Pattern Transformation-Invariant Schemes for Wireless Sensor Networks Based on an Edge Detection Gradient-based Mechanism

Mohammed Abdulaziz A. Al-Naeem
Bachelor of Computer Information System (King Faisal University)
Master of Network Computing (Monash University)

A thesis submitted for the degree of Doctor of Philosophy at
Monash University in 2015
Faculty of Information Technology

Copyright notice

Abstract

In the last decade, a new computing platform, Wireless Sensor Networks (WSN), has emerged. This platform is an interconnected and distributed transducer network of tiny, inexpensive sensors, and it has emerged as one of the essentials of contemporary ubiquitous computing. A WSN's many tiny sensor nodes work together to perform one or more tasks, normally involving some type of monitoring, tracking and controlling. One of the main goals of WSNs is to sense physical environments and detect events occurring in the field of interest. Detection of events or materials of interest can be done by processing and analysing sensory information obtained by sensor nodes. Pattern recognition is one of the most useful and commonly utilised machine learning techniques in the literature for event detection in WSNs, especially when dealing with complex events. However, pattern recognition is highly affected by the limited resources offered by WSNs, including limited energy and limited computational, communicational and memory resources. In addition to limited resources, WSNs face other challenges in event detection, related to the dynamic nature of the environments in which WSNs are usually deployed. For example, a memorised pattern in a WSN pattern recognition scheme could appear in a different form, such as size dilation or location change, in the field of interest, or the WSN network's topology or sensor node locations might change, meaning the information memorised within the network will have different relations and distribution. The pattern recognition scheme also requires the capability of handling noisy patterns in order to maintain a high accuracy level. These noisy patterns are mainly the result of the monitoring environment and the limited lifetime of sensor nodes. Another issue associated with the nature of WSNs is the restricted number of training instances available, as events generally occur in some form of randomness. Therefore, designing a pattern recognition scheme for event detection in WSNs is a matter of a trade-off between detection accuracy, the use of limited available resources and dealing with existing challenges.

The first goal of this research project is to propose pattern recognition schemes capable of addressing the limitations associated with resource-constrained networks such as WSNs. The research first investigates the existing learning techniques for WSNs and their limitations. Then the research proposes two novel collaborative *in-network* pattern recognition-based event detection schemes which are lightweight and scalable and which suit resource-constrained networks such as WSNs well. In this research, two pattern recognition schemes are proposed: the Macroscopic Object Heuristics Algorithm (MOHA) and the Light Macroscopic Object Heuristics Algorithm (LMOHA). The main aim for proposing the second scheme (LMOHA) is to reduce the overall computational complexity of the MOHA scheme for event detection and pattern recognition. The proposed schemes adopt the distributed parallel recognition mechanisms of Graph Neuron (GN) to minimise recognition computations and communications and thus will lead to maintaining low levels of consumption of the limited resources. The distributed network structure of the proposed schemes will result in loosely coupled connectivity between a network's nodes and will avoid iterative learning. Therefore, the proposed schemes will perform recognition operations in a single learning cycle of predictable duration, which will make them good candidates for implementation of large-scale, real-time problems.

The second aim of this research project is to deal with a WSN's dynamic nature and limited prior knowledge of events. Thus, pattern transformation invariant schemes are proposed in this research. The first proposed scheme (i.e. MOHA) implements an edge detection gradient-based mechanism that searches the edges and boundaries of patterns and replaces traditional local information storing. The second proposed scheme (i.e. LMOHA) implements a similar mechanism as MOHA; however, its mechanism searches for the sensory-based shapes of patterns. These mechanisms allow the proposed schemes to identify dynamic and continuous changes in patterns. Consequently, the proposed schemes will be capable of performing recognition operations in dynamic environments and will also provide a high level of detection accuracy using a minimal amount of available information about patterns. Required protocols for performing the schemes' operations are also presented and discussed.

Theoretical and experimental analysis and evaluation of the presented schemes is conducted in the research. The evaluation includes time complexity, recognition accuracy, communicational and computational overhead, energy consumption and lifetime analysis. The schemes' performance is also compared with that of existing recognition schemes. This shows that the proposed schemes are capable of minimising computational and communicational overheads in resource-constrained networks, enabling those networks to perform efficient recognition activities for patterns that involve transformations within a single learning cycle while maintaining a high level of scalability and accuracy. The results show that a network that implements mica 2 motes and requires 3.0625 milliseconds to send a single message can perform recognition operations within a single learning cycle duration, ranging between 5.17 and 2231.39 milliseconds using the MOHA scheme and 5.17 and 16,441.33 milliseconds using the LMOHA scheme, for 40,000- and 65,536-node network settings, respectively. The results also show that using a multi-channel MAC message exchange model in both proposed schemes will considerably reduce the network's learning cycle time. The results also show that energy requirements can be decreased by up to 75.86 per cent using the MOHA scheme and by 70.69 per cent using the LMOHA scheme, in comparison to other recognition techniques. In terms of efficiency, theoretical and experimental analyses show that both proposed schemes are highly capable of dealing with noisy and transformed patterns with a high level of accuracy. However, each presented different limits of tolerance to noisy patterns and different types of transformed patterns. The results show that the MOHA scheme offers more accurate recognition for scaled patterns than the LMOHA scheme. However, the LMOHA scheme provides more accurate recognition for noisy and rotated patterns than the MOHA scheme. In conclusion, both proposed schemes showed a very significant capability of performing pattern recognition in WSNs, as they showed a very good capability of handling noisy and transformed patterns and limiting the number of communications, and hence, limiting the use of energy resources.

Finally, the research presents and discusses several simulations for each proposed scheme. The results of these simulations show that the proposed schemes have a very high accuracy in dealing with transformed patterns compared to other existing schemes. The results also show the capability of the proposed schemes' networks of performing complex and real-life recognition problems by using a minimal amount of training information. They also show the feasibility of utilising the proposed schemes in real scenarios and different application domains.

Declaration

In accordance with MONASH University Doctorate Regulation 17 / Doctor of Philosophy and Master of Philosophy (MPhil) regulations the following declarations are made:

I hereby declare that this thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

[redacted]

Mohammed A. Al-Naeem
August 31, 2015

This thesis is dedicated to my beloved parents, my wife Fatimah, and my children, who inspired me and sparked my interest to pursue higher education and who provided me with support, help and encouragement every moment along the long academic road that I followed.

Acknowledgements

I take this opportunity to thank and acknowledge everyone who has helped me and encouraged me throughout my PhD study. This research project would not have been possible without their great support.

First and foremost, I am grateful to the Almighty GOD, Allah, for the unlimited help I have received during my life and to complete this thesis.

I would like to express my sincere and boundless gratitude to my supervisors, Dr Asad Khan and A/Prof. Andrew Paplinski for their devoted support and guidance on my research and professional development. I owe this thesis to them; without their training and help, I could not have developed a solid background to carry out my research work and produce the research outcomes presented in this thesis.

I would like to thank Mrs. Bruna Pomella for proofreading my thesis. Special thanks go to my colleagues Dr Waleed Alfehaid, Dr Amiza Amir, Dr Anang Amin and Mr. Amir Basirat for their help and support.

I would also like to convey thanks to King Faisal University (KFU) for providing me with a PhD scholarship and Monash University for providing the financial means allowing me to attend a number of international conferences. I must thank A/Prof. Bader Aljohar, Dr Khalid Buragga, Dr Mohammed Alzahrani, Dr Majed Alshamari and the staff at the College of Computer

Sciences and Information Technology at KFU for encouraging me to pursue a PhD and for their unfailing support.

Last but not least, I wish to express my deepest gratitude to my beloved family for their endless encouragement throughout the duration of my studies. I thank my wonderful parents, my father Abdulaziz and my mother Badriyah, for their belief, prayers and unconditional support throughout everything. It is to them that I owe everything I am. I would like to give a special thank you to my lovely wife Fatimah for her care, patience, understanding and prayers to complete this research work. In addition, I would like to acknowledge my little cute and smart daughters Yara and Bedor who brighten my every day. I also thank my brothers and my sisters for their love.

I also thank my friends Md. Mamunur Rashid, Ahmed Alnuaim, Naif Almuslem, Abdullah Alamer, Saeed Jahlan, Yousef Sahari, Ali Alamir, Abdulrahman Almosa, Mohammed Alhabeeb, Abdullah Almuhaideb and the rest of my friends for supporting me and being my family during my PhD studies.

Thank you all for making my dreams come true.

Mohammed A. Al-Naeem

Monash University

August 2015

Outcomes/Publications

The outcomes of this thesis work have been reported in the following publications:

International Conferences

1. M. Al-Naeem and A. I. Khan, “MOHA: A Novel Target Recognition Scheme for WSNs”, The 7th International ICST Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous 2010), Sydney, Australia 2010.
2. M. Al-Naeem, “Intelligent Target Recognition in Wireless Sensor Networks Using a Pattern Recognition”, The Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering (PARENG2011), Ajaccio, Corsica, France 2011.
3. M. Al-Naeem and A. I. Khan, “A Novel Target Recognition Scheme for WSNs”, WCCI 2012 IEEE World Congress on Computational Intelligence, June, 10-15, 2012 - Brisbane, Australia.

Poster Presentations

4. M. Al-Naeem and A. I. Khan, “MOHA: A Macroscopic Object Heuristics Algorithm for WSNs”, in the Monash FIT HDR Conference 2012, Monash University, Australia, October 25, 2012.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Preamble

In the current era of technological advancements, the progress of information technology has followed Moore’s law which states: “The complexity for minimum component costs has increased at a rate of roughly a factor of two per year. Certainly over the short term this rate can be expected to continue, if not increase” [1]. The evolution of computer technology has gone a step ahead by offering the control of devices in our environment. These devices have embedded systems that are small and often have specific purposes, which can be found in countless applications and devices such as traffic lights, cars, medical equipment, and even smart phones. We are also moving towards an era of ubiquitous computing, of which Marc Weiser, the father of ubiquitous computing said, “The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it” [2]. He predicted that computers will be seamlessly and effectively integrated in all aspects of our daily lives and will be integrated in our environment without their existence being noticed.

In the last decade, a new computing platform, namely, Wireless Sensor Networks (WSNs), has emerged. This platform is an interconnected and distributed transducer network of tiny, inexpensive sensors, and has emerged as one of the essentials of contemporary ubiquitous computing. A WSN consists of many tiny sensor nodes that work together to perform one task or more, normally involving some type of monitoring, tracking, and controlling. With the recent advances in micro-electromechanical system (MEMS) technology and wireless communications, building a low-cost, low-power WSN with multifunctional sensors has become an achievable goal that is receiving a lot of attention [3-11]. These sensors are tiny in size and capable of sensing, processing data, and communicating with each other, normally via a radio frequency (RF) channel [3]. The main purpose of sensor nodes in a WSN is to provide a view of the observed area by interacting and exchanging information. These smart sensor nodes are called “motes” and were developed as part of the smart dust project at the University of California in Berkeley [12, 13].

Figure 1.1 shows an example of a WSN network in action. In this example, sensor nodes are distributed over a large geographical area in order to detect certain objects (e.g. aircraft and tank) or events (e.g. fire and other natural phenomena) in their vicinities. Each one of these nodes is supported with a CPU, a battery, a sensing capability, and a communication element. Moreover, all nodes are placed in ideal positions to help sense objects or events accurately. The sensor nodes regularly scan the observed area. Subsequently, they analyse the collected data, exchange information and pass the results to the base station where the user can utilise the information.

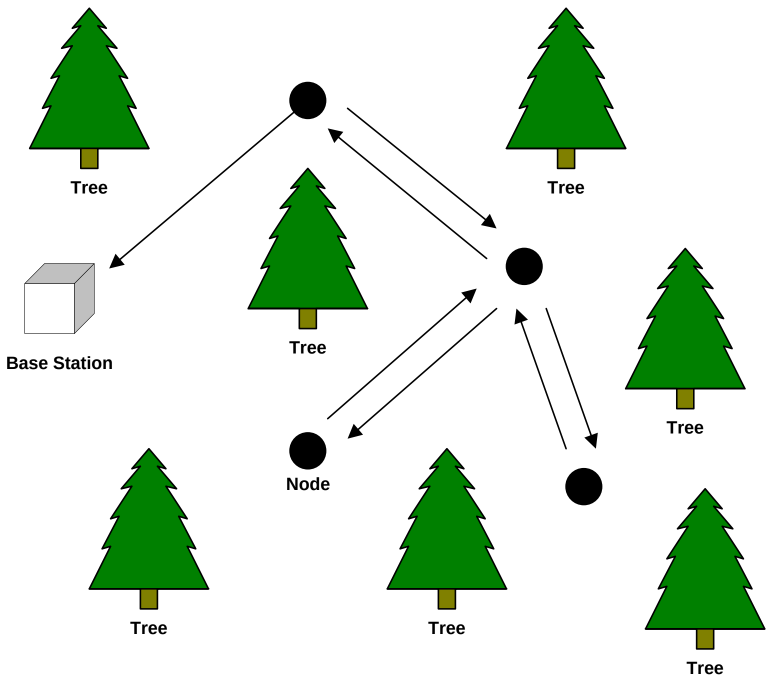

Figure 1.1: An example of a WSN [13].

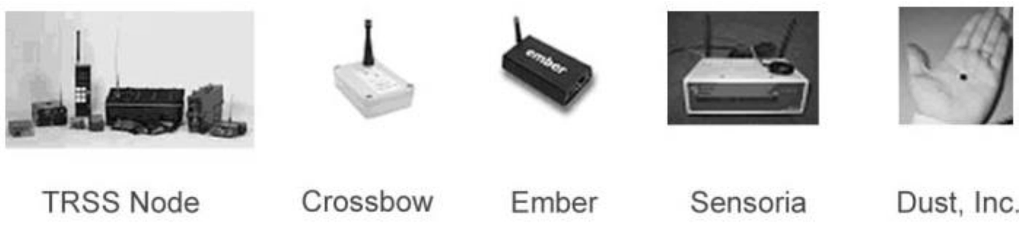

Figure 1.2: Examples of sensor nodes [14].

The size of a sensor node may vary. Some nodes are as small as a speck of dust, whereas others are nearly one cubic centimetre in size as shown in Figure 1.2. A sensor node consists of five main components: a Memory, a Controller, a Communications device, Sensors/ actuators, and a Power supply (see Figure 1.3). All these components should work together in order to accomplish the sensor's assigned task. Moreover, the limited power supply should be taken into consideration when components perform their operations [15-18].

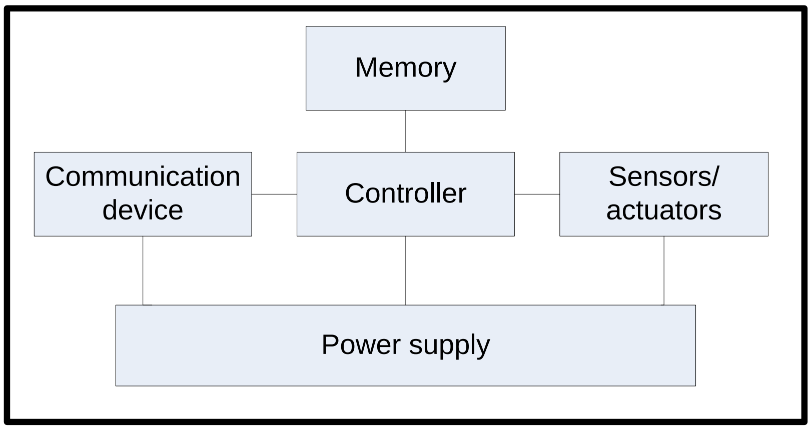

Figure 1.3: Hardware components of a sensor node [15].

The main features of sensor networks are [3, 4, 16, 19-21]:

- They are able to self-organise.
- They have short-range broadcast communication and multihop routing.
- They are densely deployed and cooperative in their efforts.
- The topology of a WSN is frequently changing owing to node failures and signal fading.
- They have limited transmission power, computing power, energy, and memory.

These features, especially the last three, make wireless sensor networks different from mesh and wireless ad hoc networks.

WSNs can consist of many different kinds of sensors including seismic, magnetic, thermal, visual, infrared, acoustic, and radar, which are capable of monitoring a wide variety of ambient conditions including [22-24]: temperature, humidity, pressure, speed, direction, movement, light, soil composition, noise levels, the presence or absence of certain kinds of objects, and mechanical stress levels on attached objects. Hence, a great variety of WSN applications are possible. This range of applications includes homeland security, monitoring of space assets for potential and human-made threats in space, ground-based monitoring of both land and water, intelligence gathering for defence, environmental monitoring, urban warfare, weather and climate analysis and prediction, battlefield monitoring and surveillance, exploration of the Solar System and beyond, monitoring of seismic acceleration, strain, temperature, wind speed and GPS data.

In order to understand the existing and potential applications of wireless sensor networks, sophisticated and extremely efficient communication protocols are required [23]. A typical WSN consists of a large number of sensor nodes densely deployed either inside a physical phenomenon or very close to it. In order to enable reliable and efficient observation of the environment and to initiate the most appropriate responses, the phenomenon’s physical features should be reliably detected/estimated from the collective information provided by the sensor nodes [22, 23]. Moreover, instead of sending the raw data to the nodes that are responsible for fusion, sensor nodes utilise their processing capabilities to locally carry out simple computations and transmit only the required and partially processed data. This process is called *in-network* processing which is intended to find the most pertinent knowledge from the observed environment or the object. Hence, these properties of wireless sensor networks present unique challenges for the development of communication protocols and detection algorithms.

1.2 Observation of the Environment

One of the main factors driving the research on wireless sensor networks is their sensing capability that allows them to measure their surrounding (pressure, light, humidity, temperature, vibration etc.), and sense motions in the physical world. Utilising satellites for observation purposes has its advantages. For example, a satellite's ability to provide a real-time weather map is outstanding. However, it cannot be used to record temperature or sounds, or monitor objects inside houses or caves. On the other hand, wireless sensor nodes can be deployed very densely within the environment, in close proximity to all events of interest [25].

Some sensor applications still depend on wired sensor nodes. However, it is anticipated that wireless sensor nodes will replace them for the following reasons [26]. Firstly, wired sensor networks require more time to be configured and deployed, precluding applications that require immediate data collection. This makes them infeasible for certain applications, such as automotive or battlefield surveillance, where mobility and rapid deployment are essential. Moreover, the cost of deploying wired sensor networks is very high because of the cost of wiring which can range from $40 to $2000 per linear foot of wire. The cost of existing applications can be reduced by replacing wired sensors with wireless sensors. Wireless sensor networks can also allow new applications that would otherwise be impossible. According to data collected from Freedonia Group and Frost & Sullivin [26, 27], the hardware market for wireless sensors is expected to rise at the rate of 20% per year, three times more than the market for wired sensors.

The Great Duck Island project (GDI) [28] is one of the earliest and most cited examples of a wireless sensor network application. The GDI project showed how the wireless sensor network technology can be generally very useful for ecosystem monitoring and particularly useful for the observation of bird-nesting grounds. In the past, scientists had to frequently visit every bird burrow and record their observations. This procedure was labour-intensive and disturbed the birds' habitat. In the GDI project, shoe-box-size sensor nodes were installed in or close to the burrows of petrels in order to collect useful live data and pass them on to the web. Moreover, scientists were able to remotely perform on-the-fly experiments and collect useful data from previously unreachable locations.

While wireless sensor networks are well-known for their ability to perform efficiently in applications concerning biodiversity monitoring and observation, they are also utilised in a variety of other applications. For instance, in recent years, wireless sensor networks have made an appearance in the healthcare domain[29]. These wireless sensor networks carry the promise of being able to greatly improve and expand the quality of care in a number of contexts and for different sectors of the population. For example, primary system prototypes have verified the potential of wireless sensor networks to: enable early detection of clinical deterioration via real-time patient monitoring in hospitals [30, 31], improve first responders' ability to provide emergency care in huge disaster zones via automatic electronic triage [32, 33], enhance the health and life quality of the elderly people via smart environments [34], and assist in large-scale field studies of human behaviour and chronic diseases [35, 36].

In wireless sensor network applications, the most predominant model involves transferring sensory data to a base-station for analysis [14, 19, 37]. After the deployment of sensor nodes in the observed field, sensor nodes collect sensory data from their particular locations either continuously, periodically, initiated by users, driven by events, or by hybrids of these methods [38]. These collected sensory data are locally processed before being transmitted to a centralised processing entity, namely, the base station, where further information processing may take place. At the base-station, the collected data is integrated and analysed to infer the status of the observed field.

1.3 The Effects of the Nature of WSNs

One of the main goals of WSNs is to sense physical environments and detect events occurring in the field of interest [39, 40]. The detection of events or objects of interest can be performed by processing and analysing sensory information obtained by sensor nodes. Pattern recognition is one of most useful and commonly utilised machine learning techniques in the literature for event detection in WSNs, especially when dealing with complex events [23, 25, 41-50]. As WSNs are normally deployed in large numbers, the network scaling property is an important consideration when designing a pattern recognition scheme for WSNs. Size scaling requires managing the way sensor nodes are going to communicate with each other. The number of communications involved in a WSN scheme design is critical as it will determine the number of communications each sensor is going to handle. This will have consequences for the sensor's lifetime as well as the time required to obtain a final result from these communications. In real-time applications, convergence time is extremely important. In pattern recognition processes, sensor nodes send data to either a fusion centre (i.e. base station) or to other sensor nodes in the network in order to conclude pattern detection. Consequently, the network's convergence time is highly dependent on the process of delivering information from one point to another. In general, the scaling of a WSN must ensure a limited amount of communication in order to conserve energy resources and speed up the recognition process to support real-time applications of WSNs.

Furthermore, a pattern recognition scheme should have several invariant features. In WSNs, the need for such features increases because WSNs are dynamic and the nature of monitored fields of interest is changing. In other words, a memorised pattern in a WSN pattern recognition scheme could appear in different form, such as size dilation or location change, in the field of interest. Or the WSN network's topology or sensor node locations might change, meaning that the memorised information within the network will have different relations and distribution. Another issue associated with the nature of WSNs is the restricted number of training instances available as events generally occur in some form of randomness [42, 51]. Therefore, the design of a pattern recognition scheme must take into account the restricted amount of training data available and the changing environment in WSN networks and fields.

1.4 Challenges in the Development of Sensor Network Applications

Sensor networks are often compared with ad-hoc networks such as Mobile Ad Hoc Networks (MANET) [14, 19, 52-54]. Similar to the MANET nodes, sensor nodes are battery powered and communicate wirelessly over a limited bandwidth with the communication channel being prone to errors. However, they have many characteristics that differentiate them from ad-hoc networks. First of all, sensor nodes are densely deployed in the field of interest sometimes over a vast geographical area to achieve a single goal, which is to sense the observed field and communicate sensory data. Highly dense deployment leads to significant amounts of redundancy in sensory data, and as a result, sensor network designs usually incorporate redundancy-removal techniques.

Additionally, sensor networks adopt *data-centric* approaches, where the emphasis is on finding methods to efficiently deliver sensory data without using nodes’ identification numbers (IDs). Utilising the nodes' identification numbers to route sensory data might not be reliable or efficient due to the sensor nodes’ dynamics, which might unexpectedly lead to those nodes going offline as a result of consuming all of their energy resources; hence, data aggregation methods can be used to reduce the amount of communicated sensory data in the sensor network [55].

Secondly, unlike *point-to-point* communication in ad hoc networks, in sensor networks, sensory data are broadcast to the base-station. As a result, sensor networks have a many-to-one data flow. Thirdly, even though both networks are battery powered, the replacement or recharging of sensor node batteries is sometimes not possible. Lastly, in contrast to user-driven ad hoc networks, sensor networks are closely coupled with the observed environment and are often driven by the stimuli of environmental events.

The main factors that uniquely challenge the design of sensor networks applications are as follows:

- **Ad hoc deployment:** Even though sensor nodes might be deployed in deliberately chosen locations, most sensor applications deploy sensors purely randomly, with no clear structure [56-58]. In the initial stages of deployment, usually sensor nodes are installed in full in the environment and then more nodes are added to the network in order to increase the coverage and the quality of measurement [58].

- **Lifetime:** Sensor networks are considered for long-term deployment in remote, sensitive, and inaccessible locations in the observed environment. In some cases, providing power sources or replacing sensor nodes is infeasible and therefore the sensor nodes continue to operate until they run out of battery power.
- **Unattended operations:** Sensor nodes are required to operate independently without any dependence on human intervention or supporting infrastructure due to the huge number of sensor nodes and the hostility of the observed environment [57, 59].
- **Dynamic changes:** Sensor networks are prone to network changes such as the addition of sensor nodes or node/link failure. Moreover, event occurrence might be unpredictable and random. Therefore, the network dynamics and changes in the environment affect the sensor network's operation and longevity [19].
- **Scale:** A sensor network may be comprised of numerous nodes in order to support large-scale temporal and spatial sampling. Furthermore, the network might have to convey information about many events occurring simultaneously over a huge geographic area [59].

1.5 Motivation

A WSN consists of many tiny sensor nodes that work together to perform one task or more, normally involving some type of monitoring, tracking, and controlling. One of the main aims of WSNs is to sense physical environments and detect events occurring in the field of interest [39, 40]. Detection of events or materials of interest can be done by processing and analysing sensory information obtained by sensor nodes. As discussed previously, pattern recognition is one of the most useful and commonly utilised machine learning techniques in the literature for event detection in WSNs, especially when dealing with complex events [23, 25, 41-50]. It is one of the key capabilities required for processing the collected information. It is envisaged that a new form of capability, in monitoring systems, can be introduced by having pattern recognition operate within a wireless sensor network. However, using a conventional pattern recognition technique within wireless sensor networks requires modifications to the existing schemes so as to address resource limitations [60].

Graph-matching pattern recognition provides a universal representation formalism that may be well-suited to pattern recognition in wireless sensor networks [61]. Unfortunately, the graph-based algorithms also become computationally prohibitive with an increase in size of the pattern database [62]. This increase makes the employment of those algorithms in resource-constrained wireless sensor networks a difficult task. Graph Neuron (GN), proposed in [63], is an approach that conserves resources through algorithmic simplicity and allows fast template matching. Hence, the scheme can decrease the computational cost of processing in-situ information gathered by sensor nodes.

However, in order to have a high level of recognition accuracy, the pattern recognition scheme for event detection in WSNs should be able to identify dynamic and continuous changes in patterns, which in this research is called 'the capability of dealing with transformed patterns'. Most pattern recognition schemes such as GN and K-nearest neighbour cannot deal with these types of patterns, as will be disscussed in Chapter 2. In WSNs, these capabilities are more important because WSNs are dynamic and the nature of the monitored fields of interest is changing. As stated previously, a memorised pattern in a WSN pattern recognition scheme could appear in a different form, such as size dilation or location change, in the field of interest. Also, the WSN network's topology or sensor node locations might change, meaning that the memorised information within the network will have different relations and distribution. Moreover, the pattern recognition scheme must be able to handle noisy patterns in order to maintain a high level of accuracy. The noisy patterns are mainly the result of the monitoring environment and the limited lifetime of sensor nodes. The noisy patterns, damage to the sensor nodes, dead sensor nodes, and lost packets might cause the loss of some parts of the detected pattern. Therefore, generally a pattern recognition scheme for WSNs should be capable of detecting events and patterns even if some parts of the detected pattern are lost. Another issue associated with the nature of WSNs is the restricted number of training instances available as events generally occur in some form of randomness [42, 51]. This randomness can be tackled by using adaptive learning techniques that are capable of searching for similarities between a stored event and a currently encountered one. These techniques store event patterns, allowing WSNs to learn from experience and develop information about patterns. Therefore, the design of a pattern recognition scheme must address two issues: the restricted amount of training data available, and the changing environment in WSN networks and fields.

Moreover, pattern recognition is highly affected by the limited resources offered by WSNs, including limited energy and limited computational, communication and memory resources. In most cases, the sensor nodes are powered by independent sources such as on-board batteries, so the lifespan of the sensor node is often determined by the amount of processing and communication it carries out. Clearly, the energy-efficiency of the network will be increased by reducing the amount of communication between sensor nodes while maintaining the overall performance of the wireless sensor network. The reduction of communication also reduces the excessive concentration of communication traffic at the base node. The issue of excessive traffic at the base node has been shown to dramatically degrade the overall performance of the network [43, 64, 65].

Moreover, the lifespan of the wireless sensor network can be increased by activating only a subset group of sensor nodes. In many sensor network applications, not every sensor node always provides useful information. As a result, pattern recognition approaches in WSNs require the activation of only a subset group of sensor nodes with information useful for pattern recognition, and switching off as many senor nodes as possible to conserve their energy.

Therefore, through this study, we shall propose a pattern recognition scheme for event detection that is capable of detecting transformed and noisy patterns using a minimal amount of available information about patterns while addressing the resource constraints of WSNs. Unlike traditional pattern schemes, this research work adopts mechanisms to conserve energy while effectively performing pattern recognition.

1.6 Hypothesis and Research Objectives

Based upon the research motivation stated in the previous section, the main hypothesis of this research is that the best means of addressing the issue of event detection in WSNs is to have a pattern recognition algorithm combined with fully distributed and parallel techniques, which works purely with localised node adjacency-based relationships (i.e. computations). Moreover, it will do so without requiring huge computational resources, making it suitable for wireless sensor networks. The parallelism helps to reduce the amount of processing required to detect an event by sharing these computations among sensor nodes. This capability makes it possible for such a scheme to perform quite complex operations by dividing them into simple processes that are suitable for resource-limited sensor nodes. Adjacency-based relationships (i.e. computations) will increase a WSN’s ability to handle complex, invariant (i.e. transformed), and noisy patterns, which will in turn increase the recognition accuracy. Additionally, this research expects that the use of a loosely coupled connectivity scheme will scale up efficiently in terms of time and resources management when it is utilised in resource-limited networks like WSNs. By offering a fast and accurate scheme that suits WSNs, it is possible to utilise such scheme to address real-time application problems.

The main objective of this thesis is to propose a pattern recognition scheme for event detection that is capable of detecting transformed and noisy patterns using a minimal amount of available information about patterns while addressing the resource constraints of WSNs. In order to achieve the main aim of this research and support its hypothesis, a number of objectives have been formulated as follows:

1- To conduct a review on current pattern recognition schemes. The capabilities of current pattern recognition schemes to be utilised in a resource constrained environment such as a wireless sensor network will be studied. Subsequently, relevant schemes will be tested to check their ability to:

- have restricted communications, computations, and memory requirements;
- scale in terms of network size;
- have predicted convergence time;
- address invariance properties for dynamic networks and changing patterns;
- detect complex and noisy patterns;
- solve randomness problems, meaning that the scheme should maintain high accuracy with a restricted number of available training instances.

2- To propose a new pattern recognition scheme for event detection in WSNs that is capable of detecting transformed and noisy patterns. Moreover, the proposed scheme will be compared with current pattern recognition schemes in order to ascertain its capabilities in performing accurate pattern recognition, which includes the ability to detect transformed, complex and noisy patterns.

3- To investigate the capability and effectiveness of the proposed scheme to address problems associated with event randomness. The proposed scheme will be compared with existing pattern recognition schemes to examine its ability to maintain high accuracy with a restricted number of available training instances.

4- To perform extensive evaluation and analysis of the complexity of the proposed scheme in terms of memory size, number of communications, and learning cycle time. Additionally, the proposed scheme will be compared with current pattern recognition schemes in these terms.

5- To increase the lifetime of the network by activating only a subset group of sensor nodes with information useful for pattern recognition and switching off as many sensor nodes as possible to conserve their energy. This will lead to an increase in the overall lifespan of the network.

6- To conduct a study of the performance of the proposed scheme under varying networking parameters. The proposed scheme will be analysed to study the influence of different types of MAC protocols on it. MAC message exchange models for the proposed scheme to function over these protocols will be proposed. It is anticipated that these models will assist in estimating the time and resource requirements of the proposed scheme in terms of the communication overhead imposed on the underlying network of wireless sensors.

7- To investigate the capability and effectiveness of the proposed scheme to be utilised for complex and real-life problems. Two of the best-known WSN applications are used for this purpose, namely, handwritten character recognition and human activity recognition. For Human Activity Recognition systems to be functional, two types of data collection methods are generally used: camera-based and sensor-based. The recognition accuracy of the proposed scheme will be evaluated based on these two types of data. These study cases will prove the usability and advantages of the proposed scheme to be utilised in real-life cases and different application domains.

1.7 Research Contributions

The contributions of this thesis can be summarised as follows:

1- **The development of new pattern-recognition-based event detection schemes for resource-constrained networks such as WSNs:** The developed schemes address the limited resources of WSNs by performing communication and computational tasks in a distributed manner. They involve a distribution technique to split the patterns into sub-patterns and utilise only a subset group of sensor nodes in the recognition process. This technique will reduce the number of sensor nodes involved in the recognition process, and in turn will reduce the amount of communication required for recognition; moreover, the remaining nodes can be switched off or deactivated. Consequently, the distributed techniques will be more suitable for real-time and real-life WSN applications, especially those that require pattern and event detections over a large area in terms of communications and computations complexity.

2- **The development of pattern transformation-invariant schemes for WSNs:** In large regions that are under surveillance, patterns and events occur randomly and changeably. A pattern or an event could occur in a certain part of the region with a set of characteristics and it may take a long time for a similar pattern or event to occur in a different part of the region with variations of the previous characteristics. Thus, the proposed detection schemes will address randomness and changing phenomena by adopting techniques that make it possible to store patterns and recognise transformations in patterns such as translation, dilation, and rotation.

3- **The development of an edge determination mechanism based on a well-established edge detection technique of image segmentation that offers to the recognition schemes the capability of transformation-invariant detection:** The event may spread over an area of irregular shape. In WSN, events can be described as shapes (i.e.

objects or boundaries) but not visual shapes. They can be described as sensory-based shapes. The knowledge obtained from edge detection techniques in image processing and segmentation can be utilised to recognise events that produce geometric shapes on the basis of the sensor readings. For example, a forest fire will have an edge at the fire front based on temperature reading, a flood will have an edge based on moisture or pressure, and a tornado will have an edge based on pressure. The proposed schemes use an edge detection mechanism to locate and determine the edges and the boundaries of an event. By describing events and patterns using their main edges and boundaries, it is possible to achieve an efficient recognition scheme that can detect transformations that may occur in these events and patterns. To the best of our knowledge, the schemes proposed in this research are the first pattern recognition schemes to utilise edge determination mechanisms based on a well-established edge detection technique of image segmentation in the recognition process, which can offer transformation-invariant detection capabilities. The first proposed scheme in this research implements an edge detection gradient-based mechanism that searches the edges and boundaries of patterns and replaces traditional local information storing. The second proposed scheme implements a mechanism similar to the first one; however, its mechanism searches for the sensory-based shapes of patterns. These mechanisms allow the proposed schemes to identify dynamic and continuous changes in patterns.

4- **The development of communication protocols for pattern recognition in resource-constrained networks:** Communication protocols are needed for WSNs to be functional in terms of detection techniques. These protocols will be presented in this thesis. They will describe the tasks and communications required by network nodes when learning and recognising patterns, from sensing the data to producing the result.

5- **The design of a pattern-recognition-based classification model for WSNs:** Complex classification problems are common challenging tasks for resource-constrained networks such as WSNs. In this thesis, a pattern-recognition-based classification model is proposed. This model demonstrates the ability of the proposed schemes to perform classification tasks with minimal resource requirements using pattern recognition capabilities while maintaining high accuracy compared to other classification schemes. This model shows the advantage of using pattern recognition capabilities in solving complex classification problems.

6- **An analysis and evaluation of the proposed schemes:** Analysis and simulations for the proposed pattern recognition schemes will be conducted in this thesis. This will include time complexity analysis, analysis of the effects of pattern translations, and determining the accuracy levels of various means of pattern detection. Moreover, this thesis will provide a comparison between the proposed schemes and other existing pattern recognition schemes in terms of accuracy, communication overhead, and computational overhead.

1.8 Thesis Structure

This thesis is divided into seven chapters as follows:

- **Chapter 1:** The introductory chapter provides overall information about the research area. The chapter also discusses the research’s motivation, objectives and contributions. Finally, it presents the structure of the thesis.
- **Chapter 2:** The Pattern Recognition in Wireless Sensor Networks chapter presents the background of pattern recognition in wireless sensor networks. It also provides a detailed analysis of existing pattern recognition schemes for WSNs, including threshold-based, template matching, nearest neighbour, statistical, syntactical, fuzzy logic, and neural networks techniques. Moreover, issues related to the implementation of such schemes for pattern recognition will be discussed. Finally, a set of metrics to evaluate the suitability of existing schemes for detecting changing events using WSNs will be presented and a comparison of these schemes provided.
- **Chapter 3:** This chapter proposes a novel pattern detection scheme for WSNs: Macroscopic Object Heuristics Algorithm (MOHA). The scheme will provide transformation and noisy pattern recognition capabilities in a

lightweight and distributed manner that suits WSNs. It implements an edge detection gradient-based mechanism that searches the edges and boundaries of patterns and replaces traditional local information storage. This mechanism allows the proposed scheme to identify dynamic and continuous changes in patterns. The scheme reduces the number of nodes and required computations, memory resources, and number of communications needed for performing pattern recognition in WSNs. This is achieved by adopting a distributed and parallel design, along with efficient activation processes. This scheme is fault-tolerant and speeds up recognition by leveraging the parallel distributed processing capabilities of WSNs. An extensive analysis of the presented scheme will be provided, including time complexity and simulation tests. Finally, the proposed scheme will be tested and compared with existing schemes in terms of accuracy.

- **Chapter 4:** The Light Macroscopic Object Heuristics Algorithm (LMOHA) chapter proposes a lighter version of MOHA scheme, intended to reduce the computational complexity of the MOHA’s S&I for pattern recognition, which will lead to a reduction in the overall computational complexity of the MOHA scheme. Moreover, the LMOHA scheme offers the same capabilities of the MOHA scheme in dealing with noisy and transformed patterns. The LMOHA implements an edge detection gradient-based mechanism (similar to the MOHA’s mechanism) to search for the sensory-based shapes of patterns. This mechanism allows the LMOHA scheme to identify dynamic and continuous changes in patterns and reduce the computational complexity of recognition operations. As with the MOHA, the scheme adopts the GN approach to maintain minimal communication and computational requirements to thus provide a lightweight pattern recognition scheme that suits resource-constrained systems and networks such as WSNs. This chapter also presents descriptions of the required protocols needed to enable the proposed scheme to be implemented in network environments. Furthermore, a series of analyses, evaluations and simulations for LMOHA implementation is provided along with a comparison between it and MOHA scheme.

- **Chapter 5:** The Medium Access Control Protocols Influence on MOHA and LMOHA chapter discusses different types of existing MAC protocols and MAC message exchange models for the MOHA and LMOHA schemes to function over these protocols. These models will help in estimating the time and resource requirements of the MOHA and LMOHA schemes in terms of the communication overhead imposed on the underlying network of wireless sensors. Additionally, this chapter presents an experiment-derived evaluation of MOHA and LMOHA schemes' communicational overhead in terms of time and energy requirements. The main objective of this experiment is to estimate the lifetime and execution duration of the networks utilising the communicational models (MAC protocols) present in the beginning of this chapter. Finally, this chapter concludes with an overall comparison of the MOHA and LMOHA schemes in terms of network structure, pattern matching criteria, handling noisy and transformed patterns, number of nodes participating in recognition, node utilisation, and network's lifetime.

- **Chapter 6:** This chapter, Evaluating the Performance of MOHA and LMOHA Schemes, describes a series of simulations conducted for both proposed schemes. Also, it compares the proposed schemes with other well-known pattern recognition schemes. Three tests will be presented and discussed in this chapter. The first test is intended to compare the accuracy of the proposed schemes with well-known existing schemes in dealing with transformed patterns. The second test is used to ascertain the capability of MOHA and LMOHA schemes to deal with complex and real-life problems. This test will involve the recognition of handwritten characters. This test will show that both proposed schemes are capable of performing a recognition operation using a minimal number of training samples while still maintaining a high level of accuracy compared with other schemes. The third and final test is intended to prove the possibility of using the proposed schemes for real classification problems. One well-known application that can be utilised for this purpose is human activity recognition. In this test, the proposed schemes will be compared with other existing techniques using a limited amount of collected data to demonstrate the capability of the scheme in addressing activity recognition problems using two different types of datasets, namely, vision-based and sensor-based datasets.

- **Chapter 7:** The Conclusion and Future Work chapter concludes the thesis by summarising the contributions of this research. Additionally, issues related to proposed schemes and future work will be presented in this chapter.

Chapter 2

Pattern Recognition in Wireless Sensor Networks

2.1 Preamble

The main task of a wireless sensor network (WSN) is to sense a physical or network environment and detect events that are occurring in the field of interest or in the monitored network [39, 40]. One of the most common application scenarios in WSNs is that of object detection and event recognition [40, 47, 66-70]. In such applications, many sensor nodes are deployed within a large-scale area to monitor the intrusion or diffusion of specific objects or material of interest such as enemy vehicles, wild fires, bio-chemical materials, and so on [40, 47]. Moreover, the sensor nodes can be deployed within the network to monitor and detect network congestions or network security attacks. According to Chandy [71], an event in general may be seen as "changes in the real state". More specifically, Johansson [72] define an event as "Changes that take place in one or more elements within a large group of these elements". In WSNs, an event is described as observable changes in the readings of the sensors [73].

More precisely, it is an event that comprises a collection of sensor readings which describes a specific or an irregular activity [74]. In general, we define an event in WSNs as a change in the state that describes a specific state of predefined phenomena in the field of interest. Examples of event detection in WSNs include the detection of fires in forests, a border intruder, network congestion or a network security attack.

The detection of events or materials of interest can be done by processing and analysing sensory information obtained by sensor nodes. To our knowledge, and according to the literature, pattern recognition is a commonly utilised technique for event detection in WSNs, especially when dealing with complex events [23, 25, 41-50]. Watanabe [75] defines a pattern as the opposite of disorder. Catania et al. [76] define a pattern as “a compact and rich in semantics representation of raw data”. In this research, we define a pattern as a set of raw sensory data that describes the main characteristics or attributes that represent an event. In other words, a pattern can be seen as the signature of an event. This research focuses on the capability to detect (i.e. classify and recognise) transformed patterns of an event.

Pattern recognition is strongly influenced by the limited resources offered by WSNs, including limited energy and limited computational, communicational, and memory resources. In addition to limited resources, WSNs pose other challenges for event detection. These challenges are related to the nature of the environments in which WSNs are usually deployed. For example, WSNs are usually deployed in hostile environments, making sensors susceptible to physical damage and intentional tampering. Additionally, sources of electricity are not usually available for running sensor nodes, requiring these sensors to be operated using batteries with a limited capacity. Also, WSNs are usually required to communicate in an ad hoc manner using low frequency radio signals due to the absence of wires in deployed environments. Furthermore, WSNs are generally deployed as large numbers of sensors in order to monitor an area of interest, which therefore requires the utilisation of low cost sensor nodes. This will limit the types of instruments and resources that can be supplied to these sensor nodes [53]. Therefore, pattern recognition in WSNs is a matter of a trade-off between detection accuracy, the use of limited available resources, and dealing with existing challenges [77].

Other challenges for pattern recognition in WSNs are application-related. WSNs support a variety of real-time applications including battlefield monitoring, environmental monitoring, emergency relief, microsurgery, human activity monitoring, and to a greater extent various disciplines such as health, environment, education, surveillance, and others. Such applications have unique requirements in order to be beneficial. These requirements are driven by the fact that real-time applications should result in decision-making at a certain point in time. Fast reporting to the WSN sink or to a monitoring server is one of these requirements. This necessitates an efficient communication methodology and a global decision-making mechanism. Reliable and accurate detection is another requirement for critical mission applications in order to reduce false alarms. Furthermore, these applications require the network to be fault-tolerant. This means that a WSN should be capable of dealing with noisy patterns and faulty nodes [78].

Luo et al. [79] state that detection accuracy versus the management of energy resources is the main challenge in providing proper event and pattern detection capabilities for WSN applications. Existing event detection and pattern recognition schemes use neural networks, support Vector Machines (SVM), Fuzzy Inference Systems (FIS), and other detection techniques. These techniques are generally tailored to provide detection capabilities for specific applications or problematic scenarios. These techniques may fulfil some of the requirements (e.g. detection accuracy) while failing at others (e.g. lightweight detection). Therefore, this thesis seeks to develop a pattern recognition scheme that leads to pattern transformation detection in WSNs which fulfil real-time application requirements by balancing detection accuracy and WSN resource constraints.

The objectives of this chapter are as follows:

1. To briefly introduce WSNs, including their network topologies, applications, and network architecture.
2. To present and analyse existing pattern recognition schemes for WSNs.
3. To present and analyse the requirements of pattern recognition in WSNs.
4. To compare existing schemes based on the requirements of pattern recognition in WSNs and discuss any issue related to these schemes.

5. To discuss the possible techniques that can be applied in order to overcome the limitations of the existing schemes.

The remainder of this chapter is organised as follows: Section 2.2 contains a brief introduction to WSNs, their network topologies, applications, and network architecture. Section 2.3 presents and analyses existing pattern recognition schemes for WSNs. Section 2.4 presents and analyses the requirements of pattern recognition in WSNs. Section 2.5 compares existing schemes based on the requirements of pattern recognition in WSNs and discusses the issues associated with these schemes. Section 2.6 discusses the possible techniques that can be applied to overcome the limitations of existing schemes. Finally, section 2.7 provides an overview of the chapter.

2.2 Wireless Sensor Networks (WSNs)

Wireless Sensor Networks (WSNs) are a specific type of ad hoc network. A WSN consists of a number of smart sensor nodes that sense physical activities such as motion, heat, speed, and many other environmental parameters. WSNs provide solutions for multiple applications such as climate sensing, factory monitoring, traffic monitoring, pollution measuring, and human activity monitoring. A WSN can scale to thousands of densely deployed sensor nodes in order to perform its tasks. Dense WSN networks are useful because sensor nodes are generally susceptible to failures. A sensor node is small, its size generally varying from the size of a grain to the size of a hand. Sensor nodes are limited in their energy, memory, and computational resources, thereby limiting the effectiveness of WSNs. Therefore, traditional applications and protocols that are applied to networks in general are usually not applicable to WSNs [11, 15, 19, 80, 81].

As discussed in Chapter 1, a sensor node consists of five main components: a memory, a controller, a communications device, sensors/ actuators, and a power supply. The controller is the main element of a wireless sensor node. It collects data from the sensor itself, processes this data, determines where and when to send the data, obtains data from other sensors, and determines the actuator's behaviour. It is also responsible for executing a variety of programs (e.g. communication protocols) on the sensor nodes. Additionally, it controls all of the behaviours of the sensor node's elements. To sum up, the controller is the Central Processing Unit (CPU) of the sensor node [15, 82, 83]. The memory component consists of two main elements: Random Access Memory (RAM) and Read-Only Memory (ROM). RAM is responsible for storing packets received from other sensor nodes, intermediate sensor readings, and so on. On the other hand, ROM is responsible for saving programs and applications. Moreover, a flash memory may be utilised as a part of the memory components to support memory tasks and to act as a back-up for RAM when it is insufficient or when it goes down [15, 82, 83]. The communication component is utilised to exchange data between a sensor node and other nodes and devices in WSN. It consists of two elements: a wireless transmitter and a wireless receiver. Both of these elements interact with the transmission medium in order to achieve the communication. For example, coding, data rates, and modulation are some of the communication component's tasks [15, 82, 83]. Sensors and actuators perform the main jobs of a sensor node. Sensors are responsible for detecting and measuring the physical parameters of the observed area or the direction of an observed object. Moreover, actuators are responsible for performing actions such as turning off a switch based on the observed and analysed data [15, 82, 83]. Power supply is the energy-conserving unit of a sensor node. The required energy for sensor node tasks may be obtained from batteries. Moreover, the energy can be obtained from the environment (e.g. solar cells) through the recharging process [15, 82, 83]. Moreover, some sensors may include other components to enhance performance or to perform specific tasks. Solar panels and GPS devices are examples of these additional components. Due to the limited size of a sensor, the capabilities and resources of its components are also limited [19, 84].

Table 2.1 shows several types of sensor hardware nodes with different capabilities. Each of these types is utilised for different purposes. Additionally, sensor nodes are usually wirelessly connected and densely deployed to construct large-scale WSNs in order to sense and monitor physical environments. The connectivity between WSN sensor nodes can be done according to different network topologies as discussed in the following sub-section.

Table 2.1: Some of the current WSN platforms [85].

Node, Year	CPU	Memory	Radio	Notes
Special-purpose Sensor Nodes				
Spec, 2003	4-8Mhz	3KB RAM	50-100Kbps	Used for low-power operation.
Generic Sensor Nodes				
Mica-2, 2001	ATMEGA 128	4KB RAM 128KB Flash	76Kbps	Primary TinyOS development platform.
Telos, 2004	Motorola HCS08	4KB RAM	250Kbps	Supports IEEE 802.15.4 standard. Allows higher-layer Zigbee standard.
Mica-Z, 2004	ATMEGA 128	4KB RAM 128KB Flash	250Kbps	Supports IEEE 802.15.4 standard. Allows higher-layer Zigbee standard.
High-bandwidth Sensor Nodes				
Imote 1.0, 2003	ARM 7TDMI 12-48Mhz	64KB SRAM 512KB Flash	Bluetooth 1.1	Multihop utilising scatternets. Easy connections to phones and PDAs.
Gateway Nodes				
PC104 nodes	X86 processor	32KB Flash 64KB SRAM	Serial connection to sensor network	Embedded Linux or Windows support.

2.2.1 WSN network topologies

Three types of network topologies are commonly in use in WSNs [86, 87]: star, peer-to peer, and two-tier (also known as cluster tree), see Figure 2.1. In the star topology, a central device is in control of the network’s communications. This device can be a sensor node, an access point or any communication unit that is capable of linking sensor nodes. In general, a central device has a larger memory capacity, higher processing capabilities and more energy resources than other sensor nodes in the network. The main issue that restricts the use of such a topology is the possibility of single-point failures. The peer-to-peer network topology allows each sensor node in the network to directly communicate with its neighbours within its communication range. If a sensor node needs to communicate with more distant nodes, routing protocols may be utilised, allowing some sensor nodes in the network to act as routers. The use of such a topology allows the network to be more fault-tolerant and flexible. However, managing a WSN that has a peer-to-peer network topology can be challenging. The two-tier network topology is a combination of star and peer-to-peer topologies. The network in this topology is divided into groups. Each group is connected utilising a star topology. The central devices communicate using a peer-to-peer topology.

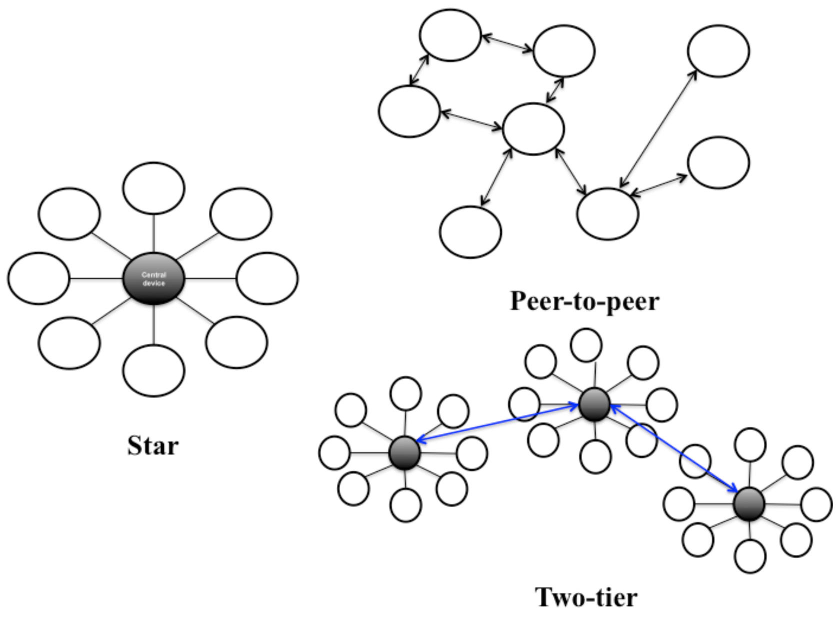

Figure 2.1: WSN network topologies.

2.2.2 WSN applications

Chen and Varshney [78] classify WSN applications based on their requirements as follows:

1. **Event-driven:** Event-driven applications in WSNs are most likely to be real-time applications where the network is analysing sensory data to detect a specific or a set of events. The events in these applications are usually infrequent, which means that sensors can remain in sleep mode for most of the time. However, these events are expected to be mission-critical and require quick recognition and reporting. In this application category, the detection requirements are fast reporting, distributed recognition, reliabity and accurate detection, noisy patterns detection, and able to provide location information.
2. **Query-driven (sink-initiated):** In this type of application, data is gathered based on sink commands. Applications of this sort are usually interactive and mission-critical. The sink can send query commands to obtain sensory data in order to take an action. The requirements of these applications are fast reporting, fast distribution of sensory data, reliable reporting, and noisy pattern detection. They may also require location information reporting capabilities.
3. **Continuous and periodic reporting**: This is where WSN nodes are continuously reporting information to the sink according to a specific timeframe. These applications can be either real-time or asynchronous. For real-time applications, the emphasis is on fast information reporting

while noisy patterns and lost information are tolerated (to a certain extent).

4. **Hybrid models**: This is where an application may combine two or more of the applications presented above. An example of these applications is tracking-based applications where the network is interested in detecting intruders in a specific location. The requirements of these types of applications depend on the number of application types being used, which might include all of the above requirements.

Additionally, Iyer et al. [88] classify the different applications of WSNs as data gathering, object tracking, and event detection. In data gathering, each sensor node sends its readings to a sink or a base station either periodically or in accordance with the sink request. Therefore, the main goal of data gathering is solely to obtain information about the field of interest without *in-network* decision-making. Intuitively, this means that the sensor nodes of a WSN do not collaborate to perform computations and/or memorisations on gathered data in this type of application. Object tracking focuses on monitoring the movement and the state of one or more objects that enter the field of interest and can use data-gathering applications to achieve this goal. As such, it is a challenging task that requires dealing with computational complexity within the resource constraints of the WSN while often also requiring real-time detection and solutions. Moreover, the WSN datasets in this type of applications are often being continuously generated, meaning that a dataset must be analysed just-in-time before the next dataset is collected. Event detection can be considered to be a higher level abstraction of the data that represents a unique occurrence or a feature in the WSN dataset.

2.2.3 WSN network architecture

WSNs are a network comprised of many sensor nodes. A set of protocols is required to perform networking functions such as processing information and communicating with other sensor nodes. In networking, such protocols are segmented into layers that distinguish between the roles for each protocol. The standard networking segmentation is the Open System Interconnection (OSI) network reference model, which was developed to standardise the protocols of networking by the ISO organisation [89]. Therefore, the term 'ISO reference model' is also utilised to represent the standard model. In this model, the network protocols are divided into seven layers: Application, Presentation, Session, Transport, Network, Datalink and Physical. Each layer is assigned with specific roles in the networking process.

Additionally, WSNs carry unique features and limitations. Therefore, there are several network architecture models available for WSNs in the literature. A common model for WSNs contains five layers [90]: Application, Transport, Network, Data-link and Physical. In the following sub-sections, the tasks for each layer are briefly presented and discussed.

2.2.3.1 Application layer

The Application layer provides high level protocols and applications that are commonly available in a WSN base station rather than in the rest of the network’s sensor nodes [91]. The absence of this layer from sensors is due to the high level of processing it requires compared to the limited computational capabilities offered by sensor nodes. A WSN base station is expected to include much higher computational capabilities that enable the hosting of such high level processing requirements and achieve a comprehensive outcome for the whole network.

2.2.3.2 Transport layer

Transport layer protocols offer reliable communication services between two ends in the network. In this layer, the protocols ensure the highest possible level of Quality of Service (QoS). This can be achieved by offering services such as message segmentation, flow control, congestion control, and message retransmission for lost packets [92]. The techniques for implementing and designing transport protocols in WSNs affect the QoS and throughput of a WSN network and should vary from one application to another as different applications have different QoS tolerance levels.

2.2.3.3 Network layer

The Network layer deals with routing packets within the network. This includes developing mechanisms for building routing tables to allow sensor nodes to direct their messages and redirect incoming messages from other sensor nodes to the proper hop. Unlike traditional networks, WSNs do not provide IP addresses and, hence, do not provide IP routing capabilities. The design of network layer protocols in WSNs should take into consideration network scaling, routing fairness, and security issues, along with the existence of resource constraints. Furthermore, network protocols in such network environments should address the problem of sensor nodes' limited lifetime by involving fault-tolerant procedures [90].

2.2.3.4 Data-link layer or MAC protocols

The Data-link layer or Medium Access Control (MAC) protocols are utilised as an underlying layer of the network layer protocols. MAC protocols control and manage the access of the shared wireless medium between WSN nodes [93]. In WSNs, the communication environment is noisy and sensor nodes' resources are limited. Therefore, MAC protocols designed for WSNs take power consumption into account and attempt to reduce collisions between communicational nodes to avoid retransmission of packets [19]. Nodes in WSNs usually alternate between active mode and low power consumption sleep mode to conserve energy resources. Hence, WSN MAC protocols should consider ways of communicating with sensor nodes that are in sleep mode in order to avoid occupying communicational channels and increase the throughput of the network [94]. Traditional MAC protocols for WSNs allow each node to have only one communication at a time. However, recent WSN

MAC protocol research trends have moved towards multichannel MAC capabilities that allow a WSN node to have multiple communications at the same time to support multi-task operations [94].

2.2.3.5 Physical layer

The lowest layer in the model is the WSN Physical layer. This layer includes all physical components of sensors such as chips, transceivers, and processors [91].

Tiny sensor design results in limited resources for WSNs as limited memory, computational, and communicational resources are available per sensor. Therefore, these constraints have to be addressed when designing WSN protocols in any layer in the model. Sensor nodes mostly work in a collaborative manner in order to tackle these limitations and create larger interactive resources to deal with detection problems.

2.3 Pattern Recognition in WSNs

Wittenburg et al. [95] and Duda et al. [96] divided the pattern recognition process into the following three main steps:

1. **Sampling:** Gathering sensing data and describing the sensed object.
2. **Feature Extraction:** Obtaining the main features of the sensed object. The main goal is to reduce data required to describe the object's pattern, which reduces the computations and resources needed to recognise it.

3. **Classification:** Classifying objects according to categories by utilising extracted features.

Pattern recognition schemes in WSNs have been classified from different perspectives. For instance, Predd et al. [97] classified distributed learning and recognition in WSNs as kernel, supervised, unsupervised, and distributed learning algorithms. Their main focus was to classify them in terms of fusion-centric and ad hoc network topologies. In a supervised pattern recognition algorithm, the system is first trained with the training pattern that has been provided by a teacher entity. After that, the system will be fed with the incoming patterns. On the other hand, in an unsupervised pattern recognition algorithm, the system starts directly to learn and deal with incoming patterns. Moreover, Nakamura et al. [98] present different classification perspectives for data fusion, one such application being pattern recognition. They classify pattern recognition into the following areas: Bayesian, Dempser-Shapher, Fuzzy Logic, Neural Networks, and Semantic.

In this section, we will discuss different pattern recognition schemes that have been utilised for WSNs in the literature. These schemes are classified as threshold-based, statistical, K-nearest neighbour, conditional, support vector machines, neural networks, and graph neuron schemes. Figure 2.2 summarises these schemes.

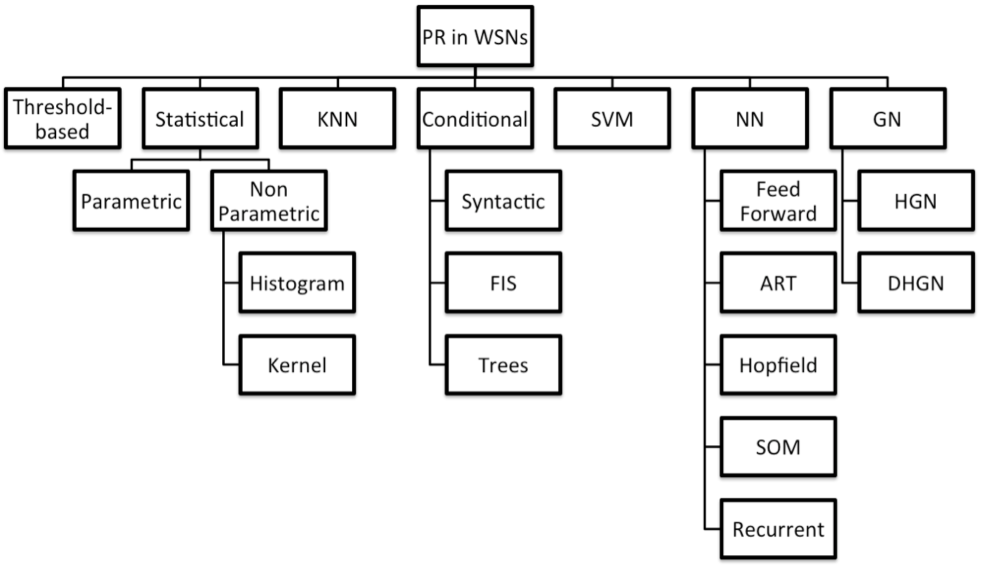

Figure 2.2: Classification of existing pattern recognition schemes for WSNs.

2.3.1 Threshold-based approaches

The threshold-based pattern recognition technique is one of the simplest, most well-known, and widely utilised pattern recognition techniques in WSNs. In this approach, one or more threshold values is assigned to each sensor node, so a sensor node will declare the detection of the event of interest when its reading value hits the threshold value. For example, Kim et al. [99] proposed a fence surveillance scheme that can detect intruders based on thresholds obtained from the average signal measurements of each sensor. In this scheme, a sensor node will send a *DETECT* signal to the base station when its reading value exceeds its threshold. Another example in the literature is the work of Jabbar et al. [100] where the authors proposed a threshold-based load balancing technique for routing problems in WSNs.

Threshold-based pattern recognition techniques are considered to be light-weight and simple. Moreover, such techniques do not require complex network communication relationships between sensor nodes. However, three main issues are associated with the implementation of threshold-based techniques for pattern recognition in WSNs. First of all, threshold-based pattern recognition cannot describe and address complex detection problems; thus, it will produce many false alarms when used to address such problems. Secondly, these techniques are limited in when it comes to dealing with noisy patterns [101]. The WSN patterns that have been discussed in the literature are commonly noisy [96]. Therefore, this limitation makes these techniques inappropriate for pattern recognition in WSNs. Thirdly, in some applications, the determining of threshold values can be very difficult and challenging [102].

Threshold-based techniques provide pattern recognition with lightweight capabilities in terms of computations and communications. However, such techniques offer problem-specific solutions and their accuracy is limited when it comes to complex problems and noisy environments.

2.3.2 Statistical approaches

Statistical pattern recognition approaches are predicated on the probability of pattern occurrence and the *Bayesian decision rule*. These approaches are based on the following assumptions: the occurrence probabilities values are known and recognition decision is achievable in terms of probability [96, 103, 104]. In these approaches, the patterns are classified in terms of *state of nature,* where every pattern can be assigned to one class (state). The likelihood of the pattern being a certain class is known as the priority probability [103]. For instance, if we want to recognise a fruit as either an apple or an orange, then the states are "apple" and "orange". In this scenario, the priority probability of an input pattern is decided in accordance with historical knowledge or data, which links the input pattern's characteristics and features (such as input shape) to one of the states (i.e., if the input pattern shape is a circle, then P(orange)=0.74 and P(apple)=0.25). The final decision can be made based on the highest prior probability, which can be expressed as follows:

$$if\ P(orange) > P(apple), then\ the\ result\ is\ orange \qquad (2.1)$$

In order to avoid making the same decision whenever the same situation is encountered, such statistical models use *class-conditional-probability* to minimise the classification error rate (i.e. the fruit could be an apple even if its shape is a circle) and can be described as an extra feature that supports the decision-making process. For example, the shape and colour can be utilised to distinguish between an apple and an orange. In order to make a decision about an incoming pattern, statistical approaches use the *Bayesian decision rule*. Let *x* be the statistical variable, *i* is the class number, and C_i is class number *i*. The Bayesian decision rule can be described as follows [96]:

$$P(C_i|x) = \frac{p(x|C_i)p(C_i)}{p(x)} \qquad (2.2)$$

where $P(C_i|x)$ is the classification of the incoming pattern given *x* (posterior), $p(x|C_i)$ is the conditional probability density of class C_i given *x*, $p(C_i)$ is the prior probability of class C_i, and $p(x)$ is the evidence probability of x for j number of entered (stored) classes that can be calculated according to the following Equation [96]:

$$P(x) = \sum_{i=1}^{j} p(x|C_i)p(C_i) \tag{2.3}$$

In practical classification problems, statistical approaches use more than one variable (i.e. feature). If the statistical relationships and dependencies between variables are known, *Bayesian belief networks* are used for solving classification problems. Figure 2.3 shows a simple example of a Bayesian belief network. In this example, four variables A, B, C, and D and their dependencies are available. The final decision is made based on the dependencies' statistics.

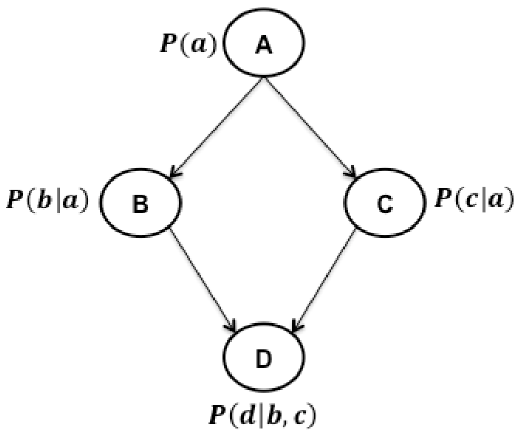

Figure 2.3: An example of Bayesian belief network.

On the other hand, *Naive Bayes* statistical classification is utilised when the statistical relationships and conditional dependencies between variables are unknown. In this classification method, the assumption that variables (e.g. a, b, and c) are conditionally independent is taken into account and can be represented as follows [96]:

$$P(a, b|c) = P(a|c)P(b|c) \tag{2.4}$$

Non-parametric statistical classification methods assume that statistical density distribution is not available. Therefore, such techniques obtain probability densities from a set of training samples. This assumption is based on the fact that in most classification problems, probability density of classes is unknown [51, 96]. Such techniques utilise distance thresholds based on probability observations to decide the incoming pattern’s class. According to Zhang et al. [51], such techniques are commonly utilised in WSNs as outlier detection methods and can be classified as histogram and kernel approaches. Histogram approaches count the probability of the occurrence of data classes and instances and compare incoming patterns with the calculated probabilities. Kernel approaches create probability distribution functions and use thresholds to determine an instance class.

Elnahraway and Nath [105] present a Naive Bayesian distributed method to detect faulty sensors. Their proposed technique provides an outlier detection method using spatio-temporal classification where each node evaluates its readings probability according to one of many predefined classes. Wu et al. [106] present a Naive Bayesian-based technique for applications in the medical domain. In their work, they utilised WSNs to monitor patients and detect abnormal gait patterns. Mittal et al. [107] present Bayesian belief network approaches to weather status detection. Their technique acquires weather attributes such as humidity and temperature values from WSNs and then applies a two-step method for classification. The first step constructs the relationship between obtained attributes and the second step performs the recognition based on the constructed relationships. Sun and Edward [108] present a non-parametric distributed statistical approach to detect specific events (e.g. loud cheering) in sports stadiums. Each sensor in a WSN deployed in a stadium decides the occurrence of an event locally, based on noise levels, and then sends the result to a cluster head. The cluster head then detects the event based on the optimal median from the information collected by all participating sensors.

The use of statistical pattern recognition approaches in WSNs poses a number of challenges. In most classification problems, such as the ones in WSNs, the prior knowledge of probability distribution is rare [51, 96, 103]. Thus, implementing most parametric statistical approaches becomes unfeasible due to the lack of such knowledge. Non-parametric statistical approaches are more feasible since such approaches do not require prior information about probability distribution. However, the accuracy of these techniques is highly dependent on the number of available training samples as they construct probability distributions based on available samples [96, 103]. In WSNs, the number of samples of patterns and events is limited due to the randomness feature of information gathering in WSNs [42, 51]. That is, the occurrence of an event may be captured on rare occasions. Additionally, obtaining enough information about an existing pattern in order to construct probability distributions is limited due to WSNs' communicational and computational limitations [109]. These limitations make the use of non-parametric approaches in WSNs challenging. Moreover, some non-parametric approaches such as histogram techniques require high communicational overheads to obtain histogram information [51]. These requirements are at odds with the limited communicational capabilities of WSNs. Other non-parametric approaches such as kernel techniques require defining thresholds in order to estimate probability densities. However, determining such thresholds may be challenging [51].

2.3.3 K-nearest neighbour (KNN)

The K-nearest neighbour (KNN) approach is one of the simplest non-parametric classification techniques, which assumes that the density distributions of pattern samples are unknown [96, 110]. The KNN approach computes distance or similarity measures between two data instances and makes a decision based on the result of the comparison. The distance between two samples is calculated according to a predefined function. One of the most popular distance functions in KNN is the Euclidian distance, which can be represented as follows [111]:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (a_i(x) - a_i(y))^2} \tag{2.5}$$

where $d(x, y)$ is the distance between instances x and y, n is the number of attributes, and a_i is the i'th attribute of the instance.

The decision in a KNN algorithm is based on the number of nearest neighbours *(k)*. After calculating the distances to each neighbour, a KNN scheme will vote among k neighbouring instances to classify an incoming pattern according to labelled classes. Such voting can be described as follows [110]:

$$C(x) = majority[C(N_1), \dots, C(N_k)\ ] \tag{2.6}$$

where $C(x)$ is the class label of instance *x*, $C(N_i)$ is the class label of the i'th nearest neighbour to instance *x*, *k* is the number of nearest neighbours assigned to KNN, $majority$ and means the highest number of instances that have the same class label. Figure 2.4 shows a simple classification example utilising KNN. In this example, seven instances are classified into two classes, C1 (Red) and C2 (Green). The task is to classify instance *x* as one of these classes using KNN. The KNN uses Euclidian distance as the distance function. In Figure 2.4, (a) shows the use of k=*1*, (b) k=*3*, and (c) k=*5*. The classification results show that when utilising k=*1*, the classification result for *x* is C1 (as the nearest neighbour is of class C1); when utilising k=*3*, the classification result for *x* is C2 as two instances out of three (majority) are of class C2, and when utilising k=*5*, the classification result for *x* is C1 as three instances out of five are of class C1.

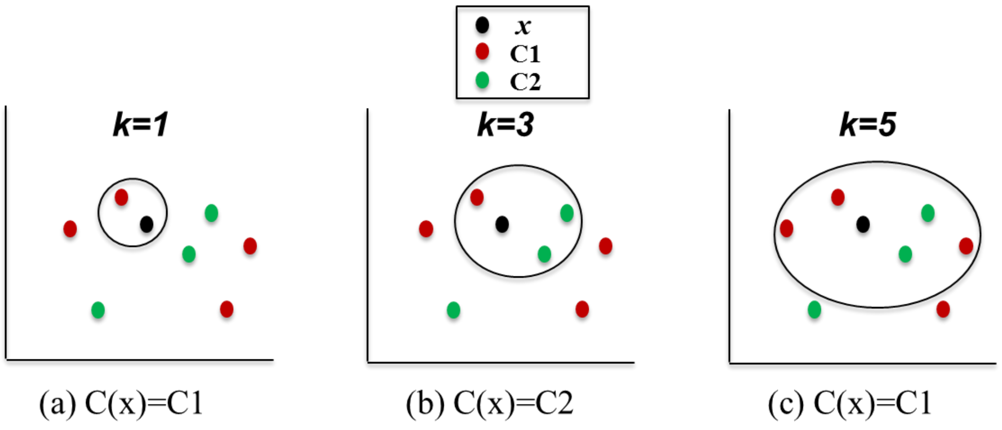

Figure 2.4: An example of KNN classification.

An example of KNN being utilised in WSNs is the work of Li et al. [112] where they utilised KNN for distinguishing between tracked and wheeled vehicles in distributed sensor networks. In [113], KNN was one of the classifiers that was used to classify the type of moving vehicles in distributed sensor networks. Moreover, KNN is commonly utilised as an outlier pattern recognition algorithm in WSNs [51]. This means that the normal activities of a network are modelled as pattern instances and stored in the network. A data instance is considered to be an outlier if its measure is far from the neighbouring instances that represent normal activities [51]. The main assumption on which such techniques are based is that an outlier occurrence takes place far from its neighbours [114]. An example of the nearest neighbour outlier detection technique for WSNs is the work of Zhang et al. [115]. In their work, they propose a tree aggregation structure where each node sends some information to its parent. Then, the sink decides the global *n* outliers for the network and sends back this information to the network for verification. The process continues until all the network nodes agree on the outliers.

There are several issues related to utilising KNN as a classifier in general and as a pattern recognition technique in WSNs. First of all, KNN depends on the utilisation of a distance function. In some classification problems, the standard Euclidian distance function does not lead to accurate classification [111]. Therefore, more complicated functions might be required. Secondly, the accuracy and complexity of KNN is dependent on the choice of the number of neighbours (*k*). Such dependency requires *k* to be tuned in such a way that it balances complexity and accuracy. Additionally, such dependency could lead to tuning *k* according to the probability distribution of data, making the process data-dependent [110]. Thirdly, the decision process in KNN is based on a majority function. In the case of tie voting, complex algorithms must be utilised to break the tie decision. Such complex algorithms could lead to higher KNN complexity [110]. Fourthly, KNN requires large memory resources to memorise distances between data instances, especially when used in WSNs [116]. Fifthly, KNN requires high computational resources to compute distances, which makes this technique lack scalability when implemented in WSNs [51].

2.3.4 Conditional and structural methods

There are a set of conditional and structural classification methods in the literature that have been utilised for classification problems in WSNs. These methods attempt to create a relationship between pattern elements and are commonly utilised when non-metric data is available [96, 103]. These techniques can be categorised as syntactical, fuzzy logic, and decision tree methods.

2.3.4.1 Syntactical approaches

The syntactic approach creates structural rules in order to describe the relationship between sub-patterns and patterns. It adopts the language theory in which letters form words and words form sentences based on grammatical rules. This approach analyses sub-pattern relationships and primitive elements in order to provide pattern recognition. Identifying primitives that describe patterns and describing the relationships (rules) between sub-patterns are the syntactic approach's main constraint [117]. This analysis can be performed by utilising different schemes such as tree grammars, transformations, neural networks, and more [118].

An example of the use of the syntactical method in WSNs is the work of Latha et al. [119] where they utilise the syntactical method in semantic tracking for wildlife preservation using WSNs. The syntactical method is utilised as a processing stage that checks a node's detection with other nodes in the same cluster. Syntactic pattern recognition provides complex pattern recognition, which is usually utilised if there is no appropriate statistical scheme available. On other hand, its recognition procedure and grammars are complex, especially in the presence of noise [120]. Another issue ralated to this technique is the large amount of training data required for training and creating relationships between sub-patterns [103].

2.3.4.2 Fuzzy logic

Fuzzy Inference System (FIS) is typically the name of a system that employs a fuzzy logic technique. FIS maps inputs to outputs with the help of fuzzy sets, rule base, and membership functions in order to arrive at conclusions [121]. This system is created based on three main elements: membership functions, rule base, and reasoning mechanism [122]. In this system, inputs should be first classified to fuzzy sets according to membership functions. For example, temperature readings could be classified into low, medium, and high. After that, the conclusion, which is based on the classified inputs, is provided by using the rule base. The rule base consists of IF-THEN rules, which receive more than one value in order to make a conclusion. An example of a fuzzy base rule is:

$$If\ x\ is\ A\ and\ y\ is\ B, then\ z = f(x, y) \tag{2.7}$$

where x and y are the classified inputs and z is the FIS output. Obtaining such rules for a FIS may need pre-knowledge regarding the relationships between variables [121]. Implementing these rules on variables is called ‘approximate reasoning mechanism’ or ‘fuzzy reasoning of a FIS’.

In [121], authors proposed a WSN activity recognition system based on FIS to support workers involved in car assembling and training. Zarei et al. [123] proposed a FIS congestion control algorithm for WSNs in order to recognise malicious node activities. Moreover, Feng et al. [124] utilised FIS to measure the distance between WSN nodes to achieve better localisation. In FIS, the main challenge is to obtain the fuzzy sets and the IF-THEN rules. According to Nakamura et al. [98], FIS is often used to control a neural network's learning rates rather than being utilised for recognition, which will lead to the same issues with pattern recognition in WSNs that will be presented in the neural networks (NNs) sub-section. Moreover, deducing rules in most WSN applications might be challenging.

2.3.4.3 Decision tree approaches

Decision tree methods are constructed from a set of nodes that are logically arranged in a tree-like shape. Each node makes a decision about the incoming pattern feature and, based on that decision, the process questions the next feature in lower level nodes. An example of a simple decision tree is shown in Figure 2.5. In this example, five fruit classes, watermelon, grape, grapefruit, lemon, and banana are to be recognised by using three features: their colour, size, and shape. The tree in Figure 2.5 has four levels, including the root of the tree. The number of levels determines the depth of a decision tree. In decision trees, the same question, Size? in this example, can appears in different places in the tree [96]. One of the most common decision tree structures used for classification purposes is the binary decision tree. In this structure, each node makes one out of two decisions and inspects a single feature at a node to minimise recognition time and complexity [103].

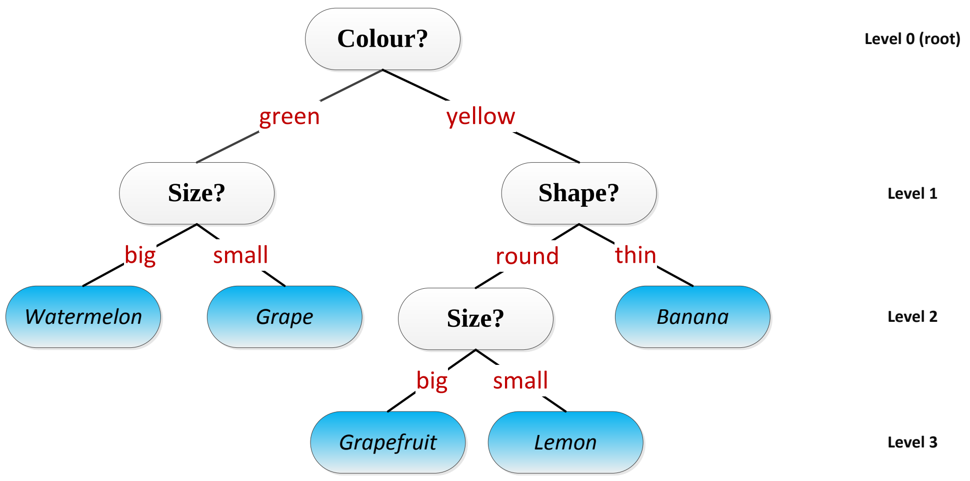

Figure 2.5: An example of a simple decision tree for fruit classification.

An example of a decision tree technique implementation in WSNs is the work of Bahrepour et al. [125]. In their work, they propose a WSN event detection mechanism based on a decision tree technique. The proposed scheme distributes features into several trees where each tree makes one decision. Finally, a voting process takes place between the results obtained by the trees to determine the detected event. Decision trees are expected to involve limited computations and communications. However, decision tree techniques are affected by noisy patterns which increase the scheme's complexity, especially for large scale trees [126]. As a result, decision trees are more useful as decision-making processes on top of another pattern recognition technique.

2.3.5 Support vector machines (SVM)

In [127], the authors (Cortes and Vapnik) propose the Support Vector Machine (SVM) as a classification and learning machine. In SVM network, input vectors are (non-linearly) mapped to a very high dimension feature space. In this feature space, a linear decision with special properties is built, which guarantees a high generalisation capability for the network. Then, the hyperplane, which performs class separation, is built in this high space. The linear decision function with a maximum margin between the vectors of two classes is called an optimal hyperplane. Figure 2.6 shows an instance of a separable issue in a two-dimensional space for two different classes. The vectors (points), located on a class margin line, are called support vectors and the distance between the margin lines of the two classes is called an optimal margin. In order to perform classification, a test vector (input pattern) should be compared with every support vector in order to identify the class to which this test vector belongs [112, 113, 127]. In practical implementation, SVMs utilise a classification function which can be calculated based on the kernel representation. Therefore, the choice of the kernel will have impacts on the classification process of the network [112].

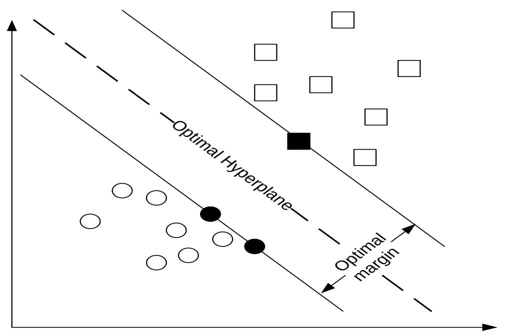

Figure 2.6: An example of a separable issue in a two-dimensional space for two different classes utilising SVM. Black marked vectors (training samples) represent the support vector for each class [127].

One of the examples of the SVMs implementation in WSNs is the work of Wang et al. [128]. In their work, they propose a target classification method based on SVM that overcomes the false alarm rate in samples in WSNs. The authors propose the use of energy consumption metrics to construct an SVM-based classifier for WSNs in two paradigms, namely, centralised and distributed. The centralised paradigm represents the traditional SVM classifier. Conversely, the distributed method attempts to utilise samples (i.e. nodes) close to hyperplanes in order to reduce classification overhead costs. The main aim of the distributed classifier is to allow a set of sensors to communicate with a set of cluster heads to construct an SVM classifier. Tran and Nguyen [129] utilise SVMs with a radial basis function network (RBFN) kernel for error tolerance localisation in WSNs. The authors propose the use of connectivity information, such as number of hops, as metrics to classify WSNs and estimate sensors locations. Another example is the work of Abu Sajana et al. [130] who utilise SVM to detect physical intrusion attacks on WSNs that contain passive infrared (PIR) sensors. The aim is to reduce false alarms caused by detecting windblown vegetation. They propose the use of Haar transformation and frequency binning along with SVMs to resolve the classification problem.

The main challenge of using SVMs for classification problems in WSNs comes from the fact that a technique who implements an SVM classifier requires centralised processing capabilities in order to create hyperplanes and classify incoming patterns based on computed information. Another challenge is related to the number of training samples required, which can lead to an extensive training period [112, 128]. In order to create separating hyperplanes between classes and correctly classify instances, SVMs require a large number of training datasets. Such requirements may be challenging in applications that expect patterns to occur randomly. Another challenging issue when utilising SVMs is their dependency on kernel functions. The use of kernel functions will tie SVM techniques to the issues related to the kernel itself. For instance, an SVM technique that implements a neural network (NN) method as its kernel function will suffer from tightly coupled connectivity between nodes and the iterative processing associated with NNs. Therefore, the choice of the type of kernel function plays a very important role in determining the suitability of an SVM technique for use in WSNs.

2.3.6 Neural networks (NNs)

Neural Networks (NNs), also known as Artificial Neural Networks (ANNs), are computational techniques that provide parallelism in pattern learning and recognition [98]. Associative Memory (AM) is one of the neural network techniques capable of memorising and retrieving patterns in a distributed manner. The literature shows that AM has been utilised to offer pattern recognition solutions based on its capability of recalling memorised templates [131]. Moreover, AM networks are capable of dealing with noisy patterns and are considered to be a robust solution [132]. In general, AM depends on utilising tiny memory chunks available in computational units to accomplish the distributed memory management. NNs schemes can be classified as feed-forward neural networks, adaptive resonance theory, hopfield networks, self-organising maps, and recurrent neural networks.

2.3.6.1 Feed-forward NNs

The feed-forward network is a layered network consisting of neurons (processing nodes), which implements a supervised associative memory approach. It describes associations between the input layer and the output layer [133]. Each neuron in a layer is connected to each neuron in the layer above by variable weight values [134]. Figure 2.7 shows the general structure of feed-forward networks [133, 134]. Such networks are commonly utilised in pattern recognition applications [96, 135]. In this network, the layers located between the input and output layers are known as the hidden layers. The input layer is not involved in any computation. Thus, all computations take place in the hidden layers. Each hidden layer calculates the inner product of inputs with weights, which is called the network's activation function. The connections between neurons are usually called synapses and the values of these synapses are called synapse weights, which are calculated using a non-linear activation function. In this approach, the activation of a neuron depends on a predefined weight and a bias unit assigned to neurons [96]. Each neuron's activation values determine the signal strength from the input layer to the output layer. To determine each neuron's weight and bias, a learning rule, such as the generalized delta-rule or the back propagation rule, will be applied.

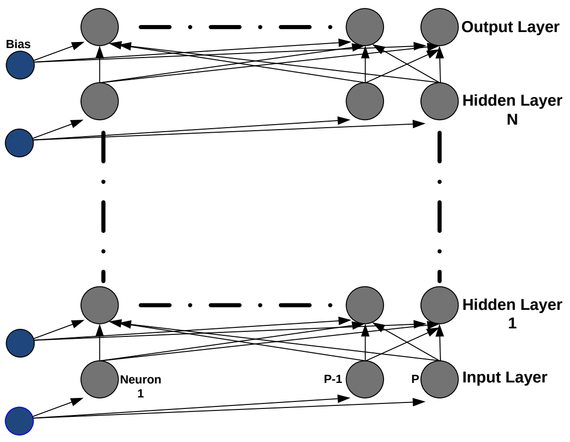

Figure 2.7: The Feed-Forward network structure that has P inputs and N hidden layers [134].

In [136], Awad et al. propose a feed-forward based recognition algorithm for localisation and location estimation in WSNs. The proposed approach utilises the feed-forward network to analyse received signal strength indicator (RSSI) to estimate the distance between two sensor nodes. Rajkamal and Ranjan [137] utilise feed-forward networks to classify exchanged packets between sensor nodes, based on the nature of the incoming data to be able to control the traffic flow in a WSN. In addition, radial basis function networks (RBFNs) are one type of the feed-forward network. RBFN consists of three layers: input, hidden, and output [138]. Ishizuka and Aida [139] utilise RBFN to achieve efficient low-power sensor placement. Tran and Nguyen [129] use RBFN as a kernel function for a support vector machine (SVM) technique in the localisation of WSNs' nodes.

2.3.6.2 Adaptive resonance theory (ART)

The Adaptive Resonance Theory (ART) is a multi-layer unsupervised neural network method, which solves the problem of the limitation on learning scalability of neural networks. This limitation is known as the stability-plasticity dilemma [140]. In an ART network, there are three main layers: input, comparison, and recognition layers (see Figure 2.8).

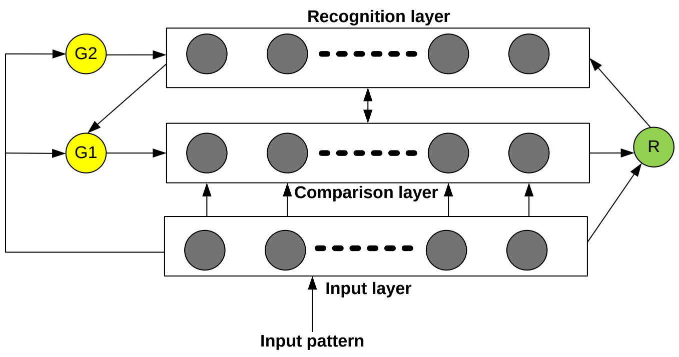

Figure 2.8: ART network architecture [140].

The input pattern is received and stored by the input layer. In such networks, each neuron in the input layer is directly connected to a corresponding neuron in the comparison layer, utilising non-modifiable weights. On the other hand, each neuron in the comparison layer is directly connected to all neurons in the recognition layer, utilising modifiable weights. In ART structure, there is also a feedback connection where each neuron in the recognition layer is linked to all neurons in the comparison layer. Moreover, this architecture utilises gain modules (G1 and G2) and the orienting sub-system (R). These are the signals that play the role of controlling the activating and deactivating neurons in the comparison and recognition layers [141]. In the comparison layer, the neurons are fed with three inputs: input pattern, feedback pattern from the recognition layer, and the gain value G1. In the recognition layer, neurons receive two inputs, from the comparison layer and G2. The recognition process is based on calculating the weights and determining the winning neuron in the recognition layer. The neuron having the highest weight will be activated and compared to the stored patterns to find a match. If no match is found, the neuron will be deactivated and another neuron will be activated and compared. This procedure continues until the network finds a match. Otherwise, the input pattern will be stored [140].

YuanYuan and Parker [142] proposed an ART-based WSN detection system to detect intruders in an unknown environment. Kumar et al. [143, 144] implement ART networks in WSNs to classify patterns in order to achieve clustering aggregation in unknown environments. Kulakov and Davcev [145] utilised ART networks as classifiers to detect unusual WSN's motes behaviour in order to identify intruders. From the above description and examples, ART networks offer scalability in terms of the total number of stored patterns. Such networks are also useful for classifying patterns without having prior information such as statistics [143, 144]. However, there is no guaranteed convergence time, which would decrease the appropriateness of this approach for use in WSNs. Moreover, the high connectivity requirements between layers would be prohibitive in terms of power.

2.3.6.3 Hopfield networks

The Hopfield network structure is based on a single-layer network in which every neuron is fully connected to all other neurons [146] (see Figure 2.9). In the Hopfield network, every connection is measured as a weight, which is assigned during the pattern learning stage. Both connections, which occur between two neurons, must have an equal weight that can be measured by utilising the following equation:

$$W_{jk} = \begin{cases} \sum_{p=1}^{T} x_j^p x_k^p, & j \neq k \\ 0, & j = k \end{cases} \tag{2.8}$$

where W_{jk} is the connection weight, x_j^p and x_k^p are the pattern number p for neurons *j* and *k* respectively, and *T* is the total number of patterns [131]. This equation describes the Hopfield neural network in the discrete representation. In [147], Massini argued that the Hopfield neural network is limited in terms of the total numbers of patterns that can be memorised and detected.

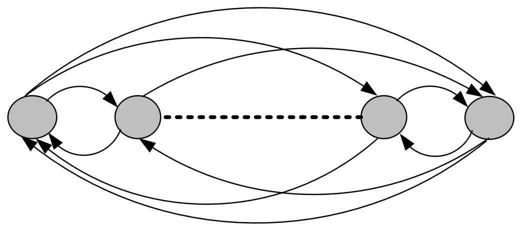

Figure 2.9: The structure of Hopfield network based on a single-layer network [147].

Additionally, Hopfield networks can be classified into two types: Discrete Hopfield Network (DHN) and Continuous Hopfield Network (CHN)

[148]. DHN is a stochastic model, which provides simple implementation and fast processing. However, DHN utilises binary values for the states of neurons, and it therefore produces only approximate results. As a result, DHN cannot offer a precise solution in a pattern recognition application. On the other hand, CHN provides a near-optimal solution by utilising a differential equation approach. This is actually an extra load for CHN, since it needs more time for its simulation. As a result, CHN is not appropriate for a pattern recognition application that needs fast recognition, such as in biometric pattern recognition.

Hopfield neural networks are one of the simplest and most common types of NNs that have been utilised for pattern recognition problems in WSNs. For instance, Chen et al. [149] utilised Hopfield networks in target tracking by matching sensor nodes' measurements with the target tracks. Wang et al. [150] used Hopfield networks to identify the sensor nodes that had the lowest power consumption rates. In another example, Levendovszky et al. [151] propose a Hopfield NN-based datalink layer algorithm for WSNs. The proposed algorithm attempts to schedule data forwarding in WSNs based on specific QoS metrics. The obtained QoS metrics are fed into Hopfield networks in order to find the data packets' optimum forward scheduling times. Tisza et al. [152] propose a multicast routing protocol for WSNs utilising Hopfield NNs. The proposed algorithm is based on the assumption that the routing information obtained by the network is incomplete. The proposed routing algorithm obtains the incomplete link's metrics from the WSN and utilises Hopfield networks to create the best routing tree that fulfils certain quality of service (QoS) criteria (e.g. routing delay).

2.3.6.4 Self-organising maps (SOM)

Self-Organizing Maps (SOM), also known as Kohonen maps, are unsupervised learning algorithms [153, 154]. In these networks, the neurons are organised in a regular manner and can be of the shape of one or many dimensional space. In the initialisation stage, each neuron in the network is assigned a random weight. The SOM training phase goes through two main stages:

- **Competition stage:** In this stage, the training samples are presented to the network. Then, they are compared to neurons' weights and finally the neuron that has the maximum value is considered the winning neuron. This comparison process is controlled by a discrimination function (such as inner product or Euclidean distance).
- **Adaptation stage:** In this stage, the winning neuron's weight is updated based on the neighbourhood function and the learning rate parameter.

In such networks, the learning procedure goes in iterative cycles. At the end of each iteration cycle, the number of neighbours and the learning rate are reduced [155]. After the learning procedure has been completed, it is possible to present patterns to the network in order to undertake classification processes. In the classification process, each of the presented pattern weights is compared to each neuron’s weight and then the neuron that has the closest weight becomes the input vector class.

One of the examples of the SOM implementation in WSNs is the work of Giorgetti et al. [156]. Here they proposed a localisation mechanism that determines nodes’ coordinates in WSNs based on SOM. Moreover, Postolache et al. [157] utilised sensor networks and a SOM algorithm to validate sensor failure and to detect pollution events. Regardless of the classification properties offered by SOM, centralised processing is required in order to compare weights and to determine the output class. Thus, tailoring SOM for utilisation in WSNs might be resource-exhaustive.

2.3.6.5 Recurrent neural networks (RNN)

Recurrent neural network (RNN), also known as feedback neural network, is a multi-layer structured neural network. As a feedback neural network, the output of RNN is fed back to the input in order to improve recognition accuracy and reduce error percentage [96]. These feedback links (i.e. from output to input) are not available in standard neural networks [158]. Figure 2.10 shows the RNN structure [96], which has an input layer, a hidden layer, and an output layer.

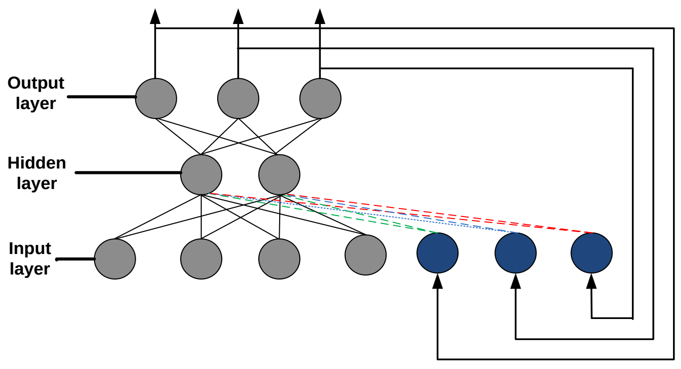

Figure 2.10: The structure of RNN [96].

Connor et al. [159] classified RNNs into two types: standard and relaxation. The standard RNNs work as standard neural networks with the feedback links. However, the relaxation RNNs perform learning and recognition continuously until the feedback inputs reach a precise predefined class. This would guarantee a predictable convergence time. However, in some scenarios and applications, such as time series prediction, it is impossible to achieve this goal.

One of the examples of the RNN implementation in WSNs is the work of Raju et al. [160]. The authors present a faulty data detection system for WSNs using RNN. The proposed system obtains the output of a sensor's neighbours to be fed as input into an RNN model in order to detect faulty information. Barron et al. [161] and Moustapha and Selmic [158] utilise RNN-based methods for fault detection in WSNs. They use an RNN to model a sensor node and its related communications with other nodes in the network. Their aim is to utilise previous output samples from communicating sensors in addition to the current and previous output samples of the modelled sensor as an input to the RNN model in order to detect failures in a dynamic environment.

2.3.6.6 Issues related to implementing NNs in WSNs

Neural Networks (NN) offer parallel pattern recognition capabilities for multiple problems. However, there are some issues that degrade the suitability of such techniques for pattern recognition in WSNs. One of the most prominent issues is the tightly coupled connectivity between neurons. In a single layered NN such as a Hopfield network, each neuron is connected to every other neuron in the network. In a multi-layer NN such as feed-forward networks, each neuron in a layer is connected to each neuron in an upper layer. Such tight connectivity between neurons will require a high number of network communications between WSN nodes, which means high power consumption. In addition, such connectivity limits a WSN that implement a NN technique from scale up in terms of network size.

Pattern recognition using NN techniques involves an iterative process. This means that a network performs actions such as weight calculations in repetitive steps until reaching an optimum status. The number of these steps is usually unpredictable and in some cases is not guaranteed to lead to an optimal solution. Consequently, the convergence time of an NN technique is high. Therefore, the suitability of such techniques in real-time WSN applications is limited. Furthermore, such iterative processes involve a large number of computations which will result in resource consumption when implemented in resource-constrained networks such as WSNs.

In some NNs such as Hopfield networks, and some types of the feed-forward networks, predetermined synaptic weights and relationships between nodes are required. Moreover, generally NNs require a large number of training samples in order to correctly classify incoming patterns. These requirements may be challenging in some applications, especially for environments where patterns are expected to occur randomly.

In general, NNs provide distributed and parallel pattern recognition capabilities. However, the performance of such schemes is affected by the large number of communications, iterative processing, and high computational resources involved, as well as the non-guaranteed convergence time and the predetermined weights and large number of pattern training samples that are required. These factors and requirements make the implementation of such schemes in resource-constrained networks such as WSNs either unfeasible or challenging.

2.3.7 Graph Neuron (GN)

A Graph Neuron (GN) is a graph-based pattern recognition algorithm that utilises a simple graph-like approach with *in-network* processing. This reduces the amount of processing required for recognition of a single pattern amongst a set of different patterns [162-164].

The use of a GN offers a parallel approach that permits several sensor nodes in the network to work together to recognise a pattern [164]. This parallelism helps to reduce the amount of processing required to identify a pattern by sharing these computations between nodes, which makes it very practical for use in wireless sensor networks [60, 164].

As an *in-network* processing algorithm, a GN's operations take place within the network body with the collaboration of nodes. This is an advantage that helps to resolve the issue that the processing required will increase as the target moves far away from the base node. Additionally, GN applications perform quite complex operations by dividing them into simple communicating sequential processes (CSPs). A GN can adapt more patterns with a reasonable increase in the overhead [164, 165]. All these characteristics make a GN a suitable pattern recognition system for wireless sensor networks.

The parallelism of the GN algorithm results from its use of associative memory (AM) systems, and with the aim of overcoming the shortcomings of other modern approaches. Moreover, the GN algorithm is an inclusive technology, with the possibility of incorporating other technologies such as spatiotemporal encoded neurons and evolutionary optimisation techniques [164].

The GN algorithm distributes its required computations between several nodes, instead of using normal sequential processing to achieve parallelism and a better AM system. Additionally, it has been implemented as a self-organising virtual network of processing nodes. Each node executes the same copy of a simple AM application, and it offers a parallelism framework. Hence, the GN algorithm is extremely suitable for parallel systems such as wireless sensor networks [63, 164]. The GN algorithm can accomplish AM by interconnecting nodes in a graph-like formation, called the GN array [166], which is illustrated in Figure 2.11.

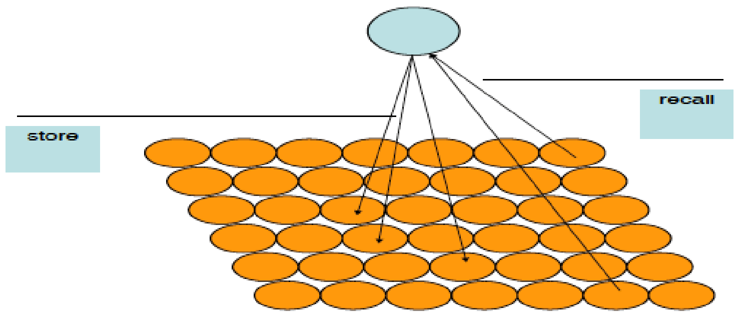

Figure 2.11: A GN array with parallel memorisation and recall operations [63].

In general, the GN algorithm performs pattern recognition in three stages [63, 166]:

1. Mapping the input patterns to the suitable nodes
2. Determining the end of the input pattern
3. Updating the bias and performing the memorisation, or recalling operation for the pattern.

Typically, these stages are executed in parallel inside a small network, and this solves the problem of finding a matching pattern in a large pattern domain [63].

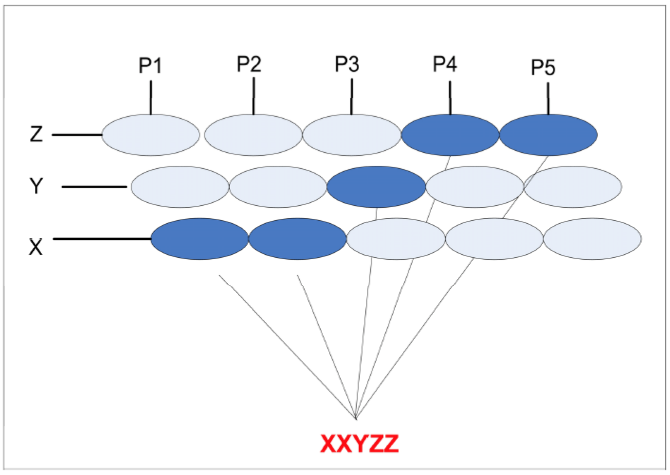

Figure 2.12: Data representation within the GN nodes for an input pattern XXYZZ.

The simplest form of a GN network can be implemented as a two-dimensional (2-D) array. In this implementation, the pattern is stored as a graph-like structure where each atomic component of the structure has its own (*value, position*) representation, as shown in Figure 2.12. In a multi-dimensional GN network, the number of values in each position may increase in order to represent the additional information. Through the implementation of such an array, the GN network is able to convert the spatial/temporal patterns into a graph-like shape. The GN then compares the edges of the graph in order to decide whether to recall or memorise the patterns. The main advantage of the graph-like form lies in its provision of a method for placing the spatial/temporal information in a single context. This provides a mechanism with which to compare the individual data points and the order in which these points occur. The disadvantage of this approach is that it requires a huge number of comparison processes to match the input patterns with the stored patterns [63, 165].

In order to perform pattern recognition [63], each node initialises a memory structure called the ‘bias array’ in which it stores the incoming pattern as sets of *p(value, position)* pairs. Each input pattern is automatically synthesised into its components by the GN array. The GN nodes corresponding to the respective *p(value, position)* pairs are activated by the input pattern. Each activated node exchanges its value and position with its neighbouring nodes (i.e. previous and next). In the memorisation process, a node will store the combinations of its own value and its neighbours’ values. For the recall process, it will look up the bias array for a matching combination. The node raises a recall, a ‘yes’ vote, if the combination is found in its bias array. Only if all GN nodes vote ‘yes’, recalled the input pattern, then the input pattern will be recalled by the network. Figure 2.13 illustrates the architecture and communications between GN nodes in a four-position GN array, with two possible values X and Y storing pattern YXYY.

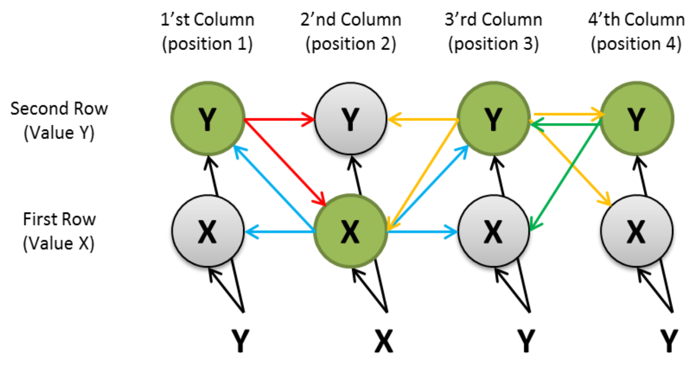

Figure 2.13: The GN array responses for the input pattern “YXYY”.

Despite the lightweight pattern recognition capabilities offered by the GN scheme, the recognition accuracy of the GN scheme is affected by the limited perspective of each neuron as each node knows only about its immediate neighbours. As a result, GN nodes may recall all of the sub-patterns together, leading to a complete recall, even when the complete received pattern has not previously been memorised. For instance, consider a GN array that can accept patterns made of six possible values, a, b, c, d, e, and f (rows), and five possible locations (columns). The pattern size is the same as the number of locations. Two input patterns have been memorised in this array: “abcdf” and “fbcde”. If the GN array receives an “abcde” pattern, then it will raise a false recall. This is known as the intersection or *crosstalk* phenomenon, which affects the accuracy of the GN algorithm. Furthermore, the use of GN scheme in WSNs is affected by the constraint that each node is required to communicate with a single entity (i.e. base station) in order to perform pattern recognition operations. Such requirements increase the number of communications and overhead throughout the network.

2.3.7.1 Hierarchal graph neuron (HGN)

The hierarchal graph neuron (HGN) [164] theory is the improved version of the basic GN theory, which can address the crosstalk phenomenon by adding higher GN array levels. The HGN fixes the crosstalk problem by using a pyramidal framework in order to obtain a better perspective of the incoming pattern. The HGN creates a set of layers above the neurons that receive the incoming pattern. The goals are to provide higher oversight over an incoming pattern and to minimise direct communications between nodes and the base station. The HGN is constructed using layers of GN arrays in a logical pyramidal shape that allows a single node (the top node in the pyramid) to classify the incoming pattern and communicate with the base station. Figure 2.14 shows an example of a simple binary HGN that handles a five-element pattern size.

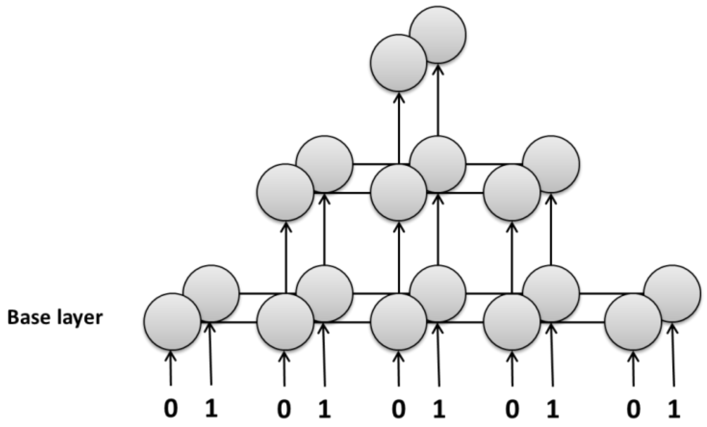

Figure 2.14: A HGN structure for a 5-element binary pattern.

The incoming pattern is firstly processed by the base layer of the HGN (i.e. base GN), and then each neuron sends its calculations to its corresponding higher level neuron. This process continues to the top node of the structure. This allows neurons in higher levels to acquire better knowledge about the incoming pattern. The top node decides the pattern's index based on a given command from the base station (memorise or recall). However, if the top node fails to classify the pattern, the base node communicates with lower level nodes to vote for an answer. It is evident that the neurons in layers higher than the base layer monitor and manage nodes. That is, these nodes do not receive pattern elements. Instead, they receive index numbers calculated by the base layer (and lower level) nodes.

We will go back to the same example that we utilised to explain the *crosstalk* issue in the previous section to demonstrate and explain how the HGN can solve this issue. In the example, the GN array can accept patterns made of six possible values, a, b, c, d, e, and f, and five possible positions. Both "abcdf" and "fbcde" input patterns have been memorised in this array. In HGN theory, when the GN array receives the "abcde" pattern, all GN nodes located in the bottom layer will raise a recall; however, when the GN array goes to the higher layers, the GN nodes located on them will discover that the underlying GN nodes have raised recalls from different stored patterns as they have a broader view of the input pattern. As a result, they will memorise the input pattern thereby resolving the pattern *crosstalk* phenomenon (see figure 2.15).

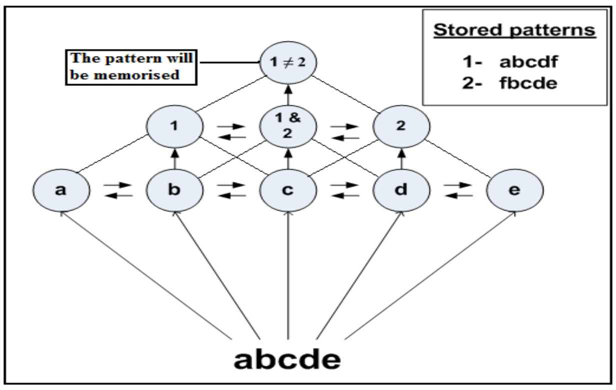

Figure 2.15: HGN array with the pattern crosstalk issue.

The HGN addresses the issue of *crosstalk* associated with GN schemes. However, the size of the HGN can scale significantly with the increase in pattern size due to the utilisation of managing neurons in its structure. If the pattern size is P_{size} and the number of possible values of a pattern element is v, then the size of the HGN network (HGN_{size}) can be calculated according to the following equation [164]:

$$HGN_{size} = v\left(\frac{P_{size}+1}{2}\right)^2 \tag{2.9}$$

Furthermore, the HGN attempts to reduce direct communications between each node in the network and the base station. However, with the presence of noisy patterns, an HGN network's top node will fail to classify the incoming pattern and the base station will communicate with nodes in lower layers to vote for an answer. Therefore, an HGN scheme's communications are affected by noisy patterns.

2.3.7.2 Distributed hierarchal graph neuron (DHGN)

The distributed hierarchal graph neuron (DHGN) [167] attempts to address the large scale of HGN and reduce the number of direct communications required for voting. DHGN splits an incoming pattern into sub-patterns so they can be processed by multiple HGN networks. Figure 2.16 shows an example of a DHGN structure for a pattern size of 9 that has been split into 3 sub-patterns and sent to 3 HGNs where each HGN processes 3 elements. Each HGN network processes the assigned sub-pattern and presents its final result through its top node. The base station conducts a voting process between the top nodes of the GN networks in order to make a final decision about an incoming pattern.

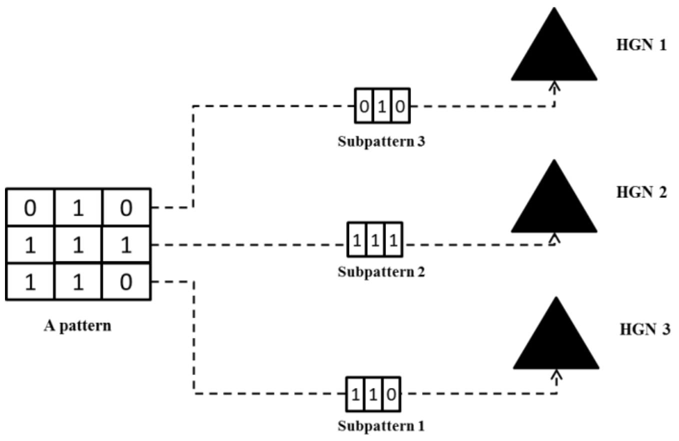

Figure 2.16: A DHGN structure for a 9 bits pattern size that has been divided into 3 sub-patterns.

DHGN decreases the number of nodes required for the construction of the network by limiting the number of managing neurons. However, the use of managing neurons leads to an increase in DHGN size with the increase in pattern size. For instance, in a uniform distribution of pattern size P_{size}, let SP_{size} be the sub-pattern size and n HGN networks. Thus, a DHGN network size ($DHGN_{size}$) can be calculated according to the following formula:

$$DHGN_{size} = n \times v \left(\frac{SP_{size} + 1}{2}\right)^2 \qquad (2.10)$$

DHGN adopts an HGN scheme. It has been shown that when a pattern is distorted (i.e. a noisy pattern), the top node of an HGN will not make a decision about the incoming pattern. Instead, the base station conducts a voting process that involves nodes in lower layers in order to inspect the result. Conversely, DHGN avoids such processes in order to limit direct communications with the base station and also to speed up the detection process. The main issue that a DHGN network may encounter is when a distributed noise is present. In this case, a DHGN network may fail to reach a conclusion about the incoming pattern. For example, if each sub-pattern in the example given in Figure 2.16 has been changed by at least one bit, each HGN network would fail to reach a conclusion regarding the incoming pattern. In other words, all top nodes will give the result 0 (i.e. fail to recognise the pattern). Since the base station conducts the voting process only amongst the top nodes of all HGN networks and does not involve lower layers, the network is unable to recognise the incoming pattern.

GN involves a limited number of communications and computations in performing learning operations. This feature makes GN a very good candidate for pattern recognition applications in resource-constrained WSNs. However, the accuracy of GN is constrained by the limited information available for each node. HGN and DHGN provide higher accuracy levels by involving a hierarchal network structure. Communications in both schemes are maintained at low numbers by adopting parallel and distributed mechanisms. However, the scalability of HGN and DHGN schemes is not the most appropriate for large scale WSNs as the number of required nodes increases exponentially with the increase of the problem (pattern) size.

2.4 Pattern Recognition Requirements in WSNs

Pattern recognition in WSNs is affected mainly by the limited physical design of sensor nodes, the nature of WSNs and the type of patterns a WSN is dealing with. The requirements of performing pattern recognition in large scale WSNs for real-time applications will be discussed in this section.

Sensor nodes are generally designed to be small in size, which restricts the resources available in each sensor. As discussed previously, a sensor consists of five main components: a processing unit (controller), a communication unit, a memory unit, an energy source (power supply), and sensing unit. The limited size of a sensor results in the limited size of these components. As a result, each task assigned to each one of this component can utilise only a restricted amount of resources. The energy source is one of the main factors influencing the performance of a sensor and the design of a WSN. In general, a sensor in WSNs utilises a battery that has a short life. Furthermore, in most applications, batteries are not likely to be replaced. This means that the lifetime of the battery determines the lifetime of the sensor. In addition, since the energy source of a sensor is limited, energy consumption caused by another sensor's components must be reduced.

In WSNs, a sensor node's communication is considered to be the most energy-consuming task, and can drain the sensor's energy resources [39]. Therefore, a pattern recognition scheme in WSNs should involve a restricted number of communications per sensor in order to increase the lifetime of sensors. Additionally, computational capabilities of a sensor node in a WSN are constrained due to the small sensor size (small processor) and limited energy available. As a result, involving large amounts of data processing in a sensor node is an exhaustive job that will shorten the life of the sensor. This is not only because data processing requires energy, but also because the sensor node will be kept in active mode for long periods of time. Moreover, the higher the processing assigned to a sensor node, the more time the sensor requires to obtain a result. If the amount of processing is large, the time required to obtain a result from this processing might be unexpected. Therefore, a pattern recognition algorithm in WSNs should involve a limited amount of data processing for each sensor node aligned with the sensor node's computational resources in order to avoid energy consumption and to ensure a timely result. The memory size of a sensor is also intuitively small. Therefore, each sensor node must hold the minimum amount of data it requires in order to process and detect patterns in a WSN pattern recognition scheme.

Other important requirements for pattern recognition in WSNs are due to the nature of the WSN network design. As WSNs are normally deployed in large numbers, network scaling is an important consideration when designing a pattern recognition scheme for WSNs. Size scaling requires managing the way in which sensor nodes are going to communicate with each other. The number of communications involved in a WSN scheme design is critical as it will determine the number of communications that each sensor is going to handle. This will have consequences for the sensor's lifetime as well as the time required to obtain a final result from these communications. In real-time applications, convergence time is extremely important. In pattern recognition processes, sensor nodes send data to either a fusion centre (i.e. base station) or to other sensor nodes in the network in order to conclude pattern detection. Consequently, the network's convergence time is highly dependent on the process of delivering information from one point to another. In general, scaling a WSN must maintain a limited method of communication to conserve energy resources and speed up the recognition process to support real-time applications in WSNs.

Furthermore, a pattern recognition scheme should have some invariant features. In WSNs, the need for such features increases because WSNs are dynamic and the nature of monitored fields of interest is changing. In other words, a memorised pattern in a WSN pattern recognition scheme could appear in a different form, such as size dilation or location change, in the field of interest. Or the WSN network's topology or sensor node locations might change, meaning the information memorised within the network will have different relations and distribution. Another issue associated with the nature of WSNs is the restricted number of training instances available as events generally occur in some form of randomness [42, 51]. Therefore, the design of a pattern recognition scheme must take into account the restricted amount of training data available as well as the changing environment in WSN networks and fields.

One of the most important requirements for a pattern recognition scheme is its ability to provide high recognition accuracy. Consequently, such a scheme should be capable of detecting events with a very small percentage of false alarms. Furthermore, it must achieve this without consuming the limited resources available in the sensor nodes. As mentioned previously in this section, a pattern recognition scheme can reduce its network consumption by involving a restricted number of communications per sensor, reducing the amounts of data processing per sensor, and retaining the minimum amount of data the sensor requires in order to process and detect patterns. In general, the pattern recognition scheme for event detection in WSNs should provide a balance between having a good level of accuracy and maintaining a restricted level of network consumption.

Furthermore, in order to have a high level of recognition accuracy, the pattern recognition scheme should be able to deal with transformed patterns. These patterns represent the state or condition of an event. In WSNs, there is a greater need for this capability because WSNs are dynamic and the nature of monitored fields of interest is changing. As previously mentioned, a memorised pattern in a WSN pattern recognition scheme could appear in a different form, such as size dilation or location change, in the field of interest. Also, the WSN network's topology or sensor node locations might change, meaning the information memorised within the network will have different relations and distribution. Moreover, the pattern recognition scheme must be able to handle noisy patterns in order to maintain a high level of accuracy. The noisy patterns are mainly the result of the monitoring environment and the limited life of sensor nodes. The noisy patterns, the sensor nodes' damages, dead sensor nodes, and lost packets might cause the loss of some parts of the detected pattern. Therefore, generally a pattern recognition scheme for WSNs should be capable to detect events and patterns even if some parts of the detected pattern are lost.

Many WSNs applications, such as fire surveillance and human activities recognition, are real-time applications and therefore require fast pattern recognition. This can be achieved by reducing the number of communications required to perform pattern detection, which will lead to the conservation of energy and acceleration of the recognition process. In general, the pattern recognition scheme in WSNs should limit the amount of communication to save energy resources and speed up the recognition process to support real-time applications in WSNs.

Generally, the main requirements of a pattern recognition scheme in large-scale WSNs suitable for real-time applications can be summarised as follows:

- Restricted communications, computations, and memory requirements.
- Ability to scale in terms of network size.
- Predicted convergence time.
- Ability to addresses invariance properties for dynamic networks and changing patterns.

- Ability to solve randomness problems, meaning that the scheme should maintain high accuracy with a restricted number of available training instances.
- Ability to detect complex and noisy patterns.
- High level of recognition accuracy while maintaining a limited level of network consumption.
- Ability to provide fast pattern recognition to support real-time applications in WSNs.

2.5 Comparing Existing Schemes

This section compares the different pattern recognition schemes presented in section 2.3 for WSNs based on the requirements listed in the previous section 2.4. The main aim of this research is to present a recognition scheme for event detection that is capable of detecting transformed and noisy patterns using a minimal amount of available information about patterns while addressing the resource constraints of WSNs.

The main aim of pattern recognition in WSNs is to offer greater recognition accuracy and reduce the number of false alarms. Simple schemes, such as threshold-based scheme, seem to be perfect for simple problems. On the other hand, these schemes fail to deal with complex patterns, leading to false alarms. The nature of WSNs and the fields they monitor can create more complex problems for recognition schemes. Sensor nodes could run out of energy or lose information because of the noise in the physical transmitting medium. Therefore, schemes should be capable of overcoming such challenges and offer recognition capabilities despite the lost information. Such problems can decrease the accuracy of several known schemes. For instance, a DHGN scheme might inaccurately classify patterns when distributed noise is present. If each cluster of a DHGN network is presented with noise, cluster heads will not be able to conclude sub-pattern detection and the final voting of the process could lead to inaccurate detection. Another instance is decision tree schemes. These techniques may fail to correctly classify noisy patterns in large-scale networks.

Most existing schemes are able to implement a number of nodes equal to the pattern size P_{size}. For example, Hopfield networks allow input/output neurons of size P_{size}. Conversely, HGNs and DHGNs require a larger number of nodes to adopt the same patterns, as can be seen from Equations 2.9 and 2.10. The higher number of nodes in HGNs and DHGNs is the result of requiring higher level neuron positions and the need to have one node for each possible value v in each position. It can be concluded from Equations 2.9 and 2.10 that the number of nodes (or sensors) increases exponentially with the increase in pattern size and the number of possible values for each pattern element.

On the other hand, the number of communications involved in neural networks such as Hopfield networks is high due to tightly coupled connectivity and iterative processing. The number of communications required for a Hopfield network is $(P_{size} \times (P_{size} - 1))$ as each node is connected to every other node. Intuitively, the number of communications grows exponentially with the increase in pattern size as this number is related to the square of the pattern size. Since neural networks require iterations to reach an optimal state, communications between neurons are repeated several times, resulting in high communicational demand that would be exhaustive if implemented on sensor nodes. Some statistical approaches such as histogram methods also involve a large number of communications in order to collect specific information from the network.

From a computational perspective, most of the existing schemes either provide distributed processing or require centralised processing. SVM, for instance, requires centralised processing in order to create the required hyperplanes and classify patterns. Statistical approaches require global information to be available in a centralised component in order to compute distributions and perform recognition. There have been attempts to distribute statistical models amongst sensor nodes in WSNs and compute these distributions locally before sending the information to a fusion centre or a base station. The work of Luo et al. [79] is an example of this technique. However, the accuracy of such techniques depends on the physical communication medium’s noise tolerance and the thresholds computed to perform computations locally in sensor nodes. Neural networks provide parallel and distributed functionality in terms of computations. However, the iterative process of neural networks requires a great amount of data processing. KNN techniques can be seen as simple, distributed approaches for pattern recognition. However, the computational complexity of a KNN scheme depends on the number of neighbours k. The higher the value of k, the more complex the scheme becomes. Therefore, the tuning of the value of k plays a crucial role in determining a KNN scheme's computational simplicity.

Memory requirements per sensor node are limited in most existing schemes. However, in KNN, each sensor node keeps information about distances to each of its neighbouring nodes. The amount of memory needed for each sensor node in this case will depend on the value of k and the number of classes. In the HGN, the higher nodes in the hierarchy require more memory. In the base layer (i.e. input layer) each sensor node is expected to hold up to (2^v) in its memory (v is the possible number of a pattern's element values) as each sensor node communicates with its two direct neighbours. Each node in the top position of the hierarchy of the HGN is expected to hold up to $\frac{Number\ of\ the\ training\ samples}{v}$.

The number of available training samples is commonly restricted in WSNs. Most existing detection schemes require a large amount of data in order to correctly recognise and classify patterns. Statistical approaches utilise training samples so as to construct distribution probabilities. The more samples the scheme is trained with, the higher the accuracy it achieves. Similarly, SVM requires large amounts of data in order to create separation hyperplanes. A limited amount of data could result in inaccurately setting hyperplanes and create large gaps between classes. Neural networks have the same requirement in order to accurately create weighting matrices. The limited number of training samples in this case affects the invariant property of a recognition scheme. Statistical, SVM and neural network approaches are the best candidates for offering this invariant feature compared to other existing schemes. However, this feature is encumbered by having to present a large number of training samples to a network implementing such schemes.

Real-time applications require fast pattern detection. In this area, GN approaches such as HGNs and DHGNs offer one-cycle recognition that suits such applications. On the other hand, neural networks, SVM, and decision trees recognition schemes may require more time to converge compared to other existing schemes. A neural network's convergence time to an optimum state depends on the number of iterations involved. A single iteration involves communications and computations to be performed by neurons. These activities can be time consuming as the number of iterations grows. In SVM schemes, recognition time depends on the selection of the kernel. If the kernel chosen is one of the time-costly techniques, such as neural networks, the detection time will intuitively increase. The choice of kernel in this case will be a trade-off between time and other factors such as accuracy. For decision trees, recognition time depends on the depth of a tree. The depth of a tree is the number of levels required to perform recognition and depends on the number of attributes the tree is inspecting. The more attributes to inspect, the greater the depth of the tree, and hence the more time it takes to reach a decision about a pattern. In addition to the depth of a tree, the method utilised to inspect each attribute affects the time cycle of detection.

It can be clearly seen that different schemes have different limitations in regards to the requirements of pattern recognition in WSNs. Table 2.2 shows a comparison between existing pattern recognition schemes in WSNs. It can be clearly seen from the table that none of the existing schemes can meet all the necessary requirements. Therefore, new schemes need to be proposed in order to fulfil all of these requirements.

2.6 Proposed Solution

Performing pattern recognition in WSNs requires tackling two main issues: correctly classifying patterns and restricting the use of constrained resources. Solving the problem of pattern recognition in WSNs is seen as a trade-off between accuracy and resources exhaustion [77, 168]. Existing solutions do not address the resource-constrained nature of WSNs and assume reliable message delivery in the network [79]. This causes such schemes to be resource-exhaustive and to require substantial tailoring to suit WSN applications [44]. Therefore, Tanenbaum et al. [53] concluded that existing techniques can be implemented only on limited-scale WSNs.

Table 2.2: The capability of the existing pattern recognition schemes to fulfil the thesis requirements.

Approach	WSN requirements					Pattern recognition requirements			
	Communications	Computations	Memory	Network	Time	Transformation invariant (Transformed patterns)	Random patterns (restricted number of available training samples)	Complex patterns	Noisy patterns
Threshold-based	Low	Low	Low	Small	Low	No	No	No	No
Statistical approaches	Low	High / Centralised	Moderate	Small	Low	Yes	No	Yes	Yes
KNN	Low	High (Depends on *k*)	High	Small	Low	No	Yes	Yes	Yes
Decision trees	Low	Dependant	Low	Small	Low	Yes	No	Yes	No
SVM	Low	Centralised	Low	Small	Low	Yes	No	Yes	Yes
Neural networks	High	High	Low	Small	High	Yes	No	Yes	Yes
Graph Neuron (GN)	Low	Low	Low	Small	Low	No	Yes	Yes	Yes
HGN	Low	Low	Moderate	Large	Low	No	Yes	Yes	Yes
DHGN	Low	Low	Moderate	Large	Low	No	Yes	Yes	No
Proposed scheme (target)	Low	Low	Low	Small	Low	Yes	Yes	Yes	Yes

In recent research, distributed approaches have been used to address the pattern recognition problem in WSNs. According to Giridhar and Kumar [169], sending information from each sensor node to the base station or a fusion centre in a WSN is inefficient. Consequently, according to the authors, the whole network should perform as a distributed cooperative computational component. Wittenburg et al. [44] suggest that the processing involved in application levels should be pushed and distributed in the network level in order to achieve conservative resource consumption schemes for WSNs. Chamberland and Veeravalli [39] suggest that the utilisation of distributed pattern recognition is the most efficient method for WSNs. They state that data should be computed by sensor nodes locally, and only part of the resulting information should be sent, in order to conserve the WSNs' limited resources. However, the authors highlight that the choice of which information should be sent to the base station of a WSN is crucial to such implementation.

Furthermore, using *in-network* processing can solve the pattern recognition problem in WSNs, since sensor nodes go beyond their basic functionality of sensing and passing sensory data to actually being involved in processing data en-route within the network [170-174]. There are many advantages in utilising *in-network* approaches. First, these approaches are more robust against individual sensor node or link failures. They are also normally designed in a distributed manner, which improves the network scalability. Finally, they require less communication, which leads to better resource and bandwidth utilisation and which in general improves the longevity of the network.

The main hypothesis of this research is that the best means of addressing the issue of event detection in WSNs is to have a pattern recognition algorithm combined with fully distributed and parallel techniques, which works purely with localised node adjacency-based relationships (i.e. computations). Moreover, it will do so without requiring huge computational resources, making it suitable for wireless sensor networks. The parallelism helps to reduce the amount of processing required to detect an event by sharing these computations among sensor nodes. This capability makes it possible for such a scheme to perform quite complex operations by dividing them into simple processes that are suitable for resource-limited sensor nodes. Adjacency-based relationships (i.e. computations) will increase a WSN's ability to handle complex, invariant (i.e. transformed), and noisy patterns, which will in turn increase the recognition accuracy. Additionally, this research expects that the use of a loosely coupled connectivity scheme will scale up efficiently in terms of time and resources management when it is utilised in resource-limited networks like WSNs. By offering a fast and accurate scheme that suits WSNs, it is possible to utilise such scheme to address real-time application problems.

2.7 Conclusion

This chapter has presented an overview of WSNs and the pattern recognition challenges that are associated with such networks. WSNs pose many challenges for complex applications such as pattern recognition and event detection. WSNs face greater challenges if an application is a real-time one and needs to be implemented on a large-scale network. These challenges and limitations stem from the constraint resources that a WSN can offer, including computational, communicational and memory resources. As a result, these limitations mean that pattern recognition issues in WSNs need to be addressed by means of a trade-off between performance and resource consumption.

Existing schemes that offer solutions for the pattern recognition issue in WSNs include threshold-based, statistical, KNN, conditional, SVM, neural networks GN, HGN, and DHGN schemes. Each one of these schemes presents several issues when implemented on WSNs. Examples of these issues include the requirements of centralised processing, iterative processing and large numbers of communications. Therefore, each one of these schemes would require substantial modification in order to be adopted by WSNs. Most existing pattern recognition schemes can be implemented on limited-scale size WSNs. Therefore, efforts should be made to enhance the scalability and performance of pattern recognition.

Distributed and *in-network* processing techniques conserve resources in a way which perfectly suits the nature of WSNs. These techniques allow the spread of computations across the network. Distributed processing can be achieved by allowing each sensor node to locally process data and send the final result to another entity in the network. However, it is important to utilise a method for choosing which other sensor nodes to communicate with, what data should be processed and what information should be sent.

In WSNs, minimising communications between nodes is one of the best approaches to resource utilisation. The main reason for this is that sending a message from one sensor node to another is the most energy-consuming task that a sensor node can perform. Therefore, this chapter proposes the use of the adjacency communication technique in order to reduce the number and range of communications. The processing of data acquired from adjacent nodes allows the network to communicate and process data in a loosely coupled fashion. In addition to conserving resources, this would allow the network to decrease the time needed for recognition. Such features will make the proposed schemes good candidates for resource-constrained networks such as WSNs, allowing them to solve complex and real-time problems.

Chapter 3

Macroscopic Object Heuristics Algorithm (MOHA)

3.1 Preamble

Wireless Sensor Networks (WSNs) make it possible to sense physical parameters in a field of interest. These sensory data can be analysed in order to detect the presence of a physical activity or events in that field of interest and take action in accordance with the detected activity. The analysis of sensory data is a processing task that can be done using two approaches: centralised and *in-network* [19, 170-174]. In centralised processing, data obtained by sensor nodes are aggregated in one machine that has the computational ability to analyse sensory information. However, this approach is considered inefficient in large-scale, resource-constrained WSNs, especially for applications that require data to be analysed quickly in order to support decision-making processes [169]. In contrast, *in-network* processing allows network sensor nodes to perform computations on obtained data locally and in a distributed manner. This causes the network to act as a cooperative computational entity and this capability allows a WSN to reduce the computational and communicational complexity of processing sensor nodes in the network.

The problem of event detection in WSNs can be addressed by utilising *in-network* pattern recognition techniques. A pattern is defined in this research as a set of raw sensory data that describes the main characteristics or attributes of an event. As discussed in Chapter 2, *In-network* pattern recognition techniques for WSNs include threshold-based, nearest neighbour, fuzzy logic, and neural networks. Existing pattern recognition schemes for WSNs are usually tailored to provide detection capabilities for specific applications or problem scenarios. However, these existing schemes may fulfil some pattern recognition requirements but fail to address WSN resource limitation issues.

As discussed in Chapter 2, Graph Neuron (GN) is a scheme that creates associative memory (AM) in a fully parallel-distributed manner over fine-grained WSNs and provides lightweight, one-shot learning capabilities. These characteristics make GN a very good candidate for real-time pattern recognition applications in WSNs. However, the recognition accuracy of GN is affected by the limited perspective of each neuron, as each sensor node knows about its immediate neighbours only. HGN and DHGN provide higher accuracy levels by involving a hierarchal network structure. Communications in both schemes are maintained at low numbers by adopting parallel and distributed mechanisms. However, the scalability of HGN and DHGN schemes is not best suited for large-scale WSNs as the number of required nodes increases exponentially with the increase of the problem (pattern) size. Moreover, HGN and DHGN schemes do not have the capability to correctly detect transformed patterns, which is one of the main objectives of this research. Thus, the first hypothesis presented in this chapter is that the development of a GN-based scheme that addresses the accuracy limitations of GN would be the best option for solving the problem of pattern transformation detection and pattern recognition in resource-constrained networks such as WSNs.

Furthermore, in some real-life applications, patterns and network nodes are subject to spatial and topological changes [175]. This means that a pattern can appear with a level of variations or transformations such as location change. For instance, a malicious intruder pattern of a WSN can change its location in the network [176]. Another instance can be seen in environmental surveillance systems where events such as forest fires and hurricanes may appear in different locations and at different magnitudes [177]. Such dynamics in pattern appearance can influence the accuracy of a recognition system. Dealing with such dynamics could require a great amount of information that need their patterns and possible dynamics to be stored in WSN in order to efficiently recognise the presence of these patterns. However, such an approach might not be feasible, for two main reasons. First, it would require a high amount of resources to store and search patterns. Second, in some applications, the amount of available information about patterns is limited, as discussed in Chapter 2. Therefore, a more efficient approach is required.

Hence, in this chapter, we propose a pattern recognition scheme that is capable of detecting events, and transformed and noisy patterns while addressing the resource constraints of WSNs. The scheme will adopt an *in-network* processing paradigm by including GN in its structure. Additionally, the scheme will address the *crosstalk* problem that affects GN accuracy by adopting network structures that allow certain nodes in the network to maintain more information about incoming patterns rather than being restricted to only adjacent nodes. Instead of storing pattern information locally on sensor nodes, the proposed scheme implements an adjacency-based relationships mechanism that searches the edges and boundaries of events and patterns. The second hypothesis in this chapter is that, by describing events and patterns using their main edges and boundaries, it is possible to achieve an efficient recognition scheme that can detect transformations that may occur in these events and patterns. The idea of using the edges and boundaries of pattern information to detect transformed patterns has also been successfully used and published in 2012 by Alfehaid et al. in [178]. However, the proposed scheme in this chapter was first published and used as a method of detecting transformed patterns in 2010 in [179]. The proposed scheme maintains limited communications and computations by involving local information exchange and reporting mechanisms that distribute resource consumption loads amongst the network's nodes. Furthermore, the proposed network structure is designed to have the same network size as the GN network, which maintains the high scalability and the one-shot learning feature of GN.

The objectives of this chapter are as follows:

1. To present and analyse existing transformation invariant pattern recognition schemes.
2. To investigate how edge detection techniques are used in both image segmentation and WSNs. This also includes a discussion on the benefits that can be obtained from using edge detection techniques in WSNs for event detection.
3. To propose a new pattern recognition scheme that is capable of detecting transformed and noisy patterns and addressing the resource constraints of WSNs.
4. To perform an extensive evaluation and analysis of the complexity of the proposed scheme in terms of memory size, number of communications, and learning cycle time. Additionally, the proposed scheme will be compared with current pattern recognition schemes in these terms.
5. To present simulation results of the proposed scheme in order to ascertain its strengths, especially when dealing with transformed and noisy patterns.

The remainder of this chapter is organised as follows: Section 3.2 contains an overview of existing transformation invariant pattern recognition schemes, and discusses the limitations of these schemes. Section 3.3 investigates how edge detection techniques are used in both image segmentation and WSNs. A discussion of the benefits that can be obtained from using edge detection techniques in WSNs for event detection will be presented at the end of this section. Section 3.4 presents a MOHA scheme structure for pattern recognition. In this section, there is a further description of the scheme and its associated components. Section 3.5 shows how the MOHA scheme receives incoming patterns. Section 3.6 presents the communicational requirements of the MOHA scheme and its network communication protocol. Section 3.7 provides an extensive review of the analyses that have been carried out on the MOHA algorithm. These analyses focus on both the complexity and scalability of the algorithm. The aim of these analyses is to validate the suitability of MOHA for use in WSN environments. A number of comparative analyses between MOHA and other existing pattern recognition algorithms are also conducted, and Section 3.8 presents a set of pattern recognition simulation studies that have been carried out using the MOHA algorithm for pattern detection and recognition. The section also presents the MOHA algorithm capabilities to handle transformed and noisy patterns. Section 3.9 provides an overall discussion of the MOHA scheme.

3.2 Existing Transformation Invariant Recognition Approaches

This section presents and analyses the most relevant research in the area of pattern transformation-invariant recognition. Le Cun et al. [180] proposes convolutional neural networks (ConvNets) that utilise multi-layer (deep learning) neural networks to encode high-dimensional patterns into a one-dimensional vector. This scheme allows feature selection by adopting local receptive fields, local weight-sharing, and down-sampling architectures [181-184]. The local receptive fields allow for the detection of the visual features of a pattern, the down-sampling architecture reduces the dimensionality of the pattern, and the concept of weight-sharing allows for a level of shift and scale invariant detection [181]. ConvNets have mostly been utilised for visual pattern recognition such as facial and handwriting recognition applications [181-184]. Farabet et al. [185] propose a highly parallelised version of ConvNet on a Field Programmable Gate Array (FPGA) to support real-time applications [185, 186]. ConvNet takes advantage of the parallel nature of FPGA and achieved a greater speed in performance [185-187]. On the other hand, shock graphs [188] attempt to recognise visual objects by determining the object's boundaries. This approach builds connected curves and points that describe certain patterns in a tree-like graph. Sebastian et al. [189] show that the use of shock patterns allows for 2-D object transformation-invariant recognition up to a certain level of transformation. The authors examined a set of pattern transformations such as articulation and deformation, illumination variations, and variations in the scale of objects. Alternatively, Map Seeking Circuits (MSC) [190] utilise the mathematical properties of pattern superpositions to perform template matching, which seeks a set of transformations of an input visual pattern for a set of stored templates. The purpose of the MSC approach is to reduce the growth in the computational complexity of template matching by parallelising computations in the hardware. MSCs offer solutions for visual applications such as stereo vision, shape recognition, and other transformation-recognition problems that can be addressed using iterative processing and the decomposition of pattern transformations [191]. Moreover, Ahmed and Faouzi in [192] present a system for classifying images in order to classify people's behaviours in the intelligent building by Scale Invariant Feature transform (SIFT) and SVM Lights. This system permits us to characterise the activity of people in a room. This information will be useful to the management system of the building which can then regulate the consumption of electrical energy in order to optimise (lighting, heating, etc...). SIFT is a good method for the parameterisation of images, robust to image rotation, scale, intensity change, and to moderate affine transformation [192, 193]. SIFT transforms an image into a large collection of local feature vectors, each of which is invariant to image translation, scaling, and rotation, and partially invariant to illumination changes and affine [194].

Even with the transformation detection capabilities discussed above, these schemes involve computationally intensive (e.g. iterative) operations that are poorly suited to real-time sensory applications due to prolonged learning cycles, the requirement for large training datasets that are often not available, and dependence on specialised hardware (like FPGA). Additionally, based on a lattice algebra approach, morphological associative memories (MAM) [181, 195] are capable of detecting pattern transformation in a single-step convergence. MAM approaches utilise a morphological neural network structure that replaces multiplication and addition operations by addition and convergence maximisation. MAM approaches have been shown to be scalable in terms of pattern storage, and capable of detecting noisy patterns. However, the length of the MAM learning cycle cannot be fully estimated a priori, as it totally depends on the size and number of stored patterns [164]. Moreover, Iftekharuddin [196] proposes an online transformation scheme for automatic target recognition. This scheme resolves the issue of recognising rotation, translation, and scaling pattern translations in images by adopting adaptive circuit design, neural networks, and reinforcement learning. Even with the level of transformation recognition that this scheme is able to provide, it depends on conventional neural network structures such as feed-forward NNs. In this network structure, tightly coupled connectivity is needed between neuron nodes in each layer when FPGA is not used. Sensory information systems, such as WSNs, have limited communicational capabilities, which make it challenging to implement the tightly coupled connectivity structure [77].

Additionally, Deng et al. [197] propose a transform-invariant PCA (TIPCA) technique intended to accurately characterise the intrinsic structures of the human face that are invariant to the in-plane transformations of the training images. The experimental results in [197] prove the significance of employing transform invariant basis (principal components in PCA) for face recognition tasks [197, 198]. However, the TIPCA framework cannot guarantee the recognition rate as it is an unsupervised learning process [198]. Also, in order to obtain a good recognition rate, numerous iterations of TIPCA learning must be applied, which is not suitable for real-time and mission-critical applications as no guarantees for time limits are available [197, 198]. Shen et al. [199] propose a Transformation Invariant Support Vector Machine (TISVM), which is able to infer critical transformations that can be applied to the existing samples to generate new training samples while simultaneously learning the large margin classifier. In [199], two experiments are conducted: one for the recognition of handwriting digits (numbers) and another one for face recognition. Both tests show that the TISVM outperforms the traditional SVM in terms of classification precision. However, in TISVM, the number of training samples increases rapidly as the scheme generates new training samples for every existing sample, which is not suitable for resource-constrained environments like WSNs.

Overall, the existing approaches can offer transformation-invariant detection capabilities for pattern recognition problems. However, these approaches have one or more significant limitations, especially if implemented in resource-constrained networks like WSNs. Iterative operations and tightly coupled connectivity structures are the main issues faced in these existing schemes as these requirements involve huge resource consumption, especially when implemented on large-scale networks. Some of these schemes need special centralised hardware settings in order to be functional. This requirement is not often feasible in resource-constrained systems and networks.

Furthermore, the reviewed schemes need a large database to store information about patterns in order to be able to provide transformation-invariant recognition capabilities. In WSN applications, the amount of information available about incoming patterns is often limited. Another main problem related to these approaches is the uncertainty in the learning cycle convergence period (the share time they take to be specified), which requires them to utilise special hardware like FPGA. This is due to the iterative operations and the duration of the learning cycle which depends on the amount of stored information. Such limitations affect the suitability of these approaches for implementation in real-time and mission-critical applications and high latency networks (like WSN) as no guarantees for time limits are available. Therefore, this chapter will present a lightweight pattern recognition scheme intended to address these limitations and provide transformation-invariant recognition capabilities. The proposed scheme minimises computations and communications by adopting an adjacency-based relationships mechanism in a single learning cycle. This will minimise resource consumption and increase the scheme’s scalability by avoiding iterative operations and limiting communications to adjacency sensor nodes. This also makes it possible to estimate the learning cycle duration. Moreover, the proposed scheme attempts to utilise minimal information to perform transformation recognition. By offering these features, the proposed scheme will be a lightweight, transformation-invariant, and scalable scheme that is suitable for real-time applications for resource-constrained systems and networks such as WSNs.

Alfehaid et al. [178] presented a scheme that adopts an adjacency-based relationships mechanism. They proposed a Cellular Weighted Pattern Recogniser (CWPR) for WSNs that provides for the storing and detecting of patterns with translation, dilation, or rotation. The input patterns are described in terms of their edges. This is achieved by computing the amount of variance between each pattern element's value and its neighbours' values. Once a pattern's edges have been determined, each edge is described in terms of weights. This is done to minimise on the number of computations and communications steps required for storing and recognising patterns. The scheme proposed in [178] might provide a suitable solution in terms of reducing the number of computations and communication steps needed for pattern recognition; however, the way it determines edges and describes them as weights limits the scheme's ability to offer high levels of recognition accuracies when dealing with pattern transformation. In [200], the simulation results indicate that CWPR was very good in detecting translated patterns at any level; however, it is capable of dealing with dilation levels only up to 26% and recognising rotated or even flipped rotated patterns up to 23 degrees in any direction. The scheme proposed in this chapter will utilise a mechanism for determining edges of an event based on a well-known edge detection technique. This is a more accurate means of determining edges, thereby producing better recognition accuracy.

Before presenting the structure of our proposed scheme for event detection and pattern recognition, the next section will present an investigation of edge detection techniques that are used in both image processing and WSNs. The outcome of this investigation will give us a better understanding of the feasibility of using edge detection methods in a recognition scheme for detecting events, and transformed and noisy patterns.

3.3 Edge Detection

This section will investigate how edge detection techniques are utilised in both image processing and WSNs. This will be followed by a sub-section that discusses the benefits of utilising edge detection techniques in WSNs.

3.3.1 Edge detection in image processing

The detection of object boundaries is an important part of the perception process [201, 202]. There is much psychological evidence, such as in our ability to understand and distinguish different objects in cartoons, that boundaries are sufficient for the perception of a broad class of objects [201, 202]. In this section, the most commonly used techniques for detecting boundaries by aggregating evidence from local discontinuities, are described.

The edge-detection techniques attempt to find local discontinuities in some image attribute, such as intensity or colour [203-207]. These discontinuities are of interest because they are likely to occur at the boundaries of objects. However, local edges may also occur due to variations in the surface characteristics of an object, changes in illumination and shadows, and so on [203-207]. An important process of perception is the organisation of the local edges into aggregates that lead to a scene segmentation [208-211]. Thus, the process of edge detection may be viewed as one stage of extracting descriptions from the image data.

Nevatia [202] defines an edge as a point in the image if some image attribute changes in value discontinuously at that point. Suman and Pawan [212] define an edge as a part of an image that contains significant variation. The edges provide important visual information since they correspond to major physical, photometrical or geometrical variations in a scene object [212].

Here we will discuss intensity edges, as edges in other attributes can be defined similarly [202, 204, 206, 207, 212]. An ideal edge, in one dimension, may be viewed as a step change in intensity. In real signals, the step change is likely to be mixed with "noise" caused by sensor, surface, or illumination variations, as shown schematically in Figure 3.1. In two dimensions, the ideal step occurs along a line of certain length, the intensity values on the two sides of the line being different.

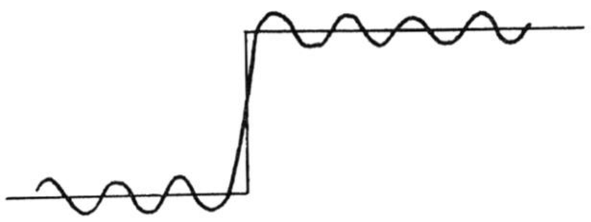

Figure 3.1: A one-dimensional edge [202].

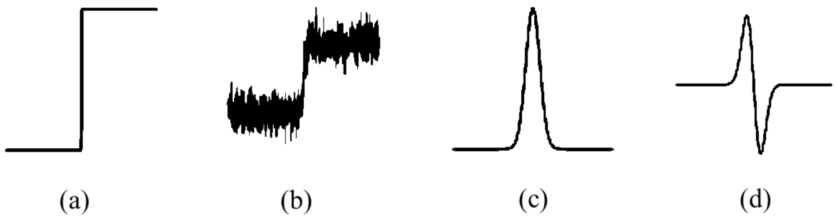

Figure 3.2: Examples of: (a) Step edge, (b) Step edge with noise, (c) First-order derivative, and (d) Second-order derivative [206, 207].

Moreover, the most common edge types are steps, lines and junctions [202, 204, 206, 207, 212]. The step edges are mainly produced by a physical edge, an object hiding another, or a shadow on a surface. It generally occurs between two regions having almost constant, but different, grey levels. The step edges are the points at which the grey level discontinuity occurs, and localised at the inflection points. They can be detected by using the gradient of intensity function of the image. Step edges are localised as positive maxima or negative minima of the first-order derivative or as zero-crossings of the second-order derivative (Figure 3.2). It is more realistic to consider a step edge as a combination of several inflection points. The most commonly used edge model is the double step edge [206]. There are two types of double edges: the pulse and the staircase (Figure 3.3).

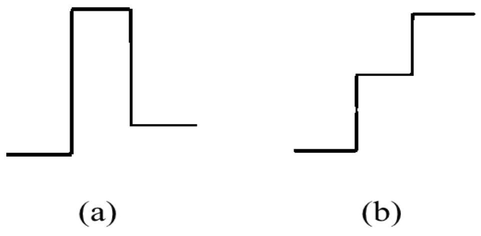

Figure 3.3: Examples of: (a) Pulse and (b) Staircase step edges [206, 207].

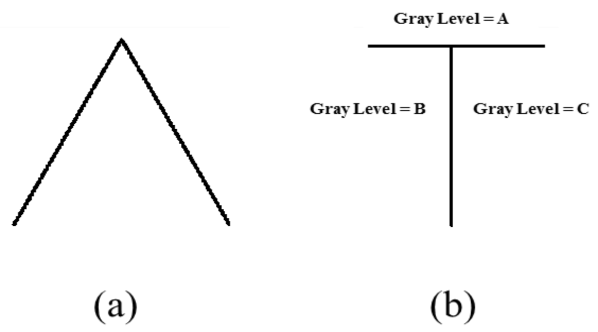

Figure 3.4: Examples of: (a) Line and (b) Junction edges [206, 207].

The line edges are often created by either a mutual illumination between two objects that are in contact or a thin object placed over a background [202, 204, 206, 207, 212]. Line edges correspond to local extremes in the intensity function. Lines correspond to local extrema of the image. They are localised as zero-crossings of the first derivative, or local maxima of the Laplacian, or local maxima of the grey level variance of the smoothed image. This type of edge is successfully used in remote sensing images for instance to detect roads and rivers [206, 213]. Finally, the junction edge is formed where two or more edges meet [202, 204, 206, 207, 212]. A physical corner is formed at the junction of at least two physical edges. Illumination effects or occlusion, in which an edge occludes another, can produce a junction edge. Figure 3.4 depicts profiles of line and junction edges. The junction can be localised in various ways: e.g., a point with high curvature, or a point with great variation in gradient direction, or a zero-crossing of the Laplacian with high curvature or near an elliptic extremum. Though, some studies encompass the all types of edges, but the majority of the reviewed literature is adapted to step edges, which are the most common [206, 207].

The edges extracted from a 2D image of a 3D scene can be classified as either viewpoint dependent or viewpoint independent [202, 206, 214]. A viewpoint independent edge typically reflects inherent properties of the 3D objects, such as surface markings and surface shape. A viewpoint dependent edge may change as the viewpoint changes, and typically reflects the geometry of the scene, such as objects occluding one another. Detection of ideal step edges, without noise, would be simple [202, 204, 206, 207, 212]. In real images, with noise and surface imperfections, a compromise must be achieved between maximising the detection of the desired edges and minimising the detection of the undesired noise edges. (Some of the "undesired" edges are due to legitimate surface variations, such as texture, and the local edge detection process is not intended to discriminate against them).

In general, the edge detection methods incorporate three operations [206, 215]: image smoothing, detection, and edge localisation. Image smoothing reduces noise and regularises the numerical differentiation. The detection step involves extracting all edge points that are possible candidates to become edge points. Finally, the edge localisation step involves selecting from the candidate edge points only the points that are true members of a set of points comprising an edge.

Edge detection techniques have been classified from different perspectives [206, 207]. From the technical perspective, the edge detection methods can be grouped into two categories: search-based and zero-crossing based. The search-based methods detect the edges by first computing a measure of ‘edge strength’, such as magnitude of gradient of the image intensity function, and then searching for local maxima in a direction that matches with the ‘edge profile’, such as the gradient direction. The first-order derivative is regularly used to express the gradient. The zero-crossing methods search for zero crossings in a second-order derivative expression computed from the image in order to find edges, such as the Laplacian or a non-linear differential expression. In the perceptual perspective, the edge detection methods are categorized into contextual and non-contextual approaches. The non-contextual methods work autonomously without any prior knowledge about the scene and the edges. They are flexible in the sense that they are not limited to specific images. However, they are based on local processing focused on the area of neighbouring pixels. The contextual methods are guided by a priori knowledge about the edges or the scene. They perform accurately only in a precise context. It is clear that autonomous detectors are appropriate for general-purpose applications. However, contextual detectors are adapted to specific applications that always include images with same scenes or objects.

In this section, we will discuss the main edge detection techniques in the literature. These techniques are classified as Gradient-based and Gaussian-based methods. Figure 3.5 summarises these existing methods.

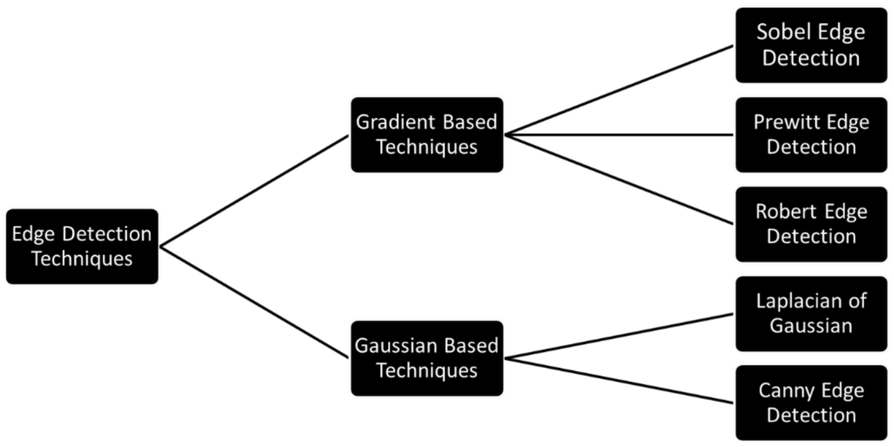

Figure 3.5: Classification of existing edge detection methods [204, 206, 211].

3.3.1.1 Gradient-based techniques

Gradient-based techniques employ small convolution masks to approximate either the first derivative or the second derivative of an image [204, 211, 215, 216]. They focus on the “edge enhancement” part of edge detection, with none or very little “smoothing”. A threshold is then applied to the output of these filters to identify the edge points. These filters, although easy to implement and generally with the advantage of speed over later edge detectors, provide very little control over smoothing and edge localisation, by means of which noise is reduced. Therefore, these filters are very noise-sensitive [204, 211, 215, 216].

The Sobel operator [204, 211, 215, 217] performs a 2-D spatial gradient measurement on an image. It uses a pair of 3×3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The Sobel masks are shown below (Figure 3.6):

-1	**0**	**+1**
-2	**0**	**+2**
-1	**0**	**+1**

$\boldsymbol{Gx}$

+1	**+2**	**+1**
0	**0**	**0**
-1	**-2**	**-1**

$\boldsymbol{G}$**y**

Figure 3.6: Masks used by the Sobel operator.

These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image in order to produce separate measurements of the gradient component in each orientation (call these Gx and Gy). These can then be combined to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:

$$|G| = \sqrt{Gx^2 + Gy^2} \tag{3.1}$$

Typically, an approximate magnitude is computed using:

$$|G| = |Gx| + |Gy| \tag{3.2}$$

which is much faster to compute. The angle of orientation of the edge (relative to the pixel grid) giving rise to the spatial gradient is given by:

$$\theta = \arctan\left(\frac{Gy}{Gx}\right) \tag{3.3}$$

The Sobel operator is computationally cheap but it has the disadvantage of poor edge detection in the presence of noise. The operator highlights the noisy areas as edges, especially when the points are as large as a pixel.

The Prewitt filter [204, 211, 215] is very similar to the Sobel filter. The 3x3 total convolution masks is used to detect gradient in the X, Y directions, as shown in Figure 3.7. The Prewitt filter is a fast method for edge detection. The difference with respect to the Sobel filter is the spectral response. It is suitable only for well-contrasted noiseless images.

-1	0	1
-1	0	1
-1	0	1

Gx

1	1	1
0	0	0
-1	-1	-1

Gy

Figure 3.7: Masks used by the Prewitt operator.

The Robert operator [204, 211, 215, 216] is similar to Sobel and Prewitt and performs two-dimensional gradient measurements on an image. Therefore, it highlights the regions which are related to high spatial frequency of edges. The operator contains a pair of 2×2 convolution masks. These masks are prepared so that they give maximal response to edges which are at 45 degrees extending to the grid of the pixel. The common mask is given by Gx and Gy, as shown in Figure 3.8.

1	0
0	-1

Gx

0	1
-1	0

Gy

Figure 3.8: Masks used by the Robert operator.

The kernels can be applied separately to the input image to produce separate measurements of the gradient component in each orientation (Gx and Gy). The gradient magnitude is given as:

$$|G| = \sqrt{Gx^2 + Gy^2} \tag{3.4}$$

Typically, an approximate magnitude is computed using:

$$|G| = |Gx| + |Gy| \tag{3.5}$$

This is much faster to compute. The angle of orientation of the edge giving rise to the spatial gradient is given by:

$$\theta = \arctan\left(\frac{Gy}{Gx}\right) - \frac{3\pi}{4} \tag{3.6}$$

The advantages of gradient-based methods are that they are relatively simple, quick, and easy to compute. However, they have the following disadvantages [204, 208, 211, 218]:

- Directional.
- Sensitive to noise, because many small local maxima will be generated by noise.
- Find only step-like edges.

- Corners are often missed due to the smallness of the1D gradient at the corners.
- The response to a step edge is across several pixels, so post-processing is needed for "edge thinning".
- The selection of an appropriate threshold is critical and always difficult.

3.3.1.2 Gaussian-based techniques

Gaussian filters are the most widely-used filters in image processing and are extremely useful as detectors for edge detection. It is proven that they play a significant role in biological vision, particularly in a human vision system [206, 211]. Gaussian-based edge detectors are developed based on several physiological observations and important properties of the Gaussian function that enable edge analysis to be performed in the scale space [206, 211].

Marr and Hildreth [219] were the pioneers who proposed an edge detector based on the Gaussian filter. Their method had been a very popular one, before Canny released his detector [206, 211]. They originally pointed out the fact that the variation of image intensity (i.e. edge) occurs at different levels. This implied the need to smooth filters with different scales, since a single filter cannot be optimal for all possible levels. They suggested the 2D Gaussian function, defined in the following, as the smoothing operator.

$$G_{\sigma}(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{3.7}$$

where σ is the standard deviation, and $(x,\ y)$ are the Cartesian coordinates of the image pixels. They showed that by applying Gaussian filters of different scales (i.e. σ) to an image, a set of images with different levels of smoothing will be obtained.

$$\hat{g}(x, y, \sigma) = G_\sigma(x, y) * g(x, y) \tag{3.8}$$

Then, to detect the edges in these images, they proposed to find the zero-crossings of their second derivatives. Marr and Hildreth achieved this by using the Laplacian of Gaussian (LOG) function as a filter. Since Laplacian is a scalar estimation of the second derivative, LOG is an orientation-independent filter (i.e. no information about the orientation) that breaks down at corners, curves, and at locations where image intensity function varies in a non-linear manner along an edge. As a result, it cannot detect edges at such positions. According to [219], the smoothing and differentiation operations can be implemented by a single operator consisting of the convolution of the image with the Laplacian of the Gaussian function. The final form of the filters, known as LOG with scale σ, which should be convolved with the image is as follows:

$$f_\sigma(x, y) = \nabla^2 G_\sigma = -\frac{1}{\pi\sigma^4}\left(1 - \frac{x^2 + y^2}{2\sigma^2}\right) exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{3.9}$$

There are advantages of the Gaussian filter that make it unique and very important in edge detection. The first concerns its output. It is proven that when an image is smoothed by a Gaussian filter, the existing zero-crossings (i.e. detected edges) disappear as they move from fine-to-coarse scale, but new ones are never created [206, 211, 220]. This unique property makes it possible to track zero-crossings (i.e. edges) over a range of scales, and allows them to be recovered at sufficiently small scales.

In spite of the unique features of the Gaussian function, the filter proposed by Marr and Hildreth had some deficiencies associated with the zero-crossing approach which is reliable only for the location of edges if they are well separated and the signal-to-noise ratio (SNR) in the image is high. It is shown that for ideal step and ramp edges, the location of the zero-crossing is exactly at the location of the edge. However, the location shifts from the true edge location for the finite-width staircase steps. This shift is a function of the standard deviation of the Gaussian. The other problem is the detection of false edges. The reason is that zero-crossings correspond to local maxima and minima in the first derivative of an image function, whereas only local maxima indicate the presence of real edges. LOG filtered images also suffer from the problem of missing edges (i.e. edges in the original image may not have corresponding edges in a filtered image).

Canny [221, 222] proposed a method that was widely considered to be the standard edge detection algorithm in the industry. In regard to regularisation explained in image smoothing, Canny saw the edge detection as an optimisation problem [206, 211, 223]. He considered three desirable criteria for any edge detector: good detection, good localisation, and only one response to a single edge. Then he developed the optimal filter by maximising the product of two expressions corresponding to two former criteria (i.e. good detection and localisation) while keeping the expression corresponding to uniqueness of the response constant and equal to a pre-defined value. The solution (i.e. optimal filter) was a rather complex exponential function, which by variations it could be well approximated by the first derivative of the Gaussian function. This implies the Gaussian function as the smoothing operator followed by the first derivative operator. Canny showed that for a 1D step edge, the derived optimal filter can be approximated by the first derivative of a Gaussian function with variance σ as follow:

$$f_\sigma(x) = \frac{dG_\sigma(x)}{dx} = -k\frac{x}{\sigma^2}\exp\left(-\frac{x^2}{\sigma^2}\right) \tag{3.10}$$

In 2D, Canny assumed that the image was affected by white noise, and proposed the use of two filters representing derivatives along the horizontal and vertical directions. In other words, the edge detection is performed by calculating the derivative along two directions of the image filtered by Gaussian function. The separability feature of the 2D Gaussian function allows us to decompose it into two 1D filters.

$$f_\sigma(x,y) = [f_\sigma(x) \times G_\sigma(y) \quad G_\sigma(x) \times f_\sigma(y)] \tag{3.11}$$

where $G_\sigma(.)$ and $f_\sigma(.)$ denotes the 1D Gaussian function and its derivative, respectively, and $f_\sigma(.,.)$ denotes 2D optimal filter. The filter [224] shows that the filtering can be applied first to columns (rows) and then to rows (columns), reducing the computational burden. The optimal filter has rather an orientation perpendicular to the direction of the detected edge. The method proposed by Canny can be used for developing filters dedicated to a specific and arbitrary edge profile. For step edges, Canny‘s optimal filter is similar to the LOG operator because the maxima in the output of a first derivative operator correspond to the zero-crossings in the Laplacian operator used by Marr and Hildreth.

Canny also proposed a scheme for combining the outputs from different scales. His strategy is fine-to-coarse and the method is called ‘feature synthesis’. It starts by marking all the edges detected by the smallest operators. It then takes the edges marked by the small operator in a specific direction and convolves them with a Gaussian normal to the edge direction of this operator so as to synthesise the large operator outputs. It then compares the actual operator outputs to the synthesised outputs. Additional edges are marked if the large operator detects a significantly greater number of edges than what is predicted by the synthesis. This process is then repeated to mark the edges from the second smallest scale that were not marked by the first, and then to mark the edges from the third scale that were not marked by either of the first two, and so on. In this way, it is possible to include edges that occur at different scales even if they do not spatially coincide [206, 211, 223].

Canny’s edge detector looks for local maxima over the first derivative of the filtered image. It uses adaptive thresholding with hysteresis to eliminate streaking of edge contours. Two thresholds are involved, with the lower threshold being used for edge elements belonging to edge segments already having points above the higher threshold. The thresholds are set according to the amount of noise in the image, which is determined by a noise estimation procedure.

The problem with Canny's edge detection is that his algorithm marks a point as an edge if its amplitude is larger than that of its neighbours without checking that the differences between this point and its neighbours are higher than what is expected for random noise. His technique causes the algorithm to be slightly more sensitive to weak edges, but it also makes it more susceptible to spurious and unstable boundaries wherever there is an insignificant change in intensity (e.g., on smoothly shaded objects and on blurred boundaries) [206, 211, 223]. Table 3.1 summarises the advantages and disadvantages of Gaussian-based techniques.

Table 3.1: Advantages and disadvantages of Gaussian-based methods [204, 208, 225].

Gaussian Based Techniques	**Advantages**	**Disadvantages**
(1) Laplacian of Gaussian	1) Due to the approximation of gradient magnitude, the cross operation detection of edges and their orientation is simple.	1) The edge magnitude degrades as noise increases due to detection of edges and their orientation.
	2) The characteristics are fixed in all directions.	2) At the corners and curves, malfunctioning are varies.
(2) Canny edge detection	1) Signal-to-noise ratio is improved.	1) Complex and time consuming computation.
	2) Better detection in noise condition.	2) False zero-crossing

In this sub-section, we have been investigated how edge detection techniques are utilised in image processing, which deal with high density pixel size images. In the next sub-section, we will study edge detection techniques in WSNs. One main challenge in applying these techniques in WSNs is the fact that sensor nodes might not be regularly placed and not placed with sufficient density. In particular, these techniques perform very well at high densities when each pixel contains a sensor node. A second challenge has to do with the notion of neighbourhood. In classical edge detection techniques, the neighbours of each pixel are the eight adjoining pixels; in the case of sensor nodes, the notion of neighbourhood is determined by the radio transmission power: each node can communicate locally with other nodes that lie within its effective radio range.

The inclusion of edge detection and determination capabilities from the image segmentation and processing field in the WSNs field will provide us with a mechanism for determining sensor nodes located at the edges of an event. This capability can reduce the communication and energy costs of event detection and extend the lifetime of the network by utilising only those sensors located at edges in the recognition procedure.

3.3.2 Edge detection in WSNs

The idea of edge detection has been utilised in different area of WSNs. In this section, we will investigate the utilising of edges in two different domains of

WSNs, namely, geometrically-representing sensed phenomena in WSN and visual sensor networks.

3.3.2.1 Geometrically representing sensed phenomena in WSN

The idea of reducing the overhead associated with data analysis via the geometric identification of sensed phenomena (event) within a WSN is broadly referenced in the literature as *edge detection* [226, 227]. A main advantage of identifying geometric representations of a sensed phenomenon is that it provides a more concise view than does the enumeration of all sensor nodes identifying a phenomenon or event, especially if the phenomenon is large [228]. A more concise view of sensed data can reduce the communication and energy costs of data analysis and extend the lifetime of the network. By geometrically identifying sensed phenomena, this also provides additional benefits such as the ability to map sensed phenomena onto known geographical features in the deployed region based on geometric shape and to coordinate sleep/wakeup cycles of sensor nodes contained within a sensed phenomena's identified perimeter [226].

Chintalapudi and Govindan [228] proposed three different approaches to identify the nodes within the edge of a sensed phenomenon inside a WSN, namely the statistical-based, the filter-based, and the classifier-based approaches. The authors define an edge as a region that intersects both the interior and exterior areas of an observed phenomenon. The authors demonstrate the techniques on networks with a single large-scale continuous phenomenon. Nowak and Mitra [229] discuss a technique for detecting an estimated boundary between two regions of relatively homogeneous sensed data. The technique takes advantage of hierarchical quad-trees (QTs) in order to identify small clusters that estimate the regions in which the boundary between two sets of sensors with differing sensor readings passes. This technique builds upon Chintalapudi and Govindan's results by considering the existence of multiple large-scale phenomena. Liao et al. [230] have proposed a technique, similar to that of the Chintalapudi and Govindan approaches, that not only identifies the nodes on the edge of a phenomenon, but also the nodes outside the phenomenon that border its edge.

Even with the edge detection of phenomena capabilities discussed above, these schemes involve computationally intensive (e.g. iterative) operations that are inappropriate for real-time sensory applications due to prolonged or inestimable learning cycles [230, 231].

Banerjee et al. [231] proposed a polynomial-based event region detection (PERD) scheme so that the occurrences of event and detection of the event boundary over the entire network could be efficiently recognised. In many applications in which the sensor readings admit a normal distribution within a bounded range, event recognition can be implemented by using a threshold-based scheme, which has a light computational overhead, rather than using fairly complicated schedules [116, 232]. PERD constructs a well-spread binary (each tree node having two children) query-tree (QT) covering the entire sensed region which produces faster dissemination of the information about the event boundary and its span of area, from any sensor to the sink. PERD can detect a single event or multiple events simultaneously. Detection of multiple events by extended PERD does not degrade the performance, making PERD scalable. For a single event scenario, direct PERD can be employed as the final polynomial corresponding to the event at the root can give all the data values in that event region. For a multiple event scenario, extended PERD does not give the polynomial corresponding to an event, thereby incurring a 6% increase in event area approximation error. However, this minute increase in percentage error is negligible compared to more than 33% savings in the communication overhead obtained with extended PERD. Similar to Nowak and Mitra approach in [229], PERD requires iterative operations for the detection of events and their boundaries which involve huge resource consumption, especially when implemented on large-scale networks [231].

Mallery and Medid [226] presented a distributed edge detection technique that identifies connected perimeters for sensed phenomena within WSN, without any unit disk communication graph assumptions. However, they assume that every sensor node knows its own position. Their approach is able to identify connected perimeters of convex shaped phenomena, irregularly shaped phenomena and phenomena that contain connectivity holes. The basis for their technique is for cooperating groups of location-aware nodes to identify their own outer perimeters using only connectivity information. Groups are comprised of connected nodes with similar sensed data values. Initially, each group that has not identified its own perimeter (i.e. a group constructed from nodes not yet bound by any identified perimeter, or captured) cooperatively identifies the group's convex hull in order to determine the nodes that must exist on the perimeter of the group. The well-known properties of convex hulls provide an intuitive correctness to their technique. The convex hull nodes then connect themselves together, using the paths within the original group of nodes. Since the process of connecting the identified convex hull nodes can leave some nodes uncaptured, the process is repeated on subgroups of uncaptured nodes until all nodes are captured by a perimeter. Once all nodes are captured, the perimeters are merged together to form a single connected perimeter for each group of nodes, one per sensed phenomena. Clearly, if sensed values are ignored, their technique is capable of identifying the outer boundary of an entire network. However, the proposed technique in [226] involves iterative operations and high communication costs in order to perform groups that are comprised of connected sensor nodes with similar sensed data values, which are not suitable for resource-constrained systems networks such as WSNs.

3.3.2.2 Visual sensor networks

Networks of visual sensors have recently emerged as a new type of sensor-based intelligent system, with performance and complexity challenges that go far beyond those of existing wireless sensor networks (WSN) [233]. For many years now, networks of cameras have been used for surveillance and security [233-235]. These networked cameras, unable to process any data locally, would send their individual streams to a centralised location and rely upon human vision and cognition to detect events and anomalies. Advances in computational technologies have allowed for much improvement in the hardware of these systems, such as better picture quality and motorisation capabilities [233]. Also, availability of low-cost CMOS cameras has created the opportunity to build low-cost Visual Sensor Network (VSN) platforms able to capture, process, and disseminate visual data collectively [233, 236]. However, only in the last few years have there been attempts to integrate the latest research developments in human and computer vision into current sensor technologies. The ultimate goal of VSN designers is to enable the cameras (nodes) to perform more advanced vision tasks locally, in a fully-automated way [233, 234]. When combined, these cameras would constitute an intelligent visual system capable of detecting not only objects within their range, but also the events created by these objects [233, 234].

A VSN consists of a group of nodes, each equipped with a low power, embedded processor, energy source, image sensor, and some type of transceiver for communication. These nodes must also be capable of communication with the network base station or sink where the data is collected and further processed for end-user consumption. The ability of these nodes to communicate with each other creates the possibility for event and anomaly detection over a group of nodes known as a cluster [233-235].

VSNs follow the three basic tenets of WSNs [233-235]: distributed sensing, low power hardware, and wireless networking. However, VSN nodes differ from typical WSN nodes in that they have the ability to perceive the environment in three dimensions rather than in one [233, 234]. Most of the sensing components on board of a WSN node can observe only single-dimension events. The image sensors on a VSN node contain photosensitive cells that provide two-dimensional data points, the typical flat picture humans observe on a display. The use of multiple nodes with varying perspectives, and knowledge of their positions, provides the third dimension, depth [233].

Vision is the most important functionality provided by VSNs [233-235]. Efficient and effective utilisation of visual data depends on the intelligent use of vision computing techniques, along with image compression and video coding [233, 234]. Because of energy efficiency considerations, VSN platforms are designed with limited hardware capabilities; hence, vision-computing in VSN platforms is highly constrained and very challenging. In the next paragraph, we provide an overview of coding and compression algorithms together with *in-network* processing techniques in the context of VSN platforms and applications.

Transform coding also known as intra-coding (e.g., Discrete Cosine Transform-DCT and Wavelet Transforms-DWT) is a method used for lossy compression of still images and video [233, 234]. Intra-coding typically achieves poor compression for video since it does not exploit temporal correlation although it is very robust [233, 234, 236]. For video compression, transform coding is combined with motion-compensated prediction (inter-coding). One of the most commonly used techniques is the block-based predictive motion compensation technique [233, 234, 236]. Inter-coding achieves high compression at the cost of high complexity. This means that one or more frames should be stored at the encoder and the decoder. Furthermore, predictive motion compensation [237] suffers from sensitivity to synchronisation between coder and decoder. Error-prone wireless channels often cause packet losses, thereby affecting the integrity of data and producing prediction mismatches. Thus, the use of predictive motion-compensated coding techniques requires powerful computation and ample storage capabilities, which are not well suited to VSN platforms [234]. A full-search block motion estimation algorithm incurs around 65,000 operations per pixel per second for a 30 fps (frames per second) video as reported in [237]. To illustrate the communication and computation energy dissipation terms, we use the example presented in [238]: processing with QCIF (176×144) resolution and 8-bit/pixel coding would cost the energy equivalent of transmission of approximately 4,400 Kb and processing with CIF (352×288) resolution and 8-bit/pixel coding would cost the energy equivalent of transmission of approximately 17,600 Kb.

Colour provides multiple measurements at a single pixel of the image [234, 236]. The 24-bit RGB encoding uses 3 bytes for each basic colour (red-green-blue) enabling $(2^8)^3$ colors. Other schemes use 15-bit RGB (5 bits for each color component) or 16-bit RGB with emphasised green sensitivity. YCbCr encoding, typically used in JPEG and MPEG coders, allows a better representation for image storage and transmission [233, 234]. The amount of data to be sent becomes larger as the color-coding technique becomes richer.

For example, a sensor using CIF (352×288) resolution with 512 KB FLASH memory can store 41.37 1-bit (black and white) frames, 5.17 8-bit colour frames or 2.58 16-bit colour frames. There is, thus, a clear trade-off between sensor capabilities (i.e., buffering, computation, and energy) and resolution, coding, compression, and video frame rate [233, 234, 236] (see Figure 3.9).

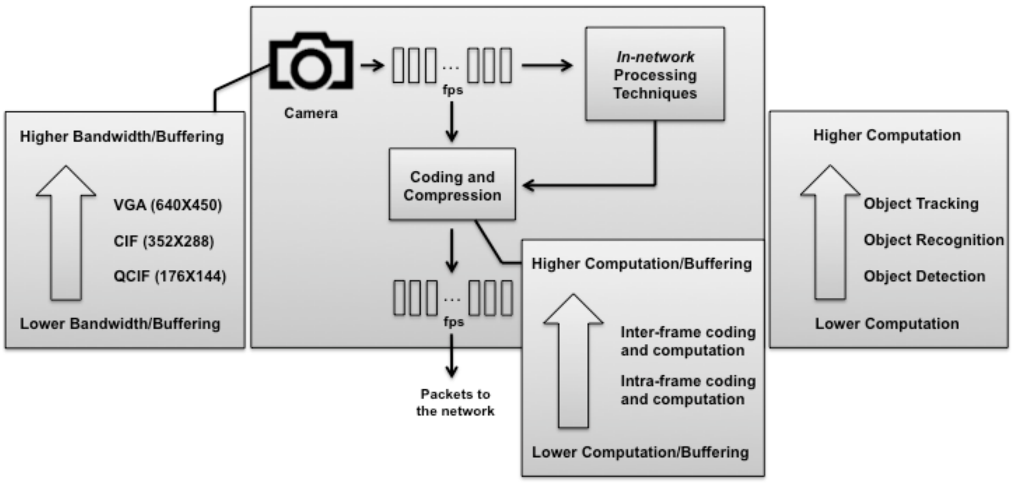
Camera
fps
In-network Processing Techniques
Higher Bandwidth/Buffering
VGA (640X450)
CIF (352X288)
QCIF (176X144)
Lower Bandwidth/Buffering
Coding and Compression
fps
Packets to the network
Higher Computation/Buffering
Inter-frame coding and computation
Intra-frame coding and computation
Lower Computation/Buffering
Higher Computation
Object Tracking
Object Recognition
Object Detection
Lower Computation

Figure 3.9: Computation trade-offs between sensor capabilities (i.e., buffering, computation, and energy) and resolution, coding, compression, and video frame rate in VSNs [234].

Data processing in VSN is more computationally intensive than scalar sensors [233, 234, 236]. That makes energy dissipation for video coding/compression an important feature to be analysed and researched. In scalar sensors, there is a clear objective in analysing the connectivity energy consumption (transmission and reception) of individual sensors and global network lifetime in terms of protocol energy performance [233, 234]. Nevertheless, visual sensors consume more data processing energy than do scalar sensors and, therefore, there is a need to optimise energy consumption for data processing in individual visual sensors [233, 234]. Hence, an analytical model [239] is introduced to characterise the relationship between encoding power consumption and its quality performance. The objective is to extend the traditional Rate-Distortion analysis to include energy constraints. Rate-Distortion and Time-Distortion trade-offs also are evaluated in CITRIC [240], where it is shown that CS Time-Distortion tends to become zero while JPEG needs a residual computation time of 70 ms per image for the lowest distortion performance. The application of power consumption and rate distortion analysis to video sensors is a promising research avenue since most of the current studies related to video coders in VSNs use intra- and inter-frame compression techniques [233, 234]. The next paragraph is dedicated to *in-network* vision computing mechanisms.

Techniques intended to manipulate image content thereby allowing the transmission of a reduced amount of information, are generally termed *in-network* processing techniques within the VSN domain [233, 234, 236]. Most of these techniques come from the vision-computing field. Vision-computing can reduce the amount of data to be sent to the sink [233-235]. Well-known visual computing techniques, such as object recognition or object tracking [241], can reduce the amount of raw data to be transmitted. Object/event detection can be done by using the identified features of the object. Object tracking can be done by means of localisation algorithms (e.g., active contour-based, feature-based, and model-based).

Vision-computing techniques implemented in VSN platforms are summarised in Table 3.2 [233-235]. It is important to mention that most of these techniques require powerful processors and large memory resources if they are not customised for implementation in VSN platforms [233, 234]. Vision-computing techniques optimised for VSN platforms are implemented, tested and reported for most of the VSN platforms. Here, similar to what we have done for compression and coding, a general overview of vision computing in VSN platforms is provided [233-235].

Most of the platforms utilise background subtraction, frame differencing, and edge detection as the main *in-network* processing techniques [233-235]. Background subtraction consists of extracting the object of interest from the image and removing the background. It must be robust against changes in illumination and should avoid detecting non-stationary background objects (moving leaves, rain, snow, etc.) [233, 234]. Moving average filters smooth backgrounds with illumination changes or moving objects. Frame differencing detects changes between two consecutive frames. Differencing the current frame with a background frame averaged over past frames adds robustness against illumination changes [234]. For example, Meerkats, Cyclops, MeshEye, MicrelEye, and CITRIC perform background subtraction and frame differencing as object detection mechanisms [233-235]. MicrelEye [242] takes a fixed background frame at the beginning and then performs pixel-by-pixel differencing between each frame and the background frame. Cyclops [243] uses moving average filters to smooth background changes. Firefly

Mosaic [244] uses a Gaussian Mixture Model (GMM) to separate foreground from background. GMM is able to detect objects even when the object is stationary.

Table 3.2: Compression/Coding techniques, *In-network* processing techniques, and Applications of VSN platforms [233, 234].

Platform	**Compression/ Coding techniques**	***In-network* processing (vision computing) techniques**	**Targeted applications**
Cyclops [243]	JPEG compression	Matrix operation libraries Edge detection (Sobel algorithm) Background subtraction (Object detection) Coordinate conversion	Object detection Hand posture recognition
MeshEye [245]	-	Background subtraction (Object detection) Stereo matching algorithm (Object location) Object acquisition and extraction	Distributed intelligent surveillance
Panoptes [246]	JPEG compression Differential JPEG compression	Motion detection filtering algorithm	Video surveillance
Meerkats [247]	JPEG compression	Object/event detection based on background subtraction or frame differencing	Moving body tracking
FireFly Mosaic [244]	JPEG compression	Frame differencing, Color tracking, Convolution, Edge detection Connected component analysis, Face detection, etc.	Assisted living application Home activity clustering
MicrelEye [242]	-	Background subtraction Feature extraction Classification algorithms	Image classification People detection
XYZ-ALOHA [248]	Address event representation	Histogram reconstruction Motion, Edge and Centroid detection	Pattern recognition (hand recognition, hand gesture recognition)
CITRIC [240]	Compressive sensing JPEG compression	Background subtraction Frame differencing Object acquisition and extraction Markov chain Monte Carlo data association	Single target tracking Camera localization using multi-target tracking
Vision Mote [249]	JPEG compression	-	Water conservation engineering

Edges could be considered as boundaries between dissimilar regions in an image. The computation of edges is fairly cheap and recognition of an object is easy since it provides strong visual clues; however, edge detection can be affected by the noise in an image [233, 234]. Although there are several different means of performing edge detection, generally they can be categorised into two classes [233, 234]: (a) gradient-based (e.g., Sobel algorithm) and (b) Laplacian-based. While the first one detects the edges by looking for the maximum and minimum in the first derivative of the image, the second one searches for zero crossings in the second derivative of the image in order to find edges. Cyclops supports Sobel libraries to perform edge detection [243]. Delay incurred by edge detection as a function of image size is investigated for XYZ platform [248]. For example, 8-bit Sobel edge detection lasts respectively 3,560 ms, 248 ms, 65 ms and 15 ms for 320×240, 256×64, 64×64 and 32×32 resolutions. However, the use of an FPGA block can decrease delay at reasonable energy costs [234, 248].

The results reported on energy dissipation and efficiency of the vision algorithms suggest that vision-computing is an essential component of the VSN paradigm [233-235]. It is shown in [247] that processing energy dissipation is no longer negligible and is comparable to (even higher than) the energy dissipation for networking. Meerkats [247] performs object detection based on background subtraction that simply detects motion taken between two snapshots at different times. Nodes wake up periodically, perform object detection, and send a wake-up message to neighbours that, in response, start taking pictures. This duty-cycle method consumes more batteries than does the use of the n-tier architecture, proposed in [250]. Power consumption and performance of object classification tasks are reported for MicrelEye [242]. The authors in [247] implement three hardware and software architectures (serial, parallelised and optimised) and demonstrate a power consumption of 430 mW at 5 fps for the serial implementation, 500 mW at 9 fps for the parallel implementation and 500 mW at 15 fps for the optimised implementation, showing that simple *in-network* processing tasks can be performed at moderate frame rates [233, 234]. Energy consumption will be one of the main challenges in the design of future *in-network* processing techniques [233-235].

By exploiting the spatial correlation between the images obtained by neighbouring nodes, power consumption can be reduced [233, 234]. Collaborative image coding [251] computes the differences between neighbour camera frames and local frames. Then, using edge detection (e.g. Sobel operator), the dominant features that are to be sent to the sink are computed. Before initiating the process, it is necessary to obtain the background in order to later reconstruct the image, where synchronisation of the cameras is the critical part. Experiments in [251] are performed using an Intel StrongARM SA 1100 as processor and LMX3162 as transceiver. An application in which, after an initial training phase, multiple overlapping cameras merge regions and collaborate in object activity detection is proposed for Firefly Mosaic platform [244]. However, the trade-off between the increase in signal processing for target detection and the transmission of data in lightweight sensors still remains as an active research issue [233, 234]. Collaborative image coding is a promising technique for VSNs; however, it also has drawbacks [233, 234]. In networks with scalar sensor nodes, the sensing area and the transmitting node's coverage area are basically the same [233, 234]. In visual sensor networks, sensing nodes are cameras with a FoV (Field of View) and a DoV (Depth of View) [233, 234]. FoV is the maximum volume visible from a camera, while DoV is the distance between the nearest and farthest objects that appear in an image [252]. Overlapping cameras can cover the same sensing area depending on the FoV and the DoV; however, if the transmission radius is not large enough, then VSN nodes with correlated data cannot directly reach each other [234]. In this situation, collaborative image coding needs specific sensor coordination algorithms in order to reach out of range nodes [234].

3.3.3 Discussion

This section discussed how edge detection techniques are used in both image processing and WSNs. In WSNs, the idea of edge detection has been used in two different domains, namely, geometrically representing sensed phenomena in WSN and VSNs. In the geometrically representing sensed phenomena in WSN field, researchers use edge detection techniques, generally not based on edge detection techniques of image processing, to determine nodes within the edge of sensed phenomenon (i.e. event). They utilise different edge detection techniques to determine the edges of an event which are, for example, based on a hierarchical boundary estimation algorithm or a tree-based boundary estimation algorithm. As discussed in sub-section 3.3.2.1, the existing edge detection approaches in WSNs are capable of edge detection of events. However, these approaches have one or more significant limitations, especially if implemented in resource-constrained networks such as WSNs. Iterative operations and high communication costs are the main issues faced in these existing schemes as such requirements involve huge resource consumption, especially when implemented in large-scale networks. The researchers in the field of VSNs use edge detection techniques of image processing; however, they are dealing with images data obtained from cameras which are types of data different from the data that concern this research.

Detecting edges of events by utilising a technique that is based on an edge detection method of image processing will offer a way to detect edges with a single-cycle process and requiring only neighbours' information, which reduces the computational and communication costs. Based on this section, the literature review and to the best of our knowledge, the scheme proposed in this chapter is the first pattern recognition scheme to utilise an edge determination mechanism based on the well-established edge detection technique of image segmentation in the recognition procedure, which can offer transformation-invariant detection capabilities.

The WSN nodes can detect events such as fire, gas leaks, chemical attacks, biological hazards etc., and report these to the sink for appropriate action. Detection of the event and the identification of the area affected by the event is a challenging task for Wireless Sensor Networks [253]. The event may be spread over an area of irregular shape. In WSNs, events can be described as shapes (i.e. objects or boundaries) but not visual shapes. They can be described as sensory-based shapes. The knowledge obtained from edge detection techniques in image processing and segmentation can be utilised to recognise events that produce geometric shapes on the basis of the sensor readings. For example, a forest fire will have an edge at the fire front based on a temperature reading; a flood will have an edge based on moisture or pressure; and a tornado will have an edge based on pressure.

The proposed scheme uses an edge detection mechanism to locate and determine the edges and the boundaries of an event. It utilises only those nodes located on the edges of an event in the recognition process, which minimises the number of sensor nodes and communications required for recognition. As mentioned previously, the second hypothesis in this chapter is that by describing events and patterns using their main edges and boundaries, it is possible to create an efficient recognition scheme that can detect transformations that may occur in these events and patterns. As discussed at the beginning of the section, edge detection techniques are generally classified as Gradient-based and Gaussian-based methods. Edge detection Gradient-based methods are relatively simple, quick, and easy in terms of computation, which make them suitable for use in resource-constrained environments like WSNs. On the other hand, Gaussian-based methods are usually complex and require time-consuming computations, which make them inappropriate to be utilised in WSNs.

3.4 Overview of MOHA Scheme

This section describes the proposed Macroscopic Object Heuristics Algorithm (MOHA) scheme. Part of this section has been published in [179, 254]. The main aim of developing the MOHA scheme is to provide efficient pattern recognition for WSNs while minimising resource consumption and network size, which is capable of detecting events, transformed, and noisy patterns. The scheme allows a WSN to collaborate and act as an associative memory in order to memorise (store) and recognise patterns. This is achieved by creating a network of GN arrays and allows these arrays to communicate in order to conclude one result. This will allow the network to process and compute information in a distributed *in-network* paradigm.

The goal of the MOHA network structure is to allow sensor nodes to exchange information about an incoming pattern for storing and recalling operations using an *in-network* processing paradigm. Two goals that the structure is attempting to achieve are low scheme complexity and high pattern recognition accuracy. To achieve the low complexity, the MOHA network structure adopts a GN scheme, which is well-known for its low computational, communicational, and time complexity. The main reason for this is the dependency on adjacency communications and computations in its structure for pattern recognition operations. Therefore, the MOHA network structure consists of multiple GN arrays where each array is assigned to process a sub-pattern of an incoming pattern. To achieve the high pattern recognition accuracy, the proposed scheme is based on coding techniques. Olshausen and Field [255] define coding technique as reproducing an incoming pattern and using a small number of active nodes for processing at any given time. They state that coding techniques can reduce the use of resources and minimise the complexity of incoming patterns so that they can be easily processed. To achieve the intended level of detection and to minimise resource consumption, the MOHA implements local adjacency-based relationships mechanism for coding purposes in a fully distributed and parallel manner that is capable of detecting events, noisy patterns, and patterns transformation such as translation, scaling (i.e. dilation) and rotation with minimal computational and communication requirements. The MOHA uses this mechanism to locate and determine the edges and the boundaries of an incoming pattern. The MOHA utilises only those nodes located on the edges of a pattern in the recognition process, which minimises the number of sensor nodes required for recognition. An edge node is determined, locally and only once, based on the relationship between its neighbours' values. A detailed discussion on how the proposed scheme can determines edge nodes will be provided in sub-section 3.4.2.1. These edge nodes report their location values to the base station in order to obtain a final decision about an incoming pattern. The hypothesis behind this approach is that patterns can be efficiently recognised based on their boundary information. Since these boundaries are detected by local nodes' computations, the number of communications and amount of resources required is minimised.

3.4.1 MOHA structure

As seen in Figure 3.10, the MOHA scheme consists of two main parts: the MOHA network and the stimulator and interpreter (S&I). S&I is an external computational node that is responsible for organising the complete recognition procedures and to determine whether to memorise or recall the input pattern, which has been proposed by [164]. The two components, presented in Figure 3.10, communicate with each other in order to conclude one decision within a predictable learning duration. The S&I sends commands to the network and the network replies with edge nodes' locations. The S&I receives the edge nodes' locations and utilises these nodes' location to determine the final decision about an incoming pattern. A command that is sent by the S&I tells network nodes whether to memorise (store) a pattern or recall (search for) it. The command will also determine the method of obtaining the pattern (i.e. sense environment).

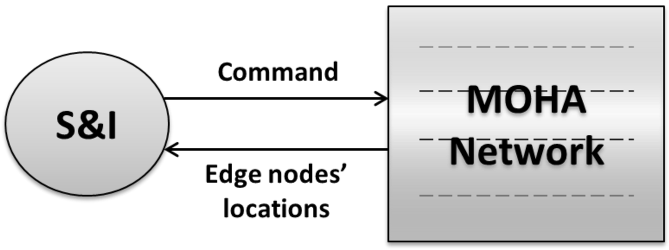

Figure 3.10: The two main components of the MOHA scheme.

Moreover, before discussing the rules and the functions of each part of the MOHA scheme in details, we must define what is a pattern in this research:

Definition 3.1: (Pattern) Given a set of possible values $v = \{a_1, a_2, \dots, a_v\}, a_i \in N$, a pattern is a set of elements that represent sensory information (e.g. describing an event) that can be send from the S&I to each node in the network or sensed by a network's nodes and can be described as follows:

$$P = \{p_1, p_2, \dots, p_S\}, \qquad p_i \in v, S \in \mathbb{N} \tag{3.12}$$

where p_i is the i^{th} element of the pattern and S is the number of elements and is called the pattern size.

3.4.2 MOHA network

The aim of the network structure is to have low pattern recognition scheme complexity, provide parallel processing, provide efficient recognition, and have a predetermined learning and recognition cycle duration. A low complexity scheme is achieved with a fully distributed structure that allows sensor nodes in the network to communicate only with adjacent nodes. The MOHA's structure allows each sensor node to communicate with four adjacent nodes to determine whether it represents a pattern boundary (i.e. edge). In the MOHA scheme, only edge nodes' information is utilised for pattern recognition. This minimises the number of sensor nodes required for recognition. Efficient recognition is provided by describing the patterns' boundaries in order to provide a transformation-invariant recognition feature to the scheme. By using only edge nodes' information in pattern recognition, this allows the detection of a particular pattern's boundaries by any node in the network. In other words, the detection of any desired pattern's boundary is not associated with static nodes. Instead, any node in the network is expected to be able to determine whether it represents a pattern boundary for such boundary information. By using specific steps for determining edge nodes and reporting their locations, The MOHA will have a single learning and recognition time cycle that can be predicted and estimated. Once all edge nodes' locations have been delivered to the S&I, the MOHA does not need further information from sensor nodes in order to declare the detection of a specific pattern (i.e. event). This feature reduces the need for communications between S&I and participant nodes in the network. Figure 3.11 shows that communication within the MOHA scheme occurs in a single-cycle environment, wherein each pattern is passed via the network only once.

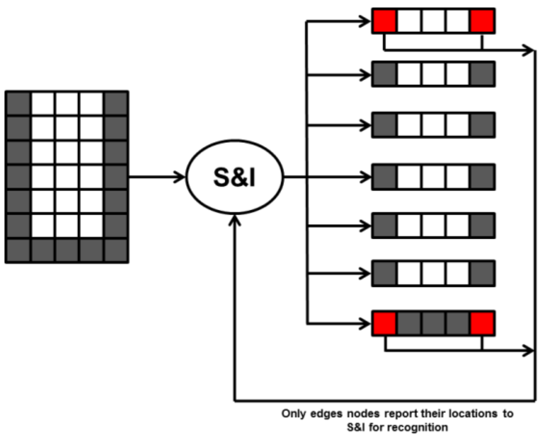
S&I
Only edges nodes report their locations to S&I for recognition

Figure 3.11: MOHA framework for pattern recognition.

The MOHA network consists of a set of GN networks. A GN network in the MOHA network structure is called a *row* and each row consists of a set of nodes that communicate with each other using *the node's exchange communications*, as described below. In the MOHA scheme, based on the received data from S&I or sensed data, each node is activated or de-activated according to activation criteria (e.g. high temperature or high pressure). Only activated nodes participate in *the node's exchange communications*. An active node can be determined by examining its received value. If a node's value complies with certain user-defined conditions, it becomes activated. One of these conditions is the attainment of a certain threshold. This is in order to maintain a limited use of node resources and to reduce the detection time by limiting the number of communicating nodes. An active node can be formally defined as follows:

Definition 3.2: (Active node) Given a MOHA node $N_i^{r_m}$ that is assigned to a value $a \in P$, where $P = \{p_1, p_2, \dots, p_S\}, p_i \in v$ is the incoming pattern, $N_i^{r_m}$ is an active node if $a \geq \varphi$.

where i is the node's position in its row, m is the node's row number, p_i is the i^{th} element of the pattern P, v is the set of possible values, and φ is the node activation threshold.

An activation of a node triggers the start of an exchange communication process for that node with its adjacent neighbours.

Definition 3.3: (Network row) A MOHA network row (*Row*) is a GN network that consists of a set of nodes where each node communicates with its adjacent nodes in the same row and in the higher and lower rows. An MOHA row can be described as follows:

$$Row_m = \{N_1^{r_m}, N_2^{r_m}, \dots, N_i^{r_m}\}, \qquad i \in \mathbb{N} \tag{3.13}$$

where $N_j^{r_m}$ is the j^{th} node in Row_m. Each node $N_i^{r_m}$ is responsible for sensing or receiving one element of an incoming pattern such that $P_{p,N}: p_i \rightarrow N_i^{r_m}, p \in v, i \in \mathbb{N}$, $P_{p,N}$ is the assignment of an incoming pattern's elements to the nodes (*N*s) using Definition 3.1.

The network row of sensor nodes is derived from a composition of interconnected GNs. The size of the row depends on the sub-pattern size utilised in the system and the number of distinct elements in the sub-pattern. Hence, in order to determine the size of each row, we must consider the number of GNs N_{gn} required for a sub-pattern with size S_{sb} and v different element (possible values) as given by the following equation:

$$N_{gn} = v \times S_{sb} \tag{3.14}$$

However, the MOHA scheme uses a multi-value approach since each node is capable of handling different element values. As a result, the number of nodes (N_{node}) required for a row (also a row size) is equal to the size of the sub-pattern S_{sb}, as given by the following equation:

$$N_{node} = S_{sb} \tag{3.15}$$

The communications between active nodes in each row can be defined as follows:

Definition 3.4: (The node exchange communications) Given a MOHA network row (*Row*) that consists of *m* nodes, exchange communications of an active node are mainly four direct connections between a node and its direct neighbours (adjacent), rather than two as in the simple GN scheme, illustrated in Figures 3.12 and 3.13. The four adjacent nodes are two nodes in the same row: the *previous* (p) and the *next* (n) nodes (like GN) and two extra nodes: one being the adjacent node in the next higher row and another one being the adjacent node in the lower row, in the form: $N_i^{r_m} \rightarrow N_{i-1}^{r_m}: v, N_i^{r_m} \rightarrow N_{i+1}^{r_m}: v, N_i^{r_m} \rightarrow N_i^{r_{m+1}}: v$, and $N_i^{r_m} \rightarrow N_i^{r_{m-1}}: v$ respectively, where $N_i^{r_m}$ is the communicating (activate) node in Row_m with a position number i and v is the value assigned to the $N_i^{r_m}$.

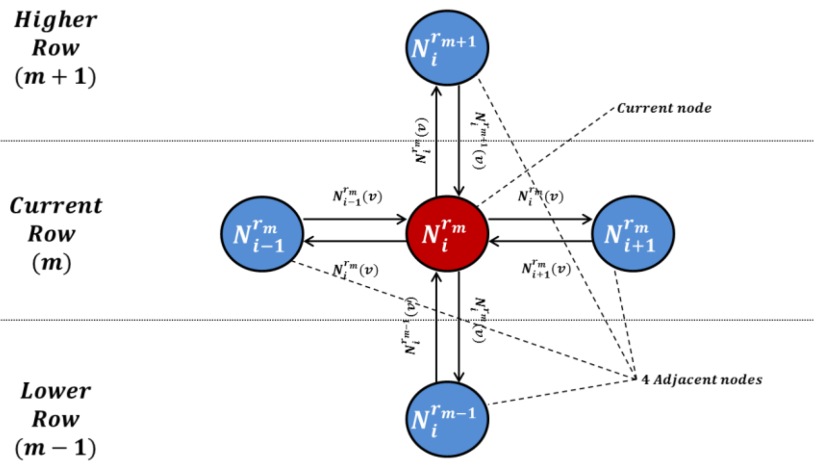

Figure 3.12: An active sensor node and its neighbours performing a node's exchange communications.

Exchanging the values with two extra adjacent nodes (rather than only two in GN) will enable the proposed scheme to recognise transformed patterns, especially the rotated ones, as it will obtain a better description about the incoming pattern and its distribution. However, in the MOHA scheme, the actual number of adjacent nodes and exchange communications of an active node varies from two to four depending on its location or position in the row and whether the row is located on the bottom, the middle, or the top of the network. The following relations describe the node's exchange communications between active nodes within each MOHA row, see Figure 3.13:

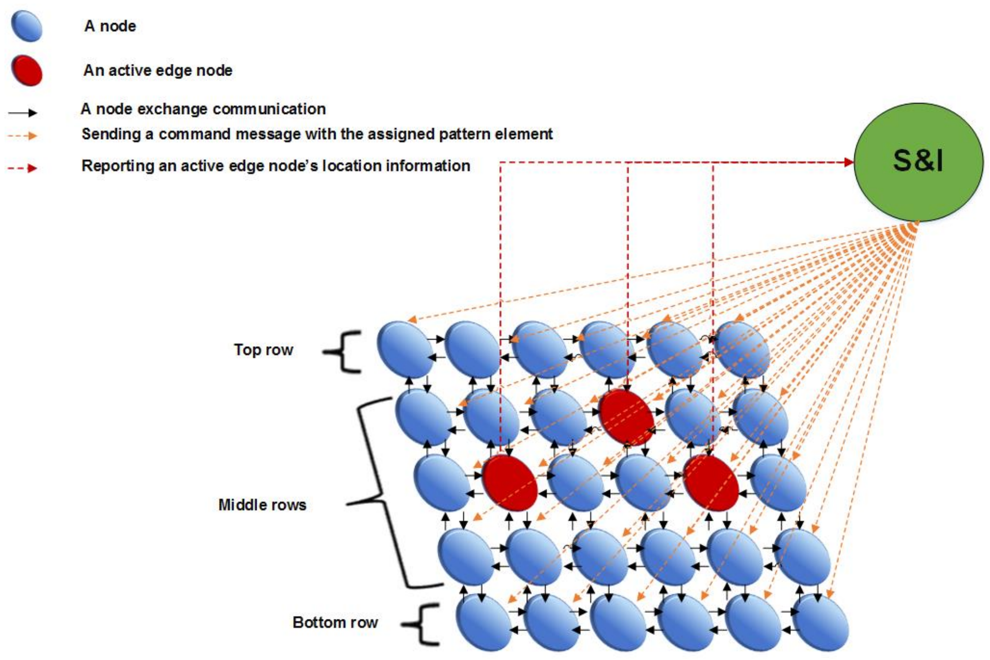
A node
An active edge node
A node exchange communication
Sending a command message with the assigned pattern element
Reporting an active edge node's location information
S&I
Top row
Middle rows
Bottom row

Figure 3.13: The communications between active nodes at different network rows.

- **Bottom row:** For active nodes at the edge of the bottom row, the number of communication messages exchanged is equal to two: one communication occurs between a node from the preceding or the succeeding columns (the *previous* (*p*) node or the *next* (*n*) node in the same row) and the other communication happens with the adjacent node at the next higher row. For non-edge active nodes, the communication is required between adjacent node in both the preceding and the succeeding columns, as well as the adjacent node in the next higher row. In this context, the number of messages exchanged is three. The cumulative communication costs involved for each input recognition process for all nodes in the bottom row, based on the assumption that all the nodes are activated, is derived using the following equation:

$$N_{r_0}^{msg} = 3(S_{sb} - 2) + 4 \tag{3.16}$$

where $N_{r_0}^{msg}$ is the total number of exchanged communications in the bottom row (r_0) and S_{sb} is the size of the sub-pattern that is assigned to this row.

- **Middle rows:** For active nodes at the edge of the middle rows, the number of communications taking place is equal to three: one communication occurs with a node from the preceding or the succeeding columns and another two communications happen with the adjacent nodes in the higher and lower rows. For non-edge active nodes, communication is required between adjacent nodes in both the preceding and the succeeding columns, as well as the adjacent node at the next higher row and the adjacent node in the lower row. In this context, the number of exchanged messages is four. The cumulative communication costs involved for each input recognition process for all nodes in one middle row, based on the assumption that all the nodes are activated, is derived through the following equation:

$$N_{r_i}^{msg} = 4(S_{sb} - 2) + 6 \tag{3.17}$$

where $N_{r_i}^{msg}$ is the total number of exchanged communications in the i^{th} row (r_i) and S_{sb} is the size of the sub-pattern that is assigned to this row.

Equation 3.18 presents the cumulative communication costs for all nodes in the all middle rows based on the assumption that all the nodes are activated:

$$N_{r_i^{total}}^{msg} = \sum_{i=1}^{top-1}(4(S_{sb} - 2) + 6) \tag{3.18}$$

where $N_{r_i^{total}}^{msg}$ is the total number of exchanged communications in all middle rows (r_i^{total}) and S_{sb} is the size of the sub-pattern that is assigned to these rows based on the assumption that all the middle rows assigned with the same S_{sb}.

- **Top row:** For active nodes at the edge of the top row, the number of communication messages exchanged is equal to two: one communication occurs with a node from the preceding or the succeeding columns and another communication happens with the adjacent node in the lower row. For non-edge active nodes, a communication is required between adjacent nodes in both the preceding and the succeeding columns, as well as the adjacent node in the lower row. In this context, the number of messages exchanged is three. As a result, similar to the bottom row, the cumulative communication costs involved for each input recognition process for all nodes in the top row, based on the assumption that all the nodes are activated, is derived from the following equation:

$$N_{r_{top}}^{msg} = 3(S_{sb} - 2) + 4 \qquad (3.19)$$

where $N_{r_{top}}^{msg}$ is the total number of exchanged communications in the top row (r_{top}) and S_{sb} is the size of the sub-pattern that is assigned to this row.

The MOHA network is designed in a multi-row structure containing multiple rows. The multiple rows in the network structure allow the parallel processing and information exchange of incoming data. This is achieved by allowing each row to perform a set of recognition operations on a sub-pattern in parallel with other rows. In other words, the MOHA adds a clustering mechanism in pattern recognition, by dividing and distributing patterns into sub-patterns and assigning each sub-pattern to a row. Furthermore, this structure enables the network to deal with multi-dimensional data types as each row will be assigned to process one dimension.

In order for the MOHA scheme to perform pattern recognition, the network topology must be generated or deployed. The network deployment process involves the construction of a collection of MOHA rows. The size of the MOHA row depends on the sub-pattern size utilised in the system (determined according to the system user or the WSN application requirements). Given an incoming pattern $P = \{p_1, p_2, \dots, p_S\}$ of size S_p, and sub-pattern length (size) S_{sb}, the number of MOHA rows N_{Row} that are to be deployed is determined by following equation:

$$N_{Row} = \frac{S_p}{S_{sb}}, \quad S_{sb} \leq S_p \tag{3.20}$$

The deployment of the network begins by implementing a number of nodes in the first row equal to the sub-pattern size (S_{sb}), according to Equation 3.15. Then, start implementing the second row nodes (equal to S_{sb}) and so on until all nodes in all rows have been deployed in the network. The MOHA network contains many sensor nodes distributed in the field of interest in a certain multi-row manner (see Figure 3.14). It is important to highlight that node deployment in this sub-section is a logical deployment method. In other words, deployment can be implemented by assigning each node to its row and its position in this row. These numbers will be used to define the tasks that each node will perform, as will be described in the network operations sub-section later in this chapter. Algorithm 3.1 depicts the network deployment process.

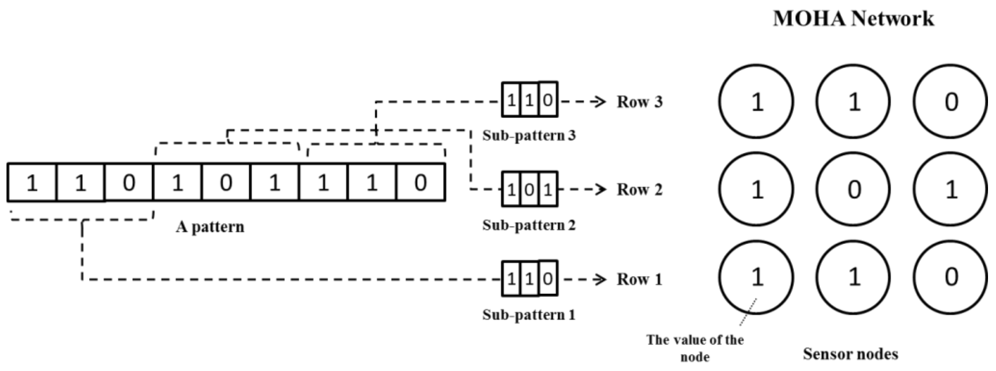

Figure 3.14: An example of MOHA network consisting of 3 rows and assigned with a 9-bit size pattern.

Algorithm 3.1: MOHA network deployment

```
1. NetworkSize = PatternSize
2. RowNumber = 1
3. RowSize = SubpatternSize
4. DeployedNode = 0
5. While (NetworkSize > 0)
6.    Deploy a node in Row(RowNumber)
7.    NetworkSize--
8.    DeployedNode++
9.    If (DeployedNode ≥ RowSize)
10.         RowNumber++
11.   End if
12. End While
```

One of the aims of providing the MOHA network with a multi-row structure is to offer each node in the network 4 adjacent nodes with which to exchange information. This enables a node to have an extended view of the incoming pattern, which allows it to determine whether or not it represents a pattern boundary (i.e. edge). An activation of a node triggers the start of an exchange communication process for that node with its adjacent neighbours, as described in Definition 3.4. After receiving information from all adjacent nodes, the activated node determines whether it represents a pattern edge according to Equation 3.29. Based on the resulting node type, it either de-activates or goes for the second level of activation called 'edge node activation' (active edge node), which can be described as follows:

Definition 3.5: (the active edge node) Given an active node $N_i^{r_m}$ and its four adjacent nodes $N_{i-1}^{r_m}$, $N_{i+1}^{r_m}$, $N_i^{r_{m+1}}$, and $N_i^{r_{m-1}}$, the node will be activated as an edge node if its node type is $edge$ ($node_{type} = edge$).

where i is the node's position in its row, m is the node's row number, and $node_{type}$ is the node's type, which is determined according to Equation 3.29. The procedure used to determine a node's type is explained in detail in the next sub-section. Edge-activated nodes are the only nodes that participate in the reporting communication described in Definition 3.6.

In the MOHA scheme, only activated edge nodes' information are used for pattern recognition. Thus, all edge nodes in the network are required to send their information (i.e. location values) to the S&I for further analysis and recognition. Here, we assume that every sensor node in the network knows its location in terms of (*x*,*y*) coordinate in space (i.e. WSN's field). Sensor nodes can also use their row numbers (*x*) and their position numbers (*y*) in the network as their locations. The communications between the MOHA network's edge nodes and the S&I are called *report communications* and can be described as follows:

Definition 3.6: (Report communications) Given a MOHA network that consists of *m* nodes, a report communication of an active edge node is the message (connection) between an edge node in the network and the S&I that contains the edge node's location information (i.e. its location in terms of (*x*,*y*) coordinate in space), in the form: $EN_i^{r_m} \rightarrow SI: \{x, y\}$.

where $EN_i^{r_m}$ is the communicating (activate) edge node in a row number m with a position number i and SI is the stimulator and interpreter (S&I). Figure

3.11 shows the report communications occurring between four edge-activated nodes and the S&I.

In *report communications*, in regards to the communication cost, the total number of messages communicated from edge nodes to the S&I, $N_{edge \to SI}^{msg}$ is equivalent to the number of activated edge nodes, N_{edge}, as given by the following equation:

$$N_{edge \to SI}^{msg} = N_{edge} \tag{3.21}$$

3.4.2.1 The activated edge nodes determination (pattern edge search)

The MOHA scheme represents patterns of events in terms of compositions of values, which are the sequence numbers of the activated edge nodes and the MOHA value (MV) that represents the relationship between these activated edge nodes. The main goals of the proposed approach are to reduce the overall computational complexity of the scheme recognition process and to enable the detection of pattern transformation (e.g. rotation, displaced positions (translation), and scaling (dilation)). To achieve these goals, the MOHA scheme searches for edges in the pattern's data domain to determine its boundaries. We assume that sensor nodes are deployed in a grid-like structure in the field of interest to obtain sensory information. As been discussed in sub-section 3.3.3, the proposed scheme will utilise an edge detection Gradient-based mechanism to locate and determine the edges and the boundaries of an event. The main reason for this choice is that edge detection Gradient-based methods are relatively simple, quick, and easy in terms of computation, which make them suitable to be used in resource-constrained environments like WSNs. To be more specific, the proposed scheme will utilise the Sobel edge detection operator which is one of the widely-used, Gradient-based methods.

Before discussing the proposed new and current technique for the activated edge nodes determination that is based on an edge detection method of image processing, the proposed scheme's initial technique for the activated edge nodes determination will be presented, which has been published in [179].

The initial technique for the activated edge nodes determination

The previous technique had been proposed at the beginning of our PhD research period as a result of our goal to create a technique that is simple, light, and fast for edge nodes determination, which makes it a suitable technique for resource-constrained WSNs.

In the initial technique, an active node type is determined according to its own value and the values received from its adjacent nodes using the node's exchange communications (see Definition 3.4). The node type is $edge$ when a node's value ($N_i^{r_m}(v)$) is equal to the values of only two adjacent nodes, the value of the next node in its row ($N_{i+1}^{r_m}(v)$) is not equal to the value of the previous node in its row ($N_{i-1}^{r_m}(v)$), and the value of the adjacent node in the next higher row ($N_i^{r_{m+1}}(v)$) is not equal to the value of the adjacent node in the lower row ($N_i^{r_{m-1}}(v)$), or when a node's value is equal to the value of only one adjacent node; otherwise, the node type will be $none$ and will be deactivated.

Now, we will present several examples which cover all types of nodes using the initial approach for determining the activated edge nodes. For example, suppose that we have a binary pattern (100011011) of 9 elements that has to be divided into 3 sub-patterns of 3 elements and each sub-pattern is assigned to a different MOHA row as in Figure 3.14. In this example, the middle element in the pattern (assigned to the middle node in the second row) is considered to be an $edge$ as it its value is equal to the values of only two adjacent elements (i.e. nodes). In another example, for the pattern (100010001), the middle element in the pattern is described as $none$, as its value does not equal the values of any adjacent elements (i.e. nodes).

The main problem with this technique to determine the activated edge nodes is that it can correctly determine activated edge nodes only in binary patterns, which is not practical for events detection and a variety of WSN applications. It is worth noting here that all of the simulations and tests conducted for this thesis are based on the new technique for determining the activated edge nodes, which will be discussed in the next paragraph.

The current technique for the activated edge nodes determination

Our new and more practical technique for determining the activated edge nodes is based on an edge detection method of image processing, which utilises an edge detection Gradient-based mechanism to locate and determine the edges and the boundaries of an event. The Sobel operator consists of a pair of 3×3 convolution masks (kernels), one estimating the gradient in the x-direction (Gx) and the other estimating the gradient in the y-direction (Gy), as shown in Figure 3.6. The kernels can be combined to find the absolute magnitude of the gradient at each sub-pattern node's value. Consider a 3×3 neighbourhood of a certain node, say N_c, with intensity values (e.g. temperature or pressure readings) as shown in Figure 3.15. The gradient magnitude of N_c can be calculated using the following equations [202, 207, 256]:

$$|G(N_c)| = \sqrt{Gx(N_c)^2 + Gy(N_c)^2} \tag{3.22}$$

where,

$$Gx(N_c) = \big(N_3(v) + 2N_4(v) + N_5(v)\big) - \big(N_1(v) + 2N_8(v) + N_7(v)\big) \tag{3.23}$$

$$Gy(N_c) = \big(N_1(v) + 2N_2(v) + N_3(v)\big) - \big(N_7(v) + 2N_6(v) + N_5(v)\big) \tag{3.24}$$

Typically, in real implementation, an approximate magnitude of N_c is computed using the following equation:

$$|G(N_c)| = |Gx(N_c)| + |Gy(N_c)| \tag{3.25}$$

which is much faster to compute.

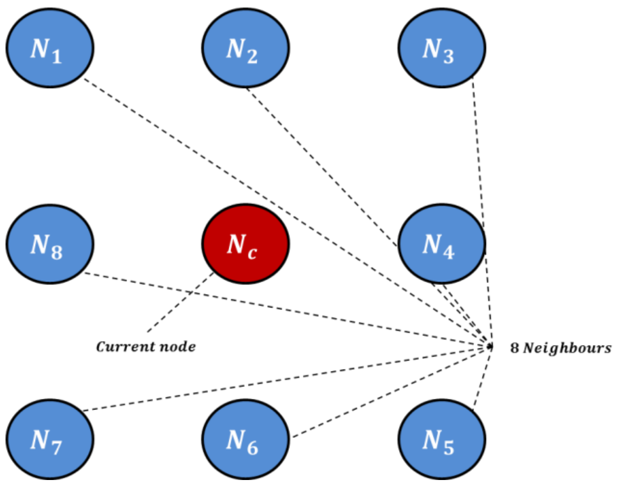

Figure 3.15: A particular node and its eight neighbours.

In the MOHA scheme, there are typically only 4 adjacent nodes (i.e. neighbours) for every sensor node in the network. Hence, the Sobel masks cannot be directly applied, which means that Equations 3.22-3.25 cannot be used directly for determining whether a sensor node lies on an edge of an event or not. To resolve this issue, Divya and Bhaskar [257] suggest that in a case where nodes may not have neighbours that fall into a particular sector of the mask, the values for nodes in those sectors are assigned the value of the node that is computing the edge, which is the central node. This will eliminate the effects of these sectors on the edge determination process.

Proposition 3.1: Given an active node $N_i^{r_m}$ in MOHA network and its four adjacent nodes $N_{i-1}^{r_m}$, $N_{i+1}^{r_m}$, $N_i^{r_{m+1}}$, and $N_i^{r_{m-1}}$, the gradient magnitude in the x-direction (Gx) of the active node is calculated as follows:

$$Gx\left(N_i^{r_m}\right) = 2N_{i+1}^{r_m}(v) - 2N_{i-1}^{r_m}(v) \tag{3.26}$$

Proof: From Equation 3.23, the gradient magnitude in the x-direction (Gx) of a certain node N_c is calculated as $Gx(N_c) = \left(N_3(v) + 2N_4(v) + N_5(v)\right) - \left(N_1(v) + 2N_8(v) + N_7(v)\right)$, where $N_i(v)$ is a value of a neighbour node in sector i. Also, [257] suggests assigning the value of the central node that is computing the edge to sectors of the mask that do not have nodes located on them.

From Equation 3.23:

$Gx(N_c) = \left(N_3(v) + 2N_4(v) + N_5(v)\right) - \left(N_1(v) + 2N_8(v) + N_7(v)\right)$

After applying the [257] suggestion $N_c(v) = N_1(v), N_3(v), N_5(v), and\ N_7(v)$:

$Gx(N_c) = \big(N_c(v) + 2N_4(v) + N_c(v)\big) - \big(N_c(v) + 2N_8(v) + N_c(v)\big)$

$Gx(N_c) = N_c(v) + 2N_4(v) + N_c(v) - N_c(v) - 2N_8(v) - N_c(v)$

$Gx(N_c) = 2N_4(v) - 2N_8(v)$

After using the MOHA scheme symbols:

$Gx\big(N_i^{r_m}\big) = 2N_{i+1}^{r_m}(v) - 2N_{i-1}^{r_m}(v)$

The gradient magnitude in the y-direction (Gy) of the active node in MOHA network is calculated as follows:

$$Gy\big(N_i^{r_m}\big) = 2N_i^{r_{m+1}}(v) - 2N_i^{r_{m-1}}(v) \qquad (3.27)$$

This equation can be proofed in the same way that the $Gx\big(N_i^{r_m}\big)$ is proofed in Proposition 3.1. Every active sensor node in the MOHA network determines the values of both x and y gradients by using Equations 3.26 and 3.27. Then, it can calculate the magnitude of its gradient $\left|G(N_i^{r_m})\right|$ by utilising Equation 3.25.

Also, we can combine the previous Equations 3.25, 3.26, and 3.27 into one single equation in order to calculate the gradient magnitude of an active node $\left|G(N_i^{r_m})\right|$ in a single step:

$$\left|G(N_i^{r_m})\right| = \left|2N_{i+1}^{r_m}(v) - 2N_{i-1}^{r_m}(v)\right| + \left|2N_i^{r_{m+1}}(v) - 2N_i^{r_{m-1}}(v)\right| \quad (3.28)$$

If the gradient magnitude of the active node is greater than a certain threshold value (φ), which is called the threshold value for edge determination, the sensor node determines and activates itself as an edge node ($node_{type} = edge$); otherwise, the node type will be $none$ and will be deactivated.

The active node type determination can be described as a function of the values of the node and its adjacent nodes in the form of $node_{type} = f(N_i^{r_m}(v), N_{i+1}^{r_m}(v), N_i^{r_{m+1}}(v), N_{i-1}^{r_m}(v), N_i^{r_{m-1}}(v))$. This relationship can be described as the following piecewise function:

$$node_{type} = \begin{cases} edge, & if\left|G(N_i^{r_m})\right| > \varphi \\ none, & Otherwise \end{cases} \tag{3.29}$$

We now give several examples which cover all types of nodes. Consider a pattern $P_1 = \{300,400,350,400,300,200,350,300,400\}$ of 9 elements that has be divided into 3 sub-patterns of 3 elements and each sub-pattern is assigned to a different MOHA row (see Figure 3.14) and the threshold value for edge determination is set to 500 ($\varphi = 500$). In this example, the middle element in the pattern (assigned to the middle node in the second row) is considered an $edge$ as its gradient magnitude is equal to 600, which is greater than the threshold value ($\left|G(N_i^{r_m})\right| > \varphi$). In another example, for the pattern $P_2 = \{350,400,500,400,350,400,300,200,250\}$, the middle element in the pattern is considered a $none$, as its gradient magnitude is equal to 400, which is less than the threshold value ($\left|G(N_i^{r_m})\right| < \varphi$).

3.4.3 MOHA network operations

Figure 3.16 shows the steps that each node in the network performs in the learning process. These steps are as follows:

1. **Receive command:** The node receives the broadcasted command from S&I that contains the operation (memorise or recall) and the

method for obtaining the pattern element (direct receive or sense). Further details about this command message will be presented in Section 3.5.

2. **Obtain pattern element:** Based on the command message, each node begins to receive its assigned pattern element p_i. Each node sets its value (*v*) according to the pattern element received. If a node's value complies with certain user-defined conditions (e.g. reaching a certain threshold), it becomes activated, according to Definition 3.2.
3. **Exchange communications:** Each activated node performs exchange communications with its neighbouring nodes, as described in Definition 3.4. Then, the activated node determines whether it represents a pattern edge according to Equation 3.29. Based on the equation result, it is either de-activated or activated as an edge node, as described in Definition 3.5.
4. **Report communications:** Each activated edge node in the network reports its location information (i.e. its location in terms of (*x*,*y*) coordinate in space) to the S&I, as described in Definition 3.6.

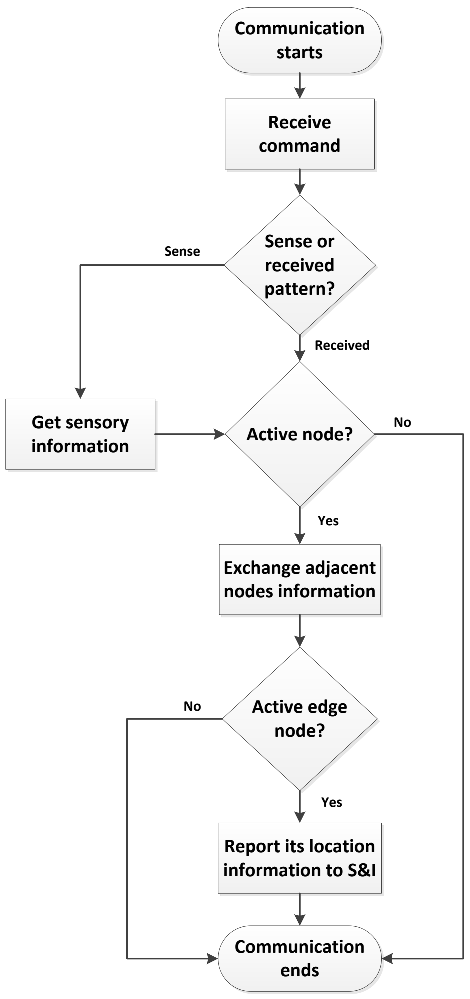

Figure 3.16: MOHA network node operations.

3.4.4 Stimulator and interpreter (S&I)

This component is responsible for sending commands to the MOHA network, receiving the locations of the activated edge nodes and making the final decision about an incoming pattern. The pattern-obtaining method that is utilised by MOHA scheme will be discussed in Section 3.5. MOHA network nodes process the S&I command and respond with the locations of the activated edge nodes. The S&I uses this location information, containing the locations of the activated edge nodes in the network in terms of (*x*,*y*) coordinate in space, to calculate the MOHA value (MV) of a sensed or received pattern. In fact, this location information (of the activated edge nodes) indicates the pattern's boundaries (i.e. a pattern edges). The S&I uses the calculated MV of a sensed or received pattern with the number of activated edge nodes to memorise or recall the pattern. In order to calculate the MV of a pattern, S&I must perform two procedures, namely *the longest distance determination* and *the critical point determination*, illustrated in Figure 3.17. The first procedure is *the longest distance determination* process that is formally defined as follows:

Definition 3.7: (The longest distance determination) Given a set of locations of the activated edge nodes in the network $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, the longest distance determination process is the process of calculating the longest distance between the edge nodes' locations and determining the locations of the two edge nodes that have the longest distance between them, which can be described as follows:

$$LongD_{l1 \to l2} = argmax\left(\sqrt{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2}\right), j = (1, \dots, n-1) \quad (3.30)$$

where $LongD_{l1 \to l2}$ is the longest distance between the edge nodes' locations, which is between the edge nodes' locations number $l1$ and $l2$, x_j is the x coordinate location of edge node number j, y_j is the y coordinate location of edge node number j, and n is the number of activated edge nodes.

The total number of times (N_{times}) the distance between the locations of activate edge nodes should be calculated in order to determine the locations of the two edge nodes that have the longest distance between them, can be determined by the number of edge nodes (n) using the following equation:

$$N_{times} = n(n-1) \div 2 \tag{3.31}$$

The second procedure is *the critical point determination*, which is defined as follows:

Definition 3.8: (The critical point determination) Given the locations of the two activated edge nodes that have the longest distance between them $\{(x_{l1}, y_{l1}), (x_{l2}, y_{l2})\}$, the critical point ($CP$) is the midpoint between these two activated edge nodes, which can be described as follows:

$$CP = \left(\frac{x_{l1} + x_{l2}}{2}, \frac{y_{l1} + y_{l2}}{2}\right) \tag{3.32}$$

where $l1$ and $l2$ are the locations numbers of the two activated edge nodes that have the longest distance between them, x_j is the *x* coordinate location of edge node number or location j, and y_j is the *y* coordinate location of edge node number or location j.

The next step for S&I is to calculate the *MV* of a sensed or received pattern, illustrated in Figure 3.17, which is defined as follows:

Definition 3.9: (The MOHA value) Given a set of locations of the activated edge nodes in the network $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, the longest distance between the edge nodes' locations ($LongD_{l1 \to l2}$), and the critical point (CP)

$\{x_{CP}, y_{CP})$, the MOHA value (MV) is equal to the total distances between the locations of activated edge nodes and the location of the critical point divided by the longest distance between the edge nodes' locations, which can be described and calculated as follows:

$$MV = \frac{\sum_{j=1}^{n}\sqrt{(x_j - x_{cp})^2 + (y_j - y_{cp})^2}}{LongD_{l1\to l2}} \tag{3.33}$$

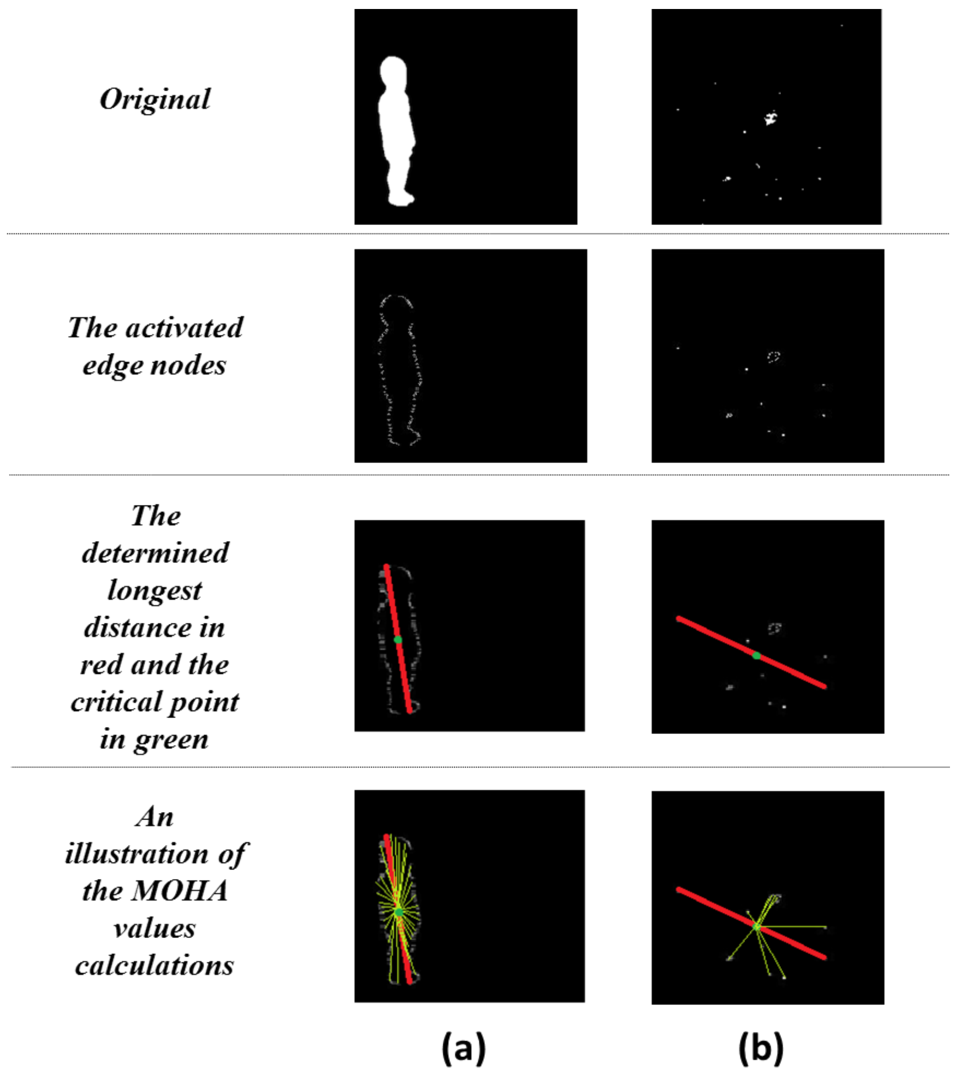
Original
The activated edge nodes
The determined longest distance in red and the critical point in green
An illustration of the MOHA values calculations
(a)
(b)

Figure 3.17: Example of the proposed scheme's recognition procedures for two different types of patterns: (a) a child shape, (b) a star map.

The main reason for dividing the MV by $LongD_{l1 \to l2}$ is to enable the MOHA scheme to recognise scaled patterns that are types of transformed patterns. After the S&I has calculated the MV of a sensed or received pattern, it begins the process of memorising or recalling this pattern. During memorisation, the S&I assigns a unique index number to the pattern, associates this number with the total number of activated edge nodes and the resulting MV, and stores the index number and the associated number of activated edge nodes and MV in its memory. This results in a set of patterns being stored in a vector that can be described as follows:

Definition 3.10: (Pattern vector) Given a set of patterns $\{P_1, P_2, \ldots, P_t\}$, the S&I memorises these patterns by obtaining each pattern's total number of activated edge nodes from a MOHA network (n_i), calculating its MOHA value (MV_i), assigning a unique index number (I_i) to each pattern, and storing the associations of patterns, the numbers of activated edge nodes, and the MOHA values as a pattern vector in the S&I in the following form:

$$\vec{P} = \{(I_1, n_1, MV_1), (I_2, n_2, MV_2), \ldots, (I_t, n_t, MV_t)\}, I_i, n_i \in \mathbb{N}, MV_i \in \mathbb{R} \quad (3.34)$$

In recall, the S&I searches the pattern vector to find a match. To match the input pattern, the distances between the input pattern and each of the stored patterns need to be computed. For instance, a distance measurement called the Hamming measurement counts the locations at which patterns differ [258]. Other distance measurement alternatives include Euclidean distance, City-Block distance, and Canberra distance [258]. The MOHA algorithm utilises the

Euclidean distance measurement, which is defined as the absolute value of the numerical difference of the associated values of the input pattern and other associated values of the stored patterns in the pattern vector, in order to match the input pattern to its nearest stored pattern. Thus, the declaration that a pattern has been detected depends on the total differences between the MOHA network's activated edge nodes number and the activated edge nodes numbers stored in the pattern vector and the pattern's MV and the MVs stored in the pattern vector as follows:

Definition 3.11: (Recalled pattern) Given a number of activated edge nodes determined by the MOHA network (n_c), a MOHA value calculated by the S&I (MV_c), and a pattern vector, the recalled pattern (RP) will be the index number with the smallest differences between the determined edge nodes number and the set of edge nodes numbers stored in the pattern vector and the calculated MV and the set of MVs stored in the pattern vector as follows:

$$RP = I[\min(\Delta n_{1c} + \Delta MV_{1c}, \Delta n_{2c} + \Delta MV_{2c}, \ldots, \Delta n_{tc} + \Delta MV_{tc})] \quad (3.35)$$

where Δn_{ic} is the difference between the i^{th} stored pattern number of activated edge nodes and the number of activated edge nodes determined by the MOHA network for an incoming pattern (current pattern) and ΔMV_{ic} is the difference between the i^{th} stored pattern MOHA value and the MOHA value calculated by the S&I for an incoming pattern.

Moreover, it is clearly shown in this sub-section and in the MOHA network sub-section that all pattern information is stored on the S&I's pattern vector. Therefore, there is no need to store any pattern information in the network nodes (like GN, DHGN, and most other pattern recognition schemes), and this successfully addresses the issue of the sensor nodes' memory limitation in WSNs. From this perspective, the MOHA scheme offers significantly higher storage efficiency compared to other pattern recognition approaches. Additionally, the main reason for giving the S&I the power to determine the final decision about an incoming pattern is that it has an overview of an incoming pattern, which makes it capable of overcoming the *crosstalk* issue and handling transformed patterns.

3.4.5 S&I operations

Figure 3.18 illustrates the S&I operations. These operations can be summarised as follows:

1. **Send command:** The S&I initiates the MOHA learning process by sending a command to the MOHA network's nodes. This command contains the operation type (memorise or recall) and the pattern-obtaining method (direct receive or sense). Further details about this command message will be presented in Section 3.5.
2. **Receive the edge nodes' locations:** The S&I receives the information regarding the locations of the activated edge nodes in the MOHA network.
3. **Determine the longest distance between the edge nodes' locations:** S&I calculates the longest distance between the edge nodes' locations

and determines the locations of the two edge nodes that have the longest distance between them, as described in Definition 3.7.

4. **Determine the critical point:** S&I determines the critical point (CP) as the midpoint between the two edge nodes that have the longest distance between them, as described in Definition 3.8.
5. **Calculate the MOHA value:** S&I calculates the MOHA value (MV) of a pattern as equal to the total distances between the locations of activated edge nodes and the location of the critical point divided by the longest distance between the edge nodes' locations, as described in Definition 3.9.
6. **Memorise or recall:** If the operation is to memorise the incoming pattern, the S&I creates a unique number, associates this index number with the number of activated edge nodes and the MV of the pattern, and stores this association in its pattern vector. However, if the operation is to recall the incoming pattern, the S&I searches for the closest number of activated edge nodes and MV in its pattern vector to the incoming pattern's number of activated edge node and MV and declares its associated pattern as RP.

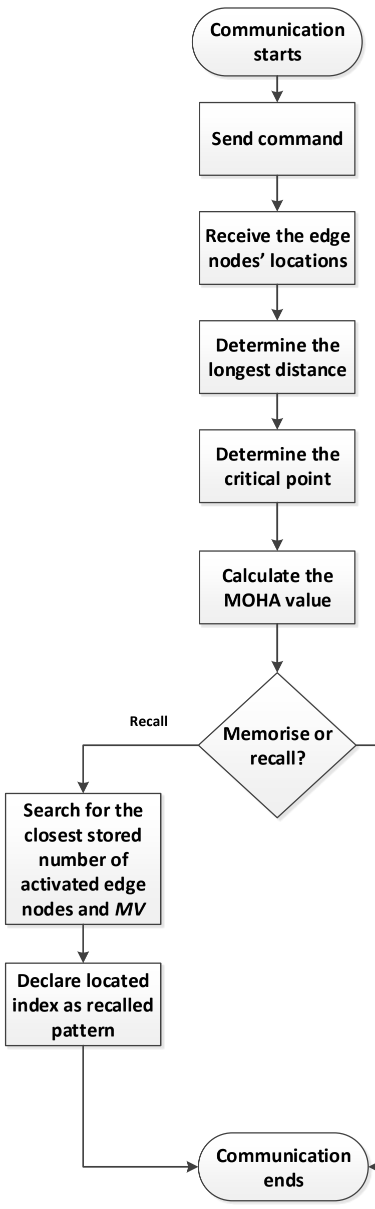
Communication starts
Send command
Receive the edge nodes' locations
Determine the longest distance
Determine the critical point
Calculate the MOHA value
Memorise or recall?
Recall
Memorise
Search for the closest stored number of activated edge nodes and *MV*
Declare located index as recalled pattern
Assign new index number
Store index with the number of edge nodes and the *MV*
Communication ends

Figure 3.18: S&I operations for memorising and recalling patterns.

3.5 Obtaining the Patterns' Data

To perform pattern recognition operations, the pattern-obtaining operations of the MOHA network are discussed in this section. The MOHA scheme has two types of operations, namely, memorisation (storing) and recall. The MOHA adopts the supervised pattern recognition manner. This means that a MOHA network will be presented with a set of patterns to store and will then recall other patterns in accordance with the stored ones. These patterns can be imposed by the S&I (in a cluster head or a base station) or obtained by sensor readings. These operations are initiated by the S&I sending command messages to sensor nodes in the network. The command message from the S&I takes the form $\{C, P\}$, which means command (“C”= ‘command’) and pattern (“P”= ‘pattern’). S&I divides an incoming pattern into sub-patterns where each sub-pattern is managed by one MOHA row (GN array).

The memorisation or training operation should always be initiated by the S&I. The S&I sends a command message to each node in the network of two parts $\{M, P\}$, which means memorise (“M”=‘memorise’) pattern element (“P”=pattern element). The pattern element part in the message can be provided by the S&I. Alternatively, the S&I will set the second part to *“S”*, meaning that the node should take its sensory information as the incoming pattern element (“S”= sense). Figure 3.19 shows an example of sending the pattern ($P = \{350,400,500,400,350,400,300,200,250\}$) to be stored in a 9-neuron (node) positions size MOHA network. The S&I will break the pattern into 3 sub-patterns (350,400,500), (400,350,400), and (300,200,250) and send these sub-patterns to rows 1, 2, and 3 respectively. The S&I will send the command $\{M, p\}$ to each node position in the row based on the value assigned to the position. Alternatively, if the existing pattern (for an event) in the sensed environment (for example, temperature, Infrared (IR), Ultrasound, or pressure readings) is to be stored, the S&I will send the command $\{M, S\}$ to all network nodes.

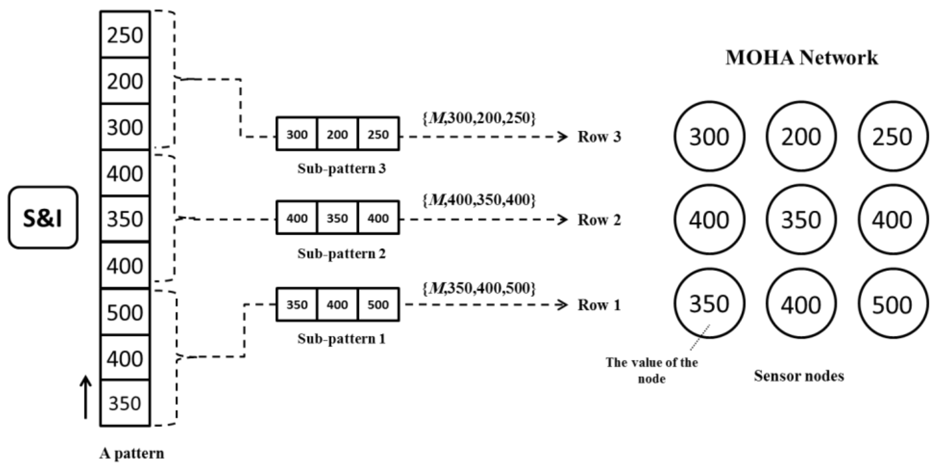

Figure 3.19: S&I divides a 9 size pattern into 3 sub-patterns and sends each sub-pattern to a MOHA row for memorisation.

The recall (recognition) operation, on the other hand, can be initiated automatically in a periodic manner or by the S&I. In a periodic recall operation, sensor nodes are given a time cycle where every sensor should take its readings as the incoming pattern. This suits automated and monitoring applications that require continuous recognition over the field of interest. S&I initiated recall operations are obtained by sending a command message of the form $\{R, P\}$ to all network nodes, meaning recall ("R"= recall) pattern element. Similar to memorised commands, the pattern element part of the message can be provided by the S&I or requested to be sensed by sensor nodes. S&I initiated recall commands suit applications that require recognition at a certain point of time such as query-driven applications.

Table 3.3 summarises the possible command messages that can be sent from the S&I to network nodes.

Table 3.3: Command messages from S&I to network nodes.

Command	Description
$\{\boldsymbol{M}, \boldsymbol{p}\}$	Memorise (store) the value p, where p is a value of a given pattern element by the S&I.
$\{\boldsymbol{M}, \boldsymbol{S}\}$	Memorise the sensory information
$\{\boldsymbol{R}, \boldsymbol{p}\}$	Recall the value p, where p is a value of a given pattern element by S&I.
$\{\boldsymbol{R}, \boldsymbol{S}\}$	Recall the sensory information

3.6 MOHA Communication Requirements and Protocol

In this section, the communication requirements and protocol of a MOHA network are described. To perform MOHA network node communications, this study assumes that a medium access control (MAC) protocol is present and available to support the network. MAC protocols control the communications of a network by setting the rules and steps for sending information amongst the network's sensor nodes so as to share the available medium. In WSNs, the efficiency of a MAC protocol will affect the sensor nodes' lifetime by reducing transmission collisions, which reduces the number of retransmissions of packets [19]. In WSNs, sensor nodes conserve energy resources by alternating between low power sleep mode and active mode. MAC protocols conserve energy resources for WSN nodes by determining timeslots for sleep and active modes. Moreover, when utilising MAC protocols, each sensor node can have a unique MAC address that differentiates it from other sensor nodes, allowing direct communication between two sensor nodes. For a MOHA scheme, the choice of a MAC protocol must take into account the special requirements and steps of the network's communications.

During the initialisation of a MOHA network, each sensor node should be provided with a row number and a position number that it will work on. This allows the sensor node to determine its communication process. For example, a sensor node in the top or the bottom rows will send its information to two or three adjacent nodes (depending on its position in that row), receive adjacent nodes' information, and report its location to the S&I if it is been activated as an edge node. A sensor node in middle rows performs the same steps except that it will exchange information with three or four adjacent nodes depending on its position in these rows. A sensor node's row and position can be determined statically or automatically during the initialisation phase of the MOHA network. Static row and position determination means that each sensor node is provided with information about its row, position, and adjacent nodes with which to exchange information. This initialisation would be less complex in terms of computations and communications. However, there will be limited flexibility in terms of adding new sensor nodes to the network or adopting dynamic changes such as mobile nodes or clusters. Automatic row and position determination can be achieved by allowing each sensor node to communicate with its neighbouring nodes and allowing the S&I (in the base station) to determine its row, position, and adjacent nodes after the deployment of the network. This approach will provide more network flexibility, thereby enabling adaptation to changes that may be required in the network design. However, this will lead to an increase in the number of communications in the network. It is also important to take into account the distance between sensor nodes and the communication ranges of sensor nodes when determining the row size.

To ensure the functionality of the MOHA network, each sensor node should be fed with sufficient information about how to react to failures or de-activated neighbouring sensor nodes. In WSNs, it is common to have failure sensor nodes that can no longer communicate due to depleted energy or physical damage. To overcome the effect of such phenomena on recognition, each sensor node should take into account the steps for dealing with unavailable sensor nodes. For adjacent sensor nodes (i.e. previous and next adjacent nodes in the same row and the adjacent nodes at the lower and higher rows), a sensor node should assume the value of a failed communicating node to be equal to its own value. This is in order to avoid an interruption to the recognition process and eliminate the effects of these failed nodes on the edge determination process.

The MOHA communication protocol describes the main steps of MOHA network communications. By completing this protocol, a MOHA network will memorise or recall an incoming pattern. The protocol consists of three main steps as follows:

Step 1: $\boldsymbol{S\&I} \Rightarrow \boldsymbol{N_1}, \boldsymbol{N_2}, \dots, \boldsymbol{N_m}: (C_i, P_i)\ \{(C_1, P_1), (C_2, P_2), \dots, (C_m, P_m)\}$

In the first step of the communication, the S&I (in the base station) sends the command C_i and the pattern elements P_i to each sensor node in the network. As explained in section 3.5, a command can be either to memorise (M) or recall (R) and the pattern element can be a value to use for training, or (S) to initiate sensors to use the sensory information. Each sensor node receives only one element of the pattern. For instance, if the pattern is (2,5,8,6), the S&I will send the values 2, 5, 8, and 6 to the sensor nodes 1, 2, 3, and 4 as pattern elements respectively. There are two methods for obtaining sensory information. In the first scenario, the S&I sends a 'memorise' or 'recall' command to sensor nodes in order to start obtaining sensory information and continue the communication steps. In the second scenario, the sensor nodes are programmed to obtain information periodically.

Step 2: $\boldsymbol{N_i^{r_m}} \Rightarrow \boldsymbol{N_{i-1}^{r_m}}, \boldsymbol{N_{i+1}^{r_m}}, \boldsymbol{N_i^{r_{m+1}}}, \boldsymbol{N_i^{r_{m-1}}}: v_i$

Each activated sensor node in the network $N_i^{r_m}$ starts the information exchange process with adjacent sensor nodes. After receiving or obtaining the pattern element information, each activated sensor node sends its value v_i to four adjacent sensor nodes: previous node in its row $N_{i-1}^{r_m}$, next node in its row $N_{i+1}^{r_m}$, adjacent node in the next higher row $N_i^{r_{m+1}}$, and adjacent node in the lower row $N_i^{r_{m-1}}$. The main aim of this step is to allow each sensor node to determine whether it represents a pattern edge according to Equation 3.29. Based on the equation result, it is either de-activated or activated as an edge node, as described in Definition 3.5. After completing this step, each sensor node will receive four values from its adjacent nodes representing the pattern elements by which a sensor node should be able to determine whether or not it will be activated as an edge node.

Step 3: $\boldsymbol{EN_c} \rightarrow \boldsymbol{S\&I}: x_c, y_c$

Each activated edge node EN_c reports its location information (i.e. its location in terms of (*x*,*y*) coordinate in space) to the S&I (in the base station) in order to store or recall the pattern. The S&I utilises the information received about locations in order to calculate the MOHA value (MV) of the pattern. The S&I retains a database of trained patterns associated with their total number of activated edge nodes and their MV values. If the pattern needs to be memorised, the S&I assigns it a new index number and associates this index number with the total number of activated edge nodes obtained by the network and its MV value. If the pattern is to be recalled, the S&I searches its database to find the value closest to the total number of activated edge nodes given by the MOHA network with the MV of the pattern and declares it as the recalled pattern.

3.7 Qualitative Performance Analysis

One of the main aims of the proposed scheme here is to provide real-time recognition capacities while maintaining a low level of resource use to suit WSNs. Therefore, in this section, a series of analyses and evaluations for the MOHA implementation were conducted, to study the complexity of the

MOHA algorithm in terms of computations, learning cycle time, and number of communications. For such evaluations, we assume that all network nodes are activated as edge nodes by a given pattern to estimate the maximum computations, time, or communications required to learn or recall an incoming pattern. Additionally, the complexity study in term of memory is not presented because the MOHA does not require storing any pattern information in the network nodes, and this successfully addresses the issue of sensor nodes having a limited memory in WSNs. The following sub-sections provide the aforementioned analysis.

3.7.1 Computational complexity evaluation

To analyse the computational complexity of the MOHA algorithm, Big-O analysis has been considered as the computational complexity indicator. It is mainly used as the computational complexity measurement tool to describe how the input data affects an algorithm's usage of computational resources. Big-O analysis can be defined as a theoretical measure of the algorithm's resource usage during execution, mostly the time or memory required, given the problem size n, which is usually the number of items [259]. Informally, if an equation is $f(n) = O(g(n))$ this means that it is less than a constant multiple of $g(n)$. In this sub-section, a comparison is made with Hopfield network, which is presented in Chapter 2. Other comparisons, with Self-Organising Map (SOM) and Distributed Hierarchical Graph Neuron (DHGN), can be found in Appendix A. It is best to note, however, that the comparative study that has been carried out does not intend to outweigh the capabilities of these algorithms. Rather, it indicates that the operations associated with the MOHA have a significantly low computational complexity. In this sub-section, we present only one comparison, as this is adequate to show the quality of the MOHA.

In order to determine the computational complexity of the MOHA and Hopfield network schemes, a 2-stage process was implemented to emulate the network generation (i.e. deployment) and recognition stages of the two approaches.

3.7.1.1 Network generation stage

This stage involves the formation of a network that comprises computing elements known as neurons (i.e. nodes). It is worth noting that this stage needs to be done only once, which makes it less important than the recognition stage in terms of determining the computational complexity. The number of nodes generated (also known as a network size) is totally dependent on the algorithm being implemented.

In the MOHA scheme implementation, the number of nodes generated (i.e. network size), G_{MOHA} depends on the number of nodes required for each row (N_{node}) and the total number of rows (N_{Row}), G_{MOHA} is given as follows:

$$G_{MOHA} = N_{node} \times N_{Row} \tag{3.36}$$

According to Equation 3.15, the number of nodes (N_{node}) required for a row is equal to the size of the sub-pattern S_{sb} and according to Equation 3.20, the total number of rows (N_{Row}) is equal to the size of a pattern (S_p) divided by the size of the sub-pattern S_{sb}. As a result, G_{MOHA} will be equal to:

$$G_{MOHA} = N_{node} \times N_{Row} = S_{sb} \times \frac{S_p}{S_{sb}} = S_p \tag{3.37}$$

On the other hand, the number of neurons generated, G_{Hopf} in a Hopfield network implementation (presented at Chapter 2) is equivalent to the pattern size [260], S_p:

$$G_{Hopf} = S_p \tag{3.38}$$

The details of the Big-O notation derived for the MOHA and Hopfield network implementations are shown in Table 3.4. The estimated time derived is based on the assumption that the instruction speed used is 1 microsecond (μs) per instruction.

Table 3.4: Big-O notations for Hopfield network and MOHA implementation in network generation stage.

Algorithm	**Big-O**	**Efficiency**	**Iteration (*n*)**	**Estimated Time (in seconds)**
MOHA	$O(n)$	Linear	G_{MOHA}	$G_{MOHA} \times 0.000001$
Hopfield	$O(n)$	Linear	G_{Hopf}	$G_{Hopf} \times 0.000001$

The results show that both the MOHA and Hopfield networks have identical computational complexity for the network generation stage. However, if parallelism is taken into account, for each MOHA row, the number of nodes generated for each row is less than the overall node initialisation within the network. Therefore, the estimated time for network generation in the MOHA is lower than in the Hopfield network implementation.

3.7.1.2 Recognition stage

This stage is the main process for the pattern recognition algorithm. Every pattern recognition algorithm has a different approach for doing so. In the Hopfield network, the recognition stage involves three sub-processes [260], which are weight accumulation, weight determination for the whole network, and network propagation to derive optimum solution. On the other hand, the MOHA algorithm demands only a single-cycle process of recognition within this recognition stage. This process of recognition involves five sub-processes, as presented in Section 3.4, which are: activated nodes determination, activated edge nodes determination, the longest distance determination between the locations of activated edge nodes, the critical point determination, and the MOHA value determination. Only the first two sub-processes are performed by the MOHA network and the rest of these sub-processes are handled by the S&I (e.g. the base station).

Table 3.5 shows the Big-O notations derived from the analysis on the Hopfield network recognition process. As stated previously, the estimated time derived is based on the assumption that the instruction speed used is 1 microsecond (μs) per instruction.

Table 3.5: Big-O notations for Hopfield network in recognition stage.

Process	**Big-O**	**Efficiency**	**Iteration (n)**	**Estimated Time (in seconds)**
Weight Accumulation	$O(n)$	Linear	S_p	$S_p \times 0.000001$
Weight Determination	$O(n^2)$	Quadratic	G_{Hopf}^2	In minutes
Network Propagation	$O(n^k)$	Polynomial	G_{Hopf}^k	In hours

The Hopfield network incurs a considerably high computational complexity, as indicated in Table 3.5 with respect to its weight determination and network propagation processes. Figure 3.20 shows the computational complexity for these three processes during the recognition stage utilising the Hopfield network implementation. Note that for the network propagation process, the value $k = 3$ was utilised for polynomial representation.

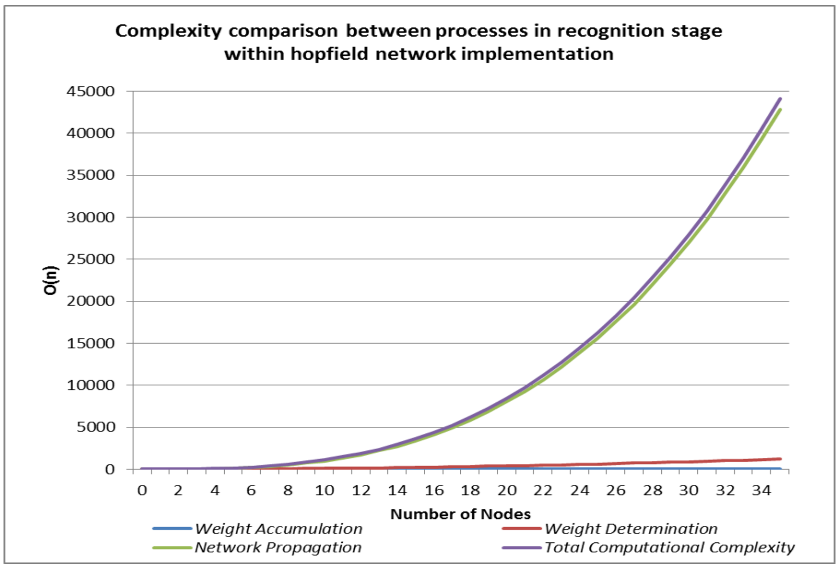

Figure 3.20: Big-O notation comparisons for processes within the Hopfield network recognition stage.

Table 3.6 shows the Big-O notations derived from the analysis on the MOHA recognition process. This is based on the assumption that all network nodes are activated as edge nodes by a given pattern in order to estimate the maximum computations required for pattern recognition. The estimated time derived is based on the assumption that the instruction speed used is 1 microsecond (μs) per instruction.

Table 3.6: Big-O notations for MOHA implementation in recognition stage.

Process	Big-O	Efficiency	Iteration (n)	Estimated Time (in seconds)
Activated Nodes Determination	$O(n)$	Linear	$G_{MOHA} = S_p$	$S_p \times 0.000001$
Activated Edge Nodes Determination	$O(n)$	Linear	$G_{MOHA} = S_p$	$S_p \times 0.000001$
The Longest Distance Determination	$O(n)$	Linear	$(N_{ED}(N_{ED} - 1) \div 2) + 1$	$\left((N_{ED}(N_{ED} - 1) \div 2) + 1\right) \times 0.000001$
The Critical Point Determination	$O(1)$	Linear	1	0.000001
The MOHA Value Determination	$O(1)$	Linear	1	0.000001

where N_{ED} is the number of activated edge nodes in the network. In other words, it is the number of the received locations information from the activated edge nodes.

The MOHA algorithm suffers a very slightly high computational complexity, as indicated in Table 3.6 with respect to the process that determines its longest distances, which is handled by the S&I (e.g. the base station) only without the MOHA network. On the other hand, the computational complexity associated with determining the critical point and the MOHA value is almost neglected. Figure 3.21 shows the computational complexity for these five processes in the recognition stage utilising the MOHA. Note that the computational complexity of the MOHA is based on the assumption that 100% of the network nodes are activated as edge nodes, which is a worst case scenario for the algorithm and rarely occurs.

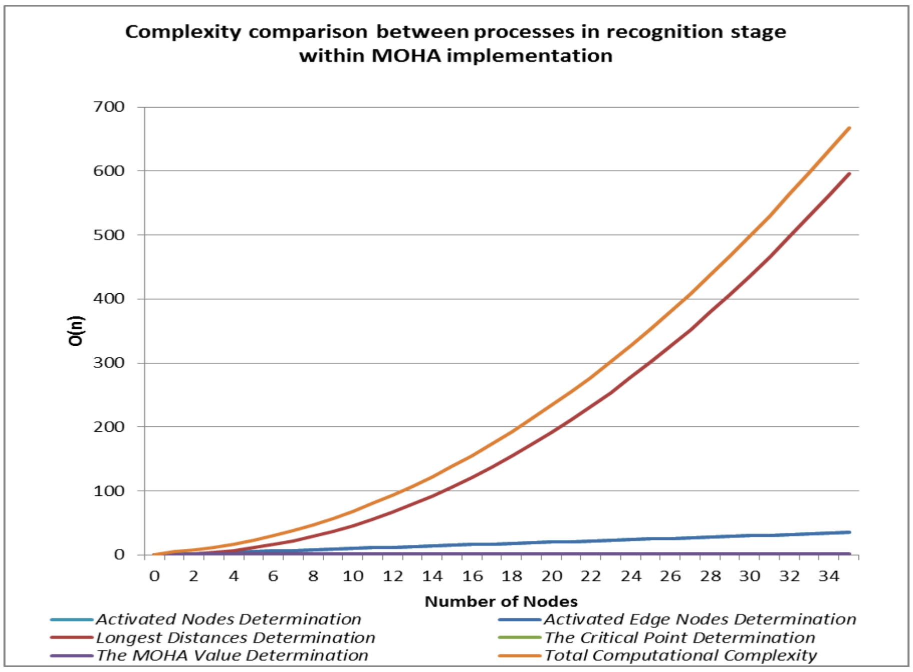

Figure 3.21: Big-O notation comparisons for processes within MOHA recognition stage.

It is clearly shown from the analysis that the MOHA implementation incurs less computational complexity during the pattern recognition process compared with that of the Hopfield network. For example, the total computational complexity (i.e. the number of instructions) for MOHA algorithm utilising 35 nodes is 668 during the recognition process. However, it is 44135 in the Hopfield network implementation for the recognition stage with the same number of nodes. This is mainly a result of employing a simple nearly linear function using the MOHA algorithm, whereas the Hopfield network employs expensive polynomial and quadratic functions.

3.7.2 Scalability analysis

As mentioned earlier, one of the goals of the MOHA scheme is to provide learning capabilities in resource-constrained WSNs while maintaining the high scalability and speed of a GN scheme. The scalability factor for the MOHA pattern recognition scheme could be determined from two different perspectives: the time analysis and the communication efficiency. Table 3.7 represents the various terms used when estimating the complexity of the MOHA algorithm.

Table 3.7: Symbols and terms for complexity estimation.

Symbol	Terms Name	Terms meaning
T_{rec}	**Pattern Receiving Time**	The time required by a MOHA network to obtain an incoming pattern including the S&I command
T_{LongD}	**Longest Distance Determination Time**	The time required by the S&I to determine the longest distance between the edge nodes' locations
T_{sense}	**Sense Time**	The time required by a node to obtain sensory information
T_{check}	**Checking Time**	The time required by a node to check whether it represents a pattern edge according to Equation 3.29.
T_{exch}	**Exchange Time**	The time required by nodes to perform exchange communications
T_{send}	**Send Time**	The time required to send a message from one node to another
T_{report}	**Report Time**	The time required by the network to conduct report communications
N_{times}	**The Number of Times**	The number of times the distance between the locations of activate edge nodes should be calculated in order to determine the locations of the two edge nodes that have the longest distance between each other.
T_{add}	**Addition Time**	Time for the node or the S&I to complete an addition operation
T_{sub}	**Subtraction Time**	Time for the node or the S&I to complete an subtraction operation
T_{pw}	**Power Time**	Time for the S&I to complete an exponent (power) operation
T_{root}	**Square Root Time**	Time for the S&I to complete a square root operation
$T_{compare}$	**Compare Time**	The time required by the S&I to compare two values
N_{ED}	**Number of Activated edge nodes**	The number of activated edge nodes in the network
T_{div}	**Division Time**	Time for the S&I to complete a division operation

T_{MV}	**MOHA Value Time**	The time required by the S&I to calculate the MOHA value of an incoming pattern
T_{recall}	**Recalling Time**	The time required by the MOHA network to recall an incoming pattern
T_{read}	**Reading Time**	The time required by the S&I to read a single pattern's stored data
M_p	**Memorised Patterns**	The number of memorised patterns in the S&I
T_{total}	**Total Network Time**	The time required by the MOHA network to perform PR operations
T_{CP}	**Critical Point Time**	The time required by the S&I to determine the critical point
T_{mem}	**Memorisation Time**	The time required by the MOHA network to memorise an incomming pattern
T_{write}	**Writing Time**	The time required by the S&I to write (store) an incoming pattern's information in ths S&I's pattern vector

3.7.2.1 Time analysis

In order to estimate the learning cycle or the recall time of the MOHA algorithm, we estimate the time taken by each of its recognition steps, and then add them together to estimate the total time needed by the algorithm. For such estimation, we assume that all network nodes are activated as edge nodes by a given pattern to estimate the maximum time required to learn or recall an incoming pattern.

The first step is the pattern-receiving step. This step involves broadcasting the command message by the S&I and sensing the pattern by the nodes. The estimation time for the first step can be calculated using the following equation:

$$T_{rec} = T_{send} + T_{sense} \tag{3.39}$$

The second step is the node's exchange of sensory information step. We assume that each node has the maximum number of adjacent nodes (4 adjacent nodes) with which to exchange information. Taking parallelism into account, the time estimate can be described as follows:

$$T_{exch} = 4T_{send} \tag{3.40}$$

This is followed by checking whether each node is activated as an edge node based on its adjacent nodes' values, according to Equations 3.28 and 3.29. Each activated edge node in the network reports its location information to the S&I. Taking parallelism into account, the time estimate can be described as follows:

$$T_{report} = T_{add} + 2T_{sub} + T_{check} + T_{send} \tag{3.41}$$

Once the activated edge nodes' location information is reported to the S&I, the S&I calculates the longest distance between the edge nodes' locations and determines the locations of the two edge nodes that have the longest distance between them. As shown in the previous sub-section, this step is considered to have the highest computational complexity in the MOHA scheme. N_{times} is the number of times the distance between the locations of activate edge nodes should be calculated in order to determine the locations of the two edge nodes that have the longest distance between them. Thus, the estimation time for this step can be calculated using this equation:

$$T_{LongD} = N_{times} \times (3T_{add} + 2T_{pw} + T_{root}) + T_{compare} \tag{3.42}$$

According to Equation 3.31, the number of times (N_{times}) can be determined by the number of activated edge nodes (N_{ED}) using the following equation:

$$N_{times} = N_{ED}(N_{ED} - 1) \div 2 \tag{3.43}$$

As a result, T_{LongD} will be equal to:

$$T_{LongD} = (N_{ED}(N_{ED} - 1) \div 2) \times \left(3T_{add} + 2T_{pw} + T_{root}\right) + T_{compare} \quad (3.44)$$

This is followed by determining the critical point, which is the midpoint between the two edge nodes that have the longest distance between them. The estimation time for this step can be calculated using this equation:

$$T_{CP} = 2T_{add} + 2T_{div} \quad (3.45)$$

In the next step, S&I calculates the MOHA value (MV) of a pattern as equal to the total distances between the locations of activated edge nodes and the location of the critical point divided by the longest distance between the edge nodes' locations. The estimation time for this step can be calculated using this equation:

$$T_{MV} = N_{ED} \times \left(3T_{add} + 2T_{pw} + T_{root}\right) + T_{div} \quad (3.46)$$

Finally, S&I utilises the number of activated edge nodes and the MV of the incoming pattern in order to store or recall the pattern. In cases of memorisation, the S&I memorising time (T_{mem}) will be equal to T_{write}. However, assuming that S&I performs a binary search to find the associated pattern, its recall time can be estimated as follows:

$$T_{recall} = \left(T_{read} + T_{compare}\right) \times \log_2 M_p \quad (3.47)$$

On the basis of previous equations, we can now construct the equation for calculating the total time taken for all steps of the MOHA for memorisation:

$$T_{total} = T_{rec} + T_{exch} + T_{report} + T_{LongD} + T_{CP} + T_{MV} + T_{mem} \quad (3.48)$$

$$T_{total} = T_{send} + T_{sense} + 4T_{send} + T_{add} + 2T_{sub} + T_{check} + T_{send} + (N_{ED}(N_{ED} - 1) \div 2) \times (3T_{add} + 2T_{pw} + T_{root}) + T_{compare} + 2T_{add} + 2T_{div} + N_{ED} \times (3T_{add} + 2T_{pw} + T_{root}) + T_{div} + T_{mem} \tag{3.49}$$

$$T_{total} = 6T_{send} + T_{sense} + T_{check} + \left(\frac{1}{2}N_{ED}^{2} + \frac{1}{2}N_{ED}\right) \times (3T_{add} + 2T_{pw} + T_{root}) + T_{compare} + 3T_{add} + 2T_{sub} + 3T_{div} + T_{write} \tag{3.50}$$

In recall, the total network time required for recall can be estimated as follows:

$$T_{total} = T_{rec} + T_{exch} + T_{report} + T_{LongD} + T_{CP} + T_{MV} + T_{recall} \tag{3.51}$$

$$T_{total} = 6T_{send} + T_{sense} + T_{check} + \left(\frac{1}{2}N_{ED}^{2} + \frac{1}{2}N_{ED}\right) \times (3T_{add} + 2T_{pw} + T_{root}) + T_{compare} + 3T_{add} + 2T_{sub} + 3T_{div} + (T_{read} + T_{compare}) \times \log_2 M_p \tag{3.52}$$

To simplify these calculations, we assume that communicational operations times (T_{send}, T_{sense}) are equal and denoted as ($T1$). Similarly, computational operations times (T_{add},T_{sub}, T_{check}, $T_{compare}$, T_{pw}, T_{root},T_{div}, T_{read}, T_{write}) are assumed to be equal, and we denote them as ($T2$). Accordingly, substituting $T1$ and $T2$ into (3.50) can provide an estimate of the total network time required for memorisation.

$$T_{total} = 7T1 + 11T2 + \left(\frac{1}{2}N_{ED}^{2} + \frac{1}{2}N_{ED}\right) \times 6T2 \tag{3.53}$$

On the other hand, substituting $T1$ and $T2$ into (3.52) can provide an estimate of the total network time required for recalling.

$$T_{total} = 7T1 + 10T2 + \left(\frac{1}{2}N_{ED}^{2} + \frac{1}{2}N_{ED}\right) \times 6T2 + 2T2 \times \log_2 M_p \quad (3.54)$$

According to these equations, the total number of nodes (the network size) in the wireless sensor network and the pattern size (S_p) do not have any influence on the total time taken, T_{total}. As a result, the MOHA algorithm can be effectively scaled to support any number of nodes in the wireless sensor network. The MOHA network's response time is proportional to the number of activated edge nodes (N_{ED}) in the network, which is generally less than the pattern size, $N_{ED} < S_p$. This minimises the effect of pattern size increase and provides the scheme with a high level of scalability. Moreover, the number of activated edge nodes N_{ED} does not have a significant influence on the total time T_{total} as it is multiplied by $T2$, which refers to the computational operations time that is dependent on the processing capability of the S&I (e.g. the base station). The S&I has often been deployed in the base station, which makes its processing capability very high. This makes the proposed algorithm capable of handling an increasing number of activated edge nodes with a minimal corresponding increase in the total time taken. From the analysis of the MOHA scheme, it can be concluded that the MOHA network is capable of maintaining the high scalability feature of the GN network. Furthermore, the analysis shows that the scheme is capable of performing learning operations in predictable time while restricting the learning cycle so that it is proportional to the number of activated edge nodes N_{ED}. These features make the scheme appropriate for tackling large-scale, real-time problems. To sum up, in an ideal wireless sensor network scenario in which all communications can occur simultaneously, the MOHA algorithm demonstrates that it is highly scalable and produces only a marginal increase in the total time required for pattern recognition.

3.7.2.2 Communication complexity analysis

As described in Chapter 2, communication operations are one of the most important factors for energy consumption in WSNs. Therefore, the number of communications involved in performing pattern recognition using the MOHA can be used as the second aspect of scalability determination. High communication costs will incur additional overhead for the network to support the core functions of the algorithm. Hence, the intention is to minimise the communication costs within MOHA. The two MOHA operation types (memorise or recall) must be considered when estimating the number of communications in a MOHA network. Both memorisation and recall operations involve the exchange of communications where each activated node sends its information to its adjacent nodes. For such estimation, we assume that all network nodes are activated by a given pattern to estimate the maximum number of communications required to learn or recall an incoming pattern. In the MOHA scheme, normally each activated node has four adjacent nodes, and it exchanges information with them. However, the actual number of adjacent nodes and exchange communications of an active node varies from two to four based on its location or position in the row and whether the row is located at the bottom, the middle, or the top of the network, as follows:

Bottom row: According to Equation 3.16, the cumulative communication costs involved for each input recognition process for all nodes in the bottom row, based on the assumption that all the nodes are activated, is derived through the following equation:

$$C_{r_0} = 3(S_{sb} - 2) + 4 \tag{3.55}$$

where C_{r_0} is the total number of exchanged communications in the bottom row (r_0) and S_{sb} is the size of the sub-pattern that is assigned to this row.

Middle rows: According to Equation 3.18, the cumulative communication costs for all nodes in the all middle rows, based on the assumption that all the nodes are activated, is derived through the following equation:

$$C_{r_i}^{total} = \sum_{i=1}^{top-1}(4(S_{sb} - 2) + 6) \tag{3.56}$$

where $C_{r_i}^{total}$ is the total number of exchanged communications in all middle rows and S_{sb} is the size of the sub-pattern that is assigned to these rows based on the assumption that all the middle rows assigned with the same S_{sb}.

Top row: According to Equation 3.19, the cumulative communication costs involved for each input recognition process for all nodes in the top row, based on the assumption that all the nodes are activated, is derived from the following equation:

$$C_{r_{top}} = 3(S_{sb} - 2) + 4 \tag{3.57}$$

where $C_{r_{top}}$ is the total number of exchanged communications in the top row (r_{top}) and S_{sb} is the size of the sub-pattern that is assigned to this row.

Each activated edge node in the network is required to give one report to the S&I. Therefore, the number of report communications can be estimated as follows:

$$C_{report} = N_{ED} \tag{3.58}$$

where C_{report} is the total number of report communications to the S&I and N_{ED} is the number of activated edge nodes in the network.

The total communication costs for MOHA network in memorisation or recalling operations could be derived as follows:

$$C_{total} = C_{r_0} + C_{r_i}^{total} + C_{r_{top}} + N_{ED} \tag{3.59}$$

In order to evaluate the complexity of the proposed scheme's communication process, it will be compared with that of the DHGN algorithm. The DHGN algorithm has been selected because it is one of pattern recognition schemes for WSNs that offers low communication cost, as discussed in Chapter 2. Table 3.8 and Figure 3.22 show the comparison of the communication costs for DHGN and MOHA. The MOHA's total communication is based on the assumption that 100% of the network nodes are activated as edge nodes. The total communication messages of both algorithms are based on the assumption that the sub-pattern size (S_{sb}) is equal to 5.

From the comparison results, it can be concluded that on average, the implementation of the MOHA achieves a saving of 26.1% for message passaging when compared to DHGN. As discussed in section 2.5, the number of nodes required in DHGN increases exponentially with the increase of the problem (pattern) size, which will lead to an increase in the number of messages communicated.

Table 3.8: Comparison between DHGN and MOHA implementations with regards to the number of messages communicated per pattern size (S_p).

S_p	C_{total}^{DHGN}	C_{total}^{MOHA}
5	28	18
15	84	59
25	140	105
35	196	151
45	252	197
55	308	243

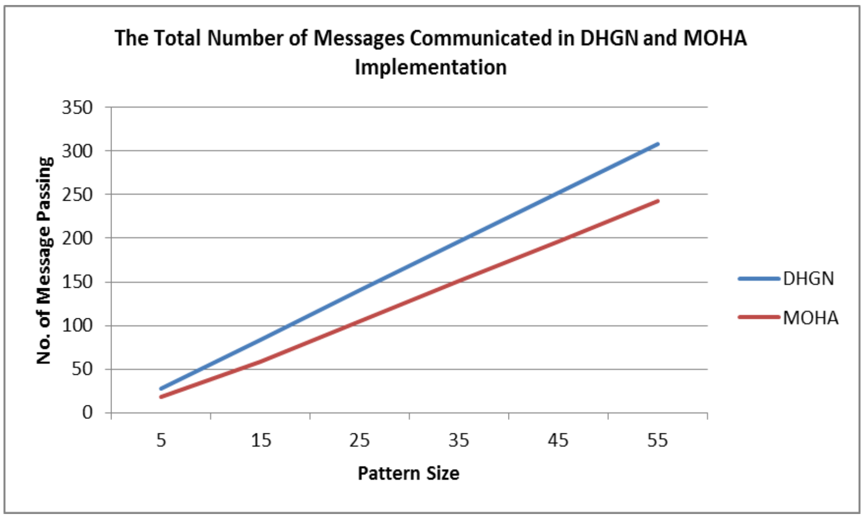

Figure 3.22: A comparison of communication costs of DHGN and MOHA schemes.

The analyses that have been carried out indicate that the MOHA provides high scalability with increasing sizes of patterns through the use of smart distributed approaches within a single-cycle learning environment. Furthermore, the MOHA has been proven to have less complexity compared to Hopfield network, in regards to its processing requirements. Moreover, this section shows that the MOHA scheme offers pattern recognition without requiring overly large numbers of message exchanges, which makes it suitable for deployment in wireless sensor networks. Clearly, reducing the amount of communication between sensor nodes will lead to an increase in the overall energy-efficiency of the network.

In order to evaluate the MOHA scheme's capabilities as a pattern recognition algorithm, a series of simulations will be analysed in the next section.

3.8 Simulations and Results Analysis

In this section, a series of tests have been conducted on the MOHA scheme. These tests were applied to examine the performance of our proposed algorithm and its accuracy as a pattern recognition scheme. The main aim of these tests is to prove the capabilities of the MOHA scheme in dealing with transformed and noisy patterns. To perform these tests, we constructed a database called *MPEG7 CE Shape-1 Part B* dataset (*shapes* dataset) that consisted of 1400 binary shape images for training and testing [261]. A set of these binary images was tested and discussed through these simulations. Inter-node communications and S&I communications were implemented utilising the MPICH-2 library for the message passing interface (MPI) [262]. The simulator itself was written in the C/C++ Integrated Development Environment.

3.8.1 Recognition test on binary images

This test was used to examine the proposed scheme's ability to detect noisy patterns and provide high classification and recall accuracy. The training dataset was constructed by creating and utilising ten shapes modelled as a binary image of size 100-by-100 pixels, as shown in Figure 3.23. These images were presented to the MOHA network for memorisation (storing). In this test, we ran a simulated MOHA network of 10000 nodes assuming that nodes are distributed as a grid and each node detects one pixel reading. We also assumed that MOHA network is divided into 100 rows and each one of these rows is utilised to handle 100-bit binary sub-patterns and the threshold values for node activation and edge determination are set to 1 ($\varphi = 1$).

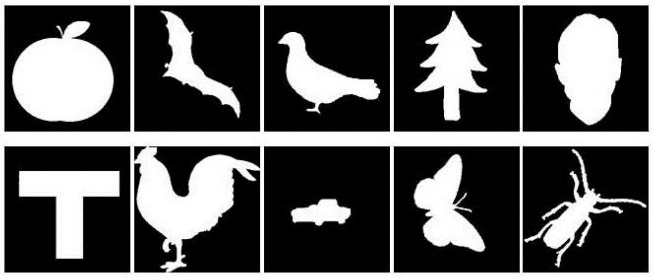
Figure 3.23: Binary images patterns used in the recognition test.

This recognition test consisted of two parts. The first part involved the recognition of the distorted image "apple" against 10 binary images previously stored in the S&I. We applied different levels of noise to the "apple" image (from 1% to 30%). The results of this test are shown in Figure 3.24. The second part used a similar configuration; however, this part involved the recognition of a variety of shapes of "apple" images to determine the classification and recall accuracy of the MOHA algorithm. Figure 3.25 shows the results for the second part of the recognition test.

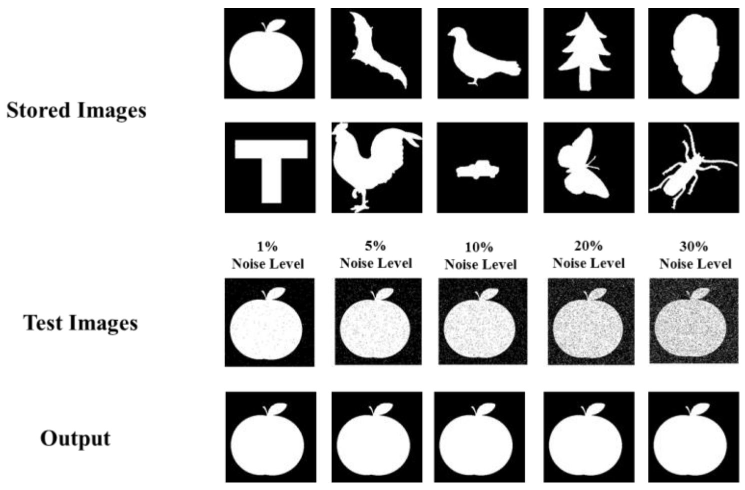

Figure 3.24: Results of image recognition test with 5 noisy images.

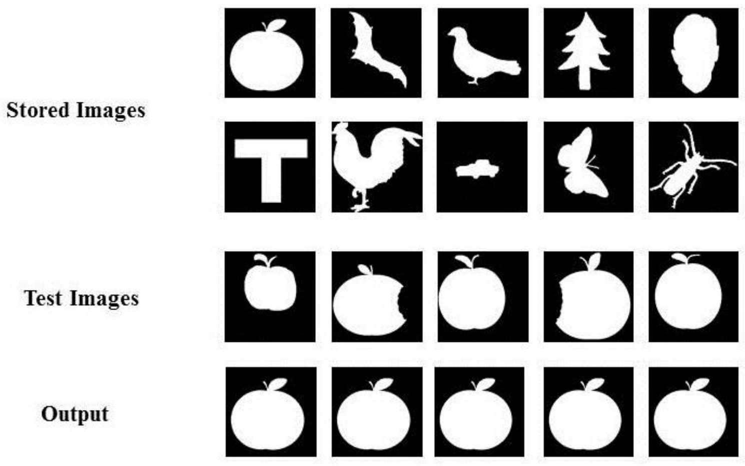

Figure 3.25: Results of image recognition test with 5 different shapes of an image.

The test results show that the MOHA pattern recognition scheme is capable of providing high recall accuracy for noisy binary image recognition (up to 30%). It is also capable of offering high classification accuracy for different shapes of the same image.

3.8.2 Capability of handling pattern transformations

MOHA achieves invariant recognition mainly by using MOHA values (MVs) rather than storing the pattern's information as is done in standard GN. The patterns' MVs are stored in the S&I of a MOHA network as a vector, according to Definition 3.10. There are three types of pattern transformations, namely displaced positions (also known as translation), scaling (also known as dilation), and rotation. In order to show the effects of these types of pattern transformations in the pattern recognition schemes, we utilised an example of a binary "table" image, shown in Figure 3.26. At the top of the figure, there is an image of a table and its corresponding binary pattern, which is stored in the system. The bottom of the figure illustrates the three pattern transformations types and how these affect pattern representation in a system. The figure clearly shows that the change in vantage point or characteristics of the pattern can make the pattern look completely different to the pattern recognition algorithms, which makes it very hard for them to recognise and classify these accurately.

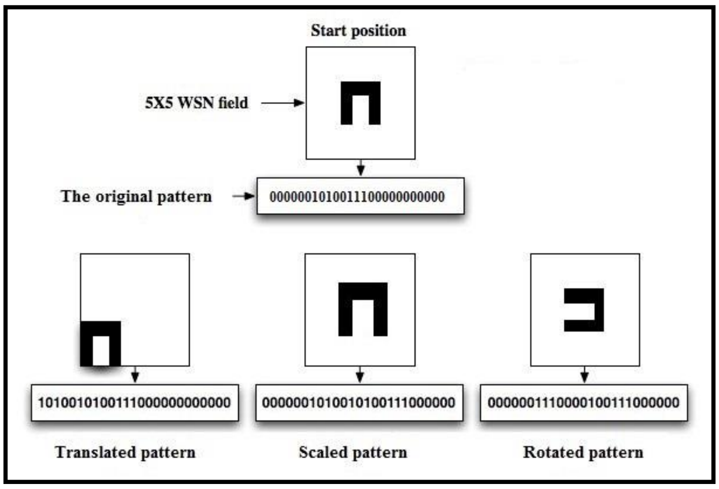

Figure 3.26: A variety of pattern transformations.

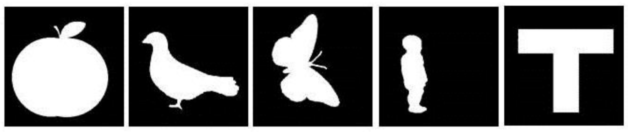

Figure 3.27: A set of binary images patterns used to check the capacities of handling pattern transformations.

A study has been conducted on MOHA's performance with respect to its ability to deal with transformed patterns. Similar to the previous test, the training dataset was constructed by randomly creating and utilising five shapes modelled as a binary image of size 100-by-100 pixels, as shown in Figure 3.27. These images were presented to the MOHA network for memorisation (storing). In this test, we ran a simulated MOHA network of 10000 nodes assuming that nodes are distributed as a grid and each node detects one pixel reading. We also assume that the MOHA network is divided into 100 rows, each of which is utilised to handle 100-bit binary sub-patterns and the threshold values for node activation and edge determination are set to 1 ($\varphi = 1$). In this test, the MOHA’s performance was compared with those of three well-known classifiers, namely Naive Bayes, K-Nearest Neighbours algorithm (KNN) with (*k*=1), and Support Vector Machine (SVM). The Weka tool [263, 264] was utilised to simulate these three schemes. In this test, Naive Bayes (one of the statistical approaches) and SVM have been chosen because of their capabilities in recognising transformations in patterns, as discussed in Section 2.5. Moreover, KNN has been chosen because it is one of the well-known classification methods that has been widely used in many application domains. However, KNN generally experiences high computational complexity due to calculating *k* number of distances. In this test, the *k* value was set to 1 (*k*=1). The main reason for this choice is that we want to compare the proposed schemes with the lowest version of the KNN scheme in terms of computational complexity. Having a lightweight scheme in terms of computational complexity is one of the thesis requirements. As discussed in sub-section 2.3.3, the KNN computational complexity increases for large *k* values [265]. On the other hand, setting the *k* value to 1 might not provide the scheme with optimal performance in terms of accuracy.

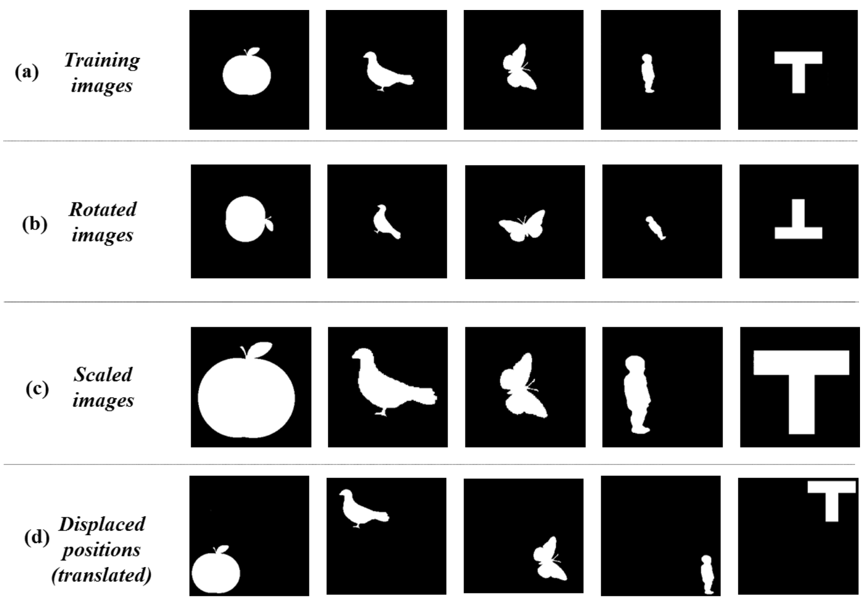

Figure 3.28: The training images and different samples of the testing images: (a) the training images, (b) a sample of rotated images, (c) a sample of scaled images, and (d) a sample of translated images.

3.8.2.1 Pattern rotation

After storing the original binary images patterns, given in Figure 3.28 (a), on the S&I of the MOHA network, each one of these images patterns was rotated, from 1 to 360 degrees with five degrees for each rotation level, and tested to see whether or not the MOHA algorithm was capable of correctly recognising these, see Figure 3.28 (b). Figure 3.29 shows the percentage of perfect recalls within each rotation degree for the MOHA, Naive Bayes, KNN (*k*=1), and SVM algorithms. The same rotated samples of the images were used for all schemes. The accuracy of the schemes is calculated as the total number of correctly recalled patterns as a percentage of the number of altered tested images. The higher the percentage, the higher the accuracy.

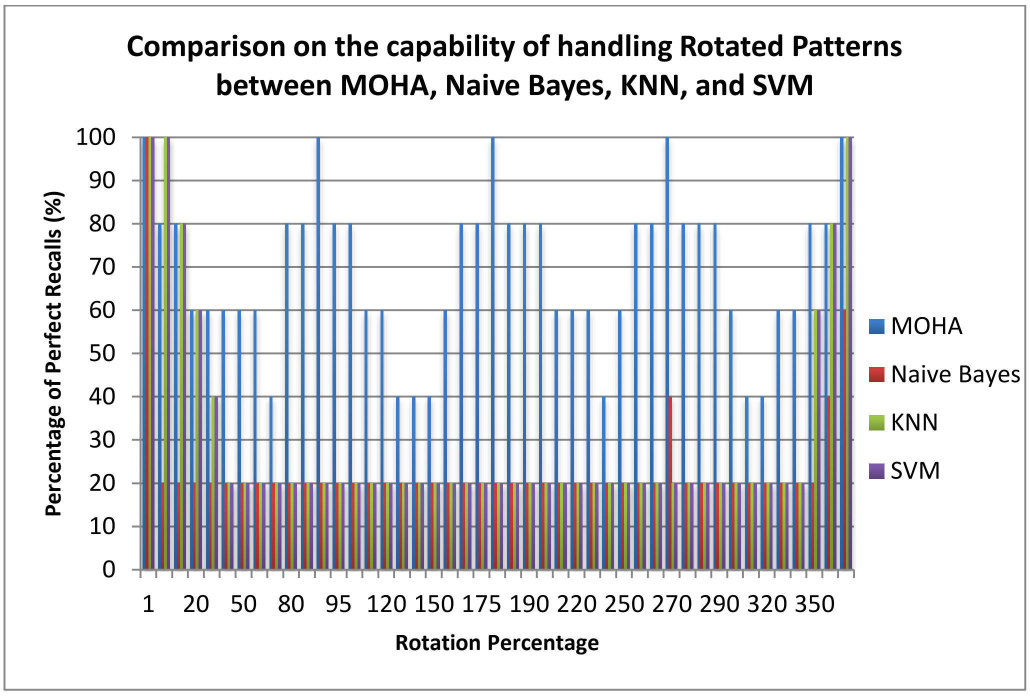

Figure 3.29: Comparison on the capability of handling rotated patterns between MOHA, Naive Bayes, KNN, and SVM.

The results presented in Figure 3.29 clearly show that MOHA algorithm provides more accurate recognition for rotated patterns than do the Naive Bayes, KNN (*k*=1), and SVM algorithms. The results also show that the MOHA offers an average of 312% better recognition accuracy than that achieved by the Naive Bayes, KNN, and SVM algorithms. The main reasons for these results are the small number of training instances (only one trained sample for each image) and the capability of each scheme to deal with these types of transformations. As discussed in Chapter 2, when the Naive Bayes scheme is used to perform pattern recognition, it attempts to build a probabilistic relationship between the training samples and input variables.

Since the number of training samples is very limited, the created probabilistic relationships cannot efficiently describe each pattern [96, 103]. The SVM scheme also requires a large number of training samples in order to create separating hyperplanes between classes and correctly classify patterns, as discussed in section 2.5. Chapter 2 discussions show that KNN does not have the capability of dealing with transformed patterns. Moreover, the results show that the MOHA scheme provides excellent accuracy levels (100% accuracy level) on 90, 180, 270, and 360 degrees rotation and around them. The main reason for this is the limited capability of each node to receive information only from four adjacent neighbours. Thus, by increasing the number of neighbours that each node interacts with, from 4 to 8 or to 16 neighbours, this will enhance the accuracy of the scheme in handling rotated patterns. On the other hand, this increasing number of neighbours will require a greater number of communication exchanges, which will add to the complexity of the MOHA algorithm.

3.8.2.2 Pattern scaling or dilation

This test utilises the same training patterns as those shown in Figure 3.28 (a). In this test, each one of these image patterns was scaled, from 1% to 100% scaling percentages in 5% steps, and tested to see whether or not the MOHA algorithm was capable of correctly recognising these, see Figure 3.28 (c). Figure 3.30 shows the percentage of perfect recalls for each scaled degree for the MOHA, Naive Bayes, KNN (*k*=1), and SVM algorithms. The same scaled samples of the images were used with all schemes.

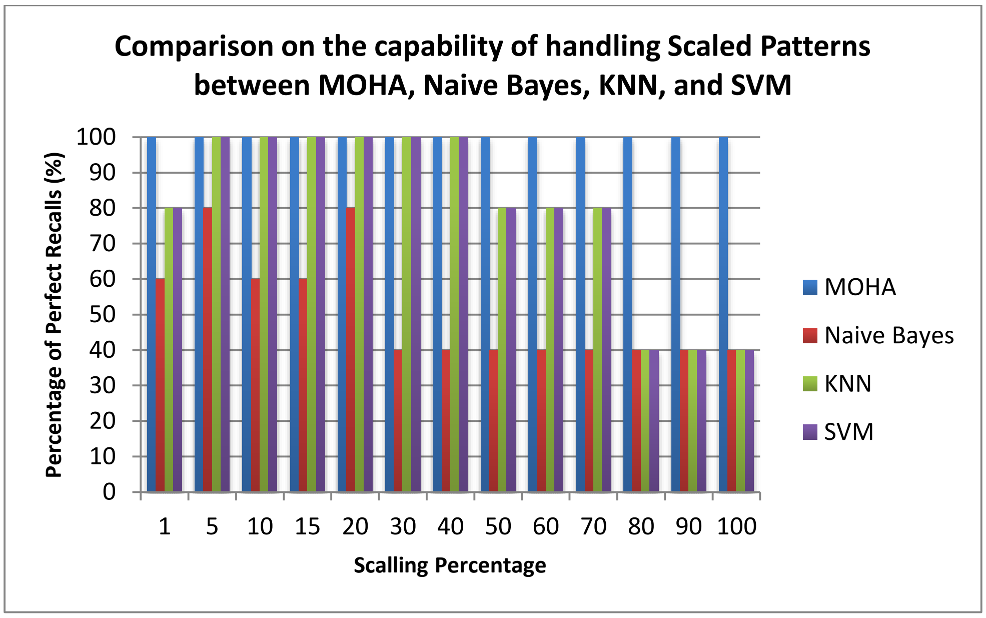

Figure 3.30: Comparison on the capability of handling scaled patterns between MOHA, Naive Bayes, KNN, and SVM.

The results presented in Figure 3.30 clearly show that the MOHA algorithm can provide 100% recognition accuracy in regard to scaled patterns. Thus, the MOHA scheme can correctly recognise different sizes of patterns (scaled patterns) for the same event. The way the MV is calculated in the MOHA algorithm, as the relative distance between all edge nodes (to the critical point) to the distance of the farthest two edge nodes from each other, is the main reason for having this capability to deal with scaled patterns. As been discussed in the previous sub-section, the main reasons for the low results produced by Naive Bayes, KNN, and SVM are the small number of training instances and the capability of each scheme to deal with these types of transformations. Moreover, the results show that KNN and SVM offer reasonable accuracy levels until the patterns are scaled by 50% and then their ability to provide accurate recognition is reduced.

3.8.2.3 Pattern displaced positions or translation

This test also used the same training pattern as the one illustrated in Figure 3.28 (a). In this simulation, each one of these image patterns was randomly translated (displaced) 10 times by shifting the pattern's location, and tested to see whether or not the MOHA algorithm was capable of correctly recognising these, see Figure 3.28 (d). Figure 3.31 shows the percentage of perfect recalls within each translated pattern's location for MOHA, Naive Bayes, KNN (*k*=1), and SVM algorithms. The same translated samples of the images were presented to all schemes.

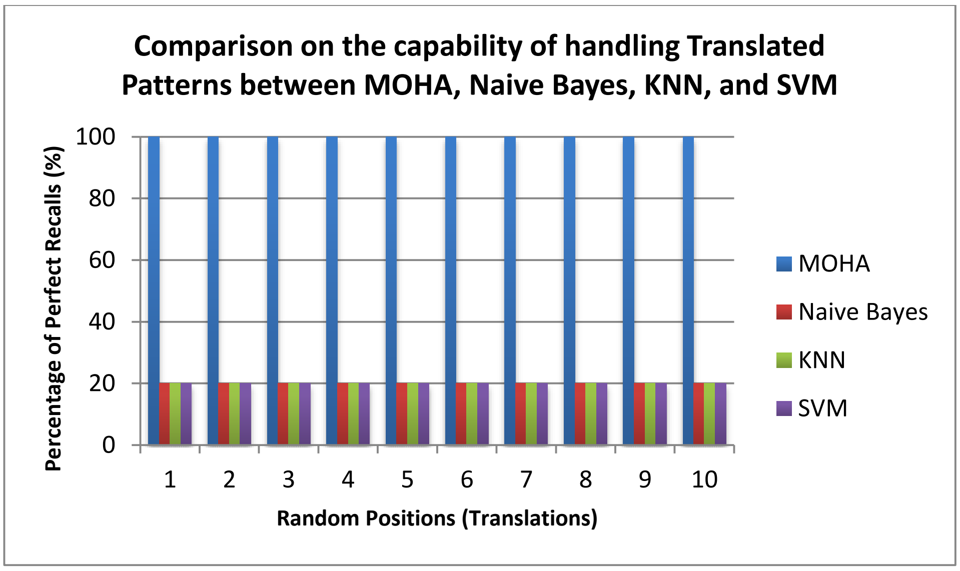

Figure 3.31: Comparison of the capability of handling pattern translation between MOHA, Naive Bayes, KNN, and SVM.

It is clearly seen from Figure 3.31 that the MOHA algorithm can provide 100% recognition accuracy even in the presence of translation issues with the pattern positions. As a result, the scheme provides very high accuracy levels regardless of the positions of patterns in the network. The S&I is responsible for controlling the entire recognition process, making the final decision regarding the patterns and storing the trained patterns. Thus, wherever the patterns appear on the network, the S&I will receive its information and recognise it. The figure also shows that Naive Bayes, KNN, and SVM algorithms have very low accuracy levels when they deal with translated patterns. As discussed in sub-section 3.8.2.1, the main reasons for these low results are the small numbers of training instances and the capability of each scheme to deal with these types of transformations.

3.9 Conclusions

This chapter has presented our proposed approach for pattern recognition, namely, the Macroscopic Object Heuristics Algorithm (MOHA). MOHA implements a divide-and-distribute approach on MOHA networks. Owing to its single-cycle learning and the *in-network* processing features of GN-based algorithms, MOHA is able to offer efficient recognition of patterns with high recall accuracy. Furthermore, MOHA’s pattern recognition scheme operates with low and stable recognition time, due to its ability to provide high recognition accuracy by utilising only the information from the activated edge nodes for recognition.

With regards to algorithmic complexity, MOHA has proven to yield a low-level complexity for its computation tasks. It performs fast recognition incurring single-cycle learning overhead, and takes a highly efficient divide-and-distribute approach. Comparisons with other pattern recognition algorithms including the Hopfield network and DHGN were conducted in order to measure the complexity.

In terms of scalability, two computational factors, time analysis and communication efficiency, were considered. The time analysis of MOHA shows that the MOHA algorithm can be effectively scaled to support any number of nodes in the network without having a noticeable influence on the time required for pattern recognition. In an ideal wireless sensor network scenario in which all communication can occur simultaneously, the MOHA algorithm is highly scalable while introducing a tiny increase in the total time required for pattern recognition. The analysis of the MOHA scheme also shows that the MOHA network is capable of maintaining the high scalability characteristic of the GN network. Furthermore, the analysis shows that the scheme is capable of performing learning operations in predictable time while restricting the learning cycle to be proportional to the number of the activated edge nodes N_{ED}. Such features make the scheme a good candidate for addressing large-scale, real-time problems. Moreover, a comparison has been made between the DHGN approach and our proposed scheme. The results showed that the MOHA could provide, on average, message exchange savings of 26.1% in comparison with DHGN.

A series of recognition tests were conducted on 100-by-100 bit binary images. However, it should be noted that there is no restriction on the overall size of the images being used. Furthermore, the results of the tests have shown that the MOHA algorithm is capable of providing high recall accuracy for noisy binary image recognition. Moreover, it is capable of offering high classification accuracies for varying shapes of the same image. The proposed scheme has been tested to check its ability to handle three different types of pattern transformations. In these tests, the performance of the proposed scheme was compared with those of three well-known classifiers, namely Naive Bayes, KNN, and SVM. The results show that MOHA offers on average 312% better recognition accuracy than Naive Bayes, KNN, and SVM algorithms for the recognition of rotated patterns. The results also show that the MOHA algorithm can provide excellent recognition accuracy in regard to scaled patterns. Furthermore, the MOHA algorithm can provide 100% recognition accuracy for translated patterns.

The computational complexity analysis of the MOHA algorithm shows that it has a very slightly high computational complexity, as indicated in Table 3.6 with respect to the process of determining its longest distances, which is handled by the S&I. It is worth noting that the overall computational complexity of MOHA is less than other existing schemes, like Hopfield network. As a result, a light version of MOHA scheme, known as Light Macroscopic Object Heuristics Algorithm (LMOHA), will be introduced in the next chapter. The main aim of this new version is to reduce the computational complexity of the S&I for pattern recognition. We shall also provide a series of analyses, evaluations and simulations for LMOHA implementation in addition to comparing the new algorithm with the MOHA algorithm.

Chapter 4

Light Macroscopic Object Heuristics Algorithm (LMOHA)

4.1 Preamble

In the previous chapter, the MOHA scheme was presented as a lightweight and distributed pattern recognition scheme that involves a limited number of communications and computations. Such features suit resource-constrained systems and networks such as WSNs. It has been shown experimentally that MOHA is capable of dealing with noisy and transformed patterns. Furthermore, MOHA's pattern recognition scheme operates with low and stable recognition time, compared with other pattern recognition algorithms, due to its ability to provide high recognition accuracy by utilising only the activated edge nodes information for recognition.

However, an analysis of the computational complexity of the MOHA algorithm shows that it has a slightly higher computational complexity with respect to its determination of longest distances, which is handled by the stimulator and interpreter (S&I) (as shown in sub-section 3.7.1). It is worth noting that the overall computational complexity of MOHA is still less than those of other existing schemes, as discussed in sub-section 3.7.1. In this chapter, a lighter version of the MOHA scheme, namely, the Light Macroscopic Object Heuristics Algorithm (LMOHA), will be presented. The main aim of this scheme is to reduce the computational complexity of the MOHA's S&I for event detection and pattern recognition, which will reduce the overall computational complexity of the MOHA scheme. Moreover, the LMOHA scheme offers the same capabilities as the MOHA scheme in dealing with noisy and transformed patterns. As shown with the MOHA in sub-section 3.4.2, the scheme adopts the GN approach to maintain minimal communicational and computational requirements in order to provide a lightweight pattern recognition scheme that suits resource-constrained systems and networks such as WSNs. Instead of using the information about the edges of the pattern for recognition as does the MOHA, the LMOHA searches for the sensory-based shapes of patterns. These sensory-based shapes will be explained in detail in the next section. The main hypothesis in this chapter is that by describing events and patterns using their sensory-based shapes, it is possible to achieve an efficient recognition scheme that has a very low computational complexity and can detect transformed and noisy patterns. The scheme maintains limited communications and computations by involving local information exchange and reporting mechanisms that distribute resource consumption loads amongst the network's nodes. In achieving efficient recognition and fast reporting and by limiting resource consumption, the

LMOHA scheme has demonstrated its suitability for real-life applications that deal with complex problems in resource-constrained networks such as WSNs. This chapter also presents descriptions of the protocols that are required in order to make the proposed scheme applicable for implementation in network environments. Furthermore, a series of analyses, evaluations and simulations for LMOHA implementation is provided along with a comparison between it and MOHA scheme.

The objectives of this chapter are as follows:

1. To propose a lighter version of the MOHA pattern recognition scheme, which is capable of reducing the S&I's computational complexity and detecting transformed and noisy patterns.
2. To perform extensive evaluation and analysis of the complexity of the LMOHA scheme in terms of computations and number of communications. Additionally, the LMOHA scheme will be compared with the MOHA scheme in these terms.
3. To present simulation results of the MOHA and LMOHA schemes in order to ascertain their strengths and limits of tolerance using different types of patterns, especially those that deal with transformed and noisy patterns.

The remainder of this chapter is organised as follows: Section 4.2 discusses the detection of patterns' shapes, focusing on the philosophical grounds for this type of pattern detection. Section 4.3 presents a LMOHA scheme structure for pattern recognition. In this section, the scheme together with its associated components will be described in more detail. Section 4.4 discusses the communicational requirements of the LMOHA scheme and its network communication protocol. Section 4.5 provides an extensive review of the analyses that have been carried out on the LMOHA algorithm. These analyses focus on the complexity of the algorithm and are intended to validate the suitability of the LMOHA for use in WSN environments. A comparative analysis of the LMOHA and MOHA pattern recognition algorithms is also conducted. In Section 4.6, a series of tests conducted on the LMOHA and MOHA schemes are described. The main aims of these tests are to compare the recognition accuracy of the LMOHA and MOHA schemes and to estimate the limits of tolerance of both schemes to transformed and noisy patterns. Section 4.7 provides an overall discussion of the LMOHA scheme.

4.2 The Philosophical Grounds of Patterns' Shapes Detection

Pattern recognition and image processing applications are typically used to recognise and classify objects based on their shape [266-268]. Usually, a fundamental step in the automatic detection and classification of the objects is to discover an object in an image utilising the features related to its shape [266-269]. Hence, shape detection plays an important role and can be used extensively in a variety of applications. As discussed previously in Chapter 3 sub-section 3.3.3, events in WSN can be described as shapes (i.e. objects or boundaries) but not visual shapes. They can be described as sensory-based shapes. The pattern sensory-based shape referred to in this thesis can be described as follows:

Definition 4.1: (the pattern sensory-based shape) Given an active node $N_i^{r_m}$ and its four adjacent nodes $N_{i-1}^{r_m}$, $N_{i+1}^{r_m}$, $N_i^{r_{m+1}}$, and $N_i^{r_{m-1}}$, the pattern sensory-based shape is a description of the relationships between the sensory-based values of the active node's 4 adjacent nodes, which is creating a sensory-based like shape (see Figure 4.1). It is a part of an event's overall sensory-based shape, which can be described as a complex edge.
where i is the node's position in its row and m is the node's row number.

Figure 4.1 shows the difference between the MOHA's proposed technique for edges determination and the new scheme proposed in this chapter (i.e. LMOHA) which is as technique used to determine sensory-based shapes for two binary character patterns (A and E). As shown in Figure 4.1, the latter technique provides extra information about the distribution of the sensory-based values of patterns or events (between the nodes or in the sensed field) compared with the edges determination technique proposed in Chapter 3. This extra information can be used to recognise events and patterns without the need to calculate the distances between the edges of patterns during the recognition procedure (like that of MOHA), which will reduce the computational complexity of the MOHA's S&I and lead to a reduction of the overall computational complexity of the MOHA scheme.

Similar to the MOHA, the knowledge obtained from edge detection techniques in image processing and segmentation will be utilised to recognise events that produce sensory-based shapes on the basis of the sensor readings. The sensory-based shapes that can be created from the relationships between the four adjacent nodes' values are: plus, T, corner, and edge shapes (as shown in Figure 4.1). As a result, the event can be identified by knowing the total appearances of each of these four sensory-based shapes in the complete pattern.

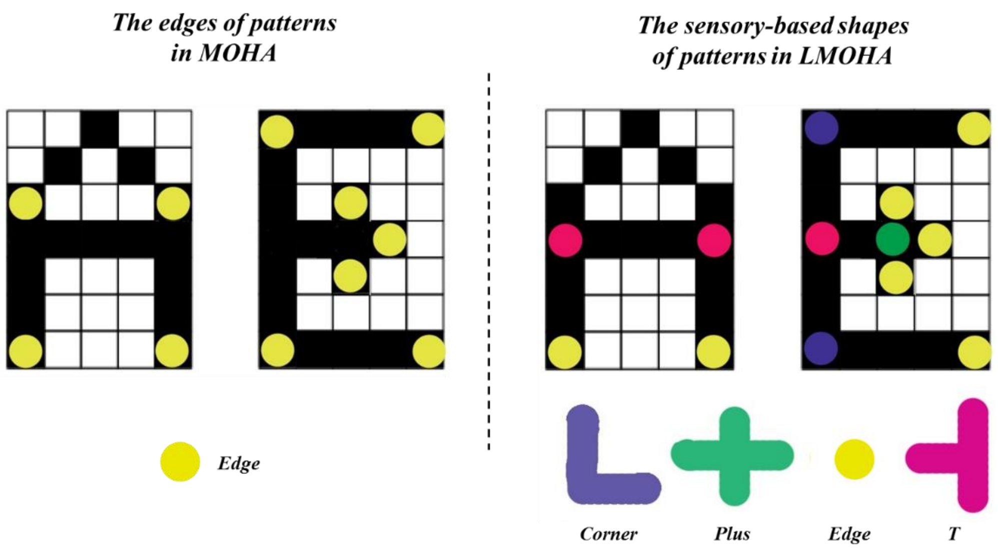
The edges of patterns in MOHA
The sensory-based shapes of patterns in LMOHA
Edge
Corner
Plus
Edge
T

Figure 4.1: Two-binary character patterns with their activated edges in MOHA and their activated shapes in LMOHA.

A pattern is divided into sub-patterns and then the sensory-based shapes of these sub-patterns are derived and stored for subsequent recognition. Thus, by identifying the frequency of occurrences on a pattern of each of these sensory-based shapes, the pattern recognition algorithm will be capable of recognising the entire pattern without requiring a huge amount of processing power. The main hypothesis in this chapter is that by describing events and patterns using their sensory-based shapes, it is possible to achieve an efficient recognition scheme that has a very low computational complexity and can detect transformed and noisy patterns. Sub-section 4.3.2 provides a detailed discussion of the way in which the proposed scheme can determine these sensory-based shapes in events or patterns.

4.3 Overview of Light Macroscopic Object Heuristics Algorithm (LMOHA)

This section describes the structure of the LMOHA and the outcomes that can be expected from such architecture. Similar to the MOHA, the main goal of developing the LMOHA scheme is to provide efficient pattern recognition for WSNs while minimising resource consumption and network size, which is capable of detecting events, transformed, and noisy patterns. Moreover, similar to the MOHA, the LMOHA scheme is based on coding techniques, which are defined in [255] as reproducing an incoming pattern and using a small number of active nodes for processing at any given time. To achieve the intended level of detection and minimise resource consumption, the LMOHA implements a local adjacency-based relationships mechanism for coding purposes in a fully distributed and parallel manner that is capable of detecting events, noisy patterns, and pattern transformations such as translation, scaling (i.e. dilation) and rotation with minimal computational and communication requirements.

The LMOHA utilises this mechanism to locate and determine the sensory-based shapes of an incoming pattern. The LMOHA utilises only those nodes located on the locations of the sensory-based shapes (i.e. plus, T, corner and edge shapes) of a pattern in the recognition process, which minimises the number of sensor nodes required for recognition. Plus, T, corner and edge nodes are determined locally and only once, based on the relationship between their neighbours' values. A detailed discussion of the way in which the proposed scheme can determine these types of nodes will be provided in sub-section 4.3.2. These nodes report their type information to the S&I in order to obtain a final decision about an incoming pattern. The hypothesis underlying this approach is that events and patterns can be efficiently recognised based on information about their sensory-based shapes. Since these shapes are detected by local nodes' computations, the number of communications and amount of resources required is minimised.

Figure 4.2 illustrates the LMOHA model. Similar to the MOHA, this model consists of two main entities: the LMOHA network and the stimulator and interpreter (S&I). The S&I is responsible for sending commands to the LMOHA network. It receives the nodes' types information, from nodes located on the locations of the sensory-based shapes (i.e. plus, T, corner and edge shapes) of a pattern, and makes a final decision about an incoming pattern. A command message can be to either memorise or recall a pattern. It also includes information about the pattern, or commands the network to obtain sensory information. The LMOHA network nodes process the S&I command

and reply with information about the nodes' types. The S&I uses this information to memorise or recall the sensed or sent pattern.

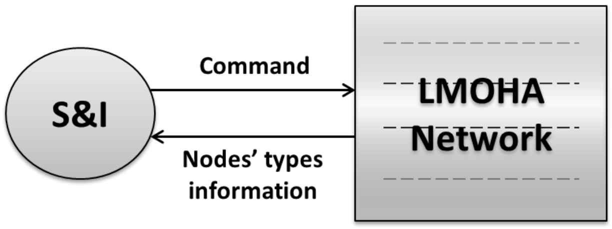

Figure 4.2: The LMOHA communication model.

4.3.1 LMOHA network

The LMOHA network structure is similar to that of the MOHA network which is presented in Chapter 3. The aim of the network structure is to: have a scheme with low pattern recognition complexity, provide parallel processing, provide efficient recognition, and have a predetermined duration for the learning and recognition cycle. A fully distributed structure will produce a scheme with low complexity that allows sensor nodes in the network to communicate only with adjacent nodes. The LMOHA's structure allows each node to communicate with four adjacent nodes in order to determine whether it represents a pattern sensory-based shape (i.e. plus, T, corner or edge). In the LMOHA scheme, only plus, T, corner and edge nodes' information are utilised for pattern recognition. This minimises the number of sensor nodes required for recognition. Efficient recognition is provided by describing the patterns' sensory-based shapes in order to provide the scheme with a transformation-invariant recognition feature. The purpose of using plus, T, corner and edge nodes types information is to allow the detection of any particular pattern's sensory-based shapes by any node in the network. In other words, the detection of any desired pattern's shapes is not associated with static nodes. Instead, any node in the network is expected to be able to determine the same sensory-based shape's type (i.e. node type) based on its adjacent nodes' values. By using specific steps to determine plus, T, corner and edge nodes and reporting information about their types, the LMOHA will have a single learning and recognition time cycle that can be predicted and estimated. Once all the information about plus, T, corner and edge node types has been delivered to S&I, the LMOHA does not need further information from sensor nodes in order to declare the detection of a specific event or pattern. This feature reduces the need for communications between S&I and participant nodes in the network. Figure 4.3 shows that communication within the LMOHA scheme occurs in a single-cycle environment, wherein, each pattern is passed via the network only once.

Similar to the MOHA, the network structure depends on the size of the problem pattern and the size of the sub-pattern (determined according to the system user or the WSN application requirements). The deployment of the network uses the deployment algorithm described in Algorithm 3.1. The deployment of the network begins by implementing a number of nodes in the first row equal to the sub-pattern size (S_{sb}), according to Equation 3.15. Then, the second row nodes (equal to S_{sb}) are implemented and so on until all nodes in all rows have been deployed in the network. The LMOHA network contains many sensor nodes distributed in the field of interest in a certain multi-row manner as shown in Figure 4.4. It is important to highlight that node deployment in this sub-section is a logical deployment method. In other words, deployment can be implemented by assigning each node to its row and its position in this row.

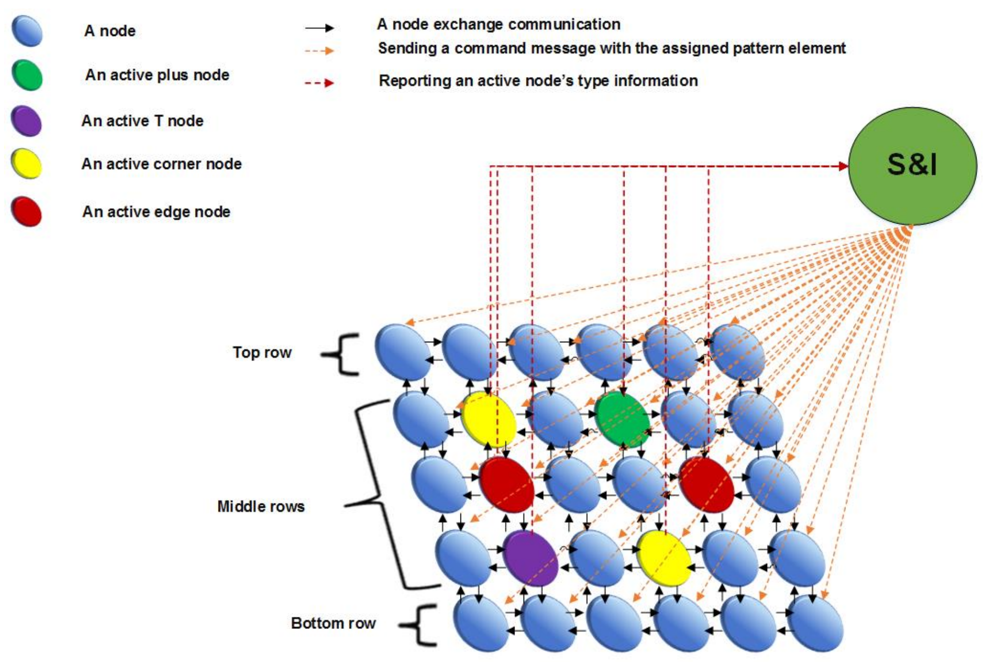
A node
An active plus node
An active T node
An active corner node
An active edge node
A node exchange communication
Sending a command message with the assigned pattern element
Reporting an active node's type information
S&I
Top row
Middle rows
Bottom row

Figure 4.3: LMOHA framework for pattern recognition.

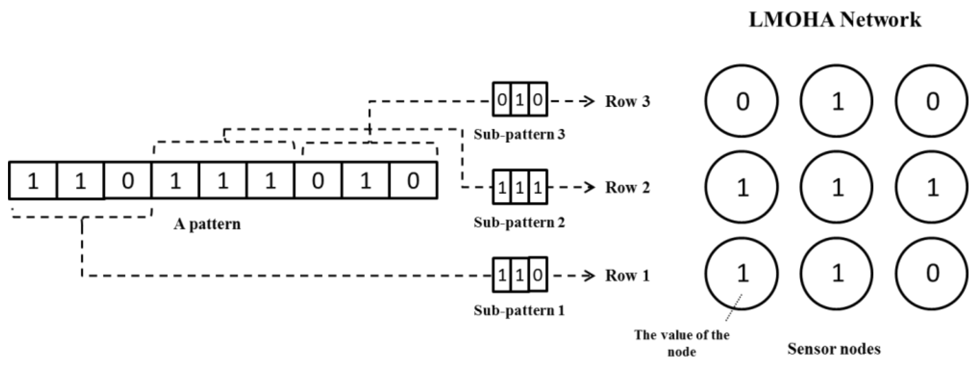

Figure 4.4: An example of an LMOHA network consisting of 3 rows and assigned with a 9-bit pattern size.

The network structure consists of multiple rows. A row is a GN array that consists of a set of nodes of the form ($S_i = S_{sb}$), where S_i is the size of the row number i (i.e. the number of nodes in row i). As discussed in Chapter 3, by involving multiple rows in the network structure, this enables parallel processing and information exchange of incoming data by dividing the pattern into a set of sub-patterns. This will also allow the network to deal with different pattern types that require multi-dimensional processing.

The network row according to Definition 3.3 is a GN network that consists of a set of nodes where each node communicates with its adjacent nodes in the same row and in the higher and lower rows. The communications between nodes in rows are called *exchange communications* (Definition 3.4). Similar to the MOHA, normally each node has four adjacent nodes with which it communicates. However, the actual number of adjacent nodes and the communication exchanges of a node vary from two to four based on its location or position in the row and whether the row is located at the base, the middle, or the top of the network, as discussed in sub-section 3.4.2. The main purpose of having four adjacent nodes is to enable the scheme to recognise transformed patterns, especially the rotated ones, as the scheme will obtain a better description about the incoming pattern and its distribution.

Each node in the LMOHA network receives its assigned command and pattern element, exchanges information with adjacent nodes, determines its type, and sends its type information to the S&I. The four commands, $\{M, p\}$, $\{M, S\}$, $\{R, p\}$, and $\{R, S\}$ described in Table 3.3 are used to train the LMOHA network and recognise patterns. Similar to the MOHA, based on data sensed or received from S&I, each node is activated or de-activated according to the activation criteria stated in Definition 3.2. Another round of the activation process is performed by each node after exchanging communications. Based on the values received from the adjacent nodes, a node can decide whether or not to be activated. If the node obtains a pattern's sensory-based shape values, it is activated. Only activated nodes participate in the pattern detection process so as to limit the use of node resources and to reduce the detection time by limiting the number of communicating nodes.

One of the reasons for having a multi-row structure in the LMOHA network is to provide each node in the network with four adjacent nodes with which it can exchange information. This enables a node to have an extended view of the incoming pattern, which allows it to determine whether or not it represents a pattern sensory-based shape (i.e. plus, T, corner or edge). The activation of a node (Definition 3.2) triggers the start of an exchange communication process for that node with its adjacent neighbours, as stated in Definition 3.4. After receiving information from all adjacent nodes, the activated node determines whether it represents a pattern sensory-based shape according to Equation 4.3. Based on the resulting node type, it either de-activates or goes for the second level of activation called 'node types activation'. There are four types of node activations in LMOHA implementation, namely *active plus node*, *active T node*, *active corner node*, and *active edge node*. These node activation types are presented in the following:

Definition 4.2: (the active plus node) Given an active node $N_i^{r_m}$ and its four adjacent nodes $N_{i-1}^{r_m}$, $N_{i+1}^{r_m}$, $N_i^{r_{m+1}}$, and $N_i^{r_{m-1}}$, the node will be activated as a plus node if its node type is $plus$ ($node_{type} = plus$).

where i is the node's position in its row, m is the node's row number, and $node_{type}$ is the node's type, which is determined according to Equation 4.3. The procedure to determine a node's type is explained in detail in the next sub-section.

Definition 4.3: (the active T node) Given an active node $N_i^{r_m}$ and its four adjacent nodes $N_{i-1}^{r_m}$, $N_{i+1}^{r_m}$, $N_i^{r_{m+1}}$, and $N_i^{r_{m-1}}$, the node will be activated as a T node if its node type is T ($node_{type} = T$).

where i is the node's position in its row, m is the node's row number, and $node_{type}$ is the node's type, which is determined according to Equation 4.3.

Definition 4.4: (the active corner node) Given an active node $N_i^{r_m}$ and its four adjacent nodes $N_{i-1}^{r_m}$, $N_{i+1}^{r_m}$, $N_i^{r_{m+1}}$, and $N_i^{r_{m-1}}$, the node will be activated as a corner node if its node type is $corner$ ($node_{type} = corner$).

where i is the node's position in its row, m is the node's row number, and $node_{type}$ is the node's type, which is determined according to Equation 4.3.

Definition 4.5: (the active edge node) Given an active node $N_i^{r_m}$ and its four adjacent nodes $N_{i-1}^{r_m}$, $N_{i+1}^{r_m}$, $N_i^{r_{m+1}}$, and $N_i^{r_{m-1}}$, the node will be activated as an edge node if its node type is $edge$ ($node_{type} = edge$).

where i is the node's position in its row, m is the node's row number, and $node_{type}$ is the node's type, which is determined according to Equation 4.3.

In the LMOHA scheme, only the information about activated plus, T, corner and edge nodes is used for pattern recognition. Therefore, all plus, T, corner and edge nodes in the network are required to send their types information to the S&I for further analysis and recognition. The communications between the LMOHA network's plus, T, corner and edge nodes and the S&I are called *report communications* and can be described as follows:

Definition 4.6: (Report communications) Given a LMOHA network that consists of m nodes, a report communication of an active plus, T, corner, or edge node is the message (connection) between a plus, T, corner, or edge node in the network and the S&I that contains the node's type information, in the form: $PN_i^{r_m} \rightarrow SI\colon \{type\}$, $TN_i^{r_m} \rightarrow SI\colon \{type\}$, $CN_i^{r_m} \rightarrow SI\colon \{type\}$, or $EN_i^{r_m} \rightarrow SI\colon \{type\}$.

where $PN_i^{r_m}$ is the communicating (activate) plus node in a row number m with a position number i, $TN_i^{r_m}$ is the communicating (activate) T node in a row number m with a position number i, $CN_i^{r_m}$ is the communicating (activate) corner node in a row number m with a position number i, $EN_i^{r_m}$ is the communicating (activate) edge node in a row number m with a position number i, SI is the stimulator and interpreter (S&I), and $type$ is the node's type information. Figure 4.3 shows the report communications occurring between 6 activated nodes, 1 plus, 1 T, 2 corner, and 2 edge nodes, and the S&I.

In *report communications*, in regards to the communication cost, the total number of messages communicated from plus, T, corner and edge nodes to the S&I, N_{report}^{msg} is equivalent to the number of activated plus, T, corner and edge nodes, N_{plus}, N_T, N_{corner}, and N_{edge}, as given by the following equation:

$$N_{report}^{msg} = N_{plus} + N_T + N_{corner} + N_{edge} \tag{4.1}$$

Figure 4.5 shows the steps that each node in the network performs in the learning process. It is worth noting that nodes in both the MOHA and the LMOHA networks perform the same first three steps. These steps are:

1. **Receive command:** The node receives the broadcasted command from S&I that contains the operation (memorise or recall) and the pattern element obtaining method (direct receive or sense). Further details about this command message are presented in section 3.5.
2. **Obtain pattern element:** Based on the command message, each node begins to receive its assigned pattern element p_i. Each node sets its value (v) according to the obtained pattern element. If a node's value complies with certain user-defined conditions (e.g. reaching a certain threshold), it is activated, according to Definition 3.2.

3. **Exchange communications:** Each activated node performs exchange communications with its neighbouring nodes, as described in Definition 3.4. Then, the activated node determines whether it represents a pattern sensory-based shape according to Equation 4.3. Based on the equation result, it either de-activated or activated as a plus, T, corner or edge node, as described in Definitions 4.2-4.5.
4. **Report communications:** Each activated plus, T, corner and edge node in the network reports its type information to the S&I, as described in Definition 4.6.

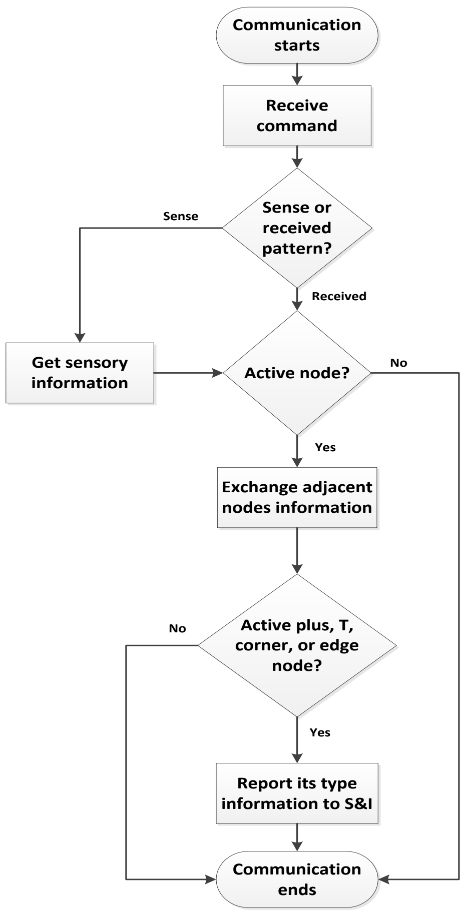
Communication starts
Receive command
Sense or received pattern?
Sense
Received
Get sensory information
Active node?
No
Yes
Exchange adjacent nodes information
Active plus, T, corner, or edge node?
No
Yes
Report its type information to S&I
Communication ends

Figure 4.5: LMOHA network node operations.

4.3.2 Determining the activated node type (pattern shape search)

The LMOHA scheme represents patterns in terms of compositions of values, which are the sequence numbers of each activated node type. The main goals of the proposed approach are to reduce the overall computational complexity of the scheme recognition process and to enable the detection of pattern transformation (e.g. rotation, displaced positions (translation), and scaling (dilation)). To achieve these goals, the LMOHA scheme searches for plus, T, corner, and edge shapes in the pattern's data domain. An active node type is determined according to the values received from its adjacent nodes using the node's exchange communications (see Definition 3.4). The LMOHA scheme describes the relationships between the sensory-based values received from the active node's 4 adjacent nodes as sensory-based shapes (see Figure 4.1). Similar to the MOHA, the knowledge obtained from edge detection techniques in image processing and segmentation, based on the Sobel edge detection operator to be more specific, will be utilised to recognise events that produce sensory-based shapes on the basis of the sensor readings.

There are five types of active nodes: the active plus node ($plus$), where an active node's gradient magnitude ($\left|G(N_i^{rm})\right|$) is almost equal zero ($\left|G(N_i^{rm})\right| \approx 0$). The active node's gradient magnitude can be calculated using Equation 3.28 in the previous chapter. This happens only when the sensory-based values of the active node's four adjacent nodes are almost identical, which creates a sensory-based plus-like shape (see Figure 4.1); the active T node (T), where an active node's gradient magnitude is greater than zero and equal or less than the threshold value for edge determination (φ), $0 < \left|G(N_i^{rm})\right| \leq \varphi$, and the angle of orientation of the edge (relative to LMOHA network structure) is equal to 0 degree ($\theta\left(G\left(N_i^{rm}\right)\right) = 0^{\circ}$). This happens only when the sensory-based values of the three adjacent nodes are almost identical, which creates a sensory-based T like shape (see Figure 4.1); the active corner node ($corner$), where an active node's gradient magnitude is greater than the threshold value for edge determination ($\left|G(N_i^{rm})\right| > \varphi$) and the angle of orientation of the edge is equal to 45 degrees ($\theta\left(G\left(N_i^{rm}\right)\right) = 45^{\circ}$). This happens when the active node's gradients of both orientations (*x*,*y*) have high and equal values, which create a sensory-based corner-like shape (see Figure 4.1); and the active edge node ($edge$), where an active node's gradient magnitude is greater than the threshold value for edge determination ($\left|G(N_i^{rm})\right| > \varphi$) and the angle of orientation of the edge is equal to 0 degree ($\theta\left(G\left(N_i^{rm}\right)\right) = 0^{\circ}$). This happens when only one orientation (*x* or *y*) of the active node's gradients have a highly noticeable value, which creates a sensory-based edge-like shape (see Figure 4.1); otherwise the node type will be set to $none$ and deactivated.

The angle of orientation of the edge (relative to LMOHA network structure) $\theta\left(G\left(N_i^{r_m}\right)\right)$, can be calculated using the following equation or piecewise function [202, 207, 256]:

$$\theta\left(G\left(N_i^{r_m}\right)\right) = \begin{cases} \arctan\left(\frac{Gy\left(N_i^{r_m}\right)}{Gx\left(N_i^{r_m}\right)}\right), & if\ Gx\left(N_i^{r_m}\right) > 0 \\ 0, & Otherwise \end{cases} \tag{4.2}$$

An active node type is determined by the following function of the node value and adjacent node values: $node_{type} = f(N_i^{r_m}(v), N_{i+1}^{r_m}(v), N_{i-1}^{r_m}(v),$ $N_i^{r_{m+1}}(v), N_i^{r_{m-1}}(v))$, where $node_{type}$ is the active node type, $N_i^{r_m}(v)$ is the value of the current node, $N_{i+1}^{r_m}(v)$ is the value of the next node in the same row, $N_{i-1}^{r_m}(v)$ is the value of the previous node in the same row, $N_i^{r_{m+1}}(v)$ is the value of the adjacent node in the next higher row, and $N_i^{r_{m-1}}(v)$ is the value of the adjacent node in the lower row. This relationship can be described through the following piecewise function:

$$node_{type} = \begin{cases} plus, & if\ \left|G\left(N_i^{r_m}\right)\right| \approx 0 \\ T, & if\ 0 < \left|G\left(N_i^{r_m}\right)\right| \leq \varphi\ and\ \theta\left(G\left(N_i^{r_m}\right)\right) = 0^\circ \\ corner, & if\ \left|G(N_i^{r_m})\right| > \varphi\ and\ \theta\left(G\left(N_i^{r_m}\right)\right) = 45^\circ \\ edge, & if\ \left|G(N_i^{r_m})\right| > \varphi\ and\ \theta\left(G\left(N_i^{r_m}\right)\right) = 0^\circ \\ none, & Otherwise \end{cases} \tag{4.3}$$

Now, we will present some examples which cover all node types and corresponding pattern distributions.

Example 1: In pattern $P_1 = \{300{,}400{,}350{,}400{,}300{,}400{,}350{,}400{,}450\}$, 9 elements have to be split into 3 sub-patterns of 3 elements with each sub-pattern assigned to a different LMOHA row (see Figure 4.4) and the threshold value for edge determination is set to 500 ($\varphi = 500$). In this example, the middle node in the second row is categorised as a $plus$, as its gradient is equal to zero ($\left|G(N_i^{rm})\right| = 0$). This means that the values of its adjacent nodes are identical.

Example 2: In pattern $P_2 = \{500,300,350,400,300,400,500,400,350\}$, the middle node in the second row is categorised as a T, as its gradient is equal to 200 (less than the threshold value for edge determination $\varphi = 500$) and the angle of orientation of the edge is equal to zero degree, $\left|G(N_i^{rm})\right| = 200\ and\ \theta\left(G\left(N_i^{rm}\right)\right) = 0^{\circ}$. This means that the sensory-based values of only 3 adjacent nodes are identical.

Example 3: In pattern $P_3 = \{500,350,350,500,300,350,400,500,350\}$, the middle node in the second row is categorised as a $corner$, as its gradient is equal to 600 (greater than the threshold value for edge determination $\varphi = 500$) and the angle of orientation of the edge is equal to 45 degrees, $\left|G(N_i^{rm})\right| = 600\ and\ \theta\left(G\left(N_i^{rm}\right)\right) = 45^{\circ}$. This means that the gradients of both orientations (*x*,*y*) have high and equal values.

Example 4: In pattern $P_4 = \{500,400,350,500,300,200,500,400,350\}$, the middle node in the second row is categorised as an $edge$, as its gradient is equal to 600 (greater than the threshold value for edge determination $\varphi = 500$) and the angle of orientation of the edge is equal to 0 degree, $\left|G(N_i^{rm})\right| =$

$600\ and\ \theta\left(G\left(N_i^{rm}\right)\right) = 0^{\circ}$. This means that only one orientation (x or y) of the node' gradients has highly noticeable value.

Example 5: In pattern $P_5 = \{400,400,400,400,350,300,450,300,400\}$, the middle node in the second row is categorised as a $none$, as its gradient is equal to 400 (less than the threshold value for edge determination $\varphi = 500$) and the angle of orientation of the edge is equal to 45 degrees, $\left|G(N_i^{rm})\right| = 400\ and\ \theta\left(G\left(N_i^{rm}\right)\right) = 45^{\circ}$. This means that its values do not meet the criteria for any of the defined node type conditions.

4.3.3 Stimulator and interpreter (S&I)

The main objective of the second version of the proposed scheme is to reduce the computational complexity of the S&I for pattern recognition, which will lead to a reduction of the overall computational complexity of the scheme. The S&I component is responsible for sending commands to the LMOHA network, receiving the types information of the activated nodes, and making the final decision about an incoming pattern. The LMOHA uses the same pattern obtaining method as MOHA, as described in section 3.5. Hence, the four commands, $\{M, p\}$, $\{M, S\}$, $\{R, p\}$, and $\{R, S\}$, described in Table 3.3 are used to train a LMOHA network and recognise patterns. The LMOHA network nodes process the S&I command and respond with information about the activated node types. This information about the activated node types indicates the pattern's sensory-based shapes. The method used to determine the activated node type was discussed previously in sub-section 4.3.2. The S&I uses the information about the activated node types to memorise or recall the sensed or received patterns. After the S&I receives this information from the LMOHA network, it begins the process of memorising or recalling a pattern. In memorisation, the S&I assigns a unique index number to the pattern, associates this number with the total received numbers of each types of activated nodes (sensory-based shapes), and stores the index number and the associated total numbers of each type of activated node in its memory. This results in a set of patterns stored in a vector that can be described as follows:

Definition 4.7: (Pattern vector) Given a set of patterns $\{P_1, P_2, \dots, P_t\}$, the S&I memorises these patterns by obtaining each pattern's total numbers of activated plus nodes (PN_i), activated T nodes (TN_i), activated corner nodes (CN_i), and activated edge nodes (EN_i) from a LMOHA network, assigning a unique index number (I_i) to each pattern, and storing the associations of patterns, the total numbers of activated plus nodes, activated T nodes, activated corner nodes, and activated edge nodes as a pattern vector in the S&I in the following form:

$$\vec{P} = \{(I_1, PN_1, TN_1, CN_1, EN_1), (I_2, PN_2, TN_2, CN_2, EN_2), \dots, (I_t, PN_t, TN_t, CN_t, EN_t)\}$$
$$, I_i, PN_i, TN_i, CN_i EN_i \in \mathbb{N} \tag{4.4}$$

In recall, the S&I searches the pattern vector to find a match. The declaration that a pattern has been detected depends on the total differences between the LMOHA network's activated plus, T, corner, and edge nodes numbers and the activated plus, T, corner, and edges nodes numbers stored in the pattern vector as follows:

Definition 4.8: (Recalled pattern) Given numbers of activated plus, T, corner, and edge nodes determined by the LMOHA network (PN_c, TN_c, CN_c, EN_c), and a pattern vector, the recalled pattern (RP) will be the index number with the smallest differences between the determined plus, T, corner, and edge nodes numbers and the set of plus, T, corner, and edge nodes numbers stored in the pattern vector as follows:

$$\begin{aligned} RP = I[\min(&\Delta PN_{1c} + \Delta TN_{1c} + \Delta CN_{1c} + \Delta EN_{1c}, \Delta PN_{2c} + \Delta TN_{2c} + \Delta CN_{2c} \\ &+ \Delta EN_{2c}, \dots, \Delta PN_{tc} + \Delta TN_{tc} + \Delta CN_{tc} + \Delta EN_{tc})] \end{aligned} \tag{4.5}$$

where ΔPN_{ic} is the difference between the i^{th} stored pattern number of activated plus nodes and the number of activated plus nodes determined by the LMOHA network for an incoming pattern (current pattern), ΔTN_{ic} is the difference between the i^{th} stored pattern number of activated T nodes and the number of activated T nodes determined by the LMOHA network for an incoming pattern, ΔCN_{ic} is the difference between the i^{th} stored pattern number of activated corner nodes and the number of activated corner nodes determined by the LMOHA network for an incoming pattern, and ΔEN_{ic} is the difference between the i^{th} stored pattern number of activated edge nodes and the number of activated edge nodes determined by the LMOHA network for an incoming pattern.

Moreover, it is clearly shown from this sub-section that the LMOHA completely removes the need to use the MOHA's distance measurements for pattern recognition, which is the main reason for the MOHA's slightly high computational complexity. Moreover, utilising distance measurements for pattern recognition (as in MOHA and KNN) is not recommended as it might lead to inaccurate classification, as discussed in Chapter 2.

Figure 4.6 illustrates the S&I operations which can be summarised as follows:

1. **Send command:** The S&I initiates the LMOHA learning process by sending a command to the LMOHA network's nodes. This command contains the operation type (memorise or recall) and the pattern obtaining method (direct receive or sense), as discussed in Section 3.5.
2. **Receive the node type information:** The S&I receives the types information of the activated nodes from the LMOHA network.
3. **Memorise or recall:** If the operation is to memorise the incoming pattern, the S&I creates a unique number, associates this index number with the total received numbers of activated plus, T, corner, and edge nodes, and stores this association in its pattern vector. However, if the operation is to recall the incoming pattern, the S&I searches for the closest numbers of activated plus, T, corner, and edge nodes in its pattern vector to the incoming pattern's numbers of activated plus, T, corner, and edge nodes and declares its associated pattern as RP.

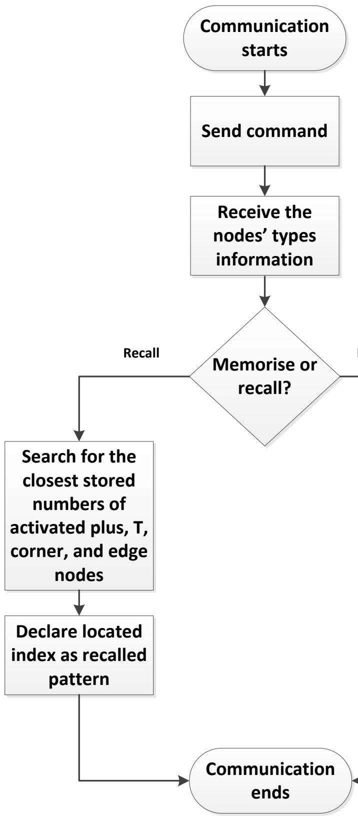
Communication starts
Send command
Receive the nodes' types information
Memorise or recall?
Recall
Memorise
Search for the closest stored numbers of activated plus, T, corner, and edge nodes
Declare located index as recalled pattern
Assign new index number
Store index with the total numbers of activated plus, T, corner, and edge nodes
Communication ends

Figure 4.6: S&I operations for memorising and recalling patterns.

4.4 LMOHA Communication Requirements and Protocol

In this section, the communication requirements and protocol of the LMOHA network are described. To perform LMOHA network node communications, this study assumes that a medium access control (MAC) protocol is present and available to support the network. By using MAC protocols, each sensor node can have a unique MAC address that differentiates it from other sensor nodes, allowing direct communication between two sensor nodes.

Similar to the MOHA, during the initialisation of an LMOHA network, each sensor node should be provided with a row number and a position number on which it will work. This allows the sensor node to determine its communication process. Determining a sensor node’s row and position can be performed statically or automatically during the initialisation phase of the LMOHA network. Static row and position determination means that each sensor node is provided with information about its row, position, and adjacent nodes with which to exchange information. This initialisation would be less complex in terms of computations and communications. However, the flexibility of adding new sensor nodes to the network or adopting dynamic changes such as mobile nodes or clusters will be limited. Automatic row and position determination can be achieved by allowing each sensor node to communicate with its neighbouring nodes and allowing the S&I (in the base station) to determine its row, position, and adjacent nodes after the deployment of the network. This approach will provide more network flexibility, allowing it to adapt to changes that may be required in the network design. However, this will lead to an increase in the number of communications in the network.

To ensure the functionality of the LMOHA network, each sensor node should be fed with sufficient information about how to react to failures or de-activated neighbouring sensor nodes. For adjacent sensor nodes, a sensor node should assume the value of a failed communicating node to be equal to its own value. This is to avoid interruption of the recognition process and eliminate the effects of these failed nodes on the process of determining the node type.

The LMOHA communication protocol describes the main steps of LMOHA network communications. By adhering to this protocol, the LMOHA network will memorise or recall an incoming pattern. The protocol consists of three main steps as follows:

Step 1: $\boldsymbol{S\&I} \boldsymbol{\Rightarrow} \boldsymbol{N_1}, \boldsymbol{N_2}, \dots, \boldsymbol{N_m}: (C_i, P_i)\ \{(C_1, P_1), (C_2, P_2), \dots, (C_m, P_m)\}$

In the first step of the communication, the S&I (in the base station) sends the command C_i and the pattern elements P_i to each sensor node in the network. As explained in section 3.5, a command can be either to memorise (M) or recall (R) and the pattern element can be a value to use for training, or (S) to initiate sensors to use the sensory information. Each sensor node receives only one element of the pattern. For instance, if the pattern is (2,5,8,6), the S&I will send the values 2, 5, 8, and 6 to the sensor nodes 1, 2, 3, and 4 as pattern elements respectively. Sensory information can be obtained by two methods. In the first scenario, the S&I sends a memorise or recall command to the sensor nodes in order to start obtaining sensory information and continue the communication steps. In the second scenario, the sensor nodes are programmed to obtain information periodically.

Step 2: $\boldsymbol{N_i^{r_m}} \boldsymbol{\Rightarrow} \boldsymbol{N_{i-1}^{r_m}}, \boldsymbol{N_{i+1}^{r_m}}, \boldsymbol{N_i^{r_{m+1}}}, \boldsymbol{N_i^{r_{m-1}}}: v_i$

Each activated sensor node in the network $N_i^{r_m}$ starts the information exchange process with adjacent sensor nodes. After receiving or obtaining the pattern element information, each activated sensor node sends its value v_i to four adjacent sensor nodes: previous node in its row $N_{i-1}^{r_m}$, next node in its row

$N_{i+1}^{r_m}$, adjacent node in the next higher row $N_i^{r_{m+1}}$, and adjacent node in the lower row $N_i^{r_{m-1}}$. The main aim of this step is to allow each sensor node to determine whether it represents a pattern sensory-based shape according to Equation 4.3. Based on the equation result, it either de-activated or activated as a plus, T, corner, or edge node, as described in Definitions 4.2-4.5. After completing this step, each sensor node will receive four values from its adjacent nodes representing the pattern elements by which a sensor node should be able to determine whether it will be activated as plus, T, corner, or edge node or not.

Step 3: $\boldsymbol{PN_c, TN_c, CN_c, EN_c} \rightarrow \boldsymbol{S}\&\boldsymbol{I}: type_c$

Each activated plus, T, corner, and edge node (PN_c, TN_c, CN_c, EN_c) reports its type information to the S&I (in the base station) in order to store or recall the pattern. The S&I holds a database of trained patterns associated with their total numbers of activated plus, T, corner, and edge nodes. If the pattern needs to be memorised, the S&I assigns a new index number to it and associates this index number with the total numbers of activated plus, T, corner, and edge nodes obtained by the network. If the pattern is to be recalled, the S&I searches its database to find the closest value to the total numbers of activated plus, T, corner, and edge nodes given by the LMOHA network and declares it as the recalled pattern.

4.5 LMOHA Complexity Analysis

The main aim of the proposed scheme here is to reduce the computational complexity of the MOHA’s S&I for pattern recognition, which will lead to a reduction of the overall computational complexity of the MOHA scheme. Therefore, a series of analyses and evaluations of the LMOHA implementation were conducted in order to study the complexity of the LMOHA algorithm in terms of computations and number of communications. These are described in this section which also includes comparisons that have been made with a MOHA scheme in order to demonstrate that the LMOHA has significantly low computational complexity and number of communications associated with its operations. The evaluations of the MOHA’s complexity, presented in section 3.7, show that the MOHA scheme has a low computational complexity compared with the Hopfield network. The evaluations also show that the MOHA scheme requires fewer communications for pattern recognition compared with the DHGN. For such evaluations, we assume that all network nodes in LMOHA scheme are activated as plus, T, corner, or edge nodes by a given pattern to estimate the maximum computations or communications required for learning or recalling an incoming pattern. We also assume that all network nodes in the MOHA scheme are activated as edge nodes by a given pattern. The following sub-sections provide the aforementioned analysis.

4.5.1 Computational complexity evaluation

To analyse the computational complexity of the LMOHA algorithm, Big-O analysis has been considered as the indicator of computational complexity. In this sub-section, comparisons have mainly been made with the MOHA algorithm to examine whether the LMOHA algorithm has less computational complexity associated with its operations.

In order to analyse and compare the LMOHA and MOHA algorithms, the notations for network generation (i.e. deployment) and recognition stages within the implementation have been derived.

4.5.1.1 Network generation stage

This stage involves the formation of a network that comprises computing elements known as neurons (i.e. nodes). It is worth noting that this stage needs to be done only once, which makes it less important than the recognition stage in terms of determining computational complexity. The number of nodes (also known as a network size) generated is totally dependent on the algorithm being implemented.

Like the MOHA algorithm, the number of nodes generated (i.e. network size) by the LMOHA scheme, G_{LMOHA} depends on the number of nodes required for each row (N_{node}) and the total number of rows (N_{Row}), G_{LMOHA} is derived using the following equation:

$$G_{LMOHA} = N_{node} \times N_{Row} \tag{4.6}$$

According to Equation 3.15, the number of nodes (N_{node}) required for a row is equal to the size of the sub-pattern S_{sb} and according to Equation 3.20, the total number of rows (N_{Row}) is equal to the size of a pattern (S_p) divided by the size of the sub-pattern S_{sb}. Therefore, G_{LMOHA} will be equal to:

$$G_{LMOHA} = N_{node} \times N_{Row} = S_{sb} \times \frac{S_p}{S_{sb}} = S_p \tag{4.7}$$

On the other hand, as shown in Equation 3.37, the number of nodes generated by the MOHA scheme is equal to the pattern size (S_p), $G_{MOHA} = S_p$. As a result, the LMOHA scheme does not provide any changes to the original MOHA scheme with regards to its technique for network generation or deployment.

The details of the Big-O notation derived for the LMOHA and MOHA network implementations are shown in Table 4.1. The estimated time derived is based on the assumption that the instruction execution speed of the underlying processor is 1 microsecond (μs) per instruction.

Table 4.1: Big-O notations for LMOHA and MOHA implementation in network generation stage.

Algorithm	**Big-O**	**Efficiency**	**Iteration (*n*)**	**Estimated Time (in seconds)**
LMOHA	$O(n)$	Linear	G_{LMOHA}	$G_{MOHA} \times 0.000001$
MOHA	$O(n)$	Linear	G_{MOHA}	$G_{MOHA} \times 0.000001$

The results show that both LMOHA and MOHA algorithms have identical computational complexity for the network generation stage. It is clearly shown in sub-section 3.7.1 that the MOHA scheme, in regards to the network generation stage, provides lower computational complexity than the Hopfield network. Thus, the LMOHA, which is another version of the MOHA algorithm, also offers a significantly low computational complexity at this stage.

4.5.1.2 Recognition stage

The pattern recognition stage is the core phase for any pattern recognition algorithm. As shown in sub-section 3.7.1, the MOHA recognition stage involves five sub-processes to determine: activated nodes, activated edge nodes, the longest distance between the locations of activated edge nodes, the critical point, and the MOHA value. Only the first two sub-processes are performed by the MOHA network and the remaining sub-processes are handled by the S&I. Table 3.6 shows the Big-O notations derived from the analysis of the MOHA recognition process. This is based on the assumption that all network nodes are activated as edge nodes by a given pattern in order to estimate the maximum computations required for pattern recognition. Figure 3.21 shows the computational complexity for the MOHA scheme in the recognition stage, which is also based on the assumption that all network nodes are activated as edge nodes.

Similar to the MOHA scheme, the LMOHA scheme involves a single-cycle process in which each input pattern will be passed through the LMOHA network once and the storing or recall process will be activated according to the instruction given. This process of recognition involves only two sub-processes, as presented in section 4.3: determination of activated nodes and determination of activated plus, T, corner, and edge nodes. Both of these sub-processes are performed by the LMOHA network. Therefore, the computational complexity of the MOHA's S&I is completely reduced and removed.

Table 4.2 shows the Big-O notations derived from the analysis of the LMOHA recognition process. This is based on the assumption that all network nodes are activated as plus, T, corner, or edge nodes by a given pattern in order to estimate the maximum computations required for pattern recognition. Similar to the previous analysis, the estimated time derived is based on the assumption that the instruction speed used is 1 microsecond (μs) per instruction.

Table 4.2: Computational complexity of the LMOHA implementation's recognition stage.

Process	**Big-O**	**Efficiency**	**Iteration (n)**	**Estimated Time (in seconds)**
Activated Nodes Determination	$O(n)$	Linear	$G_{LMOHA} = S_p$	$S_p \times 0.000001$
Activated Plus, T, Corner, and Edge Nodes Determination	$O(n)$	Linear	$G_{LMOHA} = S_p$	$S_p \times 0.000001$

Figure 4.7 shows the computational complexity for these two processes in the recognition stage utilising the LMOHA implementation. Note that the LMOHA computational complexity is based on the assumption that 100% of

the network nodes are activated as plus, T, corner, or edge nodes, which is reflecting the worst case for the algorithm that occurs very rarely.

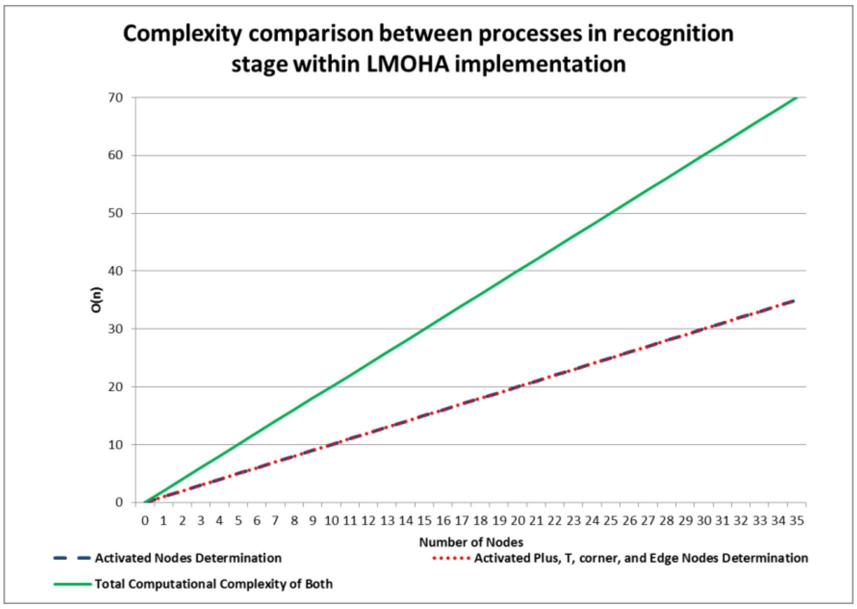

Figure 4.7: Big-O notation comparisons for processes within LMOHA recognition stage.

Figures 3.21 and 4.7 clearly show that the LMOHA incurs less computational complexity during the pattern recognition process compared with the MOHA scheme. For instance, the total computational complexity (i.e. the number of instructions) for the LMOHA algorithm utilising 35 nodes is 70 in the recognition process. However, it is 668 in the MOHA implementation for the recognition stage with the same number of nodes. This is mainly a result of reducing the computational complexity of the S&I. Furthermore, based on the numerical results, we can conclude that the time taken for the LMOHA recognition process is far less than for the MOHA.

The analysis clearly demonstrates that the LMOHA incurs less computational complexity for its pattern recognition processes compared with the MOHA implementation, and is therefore an appropriate solution for resource-constrained networks.

4.5.2 Communication complexity analysis

In this sub-section, we compare the communication complexities of the MOHA and the LMOHA schemes. In conducting an analysis of the communication costs, all the recognition steps in the pattern recognition scheme that require communications have been considered. Table 4.3 presents the definitions of the various terms used when estimating the communication complexity of the LMOHA and MOHA algorithms.

The two LMOHA operation types (memorise or recall) must be considered when estimating the number of communications in a LMOHA network. Both memorisation and recall operations involve an exchange of communications whereby each activated node sends its information to its adjacent nodes. For such estimation, we assume that all network nodes are activated as plus, T, corner, or edge nodes by a given pattern in order to estimate the maximum number of communications required to learn or recall an incoming pattern. Similar to the MOHA scheme, normally each activated node in the LMOHA network has four adjacent nodes with which it exchanges information. However, the actual number of adjacent nodes and exchange communications of an active node varies from two to four based on its location or position in the row and whether the row is located at the base, in the middle, or at top of the network, as follows:

Table 4.3: Symbols and terms for complexity estimation.

Symbol	Terms Name	Terms meaning
S_p	**Pattern Size**	The size of the pattern
S_{sb}	**Sub-pattern Size**	The size of the sub-pattern
C_{r_0}	**Bottom Layer Communications**	The total number of exchanged communications in the bottom row
$C_{r_i}^{total}$	**Middle layers Communications**	The total number of exchanged communications in all middle rows
$C_{r_{top}}$	**Top layer Communications**	The total number of exchanged communications in the top row
C_{report}	**Report Communications**	The total number of report communications to the S&I
N_{ED}^{MOHA}	**MOHA's Edge Nodes**	The number of activated edge nodes in MOHA network
N_{PL}^{LMOHA}	**LMOHA's Plus Nodes**	The number of activated plus nodes in LMOHA network
N_{T}^{LMOHA}	**LMOHA's T Nodes**	The number of activated T nodes in LMOHA network
N_{CO}^{LMOHA}	**LMOHA's Corner Nodes**	The number of activated corner nodes in LMOHA network
N_{ED}^{LMOHA}	**LMOHA's Edge Nodes**	The number of activated edge nodes in LMOHA network
C_{total}^{MOHA}	**MOHA Communications**	The total number of MOHA communications
C_{total}^{LMOHA}	**LMOHA Communications**	The total number of LMOHA communications

Bottom row: According to Equation 3.16, the cumulative communication costs involved for each input recognition process for all nodes in the bottom row is derived through the following equation:

$$C_{r_0} = 3(S_{sb} - 2) + 4 \tag{4.8}$$

Middle rows: According to Equation 3.18, the cumulative communication cost for all nodes in the all middle rows is derived through the following equation:

$$C_{r_i}^{total} = \sum_{i=1}^{top-1}(4(S_{sb} - 2) + 6) \tag{4.9}$$

Top row: According to Equation 3.19, the cumulative communication costs involved for each input recognition process for all nodes in the top row is derived from the following equation:

$$C_{r_{top}} = 3(S_{sb} - 2) + 4 \tag{4.10}$$

Up to this stage, the local communication cost of the LMOHA scheme is identical to that of the MOHA scheme. However, the total communication costs are different because the MOHA scheme utilises only the edge nodes for report communications to the S&I whereas the LMOHA scheme uses plus, T, corner, and edge nodes for report communications to the S&I, as shown in Equation 4.1. According to Equation 3.59, the total communication costs for the MOHA network in memorisation or recalling operations can be derived as follows:

$$C_{total}^{MOHA} = C_{r_0} + C_{r_i}^{total} + C_{r_{top}} + N_{ED}^{MOHA} \tag{4.11}$$

On the other hand, the total communication costs for LMOHA network in memorisation or recalling operations can be derived with:

$$C_{total}^{LMOHA} = C_{r_0} + C_{r_i}^{total} + C_{r_{top}} + N_{PL}^{LMOHA} + N_{T}^{LMOHA} + N_{CO}^{LMOHA} + N_{ED}^{LMOHA} \tag{4.12}$$

Note that in Equations 3.29 and 4.3, the number of activated edge nodes in the MOHA implementation is equal to the number of activated edge nodes in the LMOHA implementation plus the number of activated corner nodes, as follows:

$$N_{ED}^{MOHA} = N_{CO}^{LMOHA} + N_{ED}^{LMOHA} \quad (4.13)$$

Table 4.4 and Figure 4.8 show the comparison of communication costs for DHGN, MOHA and LMOHA. Overall communication required for LMOHA is based on the assumption that 100% of the network nodes are activated as plus, T, corner, or edge nodes, which is the worst case scenario for the algorithm; the overall communication required by the MOHA is based on the assumption that 50% and 100% of the network nodes are activated as edge nodes, respectively.

Table 4.4: Comparison between DHGN, MOHA, and LMOHA implementations with regards to the number of messages communicated per pattern of given size.

S_p	C_{total}^{DHGN}	C_{total}^{MOHA} 100% N_{ED}^{MOHA}	C_{total}^{MOHA} 50% N_{ED}^{MOHA}	C_{total}^{LMOHA}
5	28	18	16	18
15	84	59	52	59
25	140	105	93	105
35	196	151	134	151
45	252	197	175	197
55	308	243	216	243

The results given in Table 4.4 and Figure 4.8 show that, on average, LMOHA and MOHA implementations in comparison to DHGN have achieved message-passing savings of 26.1%. As discussed in section 2.5, the number of nodes required in DHGN increases exponentially with the increase of the problem (pattern) size, which will lead to an increase in the number of messages communicated. The number of messages communicated in both the LMOHA and MOHA implementations are identical when it comes to the worst-case scenarios of both schemes (when 100% of the network nodes are activated as edge nodes in MOHA and when 100% of the network nodes are activated as plus, T, corner, or edge nodes in LMOHA). However, it has previously been described in Equation 4.13 that $N_{ED}^{MOHA} = N_{CO}^{LMOHA} + N_{ED}^{LMOHA}$ and based on this fact we can state that: $(N_{PL}^{LMOHA} + N_{T}^{LMOHA} + N_{CO}^{LMOHA} + N_{ED}^{LMOHA}) \geq N_{ED}^{MOHA}$. As a result, in most scenarios, the cumulative communication costs for the MOHA network will be less than the total communication costs for the LMOHA network, $(C_{total}^{MOHA} < C_{total}^{LMOHA})$. Table 4.4 and Figure 4.8 also illustrate that when only 50% of the network nodes are activated as edge nodes in the MOHA scheme, on average, there will be a mere 11% saving in communication cost compared with that of the LMOHA scheme (worst scenario).

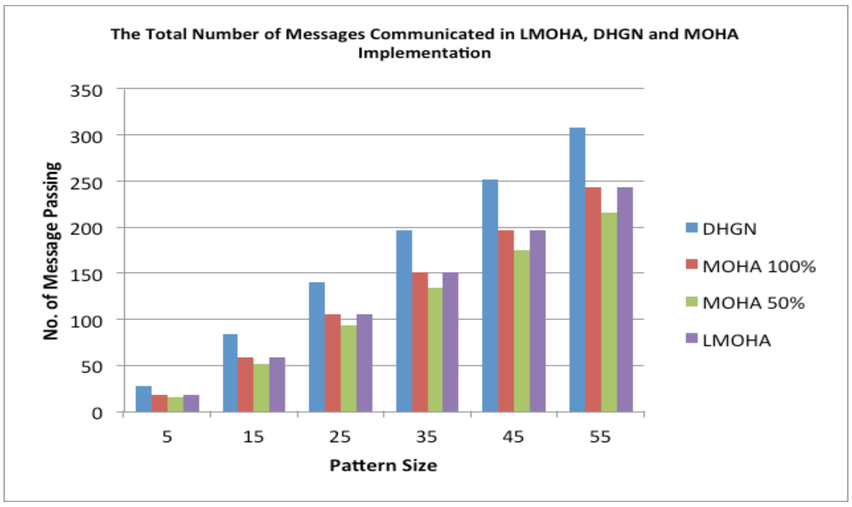

Figure 4.8: A comparison of communication costs for LMOHA, DHGN and MOHA.

We have therefore shown that the LMOHA scheme operates with less complexity when compared with the MOHA algorithm in terms of processing power consumption. Moreover, this section shows that the LMOHA scheme offers pattern recognition without requiring an excessive number of communication messages, when compared with the MOHA scheme, which makes it suitable for ready deployment on wireless sensor networks.

In order to evaluate the capabilities of the LMOHA scheme as a pattern recognition algorithm, a simulation study is presented in the next section, for subsequent comparison with the accuracy of the MOHA scheme.

4.6 Simulations and Results Analysis

In this section, a series of tests have been conducted on the LMOHA and MOHA schemes. The main aims of these tests are to compare the recognition accuracy of LMOHA and MOHA schemes and to estimate the limits of tolerance of both schemes to transformed and noisy patterns. Two types of pattern datasets were selected for these tests. One pattern dataset consists of uniform shape patterns and another pattern dataset consists of non-uniform patterns. The main reason for selecting different types of pattern datasets is to examine whether the proposed schemes accuracies are affected by the types of patterns with which they are dealing. In these tests, inter-node communications and S&I communications were implemented utilising the MPICH-2 library for the message passing interface (MPI) [262]. The simulator itself was written in the C/C++ Integrated Development Environment.

4.6.1 LMOHA and MOHA accuracy test using uniform patterns (*shapes* dataset)

The aims of this test were to compare the recognition accuracy of LMOHA and MOHA schemes and to estimate the limits of tolerance of the LMOHA and MOHA schemes to transformed and noisy patterns for uniform patterns. To perform this test, we constructed a database called *MPEG7 CE Shape-1 Part B* (*shapes* dataset) that consisted of 1400 binary shape images for training and testing [261].

The training dataset was constructed by creating and utilising five shapes modelled as a binary image of size 100-by-100 pixels, as shown in Figure 4.9 (a). These images were presented to the LMOHA and MOHA networks for memorisation (storing). To construct the testing dataset, a variety of shapes associated with these images and altered versions of these images were produced for recognition operations. In the first set, a variety of shapes for each image were tested (20 different shapes for each image). In the second test, each image was rotated counter-clockwise from 1 to 360, with five degrees for each rotation level. In the third set, each image was randomly translated 100 times by shifting the pattern's location. In the fourth set, each image was spatially dilated by scaling the object size utilising 50 scaling levels. That is, the images were scaled from 1% to 100% scaling percentages in 2% steps (increments). In the fifth set, the images were distorted by applying different levels of noise to them (from 5% to 100% in 5% steps). Figure 4.9 (b to f) shows a sample of different shapes of images and altered images utilising displaced positions (translation), scaling (dilation), rotation, and noise effects. In this test, the total number of test images is 282: 40 shapes for an image, 72 rotated images, 100 translated images, 50 scaled images, and 20 noisy (distorted) images for each original trained binary image. These images were utilised for testing to determine the recognition accuracy and the boundaries of the LMOHA's and MOHA's distortion and invariant recognition capabilities.

In the recognition test, we ran simulated LMOHA and MOHA networks of 10000 nodes assuming that nodes are distributed as a grid and each node detects one pixel reading. We also assume that each LMOHA and MOHA network is divided into 100 rows and each one of these rows is utilised to handle 100-bit binary sub-patterns and the threshold values for node activation and edge determination are set to 1 ($\varphi = 1$). We first trained the networks utilising the constructed training dataset. Then, we presented testing datasets for recall.

Figure 4.10 shows the accuracy of the LMOHA and MOHA schemes in recalling different type of images. The accuracy of the networks in Figure 4.10 is calculated as the total number of correctly classified patterns as a percentage of the number of tested images. The higher the percentage, the higher is the accuracy. The results shown in Figure 4.10 indicate that, on average, the overall recognition accuracy of the LMOHA and MOHA schemes is almost identical for the presented testing datasets. However, in order to estimate the limits of tolerance of the LMOHA and MOHA schemes to each type of presented testing dataset, we examined the performance of these schemes when dealing with these datasets individually.

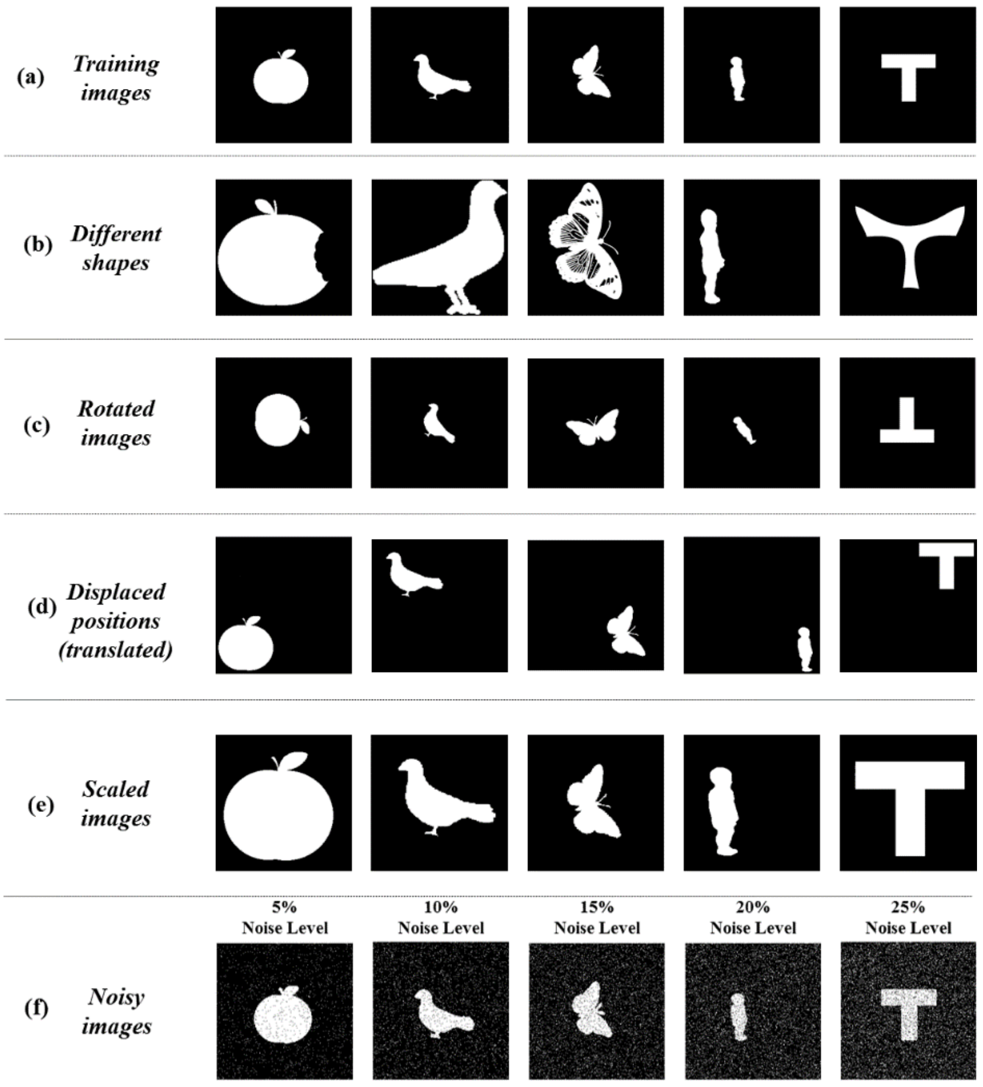

Figure 4.9: The training dataset and different samples of the testing dataset: (a) the training images, (b) a sample of different shapes of images, (c) a sample of rotated images, (d) a sample of translated images, (e) a sample of scaled images, and (f) a sample of noisy images.

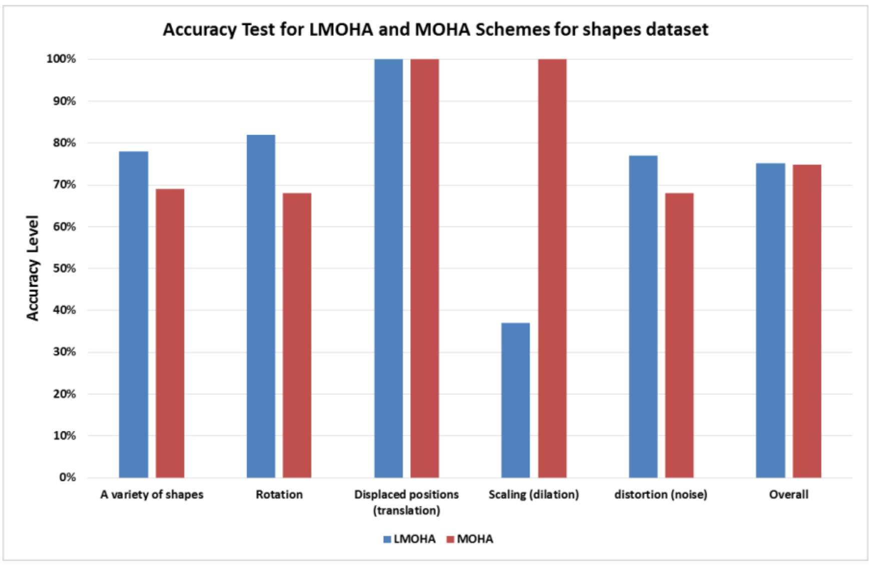

Figure 4.10: The results of the accuracy test for both the LMOHA and MOHA schemes for *shapes* dataset.

The first set of recall images is a variety of shapes for each trained image. The main aim of this test is to examine the classification capabilities of the proposed schemes. Figure 4.10 shows the accuracy of the LMOHA and MOHA schemes in recalling these types of images. It is clearly shown in Figure 4.10 that the LMOHA scheme has a higher (78%) level of accuracy when dealing with a variety of shapes for each trained image compared with the MOHA scheme, which achieved an accuracy level of a mere 69%. The main reason for this poorer result is that the number of activated nodes participating in the recalling process for the LMOHA implementation is usually more than the number of activated nodes of the MOHA recalling procedure, as shown in previous sub-section. The LMOHA scheme uses the plus, T, corner, and edge nodes in the recognition process; however, the MOHA scheme utilises only edge nodes for recognition.

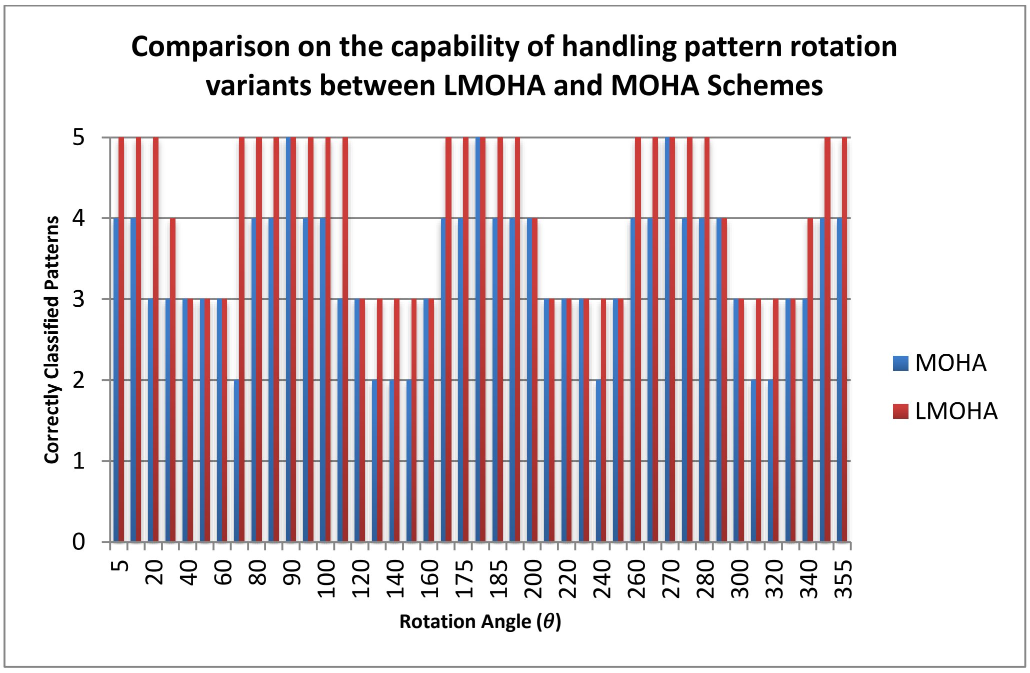

Figure 4.11: Comparison on the capability of handling pattern rotation variants between the LMOHA and MOHA schemes.

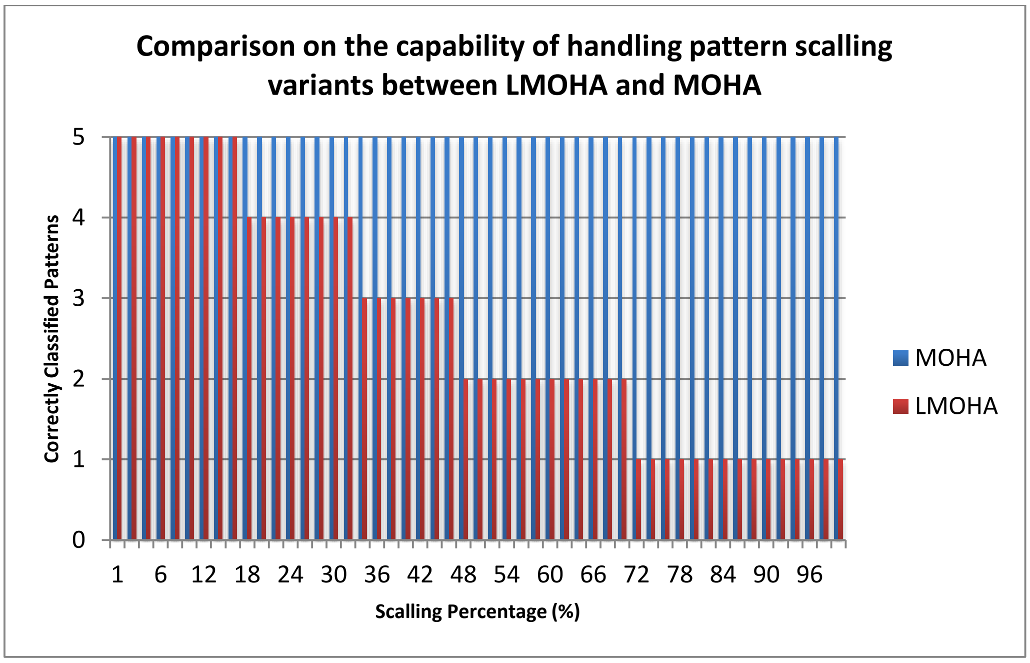

Figure 4.12: Comparison on the capability of handling pattern scaling variants between LMOHA and MOHA schemes.

The second set of the training images is the rotated samples. Figure 4.11 shows each rotation angle and the number of correctly recognised patterns at that angle. It is important to note that all five samples were presented per rotational angle. Figures 4.10 and 4.11 clearly show that the LMOHA scheme achieves more accurate recognition of rotated images than does the MOHA scheme. Similar to the previous dataset tested, the main reason for the higher accuracies is that the number of activated nodes participating in the recalling process for the LMOHA implementation is usually more than the number of activated nodes of the MOHA recalling procedure. Moreover, Figure 4.11 shows that the MOHA scheme is highly accurate (five correctly classified samples) in five rotational regions. The first region is between 0 and 4 degrees, the second region is between 86 and 94 degrees, the third region is between 176 and 184 degrees, the region where patterns are horizontally flipped or nearly flipped, the fourth region is between 266 and 274 degrees, and the fifth region is between 356 and 360 degrees. On the other hand, Figure 4.11 also shows that the LMOHA scheme is highly accurate in five rotational regions. The first area is between 0 and 29 degrees, the second area is between 61 and 119 degrees, the third area is between 161 and 199 degrees, the forth area is between 251 and 289 degrees, and the area region is between 341 and 360 degrees. In other words, the MOHA and LMOHA schemes are capable of efficiently detecting patterns rotated within these ranges, and could possibly detect higher rotational degrees.

The third set of recall images is the displaced (translated) samples. Figure 4.10 shows that both the LMOHA and MOHA schemes successfully recognise all translated patterns correctly, which confirms their capabilities of dealing with translation. The fourth set of the training images were the set of scaled (dilated) images. Figure 4.12 shows the recall accuracy of this set (250 training samples were presented for this test). The graph shows the number of correctly recalled patterns for each level of spatial scaling (dilation). Five samples were presented at each scaling level. Figures 4.10 and 4.12 clearly show that the MOHA scheme achieves more accurate recognition of scaled images than does the LMOHA scheme. Figure 4.12 also shows that the LMOHA scheme is capable of offering perfect recognition accuracy (5 correctly classified patterns) for scaling levels up to 16%. The LMOHA is also capable of correctly classifying four patterns for scaling levels up to 32%. Note that increasing the level of scaling results in a decrease in recognition accuracy. This is due to the increase in the number of activated plus, T, corner, and edge nodes in the same area, which leads to false recall. On the other hand, the results presented in Figure 4.12 clearly indicate that the MOHA algorithm can provide perfect recognition accuracy for all tested scaled patterns. The way in which the MOHA value (MV) is calculated in the MOHA scheme, as the relative distance between all activated edge nodes (to the critical point) to the distance of the farthest two activated edge nodes from each other, is the main reason for the MOHA's ability to deal with scaled patterns.

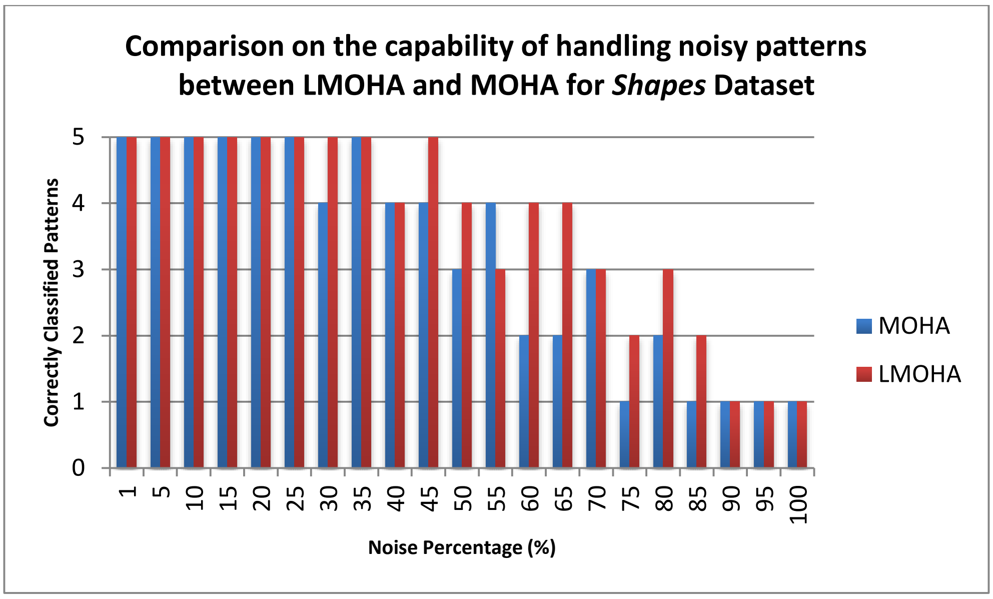

Figure 4.13: Comparison on the capability of handling distorted patterns between LMOHA and MOHA schemes.

The fifth set of recall images is the distorted (noisy) images. Figure 4.13 displays the recall accuracy of this set (100 training samples were presented for this test). The figure shows the number of correctly recognised patterns for each level of distortion (noise). Five samples were presented at each distortion level. Figures 4.10 and 4.13 show that the LMOHA scheme achieves more accurate recognition of noisy images than does the MOHA scheme. Similar to the first and second datasets tested, the main reason for the higher accuracies is that the number of activated nodes participating in the recalling process for the LMOHA implementation is usually more than the number of activated nodes of the MOHA recalling procedure. Moreover, Figure 4.13 shows that the LMOHA scheme is capable of offering perfect recognition accuracy (five correctly classified patterns) for distortion levels up to 35%. The LMOHA is also capable of correctly classifying four patterns for distortion levels up to 50%. On the other hand, Figure 4.13 shows that the MOHA scheme is capable of offering perfect recognition accuracy for distortion levels up to 25%. The MOHA is also capable of correctly classifying four patterns for distortion levels up to 45%. Note that increasing the level of distortion results in a decrease in recognition accuracy. This is due to the increase in the number of activated nodes in the pattern, which leads to false recall.

4.6.2 LMOHA and MOHA Accuracy Test using non-uniform patterns (star dataset)

This test aimed to compare the recognition accuracy of LMOHA and MOHA schemes and estimate the limits of tolerance of the LMOHA and MOHA schemes to transformed and noisy patterns for non-uniform patterns. In all previous experiments of MOHA and LMOHA schemes, *MPEG7 CE Shape-1 Part B dataset* (*shapes* dataset) has been utilised which consists uniform shape patterns. As a result, we perform this test to check the recognition accuracy of proposed schemes for these types of patterns. For this purpose, we constructed a database called *stars dataset* that has training and testing datasets. We generated a training dataset of star maps. We chose five star map images from [270]. The size of each star map is 150×150 pixels. These maps were transformed into binary star map images in order to mark the brightest pixels (stars) on these maps. Figure 4.14 (a) shows an example of one star map that has been transformed into a binary star map. Figure 4.14 (b) presents the remaining resultant binary star maps performing the same operation. To construct the testing dataset, we generated 72 rotated star maps (rotating from 1 to 360 degrees with five degrees for each rotation level), 100 randomly translated star maps, 50 scaled star maps (from 1% to 100% in 2% steps), and 20 distorted (noisy) star maps (from 5% to 100% in 5% steps) for each training star map as four testing datasets. A total of 1210 test star maps were generated for recall. In this test, we ran simulated LMOHA and MOHA networks of 22500 nodes assuming that nodes are distributed as a grid and each node detects one pixel reading. We also assume that each LMOHA and MOHA network is divided into 150 rows and each one of these rows is utilised to handle 150-bit binary sub-patterns and the threshold values for node activation and edge determination are set to 1 ($\varphi = 1$). The accuracy of the MOHA and LMOHA are calculated as the total number of correctly recalled patterns as a proportion of the number of altered star maps tested.

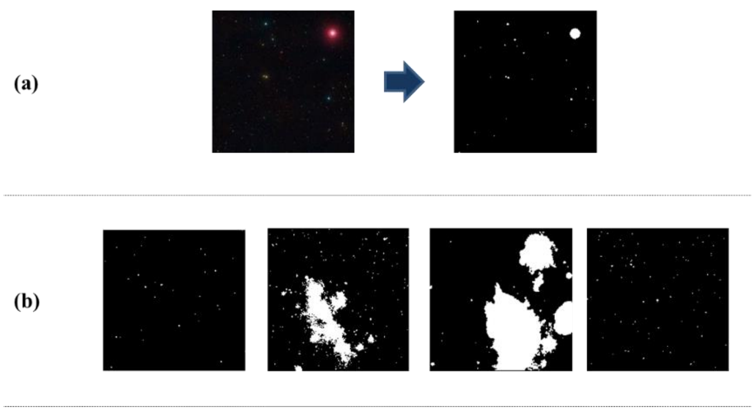

Figure 4.14: The training dataset: (a) Binary transformation of the star map, (b) the rest of the binary training patterns.

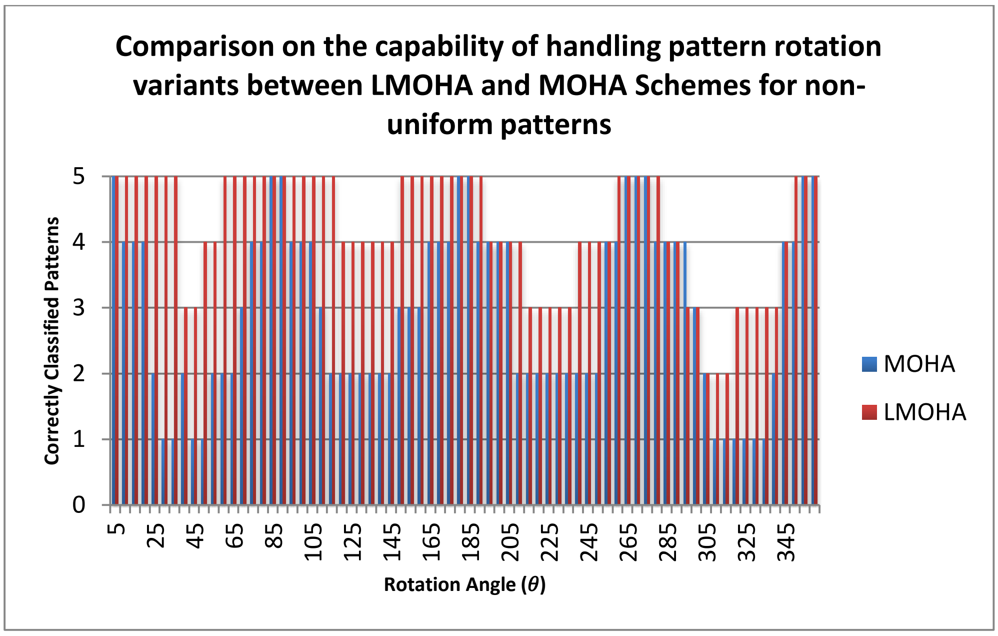

Figure 4.15: Comparison on the capability of handling rotated patterns between LMOHA and MOHA schemes for non-uniform patterns.

Figure 4.15 shows each rotation angle and the number of correctly classified patterns at that angle. It is important to note that a total number of five samples were presented per rotational angle. Similar to the rotation test conducted on the dataset of shapes, the results presented in Figure 4.15 show that the LMOHA scheme offers more accurate recognition for rotated samples than the MOHA scheme. The main reason for this is that the number of activated nodes participating in the recalling process in the LMOHA implementation is usually more than the number of activated nodes utilised by the MOHA recalling procedure. Moreover, results presented in Figure 4.15 show that the MOHA scheme is highly accurate (five correctly classified samples) in five rotational regions. The first region is between 0 and 9 degrees, the second region is between 81 and 94 degrees, the third region is between

176 and 189 degrees, the region where patterns are horizontally flipped or nearly flipped, the forth region is between 261 and 279 degrees, and the fifth region is between 351 and 360 degrees. On the other hand, results presented in Figure 4.15 also show that the LMOHA scheme is highly accurate in five rotational regions. The first area is between 0 and 39 degrees, the second area is between 56 and 119 degrees, the third area is between 146 and 194 degrees, the forth area is between 256 and 284 degrees, and the fifth area is between 346 and 360 degrees. In other words, the MOHA and LMOHA schemes are capable of efficiently detecting patterns rotated within these ranges, and could possibly detect higher rotational degrees. Additionally, Figures 4.11 and 4.15 show that the recognition accuracy of LMOHA and MOHA schemes for rotated patterns is higher for the *stars dataset* compared with the *shapes dataset*. This indicates that non-uniform patterns are less sensitive to rotation effects than uniform patterns. This also indicates that a lower number of activated nodes change their types in non-uniform patterns.

The second set of recall star maps is the displaced (translated) samples. The results of this test show that both the LMOHA and MOHA schemes successfully recognised all translated patterns correctly, which confirm their capabilities of dealing with translation. The third set of the training star maps was the set of scaled (dilated) samples. Figure 4.16 shows the recall accuracy of this set (250 training samples were presented for this test). The graph shows the number of correctly recalled patterns for each level of spatial scaling (dilation). Five samples were presented at each scaling level. Similar to the rotation test conducted on *shapes* dataset, results presented in Figure 4.16 clearly show that the MOHA scheme provides more accurate recognition for scaled patterns than does the LMOHA scheme. Figure 4.16 shows that the LMOHA scheme is capable of offering perfect recognition accuracy (five correctly classified patterns) for scaling levels up to 16%. The LMOHA is also capable of correctly classifying 4 patterns for scaling levels up to 30%. Note that increasing the level of scaling results in a decrease in recognition accuracy. This is due to the increase in the number of activated plus, T, corner, and edge nodes in the same area, which leads to false recall. On the other hand, results shown in Figure 4.16 clearly indicate that the MOHA scheme is capable of offering perfect recognition accuracy (five correctly classified patterns) for scaling levels up to 60%. The MOHA is also capable of correctly classifying four patterns for scaling levels up to 80%. The way in which the MV is calculated in the MOHA scheme, as the relative distance between all activated edge nodes (to the critical point) to the distance of the farthest two activated edge nodes from each other, is the main reason for having this impressive ability to deal with scaled patterns. On the other hand, Figures 4.12 and 4.16 show that the recognition accuracy of the LMOHA and MOHA schemes for scaled patterns is lower for the *stars dataset* compared with the *shapes* dataset. This indicates that non-uniform patterns are more sensitive to dilation effects than are the uniform patterns. This also indicates that a higher number of activated nodes change their types in non-uniform patterns.

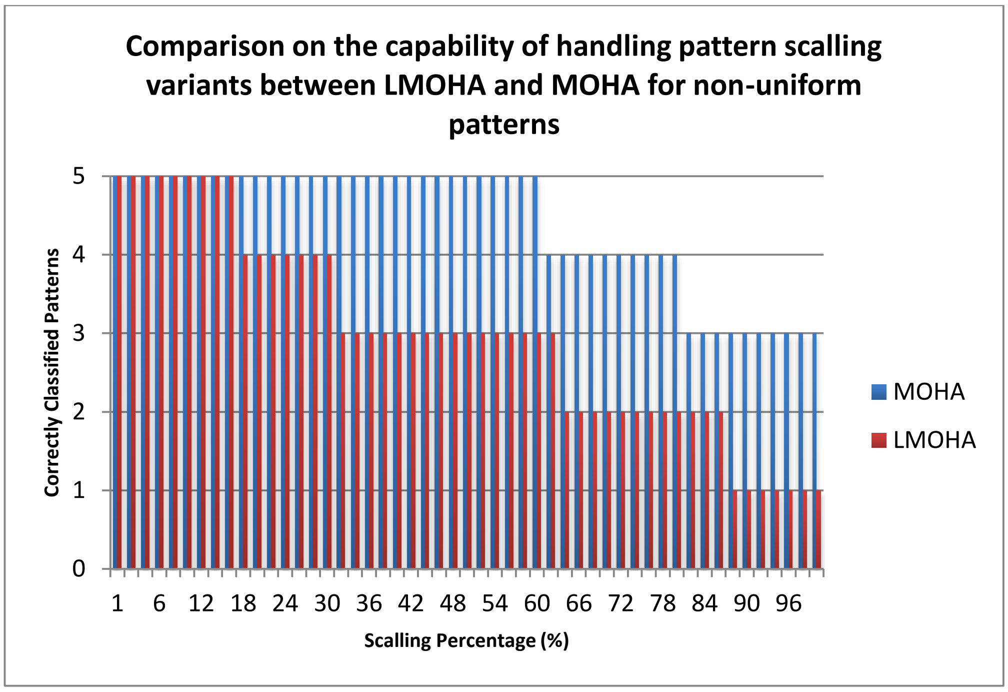

Figure 4.16: Comparison on the capability of handling pattern scaling variants between LMOHA and MOHA schemes for non-uniform patterns.

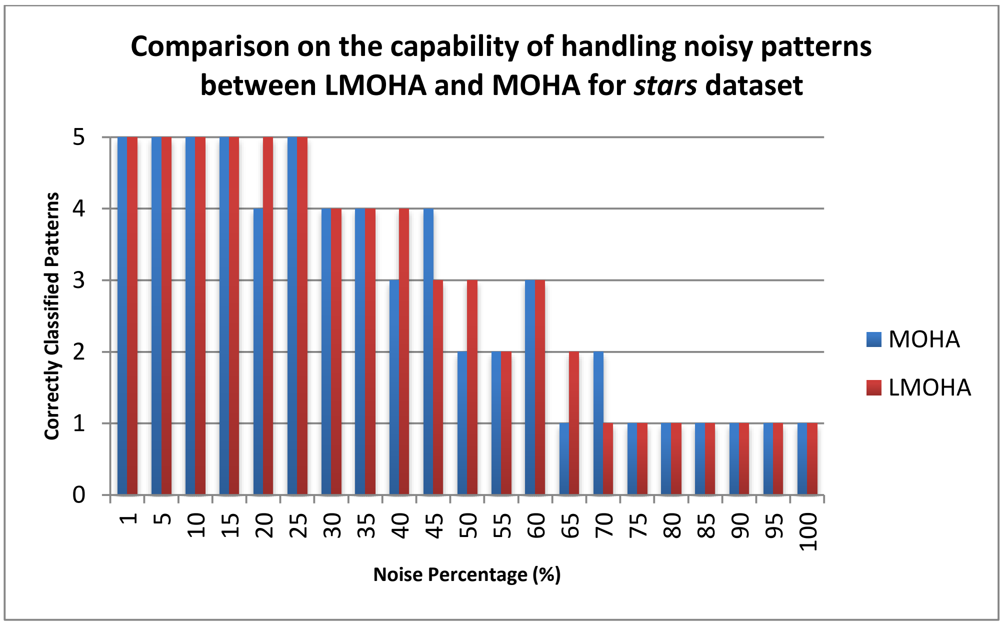

Figure 4.17: Comparison on the capability of handling pattern distortion (noise) between LMOHA and MOHA schemes for non-uniform patterns.

The third set of recall star maps is the distorted (noisy) images. Figure 4.17 displays the recall accuracy of this set (100 training samples were presented for this test). The figure shows the number of correctly recognised patterns for each level of distortion (noise). Five samples were presented at each distortion level. Figure 4.17 shows that the LMOHA scheme provides more accurate recognition for noisy images than does the MOHA scheme. Figure 4.17 also shows that the LMOHA scheme is capable of offering perfect recognition accuracy (5 correctly classified patterns) for distortion levels up to 25%. The LMOHA is also capable of correctly classifying four patterns for distortion levels up to 40%. On the other hand, Figure 4.17 shows that the MOHA scheme is capable of offering perfect recognition accuracy for distortion levels up to 15%. The MOHA is also capable of correctly classifying four patterns for distortion levels up to 35%. Note that increasing the level of distortion results in a decrease in recognition accuracy. This is due to the increase in the number of activated nodes in the pattern, which leads to false recall. Additionally, Figures 4.13 and 4.17 show that the recognition accuracy of LMOHA and MOHA schemes for distorted (noisy) patterns is lower for the *stars dataset* compared with the *shapes* dataset. Similar to the previous dataset tested, this indicates that non-uniform patterns are more sensitive to distortion (noise) effects than uniform patterns. This also indicates that a higher number of activated nodes change their types in non-uniform patterns.

4.7 Conclusions

This chapter has presented the second proposed approach for pattern recognition, known as the Light Macroscopic Object Heuristics Algorithm (LMOHA). LMOHA aims to reduce the computational complexity of the MOHA's S&I for pattern recognition, which will lead to reduction of the overall computational complexity of the MOHA scheme. Instead of using the pattern edge information for recognition (as does the MOHA), the LMOHA implements a mechanism that searches for the sensory-based shapes of patterns. This chapter showed that by describing patterns using their sensory-based shapes, it is possible to achieve an efficient recognition scheme that has a very low computational complexity and can detect transformed and noisy patterns.

With regards to the algorithmic complexity, it has been proven that the LMOHA has less complexity compared with that of the MOHA algorithm. Moreover, the analysis conducted in the chapter also shows that the LMOHA scheme does pattern recognition without requiring excessive message exchanges, when compared with the MOHA scheme.

Accuracy tests were conducted on the proposed schemes. These tests provided results used to compare the recognition accuracy of the LMOHA and MOHA schemes and to estimate the limits of tolerance of the LMOHA and MOHA schemes to transformed and noisy patterns. Two types of pattern datasets were selected for these tests. One pattern dataset consists of uniform shape patterns and another pattern dataset consists of non-uniform patterns. Furthermore, the results of the tests have shown that the LMOHA and MOHA algorithms are capable of yielding high recall accuracy for transformed and noisy patterns. Moreover, the results show that the LMOHA scheme offers better recognition accuracy than does the MOHA scheme for recognition of different shapes of images, rotated images and distorted (noisy) images. On the other hand, the MOHA algorithm provides better recognition accuracy for scaled images. Thus, we can conclude that the LMOHA is the better choice for pattern recognition applications as it provides a higher accuracy level for most types of patterns without incurring excessive communication overhead when compared with the MOHA scheme. However, when dealing with scaled patterns for WSN applications, the MOHA scheme proved to be a better choice.

In order to analyse the influence of MAC protocols on LMOHA and MOHA schemes, different types of existing MAC protocols will be presented in the next chapter (Chapter 5); moreover, in this chapter, MAC message exchange models for the MOHA and LMOHA schemes to function over these protocols will be proposed. These models will help to estimate the time and resource requirements imposed by the MOHA and LMOHA schemes on the underlying sensor network. This analysis will present different sequence models and possible scenarios for each model while analysing the time overhead for each scenario. Moreover, the analysis will show the limits of communication time overhead estimates for each MOHA and LMOHA scheme message sequence model. Moreover, experimental analysis on the proposed four sequence models will be conducted in the next chapter to evaluate the communication overhead of MOHA and LMOHA networks for each model using different types of patterns.

In Chapter 6, a series of analyses, evaluations and simulations for MOHA and LMOHA schemes will be conducted. Alongside, a comparison between the two proposed schemes and other well-known pattern recognition schemes will be provided. Moreover, Chapter 6 will describe a series of tests used to examine the capabilities of both the proposed schemes to be utilised in real-life scenarios.

Chapter 5

Medium Access Control Protocols Influence on MOHA and LMOHA

5.1 Preamble

In the previous chapters, the MOHA and LMOHA schemes were proposed as lightweight and distributed pattern recognition schemes that involve limited numbers of communications and computations. These features suit resource-constrained systems and networks such as WSNs. It has been shown through experiments that the MOHA and LMOHA are significantly capable of dealing with noisy and transformed patterns. However, each one of them presented different limits of tolerance to noisy patterns and different types of transformed patterns. Moreover, both schemes operate with low and stable recognition time, compared with other pattern recognition algorithms.

In this chapter, different types of existing MAC protocols will be presented, and MAC message exchange models for the MOHA and LMOHA schemes to function over these protocols will be proposed. The proposal of these models will help in estimating the time and resource requirements of the

MOHA and LMOHA schemes in terms of the communication overhead imposed on the underlying network of wireless sensors. Additionally, this chapter presents an experimentally evaluation of MOHA and LMOHA schemes' communicational overhead in terms of time and energy requirements. As discussed in Chapter 2, network communications are considered to be the most time and energy consuming tasks in WSNs. Therefore, this chapter presents the experiments used to estimate the MOHA and LMOHA networks' learning time cycle duration and energy consumption based on communicational requirements. The main objective of these experiments is to estimate the lifetime and execution duration of the networks utilising the communicational models (MAC protocols) present in the beginning of this chapter.

Finally, this chapter will be concluded by an overall comparison of MOHA and LMOHA schemes in terms of network structure, pattern matching criteria, handling noisy and transformed patterns, number of nodes participating in recognition, node utilisation, and network's lifetime.

The objectives of this chapter are as follows:

1. To propose different types of message sequence models for the proposed schemes based on different types of MAC protocols.
2. To perform an extensive experiment-based evaluation of the MOHA and LMOHA schemes' communicational overhead in terms of time and energy requirements.

3. To perform an overall comparison of both versions of the MOHA algorithms.

The remainder of this chapter is organised as follows: Section 5.2 provides different types of existing MAC protocols, and proposes MAC message exchange models for the MOHA and LMOHA schemes to function over these protocols. Section 5.3 provides an extensive experiment-based evaluation of the MOHA and LMOHA schemes' communicational overhead in terms of time and energy requirements. This section begins with analysing the effects of the nodes activation technique on communications in the network and then it presents a time and energy analysis, and section 5.4 presents an overall comparison of the proposed schemes in terms of network structure, pattern matching criteria, handling noisy and transformed patterns, number of nodes participating in recognition, node utilisation, and network's lifetime. Finally, section 5.5 provides an overview of the chapter.

5.2 MOHA and LMOHA Message Sequence Models

The message exchange protocol followed by both the MOHA and LMOHA helps achieve message exchange between the active nodes, at the MAC layer. In addition, the proposed models for communication attempt to minimise communicational overheads by setting a timing sequence for network nodes to exchange and report messages for pattern reconstruction i.e., recognition. This section will use the abbreviations listed in Table 5.1 when estimating communicational overheads associated with message exchange at the MAC layer for both the MOHA and LMOHA schemes.

Table 5.1: Symbols and terms for communicational overheads estimation.

Symbol	Terms Name	Terms meaning
$n_i^{r_m}$	**Active Node**	An active node located at row *m* with position number *i*
w_t	**Wait Time**	The time that a node should wait before performing other computations or communications
p_t	**Preamble Time**	The time required to send a full preamble field
Δ_t	**Error Delay Time**	The error delay time that may occur in every message exchanged between two nodes due to physical factors
T_{exch}	**Exchange Time**	The time required by two nodes to perform message exchange
T_{send}	**Send Time**	The time required to send a message from one node to another
T_{report}	**Report Time**	The time required by the network to conduct report communications
T_{comm}^{MOHA}	**MOHA Communication Time**	The total communication time for MOHA scheme
T_{comm}^{LMOHA}	**LMOHA Communication Time**	The total communication time for LMOHA scheme

Sub-section 2.2.3.4 discussed the significance of MAC protocols in WSN communications. In addition, it highlighted the fact that traditional protocols allow a single communicational channel for each node, while new research presents multi-channel protocols to support multi-task operations. In this section, the different types of existing MAC protocols will be presented, and MAC message exchange models for the MOHA and LMOHA schemes which function using these protocols will be proposed. These models will help to estimate the time and resource requirements of the MOHA and LMOHA schemes in terms of the communication overhead imposed on the underlying network of wireless sensors. The section will present different sequence models and possible scenarios for each model, and analyse the delay overhead for each scenario.

One of the earlier works to examine the influence of MAC protocols on a pattern recognition scheme's performance and the energy consumption of its network was done by Baqer in [271]. Baqer examined the influence of only contention-based and schedule-based MAC protocols in his proposed pattern recognition schemes. However, we will examine, in this section, the influence of a variety of MAC protocols in our proposed schemes, which cover most of the existing MAC protocol types.

MAC protocols use the term *frame* to describe a message from a node in the network to another node in the data-link layer. A MAC frame is a sequence of bits that contains the necessary information to deliver a message from one node to another [89, 272]. Figure 5.1 shows a general example of a MAC frame. However, different protocols may have different frame structures and bit lengths for each field. The frame shown in Figure 5.1 contains six fields. The *Preamble* field contains a set of bits that occupy the channel. A receiver will listen to a preamble bit sequence in order to identify the inception of an incoming message. In the MOHA and LMOHA communication models, there are two types of preambles, namely, *sending*, and *not sending*. The *sending* preamble indicates that a node is going to send a full frame to the receiver. The *not sending* preamble indicates that a node will have no frames to send. In the MOHA scheme, the *not sending* preamble will be used in reporting communications in order to allow an active node to inform the S&I that it is not activated as an edge node and that it will not send its location information. On the other hand, in the LMOHA scheme, the *not sending* preamble will be used for reporting so as to allow an active node to inform the S&I that it is not activated as a plus, T, corner, or edge node and that it will not send any type information to the S&I. The *Address* field is used to determine source and destination MAC addresses. The *Control* field is used for control purposes such as message sequences and acknowledgements. The *Checksum* field is used for error detection and correction purposes. The *Flag* field is used to indicate the end of message transmission.

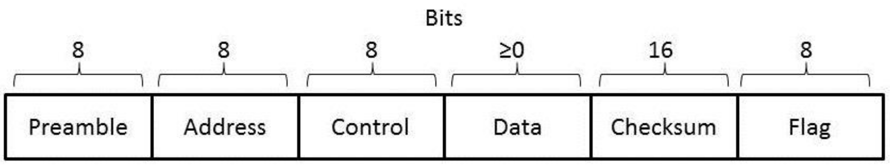

Figure 5.1: A general example of the MAC protocol frame structure.

WSN MAC protocols can be generally divided into [94]: frame-slotted synchronous (FS-Sync), frame-slotted asynchronous (FS-Async), Multi-channel synchronous (MC-Sync), and Multi-channel asynchronous (MC-Async). Table 5.2 summarises these models and the abbreviations for each one that will be utilised throughout the rest of the chapter.

Table 5.2: A Summary of existing MAC protocol types for WSNs [94].

MAC protocol type	Abbreviation	Description
Frame-slotted asynchronous	(FS-Async)	Uses one MAC channel to send or receive a message in an asynchronous mode
Frame-slotted synchronous	(FS-Sync)	Uses one MAC channel to send or receive a message in a synchronous mode
Multi-channel asynchronous	(MC-Async)	Divides the MAC channel into several channels and sends or receives multiple messages at the same time in an asynchronous mode
Multi-channel synchronous	(MC-Sync)	Divides the MAC channel into several channels and sends or receives multiple messages at the same time in a synchronous mode

5.2.1 Frame-slotted asynchronous MOHA and LMOHA models

In this model, each node is allowed to send or receive one frame at any point in time. The sending node starts by sending the preamble. A receiving node senses the channel. In message exchanges for the MOHA/LMOHA schemes, an active node (as described in Definition 3.4) will send a MAC frame that has a sending preamble with its value (v) in the data field to its next node in its row. The next node will be sensing the channel. Once a preamble is received, it starts receiving the rest of the frame. After the sending node finishes sending the frame, it starts listening for the sending preamble from the next node. If no preamble is received, it starts sending to its previous node in its row. Then, the sending node will perform the same steps with the adjacent node in the next higher row and the adjacent node in the lower row. Figure 5.2 and Figure 5.3

show the communication sequence scenarios for an active node ($n_i^{r_m}$) and its neighbouring nodes.

Figure 5.2 shows the normal message sequence when all neighbouring nodes are active. Figure 5.3 shows the scenarios when nodes adjacent to the sending one are inactive. Four different scenarios involving active/inactive nodes were studied. Figure 5.3 (a) shows the sequence when three nodes (the next node in its row: $n_{i+1}^{r_m}$, the previous node in its row: $n_{i-1}^{r_m}$ and the adjacent node in the next higher row: $n_i^{r_{m+1}}$) are active and the adjacent node in the lower row: $n_i^{r_{m-1}}$ is inactive. Figure 5.3 (b) shows the sequence when two nodes (the next node in its row: $n_{i+1}^{r_m}$ and the previous node in its row: $n_{i-1}^{r_m}$) are active and the other nodes (the adjacent node in the next higher row: $n_i^{r_{m+1}}$ and the adjacent node in the lower row: $n_i^{r_{m-1}}$) are inactive. Figure 5.3 (c) shows the sequence when one node (the next node in its row: $n_{i+1}^{r_m}$) is active and the other nodes (the previous node in its row: $n_{i-1}^{r_m}$, the adjacent node in the next higher row: $n_i^{r_{m+1}}$ and the adjacent node in the lower row: $n_i^{r_{m-1}}$) are inactive. Figure 5.3 (d) shows the sequence when all neighbouring nodes to node $n_i^{r_m}$ are inactive, w_t is the time that a node should wait before performing other computations or communications, p_t is the time required to send a full preamble field, and Δ_t is the error delay time that may occur in every communication between two nodes due to physical factors.

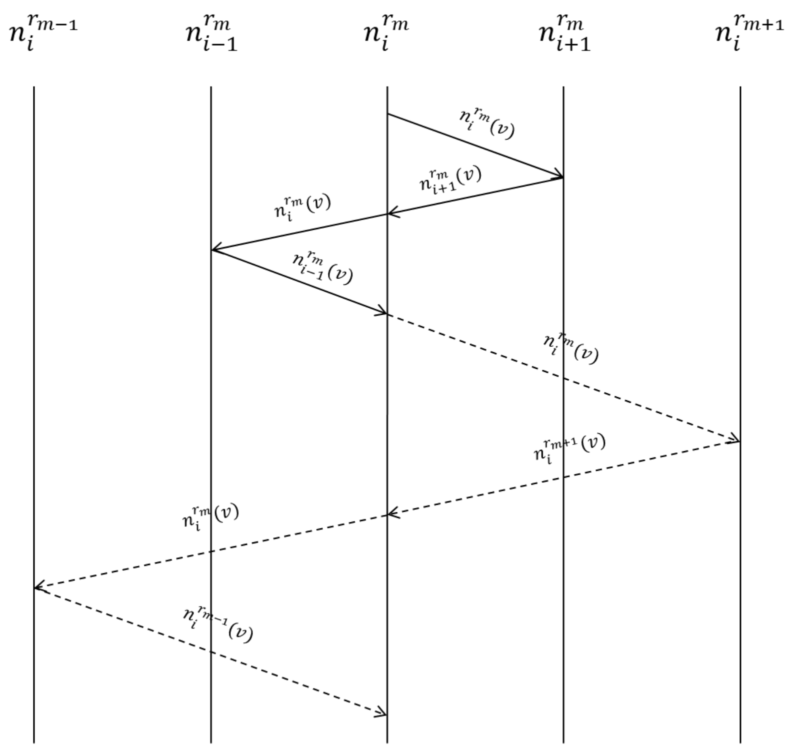

Figure 5.2: MOHA and LMOHA FS-Async message sequence model.

In the normal scenario shown in Figure 5.2, the exchange time for the node ($n_i^{r_m}$) can be estimated as $8T_{send} + 8\Delta_t$, as it will send four frames and receive four messages, where T_{send} is the time required to send one frame, and Δ_t is the error delay time that may occur in every communication between two nodes due to physical factors of the network. In the scenario shown in Figure 5.3 (a), the adjacent node in the lower row ($n_i^{r_{m-1}}$) is inactive. In this scenario, the node $n_i^{r_m}$ will wait for a fixed amount of time (w_t) that is equal to the time required to send a full preamble field (p_t) in addition to the Δ_t. Hence, the time estimation in such a scenario will be $7T_{send} + p_t + 8\Delta_t$. The same will be applicable if any node is de-activated and other nodes are active. The next scenario described in Figure 5.3 (b) is when two neighbouring nodes to $n_i^{r_m}$ are de-activated. In this scenario, the time estimation will be $6T_{send} + 2p_t + 8\Delta_t$. The next scenario described in Figure 5.3 (c) is when three neighbouring nodes to $n_i^{r_m}$ are de-activated. In this scenario, the time estimation will be $5T_{send} + 3p_t + 8\Delta_t$. The last scenario described in Figure 5.3 (d) is when all neighbouring nodes to $n_i^{r_m}$ are de-activated. In such a scenario the time estimation will be $4T_{send} + 4p_t + 8\Delta_t$. It can be concluded that the maximum time required for a node's exchange communications for the two proposed schemes is:

$$T_{exch} = 8T_{send} + 8\Delta_t \tag{5.1}$$

p_t is smaller in length than T_{send} as the p_t is involved in sending only a sub-part of the whole frame. The minimum exchange time for a node can be estimated as follows:

$$T_{exch} = 4T_{send} + 4p_t + 8\Delta_t \tag{5.2}$$

Taking the parallelism of design for the MOHA and LMOHA networks into account, these limits can be used to estimate the entire network's maximum and minimum exchange times as all the network's nodes perform exchange communications simultaneously.

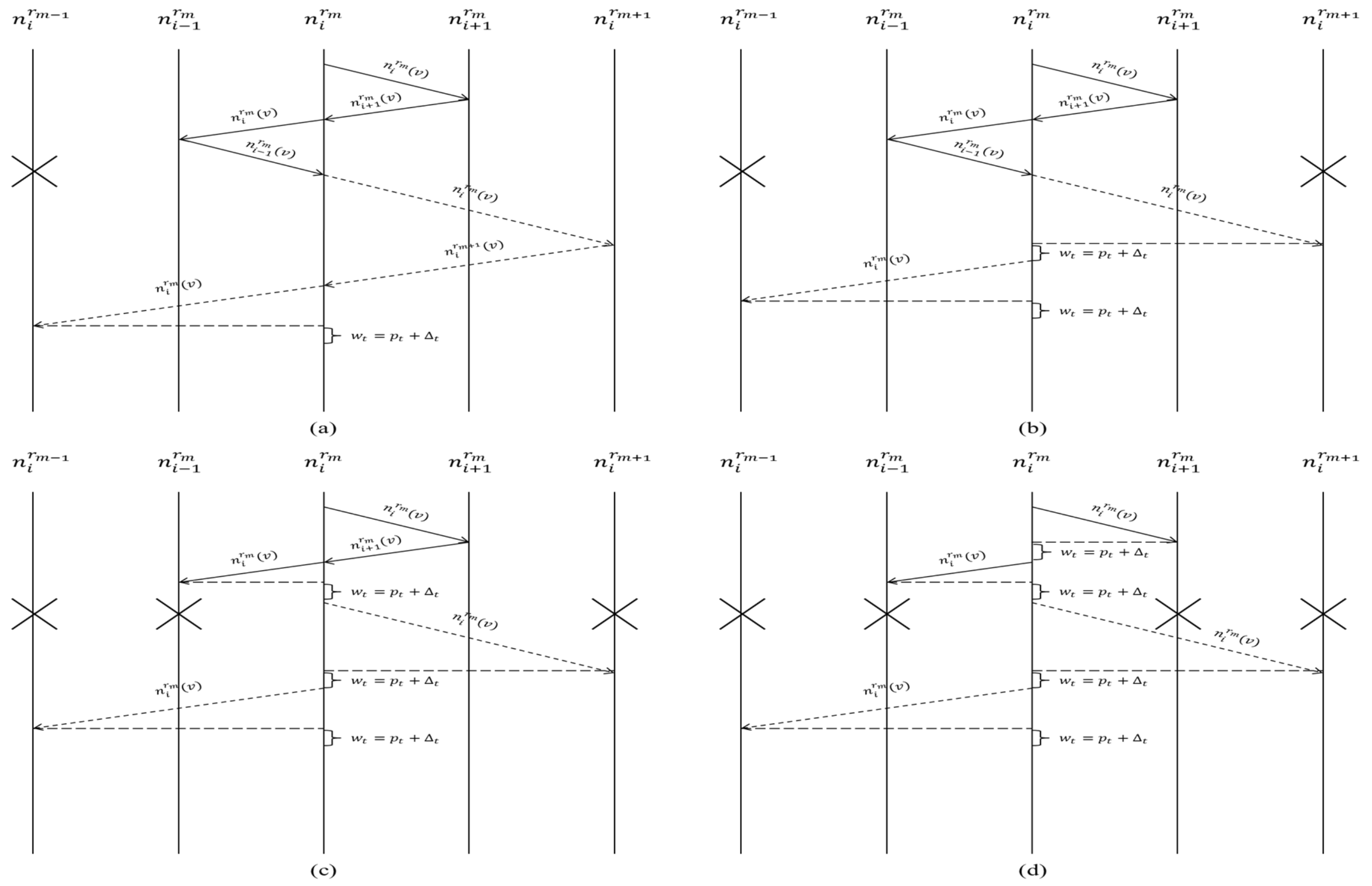

Figure 5.3: MOHA and LMOHA FS-Async message sequence model scenarios: (a) One adjacent node is inactive. (b) Two adjacent nodes are inactive. (c) Three adjacent nodes are inactive. (d) All adjacent nodes are inactive.

In the MOHA implementation, in order to perform the MOHA's report communications, an activated edge node (as described in Definition 3.6) will send a MAC frame that has a *sending* preamble, with its location ($node_{location}$) in the data field to the S&I. Once a preamble is received, it starts receiving the rest of the frame. If the active node is not activated as an edge node, the node sends a *not sending* preamble. This is to minimise the waiting time for the S&I since the preamble sending time is less than the frame sending time. Figure 5.4 shows the two possible scenarios of a communication process for an active node ($n_i^{r_m}$) and the S&I ($S\&I$), where *i* is the node's position in its row and *m* is the node's row number. Figure 5.4 (a) shows the communication sequence if the active node is activated as an edge node and Figure 5.4 (b) shows the sequence when it is inactive as an edge node.

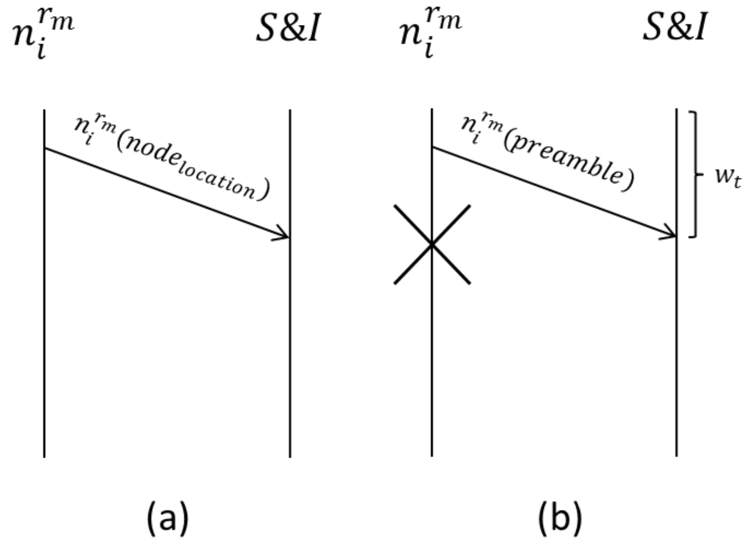

Figure 5.4: MOHA's edge nodes communication phase for FS-Async message sequence model scenarios.

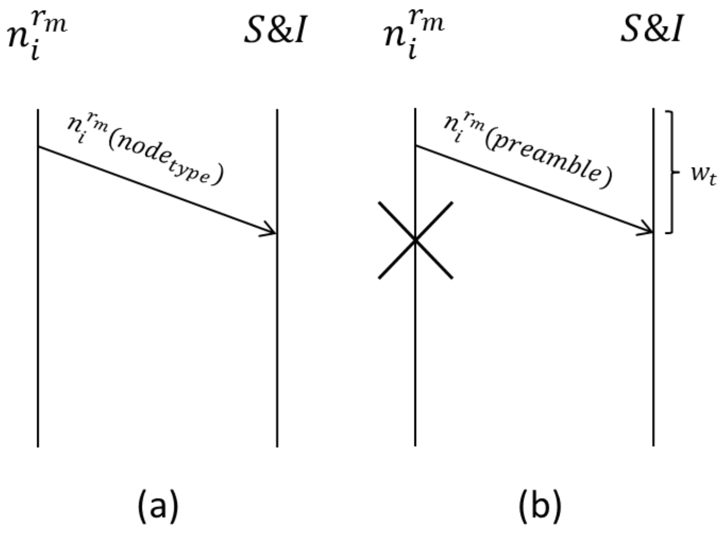

Figure 5.5: LMOHA's plus, T, corner, and edge nodes communication phase for FS-Async message sequence model scenarios.

In the first scenario, the report communication time for the activated edge node ($n_i^{r_m}$) can be estimated as $T_{send} + \Delta_t$, as each active edge node will send one frame including its location information to the S&I ($S\&I$). Since each active edge node will send a frame to the S&I, the maximum report communication time for the network can be estimated as follows:

$$T_{report} = N_{edge}(T_{send} + \Delta_t) \qquad (5.3)$$

N_{edge} represents the number of activated edge nodes in the network. The minimum communication time can be estimated using the second scenario. In this scenario, if the active node $n_i^{r_m}$ is inactive as an edge node, the waiting time for the S&I can be estimated as $w_t = p_t + \Delta_t$. And the minimum report communication time for the entire network can be estimated as follows:

$$T_{report} = N_{active}(p_t + \Delta_t) \qquad (5.4)$$

N_{active} represents the number of active nodes in the network. The total communication time (T_{comm}^{MOHA}) of the network can be calculated as the summation of T_{report} and T_{exch}. Accordingly, the maximum total communication time in such a model for the MOHA scheme can be estimated as follows:

$$T_{comm}^{MOHA} = (N_{edge} + 8)(T_{send} + \Delta_t) \tag{5.5}$$

And the minimum total communication for the MOHA scheme can be estimated as follows:

$$T_{comm}^{MOHA} = (N_{active} + 4)p_t + (N_{active} + 8)\Delta_t + 4T_{send} \tag{5.6}$$

On the other hand, in the LMOHA implementation, in order to perform the LMOHA's report communications, an activated plus, T, corner, and edge node (as described in Definition 4.6) will send a MAC frame that has a *sending* preamble, with its type information ($node_{type}$) in the data field to the S&I. Once a preamble is received, the S&I starts receiving the rest of the frame. If the active node is not activated as a plus, T, corner, or edge node, the node sends a *not sending* preamble. This is to minimise the waiting time for the S&I as preamble sending time is less than the frame sending time. Figure 5.5 shows the two possible scenarios of a communication process for an active node ($n_i^{r_m}$) and the S&I ($S\&I$), where *i* is the node's position in its row and *m* is the node's row number. Figure 5.5 (a) shows the communication sequence if the active node is activated as a plus, T, corner, or edge node and Figure 5.5 (b) shows the sequence when it is inactive as a plus, T, corner, or edge node.

In the first scenario, the report communication time for the activated plus, T, corner, and edge node ($n_i^{r_m}$) can be estimated as $T_{send} + \Delta_t$ as each active plus, T, corner, and edge node will send one frame including its type information to the S&I ($S\&I$). Since every active plus, T, corner, and edge node will send a frame to the S&I, the maximum report communication time for the network in LMOHA scheme can be estimated as follows:

$$T_{report} = (N_{plus} + N_T + N_{corner} + N_{edge})(T_{send} + \Delta_t) \qquad (5.7)$$

N_{plus} represents the number of activated plus nodes in the network, N_T represents the number of activated T nodes in the network, N_{corner} represents the number of activated corner nodes in the network, and N_{edge} represents the number of activated edge nodes in the network. The minimum report communication time can be estimated using the second scenario. In this scenario, if the active node $n_i^{r_m}$ is inactive as a plus, T, corner, or edge node, the waiting time for the S&I can be estimated as $w_t = p_t + \Delta_t$. And the minimum report communication time for the entire network can be estimated as follows:

$$T_{report} = N_{active}(p_t + \Delta_t) \qquad (5.8)$$

N_{active} represents the number of active nodes in the network. The total communication time (T_{comm}^{LMOHA}) of the network can be calculated as the summation of T_{report} and T_{exch}. Accordingly, the maximum total communication time in such a model for the LMOHA scheme can be estimated as follows:

$$T_{comm}^{LMOHA} = (N_{plus} + N_T + N_{corner} + N_{edge} + 8)(T_{send} + \Delta_t) \quad (5.9)$$

And the minimum total communication for the LMOHA scheme can be estimated as follows:

$$T_{comm}^{LMOHA} = (N_{active} + 4)p_t + (N_{active} + 8)\Delta_t + 4T_{send} \quad (5.10)$$

5.2.2 Frame-slotted synchronous MOHA and LMOHA models

A synchronous communication model is intended to guarantee message delivery using acknowledgement (Ack) messages. After sending a frame, a sending node waits a period of time to receive an Ack from the receiving node. If no Ack is received, the sending node retransmits the message assuming that the sent message was lost. In this sub-section, we discuss the synchronous model in terms of sending frames and acknowledgements without dealing with the retransmission process as it requires further discussion and research that may include the physical level of the network. Similar to the FS-Async model described in the previous sub-section, an active node starts to exchange communications by sending its value (v) in a MAC frame to its next node in its row. After receiving the frame, the next node replies with an acknowledgement frame. Then it sends its own value frame if it is active. If not, the next node sends a *not sending* preamble. This is to inform the communicating node that there is no value to be sent, and to allow it to finalise the information exchange process. After an activate node performs a successful communication exchange with its next neighbour, it performs the same steps with its previous node in its row, its adjacent node in the next higher row, and the adjacent node in the lower row. Figure 5.6 shows the sequence when all nodes are active. Figure 5.7 shows four possible scenarios for an active node and its adjacent nodes: (a) the adjacent node in the lower row ($n_i^{r_{m-1}}$) is inactive, (b) the adjacent node in the next higher row ($n_i^{r_{m+1}}$) and the adjacent node in the lower row ($n_i^{r_{m-1}}$) are inactive, (c) the previous node in its row ($n_{i-1}^{r_m}$), the adjacent node in the next higher row ($n_i^{r_{m+1}}$), and the adjacent node in the lower row ($n_i^{r_{m-1}}$) are inactive, and (d) all adjacent nodes to node $n_i^{r_m}$ are inactive.

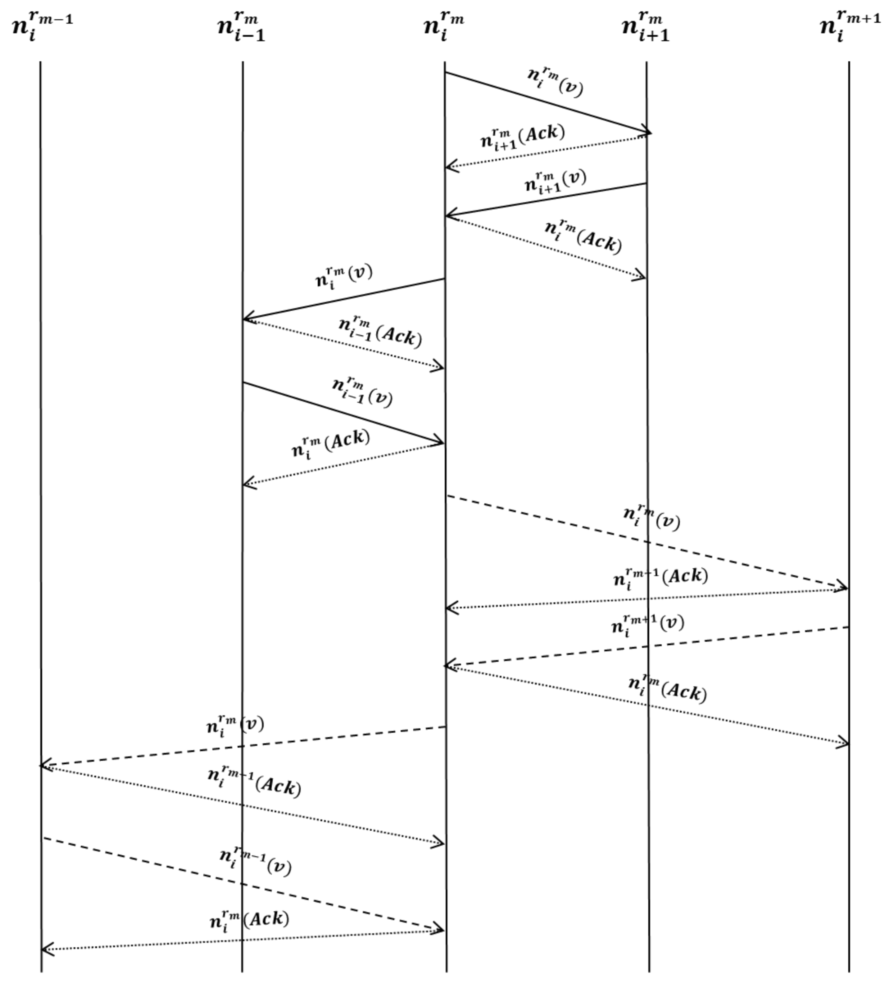

Figure 5.6: MOHA and LMOHA FS-Sync message sequence model.

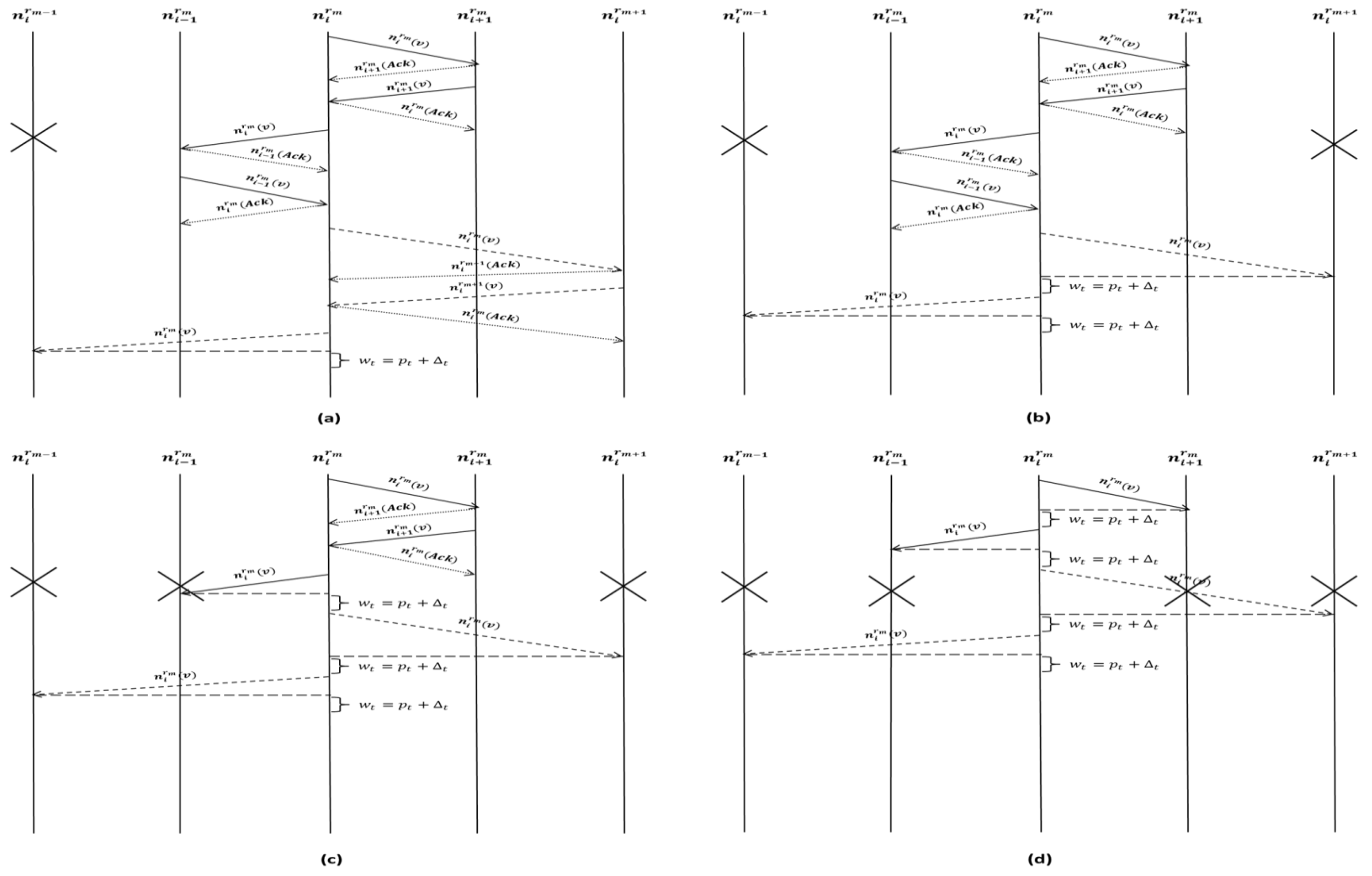

Figure 5.7: MOHA and LMOHA FS-Sync message sequence model scenarios. (a) One adjacent node is inactive. (b) Two adjacent nodes are inactive. (c) Three adjacent nodes are inactive. (d) All adjacent nodes are inactive.

From Figure 5.6 it can be seen that a full communication exchange of an active node will involve sending and receiving 16 frames. This excludes retransmissions due to errors and/or lost messages. Hence, the total exchange time for a node in such a scenario can be estimated as $T_{exch} = 16T_{send} + 16\Delta_t$. In the scenario shown in Figure 5.7 (a) the adjacent node in the lower row is inactive. Hence, node $n_i^{r_m}$ will either receive a *not sending* preamble from $n_i^{r_{m-1}}$ or wait for p_t to retransmit the sent frame. Consequently, the time required for such an exchange scenario will be $T_{exch} = 13T_{send} + 14\Delta_t + p_t$. The same applies if any node is inactive and other nodes are active. The next scenario described in Figure 5.7 (b) is when two adjacent nodes to $n_i^{r_m}$ are de-activated. In such a scenario, the time estimate is given by: $T_{exch} = 10T_{send} + 12\Delta_t + 2p_t$. The next scenario described in Figure 5.7 (c) is when three adjacent nodes to $n_i^{r_m}$ are de-activated. In such a scenario the time estimate will be $T_{exch} = 7T_{send} + 10\Delta_t + 3p_t$. The last scenario shown in Figure 5.7 (d) requires all inactive adjacent nodes to send a *not sending* preamble to allow $n_i^{r_m}$, to complete the exchange transmission process. In this case, the time estimate will be $T_{exch} = 4T_{send} + 8\Delta_t + 4p_t$. Since p_t is less than T_{send}, and all messages are communicated in parallel, the maximum exchange time for a network that runs a FS-Sync model for the MOHA and LMOHA schemes can be estimated as follows:

$$T_{exch} = 16T_{send} + 16\Delta_t \tag{5.11}$$

And the minimum time will be as follows:

$$T_{exch} = 4T_{send} + 8\Delta_t + 4p_t \quad (5.12)$$

Assuming that there is at least one active node in the entire network.

In the MOHA implementation, in order to perform the MOHA's report communications, an activated edge node (as described in Definition 3.6) will send a MAC frame that has a *sending* preamble, with its location information ($node_{location}$) in the data field to the S&I, and receives an Ack to mark completion of communication. If the active node is not activated as an edge node, the active node sends a *not sending* preamble. This is to minimise the waiting time for the S&I as the preamble sending time is less than the frame sending time. Figure 5.8 shows the two possible scenarios of a communication process for an active node ($n_i^{r_m}$) and the S&I ($S\&I$), where *i* is the node's position in its row and *m* is the node's row number. Figure 5.8 (a) shows the communication sequence if the active node is activated as an edge node and Figure 5.8 (b) shows the sequence when it is inactive as an edge node.

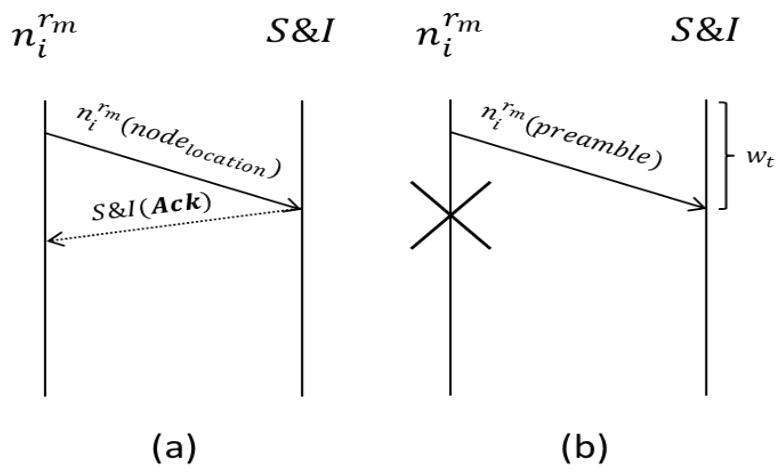

Figure 5.8: MOHA's edge nodes communication phase for FS-Sync message sequence model scenarios.

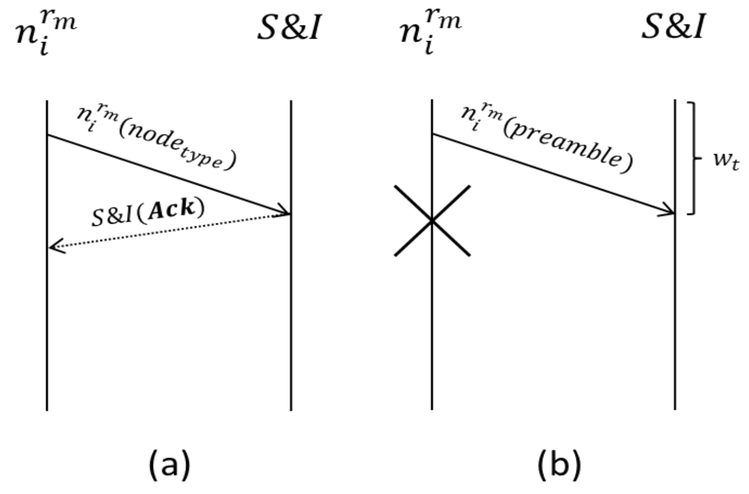

Figure 5.9: LMOHA's plus, T, corner, and edge nodes communication phase for FS-Sync message sequence model scenarios.

In the first scenario, the communication time for the activated edge node ($n_i^{r_m}$) can be estimated in a similar way to the first communication scenario presented for the FS-Async model in the previous sub-section. Taking the Ack frames into account, the maximum report communication time for the network can be estimated as follows:

$$T_{report} = 2N_{edge}(T_{send} + \Delta_t) \tag{5.13}$$

The minimum communication time can be estimated using the second scenario. In this scenario, if the active node $n_i^{r_m}$ is inactive as an edge node, the minimum report communication time of this MAC model will be similar to the minimum report communication time of the FS-Async model that was estimated based on Equation 5.4. The total communication time (T_{comm}^{MOHA}) of the network can be calculated as the summation of T_{report} and T_{exch}. Accordingly, the maximum total communication time of such a model for the MOHA scheme can be estimated as follows:

$$T_{comm}^{MOHA} = 2(N_{edge} + 8)(T_{send} + \Delta_t) \quad (5.14)$$

This is twice the maximum time of the FS-Async communication time for the MOHA scheme presented in Equation 5.5. However, the minimum communication time for both models remains the same as that derived using Equation 5.6.

On the other hand, in the LMOHA implementation, to perform the LMOHA's report communications, an activated plus, T, corner, and edge node (as described in Definition 4.6) will send a MAC frame that has a *sending* preamble, with its type information ($node_{type}$) in the data field to the S&I and receives an Ack to end the communication. If the active node is not activated as a plus, T, corner, or edge node, the active node sends a *not sending* preamble. This is to minimise the waiting time for the S&I as the preamble sending time is less than the frame sending time. Figure 5.9 shows the two possible scenarios of a communication process for an active node ($n_i^{r_m}$) and the S&I ($S\&I$), where *i* is the node's position in its row and *m* is the node's row number. Figure 5.9 (a) shows the communication sequence if the active node is activated as a plus, T, corner, or edge node and Figure 5.9 (b) shows the sequence when it is inactive as a plus, T, corner, or edge node.

In the first scenario, the report communication time for the activated plus, T, corner, or edge node ($n_i^{r_m}$) can be estimated in a similar manner as that for the first communication scenario presented for FS-Async in the previous sub-section. Taking the Ack frames into account, the maximum report communication time for the network in the LMOHA scheme can be estimated as follows:

$$T_{report} = 2(N_{plus}+N_T + N_{corner} + N_{edge})(T_{send} + \Delta_t) \quad (5.15)$$

The minimum report communication time can be estimated using the second scenario. In this scenario, if the active node $n_i^{r_m}$ is inactive as a plus, T, corner, or edge node, the minimum report communication time of this MAC model will be similar to the minimum report communication time of the FS-Async model that has been estimated according to Equation 5.8. The total communication time (T_{comm}^{LMOHA}) of the network can be calculated as the summation of T_{report} and T_{exch}. Accordingly, the maximum total communication time in such a model for the LMOHA scheme can be estimated as follows:

$$T_{comm}^{LMOHA} = 2(N_{plus}+N_T + N_{corner} + N_{edge} + 8)(T_{send} + \Delta_t) \quad (5.16)$$

This is twice the maximum time of the FS-Async communication time for the LMOHA scheme presented in Equation 5.9. However, the minimum communication time for both models remains the same as that derived in Equation 5.10.

5.2.3 Multi-channel MOHA and LMOHA models

In multi-channel (MC) models, a node is capable of handling multiple communications simultaneously. Similar to the frame-slotted models, MC can work on asynchronous and synchronous modes. Figure 5.10 shows MC-Async exchange communications when all nodes are activated. Figure 5.11 shows four possible scenarios: (a) the adjacent node in the lower row ($n_i^{r_{m-1}}$) is inactive, (b) the adjacent node in the next higher row ($n_i^{r_{m+1}}$) and the adjacent node in the lower row ($n_i^{r_{m-1}}$) are inactive, (c) the previous node in its row ($n_{i-1}^{r_m}$), the adjacent node in the next higher row ($n_i^{r_{m+1}}$), and the adjacent node in the lower row ($n_i^{r_{m-1}}$) are inactive, and (d) all nodes adjacent to node $n_i^{r_m}$ are inactive.

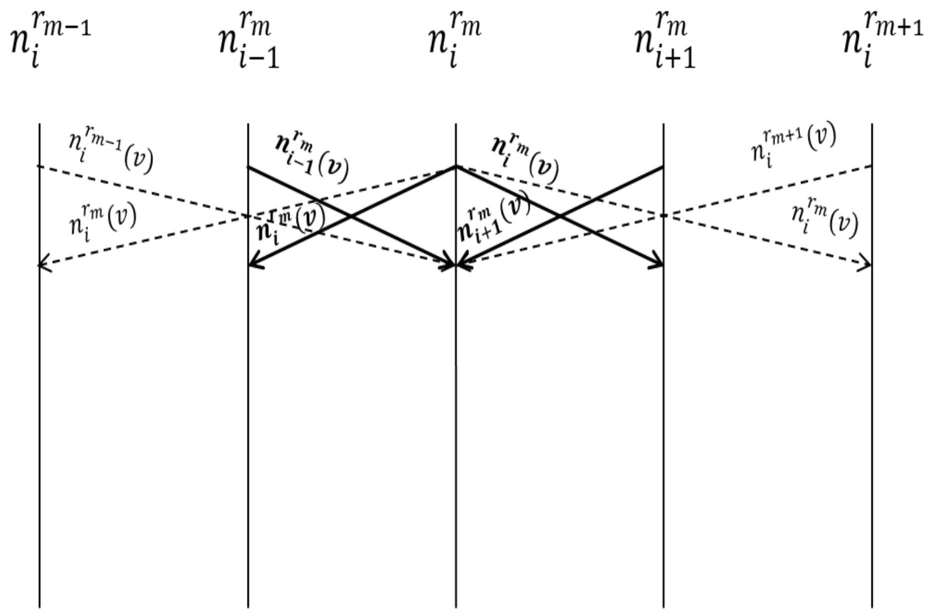

Figure 5.10: MOHA and LMOHA MC-Async message sequence model.

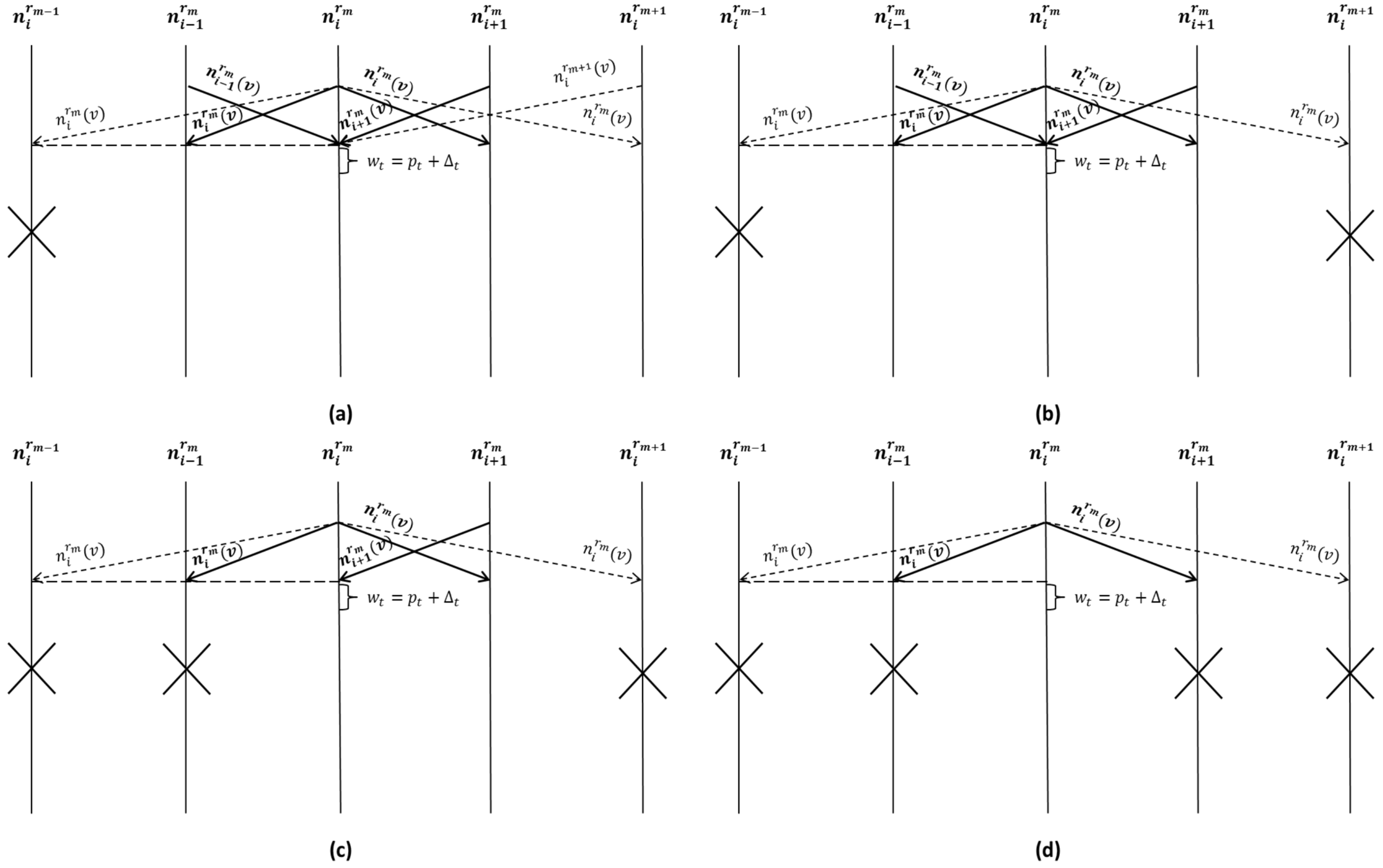

Figure 5.11: MOHA and LMOHA MC-Async message sequence model scenarios. (a) One adjacent node is inactive. (b) Two adjacent nodes are inactive. (c) Three adjacent nodes are inactive. (d) All adjacent nodes are inactive.

From Figure 5.10, if all nodes are active, all nodes will exchange frames at the same time and the time estimate can be determined as $T_{send} + \Delta_t$. However, when there is an inactive node such as the ones presented in Figures 5.11 (a), (b), (c) and (d), the time estimate is $T_{send} + p_t + 2\Delta_t$. This can be considered as the maximum exchange time as it is higher than the time estimate for the first scenario. In MOHA's report communications, the report communication time for the MC-Async model can be estimated in a similar way to the FS-Async mode (see Figure 5.4). Since all activated edge nodes will simultaneously report their locations information to the S&I, the maximum report communication time for the network can be estimated as follows:

$$T_{report} = T_{send} + \Delta_t \tag{5.17}$$

The minimum communication time can be estimated using the second scenario in Figure 5.4. In this scenario, if the active node n_i^{rm} is inactive as an edge node, the waiting time for the S&I can be estimated as $w_t = p_t + \Delta_t$. Since all active nodes will communicate with the S&I at the same time, the minimum report communication time for the entire network can be estimated as follows:

$$T_{report} = p_t + \Delta_t \tag{5.18}$$

Thus, the maximum communication time for a network running the MC-Async for the MOHA scheme can be estimated as follows:

$$T_{comm}^{MOHA} = 2T_{send} + p_t + 3\Delta_t \tag{5.19}$$

And the minimum time for the MOHA scheme can be estimated as follows:

$$T_{comm}^{MOHA} = T_{send} + p_t + 2\Delta_t \tag{5.20}$$

On other hand, Like in the MOHA's report communications, the LMOHA's report communication time for the MC-Async model can be estimated in a similar way to the FS-Async mode (see Figure 5.5). Since all activated plus, T, corner, and edge nodes will simultaneously report their types information to the S&I, the maximum report communication time for the network can be estimated as follows:

$$T_{report} = T_{send} + \Delta_t \tag{5.21}$$

The minimum communication time can be estimated using the second scenario in Figure 5.5. In this scenario, if the active node n_i^{rm} is inactive as a plus, T, corner, or edge node, the waiting time for the S&I can be estimated as $w_t = p_t + \Delta_t$. Since all active nodes will communicate with the S&I at the same time, the minimum report communication time for the entire network can be estimated as follows:

$$T_{report} = p_t + \Delta_t \tag{5.22}$$

Thus, the maximum communication time for a network running the MC-Async for the LMOHA scheme can be estimated as follows:

$$T_{comm}^{LMOHA} = 2T_{send} + p_t + 3\Delta_t \tag{5.23}$$

And minimum time for LMOHA scheme can be estimated as follows:

$$T_{comm}^{LMOHA} = T_{send} + p_t + 2\Delta_t \tag{5.24}$$

Equations 5.19, 5.20, 5.23, and 5.24 show that both proposed schemes have identical minimum and maximum communication times. The main reason for this is the nodes' ability, provided by the MC-Async model, to handle multiple communications simultaneously.

In synchronous communication, each receiving node is expected to reply to the sending node with an Ack frame to confirm the receipt of the message. Figure 5.12 shows the exchange communication sequence in an MC-Sync model. Figure 5.13 shows four possible scenarios: (a) one adjacent node is inactive, (b) two adjacent nodes are inactive, (c) three adjacent nodes are inactive and (d) all adjacent nodes are inactive.

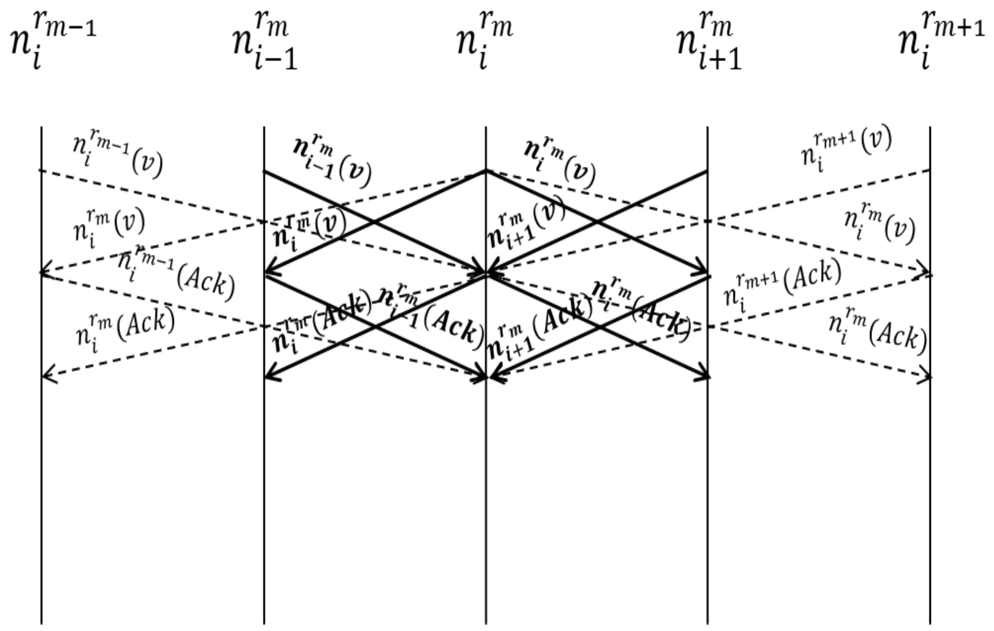

Figure 5.12: MOHA and LMOHA MC-Sync message sequence model.

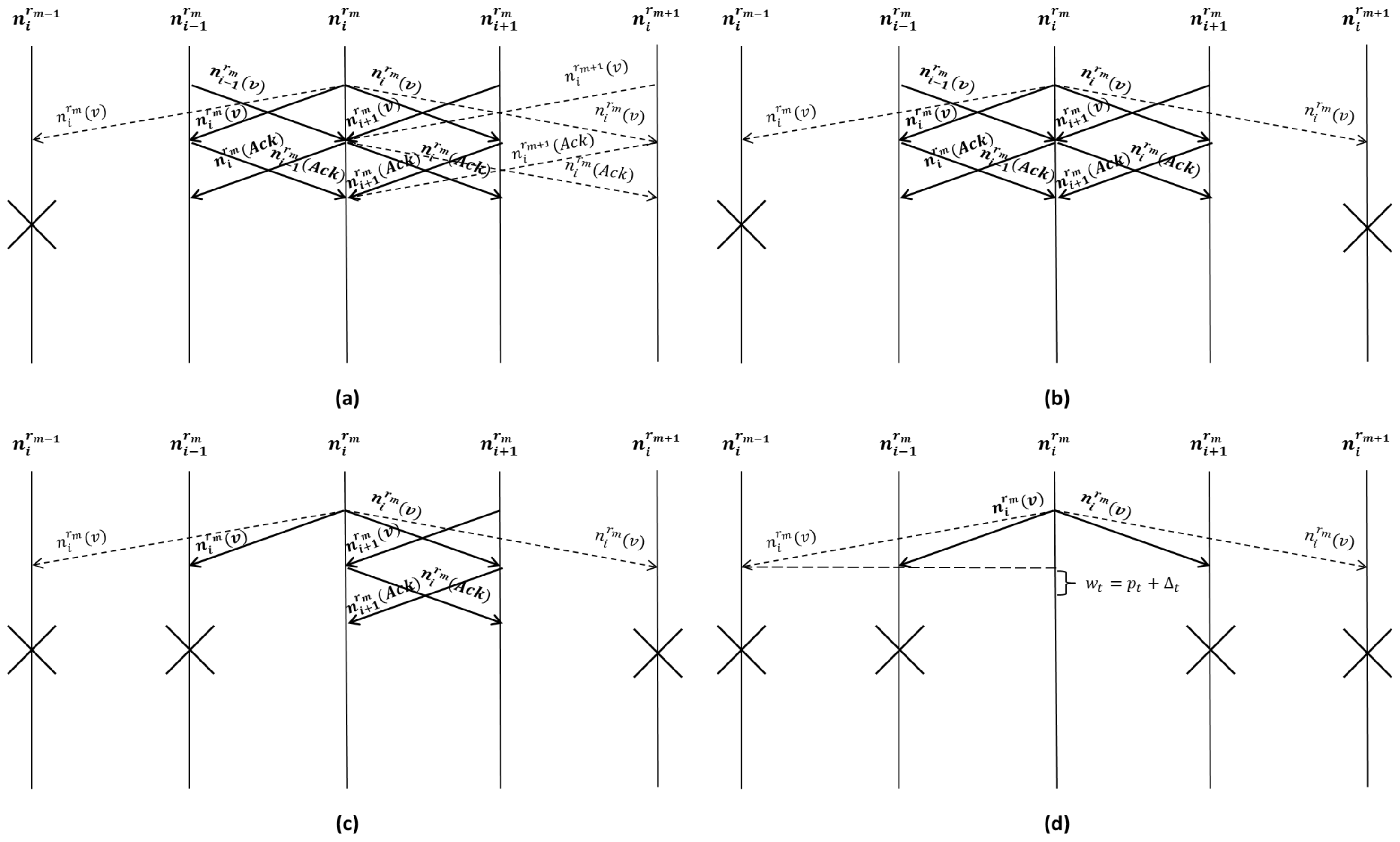

Figure 5.13: MOHA and LMOHA MC-Sync message sequence model scenarios. (a) One adjacent node is inactive. (b) Two adjacent nodes are inactive. (c) Three adjacent nodes are inactive. (d) All adjacent nodes are inactive.

From Figure 5.12, the message exchange time can be estimated as $2(T_{send} + \Delta_t)$. This estimate includes Ack frames. The same estimate can be applied to the first, second and third scenarios shown in Figure 5.13 (a), (b) and (c) as the waiting time for responses from inactive nodes overlaps with the exchange time for active nodes. In the fourth scenario shown in Figure 5.13 (d), time estimate will be $T_{send} + p_t + 2\Delta_t$ as active node n_i^{rm} will be waiting for a *not sending* preamble. In the MOHA's report communications, the report communication time for the MC-Sync model can be estimated in a similar way to the FS-Sync mode (see Figure 5.8). Since all activated edge nodes will report their locations information to the S&I simultaneously, the maximum report communication time for the network can be estimated as follows:

$$T_{report} = 2(T_{send} + \Delta_t) \tag{5.25}$$

The minimum communication time can be estimated using the second scenario in Figure 5.8. In this scenario, if the active node n_i^{rm} is inactive as an edge node, the waiting time for the S&I can be estimated as $w_t = p_t + \Delta_t$. Since all active nodes will communicate with the S&I at the same time, the minimum report communication time for the entire network can be estimated as follows:

$$T_{report} = p_t + \Delta_t \tag{5.26}$$

Thus, the maximum communication time for a network running MC-Sync for MOHA scheme can be estimated as follows:

$$T_{comm}^{MOHA} = 4(T_{send} + \Delta_t) \tag{5.27}$$

And the minimum time for the MOHA scheme can be estimated as follows:

$$T_{comm}^{MOHA} = T_{send} + 2p_t + 3\Delta_t \tag{5.28}$$

On other hand, similar to the MOHA's report communications, the LMOHA's report communication time for the MC-Sync model can be estimated in a similar way to the FS-Sync mode (see Figure 5.9). Since all activated plus, T, corner, and edge nodes will report their types information to the S&I at the same time, the maximum report communication time for the network can be estimated as follows:

$$T_{report} = 2(T_{send} + \Delta_t) \tag{5.29}$$

The minimum communication time can be estimated using the second scenario in Figure 5.9. In this scenario, if the active node n_i^{rm} is inactive as a plus, T, corner, or edge node, the waiting time for the S&I can be estimated as $w_t = p_t + \Delta_t$. Since all active nodes will simultaneously communicate with the S&I, the minimum report communication time for the entire network can be estimated as follows:

$$T_{report} = p_t + \Delta_t \tag{5.30}$$

Thus, the maximum communication time for a network running the MC-Sync for the LMOHA scheme can be estimated as follows:

$$T_{comm}^{LMOHA} = 4(T_{send} + \Delta_t) \tag{5.31}$$

And the minimum time for LMOHA scheme can be estimated as follows:

$$T_{comm}^{LMOHA} = T_{send} + 2p_t + 3\Delta_t \tag{5.32}$$

Equations 5.27, 5.28, 5.31, and 5.32 show that both proposed schemes have identical minimum and maximum communication times. The main reason for this is the nodes' ability to handle multiple communications simultaneously provided by the MC-Sync model.

Table 5.3 summarises the limits of communication time overhead for the message sequence models of both schemes. The analysis of the experiment results from the four sequence models will be conducted in the next section to evaluate the communication overhead of the MOHA and LMOHA networks for each model using different pattern types.

Table 5.3: Summary of communication time overhead (T_{comm}) limit estimates for each MOHA and LMOHA message sequence model.

Model	**Scheme**	**Minimum time estimation**	**Maximum time estimation**
FS-Async	MOHA	$(N_{active} + 4)p_t + (N_{active} + 8)\Delta_t + 4T_{send}$	$(N_{edge} + 8)(T_{send} + \Delta_t)$
	LMOHA	$(N_{active} + 4)p_t + (N_{active} + 8)\Delta_t + 4T_{send}$	$(N_{plus} + N_T + N_{corner} + N_{edge} + 8)(T_{send} + \Delta_t)$
FS-Sync	MOHA	$(N_{active} + 4)p_t + (N_{active} + 8)\Delta_t + 4T_{send}$	$2(N_{edge} + 8)(T_{send} + \Delta_t)$
	LMOHA	$(N_{active} + 4)p_t + (N_{active} + 8)\Delta_t + 4T_{send}$	$2(N_{plus} + N_T + N_{corner} + N_{edge} + 8)(T_{send} + \Delta_t)$
MC-Async	MOHA	$T_{send} + p_t + 2\Delta_t$	$2T_{send} + p_t + 3\Delta_t$
	LMOHA	$T_{send} + p_t + 2\Delta_t$	$2T_{send} + p_t + 3\Delta_t$
MC-Sync	MOHA	$T_{send} + 2p_t + 3\Delta_t$	$4(T_{send} + \Delta_t)$
	LMOHA	$T_{send} + 2p_t + 3\Delta_t$	$4(T_{send} + \Delta_t)$

5.3 MOHA and LMOHA Communicational Overhead Analysis

This section presents an experimentally-derived evaluation of the MOHA and LMOHA schemes' communicational overhead in terms of time and energy requirements. As discussed in Chapter 2, network communications are considered to be the most time and energy consuming tasks in WSNs. Therefore, through experiments, we estimate the MOHA and LMOHA networks' learning time cycle durations and energy consumption based on communicational requirements. This section begins with an analysis of the effects of the nodes activation technique on communications in the network and then it presents a time and energy analysis.

5.3.1 Node activation analysis

The nodes activation techniques of the MOHA and LMOHA schemes were discussed in sections 3.4.2 and 4.3.1 respectively. In the MOHA scheme, this technique or process involves two types of activations, namely, node and edge node activations, which are described in Definitions 3.2 and 3.5 respectively. The LMOHA scheme also has mainly two types of activations, namely, node and node types activations. However, the node types activation in the LMOHA scheme is divided into four types: plus node, T node, corner node, and edge node activations. These activation types were described in Definitions 4.2, 4.3, 4.4, and 4.5 respectively. The nodes activation technique determines the total number of the network's communications, as only the activated nodes in the network participate in the learning process. This sub-section utilises the *shapes* and *stars* datasets presented in the previous chapter (Chapter 4) to estimate the number of activated nodes for each dataset.

Figure 5.14 shows two examples of nodes activation. Figure 5.14 (a) shows the activation technique of a binary pattern taken from the *shapes* dataset. The pattern size $P_{size} = 65536$ and the threshold value for node activation was set to 1 ($\varphi = 1$). The nodes are assumed to be deployed in a grid to sense the pattern space. We also assume that the network is divided into 256 rows, each of which is utilised to handle 256-bit binary sub-patterns. Since the pattern is binary, activated nodes will form the same shape that appears in the pattern. The number of activated nodes in this example is 4783 nodes out of 65536. Please note that both schemes have an equal number of activated nodes, which is the result of using the same first step to activate the nodes. In Figure 5.14 (b), a grey scale star map is taken from the *stars* dataset as the pattern. The possible value of each element in this pattern is $v = \{0,1,2,\dots,255\}$. In this instance, the pattern size $P_{size} = 40000$ and the threshold value for node activation was set to 150 ($\varphi = 150$). We assume that the network is divided into 200 rows, each of which is utilised to handle 200 sub-patterns. The number of activated nodes in this example is 99. It can be clearly seen from the examples shown in Figure 5.14 that the number of nodes that conduct exchange communications has been reduced from the pattern size to 4783 in the first example and to 99 in the second one instead of involving all of the network's nodes in the process (65536 nodes for the first example and 40000 for the second one).

Figure 5.14 also presents the MOHA's activated edge nodes and the LMOHA's activated plus, T, corner, and edge nodes. The threshold value for edge determination was set to 1 ($\varphi = 1$) for the *shapes* dataset example and 150 ($\varphi = 150$) for the example of *stars* dataset. In the MOHA scheme, the total number of activated edge nodes in Figure 5.14 (a) is 263 nodes, whereas it is 42 nodes for the example shown in Figure 5.14 (b). On the other hand, in the LMOHA scheme, the total number of activated plus, T, corner, and edge nodes for the *shapes* dataset example was 4783 nodes, whereas it was 87 nodes for the example of *stars* dataset, shown in Figure 5.14 (a) and (b) respectively. Since the pattern in the first example is binary, each activated node will be activated as plus, T, corner, or edge node. These techniques limit the number of reporting nodes (to the S&I) in the network to the activated edge nodes in the MOHA implementation and the activated plus, T, corner, and edge nodes in the LMOHA implementation and relieves the rest of activated nodes from sending report messages. This reduces the resource consumption of these nodes.

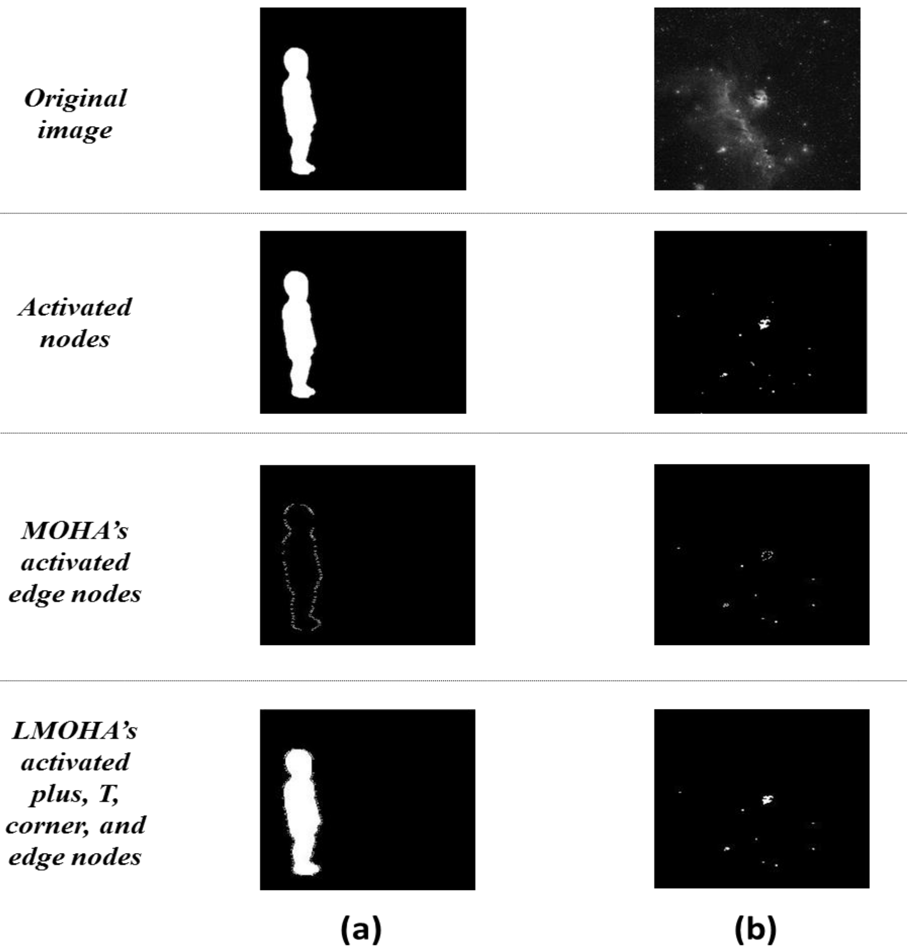
Original image
Activated nodes
MOHA's activated edge nodes
LMOHA's activated plus, T, corner, and edge nodes
(a)
(b)

Figure 5.14: Example of the proposed schemes' nodes activation technique for: (a) *shapes* dataset, (b) *stars* dataset.

Table 5.4: Average number of activated nodes for MOHA scheme.

Dataset	Nodes (Network size)	Average activated nodes	Average activated edge nodes
Shapes	65536	4883	291
Stars	40000	213	31

Table 5.5: Average number of activated nodes for LMOHA scheme.

Dataset	Nodes (Network size)	Average activated nodes	Average activated plus, T, corner, and edge nodes
Shapes	65536	4883	4883
Stars	40000	213	119

Table 5.4 and 5.5 summarises the average number of activated nodes for both datasets using the proposed schemes. Both tables clearly show that the number of activated nodes that will conduct the node's exchange communications is small compared to the total number of nodes. Moreover, the total number of activated nodes represents only 25.67% of the total number of nodes for the *shapes* dataset and 0.53% for the *stars* dataset. Table 5.4 shows that in MOHA implementation, the number of activated edge nodes that will participate in report communications represents only 0.74% of the network size for the *shapes* dataset and 0.08% for the *stars* dataset. On the other hand, Table 5.5 shows that in the LMOHA implementation, the total number of activated plus, T, corner, and edge nodes that will participate in report communications represents only 25.67% of the network size for the *shapes* dataset and 0.30% for the *stars* dataset. This reflects the amount of reduction in communicational overhead that can be achieved by using the proposed schemes. Moreover, Tables 5.4 and 5.5 clearly show that in the LMOHA implementation, the average number of activated nodes participating in the learning process is higher compared with the average number of activated nodes in the MOHA implementation. This is mainly because the number of activated nodes that participate in report communications is lower in the MOHA approach compared with that in the LMOHA approach.

5.3.2 Energy and Time Analysis

In this sub-section, we analyse the MOHA and LMOHA schemes from a practical perspective. The main objective is to estimate the lifetime and execution duration of the networks utilising the communicational models (MAC protocols) presented in the beginning of this chapter. To achieve this goal, we ran a simulation program that creates and runs MOHA and LMOHA networks on sensor nodes and obtains energy and time readings. As discussed previously, data transmission is the most time and energy consuming process in WSNs. Therefore, the simulation evaluates these parameters based on the communications involved in running the network.

Table 5.6: The assumptions utilised in the energy and time analysis.

Sensor type	Mica 2
Communication data rate	128 Kbps
Frame size	49 Bytes
Time to send or receive a full frame (T_{send})	3.0625 ms
Preamble field size	8 bytes
Time to send a one preamble	0.5 ms
Energy required to send a full frame	2.9 mJ
Energy required to receive a full frame	1.4 mJ
Error rate (Δt)	32 µs
Battery capacity	324 joules

We ran the simulation on sensor nodes based on the following assumptions, which are summarised in Table 5.6 [273-275]:

1. The sensor nodes are Mica 2 type.

2. The size of the frame is 49 bytes, which includes preamble, addresses, control, data checksum, flag, and other fields.
3. The size of the preamble field is 8 bytes.
4. Communication data rate is 128 Kbps, which means that sending a full frame takes 3.0625 ms (milliseconds) and sending a one preamble takes 0.5 ms.
5. Transmitting one byte costs 59.2 μJ (micro joules) and receiving one byte costs 28.6 μJ, which means that sending a full frame costs (49X59.2 μJ) 2.9 mJ (milli joules) and receiving a full frame costs 1.4 mJ.
6. The maximum error is transmission is equal to a clock cycle and the clock cycle takes 32 μs (microseconds), which means that $\Delta t = 32\ \mu s$.
7. Each sensor is equipped with 3V (volts) 30mAh battery. This battery is chosen as it is one of smallest commercial batteries available. As a result, the capacity of a full sensor battery is 324 joules.

To run the simulations, we utilised the *shapes* and *stars* datasets presented in sub-sections 4.6.1 and 4.6.2. The first dataset represents binary shape patterns of size 256×256 pixels. This *shapes* dataset, represents uniform patterns. For the second dataset, the *stars dataset*, we chose grey scale star maps of size 200×200 pixels, representing a non-uniform pattern type. We assume that the sensor nodes are deployed in a grid, where each node obtains a pixel value. This requires 65536 nodes to represent the first network and 40000 nodes to represent the second. Four networks for each dataset were created for each proposed scheme. The total number of simulated networks is sixteen (8 for each proposed scheme). Each network runs in one of the models discussed in section 5.2: FS-Async, FS-Sync, MC-Async, or MC-Sync. We presented each dataset to its associated network to perform a learning process. Each dataset contains both original and altered patterns (instances). The altered patterns can be a result of the original patterns being rotated, scaled, or translated. The total number of instances for the first dataset is 1310 patterns and the second is 1110 patterns.

To evaluate the proposed schemes' communicational and energy overhead, we implemented the parallel KNN presented in [96] for comparison. This scheme is selected because of its simplicity in its computations and time complexity. Figure 5.15 depicts a parallel KNN network. In a parallel KNN network, each node communicates directly with a central unit (i.e. base station). During memorisation, each node stores its associated input value. In recall, each node computes the nearest stored value to the incoming value and reports the difference to the base station, which determines the final decision. Figure 5.16 shows the average number of communications involved in performing the learning cycle for each communicational type (frame slotted and multi-channel) for the *shapes* and *stars* datasets for the proposed schemes. Moreover, Figure 5.17 shows the number of communications required to implement parallel KNN for both datasets.

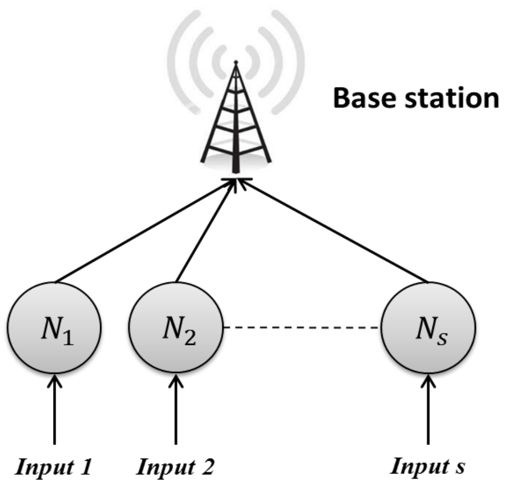

Figure 5.15: Example of a simple parallel KNN network.

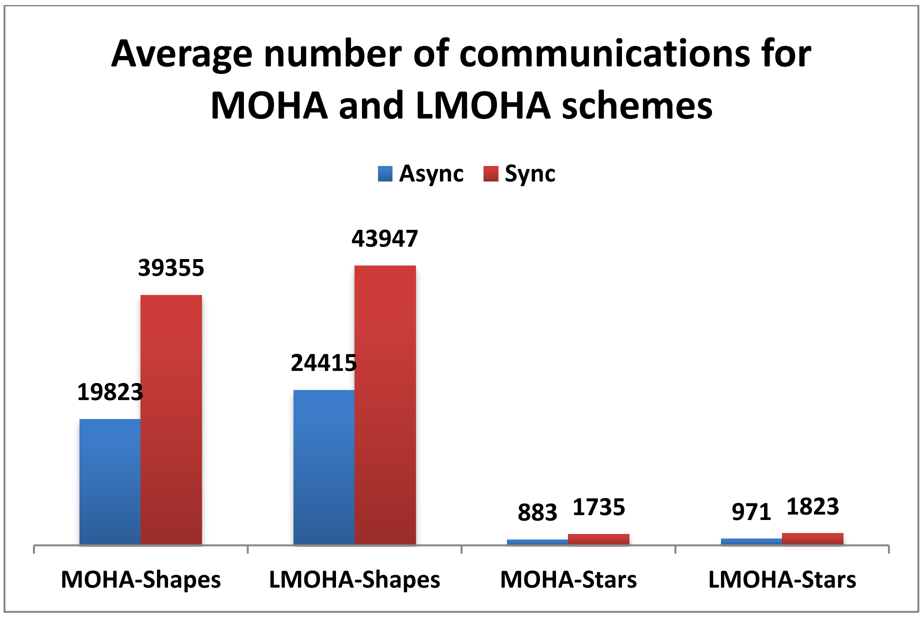

Figure 5.16: Average number of communications for the proposed schemes' Async and Sync for *shapes* and *stars* datasets.

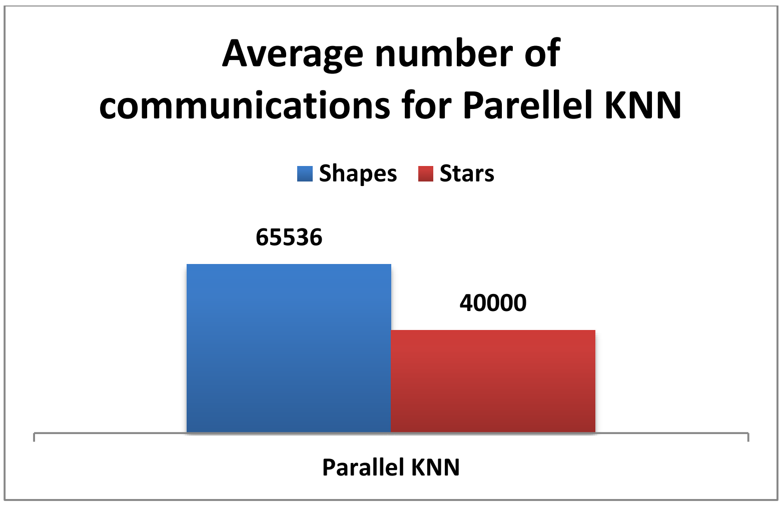

Figure 5.17: Average number of communications for the parallel KNN for *shapes* and *stars* datasets.

Figure 5.15 shows that parallel KNN requires each node to conduct one communication to the base station. As a result, the average number of its communications is equal to the pattern size (P_{size}), as shown in Figure 5.17. Moreover, Figure 5.16 shows that the average number of communications reflects the size of the networks and the number of activated nodes. For instance, in the MOHA implementation, the Async network involves 19823 communications for the 4883 activated nodes and 291 edge nodes for the *shapes* dataset, which includes both MOHA's node's exchange and report communications. This is only 30.25% of the network size and the total communication required by parallel KNN. The Sync network in MOHA implementation involves 39355 communications for the same number of activated nodes for *shapes* dataset, which is 60.05% of the network size and the total communication required by parallel KNN and almost double the average number of communications for the Async communication model. For the *stars* dataset, in the MOHA implementation, the Async network recorded an average of 883 communications, which represents only 2.21% of the network size. The Sync network for the same dataset recorded an average of 1735 communications. This is 4.34% of the network size. On the other hand, in the LMOHA implementation, the Async network involves 24415 communications for the 4883 activated nodes and 4883 activated plus, T, corner, and edge nodes for the *shapes* dataset, which includes both LMOHA's node's exchange and report communications. This is only 37.25% of the network size and the total communication required by parallel KNN. The Sync network in the LMOHA implementation involves 43947 communications for the same number of activated nodes for *shapes* dataset, which is 67.06% of the network size and the total communication required by parallel KNN and almost double the average number of communications for the Async communication model. For the *stars* dataset, in the LMOHA implementation, the Async network recorded an average of 971 communications, which represents only 2.43% of the network size. The Sync network for the same dataset recorded an average of 1823 communications. This is 4.56% of the network size. Figure 5.16 also shows that there are no big different on the average communications numbers recorded by utilising the MOHA and LMOHA schemes. This is due to the fact that most of the communications occur in the node's exchange phase whereas only few occur in the report phase in both schemes.

The average number of communications is expected to have implications for the communicational overhead in terms of energy and time. Figure 5.18 shows the average energy consumption obtained by the simulation for each network type and each dataset using the proposed schemes. The average energy consumption obtained by the simulation for parallel KNN is shown in 5.19. Figure 5.20 shows the average obtained from using both datasets. This is applied for both multi-channel and frame-slotted models, with the assumption that both models require the same amount of energy for each communication. The average values shown in Figures 5.18 and 5.20 represent the average energy required by a node to perform a full learning operation and include the energy consumption when sending and receiving.

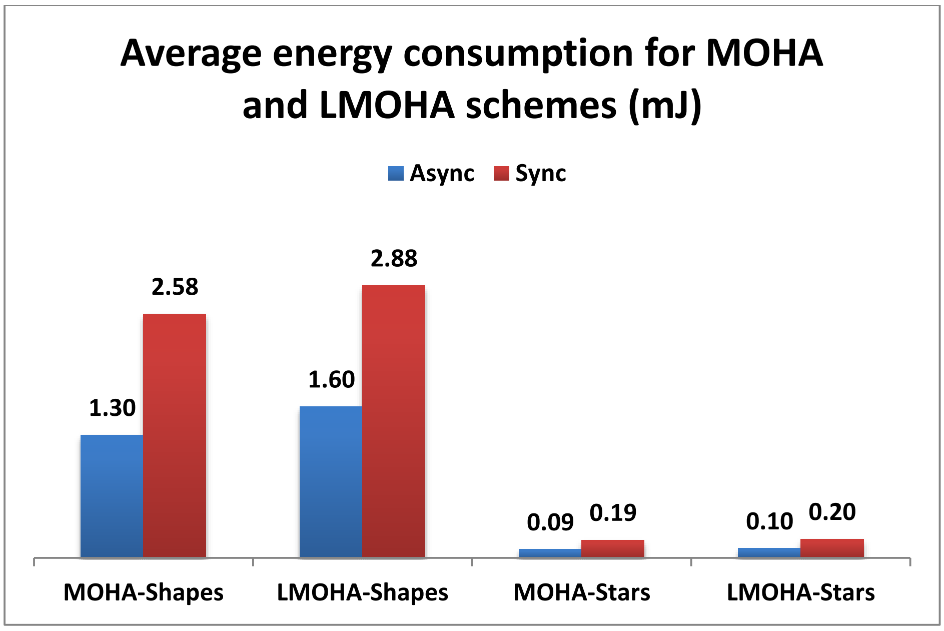

Figure 5.18: Average energy consumption for each of proposed schemes' network for each dataset in mJ, which represents the average energy required by a node to perform a full learning operation.

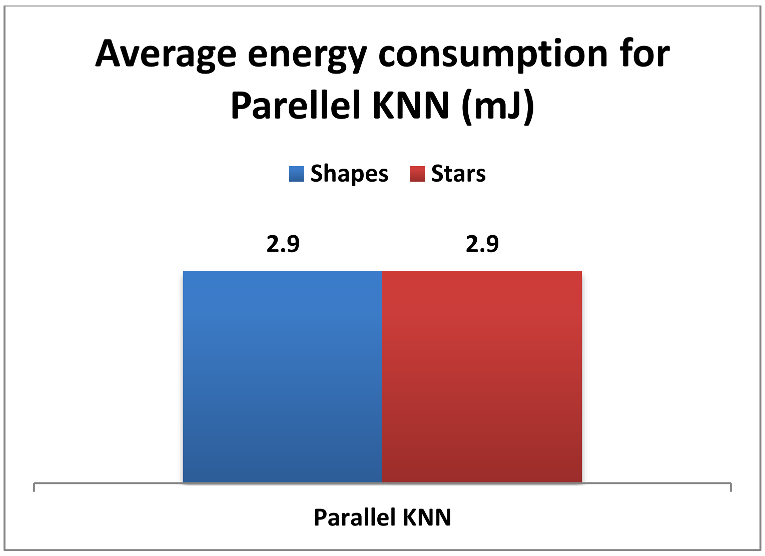

Figure 5.19: Average energy consumption for the parallel KNN for *shapes* and *stars* datasets.

Figure 5.18 confirms that the number of communications has implications for the average energy consumption. However, this is not applicable for the parallel KNN since each learning cycle requires all nodes to participate by sending one message to the base station, which means that the average in this case is equal to the amount of energy required to send one message. Moreover, Figure 5.20 clearly shows that the Sync models require almost twice the energy that the Async models do, which is caused by the *Ack* messages. However, it can be seen that this is not exactly the case as Sync average consumption is 198.57% of the Async average for the MOHA implementation and 181.18% for the LMOHA implementation. This is due to the activation techniques involved in the MOHA and LMOHA schemes, which means that not all sent messages are received. The values shown in Figure 5.20 can be utilised to estimate the average lifetime of the network. Figure 5.21

shows the average lifetime in days for each network based on the assumption that the network receives one pattern per minute.

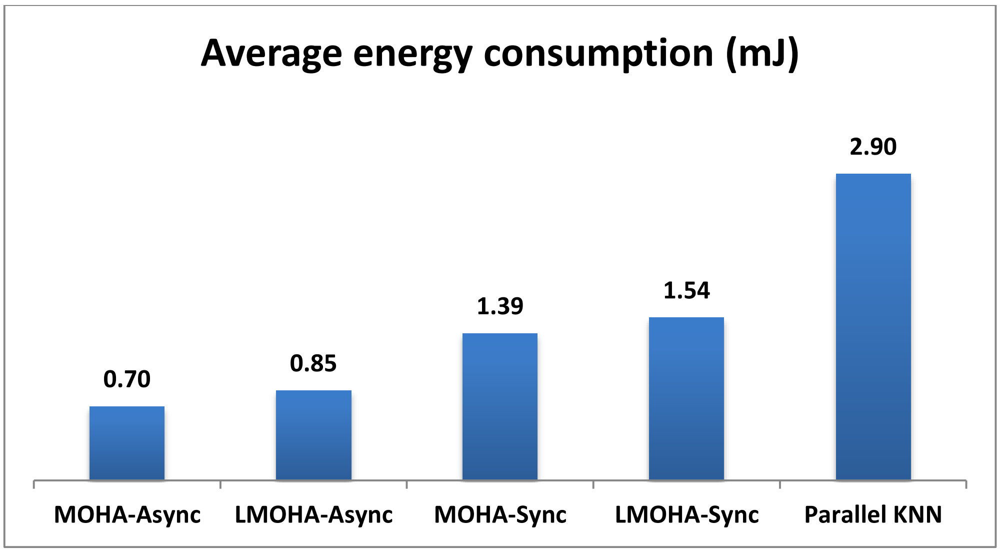

Figure 5.20: Average energy consumption for each of the proposed schemes' networks for both datasets in mJ, which represents the average energy required by a node to perform a full learning operation.

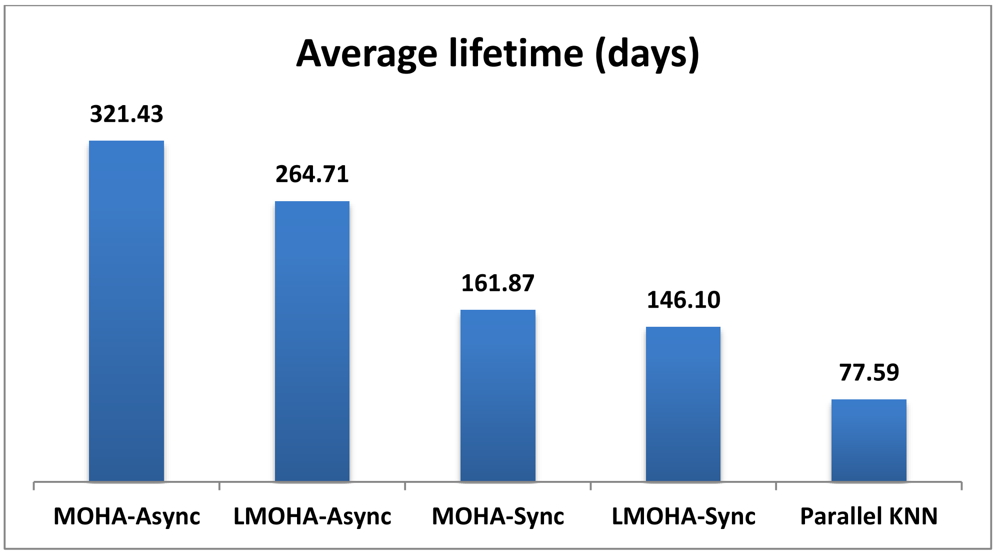

Figure 5.21: Average lifetime for each of the proposed schemes and the parallel KNN.

Figure 5.21 shows that a MOHA Async network can theoretically last for more than ten months if it is used to obtain one pattern per minute using a small 3V 30mAh battery (324 Joules). Under the same conditions, the LMOHA Async network can last for more than eight months. However, other factors may affect that figure, such as physical sensory faults. Figure 5.21 also shows that a MOHA Sync network could last for more than five months and a LMOHA Sync network could last for more than four months. On the other hand, a parallel KNN will last for less than three months under similar conditions. According to Tanenbaum [53], a short WSN lifetime is almost six months. This analysis of a MOHA network shows that a large-scale network can last for almost six months or one year. The analysis also shows that a LMOHA network can last for almost five or nine months. These reflect the amount of reduction in communicational overhead that can be achieved by using the proposed schemes. The analysis also shows that the average lifetimes of LMOHA networks are lower than the average lifetimes of MOHA networks. This is mainly because LMOHA requires, on average, a higher number of communications compared with the MOHA, which leads to the consumption of more energy, thus reducing the network lifetime (see Figure 5.16).

So far, this analysis has shown that in terms of average energy consumption, all sensor nodes are expected to encounter the same amount of communicational overhead. However, the network structure and the type of incoming patterns could affect the behaviour of the network and present different loads to its sensor nodes. Evaluating the networks' behaviour of the proposed schemes, Figure 5.22 presents the distribution of available energy in each sensor utilised in the simulation for Async and Sync communicational models for the *shapes* dataset using the MOHA and LMOHA schemes. Figure 5.23 shows the distribution for the *stars* dataset.

From Figures 5.22 and 5.23, it can be observed that pattern shape (Apple for the *shapes* dataset and a star map for *stars* dataset) can be easily seen through the energy distribution when presenting a small number of instances. However, energy distribution becomes more distributed in the field after being presented with a large number of patterns. This reflects the multi-row structure of the proposed schemes. Both the Async and Sync models have similar energy distribution but with more consumption in the Sync model. Furthermore, it can be observed that the energy distribution in the *stars* dataset is more scattered in the field, while it is concentrated in the middle rows for the *shapes* dataset, which is mainly caused by the non-uniform pattern distribution of the *stars* dataset compared to the *shapes* dataset. Finally, both figures show that energy consumption and distribution of MOHA and LMOHA is almost identical, which is due to the network structure and the fact that the node's exchange communications process consumes the most energy, and this is identical for both proposed schemes.

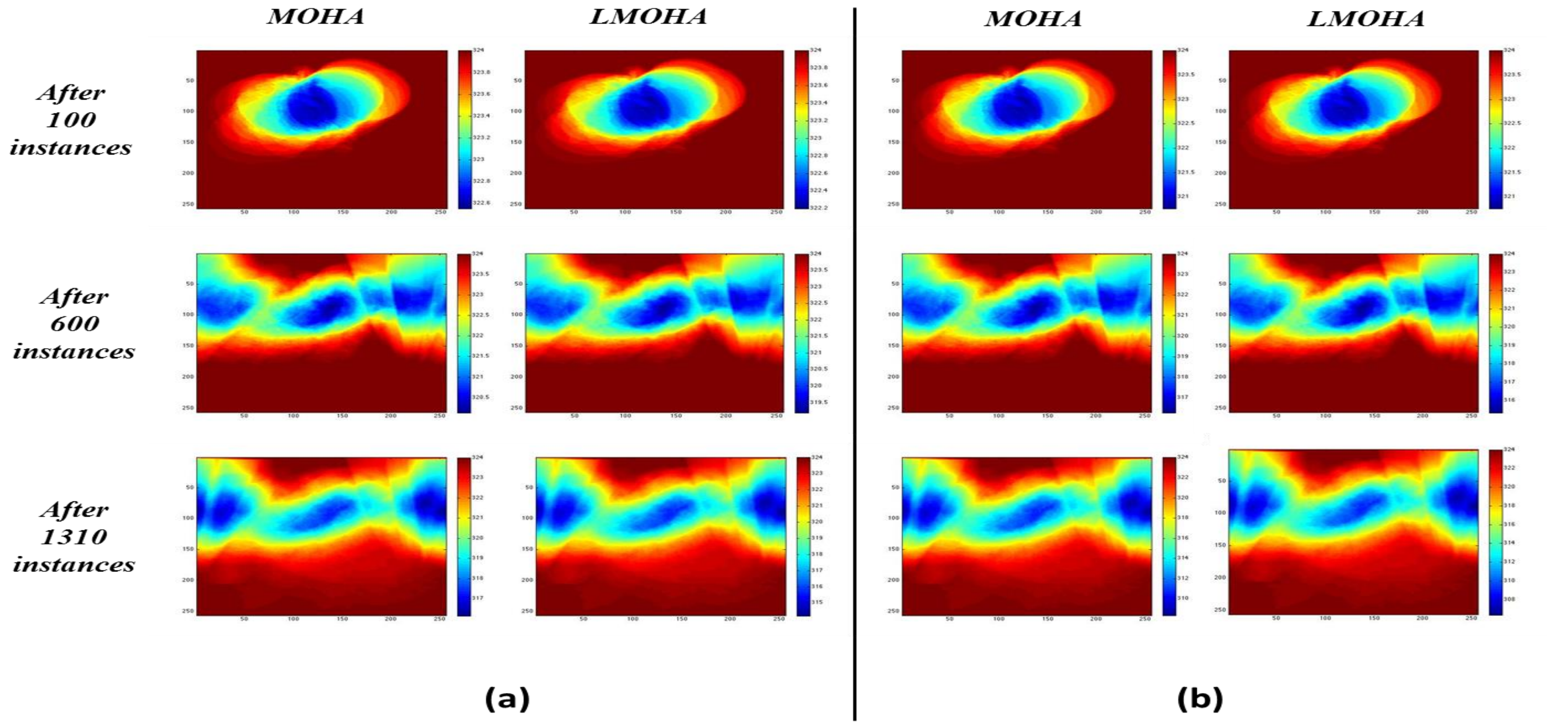

Figure 5.22: Available energy for each node in each one of the proposed schemes' networks dealing with the *shapes* dataset (in joules). (a) Network applying Async model and (b) Network applying Sync model. The nodes are distributed in the field as a grid, where each pixel depicts one node's available energy. The colours are in the range between dark red and dark blue. Dark red indicates more energy resources left in the node. Dark blue indicates less energy available. Colours in between (such as yellow and green) indicate energy levels between dark red and dark blue. Colour bars show exact energy figures.

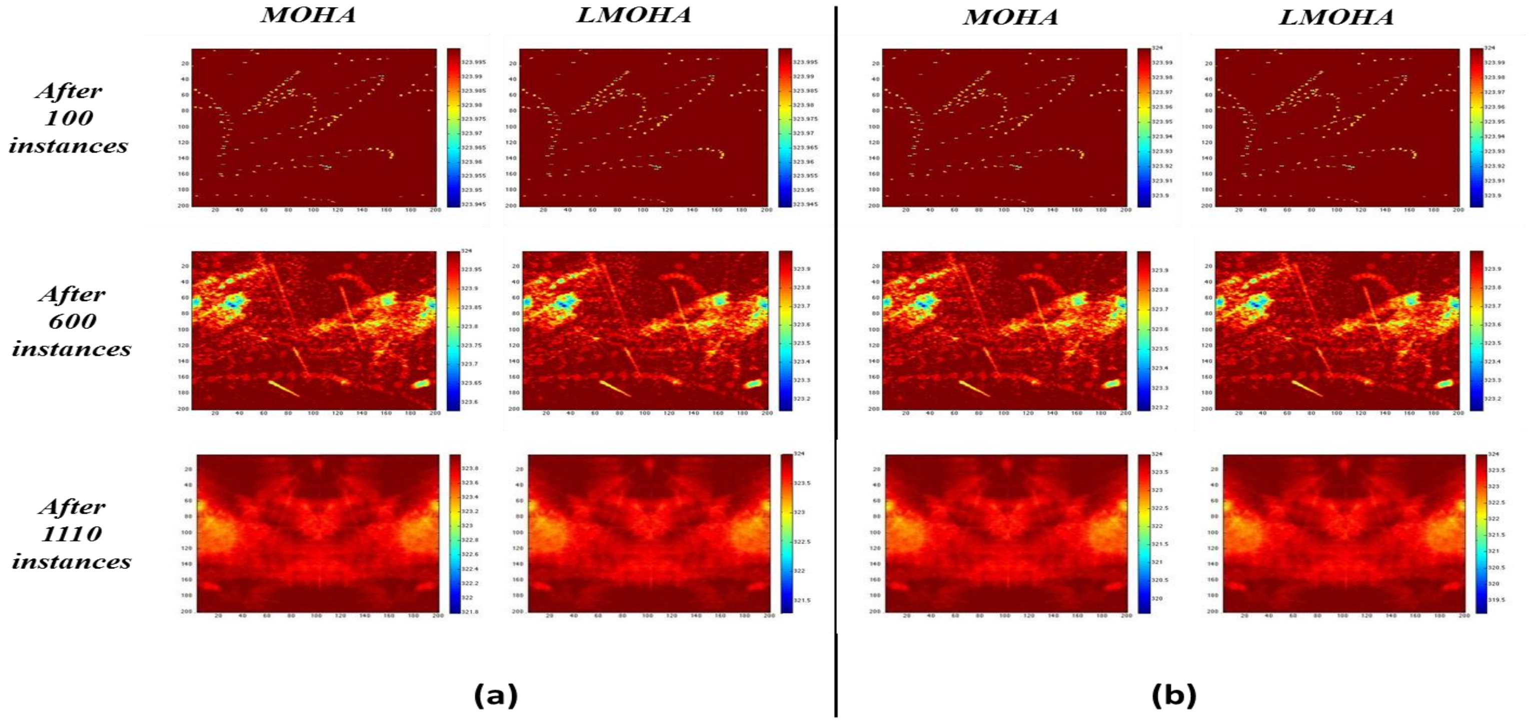

Figure 5.23: Available energy for each node in each one of the proposed schemes' networks dealing with the *stars* dataset (in joules). (a) Network applying Async model and (b) Network applying Sync model. The nodes are distributed in the field as a grid, where each pixel depicts one node's available energy. The colours are in the range between dark red and dark blue. Dark red indicates more energy resources left in the node. Dark blue indicates less energy available. Colours in between (such as yellow and green) indicate energy levels between dark red and dark blue. Colour bars show exact energy figures.

The second factor related to the communications overhead to be analysed in this sub-section is network time. Learning cycle time (memorisation or recall operations) can be estimated in terms of computations and communications. However, the network times for the proposed schemes in terms of computations have been discussed previously in sub-sections 3.7.1 and 4.5.1. Unlike energy analysis, the choice of communicational model (FS or MC) is expected to have an influence on cycle time. Figure 5.24 shows the average learning cycle time in milliseconds (ms) that was obtained from the simulation run for MOHA scheme. Figure 5.25 presents the average learning cycle time in ms that was obtained from the simulation run for LMOHA scheme.

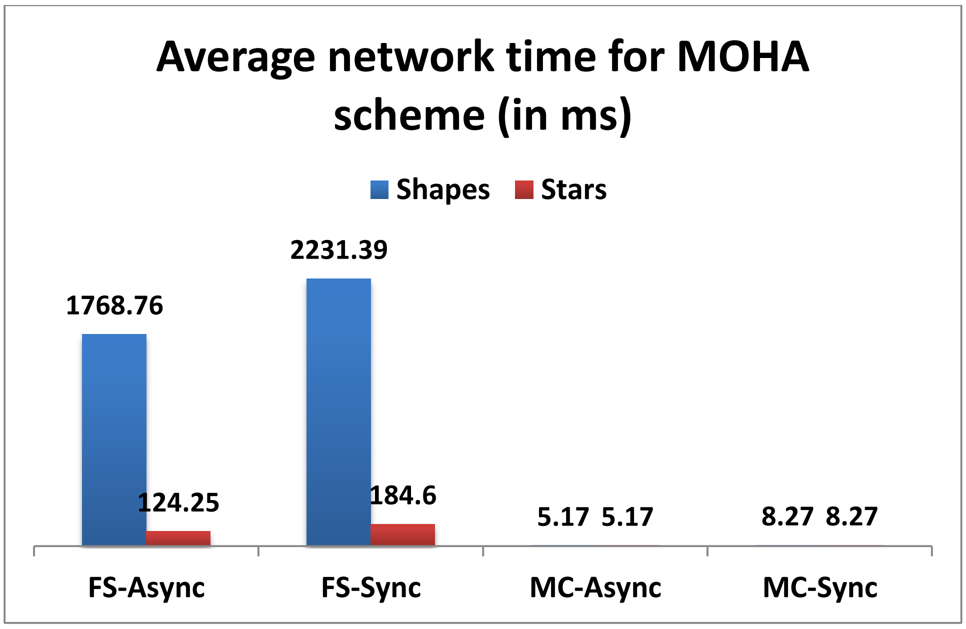

Figure 5.24: Average time of learning cycle in ms for a MOHA network that runs different communicational models.

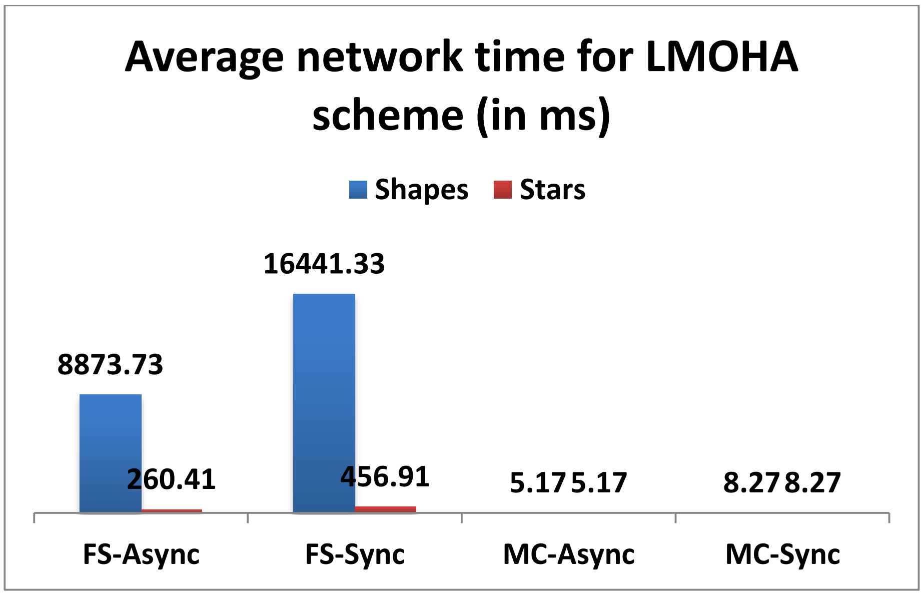

Figure 5.25: Average time of learning cycle in ms for a LMOHA network that runs different communicational models.

From Figure 5.24, the average time of the MOHA scheme ranges between 5.17 and 2231.39 ms depending on the communicational model, network size, and dataset type. This means that a large-scale network of a size between 40,000 and 65,536 nodes to memorise or recall a pattern with a rate of between 1 sample per 2 seconds and 193 samples per second. On the other hand, Figure 5.25 shows that the LMOHA scheme's average time ranges between 5.17 and 16441.33 ms, which means that a large-scale network of a size between 40,000 and 65,536 nodes needs to memorise or recall a pattern with a rate of between 1 sample per 16 seconds and 193 samples per second. The main reason for the LMOHA scheme having a fairly high learning cycle time when it deals with a *shapes* dataset is that this dataset consists of binary patterns. This causes all activated nodes to be involved in reporting to the S&I

(in the base station), which means more reporting time. Moreover, it can be clearly observed from Figures 5.24 and 5.25 that the average network time of the LMOHA scheme utilising FS models is higher compared with the average network time of the MOHA scheme using the same models. This is mainly because the LMOHA requires, on average, a higher number of communications compared with the MOHA, which requires more processing time (see Figure 5.16). It can also be clearly observed from Figures 5.24 and 5.25 that the learning cycle of a *stars* dataset requires less time compared with a *shapes* dataset when frame-slotted models are utilised. This is the result of two main factors. The first factor is the difference in network size utilised for each dataset. The second factor is the type of dataset. The *shapes* dataset represents uniform objects that usually cover a large area of the field compared to a star map. This causes more nodes to be activated and involves a greater number of exchanging and reporting times. Conversely, the *stars* dataset contains non-uniform patterns that are scattered all over the field. This will activate fewer nodes and will involve fewer exchanging and reporting times.

Another important observation that can be made from Figures 5.24 and 5.25 along with Table 5.3 is that there is a huge difference between the time requirement of the frame-slotted models and that of the multi-channel models. In both proposed schemes, the use of the MC model will speed up the exchanging and reporting times by removing the effects of the number of activated nodes in the network and the size of the network in the learning cycle time, which makes the average network time of both the proposed schemes identical. This is because the MC model enables the network to perform its communications in parallel.

This section presented a simulation analysis of the proposed schemes' network communications, including energy and time analysis. The energy analysis showed that both proposed schemes are capable of limiting the number of communications, thereby limiting the use of energy resources. It was shown that the network can have a lifetime of one year with the MOHA scheme and nine months with the LMOHA scheme using one of the smallest batteries in terms of capacity (30 mAh). These are 4 and 3 times higher than other schemes, such as parallel KNN. The time analysis shows that a MOHA network can scale up, while having the ability to converge within a time range between 5.17 ms and 2231.39 ms or a sample rate between a pattern per 2 seconds and 193 patterns per second. On the other hand, the LMOHA network can scale up, while having the ability to converge within a time range between 5.17 ms and 16441.33 ms or a sample rate between a pattern per 16 seconds and 193 patterns per second. These results were obtained by implementing the proposed schemes using different message sequence models and in accordance with the assumptions made in the beginning of the section. These results show that both proposed schemes minimise the communicational overhead, enabling WSNs to scale up efficiently and provide real-time learning capabilities.

The next section will provide a comparison between the two proposed schemes, demonstrating the capabilities of each scheme.

5.4 MOHA and LMOHA Comparison

The MOHA and LMOHA approaches are proposed for resource-constrained sensor nodes and utilise patterns to recognise the occurrence of events of interest in the physical environment. The characteristics and features of the approaches are summarised in Table 5.7.

Table 5.7: Comparison of the features of the MOHA approach and LMOHA approach.

Characteristic	MOHA	LMOHA
Network's Structure	Multi-row	Multi-row
Pattern storage	In S&I (in the base station)	In S&I (in the base station)
Pattern matching	N_{edge}, MV	$N_{plus}, N_T, N_{corner}, N_{edge}$
Node collaboration	✓	✓
Dealing with distorted (noisy) patterns	✓ **For uniform patterns**, 100% recognition accuracy for distortion levels up to 25% and 80% accuracy level for distortion levels up to 45%. **For non-uniform patterns,** 100% recognition accuracy for distortion levels up to 15% and 80% accuracy level for distortion levels up to 35%.	✓ **For uniform patterns**, 100% recognition accuracy for distortion levels up to 35% and 80% accuracy level for distortion levels up to 50%. **For non-uniform patterns,** 100% recognition accuracy for distortion levels up to 25% and 80% accuracy level for distortion levels up to 40%.
Dealing with transformed patterns	✓	✓
Dealing with translated patterns	100% recognition accuracy for uniform and non-uniform patterns.	100% recognition accuracy for uniform and non-uniform patterns.
Dealing with scaled patterns	**For uniform patterns**, 100% recognition accuracy for all tested samples. **For non-uniform patterns,** 100% recognition accuracy for scaling levels up to 60% and 80% recognition accuracy for scaling levels up to 80%.	**For uniform patterns**, 100% recognition accuracy for scaling levels up to 16% and 80% recognition accuracy for scaling levels up to 32%. **For non-uniform patterns,** 100% recognition accuracy for scaling levels up to 16% and 80% recognition accuracy for scaling levels up to 30%.

Dealing with rotated patterns	**For uniform patterns,** 100% accuracy level in five rotational regions (Between 0 and 4 degrees, 86 and 94 degrees, 176 and 184 degrees, between 266 and 274 degrees, and 356 and 360 degrees). **For non-uniform patterns,** 100% accuracy level in five rotational regions (Between 0 and 9 degrees, 81 and 94 degrees, 176 and 189 degrees, 261 and 279 degrees, and 351 and 360 degrees).	**For uniform patterns,** 100% accuracy level in five rotational regions (Between 0 and 29 degrees, 61 and 119 degrees, 161 and 199 degrees, 251 and 289 degrees, and 341 and 360 degrees). **For non-uniform patterns,** 100% accuracy level in five rotational regions (Between 0 and 39 degrees, 56 and 119 degrees, 146 and 194 degrees, 256 and 284 degrees, and 346 and 360 degrees).
Number of nodes performing learning cycle	Dynamic (only active nodes): only 25.67% of the network size for the uniform patterns dataset and 0.53% for the non-uniform patterns dataset.	Dynamic (only active nodes): only 25.67% of the network size for the uniform patterns dataset and 0.53% for the non-uniform patterns dataset.
Node utilisation (Energy distribution)	**For uniform patterns dataset**, the energy distribution is balanced and concentrated in the middle layers. **For non-uniform patterns dataset,** the energy distribution is balanced and more scattered in the field.	**For uniform patterns dataset**, the energy distribution is balanced and concentrated in the middle layers. **For non-uniform patterns dataset,** the energy distribution is balanced and more scattered in the field.
Network's lifetime (Utilising 30 mAh battery)	1 year for Async network 6 months for Sync network	9 months for Async network 5 months for Sync network

5.4.1 Common Design Aspects

Both proposed approaches distribute pattern matching by comparing input patterns with a set of reference patterns. The approaches add a clustering mechanism in pattern recognition, by dividing and distributing patterns into sub-patterns. The approaches' networks are designed in a multi-row structure containing multiple rows. The multiple rows in the network structure are intended to enable parallel processing and information exchange of incoming data. This is achieved by allowing each row to perform a set of recognition operations on a sub-pattern in parallel with other rows. Moreover, all pattern data will be stored in the S&I's pattern vector (in the base station). As a result, there is no need to store any pattern data in the sensor nodes, and this will address the nodes' limited memory issue. In both approaches, only the active nodes are used for pattern recognition, in order to conserve the nodes' resources and to reduce the detection time by limiting the number of communicating nodes. Each active node exchanges its own pattern value with its four adjacent nodes, two nodes in the preceding and the succeeding columns in the same row, one being the adjacent node at the next higher row, and one being the adjacent node at the lower row.

5.4.2 Pattern Matching Criteria

The MOHA approach reaches a decision about the input pattern by utilising the location information of the activated edge nodes. After each active node in the network exchanges its value with its adjacent nodes, it determines whether it represents a pattern edge. Then, all activated edge nodes send their location information to the S&I for further analysis and recognition. S&I calculates the MOHA value (MV) of the input pattern and utilises it with the number of activated edge nodes to compare it and match it with the stored patterns. On the other hand, the LMOHA approach reaches a decision about the pattern by using the activated plus, T, corner, and edge nodes. After each active node in the network exchanges its value with its adjacent nodes, it determines whether it represents a pattern sensory-based plus, T, corner, or edge shapes. Then, all activated plus, T, corner, and edge nodes send their types information to the

S&I for further analysis and recognition. S&I counts the number of activated plus nodes, the number of activated T nodes, the number of activated corner nodes, and the number of activated edge nodes and uses this information by comparing them and matching them with the stored patterns.

5.4.3 Handling Distorted and Transformed Patterns

In chapter 4, both approaches showed very significant capabilities to handle distorted and transformed patterns. However, each one of them presented different limits of tolerance to distorted patterns and different types of transformed patterns. Sub-section 4.6.1 showed that MOHA approach can provided 100% recognition accuracy for distortion levels up to 25% and 80% recognition accuracy for distortion levels up to 45% for distorted patterns obtained from a *shapes* dataset that contains uniform patterns. Additionally, sub-section 4.6.2 results showed that MOHA scheme is capable of offering 100% recognition accuracy for distortion levels up to 15% and 80% recognition accuracy for distortion levels up to 35% for distorted patterns obtained from a *stars* dataset that contains non-uniform patterns. On the other hand, sub-section 4.6.1 showed that LMOHA scheme is capable of offering 100% recognition accuracy for distortion levels up to 35% and 80% recognition accuracy for distortion levels up to 50% for a *shapes* dataset. Sub-section 4.6.2 results showed that LMOHA scheme is capable of offering 100% recognition accuracy for distortion levels up to 25% and 80% recognition accuracy for distortion levels up to 40% for a *stars* dataset. In conclusion, the previous results show that the LMOHA scheme provides more accurate recognition for distorted patterns than does the MOHA scheme. The results also show that the recognition accuracy of the LMOHA and MOHA schemes for distorted (noisy) patterns is lower for the *stars* dataset compared to the *shapes* dataset.

In terms of dealing with transformed patterns, both proposed approached showed 100% recognition accuracy when faced with translated patterns. However, each proposed approach showed different capabilities when dealing with scaled and rotated patterns. Sub-section 4.6.1 showed that the MOHA approach can provide 100% recognition accuracy for all tested scaled patterns obtained from *shapes* dataset that contains uniform patterns. Additionally, results presented in sub-section 4.6.2 showed that MOHA scheme is capable of offering perfect recognition accuracy (100% accuracy level) for scaling levels up to 60% for a *stars* dataset that contains non-uniform patterns. The sub-section results also showed that the MOHA is capable of offering 80% recognition accuracy level for scaling levels up to 80% for the same dataset. On the other hand, sub-section 4.6.1 showed that the LMOHA scheme is capable of offering perfect recognition accuracy (100% accuracy level) for scaling levels up to 16% for *shapes* dataset. The section also showed that the LMOHA is capable of offering 80% recognition accuracy level for scaling levels up to 32% for the uniform patterns. Moreover, sub-section 4.6.2 results showed that the LMOHA scheme is capable of offering perfect recognition accuracy (100% accuracy level) for scaling levels up to 16% for *stars* dataset. The sub-section results also showed that the LMOHA is capable of offering an 80% recognition accuracy level for scaling levels up to 30% for the non-uniform patterns. To sum up, the previous results clearly show that the MOHA scheme provides more accurate recognition for scaled patterns than does the LMOHA scheme.

Sub-section 4.6.1 showed that the MOHA approach provided high accuracy percentages (100% accuracy level) in five rotational regions for *shapes* dataset that contains uniform patterns. The first region is between 0 and 4 degrees, the second region is between 86 and 94 degrees, the third region is between 176 and 184 degrees, the region where patterns are horizontally flipped or nearly flipped, the forth region is between 266 and 274 degrees, and the fifth region is between 356 and 360 degrees. Moreover, sub-section 4.6.2 showed that MOHA approach provided high accuracy percentages in five rotational regions for *stars* dataset that contains non-uniform patterns. The first region is between 0 and 9 degrees, the second region is between 81 and 94 degrees, the third region is between 176 and 189 degrees, the forth region is between 261 and 279 degrees, and the fifth region is between 351 and 360 degrees. On the other hand, sub-section 4.6.1 showed that LMOHA approach provided high accuracy percentages in five rotational regions for *shapes* dataset. The first area is between 0 and 29 degrees, the second area is between 61 and 119 degrees, the third area is between 161 and 199 degrees, the forth area is between 251 and 289 degrees, and the fifth area is between 341 and 360 degrees. Sub-section 4.6.2 showed that the LMOHA approach provided high accuracy percentages in five rotational regions for *stars* dataset that contains non-uniform patterns. The first area is between 0 and 39 degrees, the second area is between 56 and 119 degrees, the third area is between 146 and 194 degrees, the forth area is between 256 and 284 degrees, and the fifth area is between 346 and 360 degrees. In conclusion, the previous results show that the recognition accuracy of proposed schemes for rotated patterns is higher for the *stars* dataset compared to the *shapes* dataset. They also show that the LMOHA scheme provides more accurate recognition for rotated patterns than the MOHA scheme.

In order to evaluate the capabilities of the proposed schemes in handling the transformed patterns, the next chapter will provide a comparison between the proposed schemes and other well-known pattern recognition schemes utilising transformed uniform and non-uniform patterns.

5.4.4 Number of Nodes Performing Recognition

In both proposed approaches, only the activated nodes in the network participate in the learning process. This is to conserve the nodes' resources and to reduce the detection time by limiting the number of communicating nodes. Results given in section 5.3.1 showed that, on average, the total number of activated nodes represents only 25.67% of the total number of nodes (the network size) for the *shapes* dataset (uniform patterns) and 0.53% for the *stars* dataset (non-uniform patterns). Additionally, the results presented in this section showed that in the MOHA implementation, the number of activated edge nodes that will participate in report communications represents only 0.74% of the network size for the *shapes* dataset and 0.08% for the *stars* dataset. On the other hand, in the LMOHA implementation, the total number of activated plus, T, corner, and edge nodes that will participate in report communications represents only 25.67% of the network size for the *shapes* dataset and 0.30% for the *stars* dataset. This reflects the amount of communicational overhead reduction that can be achieved by using the proposed approaches and shows that the number of activated nodes that will participate in report communications is lower in the MOHA approach compared to the LMOHA approach.

5.4.5 Node Utilisation

The previous sub-section (sub-section 5.3.2) shows that node utilisation and energy consumption and distribution of MOHA and LMOHA are almost identical. This is due to the network structure and the fact that the node's exchange communications process consumes the most energy, which is identical for both proposed schemes. Sub-section 5.3.2 shows that, in both proposed scheme, both Async and Sync models have similar energy distribution but with more consumption in the Sync model. Results given in the sub-section also show that the energy distribution in the *stars* dataset (non-uniform patterns) is more scattered in the field, whereas it is concentrated in the middle layers for the *shapes* dataset (uniform patterns). This is caused mainly by the non-uniform pattern distribution of the *stars* dataset compared with the *shapes* dataset.

5.4.6 Network's Lifetime

Sub-section 5.3.2 shows that a MOHA Async network can theoretically last for almost one year if it is used to obtain one pattern per minute using a small 3V 30mAh battery. Under the same conditions, LMOHA Async network can last for almost nine months. The sub-section also shows that a MOHA Sync network could last for almost six months and a LMOHA Sync network could last for almost five months. On the other hand, a parallel KNN will last for less than three months under similar conditions. These reflect the amount of communicational overhead reduction that can be achieved by using the proposed approaches and shows that the MOHA network has an average of 2 months more lifetime than LMOHA network. In both approaches, the use of the Async model increases the network’s lifetime compared with the Sync model.

5.4.7 MOHA and LMOHA Scenarios

In conclusion, both proposed approaches showed very significant capabilities when performing pattern recognition in WSNs as they showed very good capabilities to handle distorted and transformed patterns and limiting the number of communications, thereby limiting the use of energy resources. Moreover, the MOHA approach requires less communication (fewer activated nodes participate in reporting communications) and hence is able to conserve more energy compared with the LMOHA approach. The MOHA approach also offers more accurate recognition for scaled patterns than does LMOHA scheme. As a result, the MOHA approach is recommended for WSN applications that require huge reductions in terms of communications and energy consumption and deal with scaled patterns that are scaled more than 16%. On the other hand, the LMOHA approach provides more accurate recognition for distorted and rotated patterns than does the MOHA scheme. The LMOHA approach also significantly reduces the processing power required by the S&I to analyse and finally match the input pattern compared with the MOHA approach. As a result, the LMOHA approach is recommended for WSN applications that require the S&I to be implemented in a typical sensor node (with limited processing power) and deal mainly with distorted and rotated patterns.

In order to evaluate the capabilities of the proposed schemes in performing pattern recognition in real scenarios, the next chapter will provide a comparison between the proposed schemes and other well-known pattern recognition schemes utilising patterns obtained from real datasets.

5.5 Conclusions

This chapter has presented different types of message sequence models for the MOHA and LMOHA schemes based on different types of MAC protocols, namely, frame-slotted synchronous (FS-Sync), frame-slotted asynchronous

(FS-Async), Multi-channel synchronous (MC-Sync), and Multi-channel asynchronous (MC-Async). It analyses the MOHA and LMOHA schemes from a practical perspective and estimates the lifetime and execution duration of the networks utilising the communicational models (MAC protocols) presented at the beginning of the chapter.

This analysis begins by determining the effects of the nodes activation technique on communications in the network. In both proposed approaches, only the activated nodes in the network participate in the learning process. This is done to limit the consumption of the nodes' resources and to reduce the detection time by limiting the number of communicating nodes. Results presented in sub-section 5.3.1 showed that on average the total number of activated nodes represents only 25.67% of the total number of nodes (the network size) for the *shapes* dataset (uniform patterns) and 0.53% for the *stars* dataset (non-uniform patterns). Additionally, the results presented in the sub-section showed that in the MOHA implementation, the number of activated edge nodes that will participate in report communications represents only 0.74% of the network size for the *shapes* dataset and 0.08% for the *stars* dataset. On the other hand, in the LMOHA implementation, the total number of activated plus, T, corner, and edge nodes that will participate in report communications represents only 25.67% of the network size for the *shapes* dataset and 0.30% for the *stars* dataset. This reflects the amount of reduction in communicational overhead that can be achieved by using the proposed approaches, and shows that the number of activated nodes that will participate in report communications is lower in the MOHA approach compared with the LMOHA approach.

Moreover, this chapter presented a simulation analysis of the proposed schemes' network communications, including energy and time analysis. The energy analysis showed that both proposed schemes are capable of limiting the number of communications, thereby limiting the consumption of energy resources. It was shown that the network can have a lifetime of one year with the MOHA scheme and nine months with LMOHA scheme using one of the smallest batteries in terms of capacity (30 mAh). These are four and three times higher than other schemes, such as parallel KNN. The time analysis shows that a MOHA network can scale up, while having the ability to converge within a time range between 5.17 ms and 2231.39 ms or a sample rate between a pattern per 2 seconds and 193 patterns per second. On the other hand, a LMOHA network can scale up, while having the ability to converge within a time range between 5.17 ms and 16441.33 ms or a sample rate between a pattern per 16 seconds and 193 patterns per second. These results show that both proposed schemes minimise the communicational overhead, enabling WSNs to scale up efficiently and provide real-time learning capabilities. These results also show a huge difference between the time requirements of the frame-slotted models and those of the multi-channel models. In both proposed schemes, the MC model will speed up the exchanging and reporting times by removing the effects of the number of activated nodes in the network and the size of the network in the learning cycle time. This is because the MC model is able to perform its communications in parallel.

Finally, this chapter presented an overall comparison between both versions of MOHA algorithms. This comparison shows that both proposed approaches showed very significant capabilities when performing pattern recognition in WSNs particularly in handling the distorted and transformed patterns and limiting the number of communications, thereby limiting the use of energy resources. Moreover, the MOHA approach requires less communication (since fewer activated nodes participate in reporting communications) and hence it is able to conserve more energy compared with the LMOHA approach. The MOHA also offers more accurate recognition for scaled patterns than LMOHA scheme. As a result, the MOHA approach is recommended for WSN applications that require huge reductions in terms of communications and energy and deal with scaled patterns that are scaled more than 16%. On the other hand, the LMOHA approach provides more accurate recognition for distorted and rotated patterns than does the MOHA scheme. LMOHA approach also significantly reduces the processing power required by the S&I to analyse and finally match the input pattern compared to MOHA approach. As a result, the LMOHA approach is recommended for WSN applications that require the S&I to be implemented in typical sensor node (with limited processing power) and mainly deal with distorted and rotated patterns.

In the next chapter, a series of simulations for the MOHA and LMOHA schemes will be conducted. Furthermore, a comparison between the two proposed schemes and other well-known pattern recognition schemes will be provided. Chapter 6 also will describe a series of tests used to determine whether the capabilities of both the proposed schemes will enable them to be utilised in real-life scenarios.

Chapter 6

Evaluating the Performance of MOHA and LMOHA Schemes

6.1 Preamble

Both the MOHA and LMOHA schemes were proposed as lightweight and distributed pattern recognition schemes that involve limited numbers of communications and computations. Experiments have shown that the proposed schemes are significantly capable of dealing with noisy and transformed patterns. However, each one of them presented different limits of tolerance to noisy patterns and different types of transformed patterns. The proposed schemes operate with low and stable recognition time, compared with other pattern recognition algorithms. In both proposed schemes, only the activated nodes in the network participate in the learning process. This is in order to conserve the nodes' resources and to reduce the detection time by limiting the number of communicating nodes.

The energy analysis showed that both proposed schemes are capable of limiting the number of communications, thereby limiting the use of energy resources. It was shown that the network can have a lifetime of one year with the MOHA scheme and nine months with the LMOHA scheme using one of the smallest batteries in terms of capacity (30 mAh). These are four and three times higher than other schemes such as parallel KNN. The time analysis results show that both proposed schemes minimise communicational overhead, enabling WSNs to scale up efficiently and provide real-time learning capabilities.

In this chapter, a series of simulations for both proposed schemes will be conducted. Moreover, the proposed schemes will be compared with other well-known pattern recognition schemes. Three tests will be presented and discussed in this chapter. The first test compares the accuracy of the proposed schemes with well-known existing schemes in dealing with transformation patterns. Both the *shapes* and the *stars* datasets will be utilised in this test, which consist of uniform and non-uniform patterns, respectively. The second test is used to demonstrate the ability of the MOHA and LMOHA schemes to deal with complex and real-life problems. This test will be on handwritten character recognition. This test will show that both proposed schemes are capable of performing a recognition operation using a minimal number of training samples while still maintaining a high level of accuracy compared with other schemes. In accomplishing this goal, the MOHA and LMOHA networks are capable of addressing the problem of WSN randomness discussed in Chapter 2. The third and final test is intended to prove whether the proposed schemes can be applied to real classification problems. A well-known application that can be utilised for this purpose is human activity recognition. In this test, the proposed schemes will be compared with other existing techniques using a limited amount of collected data to demonstrate the capability of the scheme in addressing activity recognition problems using two different types of datasets, namely, vision-based and sensor-based.

The objectives of this chapter are as follows:

1. To compare the accuracy of the proposed schemes with well-known existing schemes in dealing with transformation patterns using uniform and non-uniform patterns.
2. To ascertain the capability of the proposed schemes in dealing with complex and real-life problems.
3. To ascertain the capability of the proposed schemes in performing a recognition operation using a minimal number of training samples while still maintaining a high level of accuracy compared with other schemes.
4. To check the possibility of using the proposed schemes for real classification problems and different application domains.

The remainder of this chapter is organised as follows: Section 6.2 provides a comparison between MOHA and LMOHA schemes and other existing schemes in dealing with transformation patterns using the *shapes* and *stars* datasets. Section 6.3 presents the handwritten character recognition test. This test will show that both proposed schemes are capable of performing a recognition operation using a minimal number of training samples while still maintaining a high level of accuracy compared with other schemes. Section 6.4 discusses the possibility of using the proposed schemes for real classification problems. Human activity recognition systems will be presented and discussed in the section. The vision-based human activity recognition will be discussed in sub-section 6.4.1 and the sensor-based human activity recognition will be discussed in sub-section 6.4.2. Finally, section 6.5 provides an overall discussion about the chapter.

6.2 Comparing LMOHA and MOHA Accuracy with Other Schemes in Dealing with Transformation Patterns

This test was used to compare the accuracy of the LMOHA and MOHA schemes with well-known existing schemes in dealing with transformation patterns using the *shapes* and the *stars* datasets [261, 270], which consist of uniform and non-uniform patterns, respectively. The training dataset was constructed by creating and utilising five shapes (from the *shapes* dataset) and five star map images (from the *stars* datasets) modelled as a binary image of size 200-by-200 pixels, as shown in Figure 6.1. To construct the testing dataset, we generated 180 rotated samples (rotating from 1 to 360 degrees with two degrees for each rotation level), 200 randomly translated samples, and 100 scaled samples (from 1% to 100% in 1% steps) for each training shape and star map as three testing datasets. A total of 4800 test samples were generated for recall. In this test, we ran simulated LMOHA and MOHA networks of 40000 nodes assuming that nodes are distributed as a grid and the each node detects one pixel reading. We also assume that each LMOHA and MOHA network is divided into 200 rows and each one of these rows is utilised to handle 200-bit binary sub-patterns; the threshold values for node activation and edge determination are set to 1 ($\varphi = 1$). In order to compare the proposed schemes with other schemes, the Weka tool [263, 264] was utilised to construct KNN (*k*=1), Naive Bayes, and SVM to memorise and recall patterns. For this test, Naive Bayes and SVM have been chosen because of their ability to recognise transformations in patterns as discussed in section 2.5. Moreover, KNN has been chosen because it is one of well-known classification methods that have been widely used in many application domains. However, KNN generally experiences high computational complexity due to calculating *k* number of distances. In this test, the *k* value was set to 1 (*k*=1). The main reason for this choice is that we want to compare the proposed schemes with the lowest version of the KNN scheme in terms of computational complexity. Having a lightweight scheme in terms of computational complexity is one of the thesis requirements. As discussed in sub-section 2.3.3, the KNN computational complexity increases for large *k* values [265]. On the other hand, setting *k* value to 1 might not provide the optimal performance of the scheme in terms of accuracy. The accuracy of the schemes is calculated as the total number of correctly recalled patterns as a percentage of the number of altered tested samples. The higher the percentage means the higher the accuracy.

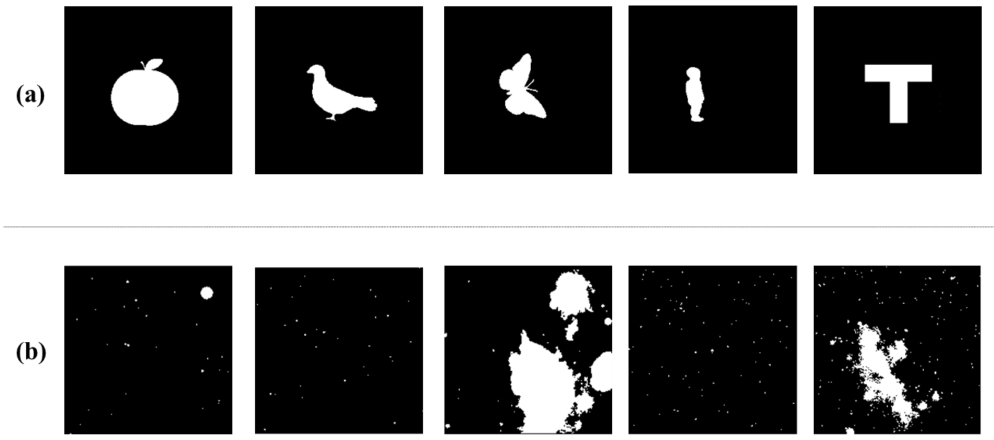

Figure 6.1: The training dataset: (a) Five shapes, (b) Five star maps.

Table 6.1: Recognition accuracy results of different schemes for the *shapes* and the *stars* datasets.

Scheme	Rotated Patterns		Translated Patterns		Scaled Patterns		Overall		
	Shapes Dataset	*Stars Dataset*	*Shapes Dataset*	*Stars Dataset*	*Shapes Dataset*	*Stars Dataset*	*Shapes Dataset*	*Stars Dataset*	**Both**
MOHA	67.78%	59.39%	100%	100%	98.20%	88.30%	88.66%	82.56%	85.61%
LMOHA	82.22%	84.50%	100%	100%	53.70%	59.65%	78.64%	81.38%	80.01%
Naive Bayes	41.11%	43.89%	30.65%	29.85%	60.90%	58.90%	44.22%	44.21%	44.22%
KNN (*k*=1)	47.78%	49.44%	33.40%	31.65%	85.60%	77.90%	55.60%	53%	54.30%
SVM	47.28%	50.56	32.45%	30.70%	87.10%	86.60%	55.61%	55.95%	55.78%

Table 6.1 shows the percentage of accurate results obtained by the MOHA, LMOHA, and other schemes. The table clearly shows that the MOHA and LMOHA schemes have a very high accuracy when dealing with transformed patterns compared to other schemes. However, this does not necessarily mean that the proposed schemes are better than the other schemes, as this test was intended to determine the accuracy of the proposed schemes in this particular case only. As discussed in sub-section 3.8.2.1, the main reasons

for the results of other schemes are the small numbers of training instances (only one trained sample for each image) and the capability of each scheme to deal with these types of transformations. When the Naive Bayes scheme is used to perform pattern recognition, it attempts to build a probabilistic relationship between the training samples and input variables. Since the number of training samples is very limited, the created probabilistic relationships cannot efficiently describe each pattern [96, 103]. The SVM scheme also requires a large number of training samples in order to create separating hyperplanes between classes and correctly classify patterns, as discussed in section 2.5. The KNN's inability to deal with transformed patterns was discussed in Chapter 2. Table 6.1 also shows that the MOHA scheme offers higher recognition accuracy than does the LMOHA scheme. The main reason for this is that the MOHA scheme has an excellent ability to handle scaled patterns. On the other hand, the LMOHA scheme offers better recognition accuracy than does the MOHA scheme for rotated patterns. Figure 6.2 shows the receiver operating characteristic (ROC) space and the plots for the two tested datasets (the *shapes* and the *stars* datasets) for the MOHA, LMOHA, Naive Bayes, KNN, and SVM schemes. A ROC graph describes the performance of each network based on the false positive rate (FPR) and true positive rate (TPR). The figure shows that, in dealing with transformation patterns, the MOHA and LMOHA schemes have higher detection accuracy and lower error rates compared with the other schemes.

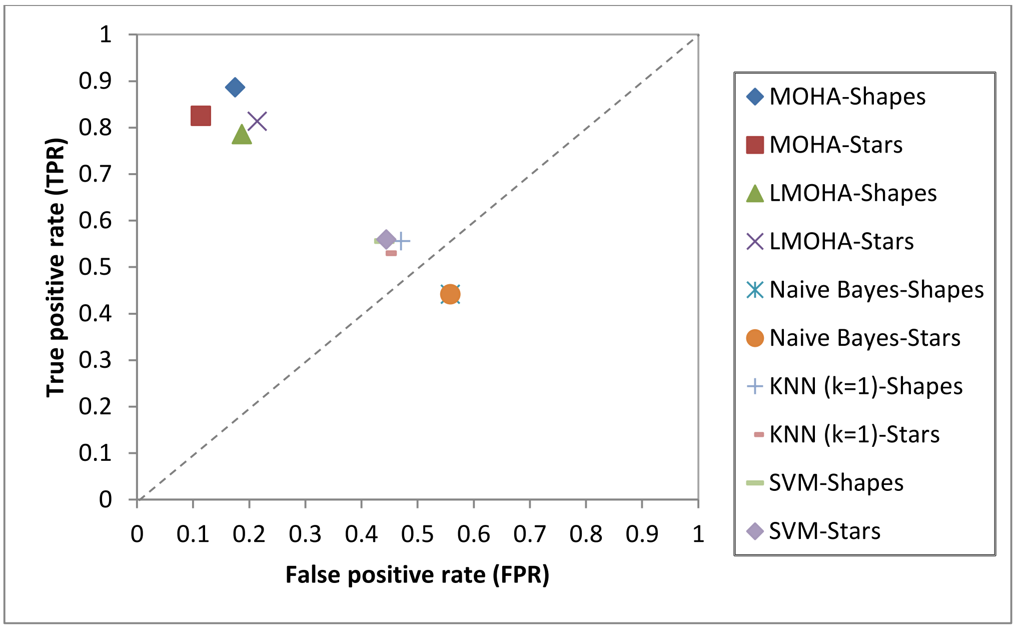

Figure 6.2: The ROC space and the plots for the two tested datasets for MOHA, LMOHA, Naive Bayes, KNN, and SVM schemes.

6.3 Handwritten Character Recognition Test

This test was used to demonstrate the ability of the MOHA and LMOHA schemes to deal with complex and real-life problems. It also aimed to compare the MOHA's and LMOHA's accuracy with those of existing pattern recognition schemes. For this purpose, we chose a handwritten character recognition problem as such problems require the memorisation of a great amount of training information. This test showed that both MOHA and LMOHA schemes are capable of performing a recognition operation using a minimal amount of training information (i.e. one sample for each class) while still maintaining a high level of accuracy compared with other schemes. In accomplishing this aim, MOHA and LMOHA networks are capable of addressing the problem of WSN randomness discussed in Chapter 2. The test used the dataset provided by [276, 277]. The dataset contains 1593 handwritten patterns for numbers from 0 to 9. Each number was represented as a 16×16 binary pattern. Each pattern was produced by scanning and pre-processing numbers handwritten by 80 different people. The dataset has 10 classes, each class representing one number.

To construct MOHA and LMOHA networks that are capable of adopting such patterns, 256 nodes distributed among 16 rows were generated. The threshold values for node activation and edge determination were set to 1 $(\varphi = 1)$. The selected classes for comparison were classes representing numbers from 0 to 5 as these numbers were distinguished in the handwritten representation. One pattern was selected randomly from each class for memorisation. This resulted in six memorised patterns to represent numbers from 0 to 5. The rest of the patterns in each class were used for recognition. This resulted in 955 recognition patterns. Figure 6.3 shows the six memorisation patterns and a sample of six recognition patterns.

The proposed schemes are compared with Naive Bayes and back propagation neural networks. Using a Weka tool [263, 264], Naive Bayes and back propagation neural networks were generated to perform storing and recognition operations. These schemes have been chosen because they are two of the best-known classification schemes for their ability to recognise transformations in patterns as discussed in section 2.5. This ability is very helpful for this type of dataset that consists of handwritten patterns for numbers, which usually vary in size, location and rotation angels. Figure 6.4 shows the average recognition accuracy of the proposed schemes compared with the Naive Bayes and back propagation NN schemes. The percentage of accuracy is calculated as the number of correctly recognised patterns to the total number of recognition patterns. In order for a comparison to be made with back propagation NN (BP), three BP implementations were used, generating a BP with a single hidden layer, generating a BP with two hidden layers, and generating a BP with three hidden layers. The number of learning iterations of each structure was set to ranges between 1 (single cycle) and 500 iterations. The best result was obtained with the implementation of BP with three hidden layers and 200 iterations. The results shown in Figure 6.4 for the BP NN are based on that structure.

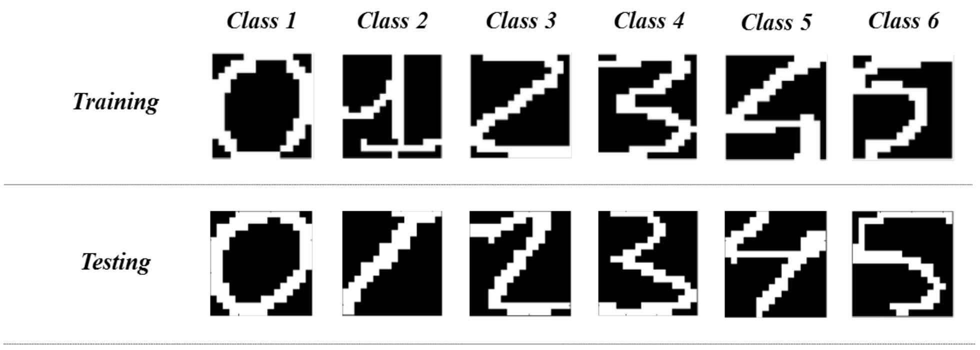

Figure 6.3: The training patterns and samples of testing patterns for the handwritten character recognition test.

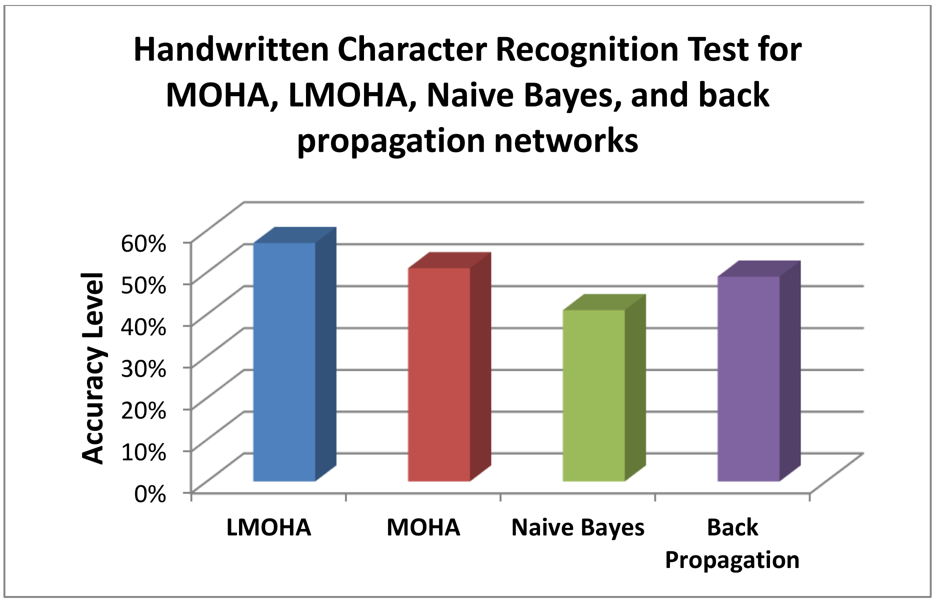

Figure 6.4: Average accuracy levels obtained by MOHA, LMOHA, Naive Bayes, and back propagation networks.

Figure 6.4 shows that MOHA and LMOHA can provide pattern recognition capabilities with better accuracy results than Naive Bayes and BP NN schemes. These results were achieved by using only one pattern for each class for memorisation. Similar to the previous test, the main reason that the Naive Bayes and BP NN schemes have lower recognition accuracies compared with the proposed schemes is their small numbers of training instances. When the Naive Bayes scheme is used to perform pattern recognition, it attempts to build a probabilistic relationship between the training samples and input variables. Since the number of training samples is very limited, the created probabilistic relationships cannot efficiently describe each pattern [96, 103]. As discussed in section 2.5, BP NN requires a large number of training samples in order to correctly classify incoming patterns. Additionally, the MOHA and

LMOHA schemes are capable of performing pattern recognition operations in a single cycle with the number of neurons (i.e. nodes) equal to the pattern size. The results also show that the LMOHA offers higher recognition accuracy when dealing with handwritten character recognition problem than does the MOHA. The main reason for this is the LMOHA scheme's excellent ability to handle rotated patterns that are the most common issues in handwritten character recognition. Compared to the back propagation setting for this example, the proposed schemes reduced the number of participating nodes, communications and iterations needed to perform pattern recognition while maintaining higher accuracy levels. This is because the back propagation networks involve more neurons to build the multi-layer structure, requiring tightly coupled connectivity between layers, and requiring 200 iterations to perform recognition operations. This test demonstrates the capability of the proposed schemes' networks to perform complex and real-life recognition problems by using a minimal amount of training information. This addresses the problem of the randomness associated with WSNs.

6.4 Human Activity Recognition Case Studies

This section discusses the feasibility of utilising the MOHA and LMOHA schemes for real classification problems and different application domains. MOHA and LMOHA models will be introduced that translate attributes into patterns and then utilise these patterns to address the classification problems. This section will show that such models are capable of performing pattern recognition and classification using a limited number of training instances with a high level of accuracy compared with some of the current well-known classification methods such as KNN (*k*=1), Naive Bayes, and Multi-layer neural networks. In this test, the *k* value for KNN was set to 1 (*k*=1). The main reason for this choice is that we want to compare the proposed schemes with the lowest version of the KNN scheme in terms of computational complexity. Having a lightweight scheme in terms of computational complexity is one of the aims of this thesis.

One of the well-known applications that can be utilised for this purpose is the human activity recognition system. In the last decade, human activity recognition has become one of the most significant emerging fields of research within context-aware systems [278]. Physical activity can be defined as "any bodily movement produced by skeletal muscles result in energy expenditure above resting level" [279]. The aim of an activity recognition system is to analyse the physical behaviours of individuals in order to interpret the actual state, action or activity a human is performing at any given time [278, 280]. Human activity recognition systems are useful in numerous types of applications such as habitat monitoring, medical monitoring, security, sports, and so on. Activity recognition systems are usually categorised in terms of the type of sensor that is used for activity monitoring: vision-based and sensor-based activity recognition [280, 281]. The first uses visual sensing facilities, such as video cameras, to monitor an actor's behaviour and environmental changes [282, 283]. The generated sensor data are video sequences or digitised visual data. The approaches in this category exploit computer vision techniques, including feature extraction, structural modelling, movement segmentation, action extraction, and movement tracking to analyse visual observations for pattern recognition [284, 285]. The second category uses emerging sensor network technologies for activity monitoring. The generated sensor data from sensor-based monitoring are mainly time series of state changes and/or various parameter values that are usually processed through data fusion, probabilistic, or statistical analysis methods and formal knowledge technologies for activity recognition. In these approaches, sensors can be attached to an actor under observation - namely wearable sensors or smart phones, or objects that constitute the activity environment - namely dense sensing. Wearable sensors often utilise inertial measurement units and radio frequency identification (RFID) tags to gather information about an actor's behaviour [286]. This approach is effective in recognising physical movements such as physical exercises [287]. In contrast, dense sensing infers activities by monitoring human-object interactions through the usage of multiple multimodal miniaturised sensors [280, 281, 288]. The main focus of this section is on vision-based and sensor-based activity recognition systems as case studies for pattern recognition using the MOHA and LMOHA schemes.

Many researches in the literature have attempted to address the issue of human activity recognition. For instance, Christian et al. [289] demonstrated that human action recognition can be achieved by combining local features with an SVM classifier. Zhu and Weihua [290] utilise both Markov models and neural networks to analyse wearable sensor readings in order to produce a human-robot interaction approach for elderly and disabled people. Parakka et al. [291] utilise PDAs and sensor devices to perform online daily life activity recognition based on a decision tree classifier. Zhang and Sawchuk [292] present a sparse representation approach which leads to reduce human activity recognition complexity utilising wearable sensor devices. Generally, classification methods like neural networks, KNN, SVM, and Naive Bayes are commonly used in addressing activity recognition problems in WSNs [280, 291]. However, no single classification method has proven to be superior to others in dealing with this issue [291]. These methods could utilise probabilistic approaches such as Naive Bayesian networks. However, such methods require huge amounts of data to be available and a great deal of analysis is required. Alternatively, other classification methods such as nearest neighbour can be utilised to create a model of relationships between collected data.

The issue of human activity recognition using sensor networks encounters several challenges that limit the capabilities of such systems. These challenges are generally associated with human behaviour or technical issues [278]. Human behaviour is one of the most important challenges to be addressed. Several difficulties related to human behaviour emerge in conducting activities. For instance, a person may perform multiple actions or activities at the same time. Here, research needs to address the question of the means of distinguishing between one activity and another. Additionally, the behaviour of targeted monitored objects differs from person to person. For instance, a person may perform a set of sequenced actions in order to complete an activity while another person may carry out a different sequence when performing the same activity. Other challenges are related to technical issues which include those pertaining to the design of sensor networks and the way these sensors are physically deployed. Activity recognition systems regularly require small wearable devices that are attached to a target or mounted on surrounding objects. The main challenge in this case is to conserve battery consumption and reduce memory requirements [278].

It has been shown in previous chapters that the MOHA and LMOHA schemes reduce the number of communications required for performing pattern recognition. Additionally, it has been shown that the proposed schemes perform learning operations with no memory requirements placed on the sensor nodes. Therefore, the MOHA and LMOHA schemes are good candidates for dealing with the issue of activity recognition as they deal with the major technical challenges that may be encountered in such applications. In this section, the MOHA and LMOHA schemes will be compared with other existing techniques using limited collected data to demonstrate the capability of the scheme in addressing activity recognition problems using vision-based and sensor-based datasets.

6.4.1 Vision-based human activity recognition

This sub-section will ascertain the ability of the MOHA and LMOHA algorithms to perform human activity recognition using a dataset recorded with video cameras. Firstly, a detailed description of the vision-based dataset will be presented. Then, the MOHA and LMOHA vision-based human activity recognition systems will be proposed and the simulation results will be discussed. The results obtained from the proposed schemes will be compared with those obtained by using three well-known classification methods, namely: KNN, Naive Bayes, and Multi-layer NN.

6.4.1.1 Action dataset

An action dataset [289] is a video database containing six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed four times (sequences) performed by 25 subjects (people) in four different scenarios: outdoors s1, outdoors with scale variation s2, outdoors with different clothes s3 and indoors s4 (see Figure 6.5). In total, the database contains 2391 sequences. Each action is repeated four times by each subject, and for the cases of “walking”, “jogging” and “running”, there are two sequences where the subject is moving leftwards and two sequences with the subject moving rightwards. All sequences were taken over homogeneous backgrounds with a static camera with 25fps frame rate. The sequences were downsampled to the spatial resolution of 160×120 pixels and have a length of four seconds on average.

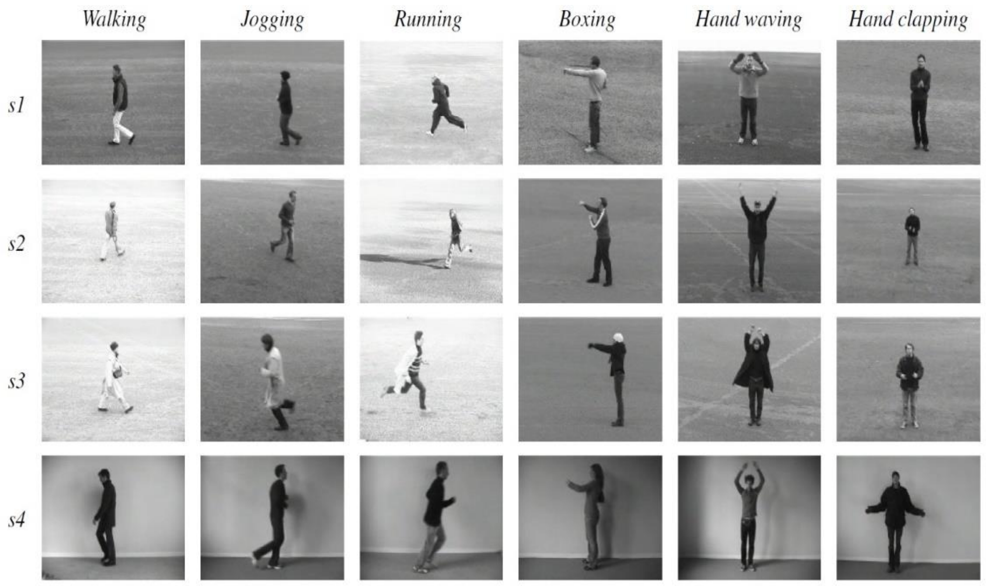

Figure 6.5: Action dataset [289].

This case study attempted to classify the six human actions based on the recorded actions. Every action is repeated four times by each person and is been labelled in terms *start-frame* and *end-frame*.

6.4.1.2 MOHA and LMOHA schemes for vision-based activity recognition

An action dataset was used in this case study to solve the issue of human activity recognition using the MOHA and LMOHA schemes. The first step in this case study was handling the dataset in order to present valid readings to the network. The dataset provided a video recording as a series of images based on action sequences in terms of *start-frame* (starting of the action) and *end-frame* (ending the action). Each data instance represents one recorded frame (image) of the performed action. The spatial resolution of each frame or image is

160×120 pixels. Thus, in this case study, the proposed schemes' networks are made up of 19200 nodes in order to handle the 160×120 pixels of the recorded images. We assume that nodes are distributed as a grid and that each node detects one pixel reading. We also assume that each LMOHA and MOHA network is divided into 160 rows and each one of these rows is utilised to handle 120 sub-patterns. In this case study, the threshold values for node activation and edge determination are set to 70 ($\varphi = 70$). Figure 6.6 shows the transformation of an image into the corresponding MOHA's activated edge nodes and LMOHA's activated plus, T, corner, and edge nodes.

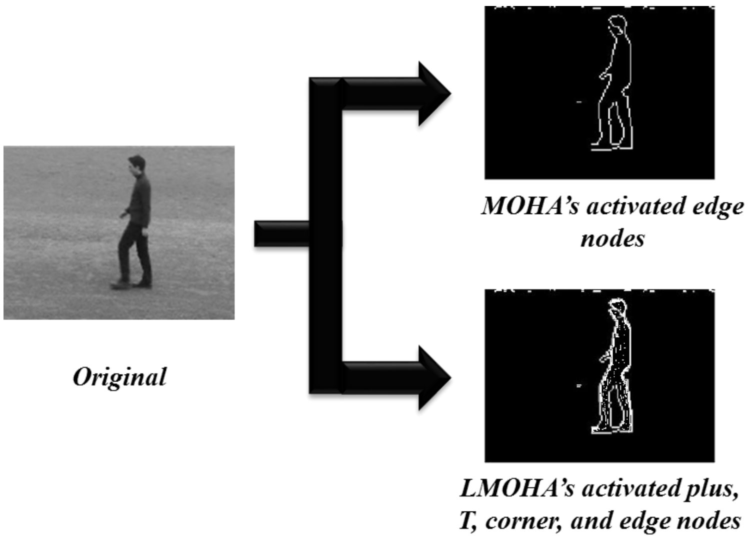
Original
MOHA's activated edge nodes
LMOHA's activated plus, T, corner, and edge nodes

Figure 6.6: An example of image transformation into the MOHA's activated edge nodes and the LMOHA's activated plus, T, corner, and edge nodes.

The second step was to address the action's time in activity recognition. MOHA and LMOHA are capable of detecting the action for a specific time. However, when it comes to activity recognition systems, then the system should be able to recognise the activity which takes a period of time $\{t_1, t_2, \ldots, t_n\}$. Therefore, in order to give this capability to the proposed schemes, we add the capabilities to the S&I (in the base station) to perform majority voting technique [293]. In this case study, majority voting is used as a means of obtaining a combined decision on the recalls made by the proposed schemes within specific time period t_n. As a result, the stored class (i.e. action) that received the highest recall over a specific time period t_n will be recognised as the performed action.

The third step was to ascertain the relationship between sensors and the S&I in the MOHA and LMOHA networks. In this case study, firstly, the S&I (in the base station) received the 160×120 pixels frame or image which represent an action in specific time. Then, it assigns the image to the network. After that, the S&I received the recognition results from the network over a period of time. In this case study, the recognition period of time was set to 50 frames or images which are almost the average time needed to perform one action sequence in this dataset. Finally, the S&I performs the majority voting in order to recognise the action that received most recalls. The proposed schemes were trained using one action sequence for each type of action (class). These are 75 data instances for walking, 50 data instances for jogging, 35 data instances for running, 95 data instances for boxing, 102 data instances for hand waving, and 112 data instances for hand clapping. To compare the proposed schemes with other schemes, the Weka tool [263, 264] was utilised to simulate three different schemes: KNN (*k*=1), Naive Bayes, and Multi-layer NN. Figure

6.7 shows the receiver operating characteristic (ROC) space and the plots for the six activity classes for the MOHA, LMOHA, KNN (*k*=1), Naive Bayes, and Multi-layer NN schemes. A ROC graph describes the performance of each scheme based on the false positive rate (FPR) and true positive rate (TPR). Table 6.2 shows the details of the MOHA, LMOHA and other schemes' recognition accuracy levels obtained for each action class. Accuracy in the table is calculated as the total number of correctly classified instances (patterns) to the total number of instances in that class. Overall accuracy is calculated as the total number of correctly classified instances compared to the total number of testing instances. The same training and testing datasets were used for all schemes.

Table 6.2: Recognition accuracy results of action dataset for different schemes.

Class	Walking	Jogging	Running	Boxing	Hand clapping	Hand waving	**Overall**
Training instances	75	50	35	95	112	102	469
Testing instances	68625	40175	39500	39625	43650	51750	283325
MOHA	78.93%	70.68%	63.41%	67.72%	64.34%	59.26%	67.39%
LMOHA	77.19%	73.09%	65.34%	68.92%	63.72%	61.12%	68.23%
Naive Bayes	89.49%	62.33%	42.77%	48.02%	40.61%	49.82%	55.51%
KNN (*k*=1)	53.43%	41.18%	46.59%	74.51%	41.79%	41.06%	49.76%
Multi-layer NN	99.31%	55.65%	48.34%	42.65%	40.20%	40.28%	54.41%

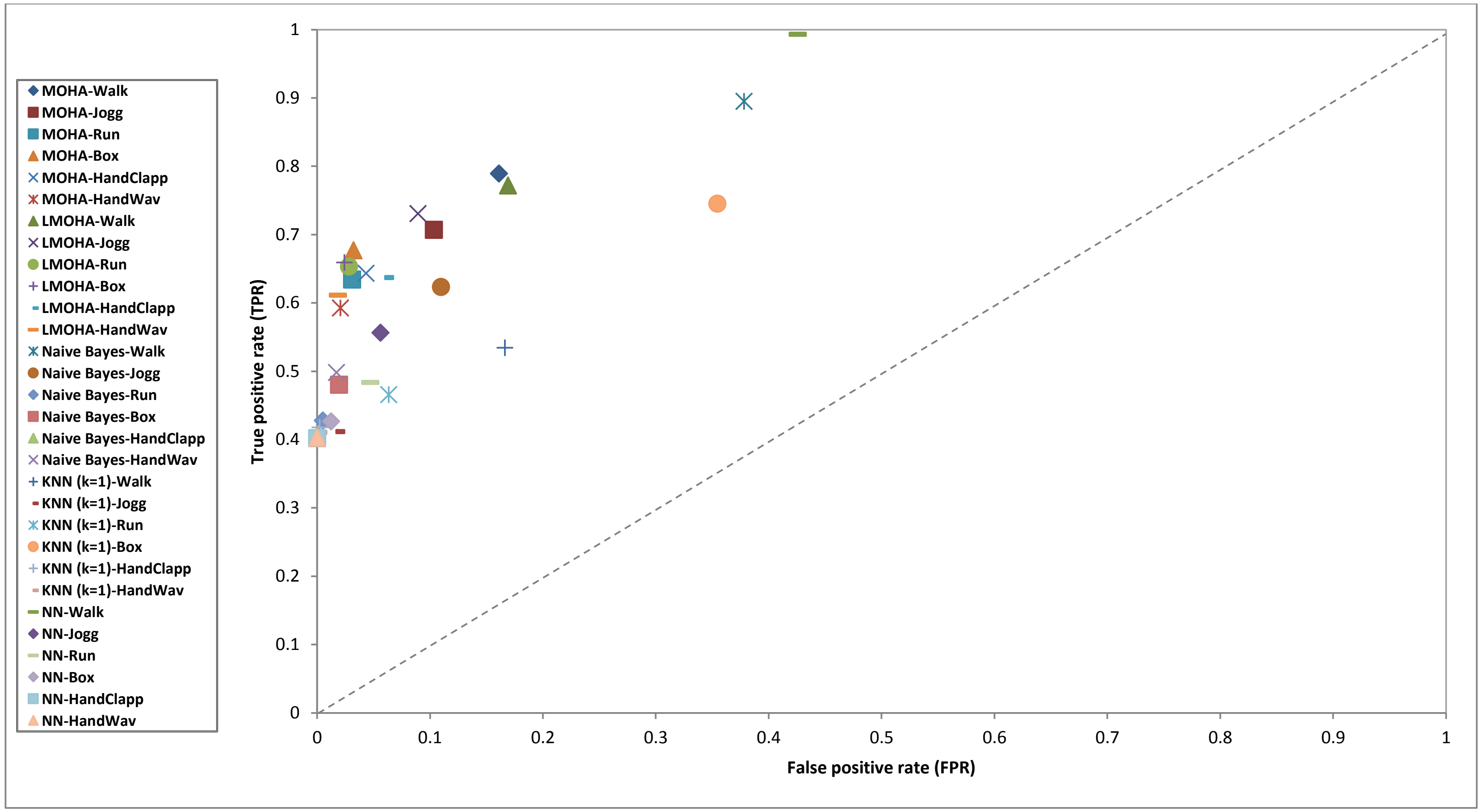

Figure 6.7: The ROC space and the plots for the action dataset for MOHA, LMOHA, Naive Bayes, KNN, and Multi-layer NN schemes.

Table 6.2 clearly shows that the MOHA and LMOHA schemes have higher recognition accuracy compared with that of other schemes. The table also shows that the LMOHA scheme offers slightly higher recognition accuracy than does the MOHA scheme. This is because only 25% of the testing dataset is affected by the scaling issue (scaled patterns), which is the main weakness of the LMOHA scheme. Figure 6.7 shows that the MOHA and LMOHA schemes have higher detection accuracy and lower error rates compared with those of the other schemes. The main reason for the latter's low recognition accuracy is the small number of training samples that might not provide enough information for them to achieve a high accuracy level. Further discussion about the other schemes' performance will be provided at the end of the next sub-section.

It can clearly be seen that the MOHA and LMOHA schemes are able to offer higher recognition accuracy levels for the action dataset with small number of training samples compared with those of other well-known schemes. Moreover, this case study proves that the proposed schemes are able to address activity recognition problems using a vision-based dataset. In the next sub-section, we will examine the proposed schemes' capabilities when addressing activity recognition problems using a sensor-based dataset; this will indicate whether they will be capable of dealing with all kinds of activity datasets.

6.4.2 Sensor-based human activity recognition

This sub-section will determine whether the MOHA and LMOHA algorithms are able to perform human activity recognition using a dataset recorded by means of ambient and wearable sensors. In recent research, sensor-based activity recognition systems have become a more attractive solution for human activity recognition compared with vision-based systems [278]. The utilisation of camera systems is acceptable and practical when activities are confined to a limited area (e.g. parts of a house or office); however, when the human activity involves going from place to place, then sensor-based systems will be the better option [46]. Moreover, one of the advantages of sensor-based activity recognition systems is that they do not interfere with the human or user privacy as is the case with vision-based systems [46].

There are several types of sensors which are capable of collecting different types of measurements. For example, accelerometer sensors measure the acceleration of an object while gyroscope sensors measure the orientation [294]. These types of sensors can be wearable devices that are attached to a user’s body. On the other hand, other types of sensors can be attached to physical objects such as doors and drawers. These sensors provide sensory information for analytical and pattern recognition systems which transform measures and readings into activities. Moreover, different sensors measure different readings which can be utilised for analysis and recognition. For instance, accelerometer sensors measure the acceleration rates of an object while gyroscope sensors measure orientation based on momentum [294].

Firstly in this sub-section, a detailed description of the sensor-based dataset will be presented. Then, the MOHA and LMOHA sensor-based human activity recognition systems will be proposed, and this is followed by a discussion of the simulation results. The result obtained from the proposed schemes will be compared against three well-known classification methods, namely: KNN, Naive Bayes, and Multi-layer NN.

6.4.2.1 Opportunity dataset

An opportunity dataset [45, 295] is a rich database of collected information from sensor devices that record the human activities of different subjects. These sensors are deployed on the body and on surrounding objects. The sensors deployed include inertial, accelerometer, and compass sensors. The sensors deployed on the body form a WSN and the ones deployed on objects form a wired network. The challenge dataset presented in [296] contains the reading measures of deployed sensors for four subjects, which is a part of the data collected in the opportunity dataset. The main focus in this case study was on the body-worn sensors that represent the WSN part of the setting. In this case study, 39 sensors were worn that contained information about subjects' activities. These sensors were deployed in different parts of each subject's body. The types of sensors were as follows: 12 accelerometer sensors, 7 inertial sensors, and 2 compasses. Each one of the accelerometer sensors provided 3D readings (x, y and z). Five inertial sensors were deployed on the upper part of the body, each one containing three 3D readings, namely:

acceleration, orientation, and magnetic field. The other two inertial sensors were deployed on the shoes (left and right) and each one obtained five 3D readings, namely: acceleration, orientation, magnetic field, rate of turn, and angular velocity. Each of the two compasses provided a single reading that gave the direction of the object. This came to 113 attributes for each data instance.

The recorded activities were manually labelled. Two types of activities were targeted, namely, locomotion and gestures. The locomotion activities included four types of activities: standing, walking, sitting, and lying down, labelled as 101, 102, 104, and 105 respectively. Gestures included detailed activities such as opening or closing a drawer. In this case study, locomotion activities were studied as gesture activities, relying on the wired objects' mounted sensors. This case study attempted to classify the four locomotion activities based only on the measures provided by the body-worn sensors. This means that the time stamp of when an activity occurred was also neglected. This is to demonstrate the capability of the MOHA and LMOHA schemes in classifying activities without recording the historical information about a subject's behaviour. This is intended to minimise memory requirements so as to meet WSN resource constraints.

6.4.2.2 MOHA and LMOHA schemes for sensor-based activity recognition

A challenge dataset (a part of the data collected in the opportunity dataset) is utilised in this case study to address the issue of human activity recognition using MOHA and LMOHA schemes. The first step in this case study was to handle the dataset in order to present a valid reading to the network. The sensory measures in this dataset are based on time. Each data instance represents the sensory measures at a specific time. Moreover, some instances in this dataset contain faulty readings which are labelled as (NaN). In this case study, the instances containing more than three invalid readings were removed.

The second step was to address the action's time in activity recognition. In this case study, we utilise the same technique as for the vision-based activity recognition to solve this issue. In this case study, we used the majority voting technique as a means of obtaining a combined decision on the recalls made by the proposed schemes within specific time period t_n. Therefore, the stored class (i.e. action) that received the highest recall over a specific time period t_n will be recognised as the performed action.

The third step was to ascertain the relationship between sensors and the S&I in the MOHA and LMOHA networks. Since the sensors provide readings in 3D format, each sensor exchanges information with its adjacent sensors. In this case study, the network is made up of 37 sensor nodes that offer the 3D readings. The two compass readings (1D) send information directly to the S&I

(in the base station) without exchanging. Each sensor node in the network senses or obtains three values. Each value represents a value in a certain dimension. Each value in a certain dimension is considered to be adjacent to the neighbour's value of the same dimension. For example, the value of x in the first sensor is adjacent to the value of x in the second sensor. Figure 6.8 shows the connectivity relationship between adjacent sensor nodes in the MOHA and LMOHA networks. In the MOHA scheme, after receiving all adjacent neighbours' values in all dimensions, each sensor node will determine whether or not it represents a pattern edge in each dimension, according to Equation 3.29. Each activated edge node in each dimension sends a message that contains its location information and its dimension number to the S&I (in the base station). After that, the S&I receives all messages from all activated edge nodes in the network along with compass sensors' readings. The S&I calculates the MOHA value (MV) for each dimension. In this case study, the MV for a pattern is a function of the number of dimensions and the value of compass sensors as follows:

$$MV = \frac{MV_{d1}+MV_{d2}+MV_{d3}}{C_1 * C_2 * 3} \qquad (6.1)$$

where MV is the MOHA value of a pattern, MV_{d1} is the MOHA value for the first dimension, MV_{d2} is the MOHA value for the second dimension, MV_{d3} is the MOHA value for the third dimension, C_1 is compass one sensor's value, and C_2 is compass two sensor's value.

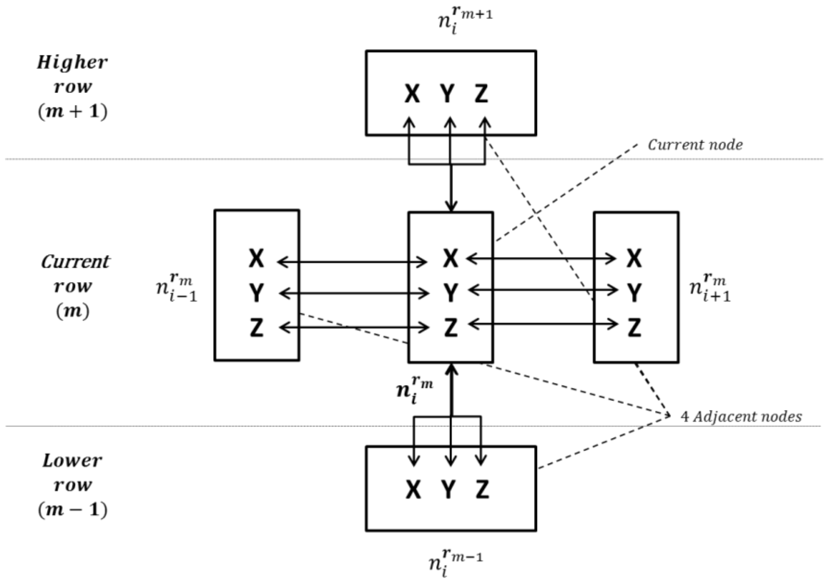

Figure 6.8: Connectivity between adjacent nodes in a 3-D MOHA and LMOHA networks.

On the other hand, in the LMOHA scheme, after receiving the values of all adjacent nodes in all dimensions, the node will determine whether or not it represents a pattern sensory-based shape in each dimension, according to Equation 4.3. Each activated plus, T, corner, and edge nodes in each dimension sends a message containing its type information and dimension number to the S&I. After that, the S&I (in the base station) receives all messages from all activated plus, T, corner, and edge nodes in the network along with compass sensors' reading. In this case study, a pattern vector is a combination of the average number of activated plus nodes of all dimensions ($\frac{N_{plus}}{3}$), the average number of activated T nodes of all dimensions ($\frac{N_T}{3}$), the average number of activated corner nodes of all dimensions ($\frac{N_{corner}}{3}$), the average number of activated edge nodes of all dimensions ($\frac{N_{edge}}{3}$), and the value of compass sensors, $\vec{P}(\frac{N_{plus}}{3}, \frac{N_T}{3}, \frac{N_{corner}}{3}, \frac{N_{edge}}{3}, C_1, C_2)$, which is utilised for recognition.

In this case study, the recognition period of time was set to 10 instances and each of the LMOHA and MOHA networks was divided into 6 rows each of which contained 7 nodes except for the higher row that contained only two nodes. The threshold value for edge determination was set to 500 ($\varphi = 500$). The proposed schemes were trained using 30 randomly selected data instances for each locomotion or action class. To compare the proposed schemes with other schemes, the Weka tool [263, 264] was utilised to simulate three different schemes: KNN (k=1), Naive Bayes, and Multi-layer NN. Figure 6.9 shows the receiver operating characteristic (ROC) space and the plots for the four activity classes for the MOHA, LMOHA, KNN (k=1), Naive Bayes, and Multi-layer NN schemes. Table 6.3 shows the details of MOHA, LMOHA and other schemes' recognition accuracy levels obtained for each locomotion class. The same training and testing datasets were used for all schemes.

Table 6.3 clearly shows that the MOHA and LMOHA schemes have an overall higher accuracy level compared with that of other schemes. The table also shows that the MOHA scheme offers slightly higher recognition accuracy than does the LMOHA scheme. This might be due to the way the MOHA scheme handles the sensors' 3D readings and calculates the MOHA values for them. Figure 6.9 also shows that the MOHA and LMOHA schemes have overall higher detection accuracy and lower error rates compared with those of the other schemes.

Table 6.3 and Figure 6.9 show that the Naive Bayes scheme recorded the lowest overall recognition accuracy compared with that of other schemes, and recorded most of the testing instances as pattern 102 (Walking). The main reason for this result is the small number of training instances. In order for the Naive Bayes scheme to perform pattern recognition, it needs to build a probabilistic relationship between the training instances and input variables. Since the number of training instances is limited, the created probabilistic relationships cannot efficiently describe each pattern [96, 103].

Table 6.3: Recognition accuracy results of challenge dataset for different schemes.

Class	101 (Standing)	102 (Walking)	104 (Sitting)	105 (Lying down)	**Overall**
Training instances	30	30	30	30	120
Testing instances	145951	80215	79982	18113	324261
MOHA	46.61%	49.42%	94.81%	42.70%	58.98%
LMOHA	47.20%	46.93%	91.88%	40.88%	57.80%
Naive Bayes	3.81%	98.01%	6.12%	12.22%	28.15%
KNN (*k*=1)	46.77%	34.55%	96.11%	39.98%	55.54%
Multi-layer NN	39.31%	43.66%	97.34%	40.64%	54.77%

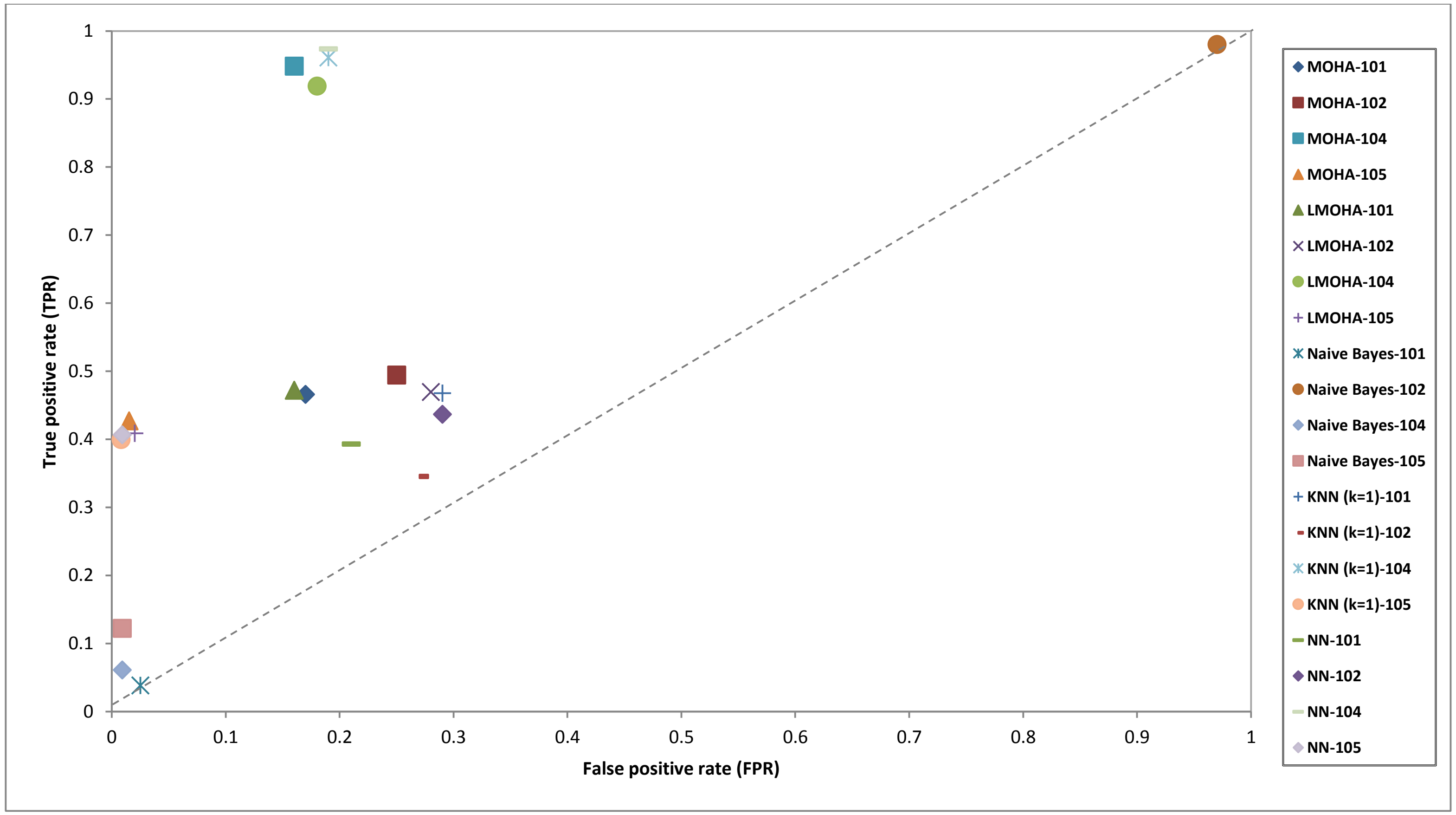

Figure 6.9: The ROC space and the plots for the challenge dataset for MOHA, LMOHA, Naive Bayes, KNN, and Multi-layer NN schemes.

Both the KNN and multi-layer NN schemes' scores on recognition accuracy are to some extent comparable to those of the proposed schemes. However, both of them have their own requirements in order to reach these accuracy levels. In the KNN (*k*=1) scheme, each node in the network saves the input values of the training data samples and then compares the incoming pattern values with the stored information. After calculating the distances, the nodes report directly to the base station, which also holds information about training samples. This process requires the memory resources available in each node in the network to store such information. By increasing the number of training instances, the memory requirements and the time required to calculate distances for each node increase. On the other hand, the nodes in the proposed schemes' networks do not store any information about training instances. Instead, each one of them reports its location or type information to the S&I (in the base station). This alleviates the requirement for memory resources in each node. Therefore, no search time is required by nodes in order to match patterns.

The multi-layer NN scheme involves a greater number of communications and iterations compared to that of the KNN (*k*=1) scheme. In this case study, multi-layer NN involved input layers, a hidden layer that contained 71 nodes, and a four-node output layer. The network structure required each node in each layer to communicate with each node in the higher layer. This means that each node in the input and the output layers had 71 connections to the hidden layer. Since the input layer contained 113 nodes (the pattern size) and the output layer contained four nodes, each node in the hidden layer had 117 connections to the input and output layers' nodes. Similarly, each node in the output layer required four connections to the hidden layers' nodes. The total number of connections in this case study was 16950. Moreover, the network required 500 iterations for each incoming pattern in order to converge. On the other hand, each node in the proposed schemes involves only four connections to its adjacent nodes with a one connection to the S&I (if it reaches the second level of activations). Since the compass sensors report directly to the S&I, each one of these nodes requires only one communication to the base station. Therefore, in the proposed schemes, the total number of communications for the worst-case scenarios (when all nodes are activated as edge nodes in MOHA scheme and when all nodes are activated as plus, T, corner, or edge nodes in LMOHA scheme) was 557. Moreover, both MOHA and LMOHA networks involve a single cycle in order to converge. This reduction in the number of communications and iterations has ramifications for the performance of the network in terms of resource consumption (e.g. energy) and convergence time.

It can clearly be seen that, compared with other well-known schemes, the MOHA and LMOHA schemes can offer higher recognition accuracy levels when dealing with challenge dataset with limited requirements in terms of the number of training instances, number of communications, and convergence time. Moreover, this case study proves that the proposed schemes have the capabilities to address activity recognition problems using sensor-based datasets. The issue of human activity recognition could involve more complicated means such as physical analysis and filters, which could improve accuracy. The presented case studies show that the MOHA and LMOHA schemes are capable of addressing human activity recognition issues in resource-constrained environments (such as WSN) as they score respectable accuracy levels with a small amount of resource consumption.

6.5 Conclusions

This chapter has presented and discussed several simulations for both proposed schemes. The results of these simulations show that the MOHA and LMOHA schemes have a very high level of accuracy when dealing with transformed patterns compared with that of other schemes. The results also show that the proposed schemes' networks are capable of addressing complex and real-life recognition problems by using a minimal amount of training information. They also show the feasibility of utilising the proposed schemes in real scenarios and different application domains.

The first test was used to compare the accuracy of the proposed schemes with that of well-known existing schemes when dealing with transformed patterns using the *shapes* and the *stars* datasets, which consist of uniform and non-uniform patterns, respectively. The results of this test show that the MOHA and LMOHA schemes have a very high accuracy rate when dealing with transformed patterns compared with that of other existing schemes. The results also show that, overall, the MOHA scheme offers higher recognition accuracy than does the LMOHA scheme when dealing with transformed patterns. The main reason for this is the MOHA scheme's excellent ability to handle scaled patterns. The ROC graph in this test shows that the MOHA and LMOHA schemes have higher detection accuracy and lower error rates compared with the other schemes when dealing with transformation patterns.

The second test involved a handwritten character recognition test. The results of this test show that the MOHA and LMOHA schemes have pattern recognition capabilities that produce better accuracy results than do the Naive Bayes and BP NN schemes. The results also show that the LMOHA offers higher recognition accuracy when dealing with the problem of handwritten character recognition than does the MOHA. The main reason for this is the LMOHA scheme's excellent ability to handle rotated patterns that are the most comment issues in handwritten character recognition. Finally, this test demonstrated that the proposed schemes' networks are capable of performing complex and real-life recognition by using a minimal amount of training information, which resolves the issue of randomness associated with WSNs.

The final test discussed the possibility of using the proposed schemes in human activity recognition systems. In this test, the proposed schemes were compared with other existing techniques using a limited amount of collected data to demonstrate the capability of the schemes in addressing activity recognition problems using vision-based and sensor-based datasets. The results of this test show that both proposed schemes are capable of offering higher recognition accuracy levels for vision-based and sensor-based datasets with limited requirements in terms of the number of training instances, number of communications, and convergence time compared with other well-known schemes. Moreover, this case study proved that the proposed schemes are capable of resolving activity recognition problems using vision-based and sensor-based datasets. In conclusion, the presented case studies showed the capabilities of the MOHA and LMOHA schemes in addressing human activity recognition issues in resource-constrained environments (such as WSN) as they score respectable accuracy levels with a small amount of resource consumption.

The next chapter will provide an overall discussion of the thesis and its limitations, in addition to possible future research directions.

Chapter 7

Conclusions and Future Work

7.1 Summary of the Research

This research proposes two distributed and parallel pattern recognition schemes for event detection that are capable of detecting transformed and noisy patterns using a minimal amount of available information about patterns while addressing the resource constraints of WSNs.

This research reviewed existing pattern recognition schemes for resource-constrained WSNs. In order to achieve a scalable scheme that meets the requirements of such networks, the scheme must combine limited communications and computations with using minimal memory resources. Additionally, the scheme must also involve low time complexity in order to serve online recognition applications. To deal with the real-life sensory problems of WSNs, one expects a good recognition scheme to be capable of dealing with complex, transformed and noisy patterns with minimal available information. These requirements for good schemes are derived from the challenges posed by WSN applications. Existing recognition schemes for

WSNs have several limitations in terms of these challenges such as iterative and fully-centralised processing.

This research proposed two novel collaborative *in-network* pattern recognition-based event detection schemes which are lightweight and scalable and well-suited to resource-constrained networks such as WSNs. In this research, two pattern recognition schemes were proposed: the Macroscopic Object Heuristics Algorithm (MOHA) and the Light Macroscopic Object Heuristics Algorithm (LMOHA). The computational complexity analysis of MOHA algorithm shows that it suffers a very slightly high computational complexity with respect to its longest distances determination process, which is handled by the S&I (as been shown in sub-section 3.7.1). However, the overall computational complexity of MOHA is still less than that of other existing schemes, as discussed in sub-section 3.7.1. Therefore, the research proposed the LMOHA algorithm to reduce the computational complexity of the MOHA's S&I for event detection and pattern recognition, which will lead to a reduction of the overall computational complexity of the MOHA scheme. Both proposed schemes adopt the distributed parallel recognition mechanisms of Graph Neuron (GN) to minimise recognition computations and communications and thus will lead to maintaining low levels of consumption of the limited resources. The distributed network structure of the proposed schemes results in a loosely coupled connectivity between a network's nodes and avoids iterative learning. The proposed schemes also adopt a two-step activation process to reduce the number of participating nodes in the recognition process so as to minimise communicational overheads and to increase network scalability. Therefore, the proposed schemes perform recognition operations in a single learning cycle of predictable duration, which make them good candidates for implementation of large-scale, real-time problems.

In order to have a high level of recognition accuracy, the pattern recognition scheme for event detection in WSNs should be able to identify dynamic and continuous changes in patterns and deal with limited prior knowledge of events. Thus, the first proposed scheme (i.e. MOHA) implements an edge detection gradient-based mechanism that searches the edges and boundaries of patterns. The second proposed scheme (i.e. LMOHA) implements a similar mechanism as MOHA; however, its mechanism searches for the sensory-based shapes of patterns. These mechanisms allow the proposed schemes to identify dynamic and continuous changes in patterns. Thus, the proposed schemes are capable of detecting different types of pattern transformation such as translation, rotation, and dilation. The presented mechanisms depend only on the change rate between the active node's four adjacent nodes to minimise communications and computations. Moreover, the proposed schemes are capable of performing recognition operations in dynamic environments and also provide a high level of detection accuracy using a minimal amount of available information about patterns. The protocols required for performing the schemes' operations were also presented and discussed.

This research presented theoretical and experimental analysis and evaluation of both proposed schemes. The evaluation includes time complexity, recognition accuracy, communicational and computational overhead, energy consumption and lifetime analysis. The schemes' performance is also compared with that of existing recognition schemes. This shows that the proposed schemes are capable of minimising computational and communicational overheads in resource-constrained networks, enabling those networks to perform efficient recognition activities for patterns that involve transformations within a single learning cycle while maintaining a high level of scalability and accuracy. The results show that a network that implements mica 2 motes and requires 3.0625 milliseconds to send a single message can perform recognition operations within a single learning cycle duration, ranging between 5.17 and 2231.39 milliseconds using the MOHA scheme and 5.17 and 16,441.33 milliseconds using the LMOHA scheme, for 40,000- and 65,536-node network settings, respectively. The results also show that using a multi-channel MAC message exchange model in both proposed schemes will considerably reduce the network's learning cycle time. The results also show that energy requirements can be decreased by up to 75.86% using the MOHA scheme and by 70.69% using the LMOHA scheme, in comparison with other recognition techniques. In terms of efficiency, theoretical and experimental analyses show that both proposed schemes are highly capable of dealing with noisy and transformed patterns with a high level of accuracy. However, each presented different limits of tolerance to noisy patterns and different types of transformed patterns. The results show that the MOHA scheme offers more accurate recognition for scaled patterns than does the LMOHA scheme. However, the LMOHA scheme provides more accurate recognition for noisy and rotated patterns than does the MOHA scheme. In conclusion, both proposed schemes showed a very significant capability of performing pattern recognition in WSNs, as they showed a very good capability of handling noisy and transformed patterns and limiting the number of communications, thereby limiting the consumption of energy resources.

Finally, the research presented and discussed several simulations for the proposed schemes. The results of these simulations showed that the proposed schemes achieve very high accuracy when dealing with transformed patterns compared to that of other existing schemes. The results also showed the capability of the proposed schemes’ networks of performing complex and real-life recognition problems by using a minimal amount of training information. They also show the feasibility of utilising the proposed schemes in real scenarios and different application domains such as handwritten character recognition and human activity recognition systems. The results of these simulations showed that the proposed pattern-recognition-based approaches can perform better than existing classifiers.

7.2 Research Contributions and Outcomes

The outcomes of this research contribute mainly to the field of pattern recognition in networks and systems such as WSNs that have limited resource.

More specifically, this research presents schemes that are capable of performing efficient online pattern recognition while maintaining minimal resource requirements that suit such limited systems. The significance of this research can be encapsulated as four major contributions: lightweight network design, online recognition performance, noisy, and transformation-invariant recognition, and adaptability to real scenarios and different application domains.

The design of lightweight and distributed schemes is the first key contribution of this research. This design ensures high network scalability and increases its lifetime. The lightweight scheme design has been achieved by using distributed communicational and computational mechanisms, along with a multi-row network design that minimises information exchange overheads. The network design of the schemes limits the number of messages required for each node to typically four exchange and one report messages. This relieves the network nodes of the tightly coupled connectivity requirements found in other schemes such as neural networks. Moreover, the designs of the proposed schemes networks adopt activation mechanisms that minimise the number of network nodes participating in the recognition process. This leads to further minimisation in the schemes' complexity, increases the network's lifetime, and increases network scalability. Experiments conducted on the MOHA and LMOHA schemes (in sub-section 5.3.1) show that the proposed activation process involved a range of only 0.53% to 25.67% of the total number of the network's nodes in the information exchange process. The same experiments showed that a range of only 0.08% to 0.74% of nodes for MOHA scheme and 0.30% to 25.67% of nodes for the LMOHA scheme were involved in the reporting process (see sub-section 5.3.1). In terms of communicational requirements, the experimental tests in sub-section 5.3.2 show that the communications required for a MOHA network of size 65536 nodes ranges between 19823 and 39355 depending on the message sequence model. This is a range between 30.25% and 60.05% of the network size and of the number of communications required by parallel KNN. On the other hand, it was shown experimentally that the communications required for a LMOHA network of size 65536 nodes ranges between 24415 and 43947 depending on the message sequence model. This is a range between 37.25% and 67.06% of the network size and of the number of communications required by parallel KNN. Moreover, it was shown experimentally that the number of communications required for a MOHA network of size 40000 nodes ranged between 883 and 1735. That is a range between 2.21% and 4.34% of the network size, and of the required communications of other schemes (see sub-section 5.3.2). On the other hand, the experimental tests show that the number of required communications for a LMOHA network of size 40000 nodes ranged between 971 and 1823. This is a range between 2.43% and 4.56% of the network size, and of the communications required by other schemes.

This minimisation of the number of participating nodes also extends the lifetime of the network. The tests conducted in this research in sub-section 5.3.2 show that the average energy consumption ranged between 0.70 and 1.39

mJ for the MOHA scheme and between 0.85 and 1.54 mJ for the LMOHA scheme depending on the sequence model involved. Other schemes such as parallel KNN scored an average energy consumption of 2.9 mJ. An analysis of these results shows that a MOHA network can have a lifetime of one year with one of the smallest batteries in terms of capacity (i.e. 30 mAh or 324 joules). That is four times higher than other schemes such as parallel KNN (see sub-section 5.3.2). On the other hand, the results show that a LMOHA network can have a lifetime of nine months with 30 mAh batteries. That is three times higher than the lifetimes of other schemes such as parallel KNN.

The second significant contribution of this research is that it demonstrates that pattern recognition can be achieved within a predictable single learning cycle time frame and with low computational complexity. This capability allows the scheme to support online and real-time applications. The time complexity analysis of the proposed schemes in sub-section 3.7.2.1 shows that the total number of nodes (the network size) in the wireless sensor network and the pattern size (S_p) do not have any influence on the total time taken for learning process. As a result, the schemes can be effectively scaled to support any number of nodes in the wireless sensor network. This time complexity of the proposed schemes has been estimated for a worst-case scenario. That is when all nodes are activated and participate in the learning process. The proposed schemes' networks response times are proportional to the number of activated nodes in the networks, which are generally less than the pattern size. These minimise the effect of pattern size increase and provide the schemes with a high level of scalability. Furthermore, the analysis shows that the schemes are capable of performing learning operations within a predictable time while restricting the learning cycle so that they are proportional to the number of activated nodes (typically less than the pattern size). These features make the schemes appropriate for tackling large-scale, real-time problems. The computational complexity analysis of the proposed scheme in sub-section 3.7.1 shows that the overall computational complexity of MOHA is less than that of other existing schemes such as the Hopfield network. For example, the total computational complexity for the MOHA algorithm utilising 35 nodes is 668 during the recognition process. However, it is 44135 in the Hopfield network implementation for the recognition stage with the same number of nodes. The computational complexity analysis of the MOHA algorithm in sub-section 3.7.1 shows that it has slightly high computational complexity with respect to its recognition process. As a result, LMOHA scheme is proposed to reduce the computational complexity of the MOHA scheme. The computational complexity analysis of the LMOHA algorithm in sub-section 4.5.1 shows that the LMOHA incurs less computational complexity during the pattern recognition process compared with the MOHA scheme. For instance, the total computational complexity for LMOHA algorithm utilising 35 nodes is 70 in recognition process. However, it is 668 in the MOHA implementation for the recognition stage with the same number of nodes as shown previously. It is clearly shown from the analysis that the MOHA and LMOHA schemes incur lesser computational complexity for their pattern recognition processes compared with other existing schemes such as the Hopfield network, and are therefore an appropriate solution for resource-constrained networks, online and real-time applications. LMOHA provide even less computational complexity compared with the MOHA, making it even more suitable for these types of networks and applications.

The third significant contribution of this research is efficient pattern recognition capabilities for noisy and transformed patterns. The MOHA and LMOHA schemes presented in Chapters 3 and 4 are designed to deal mainly with noisy and transformed patterns. The proposed schemes use an edge detection mechanism to locate and determine the edges and the boundaries of an event. By describing events and patterns using their main edges and boundaries, it is possible to achieve an efficient recognition scheme that can detect transformations that may occur in these events and patterns. To the best of our knowledge, the schemes proposed in this research are the first pattern recognition schemes to utilise edge determination mechanisms based on a well-established edge detection technique of image segmentation in the recognition process, which can offer transformation-invariant detection capabilities. The first proposed scheme in this research implements an edge detection gradient-based mechanism that searches the edges and boundaries of patterns and replaces traditional local information-storing methods. The second proposed scheme implements a mechanism similar to the first one; however, its mechanism searches for the sensory-based shapes of patterns. These mechanisms allow the proposed schemes to identify dynamic and continuous changes in patterns. Experimental analyses of the schemes presented in section 4.6 show their ability to deal with translation, rotation, and dilation pattern transformation types. In section 4.6, both approaches showed very significant ability to handle distorted (noisy) and transformed patterns. However, each one of them presented different limits of tolerance to distorted patterns and different types of transformed patterns.

Sub-section 4.6.1 showed that the MOHA approach can provide 100% recognition accuracy (five correctly classified patterns) for distortion levels up to 25% and 80% recognition accuracy (four correctly classified patterns) for distortion levels up to 45% for distorted patterns obtained from the *shapes* dataset that contains uniform patterns. Additionally, results presented in sub-section 4.6.2 showed that MOHA scheme is capable of achieving 100% recognition accuracy for distortion levels up to 15% and 80% recognition accuracy for distortion levels up to 35% for distorted patterns obtained from *stars* dataset that contains non-uniform patterns. On the other hand, sub-section 4.6.1 showed that the LMOHA scheme is capable of offering 100% recognition accuracy (five correctly classified patterns) for distortion levels up to 35% and 80% recognition accuracy (four correctly classified patterns) for distortion levels up to 50% for the *shapes* dataset. Results presented in sub-section 4.6.2 showed that LMOHA scheme is capable of offering 100% recognition accuracy for distortion levels up to 25% and 80% recognition accuracy for distortion levels up to 40% for the *stars* dataset. The previous results clearly show that the LMOHA scheme provides more accurate recognition for distorted patterns than does the MOHA scheme.

For transformed patterns, both proposed approaches showed 100% recognition accuracy when they deal with translated patterns. However, each proposed approach had different capabilities when dealing with scaled and rotated patterns. Sub-section 4.6.1 showed that the MOHA approach can provided 100% recognition accuracy (five correctly classified patterns) for all tested scaled patterns obtained from the *shapes* dataset that contains uniform patterns. Additionally, sub-section 4.6.2 results showed that MOHA scheme is capable of offering perfect recognition accuracy (100% accuracy level - five correctly classified patterns) for scaling levels up to 60% for the *stars* dataset that contains non-uniform patterns. The sub-section results also showed that the MOHA is capable of achieving 80% recognition accuracy (four correctly classified patterns) level for scaling levels up to 80% for the same dataset. On the other hand, sub-section 4.6.1 showed that the LMOHA scheme is capable of offering perfect recognition accuracy (100% accuracy level - five correctly classified patterns) for scaling levels up to 16% for the *shapes* dataset. The sub-section also showed that the LMOHA is capable of achieving 80% recognition accuracy (four correctly classified patterns) level for scaling levels up to 32% for the uniform patterns. Moreover, sub-section 4.6.2 results showed that LMOHA scheme is capable of achieving perfect recognition accuracy (100% accuracy level) for scaling levels up to 16% for *stars* dataset. The sub-section results also showed that the LMOHA is capable of offering 80% recognition accuracy level for scaling levels up to 30% for the non-uniform patterns. The previous results clearly show that the MOHA scheme provides more accurate recognition for scaled patterns than does the LMOHA scheme.

Sub-section 4.6.1 showed that that MOHA approach provided high accuracy percentages (100% accuracy level - five correctly classified patterns) in five rotational regions for a *shapes* dataset that contains uniform patterns. The first region is between 0 and 4 degrees, the second region is between 86 and 94 degrees, the third region is between 176 and 184 degrees, the region where patterns are horizontally flipped or nearly flipped, the fourth region is between 266 and 274 degrees, and the fifth region is between 356 and 360 degrees. Moreover, sub-section 4.6.2 showed that MOHA approach provided high accuracy percentages in five rotational regions for *stars* dataset that contains non-uniform patterns. The first region is between 0 and 9 degrees, the second region is between 81 and 94 degrees, the third region is between 176 and 189 degrees, the fourth region is between 261 and 279 degrees, and the fifth region is between 351 and 360 degrees. On the other hand, sub-section 4.6.1 showed that LMOHA approach provided high accuracy percentages (100% accuracy level - 5 correctly classified patterns) in five rotational regions for *shapes* dataset. The first area is between 0 and 29 degrees, the second area is between 61 and 119 degrees, the third area is between 161 and 199 degrees, the fourth area is between 251 and 289 degrees, and the fifth area is between 341 and 360 degrees. Sub-section 4.6.2 showed that the LMOHA approach provided high accuracy percentages in five rotational regions for the *stars* dataset that contains non-uniform patterns. The first area is between 0 and 39 degrees, the second area is between 56 and 119 degrees, the third area is between 146 and 194 degrees, the fourth area is between 256 and 284 degrees, and the fifth area is between 346 and 360 degrees. The previous results show that the LMOHA scheme provides more accurate recognition for rotated patterns than does the MOHA scheme.

The fourth contribution of this research is that it demonstrates the advantages of using pattern-recognition-based techniques in dealing with real scenarios and different application domains such as handwritten character recognition and human activity recognition systems. In Chapter 6, MOHA and LMOHA classification models were presented to deal with complex classification problems. The results of these simulations showed that the proposed pattern recognition-based approaches can perform better than do the existing classifiers. Such models show the ability of the schemes to perform classification tasks with minimal resource requirements using pattern recognition capabilities while maintaining high accuracy compared with other classification schemes. The examples presented in Chapter 6 show the capability of the proposed schemes to adapt to different types of complex learning applications and systems.

In general, the one of the main goals of the schemes presented in this thesis is to provide efficient pattern recognition capabilities. Both the MOHA and LMOHA schemes were compared with other schemes in terms of accuracy. In Chapter 6, a series of simulations for both proposed schemes were conducted. The performance of the proposed schemes was compared with those of other well-known pattern recognition schemes. Three tests were described and discussed in this chapter. The first test aimed to compare the accuracy of the proposed schemes with that of KNN, Naive Bayes, and SVM schemes in dealing with transformation patterns. The test results show that MOHA and LMOHA schemes have a very high accuracy in dealing with transformed patterns compared to other schemes. The MOHA scheme was capable of achieving high an accuracy level of 85.61% and LMOHA scheme was capable of achieving a higher accuracy level of 80.01% compared with those of other schemes that scored accuracy levels ranging between 44.22% and 55.78%. The previous results show that the MOHA scheme achieves higher recognition accuracy than does the LMOHA scheme. The main reason for this is the MOHA scheme’s superior ability to handle scaled patterns. The second test was intended to demonstrate whether the MOHA and LMOHA schemes are able to deal with complex and real-life problems. It also aimed to compare the MOHA’s and LMOHA’s accuracy with that of the Naive Bayes and back propagation networks using a complex handwritten recognition problem. The test results showed that the proposed schemes are capable of dealing with the problem while providing higher accuracy levels compared with those of other schemes. The results also show that the LMOHA offers higher recognition accuracy when dealing with handwritten character recognition problem than does the MOHA. The main reason for this is the LMOHA scheme’s excellent ability to handle rotated patterns that are the most common issues in handwritten character recognition. The final test examined the possibility of using the proposed schemes in human activity recognition systems. In this test, the proposed schemes were compared with KNN, Naive Bayes and Multi-layer neural networks using a limited amount of collected data to demonstrate the capability of the schemes in addressing activity recognition problems using vision-based and sensor-based datasets. The results of this test show that both proposed schemes are capable of achieving higher recognition accuracy levels for vision-based and sensor-based datasets with limited requirements in terms of the number of training instances, number of communications, and convergence time compared with those of other well-known schemes. Moreover, this case study proved the capabilities of the proposed schemes in addressing activity recognition problems using vision-based and sensor-based datasets. In conclusion, the presented case studies showed that the MOHA and LMOHA schemes are capable of resolving human activity recognition issues in resource-constrained environments (such as WSN) as they achieve respectable accuracy levels with a small amount of resource consumption.

7.3 Future Work

This research has presented schemes mainly intended to provide online and efficient pattern recognition capabilities for large-scale and resource-constrained networks such as WSNs. The features of the proposed schemes pave the way for further enhancements and research opportunities. A useful extension of this research and further studies could involve the following:

1. **Edge detection mechanisms:** The proposed schemes use an edge detection mechanism to locate and determine the edges and the boundaries of an event. By describing events and patterns using their main edges and boundaries, it is possible to achieve an efficient recognition scheme that can detect transformations that may occur in these events and patterns. The proposed schemes in this research implement an edge detection gradient-based mechanism that searches the edges and boundaries of patterns. The presented mechanism depends only on the change rate between the active node's four adjacent nodes. This mechanism dealt with transformations of patterns such as translation, rotation, and dilation. A good extension of the research in this context would be to implement different edge detection mechanisms that could lead to higher levels of detection accuracy and could deal with other types of transformations and problems. The trade-offs between implementing more complex edge detection mechanisms and network performance would be a rich research area as more complex mechanisms could lead to greater costs in terms of resource consumption and speed. The type of application and available resources are expected to be the main criteria that drive such trade-offs. Also, determining the best threshold values for edge determination can be a difficult task in some WSN applications. In this research, we

found that setting the threshold values for edge determination to the average values range provides very good results in most cases. Thus, further research could be undertaken to investigate the best way to determine the threshold values for edge determination in WSNs.

2. **Multi-dimensional design:** The proposed schemes in this thesis were modelled and designed to deal with 1-D to 3-D problems. For example, the human activity classification model presented in sub-section 6.4.2.2 deals with a 3-D problem space. However, other problems may involve a greater number of dimensionality requirements. This research proposed the use of multi-row network structures with multi-dimensional levels to deal with multi-dimensional problems. However, the design constraints of the network structure of the proposed schemes can be challenging. Hence, designing and analysing multi-dimensional network structures based upon the schemes proposed in this thesis would be a good extension of the research. New research would focus on design capabilities and prediction of the behaviour and performance of these schemes when dealing with multi-dimensional patterns.
3. **The influence of different WSNs layers:** The impact of MAC protocols was explored in this thesis. In sensor networks, however, all WSNs layers need to be co-designed in order to attain optimal performance whilst conserving the network's resources. Communication is considered to be the highest consumer of the energy of sensor nodes. Investing in the improvement of the various aspects of

communication and the network layer would result in a significant improvement of the overall performance of the network. In addition, understanding the various proposed mechanisms employed to route and process sensory data would enhance the node collaboration and the event recognition, and prolong the longevity of the network.

4. **Adaptation to different application domains:** The lightweight capabilities of the schemes proposed in this research can also be used in different types of application domains. Chapter 6 discussed the ways in which the recognition capabilities of the MOHA and LMOHA schemes can be used to solve classification problems. These indicate further research opportunities that incorporate lightweight pattern recognition-based techniques such as the MOHA and LMOHA in the steps and processes involved in solving complex real-life problems. This could open up a wide area of research in fields such as artificial intelligence, optimisation, system security and network management.

References

[1] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics,* vol. 38, pp. 82-85, 1965.

[2] M. Weiser, "The computer for the 21st century," *Scientific American,* vol. 265, pp. 94-104, 1991.

[3] M. Ilyas and I. Mahgoub, *Smart dust: sensor network applications, architecture, and design*: CRC/Taylor & Francis, 2006.

[4] C. S. R. Murthy and B. S. Manoj, *Ad Hoc wireless networks: architectures and protocols*: Prentice Hall PTR, 2004.

[5] R. Niu and P. K. Varshney, "Target location estimation in sensor networks with quantized data," *Signal Processing, IEEE Transactions on,* vol. 54, pp. 4519-4528, 2006.

[6] G. Mao, B. Fidan, and B. Anderson, "Wireless sensor network localization techniques," *Computer Networks,* vol. 51, pp. 2529-2553, 2007.

[7] A. Khan, "A peer-to-peer associative memory network for intelligent information systems," in *the Thirteenth Australasian Conference on Information Systems,* 2002.

[8] M. Baqer and A. Khan, "Event Detection in Wireless Sensor Networks Using a Decentralised Pattern Matching Algorithm," 2008.

[9] F. Zhao and L. J. Guibas, *Wireless sensor networks: an information processing approach*: Morgan Kaufmann Pub, 2004.

[10] V. Raghunathan, C. Schurgers, S. Park, and M. Srivastava, "Energy efficient design of wireless sensor nodes," *Wireless sensor networks,* pp. 51-69, 2004.

[11] A. Hac, *Wireless sensor network designs*: Citeseer, 2003.

[12] H. Chan and A. Perrig, "Security and privacy in sensor networks," *Computer,* vol. 36, pp. 103-105, 2003.

[13] J. Sarangapani, *Wireless ad hoc and sensor networks: protocols, performance, and control*: CRC Press, 2007.

[14] C. Y. Chong and S. P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proceedings of the IEEE,* vol. 91, pp. 1247-1256, 2003.

[15] H. Karl and A. Willig, *Protocols and architectures for wireless sensor networks*: Wiley-Interscience, 2007.

[16] R. Verdone, D. Dardari, and G. Mazzini, *Wireless sensor and actuator networks: technologies, analysis and design*: Elsevier/Academic Press, 2008.

[17] K. Sohraby, D. Minoli, and T. F. Znati, *Wireless sensor networks: technology, protocols, and applications*: Wiley-Interscience, 2007.

[18] A. Hodjat and I. Verbauwhede, "The Energy Cost of Embedded Security for Wireless Sensor Networks," ed: John Wiley & Sons, 2006.

[19] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE,* vol. 40, pp. 102-114, 2002.

[20] A. Boukerche and S. Nikoletseas, "Protocols for data propagation in wireless sensor networks," *Wireless communications systems and networks,* pp. 23-51, 2004.

[21] A. Swami, *Wireless sensor networks: signal processing and communications perspectives*: J. Wiley, 2007.

[22] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks,* vol. 38, pp. 393-422, 2002.

[23] I. F. Akyildiz and M. C. Vuran, *Wireless Sensor Networks*: Wiley, 2010.

[24] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking,* 1999, pp. 263-270.

[25] H. W. Tsai, C. P. Chu, and T. S. Chen, "Mobile object tracking in wireless sensor networks," *Computer Communications,* vol. 30, pp. 1811-1825, 2007.

[26] G. Zhao, *Wireless Sensor Networks: An Information Processing Approach*: Elsevier (A Divisionof Reed Elsevier India Pvt. Limited), 2009.

[27] Frost and Sullivan, "Wireless Sensors and Integrated Wireless Sensor Networks," *Frost & sullivan report,* 2003.

[28] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, 2002, pp. 88-97.

[29] J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, and M. Welsh, "Wireless sensor networks for healthcare," *Proceedings of the IEEE,* vol. 98, pp. 1947-1960, 2010.

[30] O. Chipara, C. Lu, T. C. Bailey, and G.-C. Roman, "Reliable patient monitoring: A clinical study in a step-down hospital unit," *Dept. Comput. Sci. Eng., Washington Univ. St. Louis, St. Louis, MO, technical report WUCSE-2009-82,* 2009.

[31] J. Ko, J. H. Lim, Y. Chen, R. Musvaloiu-E, A. Terzis, G. M. Masson, *et al.*, "MEDiSN: medical emergency detection in sensor networks," *ACM Transactions on Embedded Computing Systems (TECS),* vol. 10, p. 11, 2010.

[32] T. Gao, C. Pesto, L. Selavo, Y. Chen, J. G. Ko, J. H. Lim, *et al.*, "Wireless medical sensor networks in emergency response: Implementation and pilot results," in *Technologies for Homeland Security, 2008 IEEE Conference on,* 2008, pp. 187-192.

[33] D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton, "Codeblue: An ad hoc sensor network infrastructure for emergency medical care," in

International workshop on wearable and implantable body sensor networks, 2004.

[34] G. Virone, A. Wood, L. Selavo, Q. Cao, L. Fang, T. Doan, *et al.*, "An advanced wireless sensor network for health monitoring," in *Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2)*, 2006, pp. 2-4.

[35] E. Ertin, N. Stohs, S. Kumar, A. Raij, M. al'Absi, and S. Shah, "Autosense: unobtrusively wearable sensor suite for inferring the onset, causality, and consequences of stress in the field," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, 2011, pp. 274-287.

[36] K. Patrick, "A Tool for Geospatial Analysis of Physical Activity: Physical Activity Location Measurement System (PALMS)," *Med Sci Sports Exerc*, vol. 41, p. 10, 2009.

[37] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich, "Collaborative signal and information processing: an information-directed approach," *Proceedings of the IEEE*, vol. 91, pp. 1199-1209, 2003.

[38] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless micro-sensor network models," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 6, pp. 28-36, 2002.

[39] J. Chamberland and V. V. Veeravalli, "Wireless sensors in distributed detection applications," *Signal Processing Magazine, IEEE*, vol. 24, pp. 16-25, 2007.

[40] X. Ji, H. Zha, J. J. Metzner, and G. Kesidis, "Dynamic cluster structure for object detection and tracking in wireless ad-hoc sensor networks," in *Communications, 2004 IEEE International Conference on*, 2004, pp. 3807-3811.

[41] A. Boukerche, *Algorithms and Protocols for Wireless Sensor Networks*: Wiley, 2008.

[42] C. Farah, F. Schwaner, A. Abedi, and M. Worboys, "Distributed homology algorithm to detect topological events via wireless sensor networks," *Wireless Sensor Systems, IET,* vol. 1, pp. 151-160, 2011.

[43] J. H. Yoo, W. Kim, and H. J. Kim, "Event-driven Gaussian process for object localization in wireless sensor networks," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 2011, pp. 2790-2795.

[44] G. Wittenburg, N. Dziengel, S. Adler, Z. Kasmi, M. Ziegert, and J. Schiller, "Cooperative event detection in wireless sensor networks," *Communications Magazine, IEEE,* vol. 50, pp. 124-131, 2012.

[45] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. d. R. Millán, *et al.*, "The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognition Letters,* vol. 34, pp. 2033-2042, 2013.

[46] K. Altun, B. Barshan, and O. Tunçel, "Comparative study on classifying human activities with miniature inertial and magnetic sensors," *Pattern Recognition,* vol. 43, pp. 3605-3620, 2010.

[47] W.-R. Chang, H.-T. Lin, and Z.-Z. Cheng, "CODA: a continuous object detection and tracking algorithm for wireless ad hoc sensor networks," in *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*, 2008, pp. 168-174.

[48] A. Senior, Y. A. Sekercioglu, and A. I. Khan, "A polymorphic recogniser for distributed intelligence in wireless sensor networks," in *Computer and Information Sciences (ICCOINS), 2014 International Conference on*, 2014, pp. 1-5.

[49] M. Moshtaghi, "Anomaly detection in heterogeneous sensed data," 2013.

[50] V. Ç. Güngör and G. P. Hancke, *Industrial Wireless Sensor Networks: Applications, Protocols, and Standards*: Taylor & Francis, 2013.

[51] Y. Zhang, N. Meratnia, and P. Havinga, "Outlier detection techniques for wireless sensor networks: A survey," *Communications Surveys & Tutorials, IEEE,* vol. 12, pp. 159-170, 2010.

[52] M. Ilyas and I. Mahgoub, *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*: Taylor & Francis, 2004.

[53] A. S. Tanenbaum, G. Chandana, and B. Crispo, "Taking Sensor Networks from the Lab to the Jungle," *Computer,* vol. 39, pp. 98-100, 2006.

[54] Y. Yuan, Z. Yang, M. Chen, and J. He, "A survey on information processing technologies in wireless sensor networks," *International*

Journal of Ad Hoc and Ubiquitous Computing, vol. 1, pp. 103-109, 2006.

[55] B. Krishnamachari, D. Estrin, and S. Wicker, "Modelling data-centric routing in wireless sensor networks," in *IEEE infocom*, 2002, pp. 39-44.

[56] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *Wireless Communications, IEEE,* vol. 11, pp. 6-28, 2004.

[57] A. Bharathidasan and V. A. S. Ponduru, "Sensor networks: An overview," *paper, University of California, Davis, CA (August 2002),* 2002.

[58] K. Romer and F. Mattern, "The design space of wireless sensor networks," *Wireless Communications, IEEE,* vol. 11, pp. 54-61, 2004.

[59] D. Estrin, D. Culler, K. Pister, and G. Sukhatme, "Connecting the physical world with pervasive networks," *Pervasive Computing, IEEE,* vol. 1, pp. 59-69, 2002.

[60] A. Khan and P. Mihailescu, "Parallel pattern recognition computations within a wireless sensor network," *Pattern Recognition,* vol. 1, pp. 777-780, 2004.

[61] C. Irniger and H. Bunke, "Theoretical analysis and experimental comparison of graph matching algorithms for database filtering," in *Graph Based Representations in Pattern Recognition,* ed: Springer, 2003, pp. 118-129.

[62] M. R. Garey and D. S. Johnson, *Computers and intractability* vol. 174: freeman New York, 1979.

[63] A. Khan, "A peer-to-peer associative memory network for intelligent information systems," *ACIS 2002 Proceedings*, p. 6, 2002.

[64] A. Woo and D. E. Culler, "A transmission control scheme for media access in sensor networks," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, 2001, pp. 221-235.

[65] H. Zhang, A. Arora, Y.-r. Choi, and M. G. Gouda, "Reliable bursty convergecast in wireless sensor networks," *Computer Communications*, vol. 30, pp. 2560-2576, 2007.

[66] W. Zhang and G. Cao, "DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks," *Wireless Communications, IEEE Transactions on*, vol. 3, pp. 1689-1701, 2004.

[67] W. Zhang and G. Cao, "Optimizing tree reconfiguration for mobile target tracking in sensor networks," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 2004, pp. 2434-2445.

[68] Y. Xu, J. Winter, and W. C. Lee, "Prediction-based strategies for energy saving in object tracking sensor networks," in *the 2004 IEEE international conference on mobile data management (MDM'04)*, 2004.

[69] Y. Xu, J. Winter, and W. C. Lee, "Dual prediction-based reporting for object tracking sensor networks," in *the first annual international*

conference on mobile and ubiquitous systems: networking and services (MobiQuitous'04), 2004.

[70] W. P. Chen, J. C. Hou, and L. Sha, "Dynamic clustering for acoustic target tracking in wireless sensor networks," *Mobile Computing, IEEE Transactions on,* vol. 3, pp. 258-271, 2004.

[71] K. M. Chandy, "Event-driven applications: Costs, benefits and design approaches," *Gartner Application Integration and Web Services Summit,* vol. 2006, 2006.

[72] G. Johansson, "Configurations in event perception," *Perceiving events and objects*, pp. 29-122, 1994.

[73] S. Ortmann, M. Maaser, and P. Langendoerfer, "Adaptive pruning of event decision trees for energy efficient collaboration in event-driven WSN," in *Mobile and Ubiquitous Systems: Networking & Services, MobiQuitous, 2009. MobiQuitous' 09. 6th Annual International*, 2009, pp. 1-11.

[74] M. Zoumboulakis and G. Roussos, "Complex event detection in extremely resource-constrained wireless sensor networks," *Mobile Networks and Applications,* vol. 16, pp. 194-213, 2011.

[75] S. Watanabe, *Pattern recognition: human and mechanical*: Wiley, 1985.

[76] B. Catania, A. Maddalena, and M. Mazza, "Psycho: A prototype system for pattern management," in *Proceedings of the 31st international conference on Very large data bases*, 2005, pp. 1346-1349.

[77] V. Cantoni, L. Lombardi, and P. Lombardi, "Challenges for data mining in distributed sensor networks," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, 2006, pp. 1000-1007.

[78] D. Chen and P. K. Varshney, "QoS Support in Wireless Sensor Networks: A Survey," in *International Conference on Wireless Networks*, 2004, pp. 1-7.

[79] X. Luo, M. Dong, and Y. Huang, "On distributed fault-tolerant detection in wireless sensor networks," *Computers, IEEE Transactions on*, vol. 55, pp. 58-70, 2006.

[80] M. Horton and J. Suh, "A vision for wireless sensor networks," in *Microwave Symposium Digest, 2005 IEEE MTT-S International*, 2005, p. 4 pp.

[81] P. Santi, "Topology control in wireless ad hoc and sensor networks," *ACM computing surveys (CSUR)*, vol. 37, pp. 164-194, 2005.

[82] R. Adler, M. Flanigan, J. Huang, R. Kling, N. Kushalnagar, L. Nachman, *et al.*, "Intel mote 2: an advanced platform for demanding sensor network applications," 2005, pp. 298-298.

[83] L. Gu and J. A. Stankovic, "t-kernel: providing reliable OS support to wireless sensor networks," in *the 4th international conference on Embedded networked sensor systems*, 2006, pp. 1-14.

[84] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *Circuits and Systems Magazine, IEEE*, vol. 5, pp. 19-31, 2005.

[85] J. Hill, M. Horton, R. Kling, and L. Krishnamurthy, "The platforms enabling wireless sensor networks," *Communications of the ACM*, vol. 47, pp. 41-46, 2004.

[86] I. M. M. E. Emary and S. Ramakrishnan, *Wireless Sensor Networks: From Theory to Applications*: Taylor & Francis, 2013.

[87] J. P. Lynch and K. J. Loh, "A summary review of wireless sensors and sensor networks for structural health monitoring," *Shock and Vibration Digest*, vol. 38, pp. 91-130, 2006.

[88] A. Iyer, S. S. Kulkarni, V. Mhatre, and C. P. Rosenberg, "A Taxonomy-based Approach to Design of Large-scale Sensor Networks," in *Wireless Sensor Networks and Applications*, ed: Springer, 2008, pp. 3-33.

[89] A. S. Tanenbaum and D. Wetherall, *Computer networks*, 5th ed. Boston: Pearson Prentice Hall, 2011.

[90] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks*, vol. 52, pp. 2292-2330, 2008.

[91] W. Charfi, M. Masmoudi, and F. Derbel, "A layered model for wireless sensor networks," in *Systems, Signals and Devices, 2009. SSD'09. 6th International Multi-Conference on*, 2009, pp. 1-5.

[92] C. Wang, K. Sohraby, B. Li, M. Daneshmand, and Y. Hu, "A survey of transport protocols for wireless sensor networks," *Ieee Network*, vol. 20, pp. 34-40, 2006.

[93] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung, "MAC essentials for wireless sensor networks," *Communications Surveys & Tutorials, IEEE,* vol. 12, pp. 222-248, 2010.

[94] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The evolution of MAC protocols in wireless sensor networks: A survey," *Communications Surveys & Tutorials, IEEE,* vol. 15, pp. 101-120, 2013.

[95] G. Wittenburg, N. Dziengel, C. Wartenburger, and J. Schiller, "A system for distributed event detection in wireless sensor networks," 2010, pp. 94-104.

[96] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*: Wiley, 2001.

[97] J. B. Predd, S. B. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *Signal Processing Magazine, IEEE,* vol. 23, pp. 56-69, 2006.

[98] E. F. Nakamura, A. A. Loureiro, and A. C. Frery, "Information fusion for wireless sensor networks: Methods, models, and classifications," *ACM Computing Surveys (CSUR),* vol. 39, p. 9, 2007.

[99] Y. Kim, J. Kang, D. Kim, E. Kim, P. Chong, and S. Seo, "Design of a fence surveillance system based on wireless sensor networks," 2008, pp. 1-7.

[100] S. Jabbar, A. E. Butt, and A. Minhas, "Threshold based load balancing protocol for energy efficient routing in WSN," in *Advanced*

Communication Technology (ICACT), 2011 13th International Conference on, 2011, pp. 196-201.

[101] T. Bokareva, W. Hu, S. Kanhere, B. Ristic, N. Gordon, T. Bessell, *et al.*, "Wireless sensor networks for battlefield surveillance," in *Proceedings of the land warfare conference*, 2006.

[102] R. Niu, M. Moore, and D. Klamer, "Decision fusion in a wireless sensor network with a large number of sensors," 2004.

[103] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, pp. 4-37, 2000.

[104] H. Bunke and A. Sanfeliu, *Syntactic and structural pattern recognition: theory and applications*: World Scientific, 1990.

[105] E. Elnahrawy and B. Nath, "Context-aware sensors," in *Wireless Sensor Networks*, ed: Springer, 2004, pp. 77-93.

[106] W. H. Wu, A. A. Bui, M. A. Batalin, L. K. Au, J. D. Binney, and W. J. Kaiser, "MEDIC: Medical embedded device for individualized care," *Artificial Intelligence in Medicine*, vol. 42, pp. 137-152, 2008.

[107] S. Mittal, A. Aggarwal, and S. Maskara, "Application of Bayesian Belief Networks for context extraction from wireless sensors data," in *Advanced Communication Technology (ICACT), 2012 14th International Conference on*, 2012, pp. 410-415.

[108] X. Sun and E. J. Coyle, "Low-complexity algorithms for event detection in wireless sensor networks," *Selected Areas in Communications, IEEE Journal on,* vol. 28, pp. 1138-1148, 2010.

[109] J. B. Predd, S. R. Kulkarni, and H. V. Poor, *Distributed learning in wireless sensor networks*: John Wiley & Sons: Chichester, UK, 2007.

[110] S. R. Kulkarni, G. Lugosi, and S. S. Venkatesh, "Learning pattern classification-a survey," *Information Theory, IEEE Transactions on,* vol. 44, pp. 2178-2206, 1998.

[111] L. Jiang, Z. Cai, D. Wang, and S. Jiang, "Survey of Improving K-Nearest-Neighbor for Classification," in *FSKD (1)*, 2007, pp. 679-683.

[112] D. Li, K. Wong, Y. H. Hu, and A. Sayeed, "Detection, classification and tracking of targets in distributed sensor networks," *IEEE signal processing magazine,* vol. 19, pp. 17-29, 2002.

[113] M. F. Duarte and Y. Hen Hu, "Vehicle classification in distributed sensor networks," *Journal of Parallel and Distributed Computing,* vol. 64, pp. 826-838, 2004.

[114] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR),* vol. 41, p. 15, 2009.

[115] K. Zhang, S. Shi, H. Gao, and J. Li, "Unsupervised outlier detection in sensor networks using aggregation tree," in *Advanced Data Mining and Applications,* ed: Springer, 2007, pp. 158-169.

[116] D. Li, K. D. Wong, Y. H. Hu, and A. M. Sayeed, "Detection, classification, and tracking of targets," *Signal Processing Magazine, IEEE,* vol. 19, pp. 17-29, 2002.

[117] R. Rengaswamy and V. Venkatasubramanian, "A syntactic pattern-recognition approach for process monitoring and fault diagnosis," *Engineering Applications of Artificial Intelligence,* vol. 8, pp. 35-51, 1995.

[118] K.-S. Fu and B. K. Bhargava, "Tree systems for syntactic pattern recognition," *Computers, IEEE Transactions on,* vol. 100, pp. 1087-1099, 1973.

[119] V. Latha, C. Subramaniam, and S. Shanmugavel, "Fault tolerant wireless sensor network using case based reasoning with semantic tracking," in *Proceedings of the WSEAS International Conference on Communications*, 2011, pp. 240-246.

[120] R. Hamilton, R. Pringle, and P. Grant, "Syntactic techniques for pattern recognition on sampled data signals," *IEE Proceedings E (Computers and Digital Techniques),* vol. 139, pp. 156-164, 1992.

[121] M. Marin-Perianu, C. Lombriser, O. Amft, P. Havinga, and G. Tröster, "Distributed activity recognition with fuzzy-enabled wireless sensor networks," in *Distributed Computing in Sensor Systems*, ed: Springer, 2008, pp. 296-313.

[122] J. S. R. Jang and C. T. Sun, "Neuro-fuzzy modeling and control," *Proceedings of the IEEE,* vol. 83, pp. 378-406, 1995.

[123] M. Zarei, A. M. Rahmani, A. Sasan, and M. Teshnehlab, "Fuzzy based trust estimation for congestion control in wireless sensor networks," in *Intelligent Networking and Collaborative Systems, 2009. INCOS'09. International Conference on*, 2009, pp. 233-236.

[124] X. Feng, Z. Gao, M. Yang, and S. Xiong, "Fuzzy distance measuring based on RSSI in Wireless Sensor Network," in *Intelligent System and Knowledge Engineering, 2008. ISKE 2008. 3rd International Conference on*, 2008, pp. 395-400.

[125] M. Bahrepour, N. Meratnia, M. Poel, Z. Taghikhaki, and P. J. Havinga, "Distributed event detection in wireless sensor networks for disaster management," in *Intelligent Networking and Collaborative Systems (INCOS), 2010 2nd International Conference on*, 2010, pp. 507-512.

[126] F. Oldewurtel and P. Mahonen, "Neural wireless sensor networks," in *Systems and Networks Communications, 2006. ICSNC'06. International Conference on*, 2006, pp. 28-28.

[127] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, pp. 273-297, 1995.

[128] X. Wang, S. Wang, D. Bi, L. Ding, and J. Ma, "Collaborative Peer-to-Peer Training and Target Classification in Wireless Sensor Networks," in *Future Generation Communication and Networking (FGCN 2007)* 2007, pp. 208-213.

[129] D. A. Tran and T. Nguyen, "Localization in wireless sensor networks based on support vector machines," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 19, pp. 981-994, 2008.

[130] R. Abu Sajana, R. Subramanian, P. V. Kumar, S. Krishnan, B. Amrutur, J. Sebastian*, et al.*, "A low-complexity algorithm for intrusion detection in a PIR-based Wireless Sensor Network," in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on*, 2009, pp. 337-342.

[131] D. Wang, "Pattern recognition: Neural networks in perspective," *IEEE Intelligent Systems,* vol. 8, pp. 52-60, 1993.

[132] M. Hattori and M. Hagiwara, "Neural associative memory for intelligent information processing," in *Knowledge-Based Intelligent Electronic Systems, 1998. Proceedings KES'98. 1998 Second International Conference on*, 1998, pp. 377-386.

[133] J.-P. Nadal, "Study of a growth algorithm for a feedforward network," *International journal of neural systems,* vol. 1, pp. 55-59, 1989.

[134] P. Guan and X. Li, "Minimizing distribution cost of distributed neural networks in wireless sensor networks," in *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, 2007, pp. 790-794.

[135] S. i. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: four layers versus three," *Neural Networks, IEEE Transactions on,* vol. 8, pp. 251-255, 1997.

[136] A. Awad, T. Frunzke, and F. Dressler, "Adaptive distance estimation and localization in WSN using RSSI measures," in *Digital System Design Architectures, Methods and Tools, 2007. DSD 2007. 10th Euromicro Conference on*, 2007, pp. 471-478.

[137] R. Rajkamal and P. V. Ranjan, "Packet classification for network processors in wsn traffic using ann," in *Industrial Informatics, 2008. INDIN 2008. 6th IEEE International Conference on*, 2008, pp. 707-710.

[138] X. Wang and S. Wang, "Collaborative signal processing for target tracking in distributed wireless sensor networks," *Journal of Parallel and Distributed Computing,* vol. 67, pp. 501-515, 2007.

[139] M. Ishizuka and M. Aida, "Achieving power-law placement in wireless sensor networks," in *Autonomous Decentralized Systems, 2005. ISADS 2005. Proceedings*, 2005, pp. 661-666.

[140] G. Bartfai and R. White, "Adaptive resonance theory-based modular networks for incremental learning of hierarchical clusterings," *Connection Science,* vol. 9, pp. 87-112, 1997.

[141] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural networks,* vol. 4, pp. 759-771, 1991.

[142] Y. Li and L. E. Parker, "Detecting and monitoring time-related abnormal events using a wireless sensor network and mobile robot," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 3292-3298.

[143] M. Kumar, S. Verma, and P. Singh, "Data clustering in sensor networks using ART," in *Wireless Communication and Sensor Networks, 2008. WCSN 2008. Fourth International Conference on*, 2008, pp. 51-56.

[144] M. Kumar, S. Verma, and P. P. Singh, "Clustering approach to data aggregation in wireless sensor networks," in *Networks, 2008. ICON 2008. 16th IEEE International Conference on*, 2008, pp. 1-6.

[145] A. Kulakov and D. Davcev, "Tracking of unusual events in wireless sensor networks based on artificial neural-networks algorithms," in *Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on*, 2005, pp. 534-539.

[146] J. J. Hopfield and D. W. Tank, "“Neural” computation of decisions in optimization problems," *Biological cybernetics,* vol. 52, pp. 141-152, 1985.

[147] G. Massini, "Hopfield neural network," *Substance use & misuse,* vol. 33, pp. 481-488, 1998.

[148] J. H. Kim, S. H. Yoon, Y. H. Kim, E. H. Park, C. Ntuen, K. Sohn*, et al.*, "An efficient matching algorithm by a hybrid Hopfield network for object recognition," 1992, pp. 2888-2892 vol. 6.

[149] J.-Z. Chen, C. C. Yu, M. T. Hsieh, and Y. N. Chung, "Employing CHNN to develop a data refining algorithm for wireless sensor networks," in *Computer Science and Information Engineering, 2009 WRI World Congress on*, 2009, pp. 24-31.

[150] W. Xue, J. Aiguo, and W. Sheng, "Mobile agent based moving target methods in wireless sensor networks," in *Communications and Information Technology, 2005. ISCIT 2005. IEEE International Symposium on,* 2005, pp. 22-26.

[151] J. Levendovszky, K. Tornai, G. Treplán, and A. Oláh, "Novel load balancing algorithms ensuring uniform packet loss probabilities for WSN," in *Vehicular technology conference (VTC Spring), 2011 IEEE 73rd*, 2011, pp. 1-5.

[152] D. Tisza, P. Vizi, J. Levendovszky, and A. Oláh, "Multicast Routing in Wireless Sensor Networks with Incomplete Information," in *Wireless Conference 2011-Sustainable Wireless Technologies (European Wireless), 11th European*, 2011, pp. 1-5.

[153] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE,* vol. 78, pp. 1464-1480, 1990.

[154] T. Kohonen, "The self-organizing map," *Neurocomputing,* vol. 21, pp. 1-6, 1998.

[155] S. Wang, "Application of self-organising maps for data mining with incomplete data sets," *Neural Computing & Applications,* vol. 12, pp. 42-48, 2003.

[156] G. Giorgetti, S. K. Gupta, and G. Manes, "Wireless localization using self-organizing maps," in *Proceedings of the 6th international conference on Information processing in sensor networks*, 2007, pp. 293-302.

[157] O. A. Postolache, P. Girao, J. D. Pereira, and H. M. G. Ramos, "Self-organizing maps application in a remote water quality monitoring system," *IEEE transactions on instrumentation and measurement,* vol. 54, pp. 322-329, 2005.

[158] A. I. Moustapha and R. R. Selmic, "Wireless sensor network modeling using modified recurrent neural networks: Application to fault detection," *Instrumentation and Measurement, IEEE Transactions on,* vol. 57, pp. 981-988, 2008.

[159] J. T. Connor, R. D. Martin, and L. E. Atlas, "Recurrent neural networks and robust time series prediction," *Neural Networks, IEEE Transactions on,* vol. 5, pp. 240-254, 1994.

[160] A. D. J. Raju and S. Manohar, "Recurrent neural network for faulty data identification in smart grid," in *Recent Advancements in Electrical, Electronics and Control Engineering (ICONRAEeCE), 2011 International Conference on*, 2011, pp. 303-308.

[161] J. W. Barron, A. I. Moustapha, and R. R. Selmic, "Real-time implementation of fault detection in wireless sensor networks using neural networks," in *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*, 2008, pp. 378-383.

[162] Z. A. Baig, M. Baqer, and A. I. Khan, "A pattern recognition scheme for distributed denial of service (ddos) attacks in wireless sensor networks," *Pattern Recognition,* vol. 3, pp. 1050-1054, 2006.

[163] M. Baqer and A. Khan, "Energy-efficient pattern recognition approach for wireless sensor networks," pp. 509-514.

[164] B. B. Nasution and A. I. Khan, "A hierarchical graph neuron scheme for real-time pattern recognition," *Neural Networks, IEEE Transactions on,* vol. 19, pp. 212-229, 2008.

[165] A. Khan, M. Isreb, and R. Spindler, "A parallel distributed application of the wireless sensor network," in *High Performance Computing and Grid in Asia Pacific Region, 2004. Proceedings. Seventh International Conference on*, 2004, pp. 81-88.

[166] M. Baqer, A. Khan, and Z. Baig, "Implementing a graph neuron array for pattern recognition within unstructured wireless sensor networks," *Embedded and Ubiquitous Computing,* pp. 208-217, 2005.

[167] A. Khan and A. Amin, "One shot associative memory method for distorted pattern recognition," *AI 2007: Advances in Artificial Intelligence,* pp. 705-709, 2007.

[168] W. S. Hortos, "Unsupervised algorithms for intrusion detection and identification in wireless ad hoc sensor networks," in *SPIE Defense, Security, and Sensing*, 2009, pp. 73520J-73520J-12.

[169] A. Giridhar and P. Kumar, "Toward a theory of in-network computation in wireless sensor networks," *Communications Magazine, IEEE,* vol. 44, pp. 98-107, 2006.

[170] A. Boulis, S. Ganeriwal, and M. B. Srivastava, "Aggregation in sensor networks: An energy-accuracy trade-off," *Ad hoc networks,* vol. 1, pp. 317-331, 2003.

[171] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," 2006, pp. 278-287.

[172] J. Gao, L. Guibas, N. Milosavljevic, and J. Hershberger, "Sparse data aggregation in sensor networks," 2007, pp. 430-439.

[173] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *ACM SIGOPS Operating Systems Review,* vol. 36, pp. 131-146, 2002.

[174] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler, "Supporting aggregate queries over ad-hoc wireless sensor networks," 2002.

[175] C. Farah, C. Zhong, M. Worboys, and S. Nittel, "Detecting topological change using a wireless sensor network," in *Geographic Information Science,* ed: Springer, 2008, pp. 55-69.

[176] B. Sun, L. Osborne, Y. Xiao, and S. Guizani, "Intrusion detection techniques in mobile ad hoc and wireless sensor networks," *Wireless Communications, IEEE,* vol. 14, pp. 56-63, 2007.

[177] B. Son, Y.-s. Her, and J. Kim, "A design and implementation of forest-fires surveillance system based on wireless sensor networks for South Korea mountains," *International Journal of Computer Science and Network Security (IJCSNS),* vol. 6, pp. 124-130, 2006.

[178] W. M. Alfehaid, A. I. Khan, and A. H. M. Amin, "A combined pattern recognition scheme with genetic algorithms for robot guidance using Wireless Sensor Networks," in *Control Automation Robotics & Vision (ICARCV), 2012 12th International Conference on*, 2012, pp. 759-764.

[179] M. Al-Naeem and A. I. Khan, "MOHA: A Novel Target Recognition Scheme for WSNs," in *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, ed: Springer, 2010, pp. 364-365.

[180] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278-2324, 1998.

[181] J. Fan, W. Xu, Y. Wu, and Y. Gong, "Human tracking using convolutional neural networks," *Neural Networks, IEEE Transactions on*, vol. 21, pp. 1610-1623, 2010.

[182] C. Nebauer, "Evaluation of convolutional neural networks for visual recognition," *Neural Networks, IEEE Transactions on*, vol. 9, pp. 685-696, 1998.

[183] R. Bekkerman, M. Bilenko, and J. Langford, *Scaling up Machine Learning: Parallel and Distributed Approaches*: Cambridge University Press, 2011.

[184] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[185] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, and E. Culurciello, "Hardware accelerated convolutional neural networks for synthetic vision systems," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, 2010, pp. 257-260.

[186] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*, 2011, pp. 2809-2813.

[187] Y. Xu, H. Yao, L. Gong, M. Zhu, and R. K. Teng, "A FPGA real-time stereo vision system with luminance control and projected pattern," in *ASIC (ASICON), 2013 IEEE 10th International Conference on*, 2013, pp. 1-4.

[188] K. Siddiqi and B. B. Kimia, "A shock grammar for recognition," in *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on*, 1996, pp. 507-513.

[189] T. B. Sebastian, P. N. Klein, and B. B. Kimia, "Recognition of shapes by editing their shock graphs," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, pp. 550-571, 2004.

[190] D. Arathorn, "Recognition under transformation using superposition ordering property," *Electronics Letters*, vol. 37, pp. 164-166, 2001.

[191] T. Gedeon and D. Arathorn, "Convergence of map seeking circuits," *Journal of Mathematical Imaging and Vision,* vol. 29, pp. 235-248, 2007.

[192] H. S. Ahmed and B. M. Faouzi, "Classifications of images for the recognition of people’s behaviors by SIFT and SVM," *International Journal of Innovative Research in Advanced Engineering (IJIRAE),* vol. 2, 2015.

[193] T. Gadgi and A. Priyadarshi, "A Survey Paper on F-SIFT for Object and Copy Detection," *International Journal of Science and Research (IJSR),* vol. 3, 2014.

[194] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on,* 1999, pp. 1150-1157.

[195] G. X. Ritter, P. Sussner, and J. L. Diza-de-Leon, "Morphological associative memories," *Neural Networks, IEEE Transactions on,* vol. 9, pp. 281-293, 1998.

[196] K. M. Iftekharuddin, "Transformation invariant on-line target recognition," *Neural Networks, IEEE Transactions on,* vol. 22, pp. 906-918, 2011.

[197] W. Deng, J. Hu, J. Lu, and J. Guo, "Transform-Invariant PCA: A Unified Approach to Fully Automatic FaceAlignment, Representation, and Recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 36, pp. 1275-1284, 2014.

[198] S. Zhang, M. Zhang, R. He, and Z. Sun, "Transform-invariant dictionary learning for face recognition," in *Image Processing (ICIP), 2014 IEEE International Conference on*, 2014, pp. 348-352.

[199] B. Shen, B.-D. Liu, and J. P. Allebach, "TISVM: Large margin classifier for misaligned image classification," in *Image Processing (ICIP), 2014 IEEE International Conference on*, 2014, pp. 4251-4255.

[200] W. Alfehaid, "Distributed pattern recognition schemes for wireless sensor networks," PhD, Clayton School of Information Technology, Monash University, 2013.

[201] E. B. Goldstein, *Encyclopedia of Perception*: SAGE Publications, 2010.

[202] R. Nevatia, *Machine perception*: Prentice-Hall, 1982.

[203] S. Bhardwaj and A. Mittal, "A Survey on Various Edge Detector Techniques," *Procedia Technology,* vol. 4, pp. 220-226, 2012.

[204] Z. Hussain and D. Agarwal, "A Comparative Analysis of Edge Detection Techniques used in Flame Image Processing," *International Journal of Advance Research In Science And Engineering (IJARSE),* vol. 4, March 2015.

[205] K. Kaur and S. Malhotra, "A Survey on Edge Detection Using Different Techniques," *International Journal of Application or Innovation in Engineering & Management (IJAIEM),* vol. 2, April 2013.

[206] M. A. Oskoei and H. Hu, "A survey on edge detection methods," *University of Essex, UK,* 2010.

[207] S. Savant, "A Review on Edge Detection Techniques for Image Segmentation," *International Journal of Computer Science & Information Technologies,* vol. 5, 2014.

[208] M. Abo-Zahhad, R. R. Gharieb, S. M. Ahmed, and A. A. E.-B. Donkol, "Edge Detection with a Preprocessing Approach," *Journal of Signal and Information Processing,* vol. 5, p. 123, 2014.

[209] R. Jayakumar and B. Suresh, "A Review on Edge Detection Methods and Techniques," *International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE),* vol. 3, April 2014.

[210] R. Swarnalakshmi, "A Survey on Edge Detection Techniques using Different Types of Digital Images," *International Journal of Computer Science and Mobile Computing,* vol. 3, pp. 694-699, July 2014.

[211] R. Maini, "Analysis and development of image edge detection techniques," University College of Engineering, Punjabi University, 2012.

[212] M. suman and M. pawan, "A Survey on Various Methods of Edge Detection," *International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE),* vol. 4, May 2014.

[213] D. Ziou and S. Tabbone, "Edge detection techniques-an overview," *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii,* vol. 8, pp. 537-559, 1998.

[214] M. Nosrati, R. Karimi, M. Hariri, and K. Malekian, "Edge Detection Techniques in Processing Digital Images: Investigation of Canny Algorithm and Gabor Method," *World Applied Programming, ISSN,* pp. 2222-2510, 2013.

[215] R. K. Nigam, N. Waghmare, and A. Gupta, "Evaluation and Analysis of Different Type of Edge Detection Techniques on Cartridge Case Image," *International Journal on Recent and Innovation Trends in Computing and Communication,* vol. 2, pp. 3258 – 3261, October 2014.

[216] R. Maini and H. Aggarwal, "Study and comparison of various image edge detection techniques," *International journal of image processing (IJIP),* vol. 3, pp. 1-11, 2009.

[217] M. Wen and C. Zhong, "Application of Sobel Algorithm in Edge Detection of Images," *China High-tech Enterprise,* pp. 57-62, 2008.

[218] P. Rani and P. Tanwar, "A Nobel Hybrid Approach for Edge Detection," *International Journal of Computer Science & Engineering Survey (IJCSES),* vol. 4, April 2013.

[219] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences,* vol. 207, pp. 187-217, 1980.

[220] A. L. Yuille and T. A. Poggio, "Scaling theorems for zero crossings," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* pp. 15-25, 1986.

[221] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* pp. 679-698, 1986.

[222] J. F. Canny, "Finding edges and lines in images," *Massachusetts Inst. of Tech. Report,* vol. 1, 1983.

[223] M. Basu, "Gaussian-based edge-detection methods-a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C,* vol. 32, pp. 252-260, 2002.

[224] A. Goshtasby, "On edge focusing," *Image and Vision Computing,* vol. 12, pp. 247-256, 1994.

[225] M. K. Rashmi and R. Saxena, "Algorithm and technique on various edge detection: A survey," *Signal & Image Processing: An International Journal (SIPIJ) Vol,* vol. 4, 2013.

[226] C. J. Mallery and M. Medidi, "Robust edge detection in wireless sensor networks," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE,* 2008, pp. 1-5.

[227] K. Ren, K. Zeng, and W. Lou, "Fault-tolerant Event Boundary Detection in Wireless Sensor Networks," in *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE,* 2006, pp. 1-5.

[228] K. K. Chintalapudi and R. Govindan, "Localized edge detection in sensor fields," *Ad Hoc Networks,* vol. 1, pp. 273-291, 2003.

[229] R. Nowak and U. Mitra, "Boundary estimation in sensor networks: Theory and methods," in *Information Processing in Sensor Networks*, 2003, pp. 80-95.

[230] P.-K. Liao, M.-K. Chang, and C.-C. Kuo, "Distributed edge detection with composite hypothesis test in wireless sensor networks," in *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, 2004, pp. 129-133.

[231] T. Banerjee, D. Wang, B. Xie, and D. P. Agrawal, "PERD: Polynomial-based Event Region Detection in Wireless Sensor Networks," in *Communications, 2007. ICC'07. IEEE International Conference on*, 2007, pp. 3307-3312.

[232] T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Distributed deviation detection in sensor networks," *ACM SIGMOD Record*, vol. 32, pp. 77-82, 2003.

[233] A. Marcus and O. Marques, "An eye on visual sensor networks," *Potentials, IEEE*, vol. 31, pp. 38-43, 2012.

[234] B. Tavli, K. Bicakci, R. Zilan, and J. M. Barcelo-Ordinas, "A survey of visual sensor network platforms," *Multimedia Tools and Applications*, vol. 60, pp. 689-726, 2012.

[235] T. Teixeira, D. Lymberopoulos, E. Culurciello, Y. Aloimonos, and A. Savvides, "A lightweight camera sensor network operating on symbolic information," in *Proceedings of the 1st Workshop on Distributed Smart Cameras*, 2006.

[236] S. Soro and W. Heinzelman, "A survey of visual sensor networks," *Advances in Multimedia*, vol. 2009, 2009.

[237] R. Puri, A. Majumdar, P. Ishwar, and K. Ramchandran, "Distributed video coding in wireless sensor networks," *Signal Processing Magazine, IEEE,* vol. 23, pp. 94-106, 2006.

[238] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM,* vol. 43, pp. 51-58, 2000.

[239] Z. He and D. Wu, "Resource allocation and performance analysis of wireless video sensors," *Circuits and Systems for Video Technology, IEEE Transactions on,* vol. 16, pp. 590-599, 2006.

[240] P. Chen, P. Ahammad, C. Boyer, S.-I. Huang, L. Lin, E. Lobaton*, et al.*, "CITRIC: A low-bandwidth wireless camera network platform," in *Distributed smart cameras, 2008. ICDSC 2008. Second ACM/IEEE international conference on,* 2008, pp. 1-10.

[241] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* vol. 34, pp. 334-352, 2004.

[242] A. Kerhet, M. Magno, F. Leonardi, A. Boni, and L. Benini, "A low-power wireless video sensor node for distributed object detection," *Journal of real-time image processing,* vol. 2, pp. 331-342, 2007.

[243] M. Rahimi, R. Baer, O. I. Iroezi, J. C. Garcia, J. Warrior, D. Estrin*, et al.*, "Cyclops: in situ image sensing and interpretation in wireless sensor

networks," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, 2005, pp. 192-204.

[244] A. Rowe, D. Goel, and R. Rajkumar, "Firefly mosaic: A vision-enabled wireless sensor networking system," in *Real-time systems symposium, 2007. RTSS 2007. 28th IEEE international*, 2007, pp. 459-468.

[245] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan, "MeshEye: a hybrid-resolution smart camera mote for applications in distributed intelligent surveillance," in *Proceedings of the 6th international conference on Information processing in sensor networks*, 2007, pp. 360-369.

[246] W.-c. Feng, E. Kaiser, W. C. Feng, and M. L. Baillif, "Panoptes: scalable low-power video sensor networking technologies," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP),* vol. 1, pp. 151-167, 2005.

[247] J. Boice, X. Lu, C. Margi, G. Stanek, G. Zhang, R. Manduchi, *et al.*, "Meerkats: A power-aware, self-managing wireless camera network for wide area monitoring," in *Proc. Workshop on Distributed Smart Cameras*, 2006.

[248] T. Teixeira, E. Culurciello, J. H. Park, D. Lymberopoulos, A. Barton-Sweeney, and A. Savvides, "Address-event imagers for sensor networks: evaluation and modeling," in *Proceedings of the 5th international conference on Information processing in sensor networks*, 2006, pp. 458-466.

[249] M. Zhang and W. Cai, "Vision mesh: a novel video sensor networks platform for water conservancy engineering," in *Computer science and information technology (ICCSIT), 2010 3rd IEEE international conference on*, 2010, pp. 106-109.

[250] P. Kulkarni, D. Ganesan, P. Shenoy, and Q. Lu, "SensEye: a multi-tier camera sensor network," in *Proceedings of the 13th annual ACM international conference on Multimedia*, 2005, pp. 229-238.

[251] M. Wu and C. W. Chen, "Collaborative image coding and transmission over wireless sensor networks," *EURASIP Journal on Applied Signal Processing*, vol. 2007, pp. 223-223, 2007.

[252] U. M. Erdem and S. Sclaroff, "Optimal placement of cameras in floorplans to satisfy task requirements and cost constraints," in *OMNIVIS workshop*, 2004.

[253] S. Kundu and N. Das, "Event boundary detection and gathering in wireless sensor networks," in *Applications and Innovations in Mobile Computing (AIMoC), 2015*, 2015, pp. 62-67.

[254] M. Al-Naeem and A. I. Khan, "A novel target recognition scheme for WSNs," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, 2012, pp. 1-6.

[255] B. A. Olshausen and D. J. Field, "Sparse coding of sensory inputs," *Current opinion in neurobiology*, vol. 14, pp. 481-487, 2004.

[256] H. Zulkafli, "Object detection by using edge segmentation techniques," School of Electronics and Computer Science, University of Malaya, 2009.

[257] D. Devaguptapu and B. Krishnamachari, "Applications of localized image processing techniques in wireless sensor networks," in *AeroSense 2003*, 2003, pp. 247-256.

[258] A. R. Webb, *Statistical Pattern Recognition*: Wiley, 2003.

[259] N. C. Jones and P. Pevzner, *An Introduction to Bioinformatics Algorithms*: London, 2004.

[260] A. H. M. Amin, R. Mahmood, and A. I. Khan, "Analysis of Pattern Recognition Algorithms Using Associative Memory Approach: A Comparative Study between the Hopfield Network and Distributed Hierarchical Graph Neuron (DHGN)," in *Computer and Information Technology Workshops, 2008. CIT Workshops 2008. IEEE 8th International Conference on*, 2008, pp. 153-158.

[261] theMPEG-7group. (2000). *MPEG-7 CE Shape-1 Part-B dataset*. Available: http://www.imageprocessingplace.com/root_files_V3/image_databases.htm, Access date: 03/11/2013

[262] W. Gropp, R. Thakur, and E. Lusk, *Using MPI-2 : advanced features of the message-passing interface*: Cambridge, Mass. ; London : MIT Press 1999.

[263] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD explorations newsletter,* vol. 11, pp. 10-18, 2009.

[264] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*: Morgan Kaufmann, 2005.

[265] S. Ougiaroglou, A. Nanopoulos, A. N. Papadopoulos, Y. Manolopoulos, and T. Welzer-Druzovec, "Adaptive k-nearest-neighbor classification using a dynamic number of nearest neighbors," in *Advances in Databases and Information Systems*, 2007, pp. 66-82.

[266] M. Peker, H. Altun, and F. Karakaya, "Hardware emulation of HOG and AMDF based scale and rotation invariant robust shape detection," in *Engineering and Technology (ICET), 2012 International Conference on*, 2012, pp. 1-5.

[267] V. Ferrari, F. Jurie, and C. Schmid, "From Images to Shape Models for Object Detection," *International Journal of Computer Vision,* vol. 87, pp. 284-303, 2010/05/01 2010.

[268] D. Zhang and G. Lu, "Review of shape representation and description techniques," *Pattern recognition,* vol. 37, pp. 1-19, 2004.

[269] A. Toshev, B. Taskar, and K. Daniilidis, "Shape-based object detection via boundary structure segmentation," *International journal of computer vision,* vol. 99, pp. 123-146, 2012.

[270] D. D. Martin. (2011). *ESO/Digitized Sky Survey 2 (7/12/2011 ed.)*. Available: http://www.eso.org/, Access Date: 20/06/2013

[271] M. Baqer, "Energy Efficient Event Recognition for Wireless Sensor Networks," Monash University, 2008.

[272] A. Labrador and P. M. Wightman, *Topology Control in Wireless Sensor Networks: with a companion simulation tool for teaching and research*: Springer, 2009.

[273] S. Ping, "Delay measurement time synchronization for wireless sensor networks," *IRB-TR-03-013, Intel Research Berkeley Lab*, 2003.

[274] A. S. Wander, N. Gura, H. Eberle, V. Gupta, and S. C. Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," in *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on*, 2005, pp. 324-328.

[275] G. Guimaraes, E. Souto, D. Sadok, and J. Kelner, "Evaluation of security mechanisms in wireless sensor networks," in *Systems Communications, 2005. Proceedings*, 2005, pp. 428-433.

[276] A. Asuncion and D. J. Newman, "UCI machine learning repository," ed: http://archive.ics.uci.edu/ml, Irvine, CA: University of California, School of Information and Computer Science, 2007, Access date: 10/09/2014.

[277] S. Tactile, "Semeion Handwritten Digit Data Set," ed: Semeion Research Center of Sciences of Communication [http://www.semeion.it], Rome, Italy, 1994, Access date: 10/09/2014.

[278] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, and P. Havinga, "Activity recognition using inertial sensing for healthcare,

wellbeing and sports applications: A survey," in *Architecture of computing systems (ARCS), 2010 23rd international conference on,* 2010, pp. 1-10.

[279] C. J. Caspersen, K. E. Powell, and G. M. Christenson, "Physical activity, exercise, and physical fitness: definitions and distinctions for health-related research," *Public health reports,* vol. 100, p. 126, 1985.

[280] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* vol. 42, pp. 790-808, 2012.

[281] L. Chen, C. D. Nugent, and H. Wang, "A knowledge-driven approach to activity recognition in smart homes," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 24, pp. 961-974, 2012.

[282] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding,* vol. 104, pp. 90-126, 2006.

[283] J. Hoey and J. J. Little, "Value-directed human behavior analysis from video using partially observable markov decision processes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 29, pp. 1118-1132, 2007.

[284] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *Circuits and Systems for*

Video Technology, IEEE Transactions on, vol. 18, pp. 1473-1488, 2008.

[285] L. Fiore, D. Fehr, R. Bodor, A. Drenner, G. Somasundaram, and N. Papanikolopoulos, "Multi-camera human activity monitoring," *Journal of Intelligent and Robotic Systems,* vol. 52, pp. 5-43, 2008.

[286] S.-W. Lee and K. Mase, "Activity and location recognition using wearable sensors," *IEEE pervasive computing,* vol. 1, pp. 24-32, 2002.

[287] J. Parkka, M. Ermes, P. Korpipaa, J. Mantyjarvi, J. Peltola, and I. Korhonen, "Activity classification using realistic data from wearable sensors," *Information Technology in Biomedicine, IEEE Transactions on,* vol. 10, pp. 119-128, 2006.

[288] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, *et al.*, "Inferring activities from interactions with objects," *Pervasive Computing, IEEE,* vol. 3, pp. 50-57, 2004.

[289] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on,* 2004, pp. 32-36.

[290] C. Zhu and W. Sheng, "Wearable sensor-based hand gesture and daily activity recognition for robot-assisted living," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on,* vol. 41, pp. 569-573, 2011.

[291] J. Parkka, L. Cluitmans, and M. Ermes, "Personalization algorithm for real-time activity recognition using PDA, wireless motion bands, and

binary decision tree," *Information Technology in Biomedicine, IEEE Transactions on,* vol. 14, pp. 1211-1215, 2010.

[292] M. Zhang and A. A. Sawchuk, "Human daily activity recognition with sparse representation using wearable sensors," *Biomedical and Health Informatics, IEEE Journal of,* vol. 17, pp. 553-560, 2013.

[293] L. I. Kuncheva, *Combining pattern classifiers: methods and algorithms*: John Wiley & Sons, 2004.

[294] H. Cao, V. Leung, C. Chow, and H. Chan, "Enabling technologies for wireless body area networks: A survey and outlook," *Communications Magazine, IEEE,* vol. 47, pp. 84-93, 2009.

[295] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Forster, G. Troster, *et al.*, "Collecting complex activity datasets in highly rich networked sensor environments," in *Networked Sensing Systems (INSS), 2010 Seventh International Conference on*, 2010, pp. 233-240.

[296] *Opportunistic activity recognition systems*. Available: http://www.opportunity-project.eu, 2011, Access date:20/05/2014

[297] D. Moshou, K. Deprez, and H. Ramon, "Prediction of spreading processes using a supervised Self-Organizing Map," *Mathematics and Computers in Simulation,* vol. 65, pp. 77-85, 2004.

Appendix A

Extended Computational Complexity Analysis

A.1 Preamble

This section presents an extended analysis of the computational complexity of the MOHA algorithm. In Chapter 3, the computational complexity of the MOHA algorithm was compared with the Hopfield network. The analysis results show that the MOHA implementation incurs less computational complexity during the pattern recognition process compared with that of the Hopfield network. In this section, the computational complexity of the MOHA algorithm will be compared with the Self-Organising Map (SOM) and Distributed Hierarchical Graph Neuron (DHGN). It is best to note, however, that the comparative study that has been carried out does not intend to outweigh the capabilities of these algorithms. Rather, it indicates that the operations associated with the MOHA have a significantly low computational complexity.

A.1.1 MOHA vs. SOM

The Big-O notations for both the SOM and MOHA have been estimated in order to study their complexity levels. The supervised SOM (presented at Chapter 2) consists of three essential stages [297]: (i) weight initialisation, (ii) best-matching unit (BMU) calculation, and (iii) weight adjustment. In the weight initialisation stage, nodes are created with randomly assigned weights. At this stage, the computational complexity depends heavily on the number of created nodes. Hence, for a given weight initialisation process w, the complexity of n nodes can be simplified as $f(w) = O(n^3)$. Figure A.1 shows the estimated time taken to initialise up to 100,000 nodes. The estimated time derived is based on the assumption that the instruction speed used is 1 microsecond (µs) per instruction.

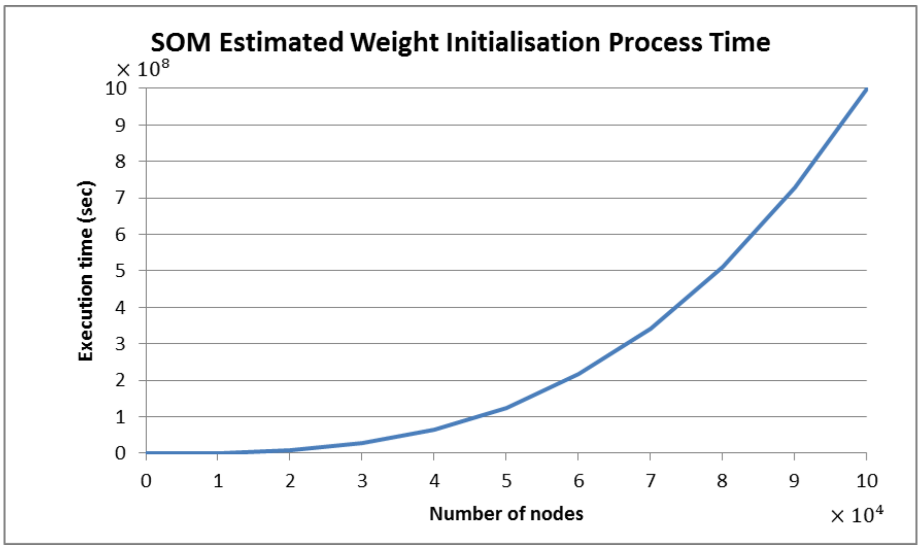

Figure A.1: Complexity measurement of SOM's weight initialisation process.

In the BMU calculation stage, the complexity depends heavily on the number of iterations during training as well as the number of input vectors. Therefore, for a given BMU calculation process m, the complexity of n training iterations can be simplified as $f(m) = O(n^4)$. The estimated time taken to perform up to 100,000 iterations of calculating the Euclidean distance between the input values and all neurons (i.e. nodes) in this stage is given in Figure A.2.

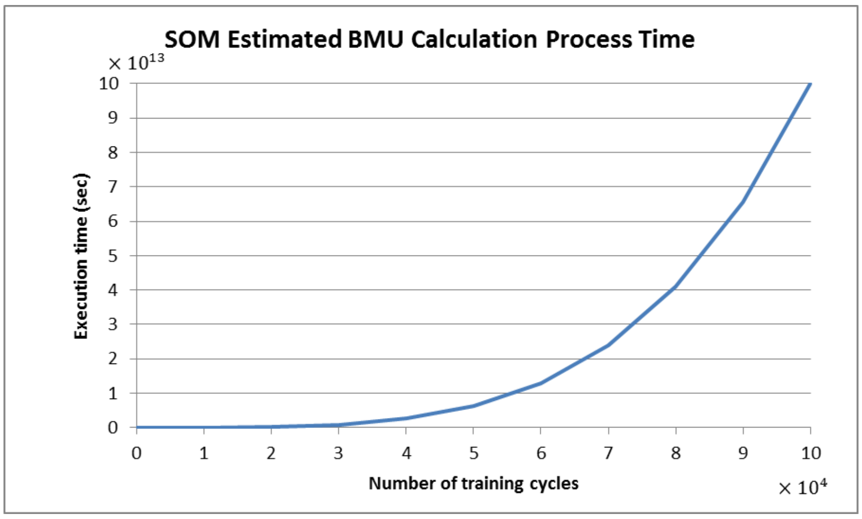

Figure A.2: Complexity measurement of SOM's BMU calculation process.

In the last stage, the weight adjustments are provided not only for the winning neuron (node) but also for its neighbours in a certain neighbourhood. The degree of adjustment depends on the degree of similarity between the neuron and the input. As a result of weight adjustment, a group of neurons is obtained to form a cluster. The Big-O for the weight adjustment is similar to the BMU calculation (refer to Figure A.2) and is therefore not provided.

The initialisation stage of the MOHA, as discussed in Chapter 3, is a low-computational process, and therefore requires less computational time compared with the SOM's weight initialisation process. Figure A.3 shows the estimated time for this process. A similar speed assumption of 1 microsecond (μs) per instruction is applied in this analysis. It can be seen that the time taken in the MOHA initialisation process is far less than for the SOM. For instance, the MOHA takes only 0.02 seconds while the SOM takes about 0.8×10^7 seconds (see Figure A.1) to initialise 20,000 nodes.

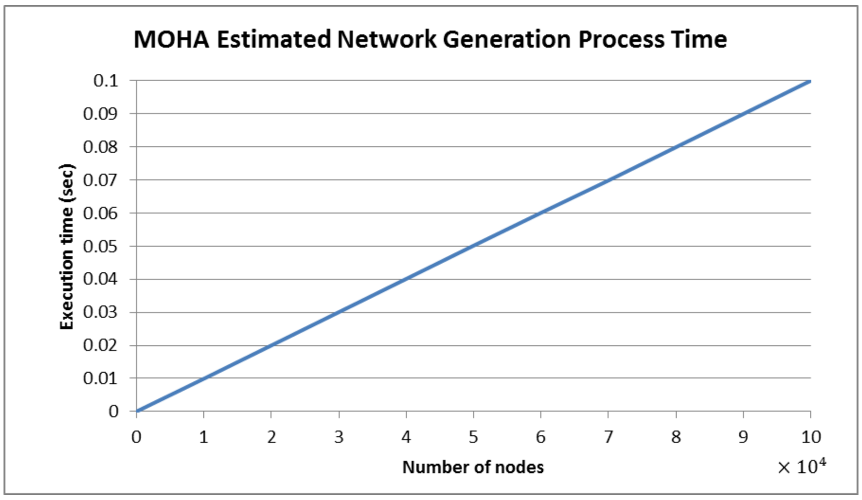

Figure A.3: Complexity measurement of MOHA's network generation process.

In the MOHA algorithm, only activated nodes and activated edge nodes are utilised in the classification process and the total number of active nodes N_{active} and edge nodes N_{ED} is usually less than the total number of generated nodes (network size) G_{MOHA}, $N_{active}, N_{ED} < G_{MOHA}$. Figure A.4 shows the time taken for classification by the MOHA algorithm based on the assumption that 100% of the network nodes are activated as edge nodes. The figure shows that the MOHA's classification process requires less computational complexity compared with the SOM's BMU calculation and weight adjustment activities. For example, the time taken for classification by the MOHA in a network of 50,000 nodes, if 100% of the nodes are activated as edge nodes, is less than 1.2×10^3 seconds as shown in Figure A.4, while the SOM's BMU calculation process alone takes about 6×10^{12} seconds to complete (see Figure A.2), and with a similar amount of time needed to perform weight adjustment.

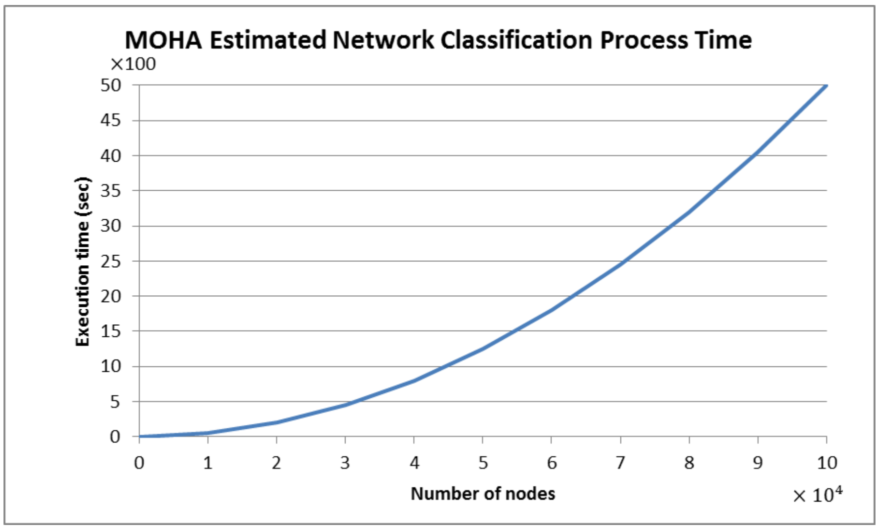

Figure A.4: Complexity measurement of MOHA's network classification process.

This proves that the MOHA provides an efficient, lightweight, and fast algorithm, comparable to SOM implementation.

A.1.2 MOHA vs. DHGN

In order to analyse and compare the MOHA and DHGN algorithms, the notations for network generation and recognition stages within the implementation have been derived.

A.1.2.1 Network generation stage

In the DHGN implementation, the number of neurons (nodes) generated, G_{DHGN} with number of different elements within the pattern v, a pattern size (S_p), and a number of DHGN subnets (N_{DHGN}), G_{DHGN} is given as follows [260]:

$$G_{DHGN} = v(\frac{\frac{S_p}{N_{DHGN}} + 1}{2})^2 \times N_{DHGN} \qquad (A.1)$$

On the other hand, as shown in Equation 3.37, the number of nodes generated, G_{MOHA} in MOHA implementation is: $G_{MOHA} = S_p$. Table A.1 shows the details of the Big-O notation derived for the MOHA and DHGN implementations.

Table A.1: Big-O notations for MOHA and DHGN implementation in network generation stage.

Algorithm	**Big-O**	**Efficiency**	**Iteration (*n*)**	**Estimated Time (in seconds)**
MOHA	$O(n)$	Linear	G_{MOHA}	$G_{MOHA} \times 0.000001$
DHGN	$O(n)$	Linear	G_{DHGN}	$G_{DHGN} \times 0.000001$

The results show that both the MOHA and DHGN have comparable computational complexity. However, in regards to the number of nodes generated at this stage, the DHGN incurs higher complexity since $G_{DHGN} > G_{MOHA}$.

A.1.2.2 Recognition stage

Similar to the MOHA scheme, the DHGN algorithm involves a single-cycle process in which each input pattern will be passed through the DHGN subnets once and the store or recall process will be activated according to the instruction given. Table A.2 shows the Big-O notation for the recognition stage using the DHGN algorithm. The Big-O notations derived from the analysis of the MOHA recognition process are presented in Table 3.6.

Table A.2: Big-O notations for DHGN implementation in recognition stage.

Algorithm	**Big-O**	**Efficiency**	**Iteration (*n*)**	**Estimated Time (in seconds)**
DHGN	$O(n)$	Linear	G_{DHGN}	$G_{DHGN} \times 0.000001$

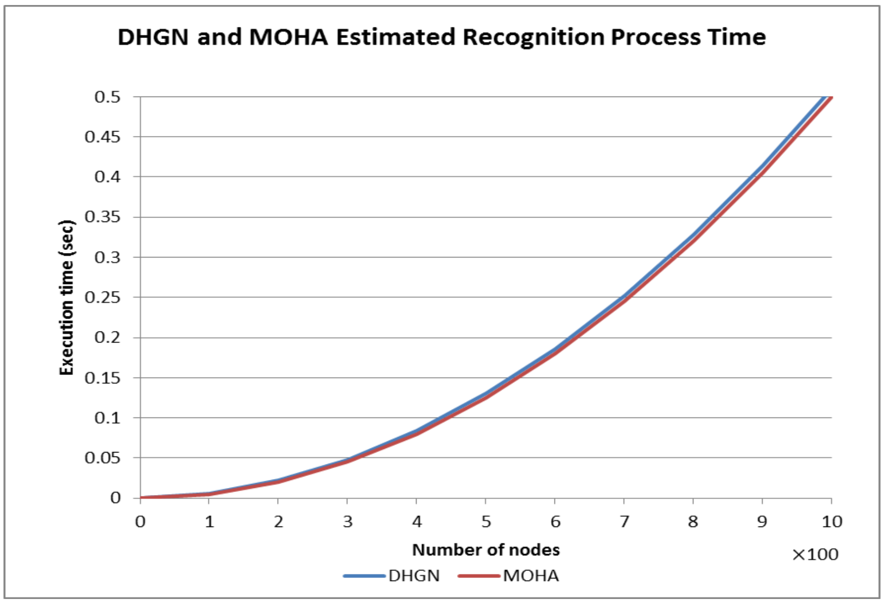

Figure A.5: Complexity measurement of DHGN's and MOHA's recognition process.

Table 3.6, Table A.2, and Figure A.5 show the comparison of computational complexity for the DHGN and MOHA. The MOHA's computational complexity is based on the assumption that 100% of the network nodes are activated as edge nodes and it has 10 MOHA rows. The DHGN's computational complexity is based on the assumption that it has 10 DHGN subnets. It is clearly shown from the analysis that the MOHA incurs slightly less computational complexity in the pattern recognition process compared with the DHGN implementation.