ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

ADDIS ABABA INSTITUTE OF TECHNOLOGY

ELECTRICAL & COMPUTER ENGINEERING DEPARTMENT

Design and Performance Analysis of Energy Efficient Technique for Wireless Multimedia Sensor Networks using Machine Learning Algorithms

By

Kibrewerk Akalu

Advisor

Dr. Kumudha Raimond

A thesis submitted to the School of Graduate Studies of Addis Ababa University in partial fulfillment of the requirements for the degree of Masters of Science in Computer Engineering

February 2011

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

ADDIS ABABA INSTITUTE OF TECHNOLOGY

ELECTRICAL & COMPUTER ENGINEERING DEPARTMENT

Design and Performance Analysis of Energy Efficient Technique for Wireless Multimedia Sensor Networks using Machine Learning Algorithms

By

Kibrewerk Akalu

Advisor

Dr. Kumudha Raimond

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

Design and Performance Analysis of Energy Efficient Technique for Wireless Multimedia Sensor Networks using Machine Learning Algorithms

By

Kibrewerk Akalu

ADDIS ABABA INSTITUTE OF TECHNOLOGY

APPROVAL BY BOARD OF EXAMINERS

Dr. Getahun Mekuria	_____
Chairman, Dept. of Graduate	Signature
Committee	
Dr.Kumudha Raimond	_____
Advisor	Signature
Mr.Yalemzewd Negash	_____
Internal Examiner	Signature
Dr.Ing. Hailu Ayele	_____
External Examiner	Signature

Abstract

Wireless multimedia sensor networks (WMSNs) are developed from wireless sensor networks (WSNs) for acquiring and transmitting multimedia data such as images, audio and video streams and scalar data.

Energy is the most critical factor in sensor networks. Its power requirement is satisfied by low capacity and low power battery. One of the reasons is the requirement of unattended operation in remote or even potentially hostile locations, sensor networks are extremely energy-limited. This constraint demands that techniques must not only be efficient but energy conscious as well, which requires new approach to addressing the common but substantial issues.

Reduction of communicated multimedia volume is an important step to reduce energy consumption in WMSNs because of the relatively huge amount of data collected by the nodes compared with scalar sensors. One of the algorithms in machine learning which can reduce the dimensionality is unsupervised learning Artificial Neural Networks which typically perform dimensionality reduction through pattern clustering. In this thesis, an attempt has been made to reduce the amount of transmitted information in WMSNs using vector quantization technique using Self Organizing Map (SOM) algorithm in order to increase the lifetime of the network.

In the Proposed Design, SOM is used to generate a codebook using single image and batch image training methods. An energy model has been designed to calculate the lifetime of the nodes taking into consideration the computational energy cost and communication energy cost. Using this energy model, the codebook size has been optimized to a size of 50 codewords through which the network lifetime has shown an increase of 158.03 percentages compared to the existing design. This amount of increase in the lifetime of the WMSNs is on a graceful-tradeoff with the image quality since the main purpose of sensor networks is the occurrence or non occurrence of things of interest rather than on excellence of image quality considering a surveillance which is the main application for the deployment of the sensors.

Keywords – Wireless Sensor Networks, Wireless Multimedia Sensor Networks, Machine Leaning Algorithms, Self Organizing Map, Codebook.

Acknowledgment

First and foremost I would like to say thank you to God who has given me the strength to endure all the difficulties and challenges of life.

Secondly, I would like to thank my advisor Dr.Kumudha Raimond for her endless patience in reading , correcting and giving me insightful ideas to improve the work and the constant motivation and support during the course of the work. I truly appreciate and value her guidance and encouragement from the commencement to the end of this thesis.

Finally, I must thank my amazing family and friends. They are the greatest blessing in my life and special thanks goes to my wife , Dana, for her patience and understanding while I have been spending long hours on this work.

Table of Contents

List of Figures

List of Tables

Abbreviations and Acronyms

2D DCT	Two dimensional discrete cosine Transform
DWT	Discrete Wavelet Transform
FND	First Node Dies
GSP	Gossip-Based Sleep Protocol
JPEG	Joint Photographic Experts Group
LEACH	Low Energy Adaptive Clustering Hierarchy
LND	Last Node Dies
MTE	Minimum Transmission Energy
PSNR	Peak Signal to Noise Ratio
RLE	Run Length Encoding
S-JPEG	Squared - JPEG
VS	Visual Sensor
VSN	Visual Sensor Network
WMSN	Wireless Multimedia Sensor Network
WSN	Wireless Sensor Network

1. Introduction

Due to advances in wireless communications and electronics over the past years, the development of low-cost, low-power, multifunctional wireless sensors have received increasing attention. These sensors are small in size and able to sense, process data, and communicate with each other, typically over an RF (radio frequency) channel. A sensor network is designed to detect events or phenomena, collect and process data, and transmit sensed information to interested users.

WSNs have recently been the focus of the research community. The main motivation has been to address the challenges posed by WSN paradigm, i.e., limited node power, communication capabilities, dense network, and multi-hop communications [1].

The availability of low-cost hardware such as CMOS cameras and microphones has fostered the development of WMSNs , i.e., networks of wirelessly interconnected devices that are able to ubiquitously retrieve multimedia content such as video and audio streams, still images, and scalar sensor data from the environment. WMSN applications, e.g., multimedia surveillance networks, target tracking, environmental monitoring, and traffic management systems, require effective harvesting and communication of event features in the form of multimedia such as audio, image, and video.

With rapid improvements and miniaturization in hardware, a single sensor device can be equipped with audio and visual information collection modules. As an example, the Cyclops is a camera sensor consisting of a CMOS camera module and is designed for extremely light-weight imaging and can be interfaced with a host mote such as Crossbow's MICA2 or MICAz. It is designed with power economy in mind. Each component sleeps deeply when it is in no use.

Today as we learn how to integrate sensors in our environment we still face many challenges with imaging sensor technology. To couple imagers with the environment with high degree of longevity, we need low power image capturing and processing capability. Emergence of CMOS imaging with low power consumption and moderate imaging quality brings such dreams closer

to realm of reality. These imagers are widely used in electronic gadgets such as cell phones and typically include a lens, an image sensor, and image processing in a small package. In spite of this high level of integration, these devices are still difficult to use in typical lightweight nodes. Image data must be captured synchronously at high rates of speed, and programming is complex [57].

In addition to the ability to acquire multimedia data, WMSN is also able to store, process in real-time, correlate and fuse multimedia data originating from heterogeneous sources. For performing in-network processing on the raw data extracted from the environment WMSN requires new architectures for collaborative, distributed, and resource-constrained processing that allow for filtering and extraction of semantically relevant information at the edge of the sensor network [2]. WMSN could be deployed in a number of ways to monitor a given area of interest.

Different scenarios for the deployment of WMSNs are shown in Figure 1.1 from [2], where three sensor networks with different characteristics are shown, possibly deployed in different physical locations. The first cloud on the left shows a single-tier network of homogeneous video sensors. Subsets of the deployed sensors have higher processing capabilities, and is thus referred to as processing hubs. The union of the processing hubs constitutes a distributed processing architecture. The multimedia content gathered is relayed to a wireless gateway through a multi-hop path. The gateway is interconnected to a storage hub that is in charge of storing multimedia content locally for subsequent retrieval. Clearly, more complex architectures for distributed storage can be implemented when allowed by the environment and the application, which may result in energy savings since by storing it locally, the multimedia content does not need to be wirelessly relayed to remote locations. The wireless gateway is also connected to a central sink, which implements the front-end software for network querying and tasking.

The second cloud represents a single-tiered clustered architecture of heterogeneous sensors (only one cluster is depicted). Video, audio, and scalar sensors relay data to a central cluster head, which is also in charge of performing intensive multimedia processing on the data (processing hub). The cluster head relays the gathered content to the wireless gateway and to the storage hub.

The last cloud on the right represents a multi-tiered network, with heterogeneous sensors. Each tier is in charge of a subset of the functionalities. Resource-constrained, low-power scalar sensors are in charge of performing simpler tasks, such as detecting scalar measurements, while resource-rich, high-power devices are responsible for more complex tasks. Data processing and storage can be performed in a distributed fashion at each different tier [2].

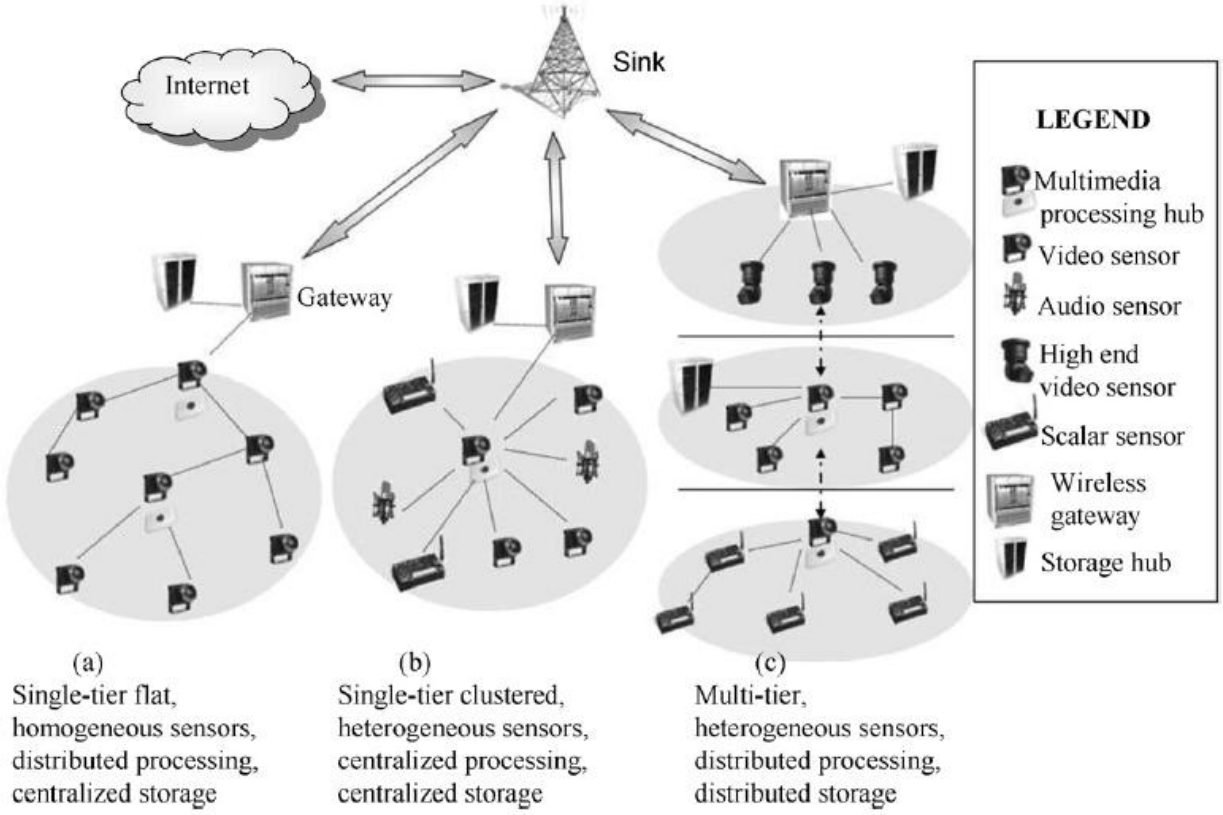

Figure 1.1 WMSN deployment scenarios

1.1 Motivation

Energy is a valuable resource in WMSNs. The status of energy consumption should be continuously monitored after network deployment. The information about energy status can be used to notify both sensor nodes and network delayers about resource depletion in some parts of the network. It can also be used to perform energy-efficient routing in WMSNs. Since more energy is consumed during communication, reducing the amount of transmitted information through in-network processing has a significant impact [15].

In WMSN, information processing at sensor nodes has been recognized as powerful methods to reduce the load on the network and to prolong its lifetime. The processing power required to process multimedia content can be obtained through coordinated use of multiple sensor nodes. The processing can be carried out to aggregate correlated multimedia streams, to compress data, or extract application-specific information. As an example, two still images obtained from overlapping areas can be collated to generate a smaller volume image using image registration algorithms. Similarly, images obtained from multiple cameras can be processed in the network to deliver the location of an object rather than raw image data [16].

The motivation is that, WMSNs are generally deployed in remote areas where no infrastructure is available. This imposes the use of battery operated devices which seriously limits the lifetime of the network. This energy depletion is more serious in case of WMSN as it transfers voluminous amount of multimedia data than WSN. This demands to strive for a suitable approach to reduce the amount of data transferred. Many researchers have addressed this problem and worked on routing algorithms to get energy efficient way of relaying the sensed information to the data sink. However, few works have been done on reducing the data size to increase the lifetime of sensor nodes to the best of our knowledge. One such work is using an adaptive resonance theory to cluster the data that could be sent and instead of sending the whole data only the cluster number will be sent by knowing the nature of the sensed information beforehand [59]. This work has indicated that machine learning algorithms are best approaches to reduce the amount of information being sent in WMSNs. And no such work is done using neural network approach for WMSNS to the best of our knowledge and hence this research has been motivated to use such an approach.

The other motivation to perform in-network processing in WMSNs is the idea that in-network filtering and processing techniques can help to conserve the scarce energy resources and analytical result proves that communicating 1 bit over the wireless network is equivalent to performing approximately 1000 CPU instructions [55][56].

And hence using the above facts it can be seen that to improve the lifetime of WMSNs some of the algorithms developed within neural-networks and in machine learning in general for over the years, are well suited to fit into the requirements imposed to sensor networks for: simple parallel distributed computation, distributed storage, data robustness and auto-classification of sensor

readings. As a result of the dimensionality reduction that can be obtained easily from the outputs of the neural-networks clustering algorithms, lower communication costs and thus bigger energy savings can be obtained.

1.2 Objective

General Objective:

- To analyze the performance of a design for WMSNs by exploiting the properties of machine learning algorithms for dimensionality reduction tasks in sensor networks.

Specific Objectives:

1. To investigate the existing energy minimization algorithms and models for WMSNs.

2. To analyze the effectiveness of machine learning algorithms for dimensionality reduction.

3. To propose, design and simulate a solution to minimize the energy required for transmission.

4. To formulate a computational energy cost model for the proposed design.

5. To evaluate the performance of the proposed approach using a given routing protocol.

6. To compare the proposed design with an existing design through simulation.

1.3 Methodology

Literature survey in the area:

After enough literature review on previous works in the area and examining different algorithms a design of vector quantization based on neural network has been choosen.

Analysis and Modeling:

The contribution of this proposed design for increasing the lifetime of WMSN is analyzed by simulation.

Mechanism of Driving Conclusion:

The simulation result of the new energy minimizing design will be compared with a chosen existing design. Matlab simulation tool is used for implementation of the works of neural network and simulation of the WMSN.

1.4 Related Works

Cluster-based routing algorithms are generally divided into two categories: centralized and distributed algorithms. One of the most popular distributed algorithms is Low-Energy Adaptive Clustering Hierarchy (LEACH) [17]. With its cluster head election algorithm together with a cluster head rotation mechanism and data fusion within each cluster head, LEACH manages to reduce the energy dissipation by a factor of 7 when compared to direct communication and by a factor of $4 - 8$ when compared to MTE (minimum transmission energy) [18]. Whilst there are advantages of using LEACH's distributed cluster formation algorithm, this protocol offers no guarantee on the placement and/or number of elected cluster head nodes. This may lead to inefficient setups which do not exploit the full potential of the algorithm posing a limit on the achievable network lifetime. In contrast, a centralized clustering system, such as LEACH-C [17], uses its global knowledge of the network to produce better clusters by dispersing the cluster head nodes throughout the network. During the cluster setup phase of LEACH-C, each node sends information about its current location (determined using a GPS receiver) and energy level to the base station. The energy level is used to ensure that the energy load is evenly distributed among all the nodes in the network whereas the location information provides for global knowledge of the network. This algorithm minimizes the energy dissipation of the sensor nodes while transmitting their data to the cluster head by minimizing the total sum of squared distances between all the non-cluster head nodes and the closest cluster head. This leads to an improvement in network lifetime of 62.5% (in terms of FND) when compared to LEACH [8].

Although centralized cluster-based algorithms tend to yield a better performance when compared to their distributed counterparts, this improvement comes at the expense of having sensor nodes equipped with extra hardware and a high energy cost network initialization phase. In order to find a good compromise between these two approaches, the authors in [19] have presented a self

organizing map (SOM) neural network-based clustering algorithm whereby the base station collects network topological information through a low energy cost network initialization phase. The reported SOM algorithm elects an optimal number of cluster heads which are well dispersed throughout the network, reducing the energy dissipation and improving energy balancing between the nodes. Results presented show that an improvement in network lifetime of around 57.2% (in terms of FND) is achieved when compared to LEACH [19]. Apart from reducing the transmission distance of each sensor node, it is also imperative to reduce the amount of transmitted information.

Vincent et al. [20] proposed a self-adaptive image transmission scheme driven by energy efficiency considerations in order to be suitable for WSNs. It is based on wavelet image transform and semi-reliable transmission to achieve energy conservation. Wavelet image transform provides data decomposition into multiple levels of resolution, so the image can be divided into packets with different priorities. Semi-reliable transmission enables priority-based packet discarding by intermediate nodes according to their battery's state of charge. Such image transmission approach provides a graceful trade-off between the reconstructed images quality and the sensor nodes' lifetime. In their work, an analytical study in terms of dissipated energy is performed to compare the self-adaptive image transmission scheme to a fully reliable scheme. Since image processing is computationally intensive and operates on large data set, the cost of the wavelet image transform is considered in the energy consumption analysis. In their analysis, they evaluated the proposed protocol using parameters derived from the Mica2 Crossbow motes characteristics. They have considered a transmission power of 0dBm and a power supply provided by two AA batteries (3 volts). The considered image in their scenario is an 8-bpp monochrome image of 128 X 128 pixels. And it has been given in their simulation, with one and two Discrete Wavelet Transform (DWT) and 50 intermediate nodes, the consumed energy is of about 247 and 87mJ corresponding to a decrease of 72 and 90% respectively of the consumed energy when no DWT is applied (877mJ).

According to [21], the problem of modeling and adapting JPEG to the energy requirement of visual sensor networks (VSN) is addressed. And an energy-aware adapted version of JPEG called squared JPEG(S-JPEG) in their terminology has been proposed. S-JPEG exploits the energy compaction property of the two-dimensional discrete cosine transform (2D DCT), to

reduce both the redundant data within the sent image and the energy dissipated by a source node. S-JPEG addresses the VSN applications that face severe energy constraints and do not need a full image quality at the sink, such as video surveillance. They have developed an energy model for both JPEG and S-JPEG. And they have used a simplified routing protocol to validate their theory. After conducting a set of simulations to show the performance the S-JPEG improves JPEG energetically, while maintaining an adequate image quality at the sink.

Yang Xiaobo et al. [22] argue that because of the high bandwidth demands of multimedia data, the transmission of raw data collected at sensor nodes consumes a large amount of resources. And they have proposed a distributed image compression as a means of overcoming the computation and energy limitation of individual nodes by sharing the processing tasks.In their work they have argued that it can prolong the overall lifetime of network by assigning compression tasks to idle nodes. In this paper a clustering approach for WMSNs based on LEACH protocol is used. And by assigning image compression tasks to nodes other than the image capture node, this algorithm can achieve efficient image compression in WMSN with limited energy. However, this work has this limitation of an assumption that a given image sensor node sends data to cluster head in a cluster and all the other nodes are idle. And it also assumes that the initial energy of all the sensor nodes is different.And under the mentioned conditions they have been able to show their algorithm extends the network life compared with JPEG centralized compression by nearly 100 percent.

In the work of Heinzelman [23], it is presented that the energy consumption rate for sensors in a WSN varies greatly based on the protocols the sensors use for communications.They have used the Gossip-based sleep protocol(GSP),with this protocol each node goes to sleep for some time with gossip sleep probability p. And the main focus of their work is the implementation of GSP on the mica2 platform and simulations were conducted to determine the improvement in network lifetime. Results for energy consumption, transmitted and received power, minimum voltage supply required for operation,effect of transmission power on energy consumption, different methods for measuring lifetime of a sensor node are presented.

One of the contributions of the work in [23] is that measured results that were used to derive a general expression of energy consumption in GSP protocol can be extended to other protocols, provided that all the time the radio spends in different states is known. And comparing to the analytical model presented by Heinzelman et al. in [17], energy consumption found in this work is two orders of magnitude larger,including the consumption of the microcontroller. Contrary to the ideas in [17] , measurements show that energy consumed in transmitting one bit is almost twice the energy in receiving one bit. The difference between the measurement model presented here and the Heinzelman model is that these measurements use a fixed energy level for transmissions. The Heinzelman model implicitly assumes perfect power control, i.e., a perfect knowledge of distance between the sender and receiver, a continuously variable transmission power level and no channel fading.

In the experimental platform of [23], CPU energy consumption in active state is three orders of magnitude smaller than energy spent in transmitting or receiving a bit. So, in this case CPU can execute roughly one thousand instructions with an energy equivalent to transmit just one bit. Energy consumption differs when different transmission power levels are used. And the idea and theory of this work are a good contribution to this thesis work.

1.5 Scope

Multimedia data include snapshot and streaming multimedia content. Snapshot-type multimedia data contain event triggered observations obtained in short time period (e.g. still image). Streaming multimedia content is generated over long time period, requires sustained information delivery and typically needs to be delivered in real time. The snapshot multimedia can initiate the streaming, in the sense, when a given static image is obtained from the sensor node, the base station can request a streaming video or audio instead of continuously transmitting video or audio. And hence, the focus of this thesis is on still images.

Since the intention is to reduce the amount of transmitted information in WSNs we will only focus on data reduction techniques and limit the work to a given routing algorithm to determine the amount of used energy in the given nodes. The energy consumption of the different states such as idle state ,sensing , deep sleep and other energy consumption states that consume the same amount of energy are not considered since they are not a differential factor of the designs

that will be analyzed. And only the transmission and reception energy consumption in addition to computational energy consumption are considered in this work. And it is also assumed that the nodes are always active and they have data to send.

1.6 Contribution

The contribution of this work is the use and the analysis of energy minimizing algorithm by compressing the size of the image to increase the lifetime by graceful trade-off with the image quality since WMSNs are mostly used in survillance applications and the occurence and the non occurence of the things is the critical question rather than the quality of the image. And hence by including the overhead computational energy cost and the communication energy cost it is shown that the lifetime of the network has increased. And as an additional contribution the energy cost of the proposed algorithm is formulated and its computational cost is calculated. In general even though the design of image compressing algorithm, vector quantization, used in this work is used in other works for other purposes , it has never been tried for the WMSNs which contains nodes that are energy constrained as far as it can be seen in an extensive literature survey to the best of our knowledge. And the vector quantization with the codebook size of 50 which is found to be optimal for this work after rigorous experimentation has never been tried also.

1.7 Thesis Organization

In Section 2, the background theory and general feature of WMSN is described besides an energy consumption issue. In Section 3 the overview of neural network is presented , overview of Mica2 mote is introduced in section 4. Sections 5 and 6 deeply explore the designs of both the existing and the proposed respectively.Section 7 shows the performnace analysis and the last section explains the conclusion and the future work.

2. Overview of WMSN

2.1 Background

Building a WSN first of all requires the constituting nodes to be developed and available. These nodes have to meet the requirements that come from the specific requirements of a given application: they might have to be small, cheap, or energy efficient, they have to be equipped with the right sensors, the necessary computation and memory resources, and communication facilities. The reason for this chapter to be included and discussed is to understand the inner details of WMSNs so that the features of the hardware can be used to derive energy consumptions for simulations purposes.

2.1.1 Hardware components

When choosing the hardware components for a wireless sensor node, evidently the application's requirements play a decisive factor with regard mostly to size, costs, and energy consumption of the nodes – communication and computation facilities as such are often considered to be of acceptable quality, but the trade-offs between features and costs is crucial. A basic sensor node comprises of five main components (shown in the Figure 2.1).

Controller: A controller to process all the relevant data, capable of executing arbitrary code.

Memory: Some memory to store programs and intermediate data; usually, different types of memory are used for programs and data.

Sensors and actuators: The actual interface to the physical world: devices that can observe or control physical parameters of the environment.

Communication: Turning sensor nodes into a network requires a device for sending and receiving information over a wireless channel.

Power supply: As usually no tethered power supply is available, some forms of batteries are necessary to provide energy.

The components are discussed in detail below.

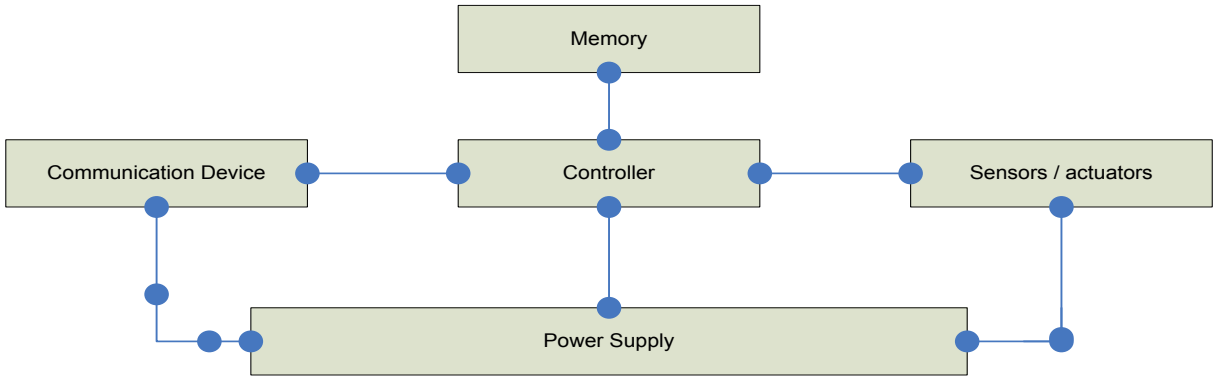

Figure 2.1 Overview of main sensor hardware components

A. Controller

The controller is the core of a wireless sensor node. It collects data from the sensors, processes these data, decides when and where to send it, receives data from other sensor nodes, and decides on the actuator's behavior. It has to execute various programs, ranging from time-critical signal processing and communication protocols to application programs; it is the central processing unit (CPU) of the node.

Such a variety of processing tasks can be performed on various controller architectures, representing trade-offs between flexibility, performance, energy efficiency, and costs. One solution is to use general-purpose processors, like those known from desktop computers. These processors are highly overpowered, and their energy consumption is excessive. But simpler processors do exist, specifically geared towards usage in embedded systems. These processors are commonly referred to as microcontrollers.

Some of the key characteristics that make these microcontrollers particularly suited to embedded systems are their flexibility in connecting with other devices (like sensors), their instruction set amenable to time-critical signal processing, and their typically low power consumption. They are also convenient as they often have built in memory. In addition, they are freely programmable and hence very flexible. Microcontrollers are also suitable for WSNs since they commonly have the possibility to reduce their power consumption by going into sleep states where only parts of

the controller are active. Details vary considerably between different controllers. The Atmel ATmega 128L is an 8-bit microcontroller, also intended for usage in embedded applications and equipped with relevant external interfaces for common peripherals. This microcontroller is used with Crossbow's MICA2 sensor mote.

B. Memory

The memory component is fairly straightforward. Evidently, there is a need for Random Access Memory (RAM) to store intermediate sensor readings, packets from other nodes, and so on. While RAM is fast, its main disadvantage is that it loses its content if power supply is interrupted. Program code can be stored in Read-Only Memory (ROM) or, more typically, in Electrically Erasable Programmable Read-Only Memory (EEPROM) or flash memory (the latter being similar to EEPROM but allowing data to be erased or written in blocks instead of only a byte at a time). Flash memory can also serve as intermediate storage of data in case RAM is insufficient or when the power supply of RAM should be shut down for some time.

C. Communication device

The communication device is used to exchange data between individual nodes. For actual communication, both a transmitter and a receiver are required in a sensor node. The essential task is to convert a bit stream coming from a microcontroller (or a sequence of bytes or frames) and convert them to and from radio waves. For practical purposes, it is usually convenient to use a device that combines these two tasks in a single entity. Such combined devices are called transceivers. Usually, half-duplex operation is realized since transmitting and receiving at the same time on a wireless medium is impractical in most cases (the receiver would only hear its own transmitter anyway). Some transceivers can directly provide control over transmission power to be used; some require some external circuitry for that purpose. Usually, only a discrete number of power levels are available from which the actual transmission power can be chosen. Maximum output power is usually determined by regulations. And this helps to limit transmission range which is dependent on transmission power. However, the range is considered in absence of interference; it evidently depends on the maximum transmission power, on the antenna characteristics, on the attenuation caused by the environment, which in turn depends on the carrier frequency used, on the modulation/coding scheme that is used, and on the bit error

rate that one is willing to accept at the receiver. It also depends on the quality of the receiver, essentially its sensitivity. Typical values are difficult to give here, but prototypes or products with ranges between a few meters and several hundreds of meters are available. And also the signal strength at which an incoming data packet has been received can provide useful information (e.g. a rough estimate about the distance from the transmitter assuming the transmission power is known); a receiver has to provide this information in the Received Signal Strength Indicator (RSSI).

D. Sensors and actuators

Without the actual sensors and actuators, a WSN would be beside the point entirely. But as the discussion of possible application areas has already been indicated, the possible range of sensors is vast. Sensors can be roughly categorized into three categories:

Passive, omnidirectional sensors: These sensors can measure a physical quantity at the point of the sensor node without actually manipulating the environment by active probing – in this sense, they are passive. Moreover, some of these sensors actually are self-powered in the sense that they obtain the energy they need from the environment – energy is only needed to amplify their analog signal. There is no notion of "direction" involved in these measurements. Typical examples for such sensors include thermometer, light sensors, vibration, microphones, humidity, mechanical stress or tension in materials, chemical sensors sensitive for given substances, smoke detectors, air pressure, and so on.

Passive, narrow-beam sensors: These sensors are passive as well, but have a well-defined notion of direction of measurement. A typical example is a camera, which can "take measurements "in a given direction, and can be rotated if needed.

Active sensors: This last group of sensors actively probes the environment, for example, a sonar or radar sensor or some types of seismic sensors, which generate shock waves by small explosions. These are quite specific – triggering an explosion is certainly not a lightly undertaken action – and require quite special attention.

Actuators are just about as diverse as sensors, yet for the purposes of designing a WSN, they are a bit simpler to take account of: In principle, all that a sensor node can do is to open or close a switch or a relay or to set a value in some way. Whether this controls a motor, a light bulb, or some other physical object is not really of concern to the way communication protocols are designed.

E. Power supply of sensor nodes

For untethered wireless sensor nodes, the power supply is a crucial system component. There are essentially two aspects: First, storing energy and providing power in the required form; second, attempting to replenish consumed energy by "scavenging" it from some node-external power source over time. Storing power is conventionally done using batteries. As a rough orientation, a normal AA battery stores about 2.2–2.5 Ah at 1.5 V. Battery design is a science and industry in itself and energy scavenging has attracted a lot of attention in research.

2.1.2 Energy consumption of sensor nodes

As the previous section has shown, energy supply for a sensor node is at a premium: batteries have small capacity, and recharging by energy scavenging is complicated and volatile. Hence, the energy consumption of a sensor node must be tightly controlled. The main consumers of energy are the controller, the radio front ends depending on the type of the sensors.

One important contribution to reduce power consumption of these components comes from chip-level and lower technologies: Designing low-power chips is the best starting point for an energy-efficient sensor node. But this is only one half of the picture, as any advantages gained by such designs can easily be squandered when the components are improperly operated.

The crucial observation is that most of the time a wireless sensor node has nothing to do. Hence, it is best to turn it off. Naturally, it should be able to wake up again, on the basis of external stimuli or on the basis of time. Therefore, completely turning off a node is not possible, but rather, its operational state can be adapted to the tasks at hand. Introducing and using multiple states of operation with reduced energy consumption in return for reduced functionality is the core technique for energy-efficient wireless sensor node.

Different models usually support different numbers of such sleep states with different characteristics. For a controller, typical states are “active”, “idle”, and “sleep”; a radio modem could turn transmitter or receiver, or both on or off; sensors and memory could also be turned on or off. The usual terminology is to speak of a “deeper” sleep state if less power is consumed. While such a graded sleep state model is straightforward enough, it is complicated by the fact that transitions between states take both time and energy. The usual assumption is that the deeper the sleep state, the more time and energy it takes to wake up again to fully operational state (or to another, less deep sleep state). Hence, it may be worthwhile to remain in an idle state instead of going to deeper sleep states even from an energy consumption point of view.

Radio transceivers

A radio transceiver has essentially two tasks: transmitting and receiving data between a pair of nodes. Similar to microcontrollers, radio transceivers can operate in different modes; the simplest ones are being turned on or turned off. To accommodate the necessary low total energy consumption, the transceivers should be turned off most of the time and only be activated when necessary – they work at a low duty cycle. But this incurs additional complexity, time and power overhead that have to be taken into account. To understand the energy consumption behavior of radio transceivers and their impact on the protocol design, models for the energy consumption per bit for both sending and receiving are required.

Modeling energy consumption during transmission

In principle, the energy consumed by a transmitter is due to two sources: one part is due to RF signal generation, which mostly depends on chosen modulation and target distance and hence on the transmission power Ptx, that is, the power radiated by the antenna. A second part is due to electronic components necessary for frequency synthesis, frequency conversion, filters, and so on. These costs are basically constant. The energy to transmit a packet of n-bits long (including all headers) then depends on how long it takes to send the packet, determined by the nominal bit rate and the coding rate, and on the total consumed power during transmission.

Modeling energy consumption during reception

Similar to the transmitter, the receiver can be either turned off or on. While being turned on, it can either actively receive a packet or can be idle, observing the channel and ready to receive. Evidently, the power consumption while it is turned off is negligible. Even the difference between idling and actually receiving is very small and can, for most purposes, be assumed to be zero.

Relationship between computation and communication

Looking at the energy consumption numbers for both microcontrollers and radio transceivers, an obvious question to ask is which is the best way to invest the precious energy resources of a sensor node. Is it better to send data or to compute? What is the relation in energy consumption between sending data and computing? Details about this relationship heavily depend on the particular hardware in use, but a few rule-of-thumb figures can be given here. For several hardwares, the ratio of the energy consumption to send one bit compared to computing a single instruction is between 1500 to 2700 for Rockwell WINS nodes, between 220 to 2900 for MEDUSA II nodes, and about 1400 for WINS NG 2.0 nodes [6]. Hill et al. [7] note, for the RFM TR1000 radio transceiver, 1 μJ to transmit a single bit and 0.5 μJ to receive one; their processor takes about 8 nJ per instruction. This results in a (actually quite good) ratio of about 190 for communication to computation costs. In a slightly different perspective, communicating 1 kB of data over 100 m consumes roughly the same amount of energy as computing three million instructions [8].

Disregarding the details, it is clear that communication is a considerably more expensive undertaking than computation. Still, energy required for computation cannot be simply ignored; depending on the computational task, it is usually still smaller than the energy for communication, but still noticeable. This basic observation motivates a number of approaches and design decisions for the networking architecture of WSNs. The core idea is to invest in computation within the network whenever possible to save on communication energy costs, leading to the notion of in-network processing and aggregation [3].

2.1.3 Typical example of sensor nodes

There are quite a number of actual nodes available for use in WSN research and development. Again, depending on the intended application scenarios, they have to fulfill quite different requirements regarding battery life, mechanical robustness of the node's housing, size, and so on. The mica mote is chosen as a typical example of motes.

The "Mica Mote" family

Starting in the late 1990s, an entire family of nodes has evolved out of research projects at the University of California at Berkeley, partially with the collaboration of Intel, over the years. They are commonly known as the Mica motes, with different versions (Mica, Mica2, Mica2Dot) having been designed [9][10][11]; references [12, 13] have an overview table of the family members; schematics for some of these designs are available from [14]. They are commercially available via the company Crossbow in different versions and different kits. TinyOS is the operating system usually used for these nodes. All these boards feature a microcontroller belonging to the Atmel family, a simple radio modem and various connections to the outside. In addition, it is possible to connect additional "sensor boards" with, for example, barometric or humidity sensors, to the node as such, enabling a wider range of applications and experiments. Sensors are connected to the controller via an I2C bus or via SPI, depending on the version [3].

2.1.4 Routing Protocol

Several routing protocols are used for WMSN nodes to relay their information to the sink nodes. LEACH, the widely used routing algorithm is used in this work. The LEACH protocol (Low-energy Adaptive Clustering Hierarchy) presented by Heinzelmann et al. [17] assumes a dense sensor network of homogeneous, energy-constrained nodes, which shall report their data to a sink node. LEACH partitions the nodes into clusters and in each cluster a dedicated node, the cluster head, is responsible for creating and maintaining a schedule; all the other nodes of a cluster are member nodes. The operation of LEACH is divided into rounds. Each round begins with a set-up phase when the clusters are organized , followed by a steady-state phase when several frames of data are transfered from the nodes to the cluster-head and on to the base station. LEACH forms clusters by using a distributed algorithm, where nodes make autonomous

distances without any centralized control. In order to minimize overhead, the steady-state phase is long compared to the set-up phase. The following elaborates LEACH in detail.

1. Advertisement Phase/cluster setup phase/

Initially, when clusters are being created, each node decides whether or not to become a cluster-head for the current round. This decision is based on the suggested percentage of cluster heads for the network (determined a priori) and the number of times the node has been a cluster-head so far. This decision is made by the node n by choosing a random number between 0 and 1. If the number is less than a threshold T(n), the node becomes a cluster-head for the current round. The threshold is set as:

$$T(N) = \begin{cases} \frac{P}{1-P*(r \bmod \frac{1}{P})} : if\ n \in G \\ 0 : \qquad otherwise \end{cases} \tag{2.1}$$

where P = the desired percentage of cluster heads, r is the current round, and G is the set of nodes that have not been cluster-heads in the last 1/P rounds. Using this threshold, each node will be a cluster-head at some point within 1/P rounds. During round 0 (r = 0), each node has a probability P of becoming a cluster-head. The nodes that are cluster-heads in round 0 cannot be cluster-heads for the next 1/P rounds. Thus the probability that the remaining nodes are cluster-heads must be increased, since there are fewer nodes that are eligible to become cluster-heads. After 1/P -1 rounds, T=1 for any nodes that have not yet been cluster-heads, and after 1/P rounds,all nodes are once again eligible to become cluster-heads.

In this work, it is assumed that all nodes begin with the same amount of energy and being a cluster-head removes approximately the same amount of energy for each node. Each node that has elected itself as a cluster-head for the current round broadcasts an advertisement message to the rest of the nodes. For this “cluster-head-advertisement” phase, the cluster-heads use a CSMA MAC protocol, and all cluster-heads transmit their advertisement using the same transmit energy.

The non-cluster-head nodes must keep their receivers on during this phase of set-up to hear the advertisements of all the cluster-head nodes. After this phase is complete, each non-cluster-head node decides the cluster to which it will belong for this round. This decision is based on the received signal strength of the advertisement. Assuming symmetric propagation channels, the cluster-head advertisement heard with the largest signal strength is the cluster-head to whom the minimum amount of transmitted energy is needed for communication. In the case of ties, a random cluster-head is chosen.

2. Steady state phase/transmission phase/

After each node has decided to which cluster it belongs, it must inform the cluster-head node that it will be a member of the cluster. Each node transmits this information back to the cluster-head again using a CSMA MAC protocol. During this phase, all cluster-head nodes must keep their receivers on.

I. Schedule Creation

The cluster-head node receives all the messages for nodes that would like to be included in the cluster. Based on the number of nodes in the cluster, the cluster head node creates a TDMA schedule telling each node when it can transmit. This schedule is broadcast back to the nodes in the cluster.

II. Data Transmission

Once the clusters are created and the TDMA schedule is fixed, data transmission can begin. Assuming nodes always have data to send, they send it during their allocated transmission time to the cluster head. This transmission uses a minimal amount of energy (chosen based on the received strength of the cluster-head advertisement). The radio of each non-clusterhead node can be turned off until the node's allocated transmission time,thus minimizing energy dissipation in these nodes. The cluster-head node must keep its receiver on to receive all the data from the nodes in the cluster.This is the steady-state operation of LEACH networks.After a certain time, which is determined a priori, the next round begins with each node determining if it should be a cluster-head for this round and advertising this information.

3. Overview of Neural Network

An artificial neural network is an information processing system that has certain performance characteristics in common with biological neural networks. Artificial neural networks have been developed as generalizations of mathematical models of human cognition or neural biology, based on the assumptions that:

1. Information processing occurs at many simple elements called neurons.
2. Signals are passed between neurons over connection links.
3. Each connection link has an associated weight, which, in a typical neural net, multiplies the signal transmitted.
4. Each neuron applies an activation function (usually nonlinear) to its net input (sum of weighted input signals) to determine its output signal.

A neural network is characterized by (1) its pattern of connections between the neurons (called its architecture), (2) its method of determining the weights on the connections (called its training, or learning algorithm), and (3) its activation function.

A neural net consists of a large number of simple processing elements called neurons, units, cells, or nodes. Each neuron is connected to other neurons by means of directed communications links, each with associated weight. The weights represent information being used by the net to solve a problem. Neural nets can be applied to a wide variety of problems, such as storing and recalling data or patterns, classifying patterns, performing general mappings from input patterns to output patterns, grouping similar patterns, or finding solutions to constrained optimization problems [27].

How do neural networks learn? First off, we must define learning. Biologically, learning is an experience that changes the state of an organism such that the new state leads to an improved performance in subsequent situations. Mechanically, it is similar: Computational methods for acquiring and organizing new knowledge that will lead to new skills. Before the network can become useful, it must learn about the information at hand. After training, it can then be used for pragmatic purposes. In general, there are two flavors of learning

- ✓ **Supervised Learning:** The correct answers are known and this information is used to train the network for the given problem. This type of learning utilizes both input vectors and output vectors. The input vectors are used to provide the starting data, and the output vectors can be used to compare with the input vectors to determine some error. In a special type of supervised learning, reinforcement learning, the network is only told if its output is right or wrong. Back-propagation algorithms make use of this style.

- ✓ **Unsupervised Learning:** The correct answers are not known (or just not told to the network). The network must try on its own to discover patterns in the input data. The input vectors are used solely. Output vectors generated will not be used to learn from. Also and possibly most importantly: no human interaction is needed for unsupervised learning. This can be an extremely important feature, especially when dealing with a large and/or complex data set that would be time-consuming or difficult for a human to compute. Self-Organizing Maps utilize this style of learning.

As can be seen from the above descriptions, a neural network can have many different features compared to other neural network algorithms. Because of this, there are many different types of neural networks. One in particular, the Self-Organizing Map, is discussed next.

3.1 Self Organizing Map

A self-organizing map or self-organizing feature map (SOFM) is a type of artificial neural network that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discredited representation of the input space of the training samples, called a map. Generally, SOFMs learn to classify input vectors according to how they are grouped in the input space into clusters. Clustering is the process of grouping the data into classes (clusters) so that the data objects are similar to one another within the same cluster and dissimilar to the objects in other clusters. Clustering is unsupervised classification which means that there are no predefined classes [28].

Kohonen Self-Organizing Maps (or just Self-Organizing Maps, or SOMs for short), are a type of neural network. They were developed in 1982 by Tuevo Kohonen, a professor emeritus of the Academy of Finland. SOMs are aptly named, "Self-Organizing" as no supervision is required. SOMs learn on their own through unsupervised competitive learning. "Maps" is because they

attempt to map their weights to conform to the given input data. The nodes in a SOM network attempt to become like the inputs presented to them. In this sense, this is how they learn. They can also be called “Feature Maps”, as in Self-Organizing Feature Maps. Retaining principal 'features' of the input data is a fundamental principle of SOMs, and one of the things that makes them so valuable. Specifically, the topological relationships between input data are preserved when mapped to a SOM network. This has a pragmatic value of representing complex data. For example, the weight vector for a cluster unit serves as an exemplar of the input patterns associated with that cluster. During the self-organization process, the cluster unit whose vector matches the input pattern most closely (typically, the minimum of the squared Euclidean distance) is chosen as the winner. The winning unit and its neighboring units (in terms of the topology of the cluster units) update their weights. The architecture and algorithm that follow for the net can be used to cluster a set of p continuous-valued vectors $X_{1\ldots P}$ = (x1, x2. . . xi, . . . ,xn) into m clusters.

Another intrinsic property of SOMs is known as vector quantization. This is a data compression technique. SOMs provide a way of representing multidimensional data in a much lower dimensional space – typically one or two dimensions.

3.2 Architecture and algorithm

The architecture of the Kohonen self-organizing map is shown in Figure 3.1.

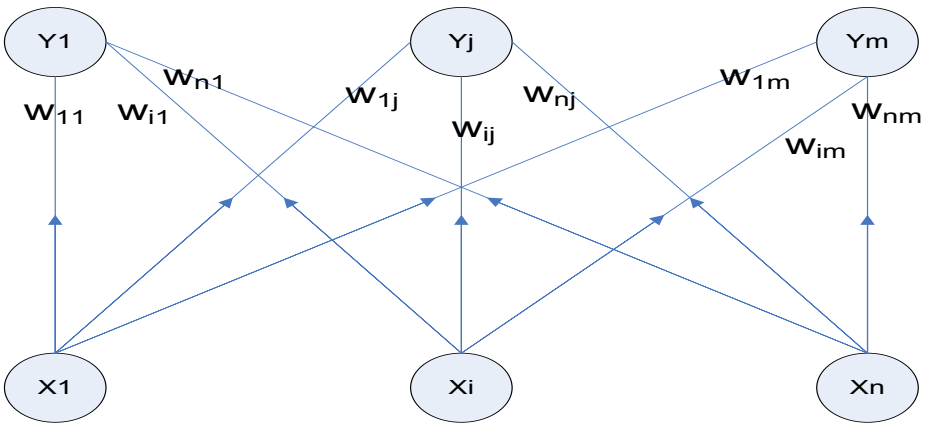

Figure 3.1 Kohonen self-organizing map

SOFM learn to classify input vectors according to how they are grouped in the input space. The algorithm used for SOFM is as explained in several steps in the figure 3.2.

The weight vector for a cluster unit serves as an exemplar of the input patterns associated with that cluster. During the self-organization process, the cluster unit whose weight vector matches the input pattern most closely (typically, the minimum of the squared Euclidean distance) is chosen as the winner. The winning unit and its neighboring units (in terms of the topology of the cluster units) update their weights. This updating process can be continued for certain number of epochs or to any stopping criteria. The weight vectors of neighboring units are not, in general, close to the input pattern. The architecture and the algorithm that are mentioned in this section can be used to cluster a set of continuous-valued vectors into clusters.

Random values may be assigned for the initial weights. If some information is available concerning the distribution of clusters that might be appropriate for a particular problem, the initial weights can be taken to reflect that prior knowledge [17].

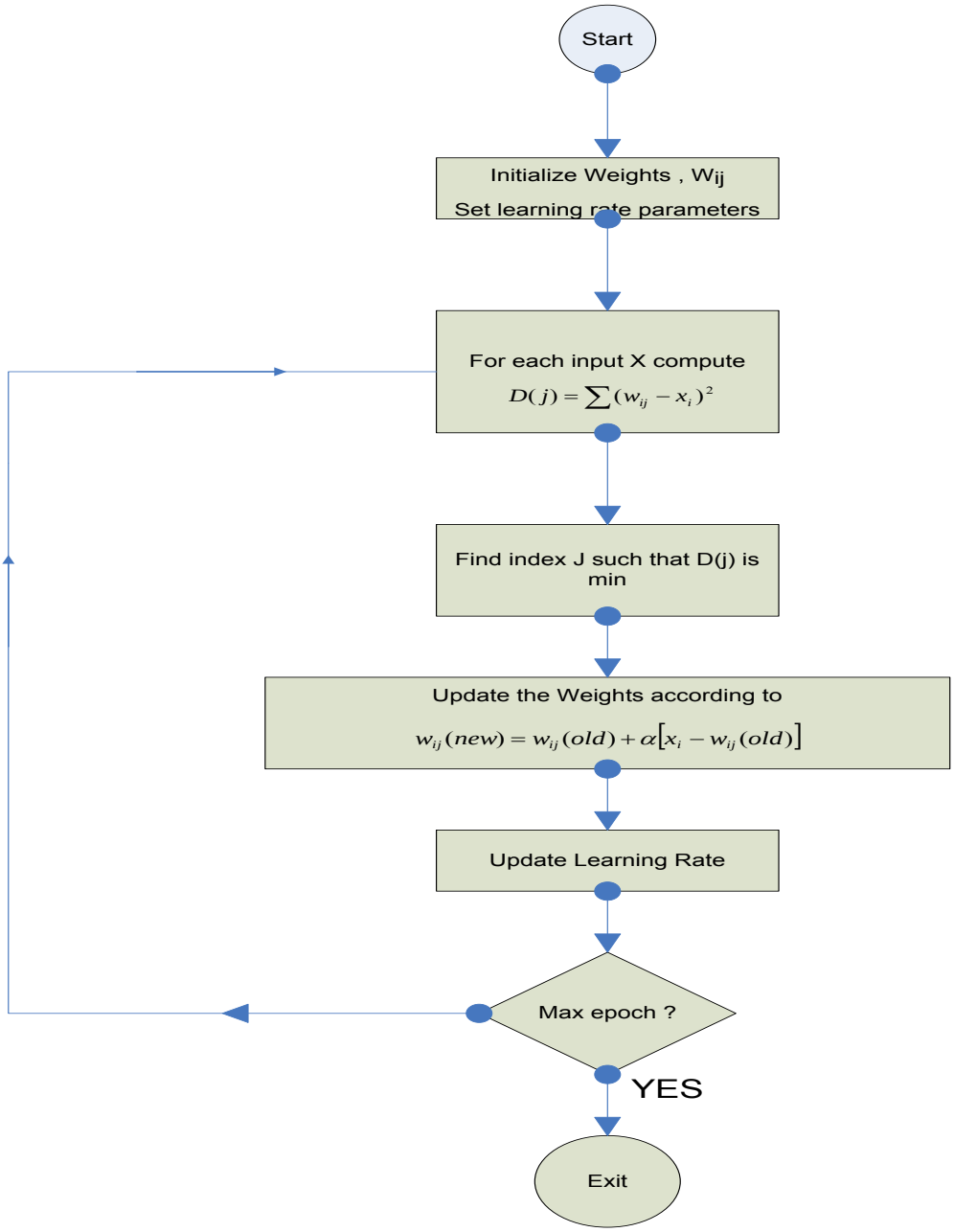

Figure 3.2 Algorithm for training SOFM

4. Overview of Mica2 Motes

4.1 Mica2 Mote

In this work, the focus is on sensor networks based on small, low-power sensors such as the Mica2 sensor node, developed by Crossbow Technology Inc. The MICA2 Mote is a third generation mote module used for enabling low-power, wireless, sensor networks in the generation of Mica motes. It is designed specifically for deeply embedded sensor networks. It includes an expansion connector for light, Temperature, Barometric, Pressure, Acceleration/Seismic, Acoustic, Magnetic and other crossbow sensor boards. The reason for this chapter to be included and discussed is to understand the inner details of mica motes so that energy consumption models could be derived for simulation purposes.

The first challenge in simulating the power consumption of sensor network devices is obtaining an accurate, detailed model of a typical node's power consumption. Extensive literature surveys have been conducted to get the power profiling of the Mica2 platform using the standard Mica2 sensor board. The CC1000 radio supports programmable transmission power levels ranging from −20 dBm to + 5 dBm [39].

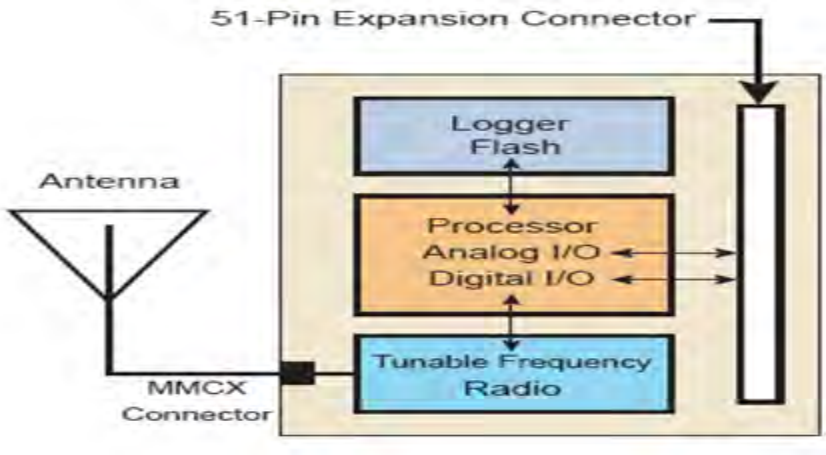

Figure 4.1 Main Characteristics of MPR400CB Processor/Radio Board [24]

The Mica2 mote is used in several applications of WSNs such as security, surveillance and environmental monitoring. In figure 4.1 a typical Mica2 mote board MPR400CB is shown and a summary of the characterstics of a given Mica2 mote is shown in table 4-1.

Table 4-1 Parameters of Mica 2 mote

Mote Type	Mica2
Microcontroller Type	Atmega128L
CPU Clock(MHz)	8
Program Memory(KB)	128
RAM(KB)	4
Non-Volatile storage Size(KB)	512
Radio Communication	Chipcom CC1000
Frequency (MHz)	916/433
Transmit Power Control	Programmable via CC1000 registers -20 to +5 dBm

The MICA2 Mote features several new improvements over the original MICA Mote. The following features make the MICA2 better suited to commercial deployment:

1. 868/916 MHz, 433 MHz or 315 MHz multi-channel transceiver with extended range
2. TinyOS (TOS) Distributed Software Operating System v1.0 with improved networking stack and improved debugging features
3. Support for wireless remote reprogramming
4. Wide range of sensor boards and data acquisition add-on boards

TinyOS 1.0 is a small, open-source, energy efficient, software operating system developed by UC Berkeley (Berkeley University of California) which supports large scale, self configuring sensor networks. The usually used processor and radio platform module in Mica2 mote is MPR400CB. The MPR400CB is based on the Atmel Atmega 128L. The Atmega128L is a low-power microcontroller which runs TOS from its internal flash memory. Using TOS, a single processor board (MPR400CB) can be configured to run sensor application/processing and the network/radio communications stack simultaneously. The Mica2 51-pin expansion connector supports analog inputs, Digital I/O, I2C ,SPI and UART interfaces. These interfaces make it easy to connect to a wide variety of external peripherals. Crossbow offers a variety of sensor and data

acquisition boards for the mica2 mote. All of these boards connect to the Mica2 via the standard 51-pin expansion connector. Custom sensor and data acquisition boards are also available.
A base station allows the aggregation of sensor network data onto a PC or other computer platform. Any Mica2 mote can function as a base station when it is connected to a standard PC interface or gateway board. The MIB510CA/MIB520CA provides a serial/USB interface for both programming and data communications.

Crossbow Mica2 motes were employed as sensor nodes in the experimental simulation test bed. A typical Mica mote is shown in figure 4.2. Its characterstics has been incorporated as simulation parameters. The Mica2 sensors receive energy from two AA alkaline batteries and use an Atmel Atmega 128L microcontroller circuit and the CC1000 integrated radio circuit [4]. According to Crossbow, the module should be powered with DC voltage between 2.7 and 3.3 Volts [29].

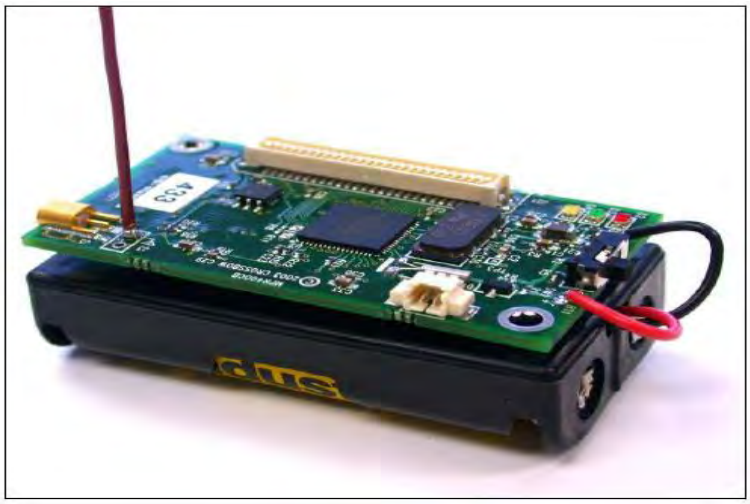

Figure 4.2 Mica2 Mote manufactured by Cross Bow Technology [30]

Because of the small physical size of the mote, the usual antenna chosen is a length of insulated wire called the monopole whip antenna one – quarter wavelength long. The applications written on computer using nesC for Mica2 mote are transferred to the mote using a programming board.

4.2 Energy Consumption Model

In order to incorporate the energy model of Mica2 mote the energy consumption of the two main energy consuming models have to be studied based on extensive literature surveys. The two most important energy consuming modules of the Mica2 mote are:

1. Radio Module energy consumption
2. Microcontroller energy consumption(The Atmel Atmega128L)

The other energy consuming activities such as sensing and taking an image are ignored for simlicity purposes since their addition will not make any difference for the comparative study of the performance analysis of the designs which is the target of this work. Therefore the two models will be discussed in the upcoming sections.

4.2.1 Radio Module Energy consumption

Nodes consume energy when they are transmitting or receiving. We need to know the amount of energy consumed by our nodes in order to simulate later the effect that this energy consumption can have over the sensors. More specifically, this will affect the design since in the moment that a sensor node runs out of energy and it can no longer transmit packets. The CC1000 radio device energy consumption is studied, because it is the one used by MICA2 motes and it is the one that will be used during the evaluation of the proposed design.

This section presents the radio propagation model mentioned in several literatures. Radio propagation is the behavior of radio waves when they are transmitted, or propagated from one point on the Earth to another, or into various parts of the atmosphere. Like light waves, radio waves are affected by the phenomena of reflection, refraction, diffraction, absorption, polarization and scattering [54].

Radio propagation is affected by the daily changes of water vapor in the troposphere and ionization in the upper atmosphere, due to the Sun. Understanding the effects of varying conditions on radio propagation has many practical applications. A radio propagation model, also known as the Radio Wave Propagation Model or the Radio Frequency Propagation Model, is an empirical mathematical formulation for the characterization of radio wave propagation as a function of frequency, distance and other conditions. A single model is usually developed to predict the behavior of propagation for all similar links under similar constraints. However, the works of the authors mentioned in the discussion below are chosen since their work is based on experimental evidences and instead of using a mathematical formulation the energy cost and the distance coverage is used [54].

Most of the clustering protocols described assume that sensor nodes can adjust their power according to the exact distance. Hence the corresponding energy consumption computed in this manner will always be different for two different measurements of distance'd'. In reality, the transmit power level of the sensor node can only be adjusted to discrete values which may result in one power level for multiple values of distance. Therefore the resulting energy consumption for the two different distances would be same [45].

This section introduces a radio model, which dynamically determines which power level setting should be used to transmit between two nodes. Using the power level setting, the cost of transmissions is calculated based on the chip specifications to ensure an accurate estimation. Received Signal Strength Indication (RSSI) is used to determine which power level setting is needed to transmit directly between two nodes. In most work, authors assume that the power level can be adjusted to the exact needs and calculate the energy cost using these exact values. In reality this is not the case as the radio can only be adjusted to one of the associated power levels and not set to the exact transmission power needed. Using the assumption that there is an infinite amount of transmission levels, previous work makes the assumption that the longer links will cost more to transmit a packet. In many situations two links of different lengths will need to transmit at the same power level setting in order for the packet to be received and therefore the cost to transmit over different distances can be equivalent. In the work of [54] it is shown that instead of infinite power levels, the radio must transmit at a set power level according to the chip specifications. Now that the required power level has been calculated for transmission between two nodes one can determine the transmission cost in joules. Joules are the unit that most simulators use to when determining battery lifetime.

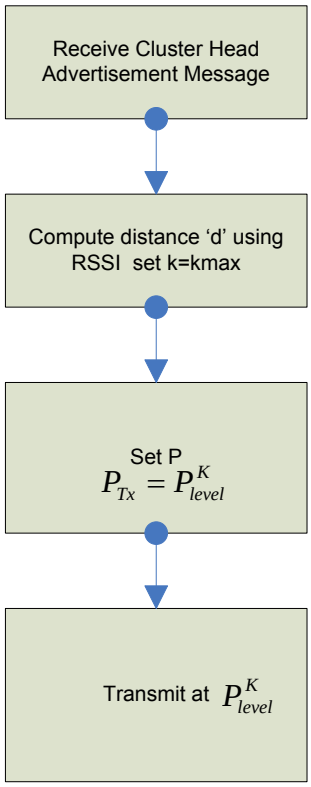

Figure 4.3 Flowchart for discrete power level selection

In the operational model used for most clustering protocols the advertisement messages from Cluster Heads (CHs) are sent at the fixed (known) power levels. The cluster membership phase involves sensor nodes to adjust their power levels according to their distance from the CH. Then the algorithm allows sensor nodes to select the appropriate power level for communicating to the CH. Once sensor node has estimated the distance to the CH, it then sets it's transmit power level to the appropriate value. The level of power could be chosen as shown in figure 4.3.

Recent relevant studies, [48][49][50], have focused on incorporating the realistic radio and energy consumption models in WSNs. In WSNs, the signal is transmitted on RF channel. It will be influenced by environmental factors such as interferences, reflections, scattering and shadowing. Due to the influence of these factors the signal may not reach the receiver node with enough power level and the packet could be dropped by the MAC layer. The power level necessary to detect a signal in the receiver node is called the sensitivity threshold. The received

power depends on the amount of power transmitted by the sender node and on the path loss. There are different models to predict the received signal strength and its degradation with the distance [16], [17], [18]. As explained in the section above the nodes consume energy when they are transmitting, receiving or idle. In our simulator we will not consider the power consumption when the nodes are idle because it affects every node in the same way and it would not be a differential factor. Initially, all nodes are working with their batteries completely charged.

In this work, the power consumption data from the CC1000 radio device has been used as reference for simulation because it is the one used by MICA2 motes. The total energy cost depends on the number of sent or received bytes in each packet.

During the simulation the nodes will decrease their energy level every time they receive or transmit a message, according to table 4-2. If a node runs out of all its available energy, it will no longer appear like a working node in the network. In [27] Polastre et al. proposed a model that presents the total energy consumption for Mica2 as the summation of energy transmitting, receiving, listening, sampling data and sleeping. Values are calculated using the expected consumption of the CPU and the radio, which can be found in specific datasheets [31]. Table 4-2 presents a revised energy consumption model for Mica2 mote [54]. By measuring energy consumption in Mica2 motes, this model shows that the energy consumption in transmission and reception should be higher than values in the classic radio model [23]. Thus, this analysis will use this energy consumption model values.

Table 4-2 Crossbow Mica2 Mote Measured Energy Consumption in Joules/Bit

	5 dBm	0 dBm	-20 dBm
Transmit	4.28 μJoules/bit	3.07 μJoules/bit	2.35 μJoules/bit
Receive	2.36 μJoules/bit	2.21 μJoules/bit	2.35 μJoules/bit

Table 4-3 Mica2 average transmission range for different power levels

Tx Power(dBm)	Transmission Range(m)
-20	5
0	50
5	around 70

The table 4-3 presents the result of an experimental analysis conducted by Anastasi et al. [40], the transmission range and the power levels. According to the transmission range of nodes in our simulation scenarios; we set the transmission and reception power levels.

4.2.2 Microcontroller Energy Consumption

The computational cost in an algorithm depends on the processor used to execute the program and on the number of operations in the code. The processor used in this analysis is Atmel Atmega128L. And the computational cost of the arithmetic operations which are used in the proposed design will be investigated. In this study the computational cost of the design is evaluated by means of its energy consumption, assuming that the energy per CPU cycle is fixed. For an AtMega128L microcontroller, where theoretically the energy per arithmetic instruction (CPU cycle) in this device is 3.3 nJ [41][42]. Details are shown in table 4-4.

Note that this amount of energy is much smaller than the energy used to send or receive a byte, when compared to the above section. Even if we used another microcontroller, the order of magnitude of the energy will be more or less the same compared with the energy used to transmit signals, and this demonstrates that the main issue concerning constrained energy usage will always be the radio communication, and not the algorithm processing.

We will give the nodes a fixed energy at the beginning of the simulation, and we will decrease it after every action they make. When the energy from a node falls below a threshold, the node will stop working, and we will have to remove it from the simulation.

The simple microcontrollers used on current sensor node designs consume approximately constant power while executing arithmetic instructions. Therefore, to compute CPU energy usage it is adequate to track the amount of time the CPU spends in each power state. The amount of time that a node spends in idle mode depends on external factors, such as the timing. Determining CPU execution time can be accomplished by simulating the execution of each instruction, although doing so for large sensor networks would incur a tremendous performance penalty.

The next step is to map each basic block to the appropriate number of CPU instructions as executed by the Atmega128L on the Mica2. To obtain CPU cycle counts, the number of cycles for each instruction in the basic block (obtained from the ATmega128L data sheet [31]) is totaled. In cases where an instruction can consume a variable number of cycles, the expected number of cycles for each instruction is used [41][42].

Table 4-4 Parameters for Mica2 motes

Variables	Description	Value
V	Voltage provided by the power source of the ith node	3V
C(TX) (0)	Current consumed for the radio of the ith node for sending 1 byte (with 0 dBm)	20mA
C(RX)	Current consumed for the radio of the ith node for receiving 1 byte	15mA
T(TX)	Time Spent for the radio of the ith node for sending 1 byte	416E-6s
T(RX)	Time Spent for the radio of the ith node for receiving 1 byte	416E-6s
ε(shift)	Energy consumed for a microcontroller to execute a shift operation over 1 byte	3.3 nJ
ε(add)	Energy consumed for a microcontroller to execute an addition operation over 1 byte	3.3 nJ

5. Existing Design

Wireless Multimedia sensor Network is an ad-hoc WSN made of visual sensors (VS) incorporating image retrieving, processing and communication functionalities. WMSN is gaining attention in a wide range of applications such as video-surveillance, object detection/tracking and environmental monitoring [53]. One of the biggest challenges facing WMSN is a huge amount of visual information processed and sent by each sensor node. This information requires important resources such as memory, processor and communication energy. Typically, visual information needs to be compressed using one of the well known compression standards such as JPEG or JPEG2000 to save energy and to prolong the network lifetime [21]. Based on extensive literature surveys it has been seen that JPEG is an image compression design used in most WMSNs and hence for the comparison and performance analysis of the proposed design, JPEG is used as the existing design [21][22]. The design and the computational energy cost of JPEG is discussed and analyzed in the next discussions.

5.1 JPEG Design

JPEG is a commonly used method of lossy compression for digital photography. The compression of an image at the sensor node includes several steps. The image is first captured from the camera. It is then transformed into a format suitable for image compression.

5.1.1 Shifting Mid Range

Each component of the image is first split into 8x8 blocks. Each 8x8 block of each component is converted to a frequency-domain representation using discrete cosine transform(DCT). Before computing the DCT of the 8x8 block, its values are shifted from a positive range to one centered around zero. For an 8-bit image, each entry in the original block falls in the range [0,255]. The mid-point of the range (in this case, the value 128) is subtracted from each entry to produce a data range that is centered around zero, so that the modified range is [-128,127]. This step reduces the dynamic range requirements in the DCT processing stage that follows [58].

5.1.2 Discrete Cosine Transform(DCT)

After shifting the mid range each 8×8 block of each component is converted to a frequency-domain representation, using a normalized, two-dimensional type-II DCT. This is given by [21]:

$$G_{u,v} = \alpha(u)\alpha(v)\sum_{x=0}^{7}\sum_{y=0}^{7} g_{x,y}\cos\left[\frac{\pi}{8}\left(x+\frac{1}{2}\right)u\right]\cos\left[\frac{\pi}{8}\left(y+\frac{1}{2}\right)v\right] \quad (5.1)$$

Where

- u is the horizontal spatial frequency , for the integers $0 \leq u \leq 8$
- v is the vertical spatial frequency ,for the integers $0 \leq v \leq 8$
- $\alpha_p(n) = \begin{cases} \sqrt{\frac{1}{8}}, if\ n = 0 \\ \sqrt{\frac{2}{8}}, otherwise \end{cases}$ *is a normalizing function*
- $g_{x,y}$ is the pixel value at coordinates (x,y)
- $G_{u,v}$ is the DCT coefficient at coordinates (u,v)

The advantage of the DCT is its tendency to aggregate most of the signal in one corner of the result. The quantization step to follow accentuates this effect while simultaneously reducing the overall size of the DCT coefficients, resulting in a signal that is easy to compress efficiently in the entropy stage.

5.1.3 Quantization

The human eye is good at seeing small differences in brightness over a relatively large area, but not so good at distinguishing the exact strength of a high frequency brightness variation. This allows one to greatly reduce the amount of information in the high frequency components. This is done by simply dividing each component in the frequency domain by a constant, and then rounding to the nearest integer. This rounding is the only lossy operation in the whole process if the DCT computation is performed with sufficiently high precision. As a result of this it is typically the case that many of the higher frequency components are rounded to zero, and many of the rest become small positive or negative numbers, which take many fewer bits to represent.

5.1.4 Entropy Coding

Entropy coding is a special form of lossless data compression. It involves arranging the image components in a "zigzag" order employing run-length encoding (RLE) algorithm that groups similar frequencies together. The details of the algorithm are shown in the figure 5.1.

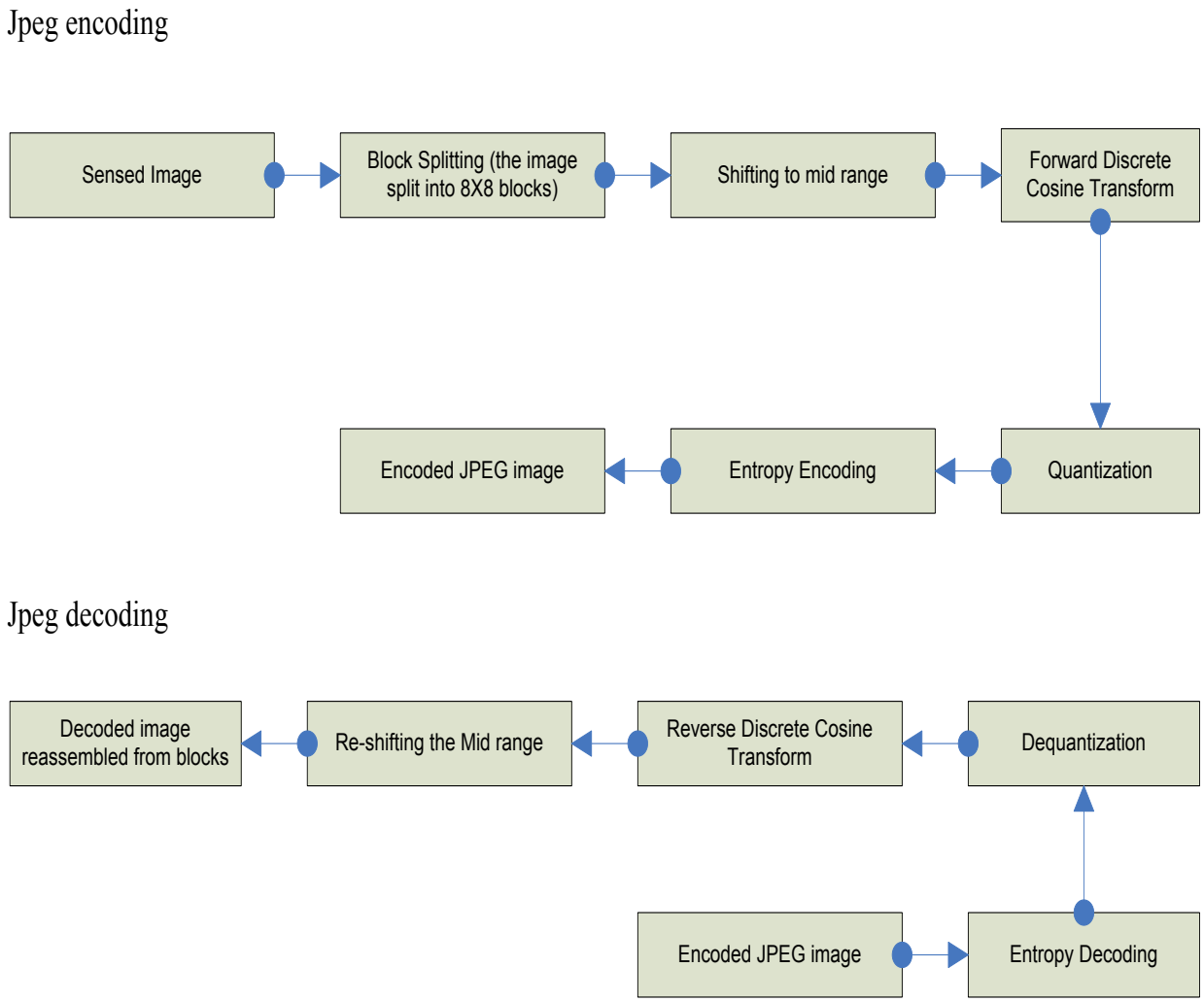

Figure 5.1 Jpeg Encoding and Decoding

5.2 Energy Model of JPEG

An energy consumption model for JPEG compression chain is formulated using a literature surveys and jpeg algorithm. The energy dissipated in compressing the image at the source node is expressed in terms of the number of operations needed at all stages of the compression scheme.

Let E_p be the required energy to process the image data for JPEG stages. Then E_p can be formulated as follows:

$$E_p = E_{shifting} + E_{dct} + E_q + E_{enc} \quad (5.2)$$

Where $E_{shifting}$, E_{dct}, E_q & E_{enc} represent the energies consumed at shifting the mid range, DCT, quantization and entropy encoding stages respectively. The energy consumed for image splitting is neglected for simplification reasons.

5.2.1 Shifting Energy model

Shifting the mid range requires subtracting the constant value from all the entries of the given image. The energy consumption can be obtained using e_{sub} , representing the energy consumption for subtraction instruction, and using an image of size N X N.

$$E_{shift} = (N)^2 e_{sub} \quad (5.3)$$

5.2.2 DCT Energy model

According to [21] an energy consumption model for DCT has been developed based on the general equation 5.1. Therefore, the energy dissipated for a given block of image ,k x k, is $2k^2(ke_{mult} + (k-1)e_{add})$, where e_{mult} and e_{add} represent the energy consumption for mult and add instructions,repectively.For an image of size N X N having $(\frac{N}{k})^2$ blocks, the total energy dissipated in this stage is:

$$E_{dct} = (\frac{N}{k})^2 e_{dct} = \left(\frac{N}{k}\right)^2 2k^2(ke_{mult} + (k-1)e_{add}) \quad (5.3)$$

5.2.3 Quantization Energy Model

According to [21] it has been found that there are k^2 divisions and k^2round operations per block and energy dissipated is $e_q = k^2(e_{div} + e_r)$,where e_{div} and e_r are the energy consumed for div and round instructions,respectively. For one image N X N,the total energy dissipated in this stage is:

$$E_q = (\frac{N}{k})^2 e_q = \left(\frac{N}{k}\right)^2 k^2(e_{div} + e_r) \quad (5.4)$$

5.2.4 Entropy Encoding

5.2.4.1 Zigzagging energy model

This stage can be interpreted as a simple rearrangement of the $(k^2 - 1)$ AC coefficients ordered in a zigzagging way, from lower to higher frequencies. Each AC coefficient is then moved (shifted) to its new postion. The zigzagging energy consumption on each block can be calculated as: $e_z = (k^2 - 1)e_{sh}$, where e_{sh} represents the energy consumption of the shift process. For one input image of size N X N, the total energy consumption related to the zigzagging stage in [21] is:

$$E_z = (\frac{N}{k})^2 e_z = \left(\frac{N}{k}\right)^2 (k^2 - 1)e_{sh} \quad (5.5)$$

5.2.4.2 RLE energy model

Run length encoding of the $(k^2 - 1)$ AC coefficients can be interpreted mainly as a rearrangement of the zigzagged sequence. And this energy consumption is neglected for simplicity due to the less number of operations compared to the other stages.

5.3 Computational cost

The energy of the computation involved in the existing design can be calculated using the energy model formulated above and the energy per bit consumption of the microcontroller using the data sheet [25] and section 4.2. In the table below the energy cost of the operations is shown.

Table 5-1 Energy consumption per operation

Parameter	Value(nJ)
ε (sub)	3.3
ε (mult)	6.6
ε (add)	3.3
ε (div)	6.6
ε (sh)	3.3
ε (r)	3.3

The total energy consumed during JPEG compression can be approximated as the sum of the different operations. Using equation 5.2 which is:

$E_p = E_{shift} + E_{dct} + E_q + E_{enc}$ and this gives

$$E_p = (N)^2 e_{sub} + \left(\frac{N}{k}\right)^2 2k^2(ke_{mult} + (k-1)e_{add}) + \left(\frac{N}{k}\right)^2 k^2(e_{div} + e_r) + \left(\frac{N}{k}\right)^2 (k^2 - 1)e_{sh}$$

Using N x N=128 x 128 image pixel size and k is the block size and its standard value is 8 since JPEG compression only works on 8 x 8 blocks.

$E_p = 2.7566$ mJ

E_p amount of energy is consumed during compression using Jpeg algorithm and this amount is dissipated in compressing a single image. This amount of energy is always deducted from a given source node while transmitting a given image of the specified pixel size.

6. Proposed Design

6.1 Description of Proposed Design

In this chapter, a design has been proposed to transmit still images in an energy efficient manner from WMSN nodes to the base station. The design shown in Fig.6.1 is the the proposed WMSNs design including the transmitter and receiver modules at the respective sensor nodes. It describes the transmission of a sensed image from a given WMSN to a base station where the information in the sent data packet is interpreted into a useful image. A given sensed image after being encoded using the proposed design, will be transmitted using wireless routing protocol. Then, after being received by the base station, the received data will be decoded to reconstruct the image. There are three basic steps in the proposed design for transmission and reception of the sensed image data.

1. Codebook Generation using SOFM
2. Encoding Phase
3. Decoding Phase

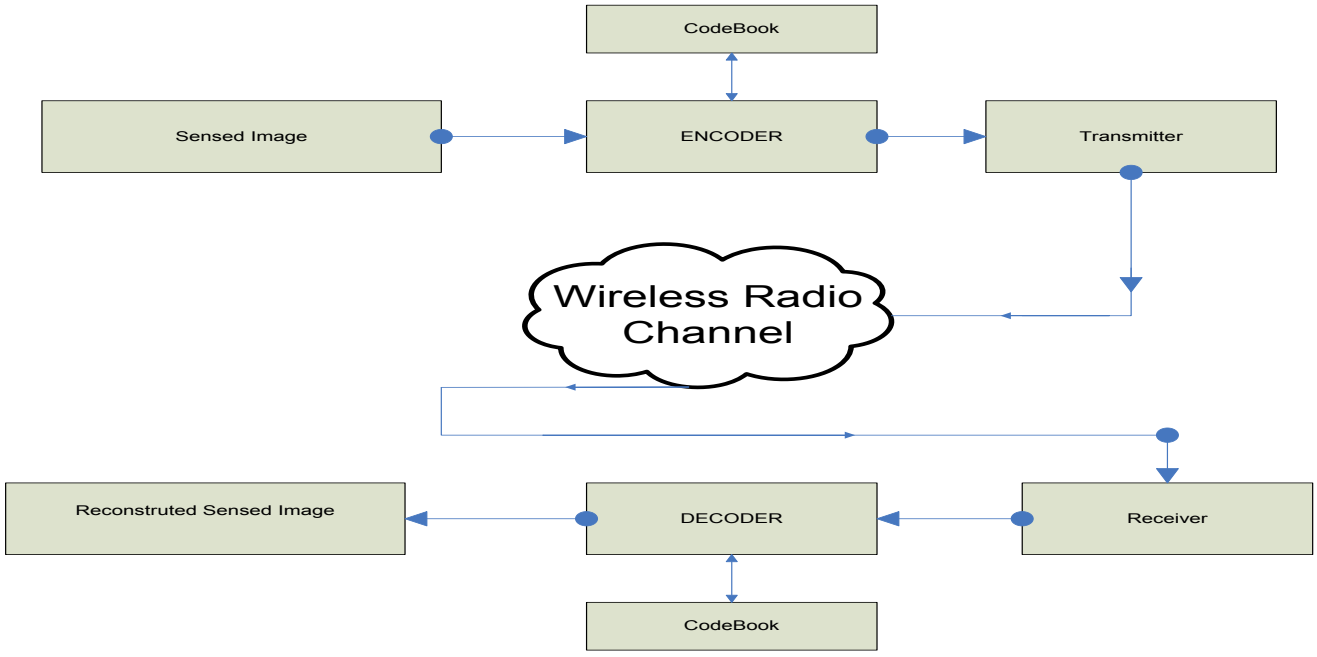

Figure 6.1 The proposed design

As previously mentioned, data transmission consumes a lot of energy in WMSN. So, it is aimed to reduce the transmission energy by dimensionality reduction of the input vectors in this work. One of the techniques that are more suitable to reduce the dimension is SOM's vector quantization technique. As noted in section 3, SOM can be used to encode a large set of input vectors by finding a smaller set of "representatives" or "prototypes" or "code-book vectors" that provide a good approximation to the original input space. This is the basic idea of vector quantization theory, the motivation of which is dimensionality reduction. So, in this proposed design, a vector quantization technique to compress the image data is used.

To design and implement an image vector quantization, it is necessary to consider two fundamental aspects of vector quantization: the codebook design algorithms and code vector searching techniques. In the generation of codebooks, the main concern is on the optimality of the resulting codebook as related to classes of images for a particular application even though the speed of the design is also important. Algorithms for optimal codebook generation have been developed in this work. As the codebook design procedures are usually extremely time consuming, fast clustering techniques have been used.

This work uses Kohonen's SOFM for designing codebook. The code vectors are generated by evaluating the characteristics of the specific standard images, which are obtained from image processing laboratories.

The generated codebook will be stored at every node including the base station in offline mode. Image transmission and reconstruction at the base station consists of two functions: Encoding and Decoding. During the encoding phase, the input image is divided into blocks and the corresponding codeword is obtained by searching the codebook and transmitted to the base station. During the decoding phase, the transmitted codeword is converted to the code vectors. The size of the codebook decides the compression speed as well as the quality of the decompressed images. The details of each of the phases are described below.

6.1.1 CodeBook Generation

The codebook is generated by training SOFM following two approaches of training the network: these are single image training and batch training. For the purpose of codebook generation, the input image is first divided into several non-overlapping rectangular blocks which form vectors of fixed dimension. These blocks are called training vectors and the collection of training vectors form the training set of size V. V is computed using the equation.

$$V = \frac{N\ x\ N}{p\ x\ p} \tag{6.1}$$

where k is the dimension of the vector ($k = p\ x\ p$) and $N\ x\ N$ is the number of pixels in the input image. The algorithm used to train the neural network is shown in the figure 3.2. After the training images are prepared the following steps are followed to generate the codebook.

The size of the codebook used in this paper is M and the codeword is a vector of size k. The cyclops camera(used by most WMSNs) is capable of capturing 3 types of images in 3 different resolutions. It can capture 1 bit per pixel black and white images, 1 byte per pixel greyscale images in 32x32, 64x64 and 128x128 resolutions [26]. Since these are the only resolutions that are supported by WMSN nodes an image resolution of 128 x 128 is used in the simulation. After an extensive study on different codebook sizes considering the image quality and the energy consumed during encoding the size of the codebook in this paper is chosen to be 50 and each code vector size is 16 as the image is divided into 4 x 4 non overlapping blocks. After preparing the training images, the steps in figure 6.2 are followed to train and to generate the codebook. Final codebook (weights) of size 50 x 16 is stored at each node in an offline mode.

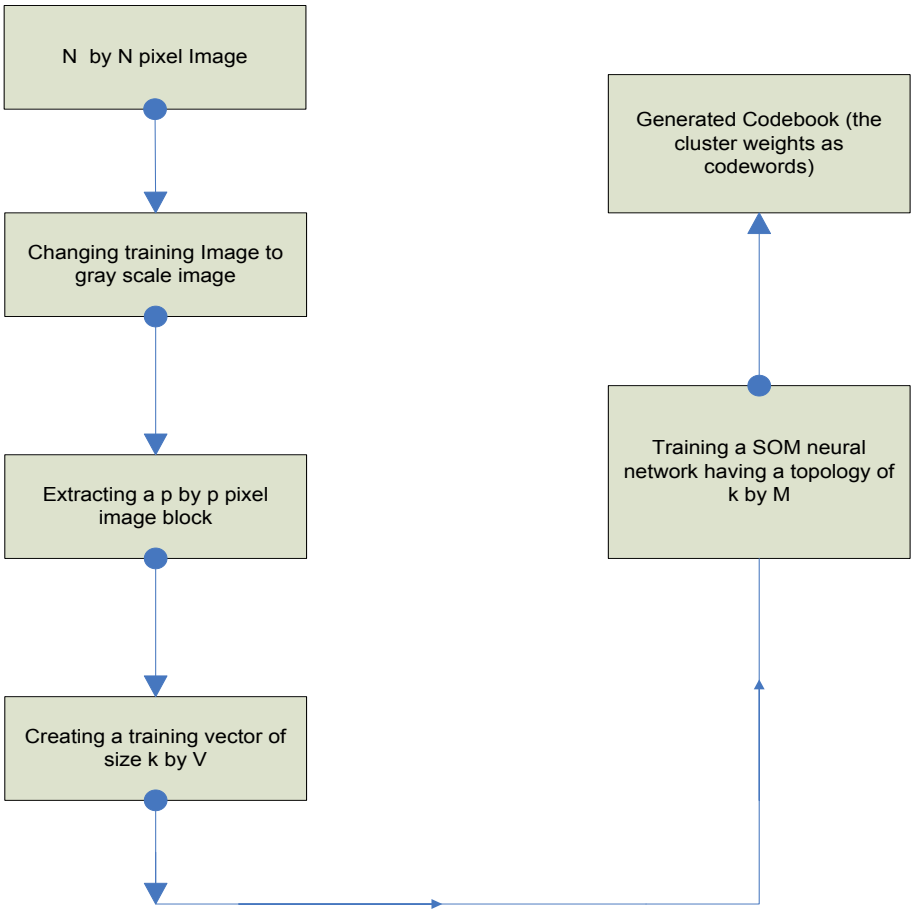

Figure 6.2 Codebook Generation

6.1.2 Encoding Phase

After a sensed image has been taken by the wireless multimedia sensor node camera, it will first be preprocessed and an image block of p x p pixel should be prepared as shown in the general encoding phase diagram in figure 6.3. After all the encoding steps have been performed the index values corresponding to each image block will be transmitted.

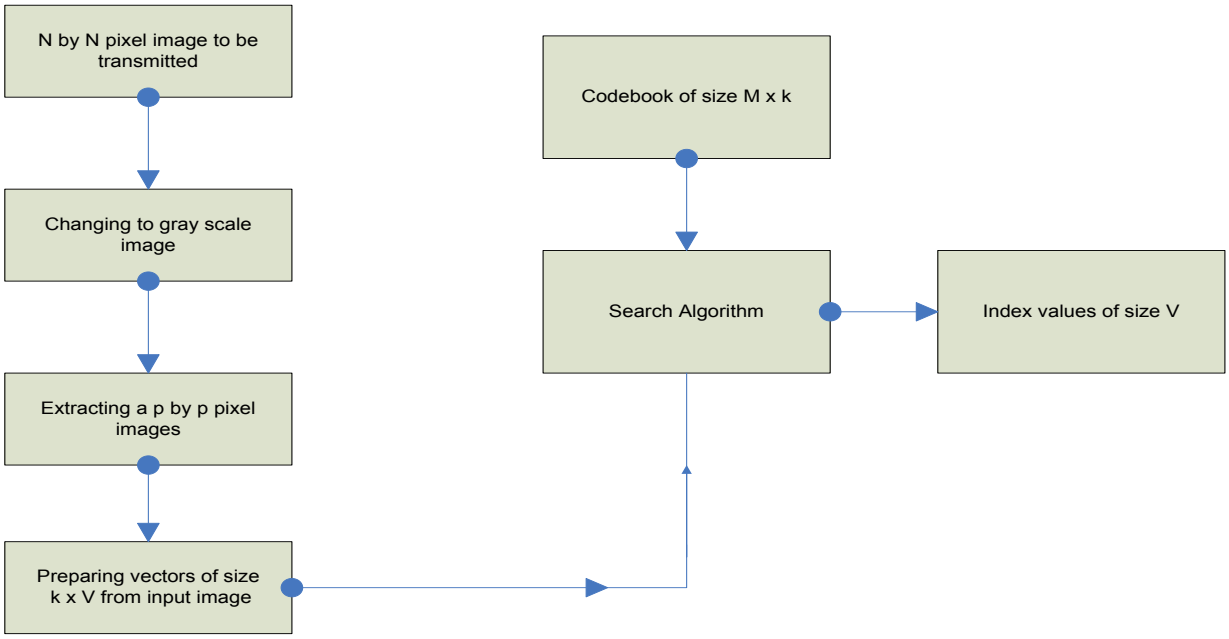

Figure 6.3 Encoding Phase

Encoding involves codeword searching to find the representative code vector for each input vector. The input vector X to the encoder is of dimension k. In this work, X is a block of pixels from an image. The image is partitioned into non-overlapping square blocks which are p x p pixels.

During encoding the distortion $d(X, Y_i)$ between the input X and each codeword in the codebook $Y_i, i = 1,2, \ldots, M$ is calculated as shown in fig 6.4. The optimum encoding rule is the nearest neighbor rule, in which the index i is transmitted to the decoder if codevector Y_I yields the least distortion [33]. A distortion measure is used in this work is the euclidean distance measure is given by equation 6.2.

$$d(X, Y_i) = \|X, Y_i\|^2 = \sum_{j=1}^{k} \left(x_j - y_{ij}\right)^2 \quad (6.2)$$

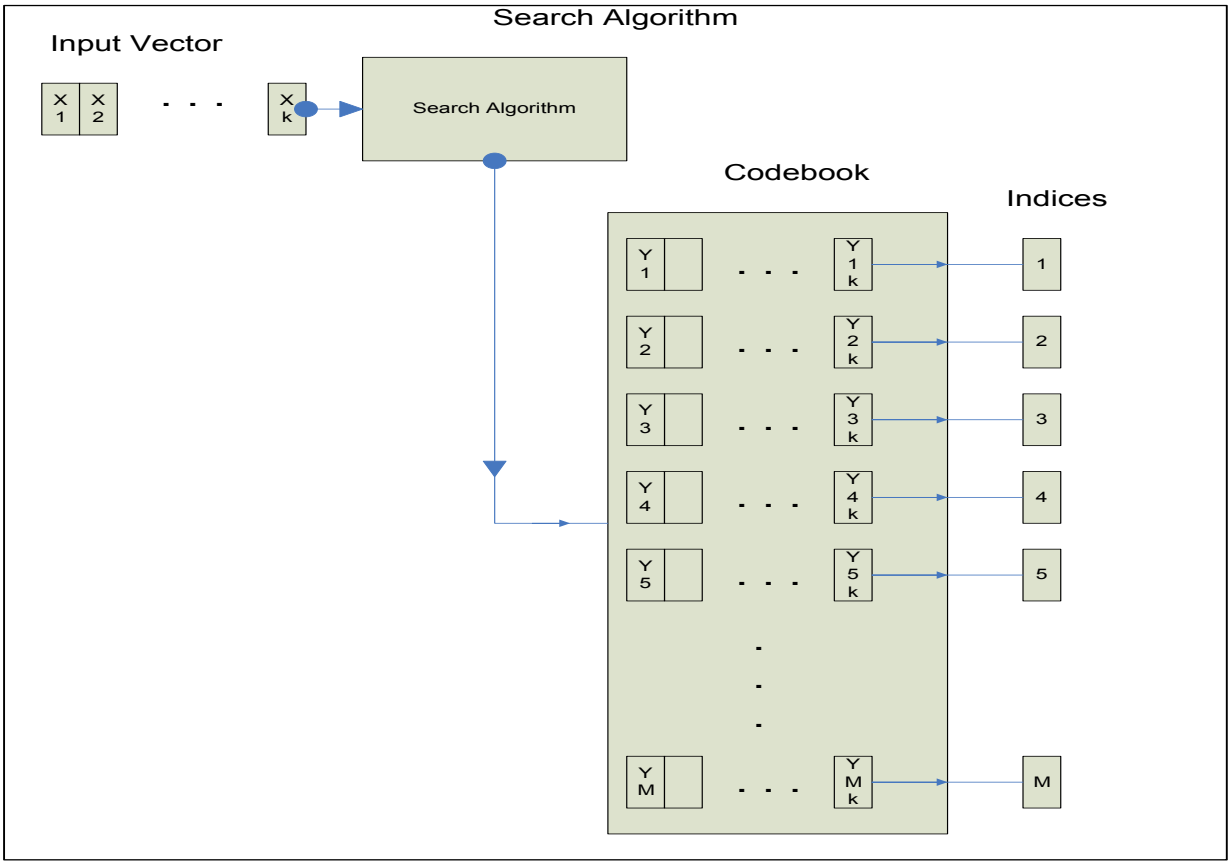

Figure 6.4 Codebook Search in Enoding phase

6.1.3 Decoding Phase

The decoding phase for WMSNs is performed at the base station where the information is finally reconstructed to get a visual image of the sensed environment for further decision by the network deployer. This part of the architeture will be discussed but it will not be included in the energy consumption model for the proposed design because this algorithm is not going to be executed at the sensor nodes where there is a constraint of energy.

The decoding phase starts after the transmitted index values have reached the base station from the respective WMSN nodes.The reconstruction of the code vectors from transmitted indices is shown in figure 6.5.

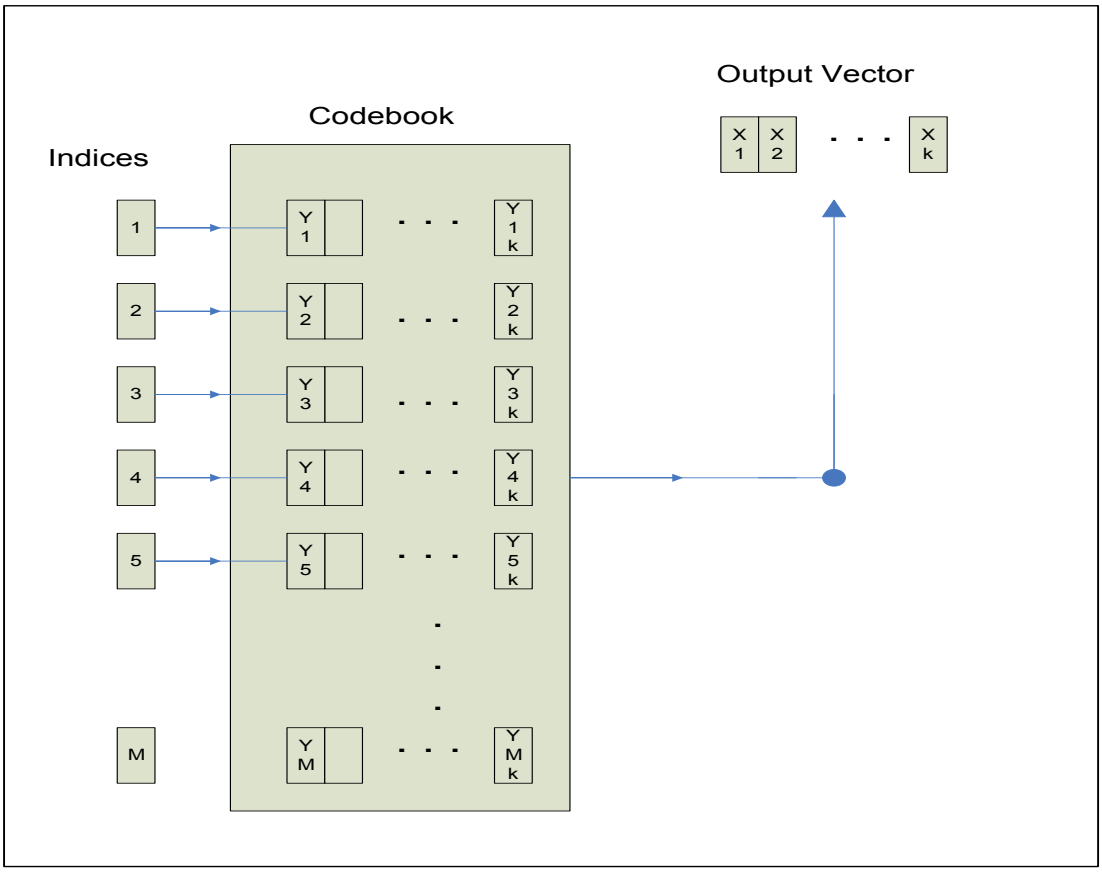

Figure 6.5 Decoding Phase

6.2 Standard Images

A standard test image is a digital image file used across different institutions to test image processing and image compression algorithms. By using the same standard test images, different labs are able to compare results, both visually and quantitatively. The images are in many cases chosen to represent natural or typical images that a class of processing techniques would need to deal with. Other test images are chosen because they present a range of challenges to image reconstruction algorithms, such as the reproduction of fine detail and textures, sharp transitions and edges, and uniform regions [35].

The standard images are taken from [36]. The images are maintained primarily to support research in image processing ,image analysis, and machine vision. The images used in this work are bitmap (BMP) image file formats of size 128 by 128 pixels.Why bitmap images are chosen is that since there are two distinct types of digital graphics, vector and bitmap. JPEG compression

only works on bitmap images since vector graphics are not digital images, and cannot be made any smaller. Bmp is one of the most widely used uncompressed and raw file formats used in image sensing hardwares.Graphic files with a bmp format are uncompressed bitmapped images. In bmp format files, each and every pixel has its own specific color, laying out a detailed map of the picture. Bmp images, because they are uncompressed, are larger in size when compared to JPEG images [60].

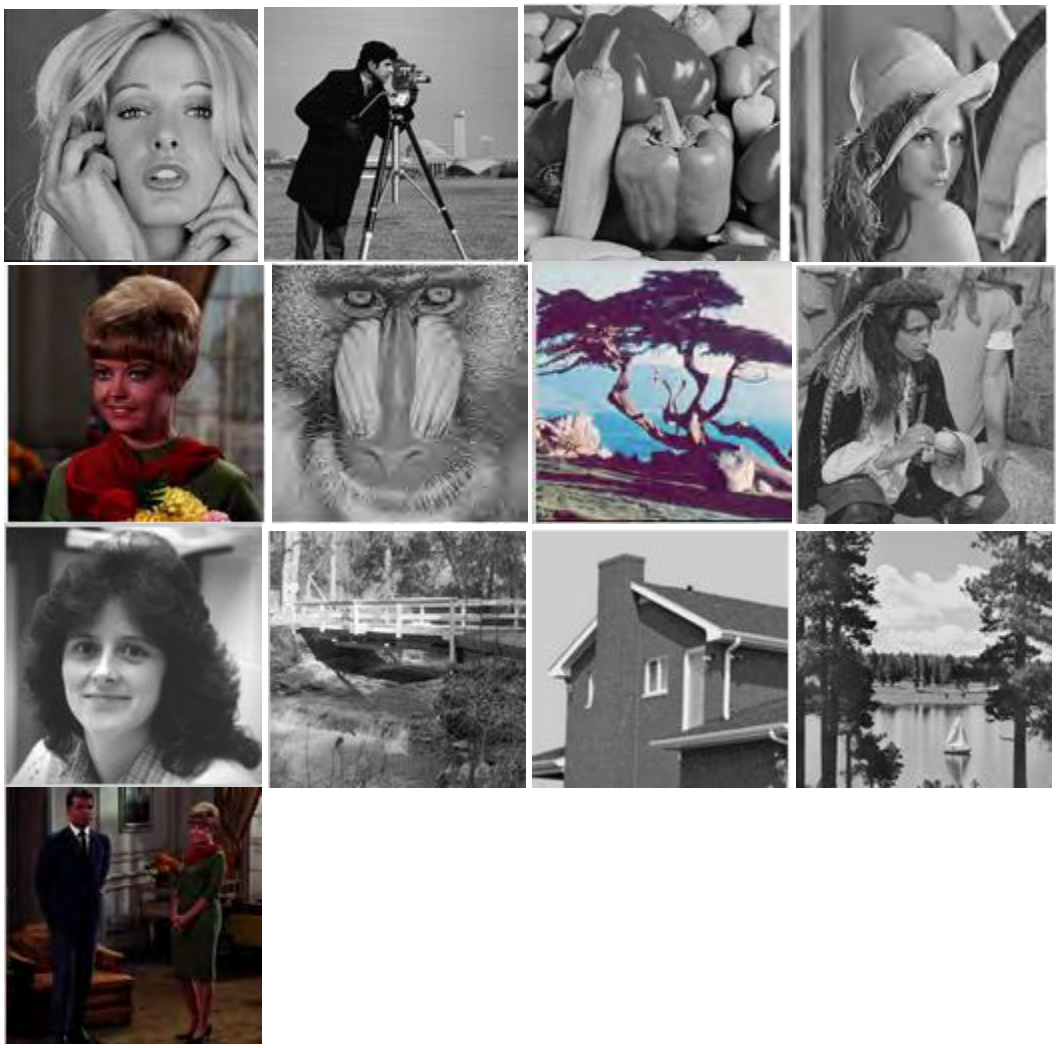

Figure 6.6 Standard Images

6.3 SOFM Training

SOFM is one of the most fascinating unsupervised algorithms in the neural network field. Such networks can learn to detect regularities and correlations in their input vectors and cluster the inputs based on similarities. It learns to recognize groups of similar input vectors in such a way that neurons physically near each other in the competitive layer respond to similar input vectors. SOMs do not have target vectors, since their purpose is to divide the input vectors into clusters of similar vectors. There is no desired output for these types of networks [34].

The architecture of SOM neural network used for codebook generation is shown in fig 6.7. The network consist of k input nodes corresponding to the number of input vector elements, M output nodes which are arranged in one dimensional array and M weight vectors of size k which represents code vectors. In this work codebook is generated using samples from different images and by varying SOFM network parameters such as having adjustable learning rate and weight initializations. In single image training image sub samples are taken only from one image such as Lena, which contains low to mid frequency components. And during batch training six standard images are used. During training first the image of size 128x128 pixel is divided into pxp pixel blocks in order to generate a k-element vector, which acts as input to SOFM neural network. Weights of SOFM neural net serves as code-vectors in codebook, learning rate and also number of epochs are changed in order to prevent over training of certain input vectors.

A k by M SOM neural network is used for codebook generation because of its clustering ability of high dimensional data. Standard images, shown in the section 6.2, for image processing are used as training images and test images.

For the generation of the codebook two neural networks are created.

1. Single image training (representative image Lena)
2. Batch training of 6 standard images (Lena, Tiffany, Girl, Cameraman, Baboon, Peppers)

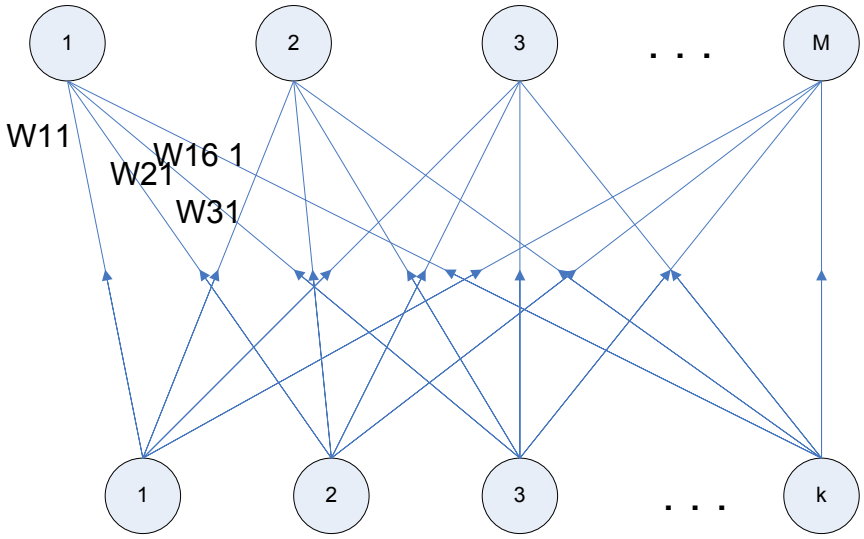

Figure 6.7 SOFM neural network of size k X M

The initial weights used for training of the neural network is taken from the training image itself by extracting randomly within a specific pattern.The following parameters are used for the training of the SOM neural network after initially setting them to some random values. And by rigorous experimentation, the PSNR values of the test images after decoding are checked and the values of the parameters are changed accordingly.

1. The number of epochs is 2000.
2. The learning rate is 0.6
3. The decrement of learning rate is 0.5

The steps of training algorithm for the SOM neural network are given below [27]

Step 0: Initialize weights Wij (set with radomly from the training image)

Set learning parameters

Step 1: While stopping condition is false, do steps 2 – 7

Step 2: For each input vector X, do steps 3 – 5

Step 3: For each j, compute:

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

Step 4: Find index J such that D(J) is a minimum.

Step 5; For all units j within a specified neighborhood of J , and for all i:

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + \alpha[x_i - w_{ij}(\text{old})]$$

Step 6: Update learning rate.

$$\alpha \text{ (new)} = \alpha \text{ (old) X decrement of } \alpha$$

Step 7: Test stopping condition

It has been arrived at the learning rate parameters after several experiments and these values are optimal for this work.

6.3.1 Single Image Training

For the single image training the standard image, Lena, is used. After several experimentation by using a Lena trained neural network, different codebook sizes are created and tabulated below in order to choose an optimal size of code book. For the training SOFM network first the image is divided into non overlapping blocks of size 4 x 4 pixels that give 16 x 1024 size matrix of pixel values ready for training from the original size of 128 x 128 of Lena.

Figure 6.8 Lena

After the network is fully trained, for the different codebook (CB) sizes the PSNR values are calculated and codebook size of 50 is chosen as there is not much difference in the image quality. As can be seen from table 6-1. Codebook sizes lower than 50 are checked but the image quality starts to decline badly and hence are not included for comparison purposes.

Table 6-1 PSNR values for the respective codebook sizes in single image training

Test Images	128 CB	100CB	90CB	80CB	70CB	60CB	50CB
LENA	24.41	24.37	24.36	24.19	24.3	24.06	23.97
GIRL	21.29	21.19	21.20	21.17	21.02	21.14	21.07
CAMERAMAN	19.97	19.88	19.92	19.87	19.83	19.83	19.79
BABOON	23.73	23.64	23.68	23.58	23.50	23.42	23.36
TIFFANY	23.97	23.84	23.75	23.71	23.67	23.53	23.56
PEPPERS	21.91	21.88	21.86	21.774	21.70	21.61	21.61
COUPLE	22.89	22.84	22.84	22.74	22.67	22.62	22.59

The chosen code book of lower value has an effect during encoding both for decreasing the energy value and the speed of encoding.

6.3.2Batch Image Training

During the batch training six standard images are chosen to train the network. First all the images are divided into non-overlapping images of size 4 x 4 which gives a matrix of 16 x 1024. After preparing all the images in such ways all the matrices of the image pixel values are concatenated horizontally to give a matrix of size 16 x 6144. And this is used to train the network using a batch training mode of SOFM network. The six standard images used for batch training are shown in the figure 6.9.

Tiffany Cameraman Peppers Lena

Girl Baboon

Figure 6.9 Images for batch training

After the SOFM neural network is fully trained the different codebooks are checked for image quality using the PSNR values. And the lower codebook size is chosen since it gives an image quality not different than the other sizes. And a graceful trade-off is considered to choose the codebook size of 50 for the proposed design. Lower codebook sizes ,lower than 50,resulted in a bad image quality and hence not considered for comparison.

The codebook generation using batch training method is best suited to several applications since it trains the network with images of several natures and hence for the application of WMSNs, if the area of application is known before hand an optimal codebook could be generated off-line. And this codebook can give a better image quality. Therefore for the proposed algorithm even though two ways of generating a codebook are tested and good image qualities are obtained the latter is chosen to provide the codebook for the work. The results of the psnr values for the different codebook sizes is tabulated in the table 6-2.

Table 6-2 PSNR values for the respective codebook sizes in batch image training

TEST IMAGES	128 CB	100CB	90CB	80CB	70CB	60CB	50CB
LENA	24.45	24.29	24.18	24.12	23.99	23.79	23.54
GIRL	27.51	27.24	27.1	26.97	26.71	26.50	26.24
CAMERAMAN	26.10	25.47	25.32	25.21	24.84	24.42	24.21
BABOON	24.35	24.18	24.09	24.01	23.91	23.79	23.57
TIFFANY	25.82	25.58	25.43	25.18	25.05	24.94	24.71
PEPPERS	25.26	24.92	24.83	24.72	24.46	24.16	23.98
COUPLE	27.05	26.81	26.55	26.49	26.22	26.05	25.89
PIRATE	24.30	24.16	24.08	23.98	23.92	23.81	23.58
WALKBRIDGE	22.29	22.17	22.05	21.90	21.81	21.71	21.49
WOMAN-DARKHAIRED	24.88	24.69	24.59	24.23	24.23	23.95	23.47
HOUSE	24.36	24.21	23.99	23.57	23.42	23.12	22.78
LAKE	21.76	21.62	21.57	21.35	21.33	21.07	20.85
TREE	21.17	21.01	20.89	20.63	20.58	20.41	20.06

6.3.3 Codebook and Codeword Sizes

After several experimentation of code book sizes considering the energy consumption and encoding time lower codebook sizes are better and hence a code book of 50 is chosen. The code word size is chosen to be 4 x 4 because at size 2 x2 even if the images obtained after

reconstruction are higher qualities the time of encoding and the energy consumption are higher. And at sizes 8 x 8 the reconstructed image is blocky and does not have a good image quality.

6.4 Computational Cost

The proposed design consumes a given amount of energy due to the computational overhead it creates on the WMSN node and this energy is calculated using the energy consumption value of different operations on Atmel AtMega128L microcontroller. From the microcontroller datasheet and section 4.2.2 , the energy consumption of the following arthimetic operations as shown in table 6.3 can be obtained.

Table 6-3 Energy of instructions

Arithmetic Instruction Operation energy	**Energy Consumption value**
ε(add)	3.3 nJ/byte
ε(mult)	6.6 nJ/byte
ε(sub)	3.3 nJ/byte
ε(cmp)	3.3 nJ/byte

Assuming M to be the codebook size and K to be the codeword size the following operations are performed on the proposed architecure.

1. K X M multiplication operations
2. (K-1) X M addition operations
3. K X M subtraction operations
4. M Comparison operations

The Energy consumed during encoding using the proposed algorithm is given by

Energyconsumed/ code vector

$$= (K \times M) \times \varepsilon(mult) + ((K-1) \times M) \times \varepsilon(add) + (K \times M) \times \varepsilon(sub) + M \times \varepsilon(cmp) \quad (6.3)$$

And hence the total energy consumption per encoded image can be obtained by multiplying by the size of the encoded image. Let the size of the encoded image blocks be H and then the total energy per encoding algorithm per image can be obtained as follows:

EnergyConsumed/ Encoded image= H * Energyconsumed/ code vector (6.4)

And hence the energy consumption for this work can be calculated based on the following parameters.These are: K being the codeword size is equal to 16 and M being codebook size is equal to 50 in this work and the size of the encoded image data is 1024 image block.The energy consumption calculated is, incorporating equations 6.3 and 6.4, as follows:

EnergyConsumed/ Encoded image = 10.8134 mjoule of energy is consumed.

7. Simulation and Performance Analysis

In order to analyze the performance of the proposed design , a cluster based routing protocol ,LEACH, based on the works of Heinzelman [17] is chosen.

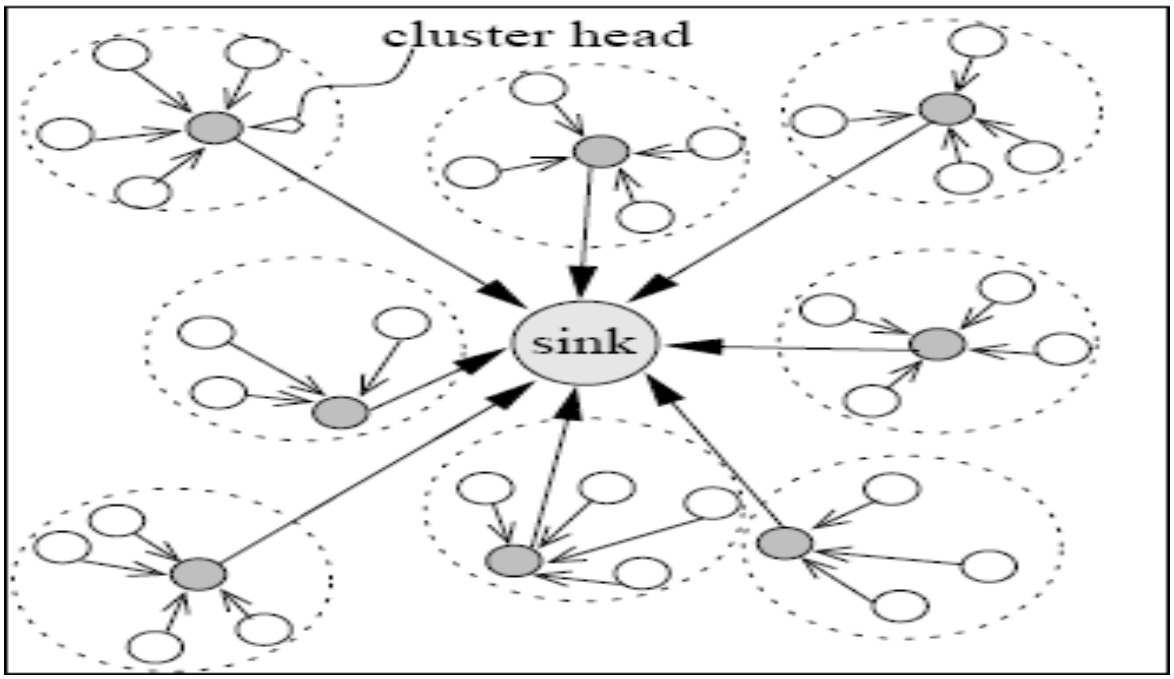

Figure 7.1 LEACH routing protocol

In cluster-based routing protocols, nodes are grouped into clusters and each cluster head node collects, processes and forwards the data from all the sensor nodes within its cluster to the base station. LEACH is an early and popular routing protocol and its target scenario is a sensor network with a known number of nodes and known area, with a dedicated data sink to which all data is to be reported.

In LEACH, the nodes organize themselves into local clusters, with one node acting as the clusterhead. All non-cluster-head nodes must transmit their data to the clusterhead, while the clusterhead node must receive data from all the cluster members , perform signal processing functions on data and transmit the data to the remote base station.

Therefore, being a cluster-head node is much more energy-intensive than being a non-cluster-head node. Thus LEACH incorporates randomized rotation of the high-energy cluster-head

position such that it rotates among the sensors in order to avoid draining the battery of any one node in the network.

7.1 Simulation Objectives

The objective of the simulation is to analyze the lifetime of the WMSNs to show the contribution of the proposed design by comparison with the selected JPEG design in increasing the lifetime of the sensor nodes so that the area under monitoring will be controlled for more time. There are two steps in the lifetime analysis:

1. The existing design ,JPEG, is used first and the energy consumption of each node during transmission and reception of each packets is calculated. The computational energy cost calculated for this design is detected when a node is transmitting or receiving.
2. The proposed design is used and the energy consumption of each node during the process of transmission and reception is calculated. The computational energy cost calculated for this design is detected when a node is transmitting or receiving.

In addition to the main simulation objective as stated above the memory requirement, time of encoding and image quality analysis are also done using simulations.

7.2 Simulation Methodology

The simulation is done using Matlab 7.9.0(R2009b). The simulation is chosen to be done in Matlab development environment since Matlab is widely used in many scientific works because of its ability to perform computaionally intensive tasks faster and because neural networks are best performed in Matlab environment and instead of using two different environments it becomes important to stick to one. The simulator consists basically of the Leach routing protocol for the transmission of the data packets sent by each sensor node. The main process consists of the simulation process which includes the following phases which are initiated for every round of the simulation.

1. Cluster Setup Phase
2. Transmission Phase

In figure 7.2 the whole simulation process is described. After the simulation has started and all the simulation parameters are set; the WMSN is deployed in a given area of interest with limited dimension randomly and the sink node being located according to simulation parameters. And after the deployment the normal sensing functionality starts and the sensors are assumed to have taken images from their field of view. In order for the transmission of the sensed images to happen the routing algorithm , LEACH, has to be set up decentrally according to its set up phase as shown in figure 7.3.

In the cluster set up phase nodes are selected to be cluster heads based on the routing protocol discussed in section 2.1.4. And the remaining nodes that are not cluster heads for the given round become non-cluster head nodes and wait cluster head announcement from the cluster heads. Calculating their distance to the cluster head according to RSSI ; the non-cluster head nodes send a request to join the cluster head nodes and wait for TDMA schedule to be assigned to them for the transmission phase.

In the transmission phase, as shown in figure 7.4, the nodes start to send data to their respective cluster head nodes and the cluster head prepares its own data to send. It is assumed in this work that the sensor nodes send a single packet per round to the cluster head. And the cluster heads receive data from their member nodes and relay the data to the sink after the respective member node sends the data. It is also assumed that the cluster head nodes send more packets compared to the member nodes in a single round based on the number of nodes in the cluster for that given round.

The whole simulation cycle continues as discussed above for every new round and the program exits when the last node dies since all the nodes are assumed to be dead and the network is malfunctioning. During this simulation all the energy consumptions namely transmission, reception and computational energies are accounted and the remaining energy at each node for every given round is tracked. This way the energy level of each and every node is recorded to analyze the performance.

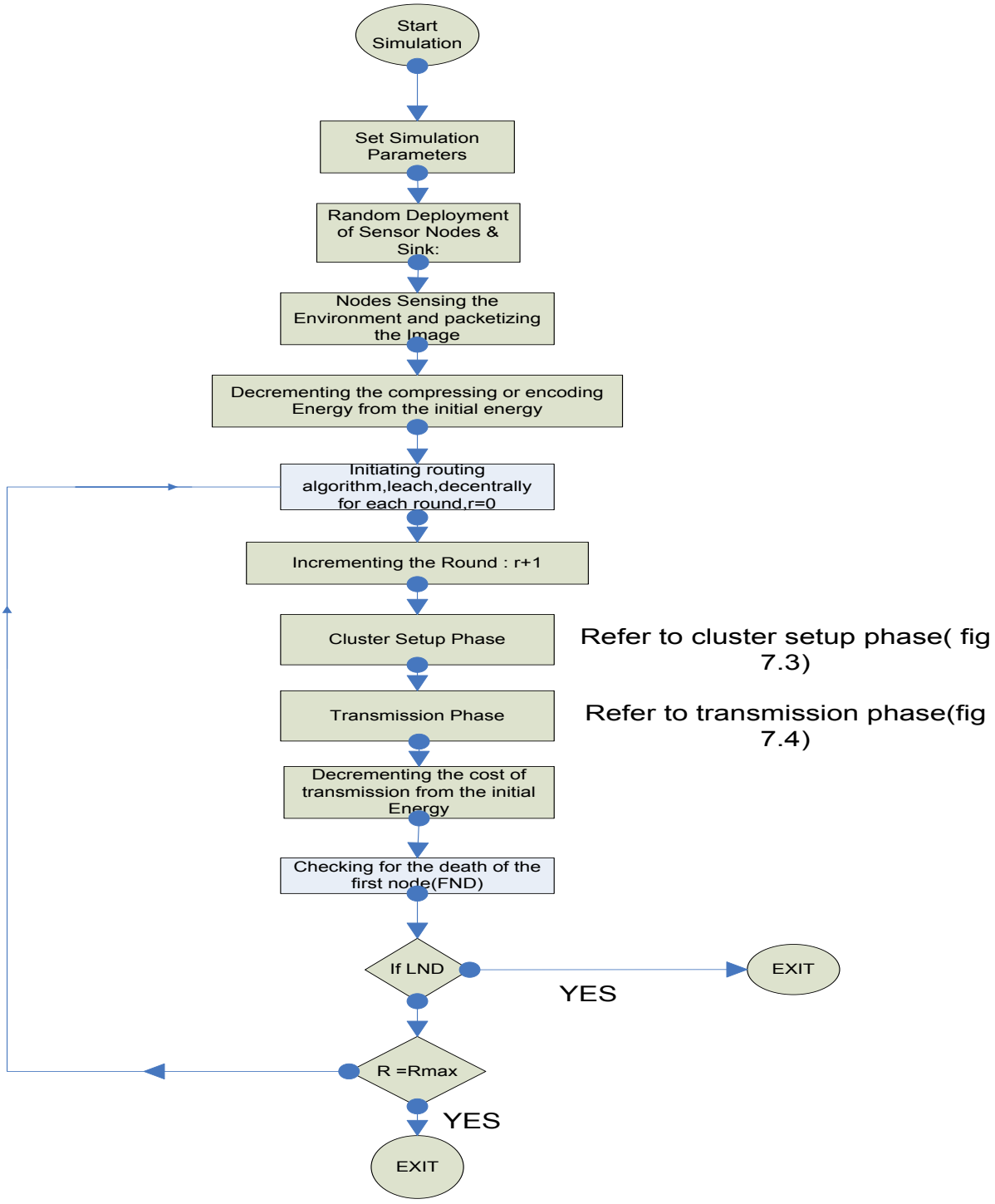

Figure 7.2 Simulation Process

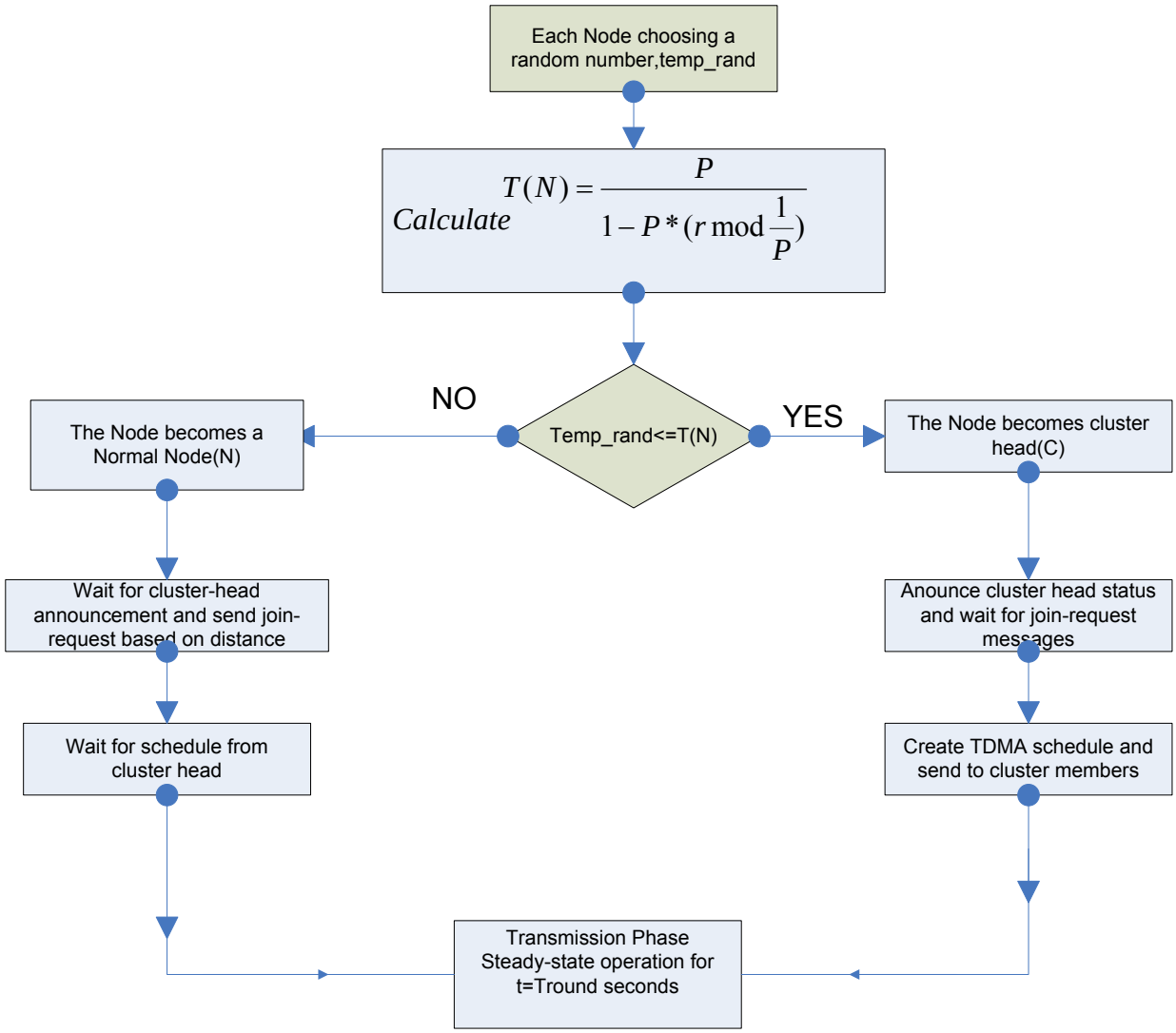

Figure 7.3 Cluster Setup Phase

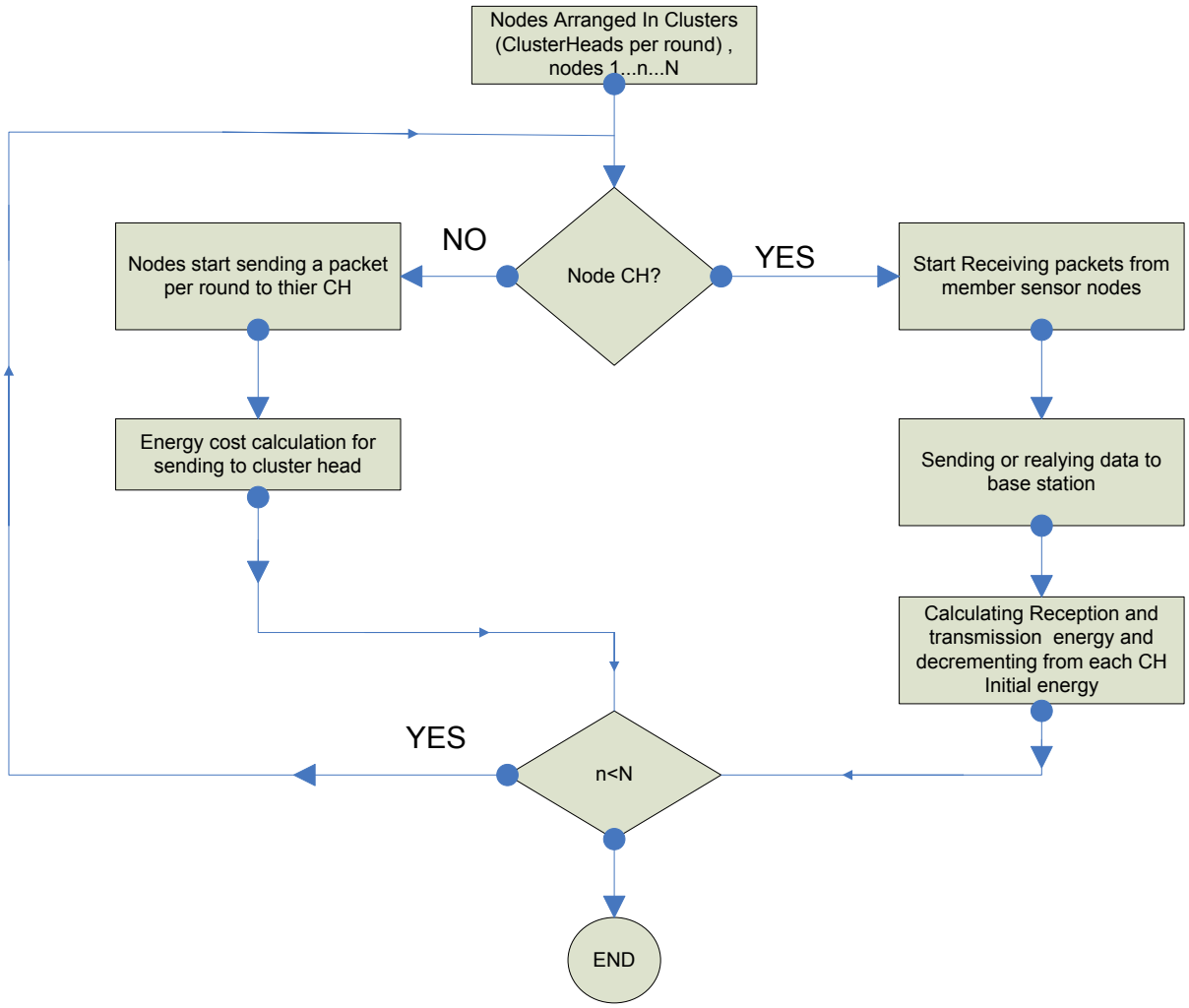

Figure 7.4 Transmission Phase

In the transmission phase the data packets that are sent to the nodes use the procedure described in figure 7.5. After the sensed image is fragmented into blocks of 512 bytes the necessary headers and trailer of 9 bytes are added to create a packet of size 521 ready for transmission. Since a given image can not be sent using a single packet the subsequent packets are sent in the coming rounds. And the energies used for the account of transmission and reception are calculated using discrete power level selection as shown in table 4.3 based on the transmitting and receiving distance.

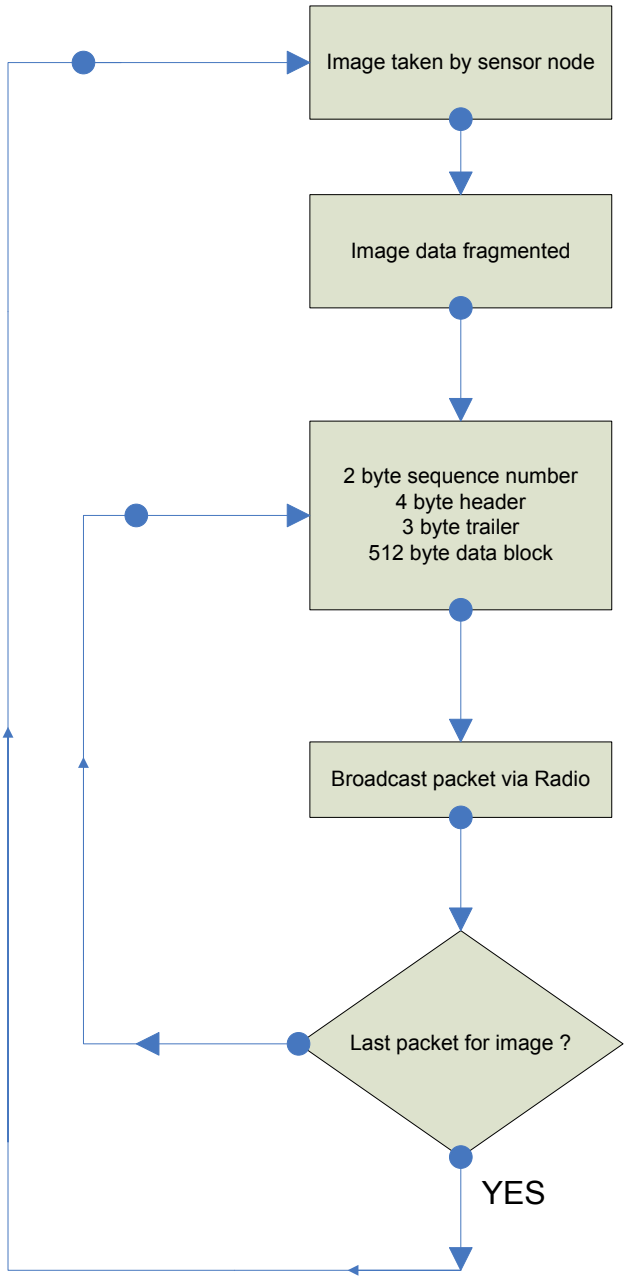

Figure 7.5 Image Data Packets for WMSNs

7.3 Simulation Setup

The flow of the simulation is as shown in the diagrams of the previous section 7.2. The simulation parameters are as shown in the table 7-1 and 7-2.

Table 7-1 Simulation Parameters

PARAMETERS	VALUES
Field Size	100m by 100m
Number of Nodes	200
Base Station location	(50m,50m)
Sensor Nodes	Mica2 Mote
Probability of becoming a cluster head	0.1
Sensed Image (Lena)	128 by 128 pixel
Packet size	521 bytes
Packets per round for member nodes	1
Initial Energy	10.0 Joules

Table 7-2 Simulation Parameters for the Transmission Phase

PARAMETERS	**JPEG**	**Proposed**
Image sent	Lena	Lena
Size of Lena Image after Compression	68800 bits	8192 bits
Packets for single Image	17	2
Image per round	1/17	1/2
PSNR	30.33	24.54
Compression Ratio	1.9051	16
Compression/Encoding time	0.8393	1.2321

During the transmission phase the Lena image is sent to test the different parameters of the performance analysis.

7.4 Life Time Analysis

The simulation model built in Matlab for evaluating the performance of the designs consists of two sub-models: a sensor node model and a base station model. In the network model of the proposed protocol, 200 nodes are distributed across an area of 100x100m with the base station located at position (50m,50m).This kind of network is assumed for this work.

The data packet size adopted in the simulations is 521 bytes. That means each node periodically transmits one data packet to the base station. The calculation of energy consumption for data transmission and reception is based on the energy model presented in table 4.2 of the crossbow tinyos mica2 mote.

The performance of the algorithm can be assessed along three metrics. These are first node dies, last node dies and transmitted images; which are defined as follows:

(1) First Node Dies (FND): This defines the number of packets transmitted when the first node dies.

(2) Last Node Dies (LND): Which defines the number of packets when the last node dies and the network becomes non-functional.

(3) Transmitted Images: Since a single image cannot be transmitted with a single packet. The number of packets transmitted at FND and LND will be determined first and then they will be divided by the number of packets required to transmit a single image based on the design used.

There are three modules contributing to the energy consumption of a sensor node: a micro controller module, a sensor module, and a radio module [58]. The microcontroller module

is responsible for controlling all activities of a node.The sensor module includes sensors attached to the node and the radio module is responsible for wireless communications.

The energy calculation for the radio module (data transmitting and data receiving) based on the energy model described in section 4 have been investigated. Besides the energy consumption of the radio module, the energy consumed by the microcontroller board is considered in order to realistically evaluate the performance of the designs in the simulations.The sensor board energy consumption is ignored since it is not a differential factor.

Based on the energy model described in section 4.1 and the parameters set in section 4.2, we can deduce the energy consumption in our simulations as follows using 0dBm power level for instance is 3.07 μjoule/bit for transmission and 2.21 μjoule/bit for reception, idle energy consumption is not considered since all the nodes are assumed to be active in the work and based on this:

- ✓ The energy consumption for receiving a packet of size 521 bytes data message is:

$$E_{Rx\_data} = lE_{rx} \quad (7.1)$$

$$= (521 \times 8)\text{bit} \times 2.21\ \mu J/\text{bit}$$

$$= 9.211\ \text{mJ}$$

- ✓ The energy consumption for transmitting a signal message is:

$$E_{Tx\_data} = lE_{tx} \quad (7.2)$$

$$= (521x8)\text{bit} \times 3.07\ \mu J/\text{bit}$$

$$= 12.795\ \text{mJ}$$

For the set of simualtions, each node beigns with only 10J of energy and has unlimited data periodically sent to the base station. The amount of energy required to send data to the base station in each round are tracked and the total energy dissipation every round is recorded. In addition , we record the number of clusters, number of member nodes in each cluster, and energy dissipation of each cluster head in each round are also recorded. Since the nodes have limited energy, they use up their energy during the course of the simulation. Once a node runs out of energy , it is considered dead and can no longer sense, transmit, or receive data. Thus, the dead nodes and the associated round when the nodes die are recorded; these results are recorded in text files generated by the simulator.

For these simulations, energy is consumed whenever a node transmits or receives data as well as while performing compression. We don't assume any static energy dissipation, nor do we consider the energy dissipation during the carrier-sense operation. The following table shows the energy of the microcontroller computational energy consumption of the two designs that are compared in this work.

Table 7-3 Computational Energy Consumption

	JPEG	Proposed
Computational energy consumption(in mjoules)	2.7566	10.8134

As can be seen on the figure 7.6 the energy consumption computation of every node is described. An initial energy is set to be 10 joules and every process is tracked that deducts a given amount of energy from this value. Every energy consumption calculation is noted based on the node being a transmitter or a receiver. As can be seen from table 7-3 , even though the amount of energy consumed during computation of the proposed design is much higher than the existing design. Since the numbers of packets that are used to send a single image in the proposed design are very much lower than the existing design the life time of the proposed design is sure to be increased, as will be seen in the upcoming discussions and analysis. The energy required to send a single packet is calculated using equation 7.2 and is found to be 12.795 mJ of energy. And hence it can be understood that reducing the number of packets has a great energy saving.

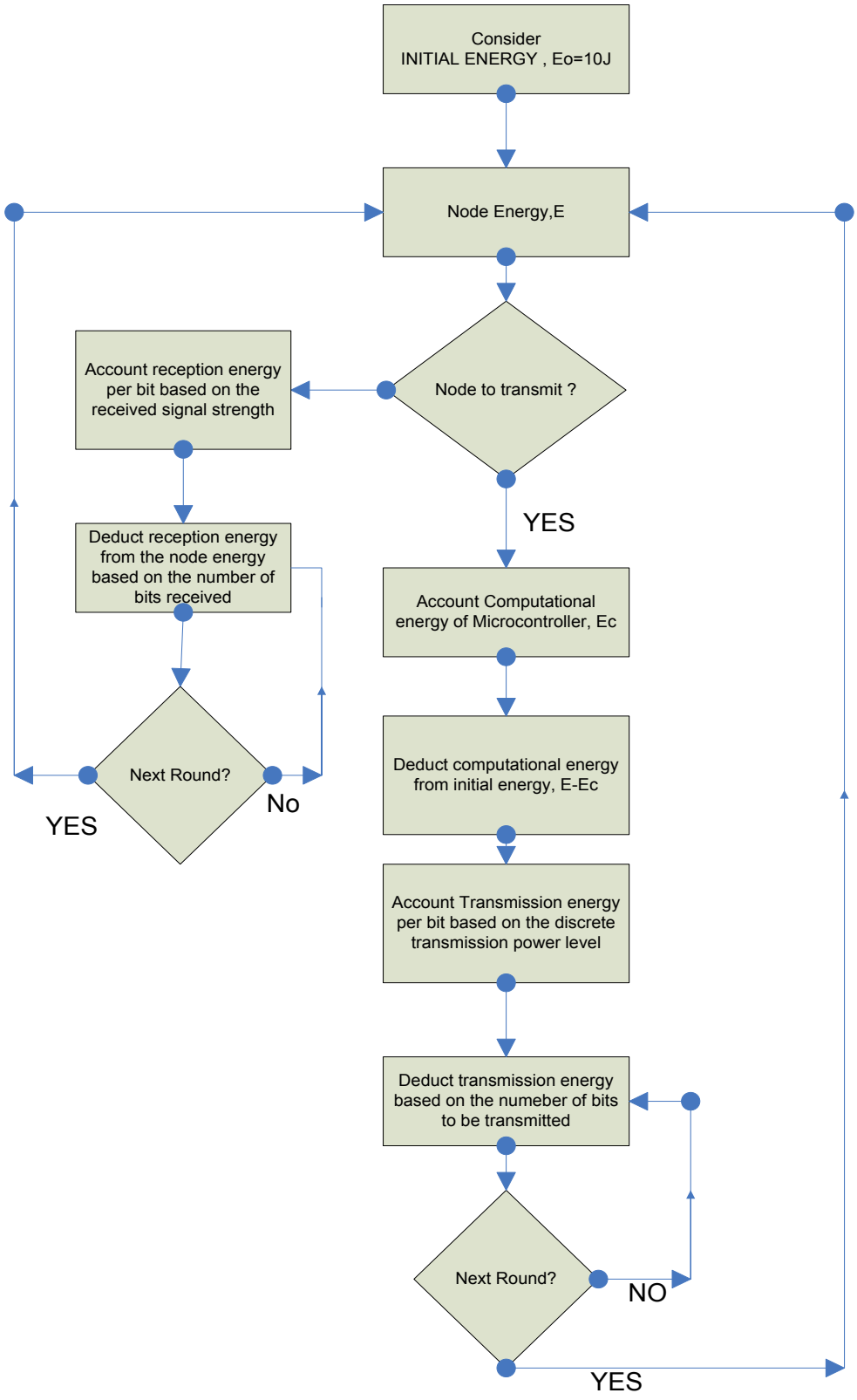

Figure 7.6 Energy Consumption Calculation Flow Chart

The following figures, results and data have been obtained using the simulator and the simulation parameters set in the previous section.

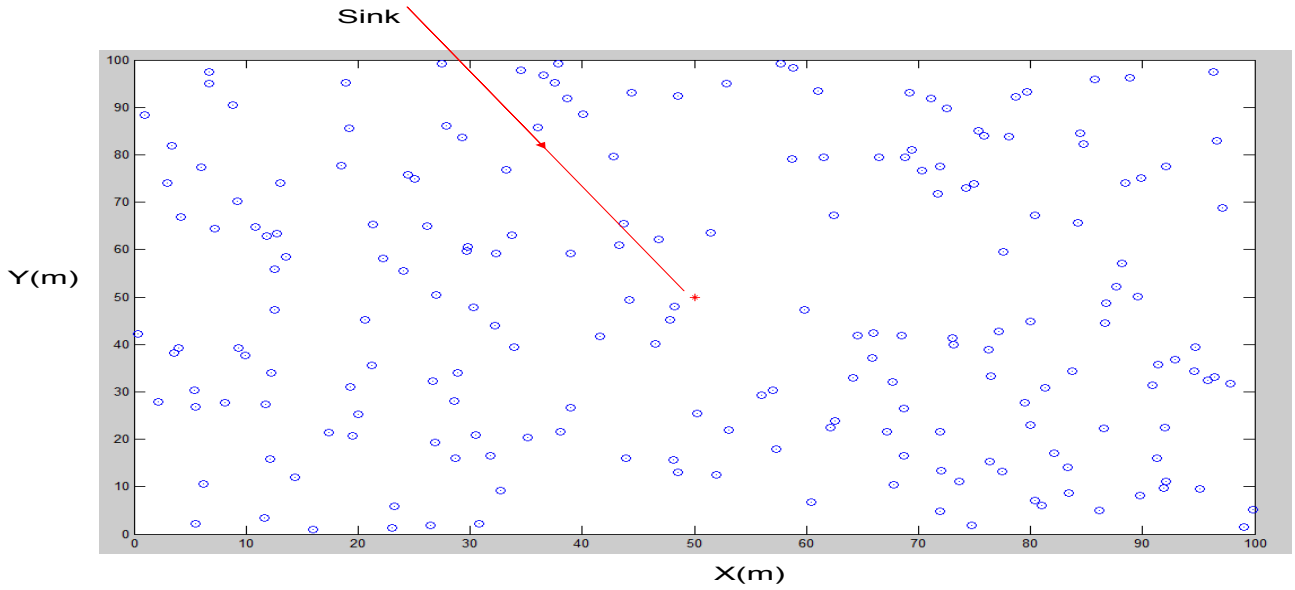

Figure 7.7 Deploying WMSNS

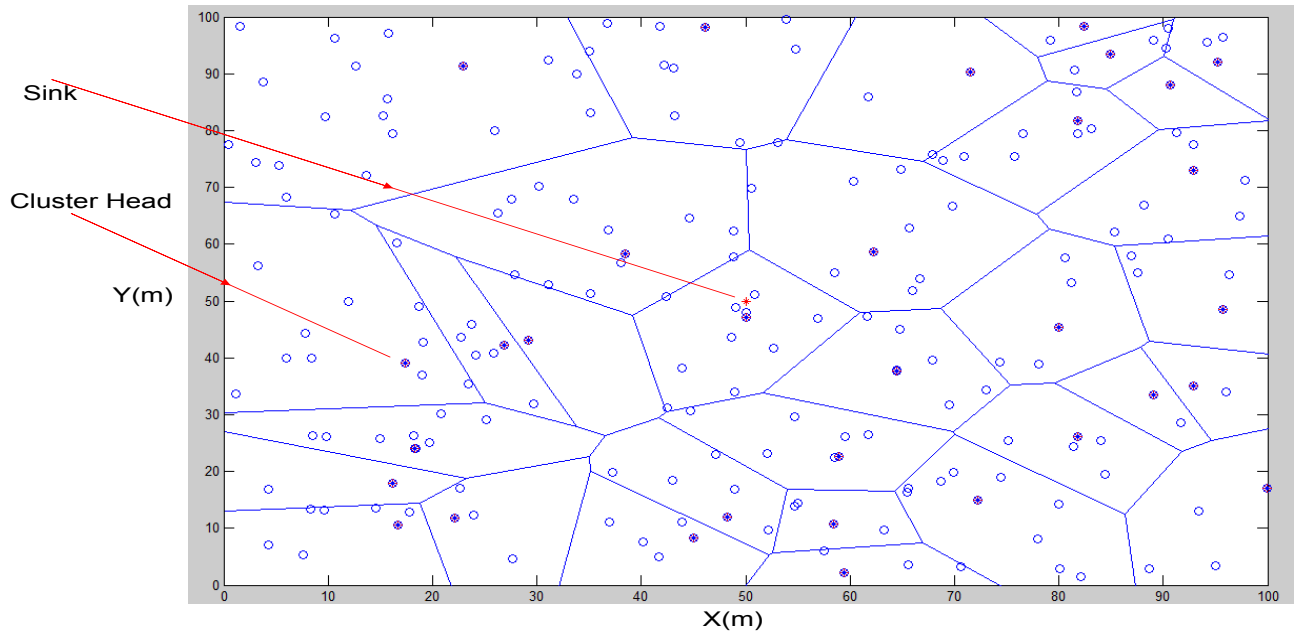

Figure 7.8 Cluster Setup phase

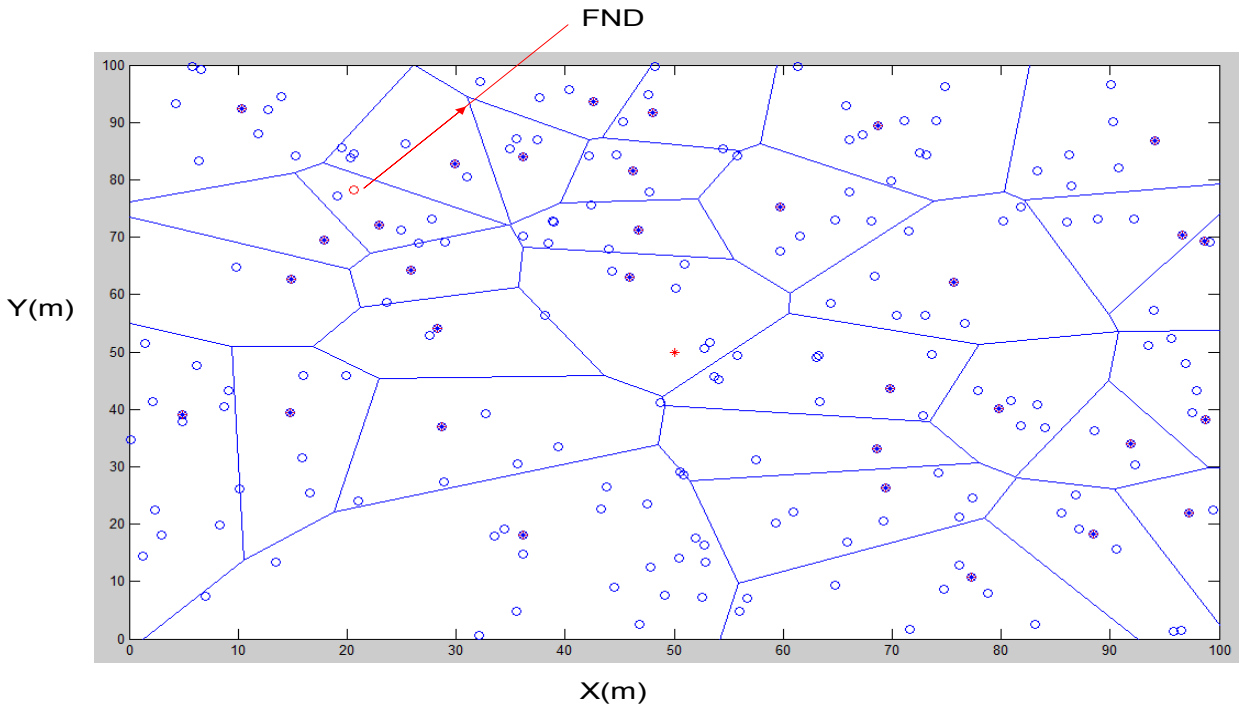

Figure 7.9 FND

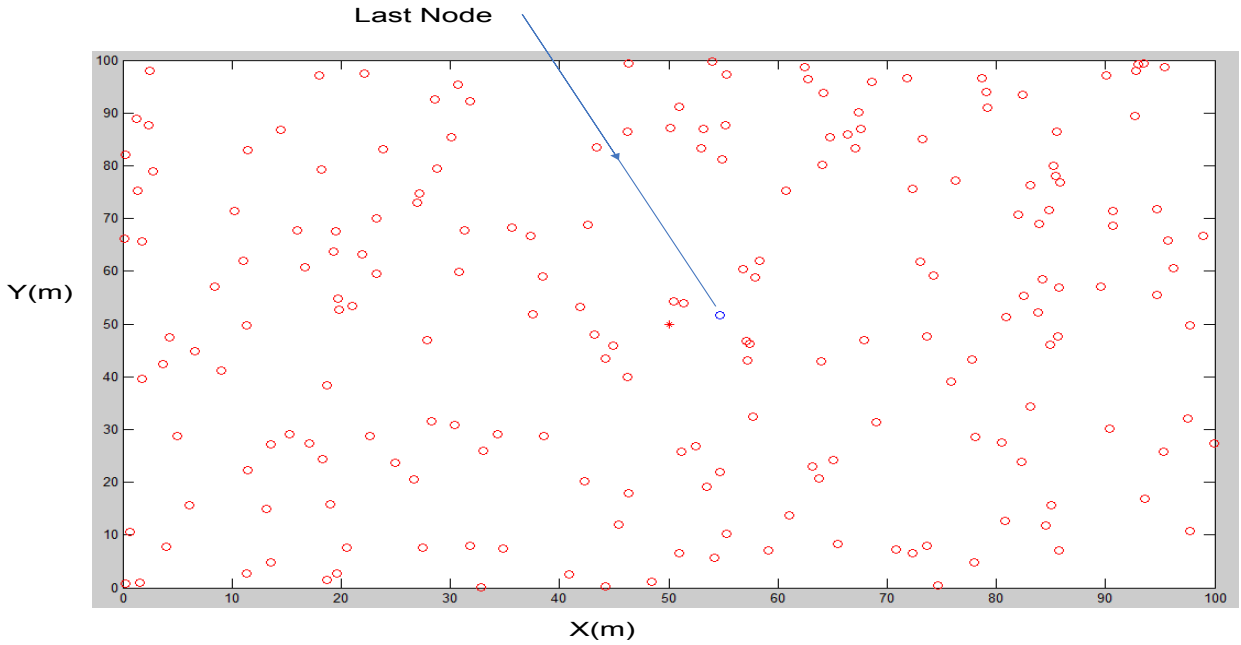

Figure 7.10 Last Remaining Node on WMSN

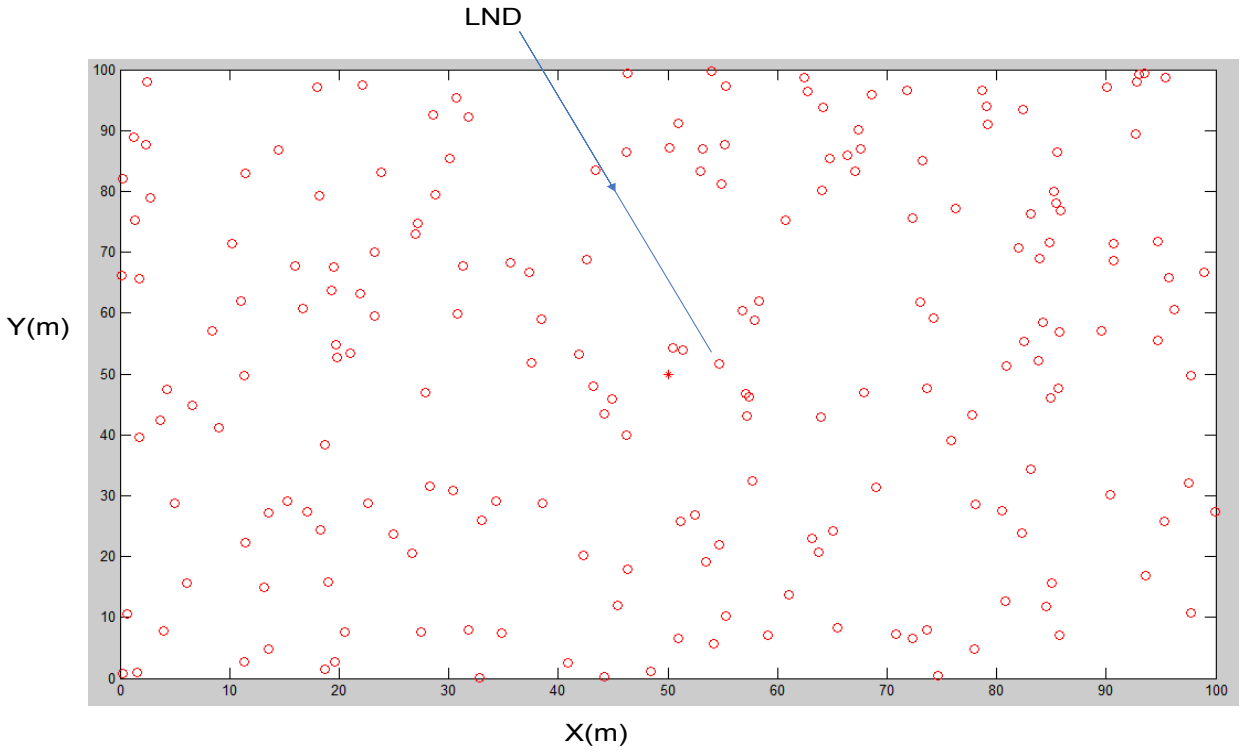

Figure 7.11 LND

Below is shown in a table 7-4 the simulation runs that are performed and the values obtained for the performance parameters. For the existing algorithm a given image could be sent to the receiver after sending 17 packets considering the packet size and two packets are only necessary to transmit an image in case of the proposed design.

Table 7-4 Simulations Runs

Simulation Runs	Existing design				Proposed design			
	No. of Round @ FND	No.of images transmitted @ FND	No. of Round @ LND	No.of images transmitted @ LND	No. of Round @ FND	No.of images transmitted @ FND	No. of Round @ LND	No.of images transmitted @ LND
Run 1	192	11.29	1926	113.29	166	83.00	560	280.00
Run 2	179	10.52	2359	138.76	170	85.00	631	315.50
Run 3	172	10.12	1866	109.76	181	90.5	558	279.00
Run 4	181	10.64	1919	112.88	164	82.00	510	255.00
Run 5	186	10.94	1700	100.00	154	77.00	530	265.00
Run 6	195	11.47	1912	112.47	184	92.00	766	383.00
Run 7	160	9.41	2381	140.06	181	90.50	618	309.00
Run 8	173	10.17	1359	79.94	156	78.00	580	290.00
Run 9	168	9.88	1542	90.71	159	79.50	505	252.50
Run 10	186	10.88	2078	114.70	197	98.50	540	270.00

After performing the above ten simulations the average values are calculated tabulated in tables 7-5 and 7-6 for analysis.

Table 7-5 Average rounds at FND and LND

PARAMETERS	Existing Design	Proposed Design
Number of Rounds at FND	179.2 rounds	171.2 rounds
Number of Rounds at LND	1904 rounds	579 rounds

Table 7-6 Average images sent at FND and LND

PARAMETERS	Existing Design	Proposed Design
Number of Images at FND	10.54 images	85.60 images
Number of images sent at LND	112 images	289 images

7.5 Memory requirement Analysis

The memory requirement of the proposed design and the existing design is analyzed against the memory storage size of the Mica2 mote. The table 7-7 shows the figures for the different memory consumptions. And as can be seen from table 4.1, parameters of mica2 mote the program memory available is 128KB. Even though the memory requirement of the proposed design is higher it can be accommodated.

Table 7-7 Memory requirements

	Required Memory by JPEG(KB)	Memory requirement of proposed(KB)
Code	5.21	1.42
Codebook	0	4.72
Total	5.21	6.14

7.6 Time of Encoding analysis

The time to encode or compress the image at the sensor node is important since it might create a delay to the transmission of the sensed image to the sink. And therefore in this work it has been found out that if the encoding is done in software it introduces a given amount of delay but if the encoding algorithm can be done in hardware a reduction of encoding time could be achieved by using a parallel search algorithms. The table 7-8 shows the encoding time comparison of the two algorithms.

Table 7-8 Time analysis

	JPEG	**Proposed**
Encoding / Compression Time(in seconds)	0.8393	1.2321

7.7 Image quality Analysis

In this section the image quality that could be retrieved at the sink or the receiver is analysed by using the two important quality measures. These are PSNR and MSE. They are quantitative values and the visual meaning can be watched to see if the images are giving a meaningful information. After the codebook is generated a test has been performed to check the quality of the images reconstructed in the decoding phase. All the images are used for training the neural net. For measuring the quality of the image a peak signal to noise ratio (PSNR) is used as a quality measure and structural content as subjective measure.

The PSNR is most commonly used as a measure of quality of reconstruction of lossy compression codecs (e.g., for image compression). The signal in this case is the original data, and the noise is the error introduced by compression. When comparing compression algorithms it is used as an approximation to human perception of reconstruction quality, therefore in some cases one reconstruction may appear to be closer to the original than another, even though it has a lower PSNR (a higher PSNR would normally indicate that the reconstruction is of higher quality). It is most easily defined via the mean squared error (MSE) which for two m×n pixel images I and K where one of the images is considered a noisy approximation of the other (the reconstructed image in this work) is defined as:

$$\mathrm{MSE} = \frac{1}{\mathrm{mn}} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \quad (7.1)$$

The PSNR is defined as:

$$\mathrm{PSNR} = 10 * \log_{10}\left(\frac{\mathrm{MAX}^2}{\mathrm{MSE}}\right) \quad (7.2)$$

Here, MAX is the maximum possible pixel value of the image. When the pixel are represented using 8 bits per pixel as here in this work the value of MAX is 255 [37].

The codebook trained using batch training is chosen to encode and decode the test images since using multiple images to train a codebook could be used based on the application of the WMSNs are deployed for. This way a better image quality at the receiver side could be easily obtained if the application is known before generating a codebook. The images shown below are the decoded standard images after being encoded using the proposed design. Their respective PSNR is shown in tables 7-9 and 7-10. Table 7-9 shows the PSNR values of the images that have been used as a training image in the generation of the codebook using SOM neural net. However, in table 7-10 the values are for images that have never been used as a training images are used for the first time just to test the algorithm.

The reconstructed standard images are shown in the figure 7.12. Even though it is true that the existing design which is jpeg can give a better image quality compared to this proposed design, the images reconstructed using the jpeg design are not included since they are almost similar to the original standard images, based on the application of the WMSNs the image quality is taken to be acceptable by this work both considering the structural contents of the images and the PSNR.

Figure 7.12 Reconstructed test images using the decoding phase

Table 7-9 Testing using Training Images

IMAGE	CAMERAMAN	LENA	PEPPERS	BABOON	GIRL	TIFFANY
PSNR	24.21	23.54	23.98	23.57	26.24	24.71

Table 7-10 Testing using other images

IMAGE	COUPLE	TREE	PIRATE	WOMAN-DARKHAIRED	WALK-BRIDGE	HOUSE	LAKE
PSNR	25.89	20.06	23.58	23.47	21.49	22.78	20.06

The test image used during the transmission process is Lena and its PSNR values are compared for the two designs as shown in table 7-11.

Table 7-11 PSNR values of Lena

	PSNR
Existing Design	30.33
Proposed Design	23.54

7.8 Summary of Analysis

Even though the image quality, time of encoding and memory requirements of the proposed design is not superior to the existing design as can be seen in tables 7-7, 7-8,7-9 and 7-10 ; it has become possible to increase the lifetime of the WMSNs to an extent that the nodes can stay more than twice of the existing design. Considering the lifetime using FND, especially to networks where the life time of the first node is critical; the FND lifetime has been increased. And during the proposed design 85 images has been transmitted before the first node is dead however in the existing design case only 10 images have been transmitted. From table 7-6 it can be seen that the number of images at FND has increased by a factor of 8.12. And again as shown in the table 7-6 the images communicated during the proposed design's life time at the LND are 289; however only 112 images are transmitted during the life time of the existing design by the time the last node dies. And taking LND as the parameter to show the lifetime of the network as a whole the increase percentage in the life time of the network can be calculated by taking the difference of the images transmitted in the two designs and dividing it by the images transmitted by the existing design (the original scenario). And a percentage increase of 158.03 is obtained. This indicates that the proposed design stays 2.58 times longer than the existing design at the LND. And from table 7-2 , comparing the size of the transmitted bits, for Lena, shows a decrease by a factor 8.39 compared to the existing design.

8. Discussion

8.1 Conclusion

The proposed design uses vector quantization using SOFM neural network for generation of a codebook. During the codebook generation two methods of training the neural network are followed and especially the batch training neural network can be used for optimal codebook for different applications of WMSNs. After comparison with the existing design a promising results have been obtained.

The size of the transmitted bit has been decreased by a factor 8.39 compared to the existing design. This decrease in the amount of transmitted bits has a double effect of increasing the networks life time as well as decreasing the bandwidth of the transmitted information. Since a given image can be sent in two packet instead of sending 17 packets for a single image it has an advantage for the intermediate nodes in which they will not be busy relaying these packets to the destination in case of multihop networks. At FND the network is able to transmit 85 images using the proposed design however it is able to transmit 10 images using the existing design.The number of images transmitted at FND has increased by a factor of 8.12 which means that a node starts to die in the existing design only after sending 10 images however in the proposed design a node dies after sending 85 images. And at the LND whcih indicates when the whole WMSN stops functioning and relaying a useful information to the sink. The existing design mulfunctions after 112 images and the proposed design after 289 images are sent by the last node to die and hence the maximum number of images that could be sent by any node at any location in the network. And hence, using the proposed design an increase of 158.03 percentages is achieved. And the network is alive more than two and a half times longer. This amount of increase in the network lifetime is obtained by using the proposed design at the graceful tradeoff with image quality. The image quality can be increased to be better by appropriate choice of training images and by adjusting the size of the code book and other network parameters.

8.2 Future Works

Future work may include measurements with different parameters of transmission power levels and also setting up an experimental test bed of the Mica2 motes. Additionally, other platforms should be considered, to create a more general model.

Future work may include performing a set of experiments with a wide experimental evaluation of the routing algorithms, in several scenarios of traffic load, node clustering, cluster size, channel conditions, and performance requirements.

Additional future work may include the study of the different protocol layers for functionality. The energy consumption of the different network layers could be considered for energy consumption on the models.

Furthermore, aggregation algorithms can be included in the cluster heads to further increase the network lifetime at an additional computational cost, and an analysis can be performed.

9. Bibliography

[1] Ozgur B. Akan, Nikil Jayant, Pascal Frossard," wireless multimedia sensor networks", in computer networks (Elsevier) journal, Atlanta, USA, August 2008.

[2] Ian F. Akyildiz , Tommaso Melodia, Kaushik R. Chowdhury," A survey on wireless multimedia sensor networks", Broadband and Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, United States, November 2006.

[3] Holger Karl,Andreas Willig, "Protocols and architectures for wireless sensor networks", John Wiley & Sons Ltd, England, 2005.

[4] Intel StrongARM SA-1100 Microprocessor Brief Data Sheet, intel product documentation, August 2000.

[5] ATmega 128(L) Preliminary Complete, ATmel product documentation, 2004.

[6] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-Aware Wireless Microsensor Networks", IEEE Signal Processing Magazine, 19: 40–50, 2002.

[7] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister "System Architecture Directions for Networked Sensors", in Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems, pages 93–104, Cambridge, MA, 2000.

[8] G. J. Pottie and W. J. Kaiser,"Embedding the Internet: Wireless Integrated Network Sensors. Communications of the ACM", 2000.

[9] J. Hill and D. Culler "MICA: A Wireless Platform for Deeply Embedded Networks", IEEE Micro, 22(6):12–24, 2002.

[10] J.Hill," System Architecture for Wireless Sensor Networks", University of California , Berkeley, 2003.

[11] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring", In Proceedings of the 1st ACM Workshop on Wireless Sensor Networks and Applications,Atlanta, GA, September 2002.

[12] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler "The nesC Language: A Holistic Approach to Networked Embedded Systems", Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation, pages 1–11. ACM Press, 2003.

[13] P. Levis and D. Culler. Mat´e, "A Tiny Virtual Machine for Sensor Networks", in Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems,San Jose, CA, October 2002.

[14] Tiny OS Hardware Designs < http://webs.cs.berkeley.edu/tos/hardware/hardware.html > , July 2010.

[15] Neeraj Kumar , Manoj Kumar and R.B.Patel, " Neural Network Based Energy Efficient Clustering and Routing in Wireless Sensor Networks" in First International Conference on Networks and communications, pp.34-39 , December 2009.

[16] Broadband Wireless networking lab, school of Electrical and computer Engineering, Georgia Institute of Technology , "Wireless Multimedia Sensor Networks (WMSN)", < http://www.ece.gatech.edu/research/labs/bwn/WMSN/ > , April 10 , 2010.

[17] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy efficient communication protocol for wireless microsensor networks", in Proc. of the Hawaii Int. Conf. on System Sciences, Hawaii, January 2000.

[18] K. Akkaya and M. Younis, "A survey of Routing Protocols for wireless sensor networks" in Elsevier Ad Hoc Network Journal, Vol. 3, pp. 325- 349, 2005.

[19] M. Cordina, C.J. Debono, "Increasing Wireless Sensor Network Lifetime through the Application of SOM Neural Networks", in Proc. Of the 3rd Int. Symp. on Communications, Control and Signal Processing (ISCCSP), pp. 467 – 471, Malta, March 2008.

[20] Vincent Lecuire, Cristian Duran-Faundez, Nicolas Krommenacker , "Energy-Efficient Transmission of Wavelet-Based images in Wireless Sensor Networks", EURASIP journal on image and video processing,article id 47345, 11 pages, Cedex, France, January 2007.

[21] Abdelhamid Mammeri, Ahmed Khoumsi, Djemel Ziou, Brahim Hadjou," Energy-aware JPEG for Visual Sensor networks",University of Sherbrooke, Maghrebian Conference on Software Engineering and Artificial Intelligence, Quebec,Canada, 2008.

[22] Yang Xiaobo, Sun Lijuan, Wang Ruchuan, "Distributed Image Compression Algorithm in Wireless Multimedia Sensor Networks", M.S. Thesis , College of computers,Nanjing University of Posts and Telecommunications,Nanjing, China, March 2010.

[23] M. Calle, "Energy Consumption in Wireless Sensor Networks Using GSP," M.S. Thesis, Department of Information Science & Telecommunications, University of Pittsburgh, Pittsburgh, PA, USA, 2006.

[24] Crossbow Technology,"Mica2 : wireless measurement system", datasheet , document part number: 6020-0042-07 Rev A, San Jose, California,USA,december 2006.

[25] Mica2 datasheet, < www.xbow.com > , August 2010.

[26] Paul A.Bender, ”Energy Efficient Image Video Sensor Networks”, Wright state university School of graduate studies, A dissertation submitted in partial fulfillment of the requirements for the degree of. Doctor of Philosophy, December 2008.

[27] Laurene Fausett,” Fundamentals of Neural Networks,architectures,algorithms and applications”, Prentice hall, December 1993.

[28] Self-organizing map, <http://en.wikipedia.org/wiki/Self-organizing_map > , sep. 2010.

[29] Crossbow Technology Inc, ” Wireless Sensor Networks, Getting started guide”, Rev.B, August 2004.

[30] Crossbow Technology Inc,” MPR-Mote Processor Radio Board MIB-Mote Interface / programming board user’s manual ” . Rev A.Document 7430-0021-05, December 2003.

[31] ATMEL, "ATmega128 ATmega128L Summary", Rev:2467LA-AVR-05/04, June 2010.

[32] Chok-Ki Chan,Lai-Man Po, “A complexity reduction technique for image vector quantization”, IEEE transactions on image processing, Vol.1,NO.3 , July 1992.

[33] Dong C. Park , SeungEok Choi, ”Modified Learning Vector Quantization for Image Compression”, Intelligent Computing Research Lab.,School of Electrical & Electronics Eng., MyongJi University, 2000.

[34] Mark Hudson Beagle, Martin T.Hagan, Howard B. Demuth,” Neural Network Toolbox 7 , User’s Guide ”, September 2010.

[35] Wikipedia,” Standard Images ”, < http://en.wikipedia.org/wiki/Standard_test_image > , September 2010.

[36] The USC-SIPI Image database, University of southern California,Signal and image processing institute Ming Hsieh Department of electrical engineering, < http://sipi.usc.edu/database/index.php > , September 2010.

[37] Peak signal to noise ratio, <http://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio>, September 2010.

[38] Charka Panditharathne, Soumya Jyoti Sen,” Energy Efficient Communication Protocol for wireless sensor networks”, Department of Electronics and Communication Engineering National Institute of Technology, Rourkela, Orissa, May 2009.

[39] Victor Shnayder, Mark Hempstead,Bor-rong Chen, Geoff Werner Allen, and Matt Welsh ,”Simulting the power consumption of large scale sensor network applications”, Harvard University, Division of Engineering and Applied Sciences, 2007.

[40] Alessio Falchi,"Sensor Networks: Performance Measurements with Motes Technology", Master thesis , University of Pisa, Engineering Faculty, Italy, 2004.

[41] Navraj chohan, "Hardware Assisted Compression in Wireless Sensor Networks", Department of Computer Engineering and computer science,university of California, Santa Barbara, UCSB Tech Report Report ID **:**2008-14 ,June 2006.

[42] G. Simon, P. V¨olgyesi, M. Mar´oti, and A. L´edeczi , "Simulation-based optimization of communication protocols for large-scale wireless sensor networks" In Proc. 2003,IEEE Aerospace Conference, Big Sky, MT, March 2003.

[43] Leif Uhsadel,"Comparison of Low-Power Public Key Cryptography on MICAz 8-bit Micro Controller",Faculty of Electrical Engineering and Information Technology, University of Bochum, Master thesis,April 2007.

[44] Atmel Corporation. 8-bit Microcontroller with 128K Bytes In-System Programmable Flash, Revision 2467N-AVR-03/06,2004.

[45] Aslam, Robetson ,Phillips, "Performance Analysis of WSN Clustering Algorithms using Discrete Power Control", Engineering Mathematics and Internet Working Department, Dalhousie,Canada, January 2009.

[46] Bhaskar Krishnamachari, "Networking Wireless Sensors", Cambridge University Press , New York, USA,January 2005.

[47] Paolo Santi ,"Topology Control in Wireless Ad Hoc and Sensor Networks", John Wiley & sons Ltd,England,July 2005.

[48] K. Dasgupta, K. Kalpakis, and P. Namjoshi, "An efficient clustering-based heuristic for data gathering and aggregation in sensor networks," Wireless Communications and Networking, WCNC , IEEE, vol. 3, 2003.

[49] M. Mallinson, P. Drane, and S. Hussain, "Discrete Radio Power Level Consumption Model in Wireless Sensor Networks," Mobile Adhoc and Sensor Systems, MASS, IEEE Internatonal Conference on, pp. 1-6,2007.

[50] A. Barberis, L. Barboni, and M. Valle, "Evaluating Energy Consumption in Wireless Sensor Networks Applications," Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)-Volume 00, pp. 455-462, 2007.

[51] Debdhanit Yupho, Joseph Babara,"The Effect of physical Topology on wireless sensor network lifetime", Graduate Program in Telecommunications and networking , University of Pittsburgh, Journal of Networks, Vol 2, No 5 (2007), 14-23, USA, September 2007.

[52] Vincent Lecuire, Cristian Duran-Faundez, Thomas Holl, Nicolas Krommenacker, Moufida Maimour, Michael David,"Energy consumption analysis of a Simple Image Transmission Protocol in wireless sensor networks", 6th IEEE International Workshop on Factory Communication systems, Italy, December 2006.

[53] K. Obraczka, R. Manduchi, and J. Garcia-Luna-Aveces, "Managing the information flow in visual sensor networks" in WPMC 2002: The Fifth International Symposium on Wireless Personal Multimedia Communication, October 2002, pages 1177 – 1181, October 2002.

[54]M. Calle and Joseph Kabara, "Measuring Energy Consumption in Wireless Sensor Networks Using GSP," in Proceedings of the 17th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications, PIMRC 2006, Helsinki, Finland.

[55] T.S. Rappaport, "Wireless Communications, Principles and Practice", Prentice Hall, 1996.

[56] Suraj Pandey, Byeong-Seob You, Ho Seok Kim, Young-Hwan Oh, and Hae-Young Bae, " Voronoi Based MBR Clustering for Spatial Queries in Sensor Networks" Department of Computer Science and Information Engineering, Inha University, the 4th Asian Symposium on Geographic Information Systems From Computer Science & Engineering View, pp.297-304 South Korea, June 2006.

[57] Mohammed Rahimi, Rick Baer,"Cyclops: Image sensing and Interpretation in wireless sensor networks", Proceedings of the 3rd international conference on Embedded networked sensor systems, New York, USA February 2005.

[58] JPEG,< http://en.wikipedia.org/wiki/JPEG>, October 2010.

[59] Andrea kulakov , Danco Davcev, and Goran Trajkovski, "Implementing Artificial Neural Networks in Wireless Sensor networks" in IEEE communication society Journal, Advances in Wired and Wireless Communication Symposium, Princeton, NJ,USA ,May 2008.

[60] Difference between bmp and jpg , <http://www.differencebetween.net/technology>, October 2010.