In this case, two line sensors are coordinated by a base station to accomplish the task.

1.3. Distributed Processing

Although the improvement of local processing capabilities is a key enabling factor for WMSNs, the frontier of the research is represented by the actual distributed processing. In the following, some of the main challenges towards the definition of distributed computer vision algorithms are identified.

1.3.1. Camera Relationship Discovery. One of the main prerogative of WMSNs is self configuration. In case of camera networks, the main aspect to be addressed is the discovery of relationships between cameras. While for non-line-of-sight sensors such issue is generally limited to a localization problem, assuming that sensor are related by proximity, in case of cameras the relationship has to be derived from the sensor measurements. In particular, for line-of-sight sensors, such as cameras, the problem is to identify the subset of cameras with overlapping Field Of View (FOV) and, possibly, to identify the overlapping regions in the image plane of the camera.

FOV lines. The concept of FOV lines has been introduced first by [**KS03**] and further extended by [**VSC+08**]. The idea is that camera extrinsic parameters, necessary to map pixels in the image plane into points in the observed scene, are supposed to be known. In this way, an object detected in the image can be associated to a position in the real word coordinates. The authors propose a mechanism to automatically detect the boundaries of a camera's FOV into the image plane of another camera. Assuming that objects move one a ground plane in the scene, this is equivalent to determining a line on such ground plane.

Localization and Calibration. In some cases, the camera extrinsic parameter might be unknown, i.e. calibration is necessary. In [**BsLS06**] the authors propose a solution to localize cameras and reconstruct their parameters. The idea is to use a LED for the setup phase of the camera network. Each camera is associated with a unique ID. When the system is going to be configured, each node produces a modulated signal with its LED that encodes the node ID. All the cameras that can observe this signals can reconstruct the position of the transmitting node in their image planes. Combining observation of different nodes, at least three, the position can be determined also in the real word coordinate (through triangulation). In this way, the calibration of the camera parameters might be performed.

1.3.2. Clustering. *Cluster* is a term that can assume different meanings depending on the research fields it is associated. In networking, it can be used to identify a subset of node that communicate in a specific region of the network, typically nodes in a single-hop range. For instance this is the case of a cluster-tree topology where the leaves of the tree are groups of nodes connected to the rest of the network by a gateway node (generally called cluster-head); the gateway nodes are organized in a tree (hierarchical topology). In this context we can say that "cluster" has a physical meaning.

The word cluster can be also used to specify logic identities. In particular the nodes can be grouped together on the basis of some relationship among themselves, i.e. sensors with dependent measurements such as cameras with overlapping/non-overlapping FOV or proximity sensor that are in the same region of the target. It is worth noticing that logical and physical cluster could not match each other. For instance two cameras looking at the same target could be far enough to use a multi-hop link to communicate.

Clustering is a technique used to cope with growth in the number of nodes [**GGGA07**, **SSC09**]. The idea is to limit the heavier part of the communication within the boundaries of a sub-set of nodes, the cluster. In order to have an effective solution, nodes are typically organized in single-hop clusters. Cluster might have a leader responsible for the intra-cluster processing and/or communication. Cluster might interact with some boundary nodes to collaborate at the level of the whole network. In camera networks it is desirable to group the nodes on the basis of the relationship between their FOVs. But

this has an important drawback. Differently than classic WSNs, camera nodes with overlapping FOV might not be in single-hop communication range [**MP09**], leading again to the problem of scalability.

Static overlapping FOV based clusters. A solution proposed in [**GGGA07**] is to statically create clusters of cameras with overlapping FOVs, i.e. any two cameras not in the same cluster shall not have overlapping FOVs. The condition is realistic for indoor surveillance scenarios (e.g. hallways, rooms).

Dynamic FOV based (single-hop) clusters. In [**MP09**] the authors solve the problem of multi-hop overlapping FOV based clusters by defining a dynamic algorithm that creates multiple single-hop cluster based on overlapping FOV. The proposed approach takes into account cluster formation (with cluster-head election), cluster propagation (i.e. dynamic management of membership and new cluster-head selection), clusters coalescence and non-coalescing inter-collaboration.

Hierarchical organization. In [**SHR05**] the authors proposed a hierarchical network organization. The idea is to dynamically create groups of nodes tracking the same target. A second layer of hierarchy is created within each group, in particular the nodes of the same group are organized in cliques of independent measurements, i.e. all the nodes of a cliques have correlated measurements. Within the clique a broadcast protocol is proposed to send the raw measurements to a master node which should execute the tracking algorithm. The inter-cliques communication is based on a peer-to-peer protocol with some ARQ technique; the tracking algorithm is basically designed at clique level rather than node level, i.e. the algorithm interactions take place only between clique master nodes.

1.3.3. Camera Selection. Sensor scheduling is a technique used for two main reason: save energy and obtain the best application performance. It is easy to understand how the usage of idle period has an impact in the reduction of the energy consumption of a sensor, but the sensor scheduling can be also useful in terms of application performance. A sensor could be selected for measurement because the information it could provide are more important for the processing algorithm with respect to other sensors. From the other hand, sensor that do not have interesting information to report might be left in an idle state.

In case of cameras, the sensor scheduling problem is known as *camera selection*. Given a set of cameras, with overlapping FOV, observing the same target, the *camera selection* problem is that of picking out the best sub-set of cameras to carry out the task at hand, e.g. tracking a moving target.

An approach proposed in [**GGGA07**], relies on the fact that the set of selectable nodes is known a priori. The set of node is a cluster. They propose to select one cluster-head at a time on the basis of some generic features extracted by the cluster-members. All the features are sent towards the cluster-head which is in charge of selecting the next cluster-head. The actual processing is then performed locally only by the cluster head.

The problem of camera selection can be modeled using the game theory as a multi-player and negotiation problem [**SSC09**]. The global utility specify the capability to track all the targets in the surveillance area. The maximization of the global utility is indirectly obtained by maximizing the local (of each node) utility functions.

1.3.4. Multi-Camera Tracking. Working with multiple view of the same object allows to improve the performance of algorithms designed for single view. The redundancy of information can be used to solve occlusion problems and to combine multiple images to obtain more accurate results. However,

some problems have to be considered to fully exploit the potentiality of multi-view processing. Among others, target labelling and hand-off are recalled here to give an idea of such issues.

Target Consistent Labelling. Given a set of cameras with overlapping FOVs and assuming that more than one target can be seen by those camera, the problem of target consistent labelling is that of keeping consistency in the target identity across the cameras.

One of the most common approach is based on the fact that the cameras can measure the position of target on a reference ground plane, i.e. common reference coordinate system. It is therefore possible to simply assess the identity of the target seen by two different cameras by matching, up to a tolerance range, the positions on the ground plane [**KS03**, **VSC+08**].

If color based tracking system are used, e.g. CamShift, a color model of the target is usually required. This model might be exchanged between two cameras to verify the identity of the detected targets. This approach has been used in [**QKR+07**]

Target Hand-Off. It is a process that carried out when a target is leaving a camera's FOV and is entering another camera's, with the purpose of maintaining the consistency on the identity of the target to possibly reuse the past tracking information.

A possible solution is to adopt a proactive scheme. The destination nodes are informed about the new target entering their FOV by the source nodes. This can be done using an explicit addressing approach as in [**GGGA07**], but can be done also with implicit addressing (e.g. broadcast, over-hearing) as in [**SSC09**].

Another possibility is to use a reactive scheme. The destination nodes request information to the source nodes when a new target appear in their FOV. Even in this case the communication can be with explicit [**WVC10**] or implicit addressing [**MP09**].

CHAPTER 2

Application-Aware Quality Of Service

THE expression Quality of Service (QoS) is historically used in the computer networking field to identify algorithms and techniques for resource reservation control. Supporting different user classes is a typical prerogative of QoS, where users (or applications) are allowed to transmit information with different priorities. Another prerogative of QoS-based system is to provide a certain level of performance to data flows: minimum bit-rate, maximum jitter, minimum end-to-end delay, and so on.

However, some considerations should be made when dealing with WMSNs. One of the most significant difference between such networks and traditional computer networks is the *purpose*. WMSNs are meant to be pervasive networks tightly coupled with the environment in which they operate, requiring a design of each component of the overall system: communication, on-board (local) logic and distributed logic. On the contrary, traditional networks are meant to be generic, essentially enabling communication for any type of application built on top of them. In other words, if traditional networks can be intended as *general purpose* networks, WMSNs are rather *specific purpose* ones.

Motivated by this observation, a reconsideration of the concept of Quality of Service for WMSNs is necessary. In the same way we are used to think of WMSNs as the whole distributed systems, a more appropriate definition should consider *quality* as a matter of the overall distributed application, therefore defining the new domain of *Application-Aware* QoS.

2.1. Background

The problem of QoS can be addressed at different layers of the ISO/OSI communication stack. The survey on streaming for WMSNs by [**MRX08**] discusses some of the most recent techniques to enforce QoS: at network (NET) layer, at the Medium Access Control (MAC) layer and through cross-layer optimizations.

2.1.1. Bandwidth Allocation. Bandwidth management represents one of the main approaches to enforce QoS in terms of performance guaranteeing. In our work we focused on the bandwidth management technique which is commonly known as dynamic bandwidth allocation (or bandwidth reservation). This problem can be simply formulated for WMSNs as that of a set of nodes competing to use a shared resource (bandwidth) where a scheduling policy (allocation) needs to be accomodated to satisfy time requirements associated to the information (messages) each node wants to transmit [**CPLL09**]. In other words, the network protocol is responsible to provide *timeliness* to end-to-end transmission communication, trying to guarantee a maximum end-to-end delay that a message can experience. Because of the similarities with the operating-system problem of task scheduling, the theory of (soft) real-time systems perfectly applies to the aforementioned scenario.

In wireless networks, uncontrolled concurrent transmissions result in collisions, packet losses, and unbounded delays. Eventually, the performance of time-sensitive applications, e.g. industrial automation, process control, distributed signal processing, will dramatically decrease. Therefore, many MAC and network protocols for WSNs have been proposed to disciplined access to the medium.

The *de-facto* standard for WSNs, the IEEE 802.15.4 [**IEE06**], provides as well a mechanism called Guaranteed Time Slot (GTS) to enforce bandwidth management. In particular, a TDMA-based access can be exploited to implement on top of the standard possible bandwidth allocation policies.

Contribution. The work presented in this Chapter faces the problem of application-aware bandwidth allocation for IEEE 802.15.4 -based networks We start considering the simple case of the star topology as defined by the beacon-enabled mode of the standard, designing a bandwidth allocation algorithm at the MAC layer. Thereafter we move to a higher level of abstraction, considering mechanism for bandwidth allocation at the network layer. Following the IEEE 802.15.4 standard specifications, we study the allocation problem in case of cluster-tree topologies.

Bandwith Allocation at Medium Access Control Layer. In WSNs, the first problem to solve is the access policy to the shared medium. Solution based to Frequency Division Multiple Access (FDMA) or Code Division Multiple Access (CDMA) are difficult to be implemented for resource-constrained node such as those typically used in WSNs. Most of the real-world implementations of MAC layers for WSNs refers to the family of Carrier Sense Multiple Access (CSMA), Time Division Multiple Access (TDMA) or a hybrid CSMA/TDMA solutions. See [**BDWL10**] for a survey on the recent development in MAC protocols for WSNs.

One of the approach used in literature for bandwidth allocation in WSNs is based on prioritized CSMA. The idea is that, given a parametric CSMA algorithm, its parameters can be set according to the priority (importance) of the packets. The problem of this solutions is that they work properly under low traffic assumptions and cannot provide guarantees in terms of minimum end-to-end delay.

Using TDMA to enforce bandwidth allocation is probably the easiest approach for single-hop communication. The most common implementations are based on centralized time-slot scheduling where the network is, generally, organized as star [**ARSP06**, **KAT06**, **HPH08**, **NYM08**, **NMS+10**]. In this case a central node, typically called coordinator or cluster head, acts as scheduler of the access to the channel (assigning time-slot to the nodes) and as clock master to keep the node synchronized. The other nodes can request bandwidth to the coordinator, specifying different prioriy indexes that can be used to accomodate an optimal allocations.

Another approach to TDMA is based on distributed scheduling algorithms [**CHTC07**]. In this case there is no central node in charge of scheduling the bandwidth, but the network is created in a way that each node run the same scheduling algorithm that produce the same results. This can be considered as a consensus algorithm to obtain the same schedule on all the nodes.

Bandwith Allocation at Network Layer. Although a lot of literature has been produced about routing protocols for WSNs (see [**AY05**] for a survey), the enforcement of real-time guarantees on end-to-end delay is still an active research field.

Many solutions for multi-hop communication are based on contention access mechanism. The problem is that the utilization obtained with this kind of solution are very low. It has been shown by Li et.al. [**LBD+01**] that in case of a line and a grid of nodes the end-to-end throughput is respectively

1/8 and 1/24 of the link throughput. Moreover, it is not possible to provide guarantees on the end-to-end delay.

The problem has been shown to be NP-Hard even with simplified rules for bandwidth reservation [**AGJM04**]. Most of the existing solutions make use of approximations based on heuristics [**TXC05**, **SCCC06**, **ZC02**, **Row08**].

2.1.2. The IEEE 802.15.4 standard. Although almost ten years of research has been carried out in the field of WSNs, with several proposals for MAC and routing protocols, the development of standard protocols has been relatively slow. Up to date, the IEEE 802.15.4, whose rationals are closer to the original concept of WSNs, is the *de-facto* standard in the field. It has originally been presented in 2003 and further reviewed in 2006 [**IEE06**]. A lot of literature, particularly related to real world application, has been developed in the last years around the IEEE 802.15.4 standard. Almost all the hardware equipments (motes) currently used in actual deployments of WSNs use the IEEE 802.15.4 specification.

The standard is the official specification of the IEEE Society for the Low-Rate Wireless Personal Area Network (LR-WPAN). The specification is defined up to the MAC sub-layer of the second layer of the ISO/OSI communication stack. Therefore, only the Physical Layer, which specifies operating frequencies and digital modulations, and the Medium Access Control are covered by the standard. Other components, such as Logical link control (LLC) and multi-hop communications (routing protocols) are out of its scope. The most popular extension of the standard is *ZigBee*, an alliance of industrial companies that specify the upper layer of a protocol stack based on the IEEE 802.15.4.

The MAC sub-layer of the IEEE 802.15.4 supports two operational modes:

(1) the non beacon-enabled (or *beacon-less*) mode, in which the MAC is simply ruled by a non-slotted Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA);
(2) the *beacon-enabled* mode, where both a slotted CSMA-CA and a (TDMA) policies can be applied according the specification of a special control message called *beacon*.

The first mode allows the creation of meshed topologies, where a peer-to-peer relationship is established among the nodes of the network. Each node can directly communicate with any other node in its radio range, using the contention mechanism specified by the unslotted CSMA-CA. This operation mode is the one adopted by the specification of the ZigBee communication stack.

More interesting for time-constrained application is the second operational mode where nodes are configured in a star topology. The central node is called PAN *Coordinator* and is responsible for: controlling the association of other nodes to the star; synchronizing the associated nodes; coordinating the transmission through the beacon message. The nodes associated to a Coordinator are called *end devices*. Any communication in beacon-enabled takes place according to a global time-slot specification called *superframe*, depicted in Figure 1.

The superframe structure is defined by the content of beacon message. This message is sent by the Coordinator at the beginning of the superframe, while all the other nodes are expected to be in receiving mode. The Superframe Duration (SD) is calculated from the Superframe Order (SO) parameter which is included in the beacon. The next beacon message, which shall start a new superframe, is expected to be sent after Beacon Interval (BI) from the last received beacon. The BI is calculated from the Beacon Order (BO) parameter which is included in the beacon message.

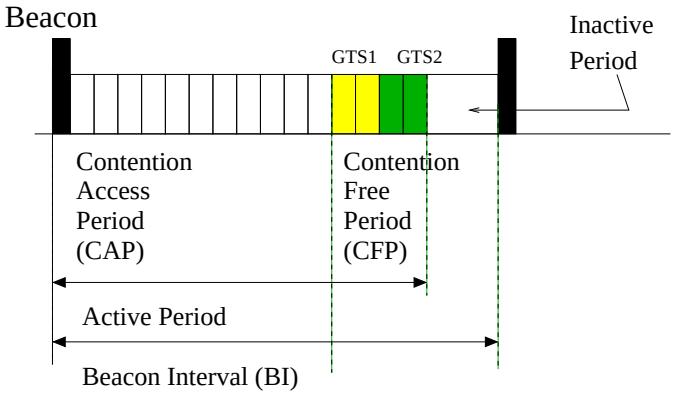


FIGURE 1. Superframe Structure in the IEEE 802.15.4 standard.

The duration of the superframe is further divided in 16 equally-sized time-slots, irrespective of the SD value, i.e. changing the SD changes the duration of the single time slot. At this point the standard allows two adopt two possible medium access policies:

- the *Contention Access Period* (CAP) represent the first part of the superframe time-slots, it is ruled by a slotted CSMA-CA and its presence in the superframe is mandatory (at least 7 time slots);
- the *Contention Free Period* (CFP) is an optional set of time-slots that follows immediately after the CAP and is ruled by a TDMA; the time-slots in the CFP are also called *Guaranteed Time Slots* (GTSs).

While the CAP is generally used for best-effort communication, the CFP allows to realize more effective communication for time-sensitive applications. The CSMA-CA used in the CAP loses performance as the number of nodes increases, since more contentions are likely to happen. Moreover, the random components of the back-off algorithm of the CSMA-CA does not allow to define a maximum end-to-end delay that a packet might experience. On the contrary, GTSs are more predictable, permitting, in principle, time-bounded transactions for time-sensitive applications. Although the nature of the wireless channel introduces several sources of indetermination so that is not possible to provide absolute guarantees for any transmission, the TDMA still represents a valid solution for the class of applications requiring soft real-time constraints. All in all, the GTS mechanism can be seen as an actual tool provided to higher layers to implement bandwidth management.

However some considerations have to be made regarding the usage of the GTSs. The IEEE 802.15.4 standard specifies a static mechanism for the allocation policy of the GTSs in the CFP. When a (associated) node wants to transmit (or receive) data with real-time constraints, it requests the coordinator to be assigned a GTSs in the next superframe(s). The node requests are processed by the Coordinator which is in charge of allocating them to each node. The allocation is then broadcasted in the next beacon message(s). According to the standard, a GTS can be deallocated either when the device node formulates an explicit request, or when the device does not make use of it for some time. A maximum

of 7 descriptors for the GTSs is allowed. All in all, the policy for GTS allocation proposed by the standard result a bit inefficient in terms of responsiveness and flexibility.

2.2. Related works on IEEE 802.15.4

To better exploit the real-time features of the IEEE 802.15.4 standard, several bandwidth allocation protocols have been proposed in literature. Some of the most relevant examples are presented in this section.

Koubâa et al. [**KATC08**] have been first authors proposing an algorithm for the IEEE 802.15.4 standard that would reuse its GTS mechanism to implement a dynamic bandwidth allocation policy. i-GAME enforces timeliness to the extent of the end-to-end delay also referred to as message *deadline*. An extension of the standard GTS request message is necessary to encapsulate the deadline and other auxiliary information. A working simulation model exists, allowing for early evaluation of generated case studies.

The basic idea is to improve on bandwidth utilization by sharing the same GTS among multiple data flows instead exclusively allocating it to a given node. The GTS is "shared" across different superframes, in the sense that the same time slot is allocable to different nodes for each subsequent superframe. The Coordinator has to arrange a schedule that satisfies the deadline requirements of each node, for instance following a Round Robin policy. The schedule can be updated by the Coordinator upon arrival of new bandwidth request, each specifying the rate and the deadline of the related data flow. The bandwidth is assigned preserving the time requirements of the already existing flows. In other words, the bandwidth requests coming from the nodes are served following a First Come First Served policy.

- i-GAME performs reasonably well for time-driven systems, where sensor nodes are expected to periodically send information. This is the typical scenario of classic monitoring applications, where simple scalar variable (e.g. temperature, luminosity) are periodically sampled and transmitted to a sink node. New generation of WSNs are moving towards an event-driven paradigm, where sensor nodes are capable to acquire more complex variables (e.g. images, sound), to take local decision about the importance of the acquired data and to select whether and what time of information has to be transmitted. All in all, i-GAME is more suitable for network with a periodic traffic model, thus implicitly assuming a time-driven system. Its application to aperiodic or sporadic traffic models is not efficient because of the acceptance policy (based on FCFS).
- Low rate flows are eligible to remain unserved if activated (requested) after high rate ones. Suppose 7 high rate flows requests arrive at the PAN Coordinator. i-GAME allocates 7 GTS to those 7 nodes letting eventual low rate requests be pending forever.
- Data flows are indistinguishable for what concerns the packet information content, being the rate and the deadline the only parameters playing a role in the admission policy. In particular, it is not possible to distinguish between essential packets and packets that could be dropped in case of congestion.

The authors of GSA [**NYM08**] have proposed an allocation policy to minimize the total number of unallocated GTSs based on the Earliest Deadline First (EDF) scheduling algorithm [**LL73**]. An

access control module runs on the coordinator and the algorithm accepts and possibly schedules a set of transactions (data flows) according to their deadlines. GSA is able to handle bursty, periodic and aperiodic traffic and is shown to perform better than FCFS-based GTS allocation strategies. GSA tries to smooth out the traffic by distributing the GTSs of a transaction over as many beacon intervals as possible while satisfying its time constraints. The algorithm is proven to be optimal, in the sense that if a set of transactions is schedulable under EDF, it is also schedulable under GSA.

The proposed algorithm introduces significant improvement with respect to the iGAME proposal.

- The usage of an EDF-equivalent scheduling algorithm allows to achieve the best bandwidth utilization. Although the formulation of iGAME is general, it has only being implemented in case of a Round Robin policy, which has a sub-optimal bandwidth utilization.
- GSA supports aperiodic traffic model, enabling the reactive paradigm of event-driven WSN applications.

However, the lack of application-awareness is still problem common to both the GSA and iGAME proposals. Data flows are once more indistinguishable each other, since the GSA admission control is only based on deadline specifications.

An other solution proposed in literature to perform dynamic bandwidth allocation on top of the IEEE 802.15.4 is the Adaptive GTS Allocation (AGA) [**HPH08**]. The algorithm has been designed to solve the starvation problem providing fairness and low latency to flows. The Coordinator computes the GTS schedule for the new beacon interval depending on the bandwidth requests and traffic priority specified by end devices in the previous beacon interval.

AGA is not focused on real-time communication, since it acts on ordinary network metrics such as fairness and average latency. Data flow differentiation is out of the AGA proposal scope too.

2.3. Application Scenario

As application-driven networks, WSNs may require high data reliability to maintain detection and response capabilities. Sensor nodes are typically affected by high failure rates so that a certain level of redundancy is required to the network to enforce reliability in the measurements. Moreover, redundant observations (or observing sensor nodes) might improve the statistical confidence in the measurement, as detailed by Pagano et al. in [**PPL09**]. If on one hand sensor nodes redundancy enforces reliability [**WLR+07**], on the other hand it increases the probability of network congestion. In an overloaded network, important services cannot be guaranteed anymore unless proper scheduling strategies are applied to differentiate upon the data flows on the basis of their content. If this problem might be negligible for some first generation simple-scalar networks, it becomes dramatically important for the new generation of WMSNs, where higher volumes of data have to be managed.

In this work we consider a general application scenario as depicted in Figure 2. A WSN is used for monitoring a set of *observable variables* $\{x_1, \ldots, x_K\}$, which can reside in a range that goes from simple scalar data (e.g. temperature), to images in a video scene. To cope with the unreliability of typical low-cost sensor nodes, each variable can be monitored by more than one node. In the figure, four identical nodes are observing variable x_1, three identical nodes monitor variable x_2, and so on.

We consider *star* topologies, where N End Devices (EDs) are directly connected to the Coordinator via single hop routes. An ED can send data messages to the Coordinator upon event detection. A

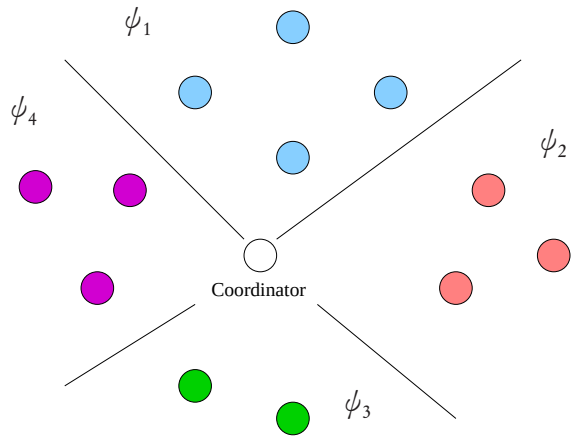


FIGURE 2. Architecture of a WSN monitoring four observables.

message consists of one or more MAC data frames, and is associated with a time constraint (deadline). We denote by *flow* a sequence of messages of the same type.

The network bandwidth is the resource shared among the nodes competing to send data, much like the CPU is shared among tasks. If we logically associate flows to tasks and messages to jobs, our bandwidth allocation strategy can be analytically studied making use of the real-time theory developed for scheduling systems. We adopt the real-time jargon and make use of the symbols defined in Table 1.

τ_i	the i-th data flow
$\Gamma = \{\tau_i\}$	the set of all the flows in the system
$\overline{C}_i$	message size, in bytes.
$J_{i,j}$	The j-th instance (message or job) of the i-th flow.
$\overline{T}_i$	period or minimum inter-arrival time between two messages of the i-th flow.
$\overline{D}_i$	the flow relative deadline.
$\overline{a}_{i,j}$	message activation time.
$\overline{d}_{i,j} = \overline{a}_{i,j} + \overline{D}_i$	message absolute deadline.
$\mathrm{J}(t) = \{J_{i,j}\}$	The set of all the active messages in the network at the time t.

TABLE 1. List of Symbols

Some information must be collected by the star Coordinator in order to accommodate a feasible schedule of the flows. In particular, each data flow τ_i, periodic or sporadic, is described by:

- the worst-case size (in bytes) $\overline{C}_i$ of its messages;
- the relative deadline $\overline{D}_i$, which represents the maximum transmission delay that a message can experience before loosing its validity;
- the period (or the minimum inter-arrival time) $\overline{T}_i$ between two consecutive messages.

Let $\Gamma = \{\tau_i\}$ be the set of all flows in the system, and let $J_{i,j}$ denote the j-th instance (or *job*) of a message of the i-th flow.

To reconstruct an event it is necessary to aggregate the readings of the nodes looking at the same event from complementary perspectives. We call these perspectives *observables*. We classify the information needed to reconstruct an event in the independent set

$$\Upsilon = \{x_1, x_2, \ldots, x_K\},$$

with K being the number of observables so that $K \leq N$. An observable measure is encoded in a message and transmitted by one or more flows. Therefore, we partition the flow set according to Υ, as

$$\Psi = \{\psi_1, \psi_2, \ldots, \psi_K\},$$

where $\psi_k = \{\tau_i^k\}$ is the set of the flows that carry data for the observable k. We call "k" the type of the flow. Note that, from the formal properties of the partition, $\Psi = \Gamma = \{\tau_i\}$ is the set of all the flows in the system. Let w_k be the cardinality of subset ψ_k, i.e. the number of flows that carry information on x_k, with $N = \sum_1^K w_j$. All flows $\tau_i \in \psi_k$ are *equivalent*, in the sense that they carry semantically equivalent information (although the actual values of the measurements can be different). Thus, we assume that all flows belonging to the same subset ψ_k have the same parameters $(\overline{T}^k, \overline{D}^k, \overline{C}^k)$.

Note that failure rate (generally high in WSN technology) is not the only reason to generate w_k independent readings of the same variable. The estimator of an experimental observable x_k is a random variable affected by statistical uncertainty σ_k. Providing a set of w_k independent measurements for x_k permits to estimate it through the arithmetic mean $\overline{x_k}$ with an error of $\sigma_k/\sqrt{w_k}$. Of course, increasing the level of redundancy imposes to filter the transmission requests introducing an admission control based on the packet content as it is discussed in the next sections.

2.4. Application-Aware Bandwidth Allocation (MAC layer)

We want to address the problem of dynamic bandwidth allocation in the aforementioned scenario, proposing the Bandwidth Allocation for Content based and Context Aware Real-time ApplicaTion (*BACCARAT*) protocol [**NMS**$^+$10]. The algorithm we proposed provides application-aware QoS, enforcing *timeliness* while preserving the performance of the overall monitoring application. A general mechanism for application-aware data flow differentiation is proposed together with a possible implementation on top of the IEEE 802.15.4 standard. The proposed approach approach is suited to follow different strategies customized to specific event signatures and arrival rates. But more importantly, our approach is explicitly designed to address the network congestion due to the redundancy of information in the system. The basic idea of BACCARAT is to reserve a certain amount of bandwidth for (off-line) guaranteed traffic. Mandatory information are meant to be scheduled as part of this traffic class. The remaining part of the bandwidth is used to enhance the performance of the system by purposely selecting data flows to be transmitted and discarded.

The main problem in WSN is that the total available bandwidth may not be enough to support transmission of all messages within a given period. However, we could dynamically reduce the redundancy level for some observables in a controlled manner, while guaranteeing a certain confidential level on the measurement.

For every flow partition ψ_k, we define w_k^G as the minimum number of flows of type k (k-flows) required by the Coordinator in order to guarantee the confidence level of the measurement. Let $\psi_k^G \subseteq$

ψ_k be any subset of w_k^G elements of ψ_k. Since all flows belonging to ψ_k have the same parameters, all possible subsets of ψ_k with the same cardinality are equivalent, we can choose any one of them during the off-line schedulability analysis.

We denote as Γ_G, the *guaranteed subset*, the union of all ψ_k^G. The off-line analysis, which will be described later on, checks that all flows in Γ_G can be safely admitted into the system. We denote as *residual subset* $\Gamma_R = \Gamma \setminus \Gamma_G$.

In event-driven communications, the EDs can trigger a transaction asynchronously following the readings of their sensors. Since the bandwidth allocation is centrally managed by the network Coordinator, it is reasonable to collapse the distributed system into a "virtual" centralized system where the available bandwidth is represented by a *virtual processor*. This permits to study bandwidth allocation as a uniprocessor scheduling problem.

The communication protocol is based on service data frames called *requests*. We identify two types of requests:

- *flow request*, used to declare the presence of a new flow in the network;
- *job request*, used to announce that a new job (a message) is ready on the ED to be transmitted.

Both flow and job requests are used to transfer information from EDs to the Coordinator. In our model, we assume that the length of the requests are negligible compared with the actual message content, so that we may suppose that all of them reach the Coordinator with a null or negligible delay.

The BACCARAT algorithm is composed by: an off-line phase, where node association is performed; and on-line phase were the bandwidth is actually allocated for message transmission. The association and transmission stages are respectively based on flow requests and job requests and are described by Algorithms 1 and 2.

Algorithm 1 Association Handshaking Algorithm

1: A new node declares its flow τ_i to the Coordinator;
2: **if** τ_i is new (a new type) and $\Gamma_G \cup \{\tau_i\}$ can be guaranteed **then**
3: Update $\Gamma_G = \Gamma_G \cup \{\tau_i\}$;
4: **else**
5: Update $\Gamma_R = \Gamma_R \cup \{\tau_i\}$;
6: **end if**

Algorithm 2 Transmission Handshaking Algorithm

1: A job $J_{i,j}$ is ready to be transmitted by an ED sending $req_{i,j}$;
2: The Coordinator schedules the message $J_{i,j}$ according to its policy;
3: The node transmits the $J_{i,j}$ when scheduled by the Coordinator.

During the start-up phase (*association* stage) the Coordinator is in charge of parsing flow requests and updating the Γ_G or Γ_R sets. At every association request the Coordinator carries out an admission test called Guaranteed Flow Admission (GFA) that verifies the schedulability of Γ_G.

At the end of the association stage, the Coordinator listens for job requests and manages the functional communication by scheduling messages (*transmission* stage). The acceptance test, called Message

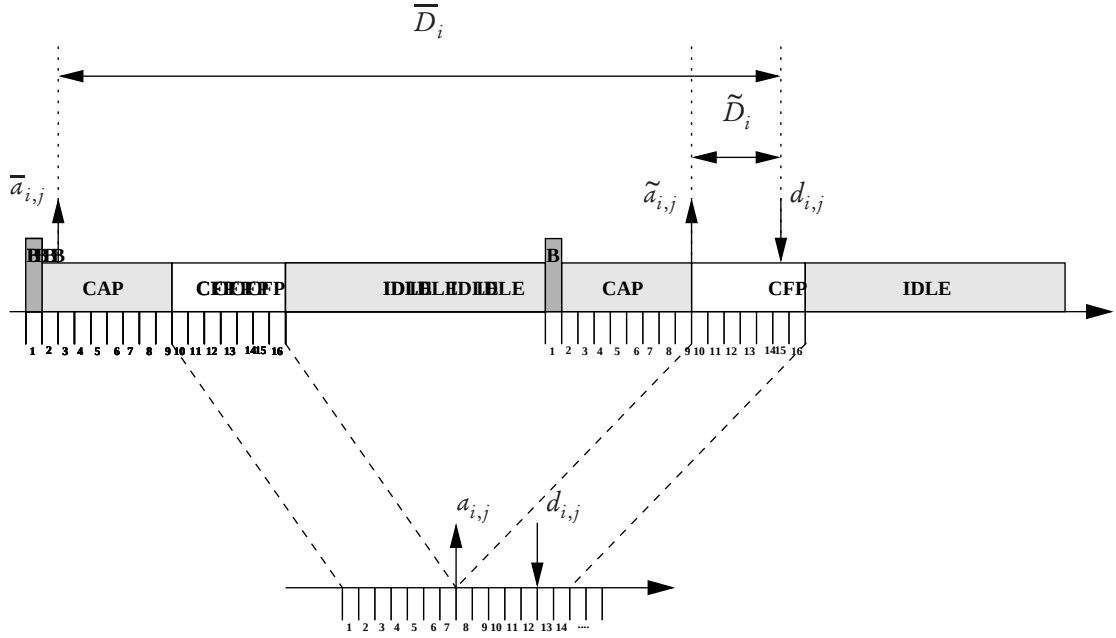


FIGURE 3. Virtual-time representation of the IEEE 802.15.4 Superframe

Acceptance and Scheduling (MAS), is applied to schedule messages based on the requests $req_{i,j}$ received at run-time. During this stage, if a new flow type requires guarantees, i.e. a new node is asking to join the network, a further admission test must be run in background.

We model the Coordinator as a hierarchical scheduler [**MF02**] with two servers managing the transmission bandwidth [**But97**]. The high priority server S_G, manages the guaranteed bandwidth U_G associated to the Γ_G flow set. The low priority server S_R, uses the residual bandwidth, $U_R = 1 - U_G$. S_G dispatches the message requests $req_{i,j}$ contained in an ordered queue J_G to schedule the $J_{i,j}$ from EDs. S_R, instead, dispatches from a different ordered queue J_R the residual message requests. Both queues J_G and J_R are ordered following the flow classification and the message request arrival as described in Section 2.4.1.

Furthermore, the two servers S_G and S_R need to interact one another in order to cope with the dynamic condition of the system. Indeed, it can happen that S_G has unused bandwidth that S_R could use to schedule its messages without jeopardizing the guaranteed schedulability of Γ_G and its entries in J_G.

2.4.1. Bandwidth allocation for IEEE 802.15.4 networks. The standard provides a slotted mode that enables a TDMA based scheme to access the channel within the so-called Contention Free Period (CFP). Every CFP is divided into at most 7 GTSs, each one spawning one or more slots. The standard limits in time the maximum CFP width, setting a minimum duration for the CAP. In other words, in a given interval of time, only a certain portion is available for TDMA.

In this real-time analysis we discuss GTS allocation: to ease the mathematical formalism we apply an axis transformation mapping the time slots into a compact "virtual time" as represented in Figure 3. The virtual time is discretized to slots.

To define this mapping, we need to convert flow attributes from one representation to the other:

$$(\overline{T}_i, \overline{D}_i, \overline{C}_i) \mapsto (T_i, D_i, C_i)\,.$$

According to the specification of the standard [**IEE06**] we denote by:

- $<TS>$ the duration of one time slot;
- B the transmission bandwidth (measured in bits per second);
- $\mathsf{ifs}(\overline{C}_i)$ the *interframe spacing* (IFS);
- $<BI>$ the beacon interval.

The worst-case message size of a flow ($\overline{C}_i$) can be mapped into the virtual time representation as:

$$C_i = \left\lceil \frac{\frac{\overline{C}_i \times 8}{B} + \mathsf{ifs}(\overline{C}_i)}{<TS>} \right\rceil <TS>, \tag{1}$$

where C_i is expressed in number of slots.

The notification of a new instance of the flow is carried by the service message $req_{i,j}$ as described in Algorithm 2. The request is received in the CAP by the Coordinator. If the job is immediately selected for transmission the Coordinator includes the appropriate GTS descriptor in the next beacon frame. Therefore, if a job is activated at time $\overline{a}_{i,j}$, the Coordinator can schedule it in the next CFP, let it be $\widetilde{a}_{i,j}$, as shown in Figure 3. However, the job absolute deadline must be set considering this activation offset. The relative deadline $\widetilde{D}_i$, in the worst-case, is not greater than $\widetilde{D}_i^{WC}$, which is:

$$\begin{aligned} \widetilde{D}_i^{WC} &= \overline{D}_i - (2<BI>) \\ D_i &= \left\lfloor \frac{\widetilde{D}_i - \mathsf{empty}(\widetilde{D}_i)}{<TS>} \right\rfloor, \end{aligned} \tag{2}$$

where D_i is expressed in number of slots. Notice that $2<BI>$ is the largest value for $(\widetilde{a}_{i,j} - \overline{a}_{i,j})$; the function $\mathsf{empty}(x)$ calculates the interval from time $\widetilde{a}_{i,j}$ to time x that cannot be used due to the interferences of CAPs and IDLEs:

$$\mathsf{empty}(x) = \left\lceil \frac{x}{<BI>} \right\rceil (<CAP_{min}> + <I>) + <I>$$

where $<CAP_{min}>$ is the minimum duration of CAP and $<I> \equiv <IDLE>$ is the idle time. Notice that the time needed to transmit a beacon is part of $<CAP_{min}>$.

Finally, the flow period can be transformed with the simple formula below:

$$T_i = \left\lfloor \frac{\overline{T}_i - \mathsf{empty}(\overline{T}_i)}{<TS>} \right\rfloor \tag{3}$$

In the IEEE 802.15.4 standard, an important problem is posed by the maximum number of the GTS descriptors in the beacon, equal to 7. We can allocate up to 7 different flows in a CFP so that an application cannot get more than 7 different flows in a Superframe, although some residual bandwidth is available. We denote this effect as the *flow-cap* effect.

THEOREM 2.4.1. *Let $C = \{C_1, C_2, ..., C_N\}$ be the set of sorted flows computation times, where $C_i \leq C_{i+1} \forall i \in [1, N-1]$. Let <CFP> be the number of TS in the CFP interval in each Superframe. Let S be a non-preemptive scheduling algorithm for the flows requests. A sufficient condition for avoiding the flow-cap effect is that:*

$$1 + \sum_{i=1}^{6} C_i \geq <CFP>.$$

PROOF. A Superframe can accommodate at most 7 different flows. $\{C_1, \ldots, C_7\}$ is the subset of the 7 flows with the lowest computation time among all the possible subsets of 7 flows of $C = \{C_1, C_2, \ldots, C_N\}$. The worst case for a Superframe allocation (the minimum numbers of slots allocated) is obtained considering $\{C_1, \ldots, C_7\}$, where C_7 has not been completed in the previous Superframe and requires 1 more slot from the actual Superframe. This way one GTS is used for 1 slot only. In the worst case the total computation time is $1 + \sum_{i=1}^{6} C_i$. If it is bigger than the available TSs in a Superframe <CFP>, the flow-cap effect cannot happen. If the flow-cap effect does not appear with the application worst case then the application is totally unaffected, which demonstrates the theorem. □

The GFA test is necessary to to guarantee that the set of Γ_G flows can always receive enough bandwidth to meet the real-time requirements. This schedulability test acts as an admission control: every time a new flow τ_i wants to join the system, a new instance of the test is executed. We apply the EDF scheduling policy to assign the bandwidth to the messages, considering the *Utilization Criterion* (UC) and the *Processor Demand Criterion* (PDC) [**But97**]. Nevertheless our test might be easily extended to different scheduling algorithms.

Every flow is denoted by a set of parameters $\tau_i(\overline{C}_i, \overline{D}_i, \overline{T}_i)$ and the partition ψ_k it belongs to. As initial step, the flow parameters are translated into the virtual time-slot representation, obtaining C_i, D_i, T_i, defined in Equations (1)-(3). If there are already w_k^G flows in the system for ψ_k, then the new flow does not need to be guaranteed and will be added to Γ_R. If the minimum number of guaranteed flows w_k^G has not been reached yet, Algorithm 3 (GFA) is executed, and τ_i is eventually added to Γ_G.

Given a new flow τ_i the GFA verifies if $\Gamma_G = \tau_i \cup \Gamma_G^{old}$ is schedulable. The test starts with the necessary condition (UC):

$$U_G = \sum_{i=1}^{N} \frac{C_i}{T_i} \leq 1$$

If it fails, τ_i is rejected (not enough bandwidth), the test returns and the system continues to work with the previous flow set $\Gamma_G = \Gamma_G^{old}$; otherwise it tries the sufficient UC to the most general case with $D_i \leq T_i$:

$$\sum_{i=1}^{N} \frac{C_i}{D_i} \leq 1.$$

If this test succeeds, the flow is accepted, otherwise the PDC is applied. Whereas the latter succeeds, Γ_G is schedulable.

Algorithm 3 GFA: admission test for new incoming flows.

if *(UC algorithm)* - $\sum_{i=1}^{N} \frac{C_i}{T} > 1$ **then**
  Reject request
  Exit test
end if
if *(UC algorithm)* - $\sum_{i=1}^{N} \frac{C_i}{D_i} \leq 1$ **then**
  Accept request
  $\widetilde{U_G} = \left\lceil \sum_{i=1}^{N} \frac{C_i}{T} \right\rceil_{TS}$
else
  if *(PDC with total utilization)* - $PDC(1)$ fails **then**
    Reject request
    Exit test
  else
    Accept request
    $\widetilde{U_G} = 1$
  end if
end if
while $PDC(\widetilde{U_G})$ succeed **do**
  $\widetilde{U_G} = \widetilde{U_G} - \Delta U$
end while
Set U_G to the last successful $\widetilde{U_G}$.

We denote by resource supply bound function (sbf) and by demand bound function (dbf)

$$sbf(t) = U_G \cdot t$$
$$dbf(t) = \sum_{\forall i:\tau_i \in \Gamma_G} \left\lfloor \left(\frac{t - D_i}{T_i} + 1 \right) \right\rfloor \cdot C_i,$$

where

$$D = \{d_k | d_k \text{is a deadline} \wedge d_k \leq min\{H, L^*\}\},$$

where H, L^* are calculated as

$$H = lcm(T_1, ..., T_n)$$
$$L^* = \frac{\sum_{i=1}^{n} (T_i - D_i) U_i}{U_G - U},$$

where n is the number of flows in Γ_G, and U_G is the resource required to have a schedulable flow set Γ_G. Yet U is the actual Γ_G bandwidth occupation, $U = \sum_{i=1}^{n} \frac{C_i}{T_i}$. Thus, the PDC [**Bar03**] states that synchronous flows are schedulable if and only if

$$\forall t \quad dbf(t) \leq sbf(t).$$

The goal of GFA is to prove the schedulability of Γ_G and to find the "best" sbf that leaves Γ_G schedulable. By *best* bound function $sbf^*(t)$ we intend:

$$sbf^*(t) = min\{sbf(t) | dbf(t) \leq sbf(t)\}, \tag{4}$$

where t is the considered time interval and $sbf^*(t) \leq t$. The optimal supply bound function found allows to save as much resource as possible leaving Γ_g schedulable. This is obtained using the PDC with decreasing utilization until the first failure. The iteration step is the equivalent utilization of a $<TS>$, computed as $\Delta U = \frac{1}{<CFP>}$ where $<CFP>$ is the CFP length in $<TS>$. Equation (4) converges to the form of $sbf^* = U_G \cdot t$. Since the first server, S_G, receives a capacity equivalent to U_G, the next one, S_R, gets the residual resource left by the first server $sbf_R(t) = t - sbf_G^*(t) \equiv U_R \cdot t = (1 - U_G) \cdot t$, to schedule the messages in Γ_R.

The complexity of the acceptance test is pseudo-polynomial. Such complexity is affordable since the algorithm is worked out off-line.

So far we have defined the mechanism used during the association stage by the Coordinator to allocate the capacity for the two servers. We now describe the on-line mechanism used during the transmission stage, i.e. how the two servers actually interacts to schedule guaranteed and residual flows.

The MAS algorithm, executed on-line by the Coordinator, controls the input message requests and classifies them into guaranteed or residual. Figure 4 shows how the service request messages are classified according to their arrival time. Within a set of equivalent flows ψ_k, the message requests

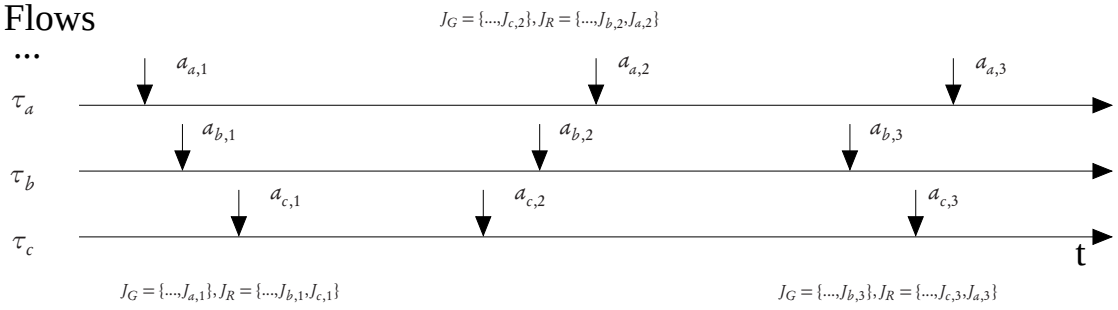


FIGURE 4. Message assignment to S_G or S_R and their queues J_G, J_R. Three flows, τ_a, τ_b and τ_c, of the same ψ_k and their messages $J_{i,j}$ have been considered.

arriving first are selected to be the guaranteed ones, while the remaining ones become the residual. Remember that all the messages in ψ_k are equivalent.

The MAS algorithm can be divided into 2 sub algorithms: *MASguaranteed*, that schedules guaranteed messages, and *MASresidual*, that schedules the residual messages and reclaims unused resource from the high priority guaranteed server S_G. We define the guaranteed and residual capacities respectively as $TS_G = \lceil U_G <CFP> \rceil$ and $TS_R = <CFP> - TS_G$.

MASguaranteed, once classified the messages, extracts the messages request from the ordered queue J_G. The queue is ordered according to the relative deadline D_i carried by the message request. Once the queue is emptied or the bandwidth U_G is exhausted, *MASguaranteed* ends and triggers the execution of *MASresidual*.

If the server S_G empties the waiting queue before exhausting the reserved TSs, then the remaining TSs are given to S_R:

$$TS_R^k = TS_R + \Delta TS_G^k,$$

where ΔTS_G^k is the number of remaining TSs in the k-th Superframe. If in the current Superframe all the guaranteed jobs have been served, all the S_G server bandwidth is reclaimed. The pending requests for S_R are sorted using a priority based algorithm that takes into account the group relevance and the number of redundant messages already transmitted, N_{x_j}.

The mechanism used to select the redundant messages depends on the specific application-level requirements. Following the scenario discussed in Section 2.3 we want to reduce the uncertainty of x_j taking by averaging from a set of independent observations. In this case the priority assigned to the j-th job to be scheduled is given by

$$p_j = \frac{p_{x_j}}{N_{x_j} + 1},$$

while jobs with the same priority are ordered by deadline. If all the observables have the same "intrinsic" priority ($p_{x_j} = p_0$), a job that belong to the group ψ_x with the minimum number of transmitted copies N_{x_j} is chosen from the pending set. These requests are served by S_R using the Superframe budget (TS_R^k) following the logic described in Algorithm 4.

Algorithm 4 MASresidual: acceptance test for S_R dispatcher.

```
1: if S_R queue is empty then
2:     Exit
3: end if
4: set the index i to the first queued request
5: while (TS_R^k > 0) & (S_R queue not empty) do
6:     Compute the request finishing time f_i
7:     if f_i > d_i then
8:         Skip the request
9:     else
10:        Accept the request
11:        Decrease the S_R server budget (TS_R^k) by C_i
12:    end if
13:    if S_R queue is NOT empty then
14:        increase the index i
15:    end if
16: end while
```

Since the accepted requests will be executed in the next Superframe, no preemption can be made by successive requests, hence they are sequentially executed. Then the finishing time f_i of job i is computed starting from the finishing time of the last accepted job f_{i-1}. This should take into account the computation time C_i and the interference induced by the server S_G. Denoting by f_0 the finishing

time of the last guaranteed job, the following relation holds:

$$f_i = f_{i-1} + C_i + \left\lceil \frac{C_i - TS_r^k}{TS_G} \right\rceil TS_G .$$

The algorithm can be implemented keeping ordered message lists so that the servers picks the proper message to be scheduled. The complexity of MAS is $O(n)$, where n is the total number of messages to the Coordinator. Due to its low complexity, such algorithm can be applied on-line.

2.4.2. Performance evaluation. Hereby we discuss a set of tests to prove the effectiveness of the BACCARAT approach in event reconstruction. We model a WSN composed by nodes observing the same events. The topology is the star one and the communication paradigm is follows the beacon-enabled mode of the IEEE 802.15.4 standard.

In the proposed case study event reconstruction depends on the reading of four independent variables x_1, x_2, x_3, x_4; the output of this measurement process will be formally denoted by the linear function:

$$f(x_1, x_2, x_3, x_4),$$

f being undefined if a minimal set of readings is not available at the Coordinator. Adopting our formalism we need at least w_k^G reports ($k = 1,2,3,4$) for each observable.

We assume that sensor readings are affected by random errors; a fortiori also f is affected by statistical uncertainty. Therefore we define the variance of f as σ. A good estimator of σ^2 is s^2 calculated through the mean value of the random variables x_i in the statistical sample. The maximum accepted value for s^2 is that of events reconstructed by the minimal set of readings. An index of the measurement accuracy can be defined by:

$$\frac{s_M^2}{s^2} \in [1; +\infty[$$

where $s_M^2 = \lambda_1^2 \frac{s_1^2}{w_1^G} + \lambda_2^2 \frac{s_2^2}{w_2^G} + \lambda_3^2 \frac{s_3^2}{w_3^G} + \lambda_4^2 \frac{s_4^2}{w_4^G}$.

We now define event reconstruction efficiency ε as the ratio between the number of completely reconstructed events and the total number of events in the simulation.

To keep track of efficiency and accuracy at the same time, we combine the previous metrics into one single quality index defined as:

$$Q = \varepsilon(s_M^2 / < s^2 >),$$

where $< s^2 >$ is the average of s^2 for the reconstructed events.

We perform simulation studies comparing BACCARAT performances in terms of reconstruction efficiency and measurement quality with two popular scheduling algorithms: FCFS, as the simplest and most used bandwidth allocation policy, and EDF, which is optimal for time constrained scheduling. The simulation engine is a discrete event generator, written in C code so that the core of the BACCARAT scheduler can be easily ported to other simulators.

We model event arrivals (detected by the EDs equipped by appropriate sensors) with a Gaussian distribution centered at a mean value (μ) of 1 second and having a standard deviation (s) of 4 milliseconds. All the flow instances at the node level might be activated at the arrival of the event; it follows that the minimum inter-arrival time ($\overline{T}_i$) is the same for all the flows and is approximated with $\mu - 3s$.

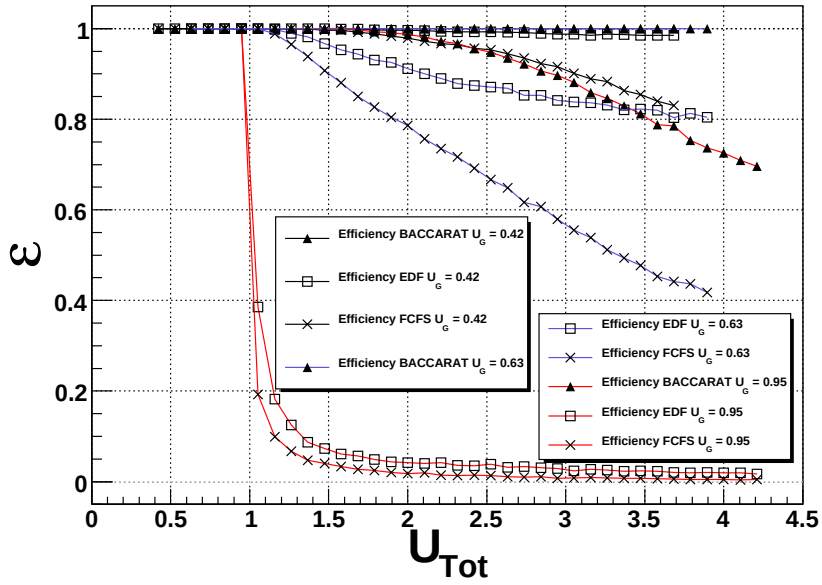


FIGURE 5. Event detection efficiency as function of U_{TOT}

We assume that all the jobs related to the event j are scheduled (so that the GTSs are allocated) before the arrival of the event $j+1$: in other words we constrain $max\{d_{i,j}\} < min\{a_{i,j+1}\}\ \forall i,j$.

Event detection misses due to local inefficiency or wireless transmission failures are not considered in this simulation since we focus on overload conditions taking place when all the nodes detect the event and consequently want to send their reports.

To model the elaboration time needed by the Coordinator to take a certain action upon the detected event, we set the relative deadline $\overline{D}_i$ for each flow to 90% of the period $\overline{T}_i$.

We assume for simplicity that the report size $\overline{C}^k$ for each observable is the same and equal to 300 bytes, corresponding to 3 MAC data frames. The Superframe is configured to have a fixed CFP length of 7 time slots.

In the test cases shown in Figure 5 and Figure 6, we study $\varepsilon(U_{TOT})$ and $Q(U_{TOT})$ for all the algorithms considered, varying the U_G parameter in a set of independent simulation runs (each counting for 10000 events).

The variation of U_G and U_{TOT} is obtained by uniformly increasing, respectively, either the minimum number of required reports per observable (w_k^G) or the total number of flows (w_k). Such uniform distribution of flows depicts the optimal scenario for EDF and FCFS because of the balanced composition in the flow nature.

In the test case of Figure 5 we show $\varepsilon(U_{TOT})$ for different values of U_G. We set $w_k^G = 6,...,10$ with $w_k = w_k^G,...,40$ for $k = 1,2,3,4$. It can be seen that BACCARAT shows full efficiency provided the off-line guaranteed condition is valid ($w_k^G < 9$, i.e. $U_G < 0.94$). Whenever that condition is no longer valid the efficiency drops: starting from $U_{TOT} = 2$, the Coordinator reconstructs less events; over that threshold of U_G, irrespective of U_{TOT}, the efficiency is zero.

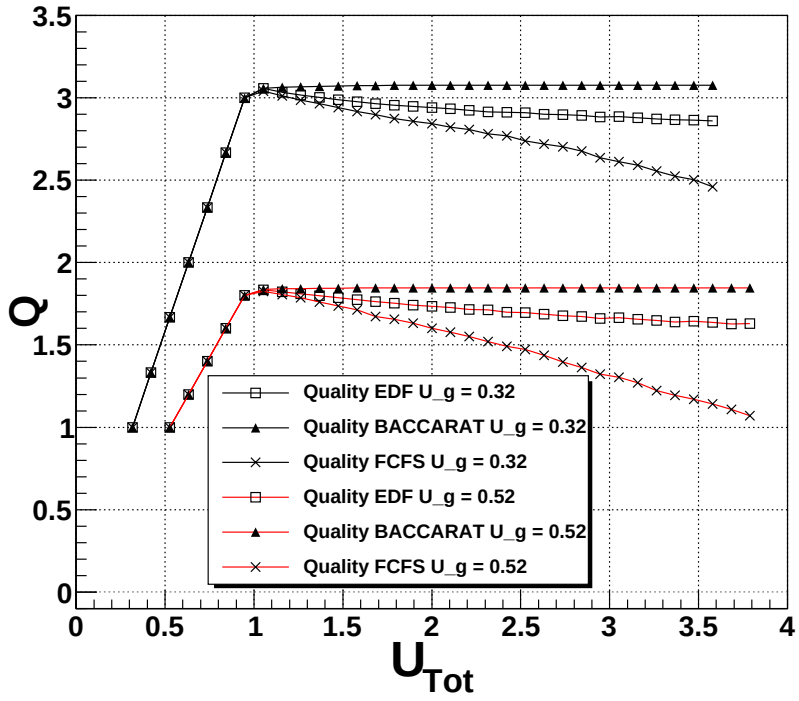


FIGURE 6. Measurement quality as function of U_{TOT}

Under values of U_{TOT} smaller than 1, EDF is performing efficiently. At larger bandwidth demand EDF performs worse and worse. FCFS is always worse than EDF.

The other interesting case is the one of Figure 6, that is the same experimental setup of the previous test case. For all the data sets the required bandwidth for the guaranteed flows is low enough that BACCARAT, EDF and FCFS reconstruct the event with $\varepsilon = 1$ provided U_{TOT} is low (portion of the plot with $U_{TOT} < 1$).

The quality index Q starts from 1, being $U_{TOT} = U_G$ thus $s^2 = s_M^2$, and increases with U_{TOT} until the system is overloaded. When U_{TOT} is close to 1, the maximum value of Q is obtained, i.e. all the bandwidth is allocated. The quality obtained by BACCARAT remains to the maximum level, while with EDF and FCFS it starts decreasing as U_{TOT} increases (EDF always overcomes FCFS). This is important because BACCARAT permits to increase the redundancy level in the network, i.e. the number of flows and thus U_{TOT}, without reducing the event reconstruction efficiency and the measurement quality.

BACCARAT is proven to perform better than popular algorithms operated at their optimal conditions. Although simplified our approach fits the usual specifications of a real-world distributed system where the sensor nodes are required to interact with a given environment and to extract from it actual physical measurements with a defined confidence. We showed the effectiveness of our solution in specific real-time distributed applications devoted to event detection and reconstruction.

2.5. Bandwidth allocation at the network layer

Starting from the star topology it is possible to create more complex scenarios. This is the case of the so called cluster-tree topology, where different coordinators interconnect each other in a hierarchical way. The IEEE 802.15.4 protocol supports the cluster-tree topology. Nevertheless it is not clear how to accomplish this in case of beacon-enabled Personal Area Networks (PANs).

Extending approaches for bandwidth allocation to the case of multi-hop networks is not a trivial task. Among the works addressing this problem we mention [**KCA07**] where Koubaa et al. present a time division mechanism and propose a methodology to achieve fair distribution of bandwidth. An adaptive beacon scheduling scheme has been proposed by Cho et al. [**CA09**], making use of power control and cluster grouping. Other works moved their focus on the real-time routing problem for the cluster-tree topology. Trdlička et al. [**TJH07**], gave an optimal solution for the off-line problem.

We addressed the problem proposing a *component-based* architecture for WSN [**SCN+10**]. Theory from Real-Time Calculus [**TCN00**] is used to model problem in a component-based manner, relying on the mathematical formalism of Real-Time Interfaces [**AH01**, **HM06**]. The network consists of a set of *components* organized in a hierarchical structure that mimics the underlying cluster-tree network topology. A component can either be a node or a cluster of nodes. Each component is able to automatically reconfigure itself and the underlying nodes upon the occurrence of events, such as detection of an environmental situation of interest, the low-battery level of a node, the insertion of a new component, or the removal of an existing component. We develop reconfiguration algorithms which seek to optimize the overall *quality* level provided by the WSN.

2.5.1. Hierarchical Network Model. We consider a classical WSN monitoring application with real-time requirements. The system consists of a set of sensor nodes that collects data about events appearing in a large area. The collected data are therefore packed in messages and sent towards a single collection point called sink, with bounded transmission delay .

Among the several possible network configurations adopted in WSNs, we consider hierarchical structure for its flexibility and scalability. At the lowest level we consider star topologies where a *Control Coordinator* (CC) node manages *End-Devices* (EDs) to form a *leaf cluster*. In order to obtain larger scale networks, the star topology can be extended by interconnecting clusters in a hierarchical way, thus creating what is called a *cluster-tree topology*.

In Figure 7 the leaves of the tree represent the EDs (n_i) with sensing capabilities that are monitoring the environment. All the other nodes are CCs (C_j) and have the main functions of maintaining the topology and allocating the bandwidth that was assigned by its upper layers to its children in a hierarchical structure. The bandwidth allocated to the leaf nodes will be used to transmit data messages. To simplify the presentation, and without loss of generality, we assume that the CCs are not sensing, thus their assigned bandwidth can be entirely redistributed to the children. This hierarchical topology results in a hierarchical bandwidth allocation problem.

To properly react to external events, the ED may operate in one of M different *operating modes*. Each mode involves a different requirement in terms of energy and bandwidth. The operating mode of an ED is controlled by its CC. In this paper we assume that all nodes execute the same application code. Therefore, all of them can operate in one of the M possible modes.

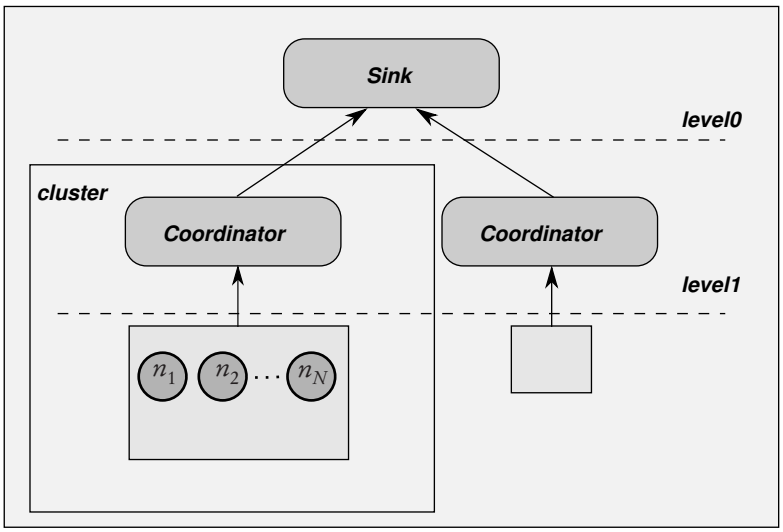


FIGURE 7. WSN hierarchical architecture with end devices, control coordinators and clusters elements.

We denote by θ_i the i-th leaf cluster; $\Theta = \{\theta_1, \cdots, \theta_{LC}\}$ is the set of all leaf clusters, with LC the total number of leaf clusters. Finally, Π is the set of all control coordinators. The notation is illustrated in Table 2.

2.5.2. Bandwidth Allocation Optimization Problem. The problem of allocating bandwidth to the clusters and assigning a mode to each ED can be modeled as an optimization problem. The goal is to maximize a global *quality index* that depends on the operating modes of the ED. Constraints are on energy and on the bandwidth requirements of the nodes.

An optimization problem maximizing a generic quality index in a WSN has to consider both topological and functional aspects of the application to be executed. These aspects are coded in the optimization problem as a set of constraints. In our case, we consider *dynamic constraints*: whenever the system has to react to an external event, the set of constraints needs to be updated due to parameter changes (energy and quality); a new problem must be solved to find a solution consistent with the new conditions.

In terms of notation, we define x_i as the number of nodes in the system operating in the i-th mode, N as the total number of nodes, M as the number of possible modes, and β_k as the bandwidth received by the k-th component.

Our proposal formulates the bandwidth allocation and mode assignment as two separate single-objective optimization problems, instead of a single problem with two objectives.

Bandwidth allocation. First, we start with the bandwidth allocation problem that considers the part of the system topology without the EDs. We introduce the concept of *cluster weight* to calculate the proportion of bandwidth to allocate, thus accomodating design preferences based on the geographic location of the clusters. This results in the cost function

$$\overline{c} \cdot \overline{\beta}, \tag{5}$$

C_i	Control Coordinator node
n_i	End Device node
θ_i	i-th leaf cluster
Θ	set of all the leaf clusters
Π	set of all control coordinators
α	component transmission arrival curve
α^d	component resource demand curve
β	component service curve
μ_j	j-th operating mode of an ED
E	ED energy level
E_i	mode i energy level
Q	ED quality level
q_i	mode i quality level
$x_{i,j}$	number of nodes of cluster i in mode μ_j
N_i	total number of nodes of cluster i
M	number of modes per node
LC	number of leaf clusters

TABLE 2. System notation

with each component c_k of $\overline{c}$ being the weight of cluster k.

At the coordinator level enforcing the network topology requires the incoming bandwidth for each cluster to be the sum of the incoming bandwidths of the child clusters/nodes. Given a cluster k, the *topology constraint* is

$$\forall \Delta \sum_{i=1}^{N_k} \beta_{k,i}(\Delta) \leq \beta_k(\Delta). \tag{6}$$

In order to operate, all the nodes of the k^{th}-leaf cluster, it demands a minimum bandwidth requirement of $N_k \alpha^d_{k,1}$ when all of its nodes are in lowest operating mode μ_1. On the other hand, the maximum bandwidth demand of the k^{th}-leaf cluster is bounded by $N_k \alpha_{k,3}$ where all the nodes operate in the highest operating mode $\mu 3$. Notice that $\alpha^d_{k,1}$ and $\alpha^d_{k,3}$ denote respectively the low-mode and high-mode bandwidth resource demand of any ED node of the k-th leaf cluster, assuming the node bandwidth requests equal for any node within a cluster. This second topology constraint has to bound the leaf cluster bandwidth to the minimum and maximum bandwidth amount requested by the ED nodes composing the cluster

$$\forall k \in \Theta \quad N_k \alpha^d_{k,1} \leq b_k \leq N_k \alpha^d_{k,3}. \tag{7}$$

Mode Assignment. In order to seek for the optimal mode assignment at any leaf cluster, the bandwidth assigned to the leaf cluster has to be known first. At the leaf cluster level we associate to each mode a quality, and we want to maximize the number of nodes in the higher quality modes, so our

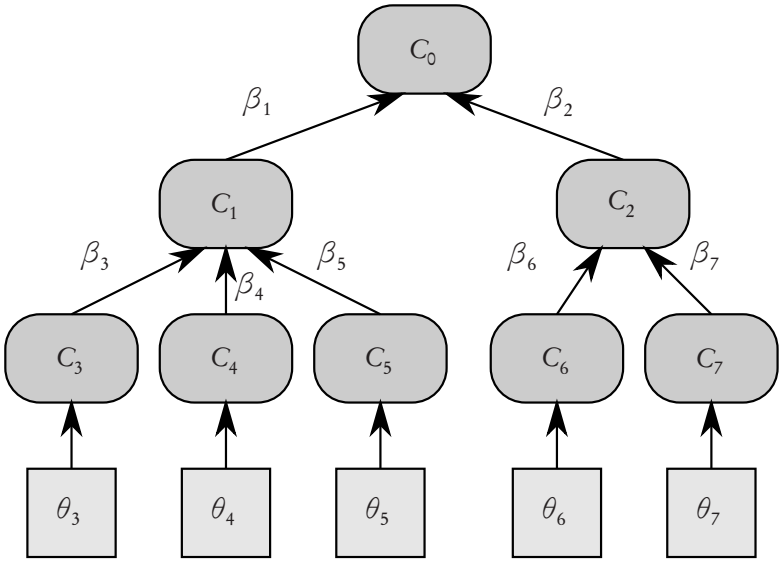


FIGURE 8. Hierarchical architecture with C_0 as the root control coordinator.

value function is:

$$\overline{q} \cdot \overline{x}, \tag{8}$$

where each component q_i is the quality index for mode i.

For any leaf cluster, the *topology constraints* define the number of EDs per cluster and enforce a feasible solution for the hierarchical topology of the cluster itself. For the i-th leaf cluster, the number of nodes is given by the topology of the network, so that the sum of the components of $\overline{x}$ is constrained to be equal to the number of nodes in the cluster:

$$\sum_{j=1}^{M} x_{i,j} = N_i. \tag{9}$$

Since an ED component transmits its information upwards in the logical network topology, it requires bandwidth from its CC. The *bandwidth constraints* describe within the leaf cluster, the dependency among the bandwidth request and availability as:

$$\forall \Delta \; \overline{\alpha}_i^d(\Delta) \cdot \overline{x}_i \leq \beta_i(\Delta), \tag{10}$$

where for a generic leaf cluster i each mode has its proper transmission demand curve α_i^d, and β_i is the bandwidth provided by the cluster coordinator.

To take into account the energy consumption of the EDs and to control the energy assigned to each cluster we introduce an additional constraint:

$$\overline{E}_i \cdot \overline{x}_i \leq E_i, \tag{11}$$

where $\overline{E}_i$ models the energy consumption of each mode and E_i is the maximum amount of energy that the i-th cluster is allowed to consume.

Linearization. Using α^d and β curves as constraints, requires a substantial amount of computation to solve a non-linear optimization problem. In order to obtain a linear problem we approximate Constraints (6), (7), and (10) using linear bounds to the arrival and service curves.

In case of arrival curves, the bounds are $a^*\Delta \leq \alpha^d(\Delta) \leq a^\sharp\Delta$; and $b^*\Delta \leq \beta(\Delta) \leq b^\sharp\Delta$ for the service curve. In a conservative approximation, we use $a\Delta$ with $a = a^\sharp$ to bound the transmission bandwidth requirement α^d, and $b\Delta$ with $b = b^*$ or $b^\sharp$ to bound service curves. In particular, b^* is applied to approximate the β at the right hand side of Equation 6, while $b^\sharp$ is applied to the left hand side βs.

The linearized bandwidth constraints are $\overline{a} \cdot \overline{x} \leq b$, and $\sum^{N_k} b_{k,i} \leq b$.

The solution obtained with the simplifications above becomes an approximated one, but makes the problem solvable with standard linear optimization techniques. See [**GB03**] for a discussion on the loss introduced by this simplification.

Referring to the topology notation previously defined, the bandwidth allocation optimization problem, with the bandwidth constraint linearization, becomes:

$$\begin{aligned} &\max\ \overline{c} \cdot \overline{b} \\ &\text{subject to} \\ &\forall k \in \Pi \ \sum_{i=1}^{N_k} b_{k,i} \leq b_k. \\ &\forall k \in \Theta \ \ N_k a_{k,1} \leq b_k \leq N_k a_{k,3} \end{aligned} \tag{12}$$

The linearized mode assignment optimization problem $\forall\ \theta_i \in \Theta$ is

$$\begin{aligned} &max\ \overline{q}_i \cdot \overline{x}_i \\ &\text{subject to} \\ &\sum_{j=1}^{M} x_{i,j} = N_i \\ &\overline{a}_i \cdot \overline{x}_i \leq b_i \\ &\overline{E}_i \cdot \overline{x}_i \leq E_i. \end{aligned} \tag{13}$$

We first solve the global bandwidth allocation problem (12), then we solve the mode assignment problem (13) for every leaf cluster.

The flexibility obtaining by decoupling the problem into two sequential optimization problems allows to model an increased set of WSN applications. Nevertheless, our approach can be easily extended by applying other objective functions taking into account different WSN conditions.

2.5.3. Optimization Algorithms. We broadly classify the algorithms to find a solution to the problem into design (off-line) and execution (on-line) algorithms.

(1) The *off-line* or *design stage* algorithm computes an optimal initial condition for the WSN system; the initial mode per each node is obtained by using the simplex algorithm on the optimization problem. The initial problem at the design stage is not constrained by energy

or quality requirements, since we assume that at the beginning all the nodes have the same battery level and quality index.

(2) The *on-line* or execution stage algorithm is employed upon occurrence of any dynamic event: whenever there is an occurrence of an event at a location monitored by a leaf cluster, it could trigger a change in the quality requirement of the nodes attached to that cluster, thus asking for a change in the demanded bandwidth. The on-line problem has to quickly provide a sub-optimal feasible solution allowing the WSN to promptly change mode within a short delay.

Off-line Optimization. The off-line solution for mode assignment is solved using the two single objective functions, one for the bandwidth allocation and the other for mode assignment. The Global Mode Assignment (GMA) algorithm is based on the linearized equations of the optimization model and it is described below. The solution is obtained with the help of the simplex algorithm. The result of the GMA algorithm is used as an initial configuration to set up the WSN. At this stag, all the leaf clusters have equal importance and the bandwidth assignment is based on the number of nodes within each cluster which determines the lower and upper bound for the incoming bandwidth.

On-line Problem. Since it is not feasible to run the optimal bandwidth and mode assignment problem on-line, we have developed algorithms to quickly achieve feasible sub-optimal solutions to the optimization problem. We consider two types of transitions that can trigger the request for on-line optimization. The first transition is the occurrence of a physical event that increases the importance of that specific leaf cluster. The second transition is due to a decrease/increase in the available energy, which forces the nodes to work in a low operating mode, thus consuming less energy. In both cases, the event can request a change to a higher quality mode (which requires more bandwidth); or a change to a lower quality mode (which frees some bandwidth). We propose two on-line algorithms for bandwidth reallocation and an on-line algorithm for mode reassignment.

Bandwidth Re-Allocation Algorithms. The on-line algorithm is invoked by occurrence of an external event. In this case, the system has to decide the importance of the event and accordingly increase the weight or priority assigned to that leaf cluster. Depending on the coverage of the search for bandwidth reclamation we propose two algorithms.

The *Local Bandwidth Re-Allocation Algorithm (LBRA)* tries to reallocate the bandwidth among the sibling clusters according to the new weight assigned to each of its child cluster. The weight assigned to the child cluster determines the fraction of the parent's bandwidth assigned to that particular cluster. Since the bandwidth is reclaimed locally within the sibling clusters, this is a local reclaiming algorithm. The complexity of the search is reduced at the expense of a potential loss of overall system quality.

The *Global Greedy Bandwidth Reclaiming Algorithm (GGBRA)* is an on-line global bandwidth search algorithm. Any occurrence of an external event triggers the algorithm, which in turn updates the weight assigned to each child cluster. The algorithm considers weight as a cluster importance. Instead of reallocating the bandwidth among the sibling clusters, this algorithm first tries to update the mode for a certain number of nodes.

The event can trigger a request for a higher quality mode or for a lower quality mode. The algorithm has to find a way to obtain more bandwidth from the sibling clusters. If it is not possible to do so, the algorithm goes up in the hierarchy triggering a request for more bandwidth to the upper layers.

Mode Re-assignment Algorithms (MRA). The mode reassignment algorithms try to update the modes of the nodes attached to those specific leaf clusters which were affected due to on-line bandwidth reclamation algorithms. In case the cluster gains more bandwidth due to higher importance, the nodes must be moved to higher quality mode and vice-versa. The mode assignment during the off-line configuration was managed by solving the simplex algorithm at every leaf cluster. Due to the high computation complexity of simplex algorithm we propose two alternative on-line algorithms for this purpose.

This problem is similar to a bin packing problem. In this case, we need to distribute b_i bandwidth among N_i nodes of a leaf cluster θ_i in order to achieve the maximum quality. Since the total system quality depends on the local quality, we solve this problem as a local problem. However, an optimal solution to this local mode assignment problem does not mean that we have a global optimal mode assignment.

The *GReedy Mode Assignment Algorithm (GRMA)* is based on the greedy solution similar to the bin packing problem. We first try to allocate as many nodes as possible in the higher quality mode, such that the required bandwidth is less than the reallocated bandwidth. Any remaining bandwidth is then assigned to lower quality modes in a greedy manner.

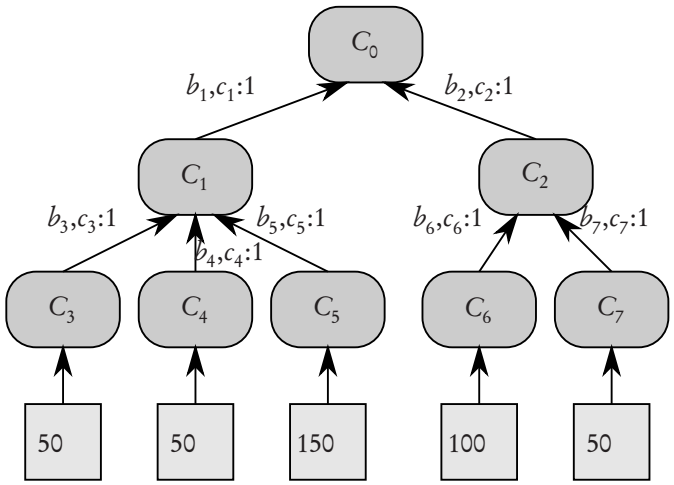
2.5.4. Example. In order to explain the bandwidth allocation and mode assignment with respect to different algorithms, we exemplify a simple dynamic topology reactive to physical events. This configuration is set up for the WSN topology as shown in Table 3. The leaf node C_3 to C_7 are the coordinators of WSN which communicate with the EDs responsible for monitoring the physical events. In order to assign the initial bandwidth allocation and modes, we run the off-line GMA algorithm. At this stage we consider that all the clusters C_i have equal importance and set the weights c_i equal to one. In this case study we have μ_1 as a high quality mode and μ_2 , μ_3 as medium and low quality operation mode respectively. The $\overline{\alpha}$ [4 2 1], $\overline{E}$ [4 2 1] and $\overline{q}$ [8 4 1] coefficients are set according to the mode quality.

Step 1: off-line optimization. We allocate an initial PAN bandwidth (root bandwidth) of 800 units for the entire system and run the off-line GMA algorithm. Every leaf cluster is assigned a maximum energy of 8000 units so that we can ignore the bandwidth and mode constraints arising due to limited energy. We show the effect of energy constraint on the total quality in our simulation.

The results of GMA are shown in Table 3. The total quality for this configuration, i.e $\sum_{\theta_i \in \Theta} c_i \overline{q}_i \cdot \overline{x}_i$, is 1429.

Step 2 (GGBRA Algorithm). The configuration obtained by GMA algorithm is deployed in a physical environment with mode settings and network bandwidth values set according to the results obtained in Step 1. At this stage, there is an important event at leaf cluster 3, which triggers a change in the system state. According to the importance of the event, we increment the weight assigned to cluster 3 by one: $c_3 = c_3 + 1 = 2$. The system either initiates the GGBRA algorithm or LBRA algorithm. Table 4 shows the change in configuration, i.e bandwidth reallocation and mode reassignment, according to GGBRA.

The GGBRA moves 10 nodes in C_3 from μ_2 to μ_1. This is because C_3 has gained more importance due to the occurrence of an external physical event. The GGBRA reclaims this bandwidth from cluster C_4 which is a sibling cluster with lower weight.Since it was possible to reclaim the required bandwidth from a sibling cluster, the GGBRA exited without moving up in the topology and searching for more



$Mode$	C_3	C_4	C_5	C_6	C_7
μ_1	16	16	41	0	0
μ_2	33	33	108	0	0
μ_3	0	0	0	100	49

b_1	b_2	b_3	b_4	b_5	b_6	b_7
649	150	133	133	383	100	50

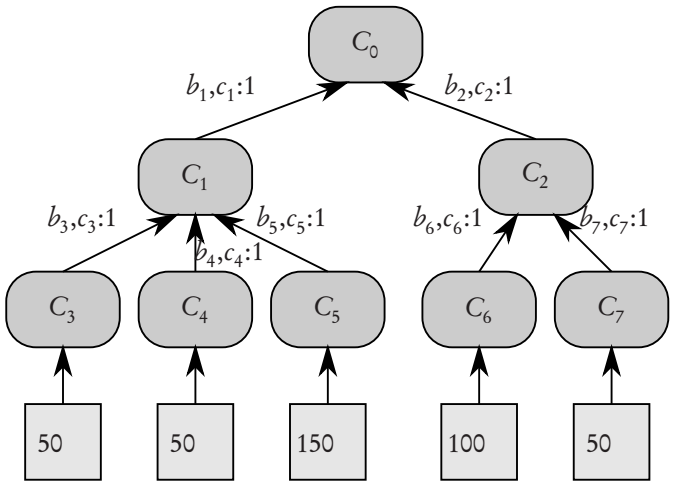
TABLE 3. Offline GMA Algorithm; initial weight $c_i = 1$

complex. The total quality for this configuration is 1718. This increase in quality with respect to the initial total quality reflects the fact that a cluster with more importance has received more bandwidth.

Step 3 (LBRA Algorithm). Similar to GGBRA, LBRA modifies the weight assigned to C_3, $c_3 = 2$. Now LBRA reallocates the bandwidth among the sibling clusters according to its weight. Then, it executes the mode reassignment algorithm to reassign the modes according to the new bandwidth. Since, this algorithm is local, the reconfiguration only affects the sibling clusters, whereas the other cluster components within the topology remains unaffected by this change. The result of LBRA algorithm is shown in Table 6. The bandwidth reallocation moves more bandwidth to cluster C_3 by reclaiming it from C_4 and C_5. The total quality as a result of LBRA algorithm is 1752 which is greater than the total quality achieved by GGBRA algorithm.

Step 4 (GMA Algorithm). Only for comparison, we show the result of GMA algorithm by assigning the same weight assigned to C_3 as in case of LBRA and GGBRA algorithm. Table 6 shows this bandwidth reallocation and mode reassignment with off-line GMA algorithm at the instance of physical event at cluster C_3.

The GMA algorithm runs an off-line simplex algorithm which globally re-distributes the bandwidth in order to get an optimal result. In this case the bandwidth is re-distributed among the sibling clusters of C_3 without effecting other clusters, due to the effect that C_6 and C_7, which have lower



$Mode$	C_3	C_4	C_5	C_6	C_7
μ_1	26	16	41	0	0
μ_2	23	16	108	0	0
μ_3	0	17	0	100	49

b_1	b_2	b_3	b_4	b_5	b_6	b_7
649	150	150	116	383	100	50

TABLE 4. Online GGBRA Algorithm as a result of an event at C_3

$Mode$	C_3	C_4	C_5	C_6	C_7
μ_1	49	16	41	0	0
μ_2	0	33	41	0	0
μ_3	0	0	67	100	49

b_1	b_2	b_3	b_4	b_5	b_6	b_7
649	150	200	133	316	100	50

TABLE 5. Online LBRA Algorithm as a result of an event at C_3

importance, are already having all its nodes in lowest operating mode. The total quality obtained by GMA is 1813 which is higher than the quality obtained by the on-line algorithms.

2.5.5. Simulation Results and Remarks. In this section we describe the performance of different algorithms with respect to the total quality of the WSN. By the total quality we mean the aggregate of number of nodes in each mode for every leaf cluster θ_j. $\sum_{\theta_i \in \Theta} c_i \overline{q}_i \cdot \overline{x}_i$ where c_i is the importance of the i-th cluster.

$Mode$	C_3	C_4	C_5	C_6	C_7
μ_1	49	12	11	0	0
μ_2	0	36	138	0	0
μ_3	0	0	0	100	49

b_1	b_2	b_3	b_4	b_5	b_6	b_7
649	150	200	133	316	100	50

TABLE 6. Offline GMA Algorithm as a result of an event at C_3

The total quality function is quite general and with its parameters allows to model most of the quality function that real applications want to pursue.

We show the analysis for a fixed topology. We compare the online algorithms with the global optimal algorithm as described in Section 2.5.3. The graph in Figure 9 compares all three algorithms against the total number of sensing nodes in the WSN. We keep the energy assigned to each cluster and total bandwidth of the network fixed. The energy value is selected such that it does not affect the bandwidth constraints, i.e, all the nodes can operate in maximum mode with sufficient available energy. In each of three algorithms the total quality increases with the number of nodes up to a certain point. This is because the total available bandwidth is more than sufficient for all the nodes to operate in the highest mode and we have maximum quality. There after the mode selection depends on the applied algorithms up to a certain point. In this range, it can be seen from the figure that the global optimal algorithm always gives better total quality compared to the on-line algorithms. After a saturation point, increasing the total number of nodes does not increase the total quality. This is due to the fact that our analysis helps us to determine the maximum number of nodes with which we can guarantee a minimum bandwidth to every node within the network.

Figure 10 shows the comparison among the total network quality values obtained with different algorithms. The total bandwidth and the number of nodes in the system is fixed; we can see how increasing energy allows nodes to operate in higher modes and hence use more bandwidth. Energy directly affects the maximum allowed bandwidth for every cluster, so as the energy increases the bandwidth increases and more nodes can operate in higher modes. This can be seen in the figure where increasing energy results in increasing total quality. This trend can be observed up to a certain point where any further increase in cluster energy cannot increase the assigned bandwidth due to the constraint of Equation 10 constraint. After this saturation point the total quality remains almost unchanged.

Remarks. We addressed the problem of adaptive bandwidth allocation for reactive WSNs, organized in a hierarchical cluster-tree structure. Our technique is based on a component-based methodology that increases the flexibility of the system by allowing to easily plug-in new components into the system. We presented an off-line optimal global solution for the initial configuration of the network. In addition, the problem of optimally allocating the available bandwidth has been decomposed into smaller local sub problems that can be solved on-line by using heuristic approximations. Our solution

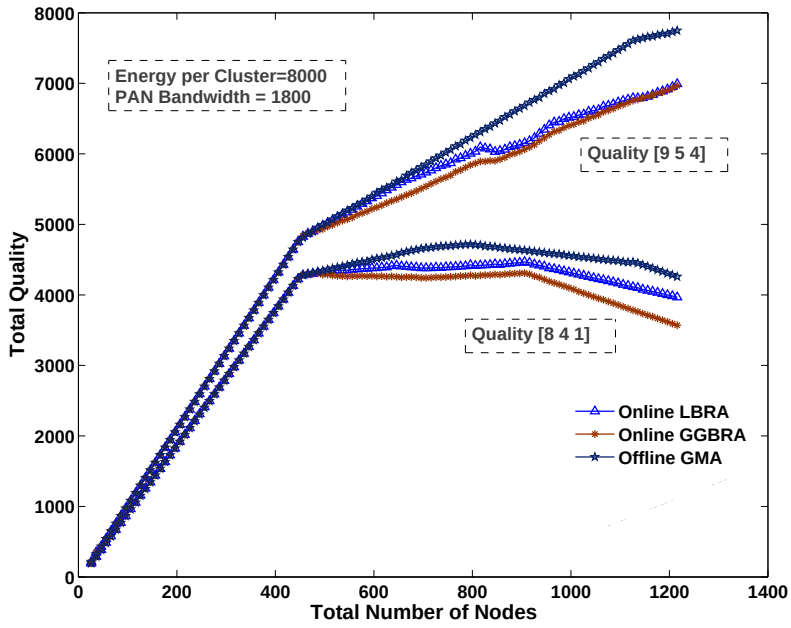


FIGURE 9. Number of Nodes vs Total Quality.

is scalable with the size of the network, and in particular it is possible to find sub-optimal solutions without the need of a complete exploration of the cluster-tree.

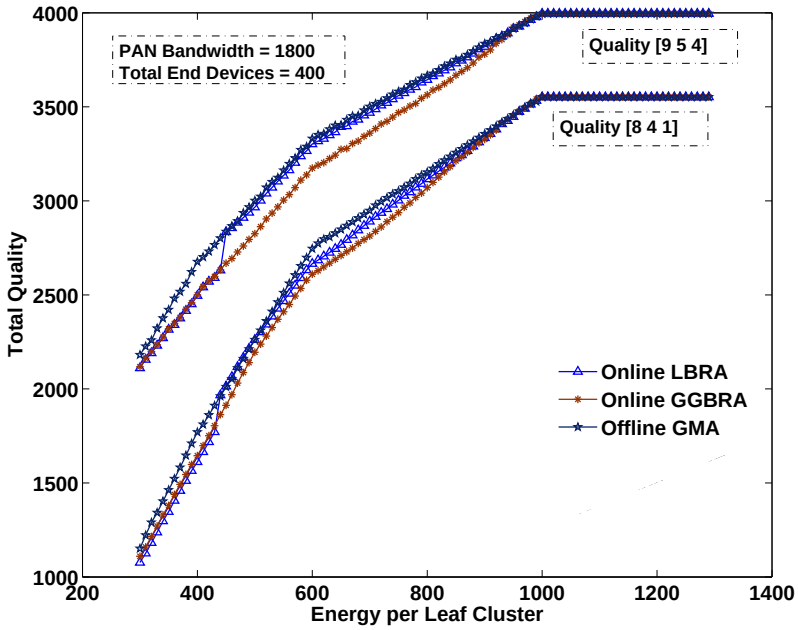


FIGURE 10. Energy per Cluster vs Total Quality.

CHAPTER 3

Local Processing: Image Mining Techniques

GATHERING information from a network of scattered cameras, possibly covering a large area, is a common feature of many video surveillance and ambient intelligent systems. However, most of classical solutions are based on a centralized approach, in which image processing is accomplished in a single unit. In this respect, the introduction of in-network processing is well motivated by several advantages [**Rem04**]:

- Speed: in-network processing is inherently parallel; in addition, low-level nodes of a hierarchical architecture permit to reduce the computational burden of the high-level decisional nodes;
- Bandwidth: in-node (local) processing permits to reduce the quantity of transmitted data, by transferring only relevant features about the observed scene and not the redundant image data stream;
- Redundancy: a distributed system may be reconfigured in case of failure of some of its components, still keeping the overall functionality;
- Autonomy: each of the nodes may process the images asynchronously and may react autonomously to the perceived changes in the scene.

These motivations suggest to move part of the intelligence towards camera nodes. In these nodes, artificial intelligence and computer vision algorithms should be able to provide autonomy and adaptation to internal conditions (e.g. hardware and software failures) as well as to external conditions (e.g. changes in weather and lighting conditions).

In a distributed reactive application, the prerogative of the WMSNs is to report any relevant change in the scene, where some real-time constraints must be satisfied. To this end, one should take into account both local computation and network communication issues to bound the maximum latency in the reaction.

In this Chapter we focus on the aspects of local processing, considering a wireless multimedia sensors network for monitoring applications [**MMN+11**]. Techniques to extract compact information from the acquired images are studied with a twofold objective: reduce the amount of data that has to be transfered by the camera node, thus limiting the network communication delay; have an light-weight efficient implementation, thus limiting the computation delay.

A hierarchical organization can be used to combine these techniques at different levels. For instance, local data mining can produce partial decisions, while the final decision is taken at an aggregation point. Two main approaches are considered: change detection and machine learning. The first, based on simple background subtraction, is generally lighter and faster, though less accurate. The second, based on a cascade of classifiers, is more effective, but requires more resources. The two can be

combined in a hierarchical manner in a two tier architecture. In the lower tier, nodes with limited power execute the change detection algorithm and produce a waking signal for the upper layer. This signal triggers the elaboration of more powerful nodes running machine learning one.

Alternatively to the cascade of classifiers, a neural network approach has been thoroughly studied. This provides a lighter implementation that, under certain operating conditions, can avoid the usage of a multi-tier architecture.

The approaches are demonstrated in the domain of Intelligent Transportation Systems (ITSs), where a parking area monitoring application is developed.

3.1. Background

Data mining of visual information involve particular problems in computer vision, such as change detection in image sequences, object detection, object recognition, tracking, and image fusion for multi view analysis.

For each of this problems, a vast literature exists with several approaches and solutions, see e.g. [**RAAKR05**] for a survey of change detection algorithms. However,most of them are not suitable for WMSNs, due to their high complexity demanding large memory and heavy computations. Nevertheless, some attempts to employ non-trivial image analysis methods to WMSNs have been done. For example, [**YGM08**] presents a WMSN able to support the query of a set of images in order to search for a specific object in the scene. To achieve this goal, the system uses a representation of the object given by the Scale Invariant Feature Transform (SIFT) descriptors [**Low04**]. SIFT descriptors are known to support robust identification of objects even among cluttered background and under partial occlusion situations, since the descriptors are invariant to scale, orientation, affine distortion and partially invariant to illumination changes. In other words, using SIFT descriptors allows retrieving the object of interest from the scene, no matter at which scale it is imaged.

Interesting computer algorithms are also provided on the CMUcam3 vision system [**RGGN07**]. Besides basic image processing filters (such as convolutions), methods for real-time tracking of blobs on the base either of color homogeneity or frame differencing are available. A customizable face detector is also included. Such detector is based on a simplified implementation of Viola Jones detector [**VJ04**], enhanced with some heuristics to further reduce the computational burden. For example, the detector does not search for faces in the regions of the image exhibiting low variance.

Camera sensors have limited FOVs and can only perceive a portion of a scene from a single view point. In addition, monocular vision totally lacks 3D information. To mine the entire scene and to deal with occlusions, it is natural to consider the multi-view capabilities of WMSNs.

Due to bandwidth and efficiency considerations, however, images cannot be routinely shared on the network, so that no dense computation of 3D properties (like disparity maps and depth) can be made. Nevertheless, the static geometrical entities observed in the scene may be suitably codified during the setup of the acquisition system. In addition, specially designed references may be introduced in the scene for obtaining an initial calibration of the views acquired by each camera, thus permitting to find geometrical correspondences among regions or points of interest seen by different nodes. To this end, a coordinator node, aware of the results of such calibration step, may be considered, so as to translate

events from the image coordinates to physical world coordinates. Such approach may produce more robust results as well as a richer description of the scene.

Neural Networks are used as core logic in WSN based systems for performing RSSI based localization [**AFD07**] or for data aggregation [**FX09**]. For high end systems, several examples exist in literature where Neural Networks have been used for image recognition. In the remainder of this Chapter we show how the same approach can be effectively adopted for resource-constrained systems such as WMSNs.

3.1.1. Neural Networks. The origin of Neural Networks dates back to the proposal of the *perceptron* by Rosenblat [**Ros88**], a computational model for mimicking the neuron activity. Similarly to human neurons, perceptrons have weighted inputs (resembling human synapses) and they are fired (they change their output) hen a sufficient stimulus is applied. Mathematically, the perceptron is defined as:

$$y = f\left(\sum_{i}^{N} w_i x_i + b\right)$$

where the output y depends on the weighted (w, vector of weights) sum of the input vector x, and on a bias component b.

Similarly to human neurons, perceptrons must be trained through a set of examples. Given an training set, composed by an input vector x^t, and an expected output value y^t, by acting on the w vector, the training process tries to minimize the quadratic error of the network output

$$E(y - y^t)^2 \quad .$$

Notice, that the training algorithm does not converge if the input vectors of the training set are not linearly separable [**Nov63**].

Different connection schemas produce networks which unique features can be exploited for solving different problems. In literature, two categories of network topologies are considered: fully connected and layered networks. In the first case, all the perceptrons are connected with the others through a complete connection of output and inputs. This topology, known as *attractors* or Hopfield Networks [**Hop88**], is generally used for its capability to store information, resembling human memory. The dimension of this storage, expressed by the number of memories that this network can conserve is proportional to the number of neurons. Layered Networks, also known as Feed-Forward Neural Networks have more similarities with human sensory cortex. They are characterized by non-linear activation functions and they process data in a sequential, step-based manner. A particular case of these networks is the so-called Multi-Layer Perceptron (MLP).

MLPs are composed of a variable number of layers, each containing a different numbers of perceptrons. The first and the last layers are known respectively as input and output layers, while the internal layers are called hidden layers. Perceptrons are locally not connected (i.e. no connection exists between elements of the same layer) but are linked with the two adjacent layers (Figure 1). A fundamental property of MLPs is that, given at least one hidden layer, they can approximate every function using a finite number of perceptrons [**Hor91**]. The successful application of MLPs to pattern recognition, data fitting and system modeling, largely depends on this property. Different MLPs configurations can be generated acting on at least four parameters:

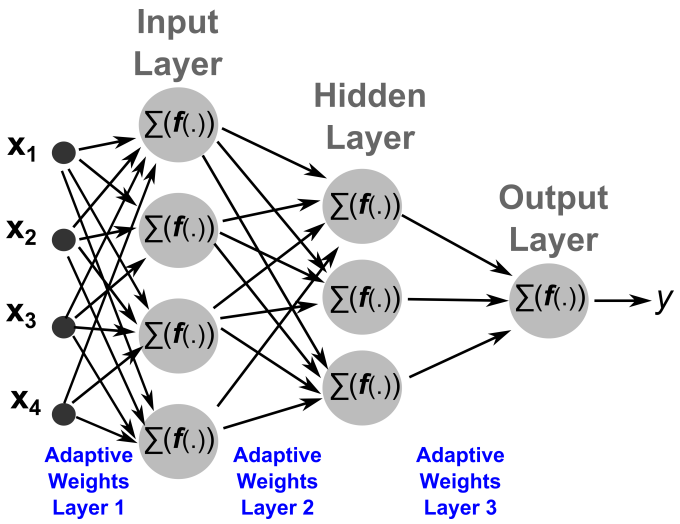


FIGURE 1. A Multi-Layer Perceptron (MLP) with three layers.

- The number of layers;
- The number of perceptrons per layer;
- The activation function of each perceptron;
- The training algorithm.

Different functions can be adopted (Table 1 lists common ones) leading to different network behaviors.

Training algorithms can be classified to three main families: supervised, reinforcement and unsupervised learning algorithms. The main differences is represented by the way the training set is created. We focus only on the first category.

The algorithm typically used for supervised training of MLPs is the back-propagation algorithm [**BH69**]. In its original formulation, this algorithm uses the *descendent gradient* to find the combination of neurons weights that minimizes the network error. This method requires the activation functions to be differentiable and the definition of a valid error function. The first condition can be satisfied by sigmoid functions (e.g. tan-sig and log-sig defined in Table 1) that are completely differentiable and by functions with a finite number of singularities (e.g. hard-limit in Table 1 has two points of singularity, -1 and 1). For the error function, the general approach is to adopt the sum of errors present at the output layer.

The idea of the back-propagation is that, the input is presented to the network and is forwarded up to the output layer. Here, the error is computed and used while moving backward in the layers (i.e. from output to input) to adjust the weights according to the method described in [**BH69**]. This forward and backward procedure is repeated until the output error is below a target threshold.

The algorithm inherits the gradient descent limitations, in particular, it can produce solutions that are local minima (instead of the desired global minimum) and it can require long time to converge. For mitigating the first issue, various algorithm modifications can be adopted: Completely re-initialize the system with different initial weights when the algorithm stops; Apply a perturbation on weights

Function	Formula	Differentiable	Use
Purelin	$f(x)=x$	Yes	Typical in output layer
Hard-limit	$f(x)=\begin{cases} 1 & x>1 \\ x & -1\leq x\leq 1 \\ -1 & x<-1 \end{cases}$	Yes, excluding discontinuity points -1 and 1	Both in output and hidden layers
Tan-Sig	$f(x)=\frac{2}{1+e^{-2x}}-1$	Yes	Hidden layers
Log-Sig	$f(x)=\frac{1}{1+e^{-x}}$	Yes	Hidden layers

TABLE 1. List of common activation functions.

when for various consecutive iterations the error remains stable; Prune or change the training set. The second issue is due to possible oscillations of the gradient descent around the solution, yielding a large number of iterations before converging. This depends on the fixed value of the algorithm step size, *the learning rate*. Known solutions to mitigate this problem are: making the learning rate a decreasing function of the error (i.e. use a faster learning rate in the beginning and a slower one when approaching the minimum); using low-pass filter on the weights update. An alternative to the gradient descent is the Resilient back-PROPagation (RPROP) [**RB93**] that shows significant improvements in terms of convergence speed. The basic idea of this approach is to decouple the weight adjustment amount from the magnitude of the gradient of the error. The gradient is only used to calculate the sign of the weight increment, while a fixed step determines the amplitude. To increase convergence speed, the weight amplitude step is a changes according to the sign of the gradient in two consecutive iteration of the learning algorithm.

3.2. Application Scenario

Intelligent Transport Systems (ITSs) are nowadays at the focus of public authorities and research communities aiming at providing effective solutions for improving citizens lifestyle and safety. The effectiveness of such kind of systems relies on the prompt processing of the acquired transport-related information for reacting to congestion, dangerous situations, and for optimizing the circulation of people and goods. To obtain a dynamic and pervasive environment where vehicles are fully integrated in the ITS, low cost technologies (capable of strongly penetrating the market) must be let available

by the effort of academic and industrial research: for example low cost wireless devices are suited to establish a large-area roadside network.

In the following we present an application scenario inspired by the ITS problem. The two approaches for image mining described in this Chapter are evaluated with respect to a prototype deployment of an ITS within the scope of the IPERMOB project [**www11a**]. IPERMOB proposes a pervasive and heterogeneous infrastructure to monitor and control urban mobility. Its multi-tier architecture aims at the integration and optimization of the chain formed by data collection systems; aggregation, management, and on-line control systems; off-line systems aiming at infrastructure planning; information systems targeted to citizens and municipalities to handle and rule the vehicle mobility. Moreover IPERMOB proposes to use camera-based Wireless Multimedia Sensor Networks to collect traffic-related data. Tiny smart cameras nodes have been prototyped and used for the actual deployment. Each node is essentially equipped with: a micro-controller; an IEEE 802.15.4 transceiver; a low-resolution CMOS camera.

Within the scope of the IPERMOB project, a parking area monitoring application is considered. In particular, a WMSN has been deployed to produce a real-time information about the occupancy of parking slots in a given area. More details on the project are given in Chapter 6.

FIGURE 2. The parking slot occupancy problem in the IPERMOB project.

The main advantage of smart camera over classic sensors is versatility. An image (and a sequence of images) contains much more information than a scalar value, therefore camera-based sensors can perform a wide range of tasks, functionally replacing different types of sensors. For example, a smart camera can be used as a light sensor, a motion detector, an occupancy sensor, etc. Smart cameras can also quantitatively replace classic sensors. For example we use a single smart camera to monitor the occupancy status of up to 10 parking slots, instead of using one inductive sensor for each space. The major drawback of smart cameras is higher cost and power consumption. Yet the higher cost per unit is offset by the reduction in the number of sensors, that also leads to cheaper deployment and maintenance. The power consumption problem is tackled adopting local (on-board) image processing:

only relevant information (e.g., parking slot occupancy status) is sent over the network, thus reducing the number of exchanged messages which are the main cause of energy consumption.

3.3. A Multi-tier Image Mining Approach

The first approach proposed is based on a hierarchical processing architecture. Essentially, three types of processing can be identified

- change detection: fast low-level algorithm that produces a preliminary information, i.e. occurrence of a change in the parking slot, which is used to trigger the processing of higher levels;
- object detection: slow and complex mid-level algorithm that produces a refined information, i.e. partial decision, about the occupancy status of the parking slot;
- decision making: highest level algorithm that aggregates the information of the lower layers, both the change detection and the object detection; such information are provided by more than one node, i.e. multiple views.

A common hypothesis for the multiple views aggregation process works on the basis of a given knowledge base of information, relative to the approximate possible positions, or Regions of Interest, where the objects can be detected. Under this assumption, the decision making algorithm can be "partially" implemented on the single node. For example, the same node performing the low- and mid-level processing can produce a partial decision on the level of occupancy of the parking slot. The decision-making node can use its knowledge about the network deployment to aggregate the nodes' partial decisions: e.g. weighted according to the nodes' view of the detected objects. In the following Section,

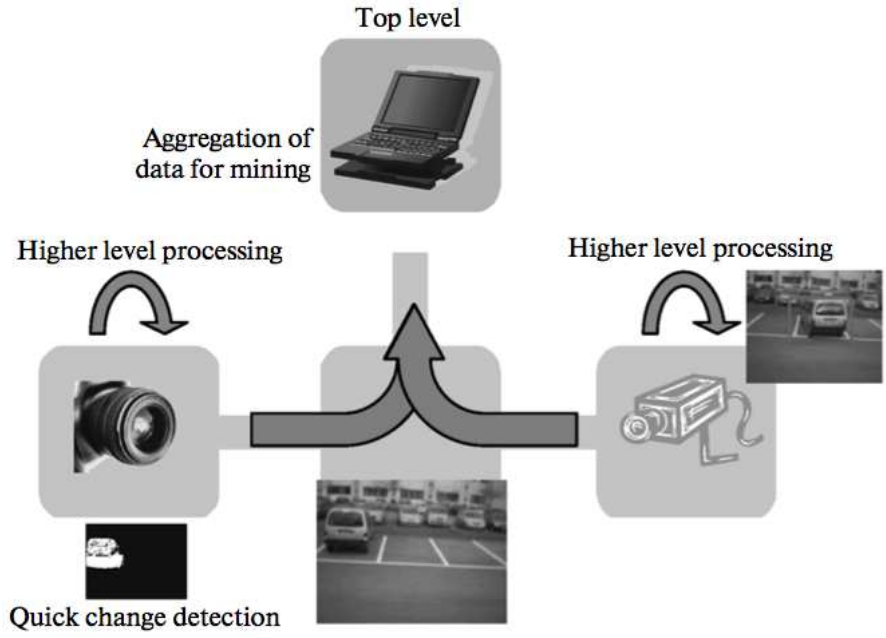


FIGURE 3. A multi-tier approach for parking slot monitoring.

some details about the low-level processing algorithms are given.

3.3.1. Change Detection. The task of identifying changes, and in particular regions in the image under analysis, taken at different times is a widely diffused topic encompassing very different disciplines [**RAAKR05**]. In this context, the goal is the identification of one or more regions of pixels, relative to a change in the scene, and occurring within a sensible and a priori known area. In order to perform fast, efficient, and enough reliable methods for this change detection, the generally adopted low-level methods are based on both frame differencing, and statistical methods. This assumption can be either the quick and immediate answer to a simple problems, or a preliminary synthesis for a deeper and more effective higher level analysis. The general model used is mainly based on background subtraction, where a single frame, acquired in a controlled situation, is stored (BG-image) and thereafter used as the zero-level, for different types of comparison in the actual processing.

The first methods are very low-level ones and are thought in such a way that it can be possible and feasible to implement them also at firmware level. Examples of these are be:

- Thresholding of the background-subtracted images;
- Basic edge detection on binary image obtained by thresholding.

Another class of methods is based on statistical histogram analysis. Template histograms of the regions of interests in the BG-image are computed and stored. Then a standard distance, such as the Kullback-Leibler divergence is computed between the region of BG-image, and the region of the actual image. If such distance overcomes a fixed threshold, then a change event is detected.

3.3.2. Object Detection. As it is well-known, the problem of detecting classes of objects in cluttered images is challenging. Supervised learning strategies have demonstrated to provide a solution in a wide variety of real cases, but there is still a strong research activity in this field. In the context of WMSN, the preliminary learning phase may be accomplished off-site, while only the already trained detectors needs to be ported to the network nodes.

Among machine learning methods, a common and efficient one is based on the sliding windows approach; namely rectangular sub-windows of the image are tested sequentially, by applying a binary classifier able to distinguish whether they contain an instance of the object class or not. A priori knowledge about the scene or information already gathered by other nodes in the network may be employed to reduce the search space either by (a) disregarding some region in the image and (b) looking for rectangular regions within a certain scale range (e.g. rectangular regions covering less than 30% of the whole image area). In our case, both information can be obtained by the change detection algorithm. Essentially, only those regions are processed by the classifier where a change has occurred and where the change has involved an area with the minimum scale requirements.

Concerning the binary classifiers itself, among various possibilities, the Viola-Jones method is particularly appealing. Indeed, such classifier is based on the use of the so-called Haar-like features, a class of features with limited flexibility but which is known to support effective learning. A Haar-like feature is essentially given by the difference between sum of pixels in rectangular blocks; such Haar-like feature is computable in constant time, once the integral image has been computed (see [**VJ04**] for details). Thus, having enough memory to store the integral image, the feature extraction process requires low

computational power. Classification is then performed by applying a cascade of classifiers, as shown in Figure 4.

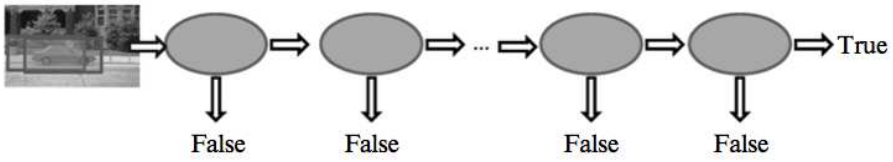


FIGURE 4. Example of a cascade of classifiers for object detection in case of parking slot monitoring.

A candidate sub-window which fails to meet the acceptance criterion in some stage of the cascade is immediately rejected and no further processed. In this way, only detection should go through the entire cascade. In addition, the cascade may be trained, for example using gentle AdaBoost, in such a way that most of the candidate sub-windows not corresponding to instances of the object class are rejected in the first stages of the cascade, with great computational advantages.

The use of a cascade of classifiers permits also to adapt the response of the detector to the particular use of its output in the network, also in a dynamical fashion, in order to properly react to changes to the internal and external conditions. First of all, the trade-off between reliability of detections and needed computational time may be controlled by adaptive real-time requirements of the overall network. Indeed, the detector may be interrupted at an earlier stage in the cascade, thus producing a quick, even though less, reliable output which may be sufficient for the current decision making problem. In the same way, by controlling the threshold in the last stage of the cascade, the WMSN may dynamically select the optimal trade-off between false alarm rate and detection rate needed in a particular context.

3.3.3. Performance. For what regards object detection, several cascade of classifiers have been trained, each one specialized in detecting particular view of cars (e.g. front, rear and side views). A large set of labeled acquisition has been made of the real case study and used for training purposes. We notice that first stages in the cascade have low computational complexity but reasonable performance (that is almost 100% detection rate but even 50% false alarm rate). Composition of the stages entails then high performance of the entire cascade, since both overall detection rate and overall false alarm rate are just the product of the detection rate and false alarm rate of the single stages. For example, using $N = 10$ stages in the cascade, each one with detection rate $\nu = 99.5\%$ and false alarm rate $\mu = 50\%$, one gets an overall detection rate $\nu_{global} = \nu_N \approx 95\%$ and false alarm rate $\mu_{global} = \mu_N \approx 0.1\%$.

3.4. A Neural Networks Approach

The second approach proposed is based on Neural Networks (NNs). Essentially this can be considered an alternative implementation of the Object Detection mechanism described in the previous Section. In the following the overall process of the definition and implementation of a Neural Network for parking slot monitoring is presented.

3.4.1. Design process for WMSNs. The implementation of a car parking monitoring system involves various steps, with a complexity level increased by the limited computational power of our target devices.

The design process requires a preliminary phase of data collection. Long-term data acquisition campaigns were performed in different parking areas. Images were acquired in various ambient conditions using prototype boards. To manage the data acquisition process we developed a configurable data acquisition software based upon ROOT (Root), the CERN framework for data processing.

The next phase involves the identification of optimal NN structure and properties. For reducing the duration of this process, we extensively exploited the Matlab™ Neural Network and Image Processing Toolboxes. These toolboxes provided a fast way for implementing and testing different configurations of the NN on real datasets. In this way it was possible to focus on the following design issues:

- definition of the input that shall be given to the NN and identification of the algorithm to extract such input;
- definition of the NN configuration that provides the best accuracy;
- acquisition of the training set.

Research of the optimal network input. The effectiveness of the pattern recognition algorithm deeply relies on the selection of the optimal neural network input. In general, when MLP Networks are used for image recognition, a step composed of one or more preprocessing algorithms is adopted to reduce the information redundancy of the image. This because inputs characterized by limited redundancy provide, in general, advantages in terms of increased network accuracy and reduced computational cost. In our case, shrinking the amount of resources required was extremely important and thus we spent a great effort in researching an optimal preprocessing algorithm.

Transformation	Complexity	Memory	Input Size
Mean and Variance	$O(N)$	N	2
M-bin Histrogram	$O(N\frac{log_2(M)}{2})$	$N+M$	M
Edge Energy	$O(KN)$	$4N$	Variable

TABLE 2. Computational cost associated with candidate pre-processing functions. Algorithm complexity and memory usage are presented for an N-pixels image. The Canny algorithm for edge detection requires at least $K > 60$ operations.

This research started with an analysis on various image processing algorithms, focusing on the amount of computational power and memory required. Table 2 lists the algorithms in order of complexity, showing: number of operations, memory and NN input size. Notice that, independently from the chosen algorithm, the input dimension impacts on the amount of memory and computational power required for both training and running the network. Indeed, for each input there is a weight that the system has to store and update. A large number of algorithms, for example SIFT [**Low04**] and SURF [**BETVG08**], have been discarded for excessive requirements in terms of memory and/or number of operations per pixel.

For all algorithms listed in Table 2, complexity and memory requirement can be reduced by feeding them with a subset of the original image. This portion of the frame is the Region of Interest (ROI). In our case, it is convenient to choose the ROI as the area of a single parking slot.

While the definition of the first two algorithms is straightforward, the Edge Energy requires additional explanations. The core of this preprocessing method is the Canny algorithm [**Can86**] used as edge-detector. This algorithm is presented in literature in various flavors, but it generally requires more than 60 operations per pixel (i.e. in Table 2, $K > 60$). The output of this algorithm is a binary image C representing the edges of the original image. The binary image has the same dimension of the original image and thus it is too big to be directly used as network input. For this reason, we computed the energy of C as

$$E_C = \sum_{i,j} C_{i,j},$$

obtaining an index representing the amount of edges in the image.

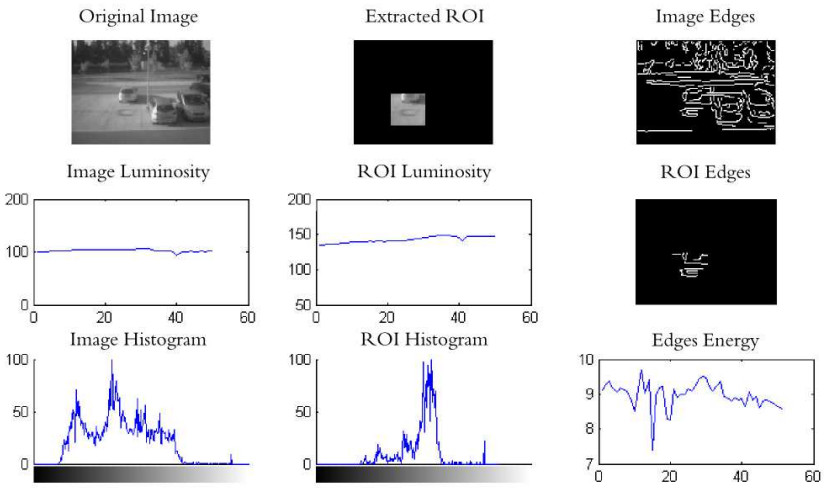


FIGURE 5. Outputs of algorithms of Table 2 when the Region Of Interest does not contain vehicles. First row contains the original gray-scale image (320x240 pixels), the ROI subset and the edges extracted from the original image. In second row, the first two plots from the left contains the trends of the luminosity value computed for the full image and for the ROI, while the plot on the right contains the edges extracted from the ROI. In third row are plotted the image and ROI normalized histograms and Edge Energy trend.

Lacking direct quantitative approaches, we continued the research of a feasible algorithm through a qualitative analysis. The candidate algorithms were fed with images extracted from a real dataset characterized by variations on slot statuses. Their outputs were plotted in order to visually analyze their behaviors in case of empty or busy slot (Figures 5 and 6 respectively).

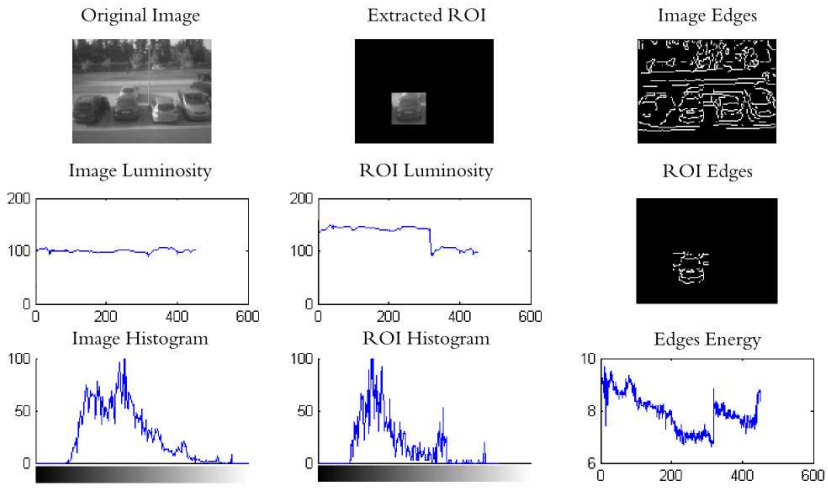


FIGURE 6. Outputs of algorithms of Table 2 when the ROI contains a vehicle.

The visual approach allowed us to infer the principal advantages and limitations of the three algorithms:

- The luminosity value computed for the complete image has a limited correlation with the status of the slot. The value computed for the ROI, instead, presents significant variations associated to the slot condition (Figures 5 - 6, second row, central column). Unfortunately, ambient light variations can not be discerned from modifications on the slot status.
- Histogram proves to be a good metric for the status of the slot but only when computed over the selected ROI. In particular, distribution computed for busy ROIs have sparser distributions than for empty ones (Figures 5 - 6, third row, central column). In our datasets, in fact, ground appears to be more homogeneous than cars. Compared to luminosity value, histograms presents a better behavior when light variations occur. Unfortunately shadows inside the ROI can be interpreted as cars, leading to wrong results.
- Edge Energy E_C varies accordingly the presence or absence of vehicles in the selected ROI (Figures 5 - 6, second row, third column). This parameter presents an useful independence from ambient light but it is unusable when the ground is characterized by significant irregularities.

From these considerations, it results that luminosity value can be a valid NN input for ITS systems operating only in artificially lighted environments (i.e. limited light intensity variance). Both the edge energy and the histogram variables, instead, have a higher tolerance to luminosity variations and thus are still valid candidates as preprocessing transformations. Concerning their limitations, they can be considered negligible in a preliminary design of the system.

Being potentially capable of providing similar performance we adopted the preprocessing algorithm less resource-hunger: the M-bins histogram. The histogram is normalized with respect to the

number of pixels in the ROI. The normalized histograms becomes independent on the number of pixel in the in the ROI image. Therefore, to some extent, the NN trained on ROI with a given scale can be used to operate on ROI with different scales.

Neural Network Topology. After the definition of the NN inputs, it was possible to evaluate the performance levels associated with different NN topologies. To verify the quality of the different configurations, we extracted two independent sets from our data traces: one to be used for training the different NN and the other for validating them (i.e. accuracy measurement).

The basic topology considered is a two/three layers MLP as depicted in Figure 7. The various NN

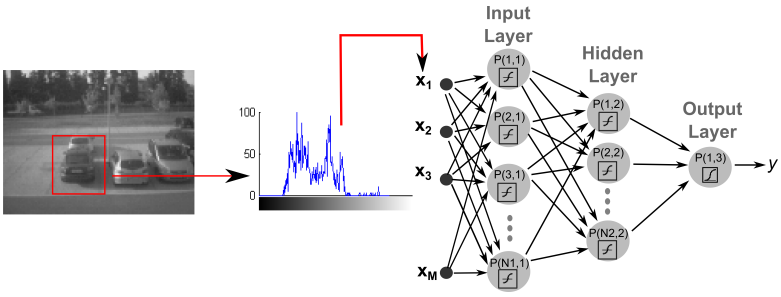


FIGURE 7. Core logic of the park slot monitoring algorithm. From a defined ROI of the original image (left picture, red square) an M-bins histogram is computed and then fed to a 2 or 3 layers MLP network. During the testing phase the number of perceptrons of the input layer N_1 were varied in the interval $[6,18]$ while for the hidden layer their number N_2 in $[0,6]$.

under test differed in four parameters:

- M the size of the input vector, i.e. the number of bins in the normalized histogram;
- N_1 the number of perceptrons of input layer;
- N_2 the number of perceptrons of the hidden layer, might be zero if no hidden layer is used;
- tr the training process.

The input layer and the hidden layer (if present) were configured with the same activation function: Tan-Sig. For the output layer the Log-Sig was adopted as activation function.

Concerning the setup used to evaluate the NN topology, 4 independent ROIs were extracted from a 5 hours acquisition trace (2950 frames). For each ROI, an input vector I and an expected output value O (i.e. ground truth) were created. The O value were manually generated, specifying whether or not the parking slot corresponding to the ROI was empty output or full. Empty and full conditions are coded in O respectively with 0 and 1.

Two of the four ROIs were used to specify the training set $T = \{(T_I, T_O), ...\}$ where T_I is an input vector for a ROI in a given frame and T_O is the expected (ground-truth) output corresponding to T_I. In particular, the input vectors were computed from ROIs with different resolutions over a limited number of frames out of the overall acquisition trace. The two other ROIs, from other frames, were used to generate a validation set $V = \{(V_I, V_O), ...\}$ in the same way of the training one.

Several network topologies have been generated varying the number of perceptrons in the input layer and the number of perceptrons in the output layer. In particular the variations as $N_1 \in [6, 18]$ and $N_2 \in [0, 6]$ were combined to generate different NNs.

Figures 8 and 9 depict the results achieved adopting:

- histogram resolutions of: 256, 64, 32 and 16 bins;
- two different training process: Gradient Descent and Resilient Back-propagation;
- different NN topologies.

The accuracy levels obtained by networks trained with the resilient back-propagation (Figure 9) are significantly higher and with lower variance than the ones achieved using the Gradient Descent algorithm (Figure 8). Concerning the topology, networks appear to be more accurate when the number of neurons is kept limited, with optimal configurations found for values of $N_1 \in [10, 18]$ and $N_2 \in [0, 4]$. Zero perceptrons in the hidden layer means that the layer is not used and the input layers is directly connected to the output one.

Figure 10 depicts the number of weights used by each configuration. This information is important to determine the amount of memory required by each NN configuration. Considering that each weight can be represented with 2 bytes (16 bit fixed point representation) or 4 bytes (32 bit floating point representation), the memory requirement can be expressed in bytes. The optimal trade-off in terms of resources request and accuracy can be reached for histogram resolutions of 96, 64 and 32 bins. With these design choices, at least 4 different neural networks, implementing different input/output transformations, can be stored in a single WMSN nodes equipped with 128KB or RAM and 512KB of flash.

Training Set Generation. The last design question that we had to address was about the training set generation and actual training of the network. We encountered this problem during the IPERMOB project, where we had to configure our algorithms for operating on specific regions of the image (i.e. ROIs).

These ROIs have been designed by implemented a simple mechanism: a human operator in a control room that queries a specific node for the transmission of a full image or part of it. The transmission is performed through the IEEE 802.15.4 channel and in case of loss of packets the operation can be requested again. With a simple Graphical User Interface (GUI) the operator can draw the ROI on top of the image. The parameters characterizing the ROI extension and width are sent back (in a protected manner) to the node that stores them for later use. The simplicity and reliability of this solution motivated the reuse of this approach for this application. In particular, we designed a simple graphical interface for training the network. The user can request a new frame, draw a ROI on it and specify the status of that ROI (i.e. busy or empty). The overall training process takes place off-line. When the training is completed, the weights and the ROI parameters are transfered back to the node.

3.4.2. Evaluation on actual WMSNs. After the definition of the main properties of the network, we obtained all the information required for implementing our ITS system on SEEDEYE boards. The SEEDEYE has been prototyped for the IPERMOB project and shall be presented in the last part of this thesis. The main characteristics of this board are:

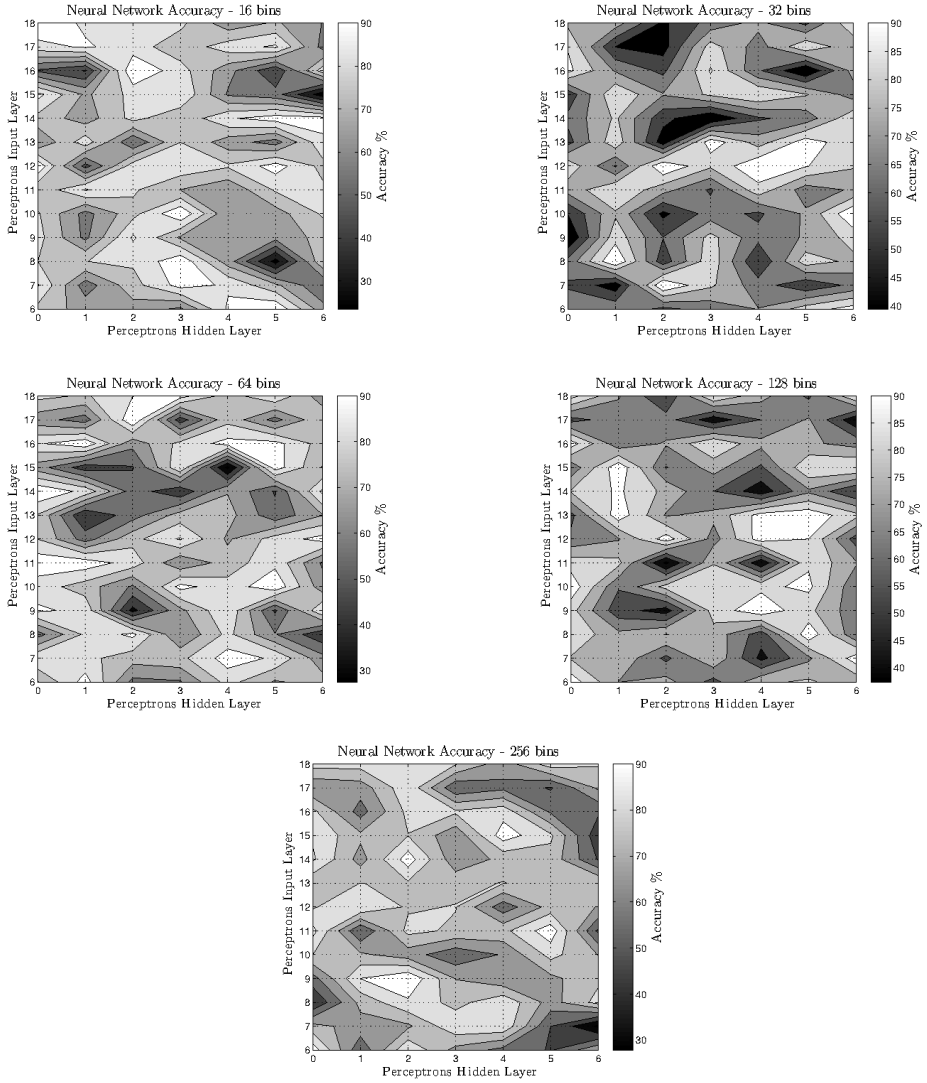


FIGURE 8. Accuracy achieved by system against the validation traces. The training set input is computed from 7 images using histogram resolutions of 256, 64, 32 and 16 bins. Results obtained by network trained with the Gradient Descent algorithm Each plot shows the accuracy reached by networks characterized by different combinations of N_1 (ordinate) and N_2 (abscise). Color bars placed on the right side of images provide a mapping between the color and the accuracy level reached by the specific network configuration.

- A Microchip micro-controller PIC32MX795F512L with a MIPS32 architecure running at 80 MHz, 512KB of flash and 128KB of RAM;
- A VGA CMOS camera HV7131GP from Hynix, operated in gray scale mode;
- An IEEE 802.15.4 transceiver.

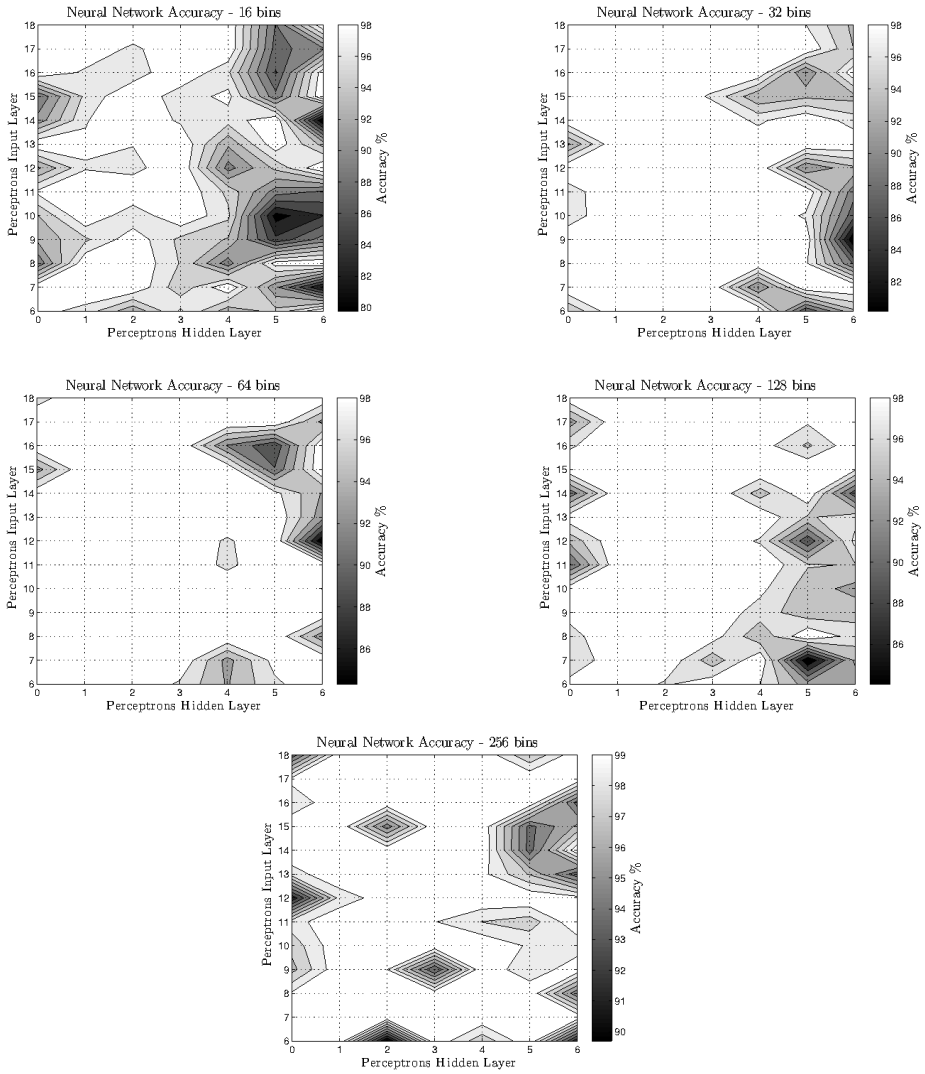


FIGURE 9. Accuracy achieved by system against the validation traces. Results obtained adopting the Resilient Back-propagation algorithm (max number of iterations: 5000, error threshold: 0.00001).

The drivers for acquiring images from the camera, as well as a low complexity version of the IEEE 802.15.4 communication stack were part of the work carried out in the IPERMOB project. The same for the ERIKA Real-Time Operating Systems, that was used to efficiently schedule the various tasks composing the system.

The core application logic related to the NN was developed as an extension of the SEEDEYE platform. The two components, histogram computation routine and neural network, were developed favoring configurability over performances. The function for computing the histogram can operate on any rectangular ROI and can output different resolution histograms. Table 3 shows the amount of time

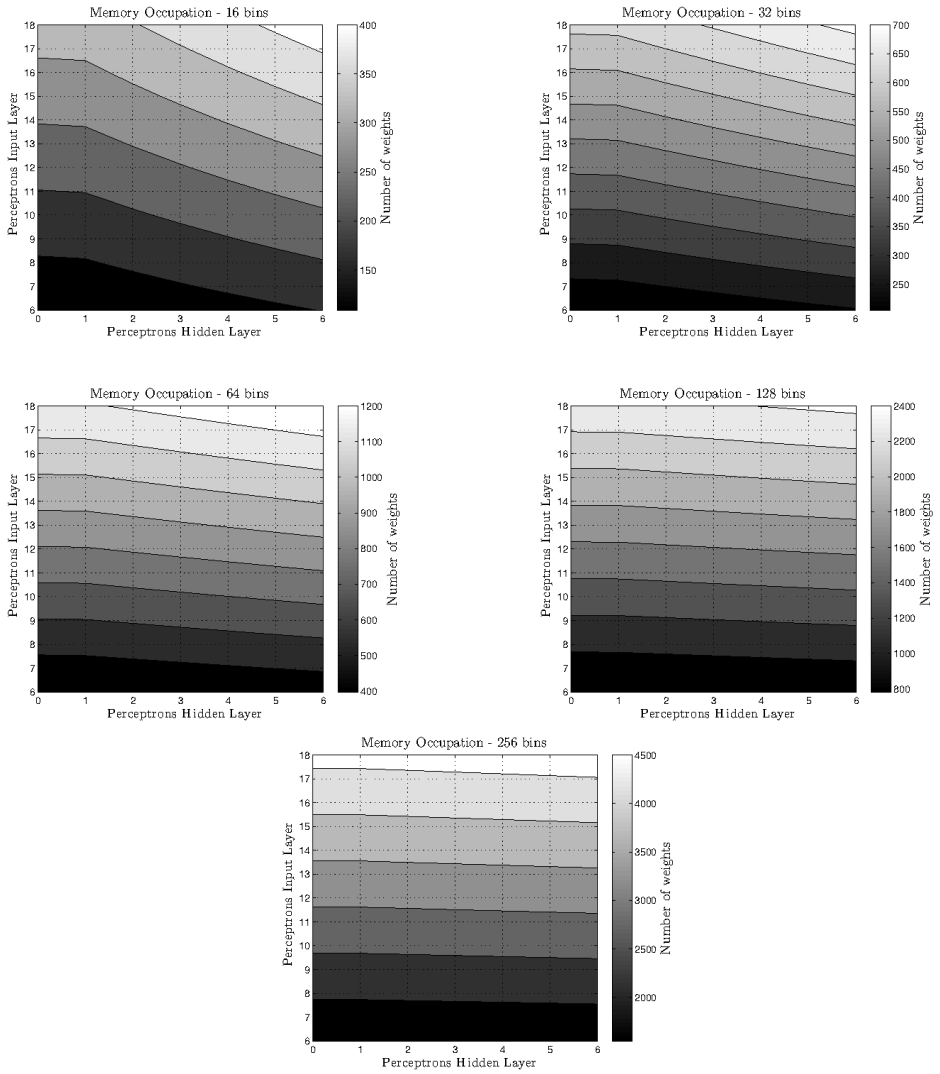


FIGURE 10. Number of weights as function of histogram resolution and network structure.

(in milliseconds) required for computing the histogram on the SEEDEYE board, varying: the size of the ROI and the number of bins.

Concerning the neural network, the design process leaded to a simplified library for MLP networks. The functions exported by the library allow to create, run and debug MLP networks on generic low-power and low-memory micro-controllers. The network topology can be configured either at compile time or at run time. The first case favors a better utilization of memory resources and a faster initialization. The second case introduces more adaptability at the cost of more run time computation resources. In particular, the run-time initialization simplifies the assignment of the network weights permitting an "over the air" update.

	128 bins	96 bins	64 bins	48 bins	32 bins
160x120 pixels	91.55	90.35	89.85	89.08	89.01
100x100 pixels	50.06	49.12	48.43	47.87	47.95
75x75 pixels	29.67	28.78	28.04	27.57	27.22
50x50 pixels	15.08	14.25	13.47	13.05	12.67
35x35 pixels	9.06	8.28	7.50	7.10	6.72

TABLE 3. Histrogram creation execution time in milliseconds for variable number of bins and variable ROI areas. Overall measurement error is in the order of $\pm 0.01ms$.

Training processes, performed generally once, can be run off-line on machines with higher computational power, obtaining coefficients that are then transmitted to SEEDEYE nodes. The benefits of this approach are:

- Execution times are significantly reduced and errors can be more easily detected;
- GUI can used for the training process, simplifying the network configuration process;
- If necessary, other training processes, too complex for embedded devices, can be exploited.

From these premises, we decided to spend more effort on optimizing the network execution than on speeding up the network training. The outcomes of this optimization process are depicted in Table 4. Adopting a 4 bytes resolution (double representation) for the weights, the results generated by the SEEDEYE completely matched with the ones produced by the Matlab™ Neural Network toolbox.

Input Size	#Perc. Input Layer	#Perc. Hidden Layer	#Perc. Output Layer	Exec. Time
96	18	4	1	3.86 ms
96	18	0	1	2.87 ms
64	18	4	1	2.86 ms
64	18	0	1	2.59 ms
48	18	4	1	2.41 ms
48	18	0	1	2.15 ms
32	18	4	1	1.83 ms
32	18	0	1	1.57 ms

TABLE 4. NN execution time for various topologies. Tan-Sig is used as activation function in input and hidden layers, while Log-Sig is adopted in the output layer. Zero hidden perceptrons measn no hidden layer. Overall measurement error is in the order of $\pm 0.01ms$.

The frame acquisition, the core logic and the transmission of the slot status have been implemented through four separated tasks, delegating their activation to the scheduler provided by ERIKA RTOS. Periodically, the scheduler executes the first task (frame acquisition) that contains the logic necessary for

extracting a new image. After the completion of this task, the histogram can be computed and its output can be passed to the neural network. When the neural network terminates its computations, the value representing the network status can be stored as a packet in the transmission FIFO of the transceiver. The last task is completed when the transceiver informs the MCU that the packet has been successfully transmitted. This process, depicted in Figure 11, can be considered concluded after the transmission completion. The time between two executions of these activities, namely T, is a configurable parameter that represents the temporal granularity of the monitoring system. As a reference, in the IPERMOB project, this parameters was set in the range of 60 to 300 seconds.

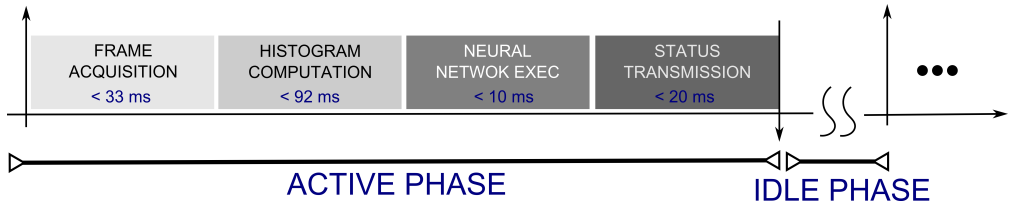


FIGURE 11. Structure of one application cycle. The functionalities are implemented through four tasks, which execution is managed by the RTOS. For each task its worst case execution time (WCET) is shown. The sum of the WCETs can be seen as the amount while the system is in an active state.

It is important to stress that the power consumption of the SEEDEYE board can be drastically reduced adopting a proper energy aware policy. This because the application duty cycle is extremely small compared. Indeed, assuming $T = 60$ seconds, the application duty cycle is:

$$D = \frac{ActivePhase}{T} = \frac{115ms}{60000ms} = 0.0026$$

thus, for more than the 99% of the time the device can be kept in the sleep state. The proper use of the low power features leads to the consumption curve depicted in Figure 12, that, in case of the device powered with a 2800 mAh 9 Volts battery packs, can extend the node lifetime from about 10 hours (i.e. when power awareness policies not adopted) up to 3 months (i.e. with power-aware policies). Considering the wireless communications, performing the car slot status extraction directly on the

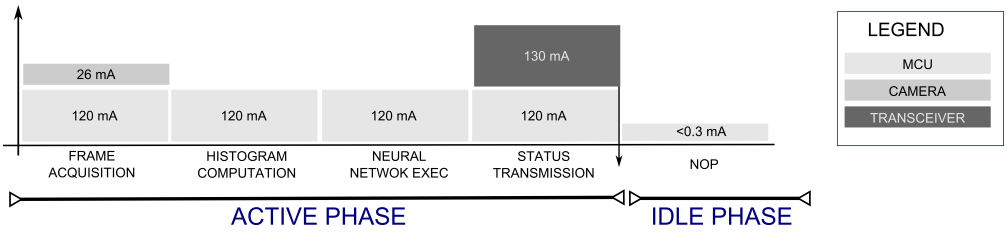


FIGURE 12. Consumption associated with one application cycle adopting the correct Power Aware Policy. During the NOP phase no operations are required.

board significantly reduces the network load. The extent of this reduction is presented in Table 5.

	1 slot monitored	4 slots monitored	8 slots monitored
ROI (70x70 pixels)	49	193	385
ROI (35x35 pixels)	13	49	97
Histogram (96 bins)	2	8	16
Histogram (32 bins)	1	3	6
NN output	1	1	1

TABLE 5. Number of packets to be transmitted for each system update in different scenarios. In the first two rows, the number of packets required for transmitting raw ROIs (1 byte per pixel) is shown. In Row 3 and 4, instead, the volume of packets required for sending histogram (2 bytes per bin) is presented. Last row contains the traffic generated by the system based on NN. Standard IEEE 802.15.4 packets with 102 bytes of payload are considered.

The proposed system can achieve an satisfactory accuracy, it is characterized by an extremely competitive cost, it requires a limited amount of maintenance and it can be easily scaled for covering wide areas. Furthermore, the developed low-power image processing algorithms can be extended to provide additional features, without requiring major changes to the proposed approach.

CHAPTER 4

Distributed Processing: Target Tracking

TRACKING is one of the most advanced and challenging applications of wireless multimedia sensor networks. Traditional approaches for the tracking problem had been formalized considering a *centralized* processing scheme. With the availability of sensor nodes capable of performing computation on-board and exchanging information, new formulations have been proposed based on *distributed* processing.

In particular, distributed target tracking has been recently reformulated more recently considering distributed particle filter techniques for wireless sensor networks. However, the adaptation of distributed tracking filters to WMSNs, i.e. sensor networks with limited field-of-view and large raw-data management constraints, under realistic networking conditions has not yet received enough attention. In this Chapter we discuss an extension to the formulation of the distributed particle filter and enable its operation in realistic scenarios by introducing a next-hop selection mechanism for the aggregation chain and a target hand-over strategy that is capable of handling detection misses and target losses. The filter is tested using a network simulation environment over an area of 6000 m^2 and networks of up to 1000 sensors.

4.1. Background

The problem of tracking is a problem of estimation of a dynamic system that changes over time, using a sequence of noisy measurement of the system. The state-space time-discrete modeling of the dynamic system is used for the formalization of the tracking problem.

The unknown state of the system, or target, is defined by a vector $x = (\xi_1, \xi_2, ..., \xi_S)$, where each component is an internal unknown variable of the state-space model of the system (e.g. position, velocity). An noisy observation of the state is performed through measurement and is defined by the vector $z = (\zeta_1, \zeta_2, ..., \zeta_T)$.

The dynamic evolution of the target is described as follows:

- x_k State of the target at time $k \in \mathbb{N}$ (discrete-time);
- z_k Measurement of the target at time $k \in \mathbb{N}$ (discrete-time);
- $\{x_k : k \in \mathbb{N}\}$ Sequence (or trajectory) of states of the target, modelled as first-order Markovian process;
- $\{z_k : k \in \mathbb{N}\}$ Sequence of measurements of the target state.

The target state evolution can be defined by the generally non-linear function

$$x_k = F_k(x_{k-1}, v_{k-1}) \quad , \tag{14}$$

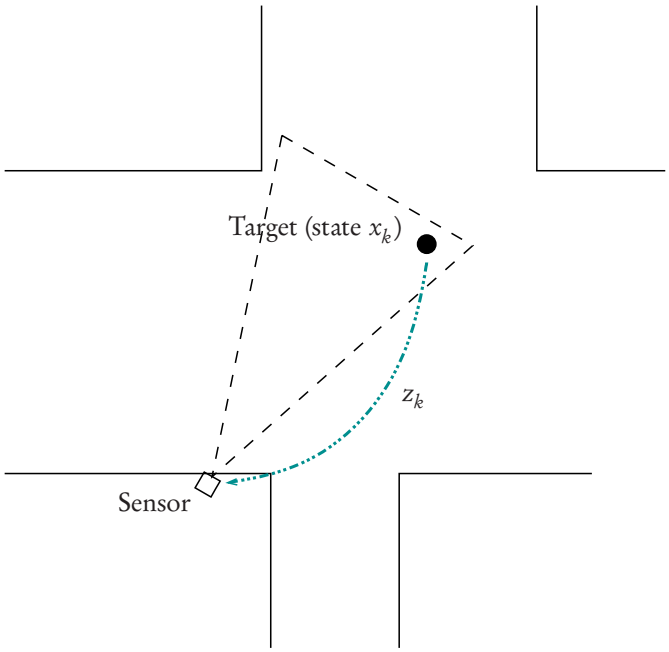


FIGURE 1. Target tracking problem. The state x_k of the target at time k is unknown and has to be estimated from the noisy measurement z_k obtained by the sensor.

the state measurement can be defined by the generally non-linear function

$$z_k = H_k(x_k, n_k) \quad , \tag{15}$$

where

$\{v_{k-1} : k \in \mathbb{N}\}$ i.i.d. sequence of process noise generally non-Gaussian;
$\{n_k : k \in \mathbb{N}\}$ i.i.d. sequence of measurement noise generally non-Gaussian;
$F_k(\cdot)$ State transition function;
$H_k(\cdot)$ Measurement function;

The state transition function F_k describes the evolution of system state to a given degree of uncertainty represented by the process noise v_{k-1}. The measurement function describes the sensor behavior that, given a system state, produces an observation thereof with a given uncertainty associate to the measurement noise n_k. The objective of target tracking is to estimate the state of the system $\bar{x}_k$ at time k based on the previous (estimated) state x_{k-1} and the sequence of measurement $\{z_{1:k}\}$[1].

4.1.1. Bayesian Formulation. The Bayesian formulation of tracking is directly derived from the previous definition. Essentially, the state transition function (Equation 14) and the measurement function (Equation 15) can be rewritten in terms of probability density function (PDF). The state transition function $F_k(\cdot)$ can be expressed as the conditional probability of the target being in state x_k given the

[1]Let us use the following convention for subscripts to indicate temporal sequences: $z_{1:k} = z_1, z_2, ..., z_k$

state x_{k-1}, thus defining the *transition PDF* as

$$F_k \;\mapsto\; p(x_k|x_{k-1}) \quad . \tag{16}$$

The measurement function $H_k(\cdot)$ can be expressed as the conditional probability of the measurement z_k given the state x_k, thus defining the *likelihood PDF* as

$$H_k \;\mapsto\; p(z_k|x_k) \quad . \tag{17}$$

The objective of target tracking is to estimate the state of the system at time k given the state at $k-1$ and the sequence of measurement from 1 to k. With the Bayesian formulation the objective is to reconstruct the *posterior* PDF

$$p(x_k|z_{1:k}) \quad . \tag{18}$$

The problem can be solved using a recursive formulation based on two steps:

(1) *prediction*: the previous (estimated) state is used to predict the current state from the transition function (14); in terms of probabilities, the prior PDF is calculated using Equation 16;
(2) *update*: the current available measurement is used to update the predicted current state using the measurement function (15); in terms of probabilities, the prior PDF is updated in the posterior PDF using Equation 17.

The mathematical formulation follows. Let us assume to know:

$p(x_0|z_0) \equiv p(x_0)$ Initial target state, i.e. the initial posterior PDF, where z_0 is the empty measurement;

$p(x_{k-1}|z_{1:k-1})$ Previous posterior PDF (at time $k-1$);

The prediction stage calculates the prior PDF at time k through the Chapman-Kolmogorov equation:

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \tag{19}$$

where the first-order Markovian process assumption has been used to simplify $p(x_k|x_{k-1}, z_{1:k-1}) = p(x_k|x_{k-1})$. The update stage calculates the posterior PDF at time k from the available measurement z_k through the Bayes' rule:

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_{k-1}|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{20}$$

where the normalization constant at the denominator is

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k \quad ,$$

derived from the likelihood function (17).

The recursive relation between (19) and (20) represents the theoretical optimal solution to the Bayesian formulation of the tracking problem. Analytic solution to the general formulation is not possible, since the exact evaluation of the integrals is not possible. Two approaches exists to solve the Bayesian problem in practice:

- Optimal Algorithm: using some restrictions on the problem formulation, analytical optimal solutions can be derived;
- Suboptimal Algorithm: restrictions are not given on the problem formulation, but the solution is an approximation of the theoretical optimal solution.

The most popular optimal algorithm for Bayesian tracking is the Kalman Filter (KF) [**Kal60**]. The KF operates under the assumptions that

- the process noise v_k and the measurement noise n_k can be drawn from Gaussian distributions (one for each noise);
- the state transition function F_k and the measurement function H_k are linear functions.

In such conditions, an analytical solution to the Bayesian tracking problem can be derived, and it is proven that the KF is an optimal algorithm.

In many cases the linearity and Gaussianity assumptions of the KF do not hold. Approximation methods have been proposed to address this issue. The Extended Kalman Filter (EKF) relaxes the assumption on linearity. Taylor expansion is used to produce a linear approximation of the non-linear transition an measurement functions.Gaussian approximation is used for the process and measurement noises. A further extension of the EKF is the Unscented Kalman Filter, which uses an unscented transform to better approximate the non-linearity of the system and the Gaussian parameters.A different approach, based on Monte Carlo methods, is the *Particle Filtering*. In this case the problem is not approximated with a linear and Gaussian one. A sequential Monte Carlo approach is used to approximate by samples the true PDFs of the Bayesian formulation. In the remainder of this Chapter we will focus our attention on the latter approach.

4.1.2. Particle Filter. The basic overview of the Particle Filter (PF) is given in this Section. For a thorough definition the reader might want to refer to [**AMGC02**].

The idea behind the PF is that probability density functions can be approximated by Monte Carlo samples of the same density. Essentially, the idea is to represent the posterior PDF (18) with a set of

- *particles*: random samples (of the state space);
- *weights*: a weight associated to each random sample.

The particle represents a possible realization of the state of the target, while the weight represents how such realization is plausible. For instance, assuming to know the exact state of the target, the particle that is closest to that state shall have the highest weight; if the particle is exactly equal to the target state its weight is 1. To some extent, we can say that a weight represent the probability of the particle to represent the target state.

Given the above definition, the PDF is then approximated by the linear combination of the particles and the weights. If the number of particles (and weights) increases, the approximation becomes more accurate; theoretically, if this number grows to infinity the linear combination converges to the true PDF. More formally, considering:

$x_k^{(i)}$	i-th particle at time k;
$w_k^{(i)}$	i-th weight at time k associated to $x_k^{(i)}$;
$\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{P}$	Particle (and weight) set, where P is the number of particles (and weights) and the weights are normalized $\sum_i w_k^{(i)} = 1$;

given the system state x_k, the posterior PDF at time k can be approximated as

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{P} w_k^{(i)} \delta(x_k - x_k^{(i)}) \quad . \tag{21}$$

Sequential Importance Sampling. The algorithm to implement the recursive predict-and-update steps to estimate the target state is known as Sequential Importance Sampling (SIS). The problem of Bayesian tracking is that the posterior PDF is not known and has to be estimated. The SIS is based on the idea that an alternative PDF proportional to the posterior PDF $\pi(\cdot) \propto p(\cdot)$. Assuming that particles are drawn from a proposal distribution $q(\cdot)$ known as *importance PDF* such that

$$\begin{array}{l} x^{(i)} \sim q(x) \\ w^{(i)} \propto \frac{\pi(x^{(i)})}{q(x^{(i)})} \end{array} \quad \forall i = 1, ..., P \quad .$$

Owing to the proportionality between $\pi(\cdot)$ and $p(\cdot)$, the approximation defined by Equation 21 is valid with weights defined as

$$w_k^{(i)} \propto \frac{p(x_k^{(i)}|z_{1:k})}{q(x_k^{(i)}|z_{1:k})} \quad .$$

From those assumptions it can be demonstrated that the recursive Bayesian solution is approximation by SIS algorithm as

(1) *initial condition*: In the initial particle set all particles are set to the initial state, since $p(x_0)$ is known, and the weights are all initialized to the same value:

$$\{x_0^{(i)}, w_0^{(i)} \ : \ x_0^{(i)} = x_0 \wedge w_0^{(i)} = 1/P\} \quad \forall i = 1, ..., P \quad ;$$

(2) *prediction*: P particles extracted from the importance function:

$$x_k^{(i)} \sim q(x_k|x_{k-1}^{(i)}, z_k) \quad \forall i = 1, ..., P \quad ;$$

(3) *update*: Weights are updated according to:

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{p(z_k|x_k^{(i)}) p(x_k^{(i)}|x_{k-1}^{(i)})}{q(x_k^{(i)}|x_{k-1}^{(i)}, z_k)} \quad s.t. \sum_{i=1}^{P} w_k^{(i)} = 1 \quad . \tag{22}$$

It can be shown that in (21) the particle approximation converges to the true posterior PDF $P(x_k|z_{1:k})$ for $P \to \infty$.

Sequential Importance Resampling. An alternative algorithm for particle filtering is the Sequential Importance Resampling, that operates under the following assumptions:

- The transition function (14) is known and samples can be drawn from the process noise v_k; in terms of Bayesian formulation, samples can be drawn from the prior PDF (16);
- The measurement function (15) is known; in terms of Bayesian formulation, point-wise evaluation of the likelihood (17) is possible (at least up to proportionality).

The SIR algorithm can be derived from the SIS algorithm by selecting the prior PDF as the importance function $q(\cdot)$. Moreover, an *resampling* step is performed at every iteration of the SIR. The resampling "transfer" the information about the plausibility of the particle from the weight to the particles. This is done by defining a new particle set where the new particles are obtained by replicating particle from the original set. In particular, a particle is replicated more times proportionally to the weights in the original set. The weights in the new set are all equals to $1/P$. Now, considering the prior PDF as the importance function

$$q(x_k^{(i)}|x_{k-1}^{(i)}, z_k) = p(x_k|x_{k-1}^{(i)})$$

and substituting in (22), and considering the resampling for every iteration, the SIR algorithm can be defined as:

(1) *initial condition*:

$$\{x_0^{(i)}, w_0^{(i)} \ : \ x_0^{(i)} = x_0 \wedge w_0^{(i)} = 1/P\} \quad \forall i = 1,...,P \quad ;$$

(2) *prediction*: P particles extracted from the prior PDF:

$$x_k^{(i)} \sim p(x_k | x_{k-1}^{(i)}) \quad \forall i = 1,...,P \quad ;$$

(3) *update*: Weights are updated according to:

$$w_k^{(i)} = p(z_k | x_k^{(i)}) \quad s.t. \sum_{i=1}^{P} w_k^{(i)} = 1 \quad . \tag{23}$$

Although the assumptions of the SIR algorithm are quite restrictive with respect to the SIS algorithm, the SIR provides a simple solution in terms of weight update and in terms of importance sampling.

4.1.3. Multi-Sensor Tracking. The formulation of the tracking problem has to be extended a to consider the distributed nature of wireless multimedia sensor network, where measurement from spatially distributed sensors need to be used to for the state estimation. *Multi-sensor tracking* aims at estimating the target state, x_k, at each discrete-time step k, given a set of measurements (observations) obtained from N sensor-nodes, each producing a local measurement z_k^n, with $n = 1...N$.

The formulation formerly introduced in this Chapter still holds when considering the (global) measurement vector z_k as the composition of the local measurements:

$$z_k = (z_k^1, z_k^2, ..., z_k^N) \quad .$$

Early approaches for multi-sensor tracking use a *centralized* filter, where all the measurement z_k^n are sent to a central node that holds the vector z_k and executes a classic single-node tracker. This approach is not scalable and it is not feasible in bandwidth-constrained wireless networks, where the physical communication link is potentially shared by all the nodes in the network. A solution to this problem is the use of a decentralized or distributed approach [**TC11a**]. In *decentralized* tracking, the network is partitioned into clusters according to some sensing relationship between the nodes [**MP09**]. For each cluster, a cluster-head is responsible of performing centralized tracking. This solution is effective when nodes observing the same target belong to the same cluster.

In *distributed* tracking, the estimation is cooperatively performed through collaboration protocols. The network has a common "distributed" knowledge of the posterior PDF $p(x_k|z_{1:k})$. This can be achieved through two different information exchange paradigms, namely consensus and token passing. The *consensus-based approach* is equivalent to a data dissemination mechanisms, but is more efficient in terms of communication bandwidth [**OsFM07**]. *Token passing* is a sequential aggregation mechanism: each node receives a partial aggregate, creates a new aggregate with its local information and sends the new aggregate to the next node.

Early proposals for multi-target tracking were operating under the linearity and Gaussianity assumptions, essentially implementing extension of the Kalman Filter to deal with distributed sensors. Originally decentralized approaches for Kalman Filtering have been studied (e.g. [**GGGA07**, **MP09**]).

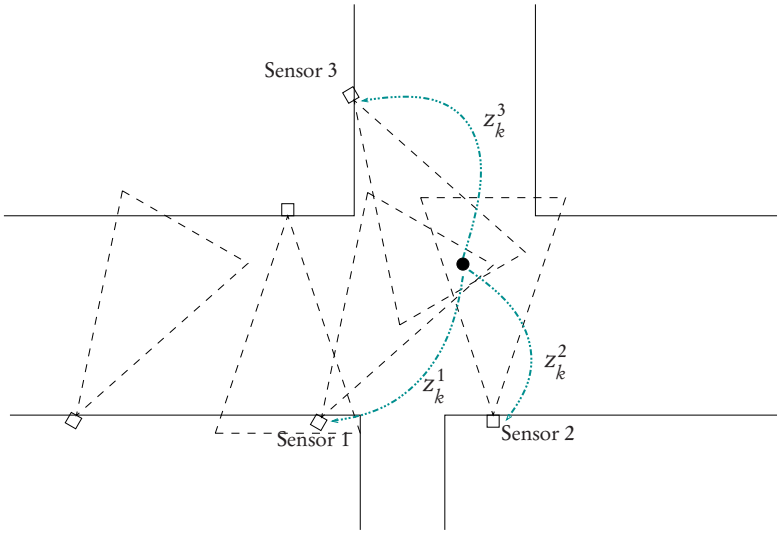


FIGURE 2. Multi-sensor target tracking problem. The state x_k of the target at time k is unknown and has to be estimated from the set of noisy measurements $z_k = \{z_k^1, z_k^2, z_k^3\}$ obtained by spatially distributed sensors.

More recently, approaches based on consensus algorithm have been proposed for truly distributed process. Those solutions are also known as Kalman Consensus Filters (KCFs). In this case each node runs a modified version of the classic Kalman Filter, where a message exchange mechanisms is introduced to obtain a common agreement on the final predicted state on all the nodes [**OSS08**, **SSC09**, **SKS+10**].

Multiple target tracking. An important issue that should be considered in large-scale systems such as WMSNs is the presence of multiple targets at the same time. In this case, the problem of target tracking needs reformulation in order to account for the possible interaction between the multiple targets. This leads to the definition of Multi-Target Tracking (MTT) [**LCR07**]. In order to give an idea of the MTT problem, let us consider the example in Figure 3. The essence of MTT is the *data association* problem. The sensors produce a measurement for each target. When the targets (and their states) are "well separated", it is trivial to associate the measurement to the corresponding target state. In this case, multiple single target tracking algorithm can operate as described before. When the targets are "close" to each other, the association of measurement to the target state might not be straightforward. In this case, multiple hypotheses (about the association) might need to be considered, e.g. for target a we should consider both $p_(x_k^a|z_k^1)$ and $p_(x_k^a|z_k^2)$. The approach of enumerating all the possible hypotheses for association is known as Multiple Hypotheses Tracking (MHT) [**Rei79**] This approach has the drawback that the number of hypothesis increases exponentially with the number of targets. Alternatively, Joint Probabilistic Data Association Filter (JPDAF) has been proposed by Fortmann, Bar-Shalom, and Scheffe [**FBSS80**]. JPDAF scales linearly with the number of the targets by taking into account a linear combination of the measurements when updating the target states. Both MHT and JPDAF might be combined with multiple (parallel) single-target tracking in order to operate only in a region of significant interaction. Essentially, when targets are far away, a single-target tracking is

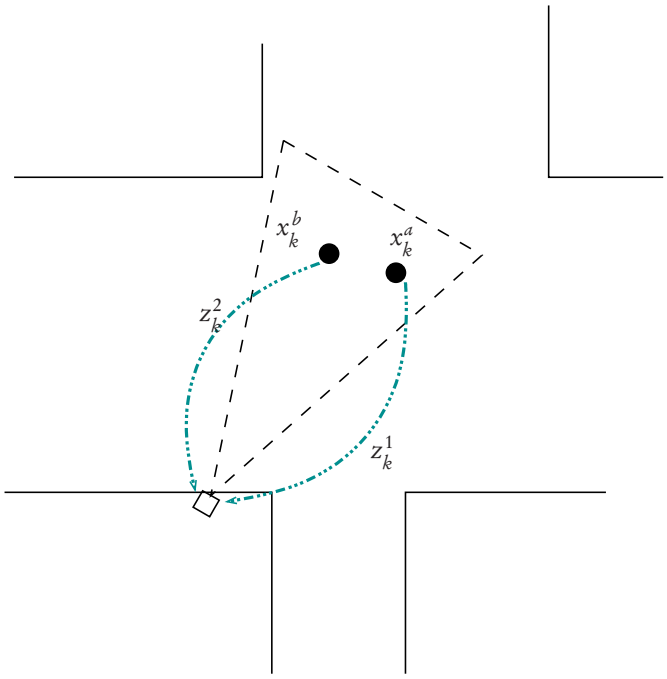


FIGURE 3. Data association problem in multi-target tracking. When the two target interact (i.e. getting close to each other) it is uncertain whether z_k^1 is associated to x_k^a (and z_k^2 to x_k^b) or vice versa.

executed; when the targets enter a region of interaction, an MTT approach is adopted. In the remainder of this Chapter we shall focus on the problem of single-tracking, assuming that the data association is solved by a possible target detection algorithm.

4.1.4. Distributed Particle Filtering. In order to relax the assumption on linearity and Gaussianity imposed by the the Kalman Filter, approaches based on Particle Filtering have been developed. Distributed Particle Filters (DPFs) for WSNs have been proposed by Coates in [**Coa04**]. Each node of the network executes a *local* Particle Filter, a slightly modified version of the classic PF. In particular, each node has a particle set and exchanges some partial information with other nodes according to a specific protocol with the goal of keeping all the particle sets consistent in all the nodes, i.e. an agreement on the particle set is reached by the nodes at each time step. Although Coates has the merit of formulating the problem of the DPF for the first time, its paper remains general and unclear about the type of information that nodes should exchange.

Solutions based on the definition of a *common posterior distribution* were proposed in [**SHR05**, **Gu07**, **HDH09**]. Each node executes the PF by drawing particles from an importance function chosen as the common posterior distribution at the previous iteration step. The goal is therefore to have the same posterior in all the nodes. However, exchanging the particle set among nodes to approximate the posterior probability is not efficient and does not scale with the size of the target's state and the number of particles. To reduce the amount of data exchanged, i.e. the particles and the weights, independence

from the number of particles is obtained by approximating the posterior with a *Gaussian Mixture Model* (GMM).

In [**SHR05**] the authors proposed two approaches based on a two-level hierarchical architecture in which nodes with correlated measurements are organized in sensor cliques, and then clique-heads are organized in clusters observing the same target. Data within a clique are sent towards the clique-head and the distributed algorithm takes place at the cluster level, i.e. among clique-heads. The first solution in [**SHR05**] is based on a spatially sequential approximation of the posterior with a GMM. The mechanism is based on a chain similar to that in [**Coa04**], but, rather than the likelihood, the posterior is approximated and transferred. The second solution in [**SHR05**] uses a parallel approach and each node derives its own GMM for the posterior. Then, all the GMMs are exchanged among the nodes and a k-mean algorithm is used to create the final GMM approximation of the posterior. A similar sequential solution is used in [**HDH09**]. The two have a slightly different formulation of the weight update function and the GMM parameter generation, although the underlying idea is the same. The authors also propose an alternative "look-ahead" technique to design a more accurate proposal distribution for the importance sampling process, making use of a two-step Gaussian sum filter. Finally, the approach presented in [**Gu07**] is based on consensus algorithms. The problem is to have the same GMM approximation in all the nodes. The global parameters of the GMM are obtained by averaging local statistics of the nodes using a consensus algorithm.

4.2. The distributed tracking algorithm

The algorithms discussed in the previous section are defined at a high level of abstraction, without considering issues related to their applicability to realistic networking scenarios. To address issues related to the distributed implementation in the context of a realistic network environment and line-of-sight sensors (cameras), in this section we discuss our extensions for realistic WMSNs of the algorithms presented in [**SHR05**, **HDH09**] .

4.2.1. Model. Let us consider for now that all the nodes, N, in the network observe the target, i.e each node has a measurement and has to be involved in the DPF iteration. A set of P particles and its relative set of weights is held by each node. The nodes' observations are synchronized, i.e. the k-th measurement is generated at the same time in each node. The DPF algorithm is based on a sequence of aggregation steps, the *aggregation chain*. The aggregation chain is performed every time a new measurement is available. Let the index k indicate a specific instance of the aggregation chain, i.e. a tracking step; whereas the index h indicates the steps of the aggregation chain, i.e. the *aggregation steps*.

In order to apply DPF to FOV-based sensors (cameras), we have to address the algorithmic constraints introduced by their limited FOV. The "content" of the measurement z_k^n is generated from the detection process performed by node n at time k. If the target is not in the node's FOV, such content cannot be generated. This problem, here referred to as *detection miss*, is not considered in the original algorithm and has an impact in the cooperative protocol. Since the target can exit the FOV of a node, we will introduce a mechanism to manage the *target hand-over*, i.e. when a target that was detected in the previous step is no longer detected in the current one. This mechanism is dependent on the role of the node in the aggregation chain. Note that we are not considering here false positive detections.

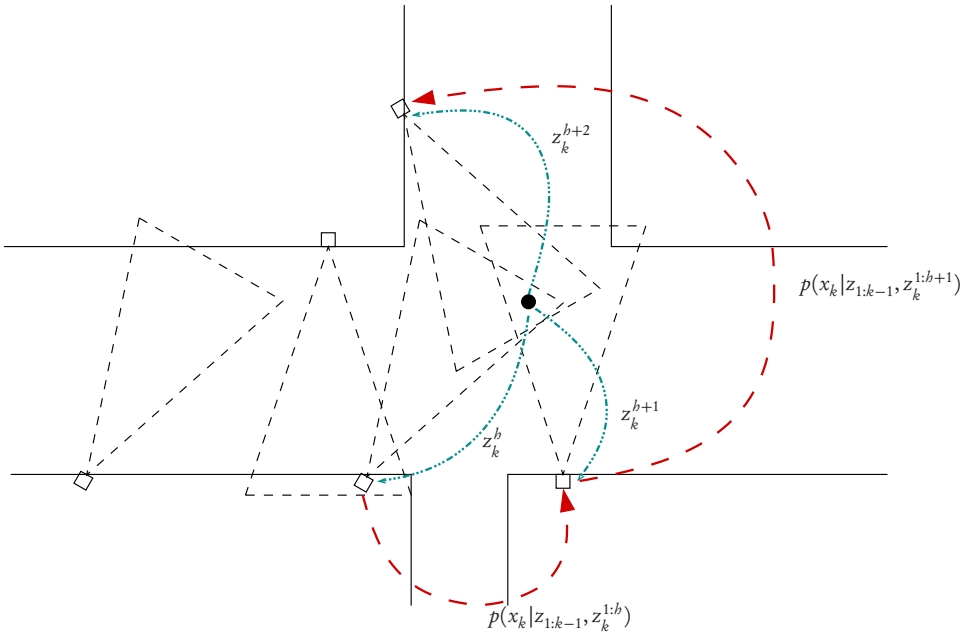


FIGURE 4. Example of aggregation sequence using GMM-PP

Partial Posterior PDF. The basic idea of the DPF is that each nodes refines the estimation of the posterior PDF making use of its local measurements. In this way, the measurement is not exchanged between the nodes and only the estimation is transfered. Because the posterior PDF at the local node is created using a subset of the entire measurement vector z_k, we refer to this posterior as *partial*. In particular, the *Partial Posterior* (PP) is the posterior PDF obtained at the intermediate steps of the aggregation chain. In order to reduce the amount of information exchanged between the nodes, the PP is approximated with a Gaussian Mixture Model. Figure 4 shows the token passing mechanism used to exchange the (GMM-) PP. We denote the GMM approximation of the partial posterior as GMM-PP:

$$p^h_{\text{GMM-PP}} \approx p^h_{\text{PP}} = p(x_k|z_{1:k-1}, z_k^{1:h}) \quad .$$

Notice that $z_k 1:k-1$ is the sequence of the full measurement vectors, while $z_k^{1:h}$ is the partial measurement vector at time k that includes the measurement from node 1 to node h.

4.2.2. Algorithm. At *initialization*, P particles $x^{(i)}$ are set equal to the initial target state x_0 and all the weights $w^{(i)}$ are set equal to $1/P$. One of the nodes is selected to be the first node starting the aggregation process.

Node Selection. The mechanism *to select the next node* in the aggregation chain is an important component of the algorithm. The node currently holding the token, n_h, queries the n_{h+1} node that is closer to the (partially) estimated target position $\hat{x}_k^h$. The queried node can refuse if it has already been included in the aggregation process or if it determines that its local measurement is not accurate enough [**HDH09**]. Unlike the original approaches, in our case the nodes are aware of the FOV information of the cameras that could be in the same active set (its logical neighborhood). However, it

could be also sufficient to know only which are the cameras of the node's neighborhood. In this case the node shall query all the nodes in the neighborhood without any preference or filter related to the estimated target's state. In our implementation the n_{h+1} is selected by considering the node that should be observing the same target (according to $\hat{x}_k^h$) at the same time. The order in which the neighbor nodes are queried is related to the expected proximity of the target to the center of the neighbor's FOV.

Active Nodes. Another important limitation of the original algorithm is that all the nodes in the network must observe the target at any time, which in case of cameras would mean that the FOVs of all the nodes should overlap and that a target only moves inside the overlapping region. To remove this unrealistic assumption we define a subset of *active nodes* that, for a given tracking step, are able to generate a detection of the target.

First Node Iteration. The **first node**, n_1, starts the iteration for the current tracking step k. The node n_1 has the global posterior probability from time $k-1$ (and does not receive a PP at time k). This could be possible, for instance, if the first node in the current tracking step is also the last one of the previous one. Node n_1 performs the posterior prediction and measurement update steps in order to create the first PP from the previous posterior.

The particle set $\{x_{k-1}^{(i)}, w_{k-1}^{(i)}\}_{i=1}^{P}$ from the previous tracking step describes the previous posterior probability. This particle set is re-sampled, generating the set

$$\{\bar{x}_{k-1}^{(i)}, \bar{w}_{k-1}^{(i)}\}_{i=1}^{P} \quad .$$

The *prediction* step is obtained by drawing a new particle from the state transition function for each re-sampled particle:

$$x_k^{(i)} \sim p(x_k|\bar{x}_{k-1}^{(i)}) \qquad \forall i = 1,\ldots,P \quad . \tag{24}$$

The *measurement* update uses the local measurement z_k^1 to update the weights for the new particle set through the likelihood function,

$$w_k^{(i)} = \frac{f(z_k^1|x_k^{(i)})}{\sum_{j=1}^{P} f(z_k^1|x_k^{(j)})} \qquad \forall i = 1,\ldots,P \; . \tag{25}$$

The new particle set $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{P}$ represents the PP in the first node. The GMM-PP is then created and sent to the next node. The creation of the GMM-PP is based on unsupervised algorithm that uses expectation maximization and minimum description length order estimation [**Bou97**].

In case of a detection miss in the first node of the aggregation chain, the algorithm shall not start until a new first node is found. A solution could be to restart the first-node selection mechanism, but this will require several iterations to converge. In the proposed solution the first node with a detection miss selects the next node, according to the standard procedure, to which it forwards the previous GMM-Posterior and a notification of its failure. The receiving node is now in charge of starting the aggregation chain. If this node fails too, the same procedure is repeated until a new first node is found or a target loss happens.

Intermediate Nodes. The **intermediate nodes**, n_h with $h = 2,\ldots,N-1$, i.e. all nodes but the first and the last one, receive a PP as input and send a new PP as output. Each n_h receives the (GMM-)PP from the previous aggregation step $(h-1)$, updates the PP with its local measurement z_k^{h-1}, creates a GMM approximation of the PP and sends it to the next node n_{h+1}.

The received GMM-PP is used as importance function $q_k(x_k)$ for the Sequential Importance Sampling algorithm. A new particle set is created by drawing P particles from the received GMM-PP:

$$x_k^{(i)} \sim p_{\text{GMM-PP}}(x_k|z_k^{1:h-1}) \qquad \forall i = 1,\dots,P\ . \tag{26}$$

The relative weights are calculated according to the importance sampling rule

$$\tilde{w}_k^{(i)} = \frac{p(z_k^1|x_k^{(i)})p_{\text{GMM-PP}}(x_k|z_k^{1:h-1})}{p_k(x_k^{(i)})} \qquad \forall i = 1,\dots,P$$

$$w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{j=1}^{P}\tilde{w}_k^{(j)}}\ .$$

However, since the importance function is equal to the incoming GMM-PP, the weight calculation is simplified as in (25):

$$w_k^{(i)} = \frac{q(z_k^h|x_k^{(i)})}{\sum_{j=1}^{P} p(z_k^h|x_k^{(j)})} \qquad \forall i = 1,\dots,P\ . \tag{27}$$

The new PP is then approximated with a GMM-PP that is sent to the next node n_{h+1}.

In case of a detection miss in an intermediate nodes, the problem is solved by skipping the local processing (since the detection is not available) and forwarding the incoming GMM-PP to the next node directly.

Last Node. The **last node** of the aggregation chain, n_N, reconstructs the global posterior probability and performs the estimation of the target's state. The global posterior is reconstructed following the same mechanism described for the intermediate nodes. In the last node, the PP resulting from Equations (26) and (27) is also the global posterior,

$$p_{\text{PP}}^N = p(x_k|z_k^{1:N}) = p(x_k|z_k)\ .$$

The target's state estimation is then possible in n_N as

$$\hat{x}_k = \sum_{i=1}^{P} w^{(i)}x^{(i)}\ . \tag{28}$$

This last node will act as first node in the next tracking step [**HDH09**]. Note that, however, the algorithm could be easily modified by selecting a different first node, provided the final global posterior is transferred from $n_{N,k}$ to $n_{1,k+1}$.

A detection miss at the last node does not require particular procedures. The node will produce the estimate considering the incoming GMM-PP as the final GMM-posterior.

Target Loss. If all the nodes appear to have lost the target, a *target loss* condition happens. This condition is due to the target hand-over and takes place when no first node can be selected from the candidate neighborhood. As soon as the target is lost, the node detecting this condition notifies all the nodes in the network with a broadcast message. This message includes the last known posterior. When the target is visible again, the node currently observing the target shall start the first node selection process. Note that, owing to imprecisions of the state estimation, it might be possible that the target was visible by other nodes not originally included in the neighborhood. In this case the tracking will immediately restart.

4.3. Validation

In this section, we evaluate the performance with (and without) limited resources of the distributed particle filter we presented in the previous section. Without resource limitations, the target's state estimation is immediately produced for each new observation by the distributed tracking algorithm. With resource limitations, network delays and packet loss may reduce the number of estimations, compared to the total number of available measurements.

Let K_{tr} be the number of estimations and let an estimation be available after a certain delay $d(k)$. The average estimation delay, $\overline{D}$, is defined as

$$\overline{D} = \frac{1}{K_{tr}} \sum_{k=1}^{K_{tr}} d(k) \,, \tag{29}$$

whereas the network estimation efficiency, E, is defined as

$$E = \frac{K_{tr}}{K} \,. \tag{30}$$

E quantifies the total number of estimations (detected events) over the total number of observations (all the events).

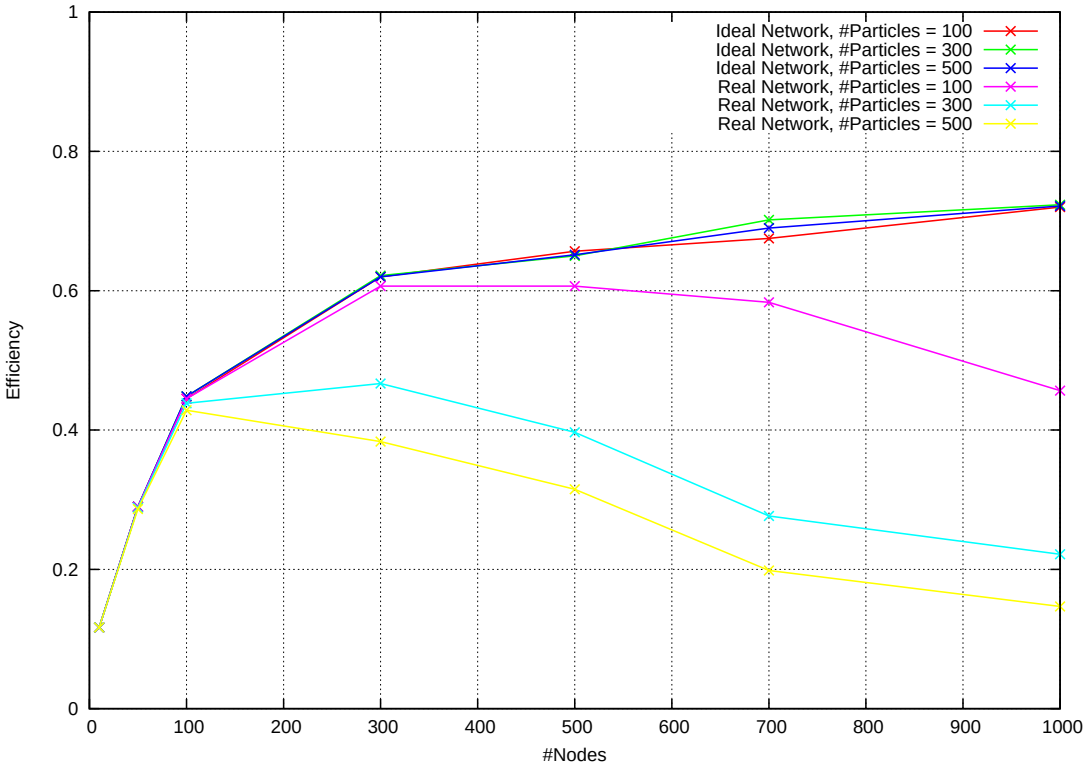
We define an area of $100m \times 60m$ surveilled by the N cameras having a FOV of $10m \times 6m$ and with a top-down view. The FOV of the camera is calculated considering angles of view of cameras and assuming a distance of $6m$ from the ground plane. The camera are positioned according to a random uniform distribution. The sampling period is $T_s = 1s$ and there are $K = 600$ observations per run. The network is based on the T-MAC protocol [**vDL03**], configured with the request-to-send/clear-to-send and acknowledged-transmission mechanisms and with a number of retransmissions set to 10. The bandwidth is $BW = 250$ kbps. The target motion is a linear motion with a random walk that follows a zero-mean normal distribution with a variance of $0.3m$. We reproduce the realistic networking environment using a network simulator engine based on Castalia [**cas**] and Omnet++ [**www10d**]. Due to the probabilistic nature of the filter, 100 simulation runs, each of 10 minutes, are generated by varying the number of nodes in the network from $N = 10$ to $N = 1000$. The average values with standard deviation are considered in the analysis.

Figure 5(a) compares under ideal (i.e. no delays) and realistic networking conditions the efficiency of a traditional DPF (0-GMM), when the whole particle set is exchanged across nodes, while varying the number of sensors from 10 to 1000 and the number of particles from 100 to 500. It is possible to notice that in case of ideal networking conditions, the efficiency improves as the number of nodes increases, as deploying more nodes gives a higher degree of coverage of the surveillance area. However, *when realistic networking conditions are modeled, the performances of the system are sensibly different*. While E is still decreasing in the first part of the curves ($N = 10, 50, 100$), for larger values of N it increases with the number of nodes. This loss of performance is related to the network delays, which are clearly visible in Figure 5(b). Because of the limited communication bandwidth, the delay required to completely perform a tracking step increases with N. Indeed, more and more nodes have to be involved in the iteration mechanism of the aggregation process, thus delaying the time when the estimation is ready. The larger the number of particles, the steeper the slope of the $\overline{D}$ when increasing the number of

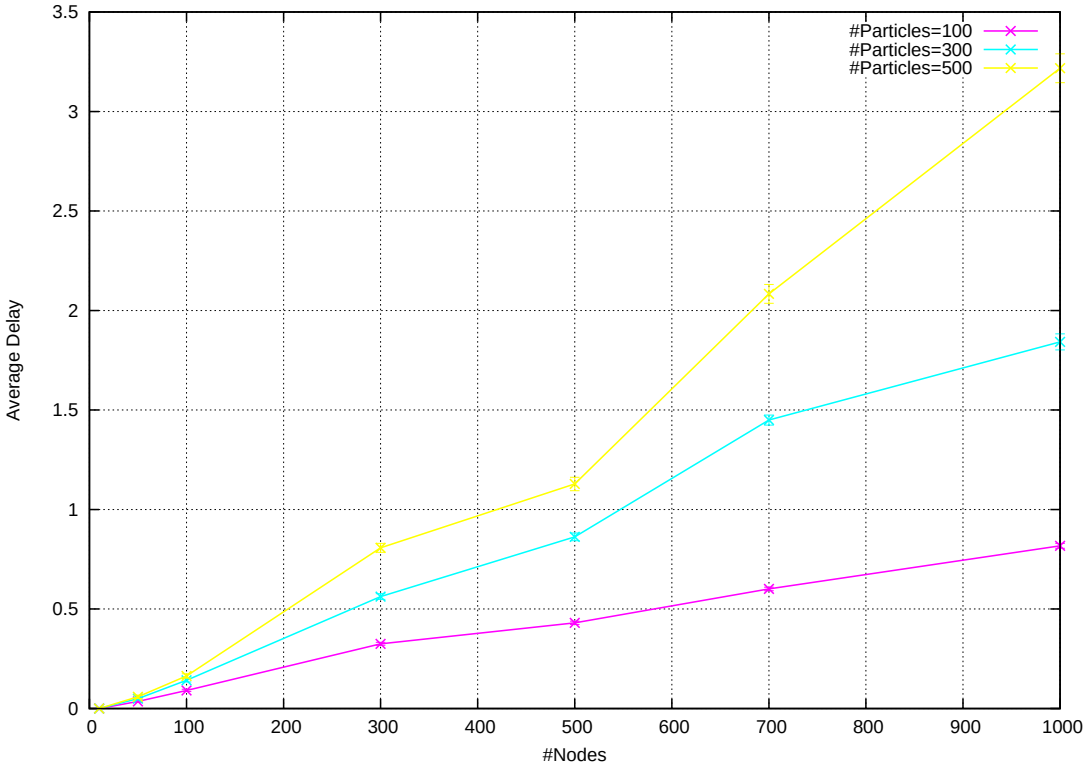
sensors. When the delay is larger than the sampling time, some observations are ignored: if a new sample is ready but the network is still completing the aggregation process, that sample is dropped.

Figure 6 shows the impact of using the GMM approximation, as opposed to sending the entire particle set, on the efficiency E when using the realistic network model. We consider three configurations, namely the traditional DPF with no GMM (0-GMM); the single Gaussian approximation of the DPF (1-GMM); and the approximation with a mixture of five Gaussian components (5-GMM). The GMM approximation considerably reduces the amount of information that the nodes have to exchange. As it can be observed from Figure 6(b), the delay increases much more slowly for 1-GMM and 5-GMM than for 0-GMM. When the GMM approximation are used, the delay remains sufficiently below the sampling time limit ($1s$). It can be also observed that transferring 5 components of the GMM demands clearly more bandwidth (higher delay) than 1 component only. Notice also that the performance of the 1-GMM and 5-GMM are almost comparable to those obtained with 0-GMM under ideal assumptions.

Remarks. We have addressed the problem of distributed target tracking for WMSNs using distributed particle filters and extended the formulation of a sequential algorithm to deal with realistic network scenarios. More specifically, we designed the algorithm to work with sensors with limited field of view, dealing with problems such as detection miss, target hand-over and target-loss. We demonstrated the proposed algorithm using a network simulator. Simulation results showed the importance of the co-design of distributed tracking algorithms and communication protocols.

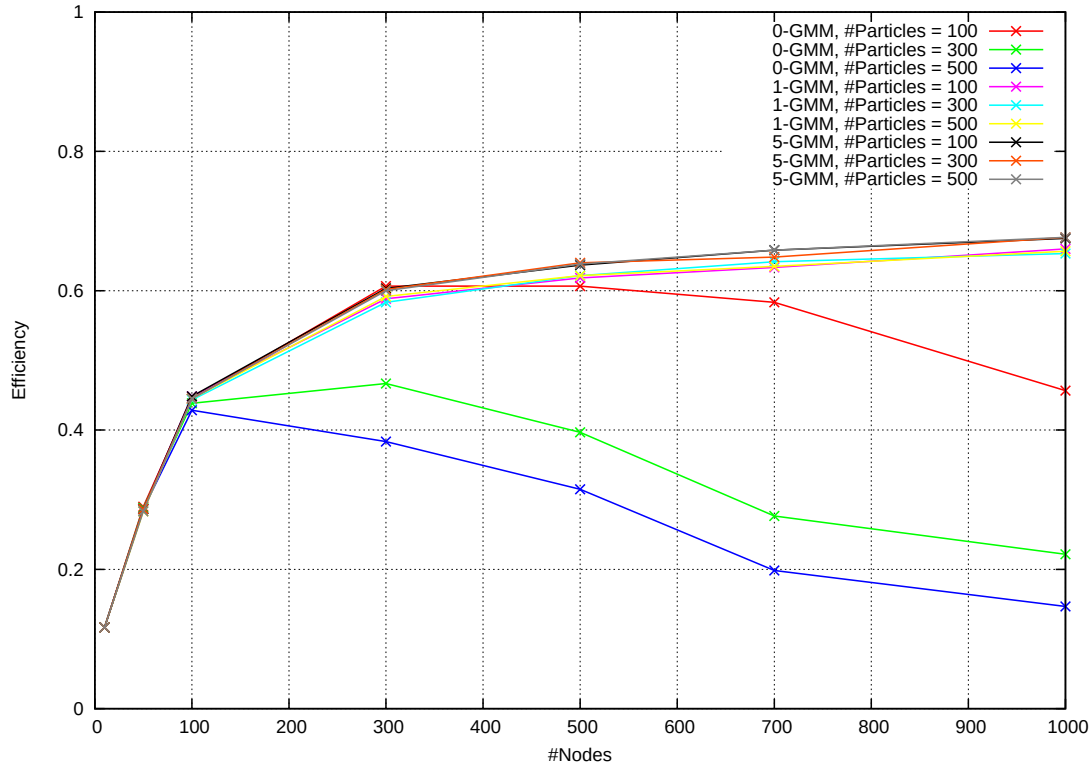


(a)

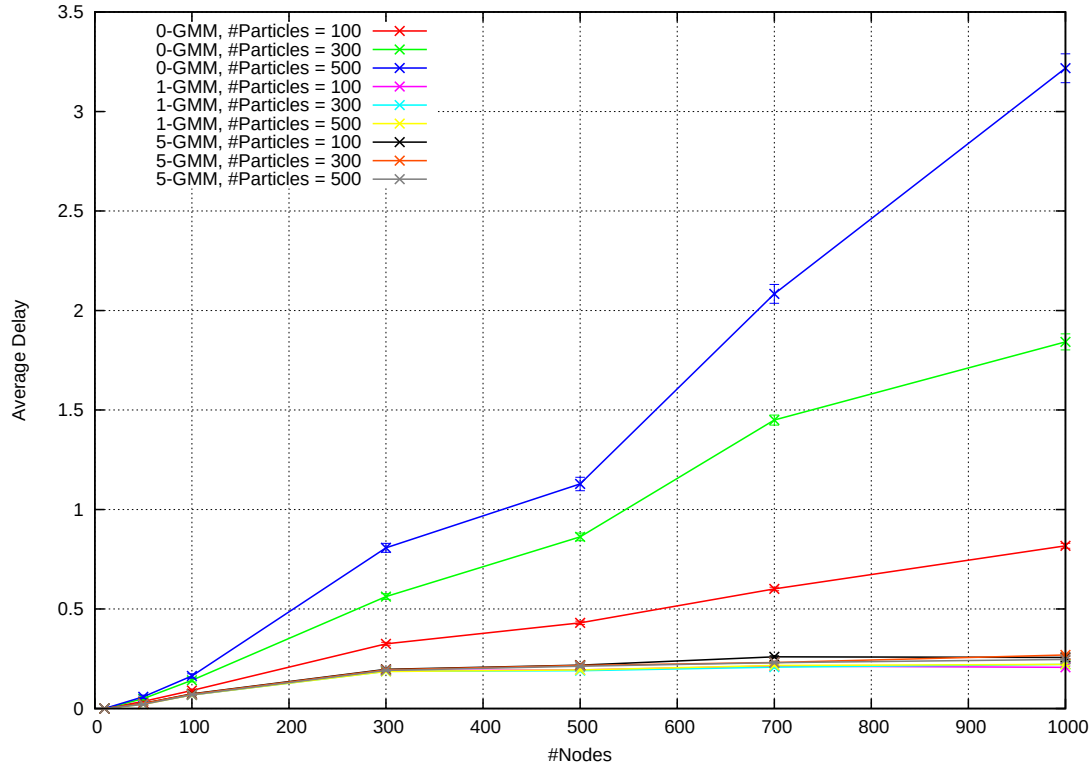


(b)

FIGURE 5. (a) Efficiency of a traditional DPF (0-GMM) while varying the number of sensors and particles, under ideal and realistic networking conditions. (b) Average delay of a traditional DPF (0-GMM) under realistic networking conditions.



(a)



(b)

FIGURE 6. (a) Efficiency and (b) average delay of the methods under analysis under realistic network conditions

CHAPTER 5

Towards Real Systems: Tools and Platforms

RECENT technological advances have favoured the evolution of traditional WSNs operating on simple scalar measurements towards WMSNs dealing with more complex vectorial data such as video and audio. Although the literature on WMSNs has mainly focused on multimedia streaming [**MRX08**] and bandwidth allocation [**CPLL09**], new applications such as human behaviour recognition and wide-area target tracking are demanding the definition of cooperative distributed applications for WMSNs.

Because of the complexity of such systems, it is common to undertake pure theoretical or simplistic approaches, thus moving too far away from actual problems. In order to drive the research efforts towards real-world problems, it is fundamental, in our opinion, to seriously take into account a methodology based on:

(1) simulations environments: that should allow to model complex aspects inherent to the different disciplines involved in the problem (e.g. networking, real-time operating systems, computer vision);
(2) real platforms: the availability of programmable devices is crucial, since technology highly motivates research challenges, and solutions, for WMSNs;
(3) real deployments: ultimately, the research efforts should allow to put in place real network of sensors capable of operating in real-world environment.

5.1. Background

In the following Section, a brief review is given of the state of the art for: simulation environment, sensor-node platforms, real deployments.

5.1.1. Simulation Environments. The understanding and the modeling of the complexity of distributed applications based on WMSNs require competences from several areas, ranging from networking to control theory, and from computer vision to data management. However, until now researchers have studied algorithms, applications and protocols for WMSNs without a holistic approach that addresses complementary and interconnected issues from all these disciplines. In particular, one of the main obstacles towards an integrated approach is the *lack of a common simulation framework where all the aspects related to different disciplines can be modeled and analysed simulteneously*. Some limited attempts have been made in this direction, such as in [**Pha10**], although specifically tailored to surveillance applications.

In the following we first discuss the state of the art in network simulators and then we focus on the main approaches adopted in computer vision to simulate distributed applications with WMSNs.

Discrete Event Simulation. In discrete event simulation, the operation of a system is represented as a chronological sequence of events. Each event occurs at an instant in time and marks a change of state in the system. Discrete event simulators are usually implemented making use of a resizeable event queue where to post and pop events for appropriate processing. For instance, time-triggered activities regularly post expiration events into the queue to produce a periodic sequence of actions. The queue is reordered at every post to always keep the closest event in front; the physical notion of time is discretized and incrementally elapses by the interval between the two latest expiration events at every pop.

Network Simulators. Popular network simulators, including NS-2, OMNet++, TrueTime, and OPNET [**www10c**,**www10d**,**Lun10**,**www10e**], can be compared based on the programming language they use, their script-ability, the support for standard protocols, and their extendibility/integrability. The last two properties of a network simulators are important for the appropriate modeling of local processing at each node. Table 1 summarizes the considered network simulators with their characteristics. A comparative review of the most popular network simulators is presented also in [**TGMB+08**].

	NS-2	*OMNet++*	*TrueTime*	*OPNET*
Language	C++	C++	Matlab	C++
Config/Setup	Tcl/Tk Scripts	*NED* files, *ini* files	Matlab	GUI
IEEE 802.15.4	with GTS	with GTS	CSMA only	-
OS aspects	Through Extensions [**PCL+10**]	-	Native	-
Extendibility/Integrability	Bad	Good	Good in Matlab	-

TABLE 1. Comparison of network simulators

NS-2, the most popular network simulator [**www10c**], is written in C++ and has a simulation interface based on TCL scripts. It was originally designed for simulations based on TCP/IP computer networks, but extensions have been provided to support mobile ad-hoc networks. Mobility of nodes requires further extensions, and mixing wired and wireless nodes in the same simulation is not straightforward. Since NS-2 was originally designed for networks of computers (high-end systems), the network stack of the nodes results more complex than in actual WSNs. Support for the IEEE 802.11 and IEEE 802.15.4 standards is provided by external contribution modules.

OMNet++, a component-based network simulator designed for hierarchical nested architectures [**www10d**], provides a discrete-event simulation engine and a GUI interface. The modules are written in C++, while a high-level Network DEscription (NED) language is used to assemble modules in components and to interconnect components to each other. A configuration file is used to specify the parameters of the simulation and the parameters of the modules. The tool can be used in a command line environment quite easily, since it provides several auto-tools to simplify the generation of make-files. However, the version 4.x is also based on the eclipse IDE. OMNet++ is highly flexible and the simulator can be integrated with external libraries, such as API to access Matlab from C++ programs and framework with advanced statistical analysis. Modeling extensions are provided to support for example mobility, wired/wireless standard and non-standard protocols, and energy models. Different

simulation packages provide an implementation of the IEEE 802.15.4 standard. To the best of our knowledge, at the time of writing this thesis, there are very few example of extensions that integrate OS aspects in OMNet++ (e.g. an attempt can be found in [**HLN09**]).

TrueTime, a Simulink toolbox developed for Matlab [**Lun10**], has been designed to test the effect of task scheduling on control algorithms, and it also includes simple high-level network models. Although TrueTime cannot be considered a full network simulation environment, it could be an easy way to add networking aspects to already existing Matlab-based simulations. The toolbox can be integrated with any type of Matlab functions, since the activities performed by the nodes have to be coded as Matlab functions (m-files or s-functions). A simple implementation of the IEEE 802.15.4 standard is also included.

Finally *OPNET*, a commercial network simulation software [**www10e**], whose source code is based on C++, has a GUI to configure the simulation scenarios and to develop network models. There are three level of configuration, namely the network level (to create the topology); the node level (to define the behaviour of the node and the data flow among the node components); and the process level (to define the protocols through a formalism based on finite state machines).

The Real-Time Network Simulator. Motivated by the need to jointly explore the two dimensional design space with respect to computation and communication, we have extended the NS-2 simulator in the Real-Time Network Simulator (*RTNS*) [**PCL+10**]. This essentially adds support for the modeling local processing, i.e. simulating operating systems. The extension also include the support full support for the IEEE 802.15.4. RTNS is a simulation suite to model operating system mechanisms for distributed networked applications, built by integrating the popular NS-2 *RTSIM* (Real Time operating system SIMulator [**PLL+02**]) packages. This tool facilitates designers to work in the two dimensions of communication (packet priority) and computation (task priority) in a systematic manner to achieve a more accurate and realistic WSN system's performance evaluation.

Computer Vision Simulation. Three main simulation categories can be identified based on the level of abstraction with respect to the vision problem used to validate distributed computer vision algorithms. These category are frameworks based on a *simplistic world assumption*, on virtual reality or on using directly real-world datasets.

Frameworks using a *simplistic world* approach assume point-like objects that have basic projection models in the image plane of the cameras and move on a ground plane with or without obstacles and boundaries (Figure 1). The use of point-like objects simplifies the problem of extracting features from objects. This approach is useful when one wants to focus on high-level problems, such as the coordination of a distributed application, while neglecting aspects related to the vision pipeline. For example, most works in literature on distributed tracking with WMSNs adopt this simplistic model. An example is [**ST10**] for distributed target tracking with smart camera networks. The authors show a simulation based on Matlab with a simplistic communication mechanism. Cameras have a field of view identified by an angle on the 2D world plane and the connectivity among cameras is based on circular communication ranges.

A second level of abstraction is reached when using *virtual reality* environments. In this case the objects and the surveillance areas are more complex, and the camera sensors produce more realistic synthetic images. *Virtual Vision* [**QT08**] can reproduce complex situations by simulating scenarios

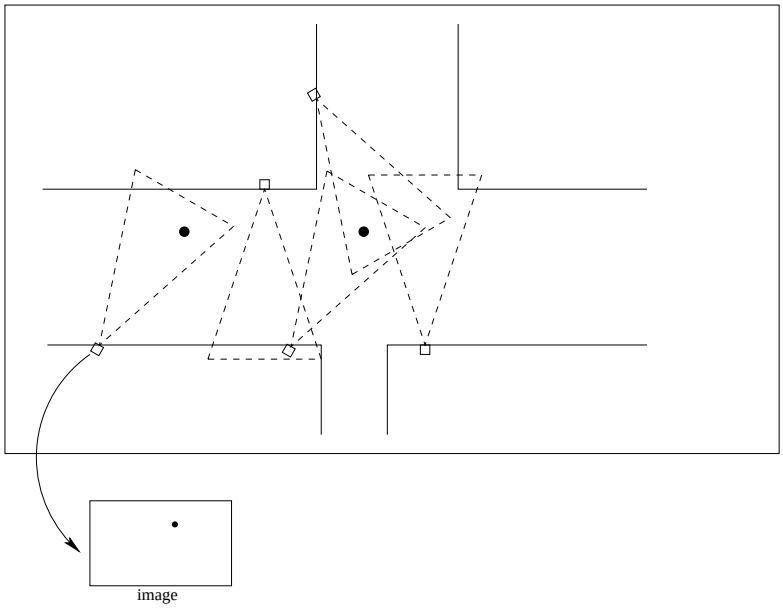


FIGURE 1. Example of simplistic world simulation using a poin target and a simple 2D space

where recording real images is difficult, expensive or not feasible. However, the effectiveness of this framework depends on the degree of accuracy achieved when defining and modeling the necessary details in the simulation environment.

The most realistic computer vision simulations use *real world datasets*. While this approach is very common for single-node computer vision, in distributed computer vision there are important issues to be considered in order to use the datasets in realistic communication and topology conditions within a network simulator. Basically, we must be able to reproduce the real world in the simulation environment itself, e.g. obstacles and absolute distances between nodes. For instance, a complete 3D representation of the scene should be provided together with the multi-camera dataset. To the best of our knowledge such a fine simulation approach has not yet been considered for distributed applications in WMSNs.

	Shirmohammadi 2010 [**ST10**]	*WVSN* [**Pha10**]	*Virtual Vision* [**QT08**]
Platform	Matlab	OMNet++	-
World	2D, simple	2D (3D), simple	3D, complex
Targets	multiple, point-like	single, point-like	virtual reality objects
Images	point-like obj (2D image)	point-like obj	synthetic images
Network	simple	complex	med

TABLE 2. Comparison of WCSNs Simulations

5.1.2. Sensor Node Platforms. Technology was, by far, the early driving force for the research community towards Wireless (Multimedia) Sensor Networks. The popularity and the breadth of the literature produced around WSNs and WMSNs has been possible because of two fundamental enabling aspects:

- Hardware: the availability of small embedded programmable boards, equipped with sensors, capable of communicating and processing data;
- Software: the availability of high-level programming abstractions such as operating systems with hardware abstraction APIs, communication stacks, high-level libraries, middleware and so on.

Early Hardware Platforms. Mote is the name universally used to identify the hardware platform of a sensor node. The origin of Wireless Sensors Networks, as autonomous systems with communication and computation capabilities, dates back to the Smartdust [**sma**] project (1998). The futuristic vision was to create motes with a size in the order of the cubic millimeter, but the project ended quite early in 2001. However, many similar projects followed up in the major research center. From such experiences, companies such as Crossbow [**xbo**] and Dust Networks [**dus**] started producing the first examples of commercially available motes.

The *Mica* family by Crossbow is based on the Atmel ATmega128L micro-controller boards with 4 kB of EEPROM and 128 kB of Flash. In this family, the MicaZ is equipped with an IEEE 802.15.4 compliant transceiver, capable of transmitting data to 250 kbps. The boards have connector to plug external peripherals, and in some releases they are already provided with light sensors as well as buttons and LEDs.

FIGURE 2. The MicaZ board.

A similar board is the *T-mote sky* by Moteiv. The board is based on the TexasInstruments MSP430 F1611 micro-controller, with 10 kB of RAM and 48 kB of Flash. The same IEEE 802.15.4 compliant transceiver is also available on this board. T-mote has an USB connection to the host computer to simplify development. The board integrates a humidity, temperature and light sensors.

Smart Camera Hardware Platforms. Technological achievements have extended the capabilities of the early prototypes of motes to manage more complex sensors and data such as cameras and images. Several research projects produced prototypes of embedded vision platforms suitable for application of WMSNs.

Among the first experiences, Panoptes project [**FKFB05**] aimed at developing a scalable architecture for video sensor networking applications. The key features of Panoptes sensor are a relatively

low-power and high-quality video capturing device, a prioritizing buffer management algorithm to save power and a bit-mapping algorithm for the efficient querying and retrieval of video data. Nevertheless the size of the sensor, the high power consumption, the relatively high computational power and storage capabilities make the Panoptes sensor more an high-level device than a sensor node.

The Cyclops project [**RBI+05**] provided another representative smart camera for sensor networks. The camera node is equipped with a low-power ATmega128 8-bit micro-controller. From the storage memory point of view the system is very constrained, with 128 KB of Flash program memory and only 4 KB of RAM data memory. The CMOS sensor supports three image formats: 8-bit monochrome, 24-bit RGB color, and 16-bit YCbCr color at CIF resolution (352 x 288). In the Cyclops board, the camera module contains a complete image processing pipeline for performing image size scaling, color correction, tone correction and color space conversion.

FIGURE 3. The Cyclops board [**RBI+05**].

In the MeshEye project [**HA06**], an energy-efficient smart camera mote architecture was designed, mainly with intelligent surveillance as target application. MeshEye mote has an interesting special vision system based on a stereo configuration of two low-resolution low-power cameras, coupled with a high resolution color camera. In particular, the stereo vision system continuously determines position, range, and size of moving objects entering its fields of view. This information triggers the color camera to acquire the high-resolution image sub-window containing the object of interest, which can then be efficiently processed.

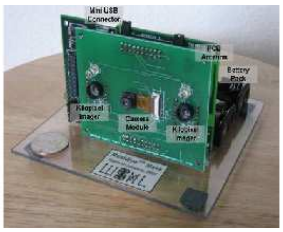


FIGURE 4. The MeshEye board [**HA06**].

Another interesting example of embedded vision system is represented by the CMUcam3 [**RGGN07**], developed at the Carnegie Mellon University. More precisely, the CMUcam3 is the third generation of the CMUcam series, which has been specially designed to provide an open-source, flexible and easy development platform with robotics and surveillance as target applications. The hardware platform is more powerful with respect to its predecessors and may be used to equip low-cost embedded system

with simple vision capabilities, so as to obtain smart sensors. The hardware platform is constituted by a CMOS camera, an ARM7 processor and a slot for MMC cards. Standard RF transceiver (e.g., TELOS mote) can be easily integrated.

FIGURE 5. The CMUcam3 board [**RGGN07**].

More recently, the CITRIC platform [**CAB+08**] integrates in one device a camera sensor, a CPU (with frequency scalable up to 624 MHz), a 16 MB Flash memory and a 64 MB RAM. Such a device, once equipped with a standard RF transceiver, is suitable for the development of visual WMSN. The design of the CITRIC system allows to perform moderate image processing task in-network, that is along the nodes of the network. In this way, there are less stringent issues regarding transmission bandwidth than with respect to centralized solutions. Such results have been illustrated by 3 sample applications, namely (i) image compression, (ii) object tracking by means of background subtraction and (iii) self-localization of the camera nodes in the network.

FIGURE 6. The CITRIC board [**CAB+08**].

Software Platforms. A fundamental enabling factor for the research community is the availability of programming abstractions. This allows researchers to focus more on algorithms and general problems than to spend long time on low-level implementation details. Two are the main features of a software platform that efficiently supports the underlying hardware while exposing a simplified and effective interface to the programmer: an operating systems and a communication stack.

The Operating System (OS) provides an abstraction of the machine hardware and is in charge of reacting to events and handling access to memory, CPU, and hardware peripherals. Especially in constrained hardware devices like those of sensor boards, the effectiveness in the OS paradigms largely affects the response in the target application. The execution model is the key factor differentiating the many solutions in existing OSs for WSNs.

TinyOS [**tin**] is the most popular OS for WSNs. It has been developed by the University of Berkeley in collaboration with Intel Research and Crossbow. TinyOS has been the first OS for WSN motes, providing an high-level programming language: network embedded system C (nesC). The language follows a component-based, event-driven programming paradigm. Concerning its internal features, TinyOS uses a stack shared among the processes and no heap. Each instance of the task runs until the end of the code unless it is preempted by an Interrupt Service Request (ISR) activated by an event occurrence; ISRs can in turn spawn a new task or call a function (command). The task scheduler implements a First Come First Served (FCFS) strategy. Lacking priorities and preemption, it is impossible to give precedence to more important activities.

Other Operating Systems (e.g. ERIKA , nanoRK [**Esw05**]) allow task preemption and real-time priority-driven scheduling. For this reason, the term Real-Time Operating System (RTOS) is generally used. Tasks can block on certain events, can activated upon occurrence of internal or external events (the reception of a network message or other hardware interrupts, or explicit activation by other tasks), or upon expiration of software timers. To permit pre-emption, some machine-dependent mechanisms must be implemented to save the "context" of the task (registers and stack pointer) at suspension occurrence. Such mechanism permits to resume the suspended computation when the task is rescheduled.

An intermediate software solution is given by Contiki [**DGV04**]. This OS uses a monostack memory model for an event-driven kernel. The application programs are dynamically loaded at runtime. It supports a thread-like coding style (protothreads) but enforcing a sequential flow of control; optionally multi-threading can be adopted, linking to a specific library.

ERIKA Enterprise RTOS [**eri**] is a multi-processor real-time operating system, implementing a collection of Application Programming Interfaces (APIs) similar to those of OSEK/VDX standard for automotive embedded controllers. ERIKA is available for several hardware platforms and it introduces innovative concepts, mechanisms and programming features to support micro-controllers and multi-core systems-on-a-chip. ERIKA features a real-time scheduler and resource managers, allowing the full exploitation of the power of new generation micro-controllers. Tasks in ERIKA are scheduled according to fixed and dynamic priorities, and share resources using the Immediate Priority Ceiling protocol. Interrupts always pre-empt the running task to execute urgent operations required by peripherals. RT-Druid [**eri**] is the Eclipse-based development environment for ERIKA Enterprise that allows writing, compiling, and analyzing an application. RT-Druid is composed by a set of plugins for the Eclipse Framework [**ecl**]. The RT-Druid Code Generator plug-in implements the OIL file editor and configurator (for a review on OSEK/VDX standard and OIL language see [**ose**]), together with target independent code generation routines for ERIKA Enterprise.

TinyOS is distributed with a collection of state-of-the communication protocols. The support for the IEEE 802.15.4 was provided as external contribution by the Open-ZB project. The Open-ZB toolset is available as open-source code [**ope**] and aims at implementing, testing and evaluating the current functionalities defined in the IEEE 802.15.4/ZigBee protocols as well as new add-ons. It includes: (1) the IEEE 802.15.4 protocol developed in TinyOS (v1.15 and v2.0), for the MICAz and Telos-B motes; (2) ZigBee Network Layer functionalities for supporting the Cluster-Tree network model (synchronized multiple cluster topologies) developed in TinyOS, for the Telos-B motes; (3) a simulation model of the IEEE 802.15.4 protocol developed in OPNET [**opn**]; and (4) several tools for network

dimensioning and timing analysis. The IEEE 802.15.4 implementation supports the beacon-enabled mode, therefore includes functionalities such as the slotted version of the CSMA/CA algorithm, different types of transmission scenarios (e.g. direct, indirect and GTS transmissions), network management (devices association and disassociation), channel scans (e.g. energy detection and passive scan) and beacon management.

One of the main drawback of ERIKA ROTS with respect to the other OSs, is that it does not provide a native communication stack. An IEEE 802.15.4 compliant stack has been provided as extension by the author.

5.2. A Simulation Environment for WMSNs

In this Section we present *WiSE-MNet*, a Wireless Simulation Environment for Multimedia Networks. WiSE-MNet is a generic network-oriented simulation environment that addresses the need for co-design of network protocols and distributed algorithms for WMSNs. To the best of our knowledge, this is the first unified environment for WMSNs that exhaustively addresses networking issues as well as computer vision and distributed application constraints. A modeling of the WMSNs is presented considering camera nodes with a simplified perception model. A distributed tracking algorithm is also developed over the proposed simulation environment to demonstrate the effectiveness of WiSE-MNet in modeling cooperative applications.

The simulator is an open-source project and is available online [**www11b**].

5.2.1. The *Castalia/OMNet++* network simulator. Our goal is to define a flexible simulation environment for distributed applications in WMSNs. The simulation environment we propose is based on the *OMNet++* framework [**www10d**]. The core of the simulator is a discrete event simulation engine based on modules and message exchange among modules. A message exchange mechanism allows to define local events (self messages) and remote events (messages to other modules). In OMNet++ there is no concept of network node: everything is either a simple or a composite module. A node is generally defined as a composition of modules, and, given the aforementioned paradigm, it is possible to define local and distributed behaviors in the same way: the OMNet++ simulation core is extremely flexible in terms of definition of local and distributed elements.

Simulation models available for the OMNet++ framework are sets of predefined modules providing the user an interface to simulate state-of-the-art network protocols. Examples of simulation models are the INET framework [**www10a**], which contains the most popular wireless and wired protocols (e.g. UDP, TCP, IP, OSPF); and the MiXiM project [**www10b**], a collection of mobile and fixed wireless networking protocols. We propose to use the *Castalia* simulation model for OMNet++ [**cas**] as it has been designed to model distributed algorithms for classic WSNs under realistic communication conditions. Castalia enables extensions and includes advanced wireless channel and radio models; a physical process (events) and sensing model; a model for node CPU with clock drift and power consumption; and the availability of MAC and routing protocols, including IEEE 802.15.4.

The nodes are interconnected through a *wireless channel* module, which is responsible for modeling the wireless link. Each node is also interconnected with one or more *physical processes* that model events

occurring in the external environment. Each node is the composition of a communication module, a sensor manager, an application module, a resource manager and a mobility manager.

The distributed algorithm to be tested has to be defined in the *application module*. From a networking view-point this is the *application layer*. The *communication module* uses a simplified network stack model based on three layers, namely radio (physical layer definition), MAC and Routing; and provides communication capabilities to the application module. The *sensor manager* is responsible for providing to the application module new samples from the external environment by interacting with the physical processes. The *mobility manager* is responsible for the location of the sensor in the simulation area (however note that we will consider here only sensors in fixed positions). The *resource manager* can be used to model the local resource usage such as energy consumption, memory usage, and CPU states.

5.2.2. Extensions. Although Castalia has been wisely designed for WSNs, its adaptation to WMSNs requires extensions to the original framework. These extensions include a generalized structure to support *complex data types*, rather than simple scalars; a model for the *target* movement and for the video *sensor* (camera); an *idealistic communication mechanism* to test algorithms without considering the impact of the network, while still designing them in a distributed manner; and a simple *GUI for 2D world scenarios*. Figure 7 summarises our extensions to the network and node model.

One of the main limitation of Castalia is the sensor data-type that is forced to be a scalar value, while WMSNs are based on vectorial data such as images. We proposed a generalized sensor data-type exploiting the object oriented paradigm provided by C++ and NED. The information exchanged between the physical processes and the sensor managers is defined through a base class called *WmsnPhysicalProcessMessage* which can be specialized in concrete classes containing any type of data. To support this extension, even the physical process module has been generalized, so that we could either use a *WmsnMovingTarget* class to model the movement of a target in a 2D plane, or use a *WiseVideoFile* to feed the sensor manager with real-world video sequence.

Following the same principle, the sensor manager and application modules have been extended. In this case a *WmsnCameraManger* class has been provided to model a simplistic sensor-camera, while different application classes have been developed to test different algorithm. Indeed, the *WmsnBaseApplication* is the basic application class that the user should derive to implement the logic of its own distributed application.

The idealistic communication mechanism is necessary because there is no possibility in Castalia to entirely bypass the communication components. More precisely, Castalia gives the opportunity to implement a pass-through communication stack with ideal radio. Unfortunately, with this configuration, if two nodes attempt to communicate at the same time the result will be a failure in the radio component, since the Castalia ideal radio cannot send and receive at the same time. We proposed an alternative module that bypasses the Castalia communication modules by interconnecting directly application layers of the nodes.

Simplistic and complex network. The underlying *network infrastructure* is configurable and the following setups are possible:

- a fully connected network with infinite bandwidth and each node can communicate with all the other nodes without delays;

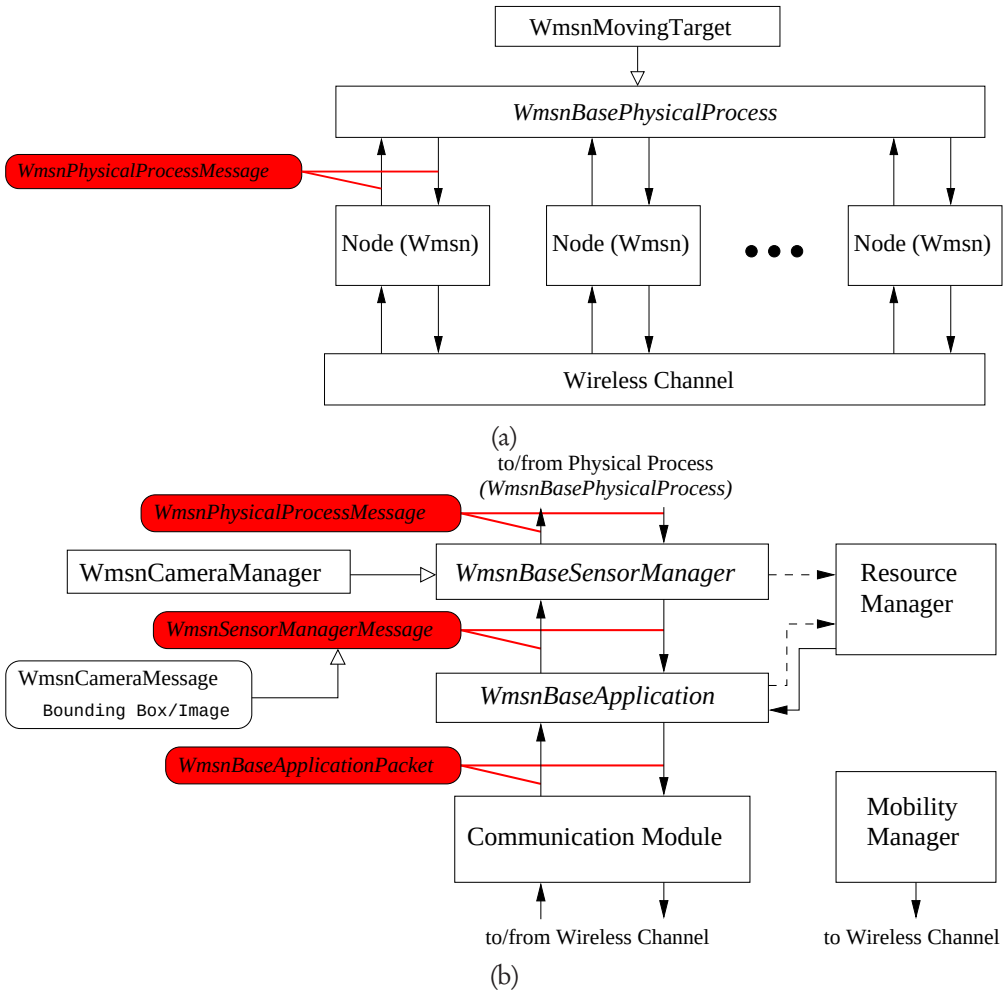


FIGURE 7. Extensions introduced to *Castalia* for (a) the network model and for (b) the node model

- limited transmission ranges with infinite bandwidth;
- limited transmission ranges with finite bandwidth (communication delays);
- realistic wireless stacks, with standard protocols.

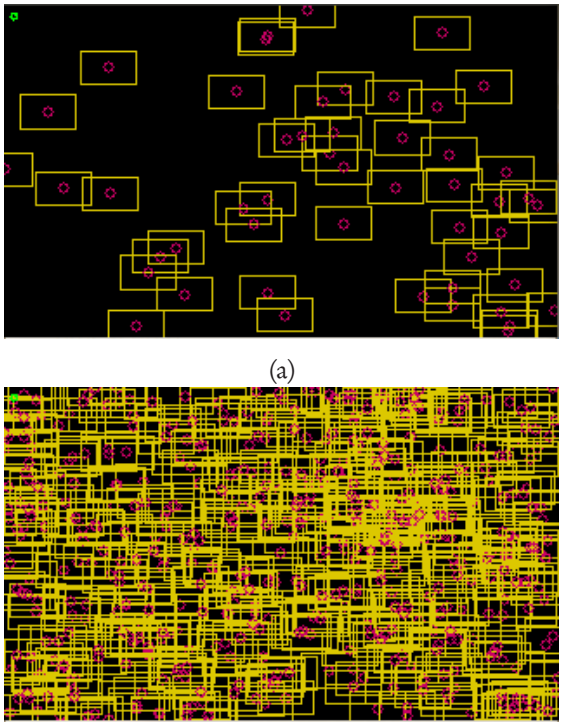
Energy consumption is not currently considered, although it is already supported by Castalia.

Simplistic world model. The current version of WiSE-MNet works with a simplistic world model and extensions to the case of real-world dataset is possible with minimal effort. Similarly to [**ST10**], the surveillance area is modeled as a 2D plane and the extension to the 3D world is simplified by the support provided in OMNet++ and Castalia. Given the native support of OMNet++ and Castalia to 3D node positioning, this will be straightforward as long as we take into account 3D world coordinates in our extension code.

Unlike [**ST10**] that represents objects as points, to model the results of an object detection stage applied to real-world images we describe objects with a (bounding) box. The cameras are assumed to be top-down facing, thus obtaining a simplified projection model. More complex placement and projection models will have no major impacts on the definition of the distributed application. The relationship among cameras (calibration, overlapping/non-overlapping fields of view) are statically defined.

5.2.3. A distributed tracking application example. In this section we present an example where a distributed particle filter (DPF) inspired by [**HDH09**,**SHR05**] is implemented in a network of cameras and is used to estimate the position of a moving target [**NC11**]. The nodes exchange information about their (partial) posterior probability using a Gaussian Mixture Model (GMM) to reduce the amount data. We configure the algorithm to operate with a set of 500 particles and with a 5-component GMM approximation. The field of view (FOV) of a camera is calculated considering its view angle and assuming a distance of $6m$ from the ground-plane, thus resulting approximately in a FOV of $10m \times 6m$. The camera are positioned according to a random uniform distribution, and the overall area under surveillance is of $100m \times 60m$.

The communication modules provided with the Castalia package and the proposed extensions allow to configure the network in different ways. In this demonstration we adopt the T-MAC communication protocol [**vDL03**]. We configure the protocol as follows. The *request-to-send* (RTS) and the *clear-to-send* (CTS) mechanisms are used. The RTS/CTS is a classic mechanism to avoid (or to reduce) the number of collisions in wireless communication protocols. The basic idea is that a node can transmit only when the access to the channel has been granted. The RTS/CTS mechanism has an overhead, which reduces the effective available bandwidth for transmission, although retransmissions due to collision might be saved. *Acknowledged transmission* is used. The use of acknowledged transmission is required if we want a packet to be delivered after a collision takes place. If for some reason (generally a collision) the packet is not received, the source node can be notified by a missing acknowledgement from the destination and restart the transmission. If the acknowledgements are not supported a different mechanism to detect the failure might be required. The inactive period (node in sleep mode) is disabled. The inactive period is designed to extend the lifetime of the nodes by periodically switching from an active state (normal operation condition) to a sleep state, where the node cannot perform any operation (including communication). The inactive period has been disabled to avoid the situation in which a node attempt to communicate during the sleep mode. However, it is possible to consider a different definition of the DPF algorithm, which is aware of those inactive intervals, thus saving node energy. The number of *retransmission attempts* is 10 and the *transmission bandwidth* is $250kbps$. Let us consider two setups, one with a network deployment of 50 nodes (Figure 8(a)) and one with the number of nodes increased to 500 (Figure 8(b)). The true trajectories of the target (ground truth) and the corresponding estimation produced by the network using the DPF are shown in Figure 9. As it can be observed by comparing the results for the trajectories, the performance obtained in the deployment of 500 nodes are much better than the case with 50 nodes. This is mainly due to the poor coverage that is obtained in the latter case. When the target moves in the portion of the surveillance area uncovered by the cameras' FOVs, the estimation of its position is not possible. In case of the deployment with 500



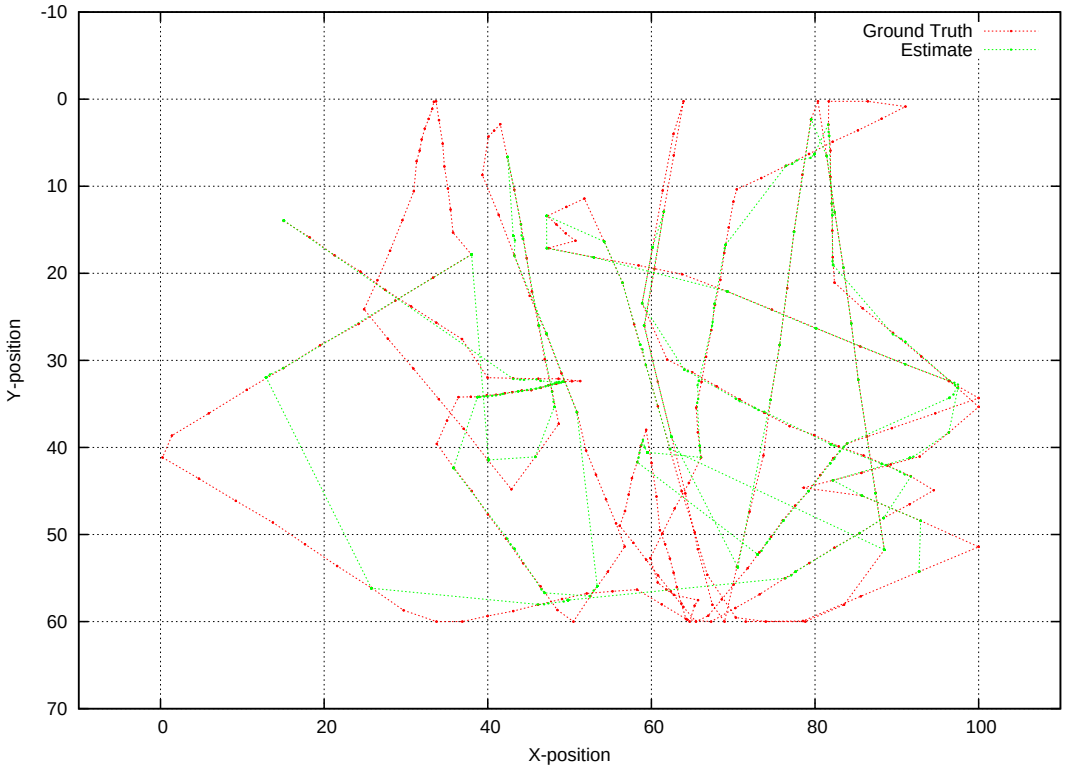
(a)

(b)

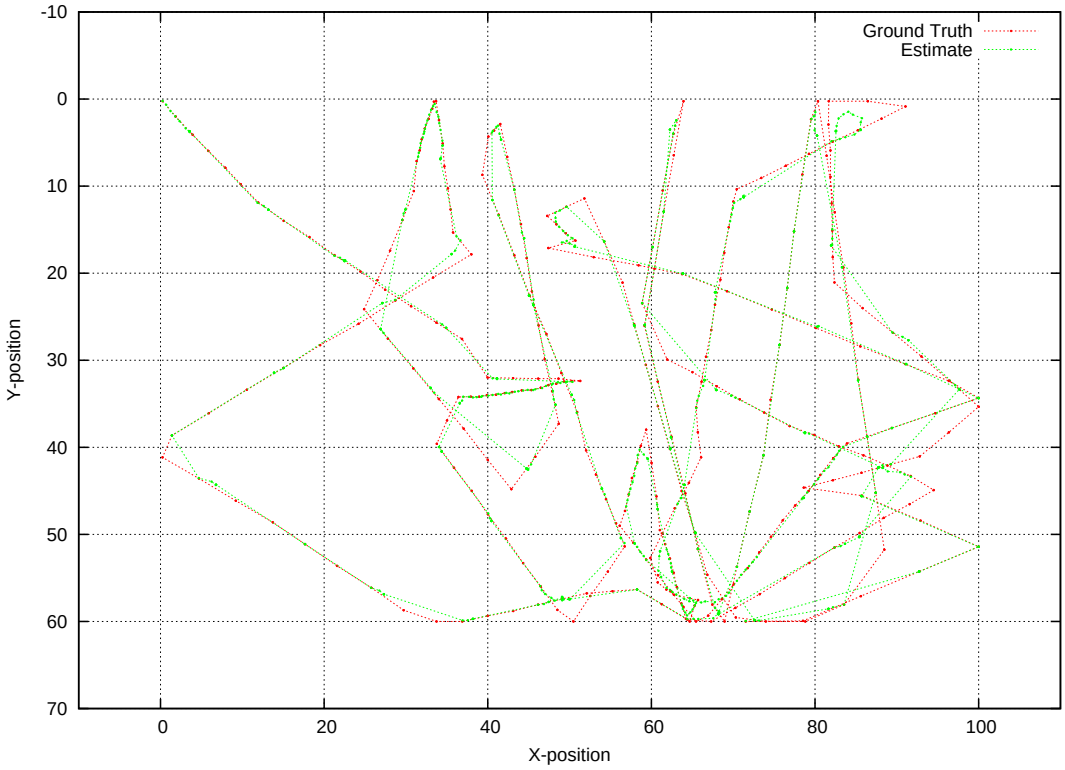
FIGURE 8. Example of sensor deployment in WiSE-MNet. Coverage of the area under surveillance with (a) 50 nodes and with (b) 500 nodes. The yellow boxes represent the FOV of each camera, whose center is identified by the magenta circle in the middle. The target is the small green box in the top left side of each figure

nodes, the surveillance are is completely covered. There is also a second contribution which is related to the redundancy of camera views. In the setup with 500 nodes the number of camera observing the target at the same time is higher than the case of 50 nodes. These multiple observations are exploited by the distributed particle filter to improve the estimation with respect to few (or single) observations. Notice that this consideration is valid under the assumption that the bandwith demand related to the communication of the 5 GMM components is compatible with the available transmission bandwidth.

To understand the effect of the network delay we runned a simulation for the setup with 500 nodes, comparing two configurations of the DPF: with the GMM approximation of 5 components (5-GMM); without the GMM (0-GMM), i.e. nodes exchange the whole particle set. The results are shown in Figure 10. Although avoiding the Gaussian-mixture approximation is theoretically more efficient, since the partial posterior of the DPF is represented with the entire particle set, in a realistic networking scenario the situation is different. Owing to the larger amount of data produced in case of 0-GMM, the network delay to transfer the partial posterior from a node to the other increases considerably with respect to the 5-GMM case. Consequently, the estimation process is slower in case of 0-GMM, resulting in a loss of performances with respect to 5-GMM.



(a)



(b)

FIGURE 9. True and estimated trajectory for the sensor deployment (a) with 50 nodes and (b) with 500 nodes

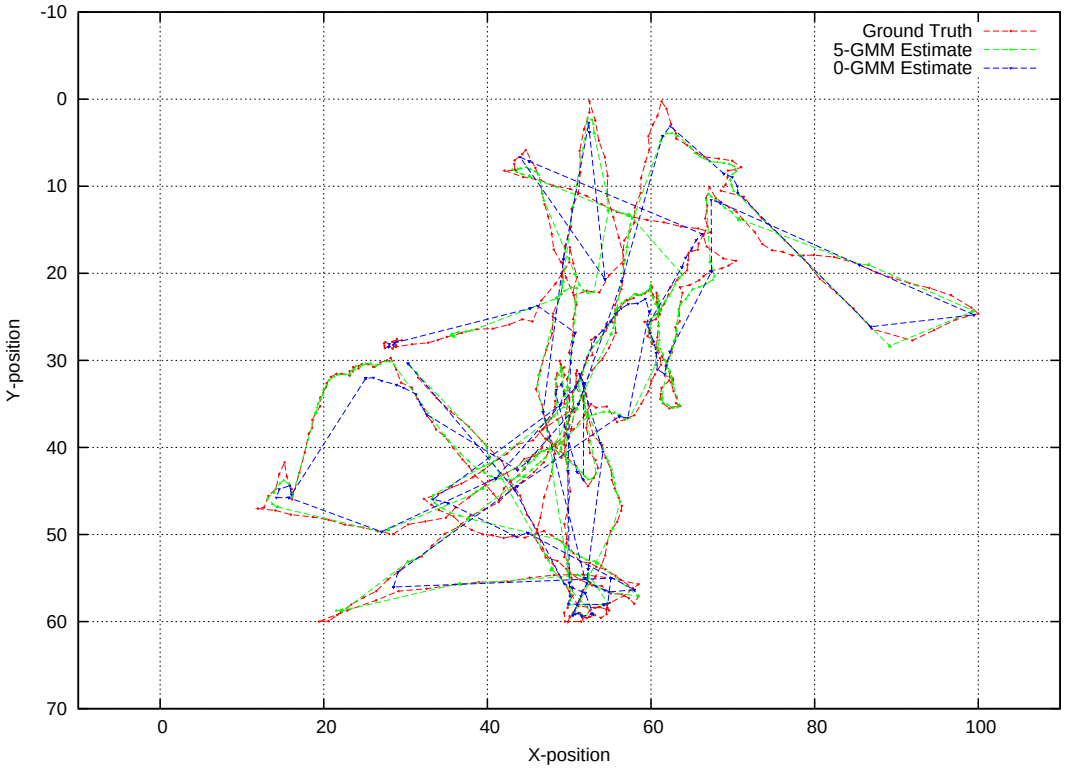


FIGURE 10. True and estimated trajectory for 0-GMM and 5-GMM for the sensor deployment with 500 nodes

5.3. A Sensor-Node Platform for WMSNs

Design of a node for Wireless Sensor Networks is a complex task, being the cost effectiveness and power consumption the first two design target. In wireless multimedia sensor networks, the design is furthermore complicated by the presence of complex sensors (i.e. camera and/or microphones) requiring additional significant amounts of computational power and memory. In 2009, during the IPERMOB project [**www11a**], our group was in charge of designing a WSN device equipped with a camera to be adopted in an ITS application.

5.3.1. SEEDEYE : a board for WMSNs. Due to the complexity of the design, we conducted a preliminary investigation for identifying the various design constraints present. The outcomes of this analysis,

Requirements	High Level Design Constraint	Low Level Design Constraint
Nodes must be battery powered.	Low power consumption hardware components.	Consumption on wake, idle, sleep states of: Micro-controller, Transceiver and Camera.
Maintenance must be minimal.	Energy management policies.	
Nodes must acquire color or gray-scale images with sufficient resolution.	Camera must support required resolution.	Camera hardware model.
	MCU must store/retrieve images for elaboration.	MCU internal memory, number/speed of I/O ports.
Network must be reliable.	Transceiver must comply with public standards and specifications.	Transceiver with standard transmission bandwidth.
Wider coverage is preferred.	Transceiver transmits with sufficient energy.	Transceiver configurable with high transmitter/receiver gains.

TABLE 3. Requirements for the SEEDEYE board.

summarized in Table 3, well depicts the complexity of the design process, emphasizing how crucial and strictly related were the choices associated with the micro-controller and the camera.

Memory Requirement. Camera resolution affects the amount of memory required to the system. Considering a low resolution image in the QVGA format (320 x 240 pixels), a single frame occupies 76800 bytes for gray-scale images and 153600 bytes for color images (2 bytes per pixel required by 5-6-5 RGB or 4-2-2 YUV formats). Standard low power MCUs adopted by WSN nodes, such as the ATmega1281 installed on Cyclops possess an extremely limited RAM memory (8-16 KByte) and thus are incompatible with our requirements. In IPERMOB , when we encountered this limitation, we initially decided to extend the limited internal RAM connecting an external memory. The choice of this component was not trivial due to strict requirements in terms of power consumption, access speed and Read/Write latencies. Furthermore, the reduced number of I/O pins available on low power micro-controller (usually less than 50), was incompatible with a large number of parallel memories. The solution we adopted for a first prototype was the Ramtrom FM25V10, a serial 8 Mbit Ferromagnetic RAM. This satisfied all these constraints but was later discarded due to its high cost.

Micro-controller. In the final prototype we decided for a Microchip MCU PIC32MX795F512L, characterized by a sufficient amount of internal RAM (128 KB). While being more consuming than popular MSP430 platform (i.e. widely adopted in WSN applications) and 30% more costly, the PIC32 satisfied memory requirements providing, at the same time, all the necessary I/O peripherals and up to 80 MIPS of computational power. It must be point out that, being an MCU designed for embedded systems, (i.e. no dedicated floating point unit, reduced instruction set), the design of internal software logic was not a trivial process.

Camera. Concerning the image sensor, since the first release, we chosen an highly configurable module from Hynix (HV7131GP, CMOS camera) originally designed for the cell phones market. For reducing the system power consumption, the camera was directly connected to the MCU, without relying on external frame-grabber devices. Through a control interface, it is possible to configure the camera for acquiring images with periods of seconds or minutes. Furthermore, this device can be set into a low power state, highly reducing energy consumption.

Radio Transceiver. Wireless connectivity was made possible by Microchip MRF24J40MB devices. These IEEE 802.15.4 compliant transceivers implement in hardware the full physical stack layer and some functionalities of the MAC layer. Furthermore, they possess the highest transmission gain (up to +20dBm, limited by ETSI rules to +9dBm) available on commercial IEEE 802.15.4 devices. This feature allows communications characterized by reduced Bit Error Rates [**PGS+11**] also for nodes spaced up to 50 meters. Similarly to the other electronic components, also this device can be set into a low power state through its control interface.

Other Interfaces. Concerning the wired connections, the board provides an Ethernet, a USB and two RS232 interfaces. The last release of the device (namely SEEDEYE) used in the IPERMOB final testbed is depicted in Figure 11. Concerning the power consumption of the device, Table 4 remarks the

FIGURE 11. The SEEDEYE board.

importance of a correct setup of the hardware components. In particular, the total consumption can be significantly reduced by bringing MCU and camera into their low power state (sleep mode) when no elaborations are required and by minimizing wireless transmissions.

Component	Active Consumption	Low Power Consumption
MCU 120	mA	40 μA
CAMERA	26 mA	48 μA
TRANSCEIVER	130 mA (data TX) 25 mA (data RX)	5 μA

TABLE 4. Comparison of network simulators

5.3.2. Porting ERIKA RTOS. In order to provide an programming abstraction for the new platform, the porting of the ERIKA RTOS has been implemented. The porting of a minimalistic embedded OS requires two main aspects to be addressed:

- multi-tasking support;
- device drivers.

For multi-tasking support, ERIKA has a modular architecture based on an Hardware Abstraction Layer (HAL). The HAL allows to segregate the implementation of platform-independent components from the underlying hardware. For instance: task definition, scheduling policies and synchronization primitives rely on the same C-language implementation. In this way, porting such functionalities to new platform is simplified. Essentially, the architecture-dependent context switch for task preemption and Interrupt handling is the minimal part that has to be coded to have a working kernel. For the device drivers, the original structure of ERIKA was less flexible. For this reason we have introduce a minimal architecture that allows for code reuse. The first step is to provide support for micro-controller unit (MCU) peripherals/buses such as: Digital I/O, Universal Asynchronous Transmitter-Receiver (UART), Serial Peripheral Interface (SPI), Inter Integrated Circuit (I2C). To facilitate portability of application code, the primitives to access these peripheral try to have common API and to hide the implementation details to the application layer. Once MCU buses are supported, drivers for external devices, accessible through such MCU buses, can be provided. The external device drivers developed for ERIKA are structured in two components: a peripheral-dependent part and an MCU-dependent part. The first one implement the logic necessary for the drive the external peripheral, while the second one hides the implementation details related to the MCU bus being used. For instance, given a radio transceiver, this is connected to the MCU through the SPI bus. The logic related to control of the transceiver, i.e. the control commands and data exchange, is can be coded irrespective on the underlying implementation of the SPI bus. Such logic is the peripheral-dependent part. The MCU-dependent one is responsible for the management of the SPI bus. Owing to this separation, when the same external peripheral (e.g. radio transceiver) is used on a different micro-controller, only the MCU-dependent side (e.g. SPI) needs to be done; this might be a simple adaptation of drivers already available for the MCU bus. Using this approach, several drivers have been developed for radio transceiver, camera and others.

5.3.3. μWireless : a portable IEEE 802.15.4 stack. A fundamental component of a WMSN platform is the wireless communication stack. The de facto standard for WSN is the IEEE 802.15.4 protocol. ERIKA RTOS did not provide any native support for wireless communication. For this reason, one

the early stages of our work we have decided to provide an IEEE 802.15.4-compliant communication stack for ERIKA : μWireless .

The IEEE 802.15.4 features the current implementation of μWireless supports:

- Beacon-enabled mode;
- Coordinator/End-device time synchronization;
- Data transmission and reception in slotted and unslotted mode;
- GTS allocation and transmission;
- End-device association.

The architecture of μWireless is depicted in Figure 12. The μWireless stack uses an OS-independent

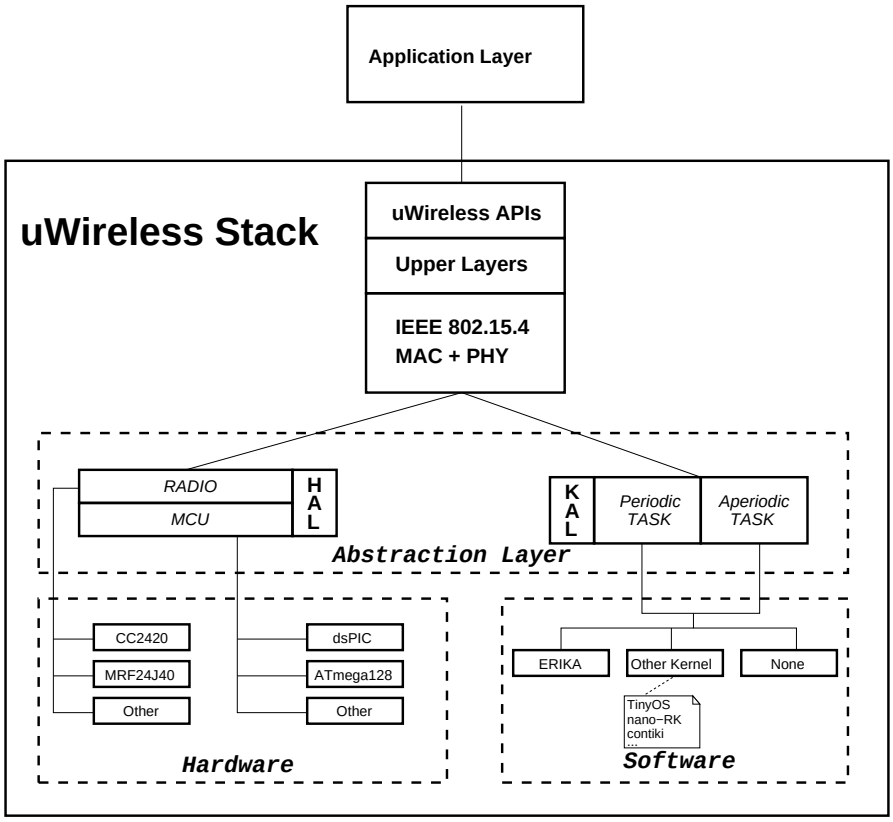


FIGURE 12. Software architecture of μWireless : HAL provides abstraction to access the radio transceiver; KAL provides abstraction for real-time multi-tasking (periodic and aperiodic tasks are required by the stack from the OS).

architecture to implement the IEEE 802.15.4 protocol. Similarly to what ERIKA does, μWireless has a modular and flexible architecture that allows independence from the underlying hardware. Moreover, μWireless allows also independence from the hosting operating system. This is possible because of an abstraction layer with two components:

- Hardware Abstraction Layer (HAL): hiding details of the hardware platform such as the drivers of the radio transceiver and other MCU-peripheral drivers (for external Interrupts, Digital I/O, etc.);
- Kernel Abstraction Layer (KAL): hiding details of the multi-tasking functionalities provided by the OS.

The real-time multi-tasking support is necessary to execute μWireless . This is essentially due to the beacon-enabled mode of the IEEE 802.15.4 standard that requires periodic operations to be performed with a precise time schedule. The KAL exports the RTOS multi-programming API in common representation, this allowing to easily port the stack on new platforms.

Summarizing, when the stack has to be ported to a new HW/SW architecture, ideally only the HAL and KAL have to coded. The current platform support is:

- micro-controllers: Microchip dsPIC-33F, Atmel Atmega128 and Microchip PIC32;
- radio transceiver: Texas Instruments CC2420 and Microchip MRF24J40;
- ROTS: ERIKA OS.

5.3.4. μCV a low-complexity computer vision library. In order to provide a basic support for the image processing, a low-complexity computer vision library was developed in the scope of the IPERMOB project: μCV . The is particularly optimized to be executed on low power micro-controllers. In particular we avoided the usage of dynamic memory allocation and floating point operations. Elementary computer vision processing are supported such as: binarization via different thresholding algorithms, morphological operations, frame differencing, histograms. The library, written in pure C-

FIGURE 13. Example of operations with μCV library: binarization and morphological closing.

language code, is architecture-independent and thus extremely portable to any HW architecture. This allowed μCV to be executed also on normal PC, an fundamental feature that permitted the development of algorithm off the embedded platform. Essentially, image processing techniques were tested on image acquired from the SEEDEYE board, but making use of all the debug tools available of normal workstation. Once the algorithms were validated, the porting to the SEEDEYE was merely a matter of verifying the execution time.

CHAPTER 6

Case study

IN the period from 2000 to 2020, reliable estimations [**Com08**] forecast a growth rate of 50% for freight transport and 35% for passenger transport. The European Commission is proposing to confine the civic and environmental problems arising from this trend by reducing road traffic congestion (amounting to 0.9-1.5% of the EU GDP), road transport-related CO_2 emissions (accounting for 72% of all transport-related emissions), and road fatalities (about 40,000 in 2006). As a matter of fact, transportation is not efficient: vehicles travel empty for more than 50% of their capacity, as capacity in the transport system is not fully utilized. Vehicles uselessly consume energy in congestion and aggressive driving, waste a lot of users' time to reach the desired destination and to look for a parking, creates social problems due to incidents and pollution generated diseases, requires maintenance interventions to repair damages caused by weather, incidents or improper usage. Constructing new infrastructures and upgrading the existing ones cannot make by themselves transport more sustainable, which means efficient, clean, safe and seamless; scientists and policy makers feel that proper exploitation of the opportunities offered by Information and Communication Technologies (ICT) can help in reaching this goal.

6.1. Background: The Urban Accessibility Problem

A prominent component of the mobility problem highlighted above is that of accessibility, most frequently viewed as a concept that somehow relates to consumers ability or willingness to enter into any system (e.g. the health care, the transport hubs like airport, rail stations, bus terminus, industrial aggregates); naively speaking, access represents the degree of "fit" between the clients and the system iteself. A unique definition for access is not widely accepted; rather a set of "accessibility indicators" are proposed in the literature [**Koe80**] in order to support evaluation studies especially at a disaggregate level; they appear to be an effective determinant of peoples' behavior and thus a key variable to be introduced into traffic or urban development models.

The accessibility problem imposes a proper management of "in-road" and "off-road" resources aiming at improving the road network system balancing the demand to the offer by:

- privileging certain kind of services;
- delivering to end-users accurate enough information appropriately filtered to match their intentions (e.g. destination, path, etc.);
- supporting the decisions to be taken by road and parking operators;
- acquiring a sufficient data set aimed at supporting infrastructure studies.

6.1.1. ITS managing urban access. In such perspective, Intelligent Transportation Systems (ITS) are the ICT proposal to control mobility and accessibility in diversified scenarios like those of urban areas (e.g. London and Stockholm congestion charge systems), the highways networks, the national road networks.

The Tuscan regional authorities have been incorporated the directives coming from the EC related to the urban systems; this special attention is motivated by the criticality of the urban road infrastructure, intermingling private services and freight logistics, exposed to pollution, thus directly impacting on the quality of life of citizens. It is therefore beneficial to *support for the wider deployment of an updated multi-modal European ITS Framework architecture for intelligent transport systems* (Action 2 in [**Com08**]); these target systems will *include an integrated approach for travel planning, transport demand, traffic management, emergency management, road pricing, and the use of parking and public transport facilities*.

The technical aspect is translated into how to take action, selecting the proper installations eligible to scale (for costs and communication capabilities) to typical urban size, and limiting the required interventions of civil infrastructures (for instance poles, supports, cables, etc.). The wireless approach is that of connecting the sensing stratum to the upper levels for computation and user applications without relying on wired backbones.

Another important aspect is that of guaranteeing interoperability/interconnection with infrastructure systems and facilities; the new ITS framework must enable users with different mandates and disciplines to operate in a cooperative and cohesive manner to acquire access, retrieve, analyze and disseminate traffic-related data and information in an easy and secure way.

Finally the ergonomics of the different classes of final users must be taken into account to permit an effective profiling and exploitation of the services provided by the ITS.

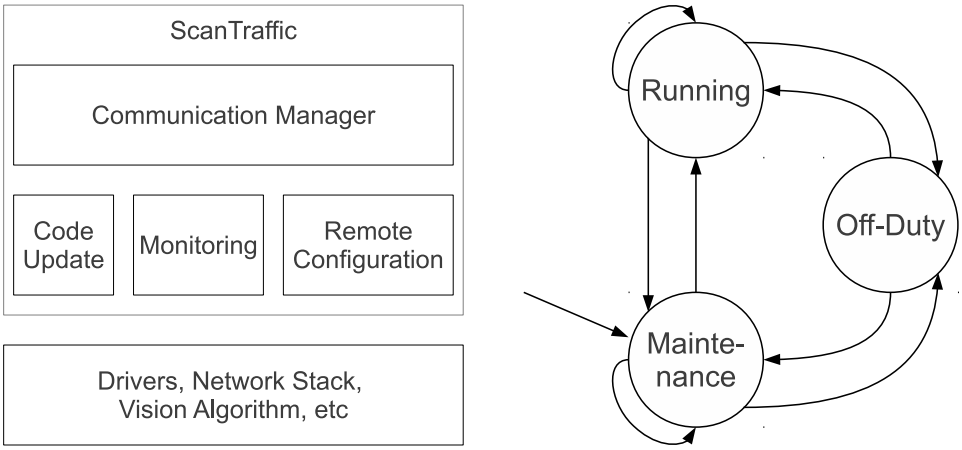
6.1.2. Contribution. This Chapter reports about an ITS project in the region of Tuscany (Italy): IPERMOB [1], a multi-tier Information System for urban mobility. IPERMOB proposes a new generation of integrated systems based on the optimization and inter-operability of the chain formed by data collection systems; aggregation, management, and on-line control systems; off-line systems aiming at infrastructure planning; information systems targeted to citizen and municipality to handle and rule the vehicle mobility. Specifically IPERMOB proposes low-cost wireless technology (like that of Wireless Sensor Network) and image processing techniques to estimate traffic-related information. As testbed, IPERMOB will provide real-time information about parking availability and vehicle flows on the land-side of the International Airport of Tuscany (Pisa, Italy).

6.2. System Design

The IPERMOB project proposes a pervasive and heterogeneous infrastructure to monitor and control urban mobility. Within the project, a prototype of such infrastructure has been implemented and deployed at the Pisa International Airport. The IPERMOB architecture has three tiers:

(1) data collection,

[1] "Infrastruttura Pervasiva Eterogenea Real-time per il controllo della Mobilità ("*A Pervasive and Heterogeneous Infrastructure to control Urban Mobility in Real-Time*"). `http://www.ipermob.org`



(a) General architecture (b) Finite State Machine

FIGURE 1. ScanTraffic software design.

(2) data sharing, and
(3) data consumption.

The data collection tier employs different technologies to acquire data related to urban mobility. Specifically, the prototype employs Vehicular Ad-hoc Networks (VANETs) and visual WSNs to collect traffic data. The data sharing tier provides, to the upper layer, a standard interface for accessing the data produced by the lower layer and stores it for future use. The data consumption tier is the application layer. Applications can be on-line control systems providing real-time information to the users (drivers, police, etc.), or off-line systems aiming at infrastructure planning. The implemented prototype provides example applications of both types.

To manage and control the visual WSNs employed in the data collection tier, we designed and implemented *ScanTraffic* [**AAP+12**], a distributed software infrastructure running within the WSN (i.e., in each node of the network) and providing three main services: monitoring, remote sensor configuration, and remote code-update. We argue that this is the minimal set of services to be provided by any visual WSN in order to fully exploit the potentiality of smart cameras. *Monitoring* is the primary service offered by any WSN. It allows to control the sensing activity performed by the nodes and to retrieve the collected data. *Remote sensor configuration* is often not necessary in traditional WSN employing simple sensors which can be configured (e.g., calibrated) before deployment. On the contrary, it is an important service for smart cameras whose visual algorithm needs to be configured for the current camera view, i.e., after deployment, and usually adjusted over time. *Remote code-update* is essential to benefit from smart camera versatility. By replacing sensor firmware, code-update allows to improve or completely change the functionality of a smart camera sensor. Figure 1(a) shows the software architecture of ScanTraffic. There is a total of 4 software modules: 3 service modules (one for each of the main services previously discussed), plus the "Communication Manager" controlling and coordinating them.

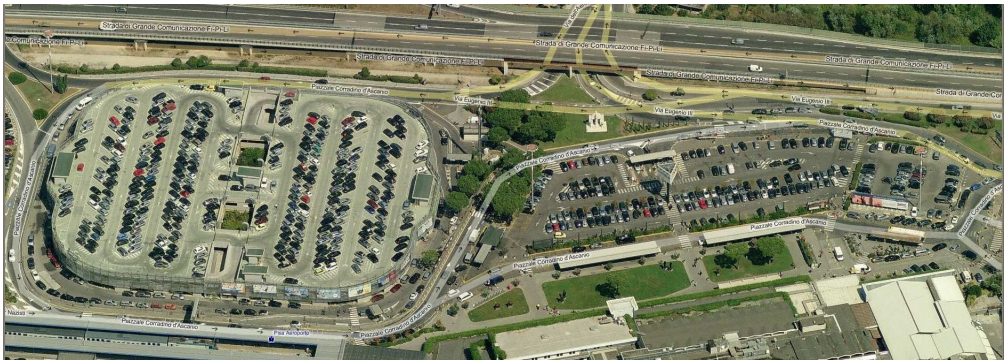


FIGURE 2. A view of the Pisa International Airport land-side. We deployed flow sensors in the main intersections, and parking sensors in both the outdoor parking lot (on the right) and the indoor parking lot (on the left).

Contributions. The contribution to the IPERMOB project are mostly related to the lower level of the general architecture depicted in Figure 1 1(a). In particular:

- Definition of and RTOS support for the hardware platform SEEDEYE
- Definition of the IEEE 801.15.4 wireless communication stack μWireless
- Definition of the computer vision processing library μCV and algorithms

6.3. Implementation and Deployment

We implemented a prototype of the system infrastructure proposed by IPERMOB and deployed it in the land-side of the Pisa International Airport (depicted in Fig. 2). The prototype serves as a proof-of-concept for the entire system, including visual WSNs managed by ScanTraffic. The goal of such visual WSNs is to collect information about parking lot occupancy and traffic flow. For this purpose, we used the two embedded vision algorithms described in [**MMN+11**]: one algorithm counts cars passing in a road section; the other algorithm detects the occupancy status of a set of parking spaces (presented in Chapter 3). For the sake of brevity, we use the name "flow sensor" to denote a smart camera running the former algorithm and the name "parking sensor" to denote a smart camera running the latter. As described in Chapter 3, those algorithms achieve an overall detection rate of 95% with a false alarm rate of 0.1%.

Each visual WSN is connected to the rest of the system via a backhaul link. A special node in the WSN, the coordinator, acts as a gateway between the WSN and the upper layers. In the IPERMOB prototype the backhaul link is a HiperLAN link (see Fig. 3) and the coordinator uses the UDP/IP protocol to communicate with upper tiers.

6.3.1. Hardware Description. The smart cameras used in the prototype were designed within the IPERMOB project. Specifically, two different boards were developed. The first board, designed to be particularly low-cost and low-power, is the SEEDEYE described earlier in this Chapter. In the development of the second board, power constraints have been slightly relaxed in order to explore a solution with more flexibility and processing capabilities. The second board is an FPGA-based device equipped

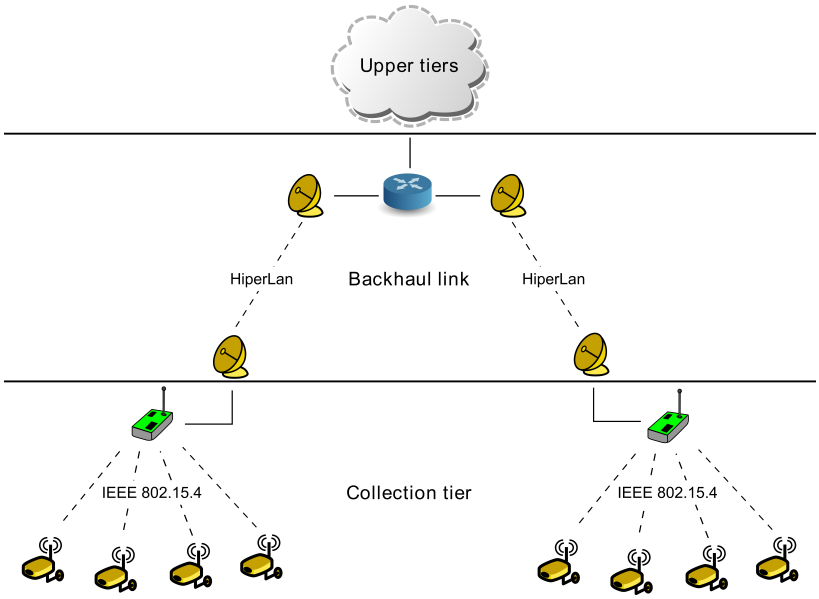


FIGURE 3. Network architecture of the IPERMOB prototype: backhaul links connect WSN coordinators to upper tiers.

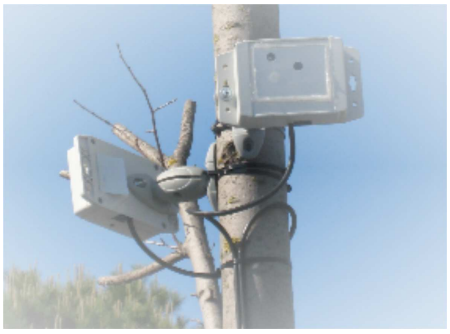


FIGURE 4. The SEEDEYE smart camera used in the IPERMOB project. The board is equipped with a Microchip PIC32 micro-controller, an IEEE 802.15.4 transceiver, a CMOS camera, and an Ethernet port used for debugging purposes. Since the device is battery-powered and communicates wireless, it can be easily installed almost everywhere.

with 32 MB of RAM. The FPGA hosts a soft-core microprocessor, thus allowing to reuse the code developed for the other board. The FPGA also drives the camera to its maximum frame (i.e., 30 fps), although the soft-core runs at only 40 MHz. This feature, together with a relatively high amount of RAM, makes the board suitable for monitoring traffic flow.

6.3.2. Software Platform. Each node runs ERIKA RTOS, both in case of SEEDEYE boards and the FPGA-based boards. ERIKA 's IEEE 802.15.4-compliant μWireless stack supports the beacon-mode and the GTS features which are required by the monitoring module of ScanTraffic, i.e., Mirtes [**APN+10**].

TABLE 1. Memory requirements for each possible combination of hardware platform and node type.

Device	Node Type	RAM [KB]	ROM [KB]
SEEDEYE	Coordinator	26.7	150.1
SEEDEYE	Parking Sensor	24.5 + Image Size	111.6
FPGA	Flow Sensor	16.0 + Image Size	80.0

ScanTraffic memory requirements for each possible combination of hardware platform and node type are shown in Table 1. Sensor node RAM requirements have a variable component depending on the chosen image resolution, e.g., in case of 160×120 frames, 19 KB of additional RAM is accounted.

6.4. Testbed Planning and Implementation

The deployment of the system in a complex urban scenario required an accurate planning, with the following objectives:

(1) minimize the number of sensing devices, keeping the number of monitored parking spaces as large as possible,
(2) minimize the number of WSN coordinators, i.e., nodes with broadband connection to the control center;
(3) reducing installation time;
(4) enhancing the reuse of existing poles and supports;
(5) maximizing system performance, carefully choosing the positions of the sensors, avoiding non-line-of-sight communication.

We deployed a total of 21 smart cameras, specifically: 14 parking sensors monitoring 83 parking spaces, and 7 flow sensors monitoring 8 traffic lanes.

The deployment of smart cameras was greatly simplified and sped up by the availability of remote configuration and code-update. The actual deployment (i.e., excluding the planning phase) was carried out by two people and it took less than three days. Most of the time was spent to physical install the sensors to pre-existing poles or trees. The configuration phase (including camera orientation) took just half a day. We programmed every smart camera with a special deployment firmware which constantly acquires the current camera view and sends it to a laptop connected to the Ethernet interface, thus allowing to properly orient the camera to capture the desired scene. Once the installation was completed, we remotely uploaded the operational firmware (flow or parking sensor) to each smart camera and we configured the algorithm using the remote configuration service. We could have used the configuration service also to remotely acquire the current camera view, thus avoiding the need for the procedure just described. However, this approach would actually have been slower, because the limited bandwidth does not permit a fast image retrieval, thus considerably slowing down the camera orientation setup.

6.4.1. Validation. The IPERMOB prototype served as a test-bed to validate ScanTraffic design and implementation. ScanTraffic real-time features (inherited from Mirtes) proved to be essential for the

TABLE 2. Power consumption of both devices for different configurations.

Device	Node Type	Image Size	Frame Rate [fps]	P_{max} [mW]	P_{idle} [mW]
SEEDEYE	Parking Sensor	160 × 120	1	450	6
		320 × 240	1	462	6
FPGA	Flow Sensor	160 × 120	30	1230	25
		320 × 240	20	1230	25

correctness of the information produced by the WSNs. Jitter-free periodic queries guarantee that the number of counted vehicles is related to a known time interval equal to the query period (tunable at run-time). It is therefore possible to precisely compute the traffic flow with a specific time resolution. The synchronous sensing enforced by ScanTraffic permits to build a time-coherent status of the whole parking lot from the single parking spaces.

The prototype deployment showed that visual WSNs running ScanTraffic can be quickly installed in a urban scenario reusing existing poles and supports, without the need for costly and time-consuming engineering works. Indeed, one unique characteristic of ScanTraffic visual WSNs is that they can be easily deployed on-request where needed, and subsequently removed. Such temporary installation can be used to provide ITS services during big events (e.g., concerts, festivals, etc.) or to collect data for planning infrastructure improvements.

Power Consumption. Currently ERIKA does not support the idle state for our boards and we did not implement it, since IPERMOB main goal is to prove the feasibility of using visual WSNs in ITSs, taking energy related issues for a possible follow-up. However, we measured the power consumption of each board in idle state (P_{idle}) and at full load (P_{max}), i.e., with the radio transmitting and both the camera and the algorithm running. Results are shown in Table 2. For the parking lot monitoring application, it is reasonable to assume that, with a monitoring period of one minute, the SEEDEYE -based board powered by 4 AA batteries, i.e., with a total voltage (V_{bat}) of 6 V, will last for about 3 weeks. Indeed, the camera and the algorithm can run just once a minute, therefore the board will be on for less than 2 seconds every minute (considering a frame rate of 1 fps, the start-up time, and the computation time). If we use a high beacon order, e.g., 10, and a low superframe order, e.g., 4, the radio is on for just 1 second every minute. The resulting duty cycle (α) is equal to 5% and, supposing a battery capacity (C_{bat}) of 2500 mAh the node life is:

$$\frac{C_{bat} \cdot V_{bat}}{\alpha \cdot P_{max} + (1-\alpha) \cdot P_{idle}} = \frac{2500 \cdot 6}{.05 \cdot 462 + .95 \cdot 6} \left[\frac{\text{mAh} \cdot \text{V}}{\text{mW}}\right] \simeq 520.8\,[\text{h}] \tag{31}$$

On the contrary, the FPGA board, used for the vehicular flow monitoring, will last for just 11 hours. The reason is not only the higher power consumption, but primarily the duty cycle of 100% required by the flow monitoring. Indeed, entering the sleep mode would cause to miss some vehicles.

Data Transmission Reliability. From a communication point of view, ScanTraffic must operate in a harsh environment. In public places as the Pisa International Airport the ISM 2.4 GHz band is

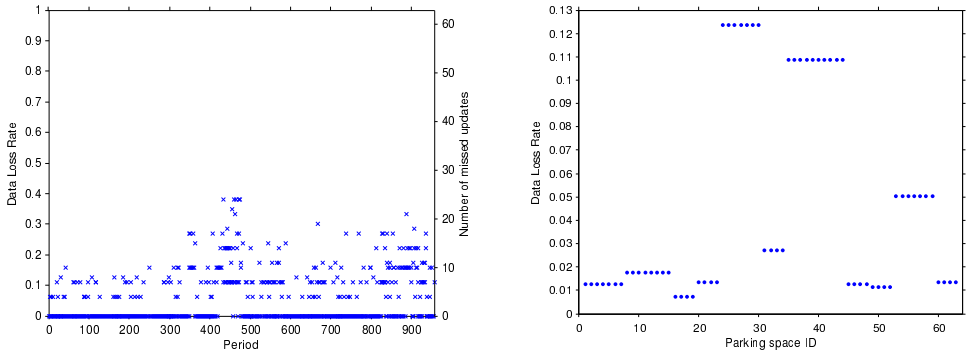


FIGURE 5. ScanTraffic prototype data transmission reliability: (a) evolution of the data loss rate during the 5-hour-long validation experiment, and (b) average data loss rate per parking space.

much crowded. Moreover cars, as well as other metal objects, highly reflect electromagnetic waves and produce multipath interference whose modeling is difficult because of the rapid changes in the scenario. The adopted data protection can reduce the probability of data loss, but cannot completely avoid it. However, as long as data loss is moderate, it is not a problem, because IPERMOB target applications can tolerate missing data. Indeed, their metrics of interest are aggregate values.

To evaluate the data transmission reliability of the deployed prototype, we set a monitoring period of 20 seconds and, for every period, we counted the number of missing parking space and flow updates. The experiment lasted for more than 5 hours, i.e., for almost 1000 periods. Fig. 5 shows the results of such experiment performed on the biggest visual WSN deployed within the prototype, having 11 sensors monitoring a total of 63 parking spaces. Therefore, in each period, 63 updates were expected. Fig. 5(a) shows the percentage of missed updates for every period. On average, just 4.5% of the data is lost (i.e., the occupancy status of 2-3 parking spaces is not updated). Fig. 5(b) shows the loss rate per parking space. Parking spaces monitored by the same sensor have the same loss rate, because the sensor groups all the updates together in one message. For most of the sensors only 2-3% of the messages are lost, with the exception of 3 sensors whose loss rate is up to 12%. Those sensors are responsible for the data loss peaks shown in Fig. 5(a). Indeed they are the farthest from the coordinator and therefore their link quality can easily drop under the minimum acceptable level. The easiest solution to fix this problem is to add another coordinator nearer to them, thus creating a new WSN. Indeed, the considered area, spanning for more than 3 000 m^2, is probably too large to be covered by just one coordinator.

Remarks. We showed how a proper network/software design is necessary to fully exploit smart cameras features. The system was built from two previous works of ours, i.e. Mirtes and the embedded vision algorithms. We successfully integrated the computation-intensive vision algorithm with Mirtes without compromising its real-time features and we added error correction techniques to Mirtes in order to address the communication issues posed by real-world scenarios. However we quickly realized that to achieve the aforementioned advantages over traditional scalar WSNs, additional services were

needed, namely remote configuration and code-update. The resulting system has been described in the paper, proving that it is suitable for real-world deployment.

Conclusions

THIS thesis deals with a notable set of issues related to Wireless Multimedia Sensor Networks; more specifically we introduced and discussed a distributed approach for computer vision algorithms. Following the motivations and challenges, as discussed in Chapter 1, we proposed a set of technological solutions allowing for pervasive computer vision and system-level QoS.

We generally followed a two-fold schema, looking at theoretical and simulation studies on one hand, and looking at real-world implementation on the other hand. This approach has provided concreteness to this research and contributed to the early exploitation of our experimental results.

We contributed to the design of a new WSN platform, encompassing all the most recent advances in terms of connectivity, computing power, and wireless communication capabilities; the Seed-Eye board, shortly going to appear on the market, complies with IEEE802.15.4 specifications. The same board can also be configured as a wireless-to-wired gateway leveraging the Ethernet and USB connection adapters.

To respond to the typical real-time requirements of vision-based applications, the ERIKA real-time micro-kernel has been ported to the PIC32 MCU embedded into the Seed-Eye, formerly unsupported before this work. ERIKA features of real-time multi-threading have also permitted a seamless and effective implementation of the IEEE802.15.4 MAC layer features (i.e., the μWireless software package).

Simulation studies tackled with the feasibility of a scheduling manager at system-level aimed at guaranteeing the schedulability of a set of flows (having real-time constraints) while preventing the system break down by undisciplined admission of data flows. We also addressed the bandwidth allocation mechanisms in cluster-tree topologies following a component-based approach. All simulation results have been achieved considering IEEE802.15.4 at the MAC layer to permit early validation of proposed bandwidth allocation mechanisms in real-world testbeds.

Moving from the network to the application layer, we discussed several problems related to computer vision in WSN. We looked at the problem of local on-board processing proposing two possible solutions based on machine learning: the heavier and more accurate one makes use of mining techniques whereas the lighter, really implemented, on hardware makes use of neural networks.

To assist the early implementation and rapid prototyping of novel, lightweight computer vision algorithms targeted to resource-constrained devices we also implemented a set of functions, organized in a software library (i.e., the μCV software package).

The results listed above have been finally deployed as hardware and software modules in the IPERMOB collection layer; this project has demonstrated the feasibility of a multi-tier information system for traffic monitoring and control at the urban scale. In IPERMOB the Seed-Eye boards run single-node logic based on computer vision to detect parking availability and vehicle flow and classification in real time.

The results obtained by IPERMOB have nonetheless been limited to single-node logic. We finally addressed the problem of multi-node tracking where the nodes are expected to collaborate to accomplish the purpose of the distributed system. We developed both the theoretical framework (based on the distributed particle filter algorithm) and the simulation counterpart (based on the OMNeT++ package) for deploying a large-scale set-up extracting the position of a mobile target by aggregating the high-level information from the collaborating nodes.

These latest results are opening a broad set of research opportunities in the domain of distributed vision algorithms, middleware design, in-network processing and data aggregation, collaborative patterns. These objectives are being addressed by the "Real-Time Networks" area of the ReTiS Lab within specific research projects and industrial collaborations.

Bibliography

[AAP+12] Daniele Alessandrelli, Andrea Azzarà, Matteo Petracca, Christian Nastasi, and Paolo Pagano. Scantraffic: Smart camera network for traffic information collection. In *to appear in the Proceedings of the 9th European conference on Wireless Sensor Networks*, EWSN'12, Berlin, Heidelberg, 2012. Springer-Verlag.

[AFD07] Abdalkarim Awad, Thorsten Frunzke, and Falko Dressler. Adaptive distance estimation and localization in wsn using rssi measures. In *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools*, DSD '07, pages 471–478, Washington, DC, USA, 2007. IEEE Computer Society.

[AGJM04] Geraud Allard, Leonidas Georgiadis, Philippe Jacquet, and Bernard Mans. Bandwidth Reservation in Multihop Wireless Networks: Complexity, Mechanisms and Heuristics. pages 1–11, 2004.

[AH01] Luca de Alfaro and Thomas A. Henzinger. Interface theories for component-based design. In *EMSOFT '01: Proceedings of the First International Workshop on Embedded Software*, pages 148–165, London, UK, 2001. Springer-Verlag.

[AMC07] Ian F. Akyildiz, Tommaso Melodia, and Kaushik R. Chowdhury. A survey on wireless multimedia sensor networks. *Computer Networks*, 51(4):921–960, March 2007.

[AMGC02] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[APN+10] Daniele Alessandrelli, Paolo Pagano, Christian Nastasi, Matteo Petracca, and Aldo Franco Dragoni. Mirtes: middleware for real-time transactions in embedded systems. In *Human System Interactions (HSI), 2010 3rd Conference on*, pages 586 –593, May 2010.

[ARSP06] N. Aslam, W. Robertson, S.C. Sivakumar, and W. Phillips. Reservation based medium access control protocol for wireless sensor networks. *CCNC 2006. 2006 3rd IEEE Consumer Communications and Networking Conference, 2006.*, pages 969–973, 2006.

[AY05] Kemal Akkaya and Mohamed Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3):325–349, May 2005.

[Bar03] Sanjoy K. Baruah. Dynamic and static-priority scheduling of recurring real-time tasks. In *Real-Time System*, pages 93–128, 2003.

[BDWL10] Abdelmalik Bachir, Mischa Dohler, Thomas Watteyne, and Kin K. Leung. MAC Essentials for Wireless Sensor Networks. *IEEE Communications Surveys & Tutorials*, 12(2):222–248, 2010.

[BETVG08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

[BH69] E. Bryson and Y. C. Ho. *Applied optimal control: optimization, estimation, and control*. Blaisdell Publishing Company, 1969.

[Bou97] C. A. Bouman. Cluster: An unsupervised algorithm for modeling Gaussian mixtures, April 1997. Available from http://www.ece.purdue.edu/ ~ bouman.

[BsLS06] Andrew Barton-sweeney, Dimitrios Lymberopoulos, and Andreas Savvides. Sensor localization and camera calibration in distributed camera sensor networks. In *in Proceedings of IEEE Basenets*, 2006.

[But97] Giorgio Buttazzo. *HARD REAL-TIME COMPUTING SYSTEMS: Predictable Scheduling Algorithms and Applications*. Kluwer Academics Publishers, 1997.

[CA09] Jaejoon Cho and Sunshin An. An adaptive beacon scheduling mechanism using power control in cluster-tree wpans. *Wirel. Pers. Commun.*, 50(2):143–160, 2009.

[CAB+08] Phoebus Chen, Parvez Ahammad, Colby Boyer, Shih i Huang, Leon Lin, Edgar Lobaton, Marci Meingast, Songhwai Oh, Simon Wang, Posu Yan, Allen Y. Yang, Chuohao Yeo, Lung chung Chang, J. D. Tygar, and S. Shankar Sastry. Citric: A low-bandwidth wireless camera network platform. In *In Proceedings of the International Conference on Distributed Smart Cameras*, 2008.

[Can86] J Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, June 1986.

[cas] The Castalia simulation model. `http://castalia.npc.nicta.com.au/`.

[CCM+12] Mangesh Chitnis, Salvadori Claudio, Petracca Matteo, Pagano Paolo, Lipari Giuseppe, and Santinelli Luca. *Visual Information Processing in Wireless Sensor Networks: Technology, Trends and Applications*, chapter Distributed Visual Surveillance with Resource Constrained Embedded Systems, pages 272–292. IGI Global, 2012.

[CHTC07] Tanya L. Crenshaw, Spencer Hoke, Ajay Tirumala, and Marco Caccamo. Robust implicit EDF. *ACM Transactions on Embedded Computing Systems*, 6(4):28–es, September 2007.

[CM02] Carla-fabiana Chiasserini and Enrico Magli. Energy consumption and image quality in wireless video-surveillance networks. In *In Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, 2002.

[Coa04] Mark Coates. Distributed particle filters for sensor networks. In *Proceedings of the third international symposium on Information processing in sensor networks - IPSN'04*, New York, USA, 2004. ACM Press.

[Com08] Commission of the European Communities. Action Plan for the Deployment of Intelligent Transport Systems in Europe. Technical Report COM(2008), 2008.

[CPLL09] Mangesh Chitnis, Paolo Pagano, Giuseppe Lipari, and Yao Liang. A Survey on Bandwidth Resource Allocation and Scheduling in Wireless Sensor Networks. *2009 International Conference on Network-Based Information Systems*, pages 121–128, August 2009.

[CV09] Mauricio Casares and Senem Velipasalar. Light-weight salient foreground detection with adaptive memory requirement. In *ICASSP '09: Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1245–1248, Washington, DC, USA, 2009. IEEE Computer Society.

[CV10] Mauricio Casares and Senem Velipasalar. An adaptive method for energy-efficiency in battery-powered embedded smart cameras. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, ICDSC '10, pages 167–174, New York, NY, USA, 2010. ACM.

[DGV04] Adam Dunkels, Bjorn Grnvall, and Thiemo Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Proc. of the First IEEE Workshop on Embedded Networked Sensors (Emnets-I)*, Tampa, Florida, USA, November 2004.

[dus] Dust Networks. `http://www.linear.com/products/wireless_sensor_networks`.

[ecl] Eclipse - an open development platform. `http://www.eclipse.org/`.

[eri] ERIKA Real-Time Operative System. `http://erika.tuxfamily.org`.

[Esw05] Eswaranv and Rowe and Rajkumar. Nano-rk: An energy-aware resource-centric operating system for sensor networks. In *Proceedings of IEEE Real-Time Systems Symposium*, 2005.

[FBSS80] Thomas Fortmann, Yaakov Bar-Shalom, and Molly Scheffe. Multi-target tracking using joint probabilistic data association. pages 807–812, 1980.

[FKFB05] Wu-Chi Feng, Ed Kaiser, Wu Chang Feng, and Mikael Le Baillif. Panoptes: scalable low-power video sensor networking technologies. *ACM Trans. Multimedia Comput. Commun. Appl.*, 1(2):151–167, May 2005.

[FX09] Xiufang Feng and Zhanwei Xu. A neural data fusion algorithm in wireless sensor network. In *Proceedings of the 2009 Pacific-Asia Conference on Circuits, Communications and Systems*, PACCS '09, pages 54–57, Washington, DC, USA, 2009. IEEE Computer Society.

[GB03] Lipari G. and E. Bini. Resource partitioning among real-time applications. In *ECRTS'03, IEEE Computer Society*, pages 151–158, 2003.

[GGGA07] R. Goshorn, J. Goshorn, D. Goshorn, and H. Aghajan. Architecture for cluster-based automated surveillance network for detecting and tracking multiple persons. pages 219–226, 2007.

[Gu07] Dongbing Gu. Distributed Particle Filter for Target Tracking. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, number April, pages 3856–3861. Ieee, April 2007.

[GW08] R.C. González and R.E. Woods. *Digital Image Processing*. Pearson/Prentice Hall, 2008.

[HA06] Stephan Hengstler and Hamid Aghajan. A smart camera mote architecture for distributed intelligent surveillance. In *In ACM SenSys Workshop on Distributed Smart Cameras (DSC*, 2006.

[HDH09] Ondrej Hlinka, Petar M Djuric, and Franz Hlawatsch. Time-space-sequential distributed particle filtering with low-rate communications. In *2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers*, pages 196–200. Ieee, November 2009.

[HLN09] Sören Höckner, Andreas Lagemann, and Jörg Nolte. Integration of event-driven embedded operating systems into omnet++: a case study with reflex. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, Simutools '09, pages 73:1–73:7, ICST, Brussels, Belgium, Belgium, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[HM06] T.A. Henzinger and S. Matic. An interface algebra for real-time components. In *Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'06)*, pages 253–266, April 2006.

[Hop88] J. J. Hopfield. Neurocomputing: foundations of research. chapter Neural networks and physical systems with emergent collective computational abilities, pages 457–464. MIT Press, Cambridge, MA, USA, 1988.

[Hor91] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Netw.*, 4(2):251–257, March 1991.

[HPH08] Yu-Kai Huang, Ai-Chun Pang, and Hui-Nien Hung. An Adaptive GTS Allocation Scheme for IEEE 802.15.4. *IEEE Transactions on Parallel and Distributed Systems*, 19(5):641–651, 2008.

[IEE06] IEEE. Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). *Control*, 2006(September), 2006.

[ISO99] ISO/IEC. JPEG2000 Requirements and Profile. ISO/IEC JTC1/ SC29/WG1 N1271, March 1999.

[Kal60] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME âĂŽÃĎÃń Journal of Basic Engineering*, (82 (Series D)):35–45, 1960.

[KAT06] a. Koubaa, M. Alves, and E. Tovar. i-GAME: An Implicit GTS Allocation Mechanism in IEEE 802.15.4 for Time-Sensitive Wireless Sensor Networks. *18th Euromicro Conference on Real-Time Systems (ECRTS'06)*, pages 183–192, 2006.

[KATC08] Anis Koubâa, Mário Alves, Eduardo Tovar, and André Cunha. An implicit gts allocation mechanism in ieee 802.15.4 for time-sensitive wireless sensor networks: theory and practice. *Real-Time Syst.*, 39(1-3):169–204, 2008.

[KCA07] Anis Koubaa, Andre Cunha, and Mario Alves. A time division beacon scheduling mechanism for ieee 802.15.4/zigbee cluster-tree wireless sensor networks. In *ECRTS '07: Proceedings of the 19th Euromicro Conference on Real-Time Systems*, pages 125–135, Washington, DC, USA, 2007. IEEE Computer Society.

[Koe80] J. G. Koenig. Indicators of urban accessibility: Theory & application, 1980.

[KS03] Sohaib Khan and Mubarak Shah. Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1355–1360, 2003.

[LBD+01] Jinyang Li, Charles Blake, Douglas S.J. De Couto, Hu Imm Lee, and Robert Morris. Capacity of Ad Hoc wireless networks. *Proceedings of the 7th annual international conference on Mobile computing and networking - MobiCom '01*, pages 61–69, 2001.

[LCR07] J. Liu, M. Chu, and J.E. Reich. Multitarget Tracking in Distributed Sensor Networks, May 2007.

[LL73] C.L. Liu and J.W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the Association for Computing Machinery*, 20(1), 1973.

[Low04] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[Lun10] Lund University. Truetime project. `http://www.control.lth.se/truetime/`, 2010.

[MF02] Aloysius K. Mok and A. K. Feng. A model of hierarchical real-time virtual resources. In *RTSS'02, IEEE Computer Society*, pages 26–35, 2002.

[MGTG11] Yajie Ma, Yike Guo, Xiangchuan Tian, and M. Ghanem. Distributed clustering-based aggregation algorithm for spatial correlated sensor networks. *Sensors Journal, IEEE*, 11(3):641 –648, march 2011.

[MMM03] E. Magli, M. Mancin, and L. Merello. Low-complexity video compression for wireless sensor networks. In *Proceedings of the 2003 International Conference on Multimedia and Expo - Volume 3 (ICME '03) - Volume 03*, ICME '03, pages 585–588, Washington, DC, USA, 2003. IEEE Computer Society.

[MMN+11] M. Magrini, D. Moroni, C. Nastasi, P. Pagano, M. Petracca, G. Pieri, C. Salvadori, and O. Salvetti. Visual sensor networks for infomobility. *Pattern Recognit. Image Anal.*, 21(1):20–29, March 2011.

[MP09] Henry Medeiros and Johnny Park. *Multi-Camera Networks: Concepts and Applications*, chapter Cluster-Based Object Tracking by Wireless Camera Networks, pages 539–572. Elsevier, 2009.

[MRX08] Satyajayant Misra, Martin Reisslein, and Guoliang Xue. A survey of multimedia streaming in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 10(4):18–39, 2008.

[NC11] Christian Nastasi and Andrea Cavallaro. Distributed target tracking under realistic network conditions. In *Sensor Signal Processing for Defence 2011 (SSPD 2011)*, 2011.

[NMS+10] C. Nastasi, M. Marinoni, L. Santinelli, P. Pagano, G. Lipari, and G. Franchino. Baccarat: a dynamic real-time bandwidth allocation policy for ieee 802.15.4. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, pages 406 –412, 29 2010-april 2 2010.

[Nov63] Albert B. Novikoff. On convergence proofs for perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume 12, pages 615–622, 1963.

[NYM08] C Na, Y Yang, and a Mishra. An optimal GTS scheduling algorithm for time-sensitive transactions in IEEE 802.15.4 networks. *Computer Networks*, 52(13):2543–2557, September 2008.

[ope] Open-ZB open-source toolset for the IEEE 802.15.4/ZigBee protocols web site. `http://www.open-zb.net`.

[opn] OPNET Technologies, Inc., Bethesda, MD, USA. The OPNET Simulator. `http://www.opnet.com/`.

[ose] OSEK/VDX Portal. `http://www.osek-vdx.org/`.

[OsFM07] Reza Olfati-saber, J. Alex Fax, and Richard M. Murray. Consensus and cooperation in networked multi-agent systems. In *Proceedings of the IEEE*, 2007.

[OSS08] R. Olfati-Saber and N.F. Sandell. Distributed tracking in sensor networks with limited sensing range. In *American Control Conference*, 2008.

[PCL+10] Paolo Pagano, Mangesh Chitnis, Giuseppe Lipari, Christian Nastasi, and Yao Liang. Simulating Real-Time Aspects of Wireless Sensor Networks. *EURASIP Journal on Wireless Communications and Networking*, 2010(i):1–20, 2010.

[PGS+11] M. Petracca, M. Ghibaudi, C. Salvadori, P. Pagano, and D. Munaretto. Performance evaluation of fec techniques based on bch codes in video streaming over wireless sensor networks. In *Proceedings of the 2011 IEEE Symposium on Computers and Communications*, ISCC '11, pages 43–48, Washington, DC, USA, 2011. IEEE Computer Society.

[Pha10] Congduc Pham. A video sensor simulation model with omnet++. `http://web.univ-pau.fr/~cpham/WSN-MODEL/wvsn.html`, 2010.

[PLL+02] Luigi Palopoli, Giuseppe Lipari, Gerardo Lamastra, Luca Abeni, Gabriele Bolognini, and Paolo Ancilotti. An object oriented tool for simulating distributed real-time control systems. *Software: Practice and Experience*, 2002.

[PPL09] Paolo Pagano, Francesco Piga, and Yao Liang. Real-time multi-view vision systems using wsns. In *SAC*, pages 2191–2196, 2009.

[PSOB05] Georgiy Pekhteryev, Zafer Sahinoglu, Philip Orlik, and Ghulam Bhatti. Image transmission over IEEE 802.15.4 and zigbee networks. In *In Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 23–26, 2005.

[QKR+07] Markus Quaritsch, Markus Kreuzthaler, Bernhard Rinner, Horst Bischof, and Bernhard Strobl. Autonomous multicamera tracking on embedded smart cameras. *EURASIP J. Embedded Syst.*, 2007(1):35–35, 2007.

[QT08] Faisal Qureshi and Demetri Terzopoulos. A simulation framework for camera sensor networks research. In *Proceedings of the 11th communications and networking simulation symposium*, CNS '08, pages 41–48, New York, NY, USA, 2008. ACM.

[RAAKR05] Richard J. Radke, Srinivas Andra, Omar Al-Kofahi, and Badrinath Roysam. Image change detection algorithms: A systematic survey. *IEEE Transactions on Image Processing*, 14:294–307, 2005.

[RB93] Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *IEEE INTERNATIONAL CONFERENCE ON NEURAL NETWORKS*, pages 586–591, 1993.

[RBI+05] Mohammad Rahimi, Rick Baer, Obimdinachi I. Iroezi, Juan C. Garcia, Jay Warrior, Deborah Estrin, and Mani Srivastava. Cyclops: In situ image sensing and interpretation in wireless sensor networks. In *In SenSys*, pages 192–204. ACM Press, 2005.

[Rei79] Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24:843–854, 1979.

[Rem04] P. Remagnino. Distributed intelligence for multi-camera visual surveillance. *Pattern Recognition*, 37(4):675–689, April 2004.

[RGGN07] Anthony Rowe, Adam G Goode, Dhiraj Goel, and Illah Nourbakhsh. Cmucam3: An open programmable embedded vision sensor. Technical Report CMU-RI-TR-07-13, Robotics Institute, Pittsburgh, PA, May 2007.

[Ros88] F. Rosenblatt. Neurocomputing: foundations of research. chapter The perception: a probabilistic model for information storage and organization in the brain, pages 89–114. MIT Press, Cambridge, MA, USA, 1988.

[Row08] a Rowe. RT-Link: A global time-synchronized link protocol for sensor networks. *Ad Hoc Networks*, 6(8):1201–1220, November 2008.

[RW09] B. Rinner and W. Wolf. *Multi-Camera Networks: Concepts and Applications*, chapter Toward Pervasive Smart Camera Networks, pages 483–496. Elsevier, 2009.

[SBAS04] Nisheeth Shrivastava, Chiranjeeb Buragohain, Divyakant Agrawal, and Subhash Suri. Medians and beyond: new aggregation techniques for sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 239–249, New York, NY, USA, 2004. ACM.

[SCCC06] K Shih, C Chang, Y Chen, and T Chuang. Dynamic bandwidth allocation for QoS routing on TDMA-based mobile ad hoc networks. *Computer Communications*, 29(9):1316–1329, May 2006.

[SCN+10] Luca Santinelli, Mangesh Chitnis, Christian Nastasi, Fabio Checconi, Giuseppe Lipari, and Paolo Pagano. A component-based architecture for adaptive bandwidth allocation in wireless sensor networks. In *SIES*, pages 174–183, 2010.

[SHR05] Xiaohong Sheng, Yu-Hen Hu, and Parameswaran Ramanathan. Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network. *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, pages 181–188, 2005.

[SKS+10] Bi Song, Ahmed T Kamal, Cristian Soto, Chong Ding, Jay a Farrell, and Amit K Roy-Chowdhury. Tracking and activity recognition through consensus in distributed camera networks. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 19(10):2564–79, October 2010.

[sma] SMART DUST: Autonomous sensing and communication in a cubic millimeter. `http://robotics.eecs.berkeley.edu/\~pister/SmartDust/`.

[SPM96] Amir Said, William A. Pearlman, and Senior Member. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6:243–250, 1996.

[SSC09] Cristian Soto, Bi Song, and Amit K. Roy Chowdhury. Distributed multi-target tracking in a self-configuring camera network. In *CVPR*, pages 1486–1493. IEEE, 2009.

[ST10] Babak Shirmohammadi and Camillo J. Taylor. Distributed target tracking using self localizing smart camera networks. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, ICDSC '10, pages 17–24, New York, NY, USA, 2010. ACM.

[TC11a] M. Taj and A. Cavallaro. Distributed and decentralized multicamera tracking. *Signal Processing Magazine, IEEE*, 28(3):46 –58, May 2011.

[TC11b] Murtaza Taj and Andrea Cavallaro. Distributed and decentralized multicamera tracking. *IEEE Signal Process. Mag.*, 28(3):46–58, 2011.

[TCN00] L. Thiele, S. Chakraborty, and M. Naedele. Real-time calculus for scheduling hard real-time systems. In *ISCAS*, volume 4, pages 101–104, 2000.

[TGMB+08] Andreas Timm-Giel, Ken Murray, Markus Becker, Ciaran Lynch, Carmelita Gorg, and Dirk Pesch. Comparative Simulations of WSN. In *ICT-MobileSummit*, 2008.

[tin] University of California, Berkeley CA, USA). The TinyOS operating system. `http://www.tinyos.net/`.

[TJH07] J. Trdlička, M. Johansson, and Z. Hanzálek. Optimal Flow Routing in Multi-hop Sensor Networks with Real-Time Constraints through Linear Programming. In *12th IEEE International Conference on Emerging Technologies and Factory Automation*, pages –, Piscataway, 2007. Institute of Electrical and Electronic Engineers.

[TMP+09] L. Tessens, M. Morbee, W. Philips, R. Kleihorst, and H. Aghajan. Efficient approximate foreground detection for low-resource devices. In *2009 Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC)*, pages 1–8. Ieee, August 2009.

[TS08] T. Teixeira and a. Savvides. Lightweight People Counting and Localizing for Easily Deployable Indoors WSNs. *IEEE Journal of Selected Topics in Signal Processing*, 2(4):493–502, August 2008.

[TXC05] Jian Tang, Guoliang Xue, and Christopher Chandler. Interference-aware routing and bandwidth allocation for qos provisioning in multihop wireless networks: Research articles. *Wirel. Commun. Mob. Comput.*, 5(8):933–943, 2005.

[vDL03] Tijs van Dam and Koen Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 171–180, New York, NY, USA, 2003. ACM.

[VJ04] Paul Viola and Michael J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, May 2004.

[VSC+08] S. Velipasalar, J. Schlessman, C. Chen, W. Wolf, and J. Singh. A scalable clustered camera system for multiple object tracking. *EURASIP Journal on Image and Video Processing*, 2008.

[WLR+07] Hao Wen, Chuang Lin, Fengyuan Ren, Yao Yue, and Xiaomeng Huang. Retransmission or redundancy: Transmission reliability in wireless sensor networks. In *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*, pages 1–7, Oct. 2007.

[WVC10] Y Wang, S Velipasalar, and M Casares. Cooperative object tracking and composite event detection with wireless embedded smart camera. *IEEE Transaction on Image Processing*, 2010.

[www10a] Inet framework for omnet++. `http://inet.omnetpp.org/`, 2010.

[www10b] Mixim project for omnet++. `http://mixim.sourceforge.net/`, 2010.

[www10c] Ns-2 project. `http://nsnam.isi.edu/nsnam/index.php/Main_Page`, 2010.

[www10d] OMNeT++ Project. `http://www.omnetpp.org/`, 2010.

[www10e] Opnet. `http://www.opnet.com/`, 2010.

[www11a] IPERMOB project: A Pervasive and Heterogeneous Infrastructure to control Urban Mobility in Real-Time. `http://www.ipermob.org/`, 2011.

[www11b] WiSE-MNet. `http://www.eecs.qmul.ac.uk/~andrea/wise-mnet.html` and `http://rtn.sssup.it/index.php/software/wise-mnet`, 2011.

[xbo] Crossbow. `http://www.xbow.com`.

[YGM08] Tingxin Yan, Deepak Ganesan, and R. Manmatha. Distributed image search in camera sensor networks. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, pages 155–168, New York, NY, USA, 2008. ACM.

[YGS+04] Haifeng Yu, Phillip B. Gibbons, Srinivasan Seshan, Suman Nath, and Suman Nath. Synopsis diffusion for robust aggregation in sensor networks. In *In SenSys*, pages 250–262. ACM Press, 2004.

[YMG08] J Yick, B Mukherjee, and D Ghosal. Wireless sensor network survey. *Computer Networks*, 52(12):2292–2330, August 2008.

[YSV04] Wei Yu, Z. Sahinoglu, and A. Vetro. Energy efficient JPEG 2000 image transmission over wireless sensor networks. *GLOBECOM '04. IEEE Global Telecommunications Conference (IEEE Cat. No.04CH37615)*, Vol.5, 2004.

[ZC02] Chenxi Zhu and M.Scott Corson. QoS routing for mobile ad hoc networks. In *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 07921, pages 958–967. Ieee, 2002.